An Integrated Development Environment for Gesture-based Interactive Applications

Instantiation to Radar Gestures

Arthur Sluÿters *UCLouvain, Belgium*

A thesis submitted in fulfillment of the requirements for the degree of

Doctor in Engineering and Technology

of the

Université catholique de Louvain

Committee in charge

Prof. Sébastien Lambot (Co-advisor)	*UCLouvain, Belgium*
Prof. Jean Vanderdonckt (Co-advisor)	*UCLouvain, Belgium*
Prof. Suzanne Kieffer (Secretary)	*UCLouvain, Belgium*
Prof. Peter Van Roy (President)	*UCLouvain, Belgium*
Prof. Paolo Roselli	*Università di Roma "Tor Vergata", Italy*
Prof. Luis A. Leiva	*University of Luxembourg, Luxembourg*



Abstract

Radar sensors combine a small form factor and low power consumption with the ability to sense gestures through opaque surfaces and under unfavorable lighting conditions, all while preserving user privacy. These unique advantages could make them a credible and useful alternative to other types of sensors, such as cameras or inertial sensors, for gesture recognition.

However, research on radar-based gestural interfaces is in its infancy and many challenges that hinder their seamless integration and widespread adoption still must be solved. In particular, the lack of standardization translates into a plethora of custom radar sensors and techniques, preventing efficient collaboration between developers. In addition, radar signals can be noisy and complex to analyze, and do not transpose well from one radar to another.

This thesis investigates and advances the state of radar-based gesture interaction in two stages. First, it explores the use of radar sensors for gesture recognition by reporting results from a targeted literature review and two systematic literature reviews, unveiling a large variety of radar sensors, gesture sets, and gesture recognition techniques, as well as the many challenges of real-time gesture recognition. Second, it provides tools and methods that facilitate the development of highly usable radar-based gesture interfaces by bridging the gap between researchers and practitioners. In particular, it introduces QuantumLeap, a modular framework for gesture recognition that acts as an intermediate layer between (radar) sensors and gesture-based applications and facilitates the performance and efficiency evaluation of gesture recognizers. Additionally, it proposes a user-centered development method for gesture-based applications and applies it to a multimedia application. Finally, it introduces and evaluates a new gesture recognition pipeline that implements advanced full-wave electromagnetic modeling and inversion to retrieve physical characteristics of gestures that are independent of the source, antennas, and radar-hand interactions.

Keywords: gesture recognition, radar, development environment.

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude and appreciation to my co-supervisors, Prof. Jean Vanderdonckt and Prof. Sébastien Lambot, who guided me throughout this thesis. Their patience, insightful feedback, and profound kindness meant a lot to me. I could not have asked for better supervisors!

I would also like to thank Prof. Suzanne Kieffer, Prof. Peter Van Roy, Prof. Paolo Roselli, and Prof. Luis A. Leiva for being part of my defense committee and sharing their expertise and knowledge to help me improve this thesis.

I extend my gratitude to Prof. Radu Vatavu and his team at the Ștefan cel Mare University of Suceava who I was lucky to visit in beautiful Romania. Thank you also to Vik Parthiban and Pattie Maes from MIT Media Lab, who provided the functional core of the LUI application, for their help and advice in this project. I am also thankful to my colleagues, in particular Quentin Sellier and Mehdi Ousmer, whom I had the pleasure of working with during these four years.

I am grateful to Fonds National de la Recherche Scientifique - FNRS for providing me with funding under Grants nos. 40001931 and 40011629. This whole endeavor would not have been possible without their funding.

Finally, I would like to thank my friends and family, who supported me all the way and kept me motivated even in the most difficult times. You all helped me grow into the person I am today and I feel very lucky to have you in my life. Du plus profond de mon cœur, merci.

Publications

- Mehdi Ousmer, Arthur Sluÿters, Nathan Magrofuoco, Paolo Roselli, and Jean Vanderdonckt. "Recognizing 3D Trajectories as 2D Multi-stroke Gestures". In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 4, Issue ISS, pages 1-21, November 2020. DOI: 10.1145/3427326.

- Arthur Sluÿters, Jean Vanderdonckt, and Radu-Daniel Vatavu. "Engineering Slidable Graphical User Interfaces with Slime". In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 5, Issue EICS, Article 200, pages 1-29, May 2021. DOI: 10.1145/3457147.

- Silvia Abrahão, Emilio Insfran, Arthur Sluÿters, and Jean Vanderdonckt. "Model-based intelligent user interface adaptation: challenges and future directions". In *Software and Systems Modeling*, Vol. 20, Issue 5, pages 1335-1349, July 2021. DOI: 10.1007/s10270-021-00909-7.

- Arthur Sluÿters, Sébastien Lambot, and Jean Vanderdonckt. "Hand Gesture Recognition for an Off-the-Shelf Radar by Electromagnetic Modeling and Inversion". In *Proceedings of the 27th International Conference on Intelligent User Interfaces (IUI '22)*, pages 506–522, March 2022. DOI: 10.1145/3490099.3511107. (**Best Paper Award**).

- Arthur Sluÿters, Mehdi Ousmer, Paolo Roselli, and Jean Vanderdonckt. "QuantumLeap, a Framework for Engineering Gestural User Interfaces based on the Leap Motion Controller". In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 6, Issue EICS, Article 161, pages 1-47, June 2022. DOI: 10.1145/3532211.

- Vik Parthiban, Pattie Maes, Quentin Sellier, Arthur Sluÿters, and Jean Vanderdonckt. "Gestural-Vocal Coordinated Interaction on Large Displays". In *Companion of the 2022 ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS 2022)*, pages 26-32, June 2022. DOI: 10.1145/3531706.3536457.

- Arthur Sluÿters. "Mid-air Gesture Recognition by Ultra-Wide Band Radar Echoes". In *Proceedings of the Workshops on Engineering Interactive Computing Systems (EICS-WS 2022)*, Vol. 3404, pages 28-39, June 2022. URL: https://ceur-ws.org/Vol-3404/paper4.pdf.

- Xavier de Ryckel, Arthur Sluÿters, and Jean Vanderdonckt. "SnappView, a Software Development Kit for Supporting End-user Mobile Interface Review". In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 6, Issue EICS, Article 173, pages 1-38, June 2022. DOI: 10.1145/3534527.

- Santiago Villarreal-Narvaez, Alexandru-Ionuţ Şiean, Arthur Sluÿters, Radu-Daniel Vatavu, and Jean Vanderdonckt. "Informing Future Gesture Elicitation Studies for Interactive Applications that Use Radar Sensing". In *Proceedings of the 2022 International Conference on Advanced Visual Interfaces (AVI 2022)*, Article 50, pages 1-3, June 2022. DOI: 10.1145/3531073.3534475.

- Arthur Sluÿters, Quentin Sellier, Jean Vanderdonckt, Vik Parthiban and Pattie Maes. "Consistent, Continuous, and Customizable Mid-Air Gesture Interaction for Browsing Multimedia Objects on Large Displays". In *International Journal of Human-Computer Interaction*, Vol. 39, Issue 12, pages 2492-2523, July 2022. DOI: 10.1080/10447318.2022.2078464.

- Mehdi Ousmer, Arthur Sluÿters, Nathan Magrofuoco, Paolo Roselli, and Jean Vanderdonckt. "A Systematic Procedure for Comparing Template-Based Gesture Recognizers". In *HCI International 2022 - Late Breaking Papers. Multimodality in Advanced Interaction Environments*, Vol. 13519, pages 160-179, October 2022. DOI: 10.1007/978-3-031-17618-0_13.

- Santiago Villarreal-Narvaez, Arthur Sluÿters, Jean Vanderdonckt, and Efrem Mbaki Luzayisu. "Theoretically-Defined vs. User-Defined Squeeze Gestures". In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 6, Issue ISS, pages 73-102, November 2022. DOI: 10.1145/3567805.

- Arthur Sluÿters. "Designing Efficient and Customizable Radar-based Gesture Interfaces". In *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces (IUI '23)*, pages 208-210, March 2023. DOI: 10.1145/3581754.3584105.

- Alexandru-Ionuţ Şiean, Cristian Pamparău, Arthur Sluÿters, Radu-Daniel Vatavu, Jean Vanderdonckt. "Flexible Gesture Input with Radars: Systematic Literature Review and Taxonomy of Radar Sensing Integration in Ambient Intelligence Environments". In *Journal of Ambient Intelligence and Humanized Computing*, Vol. 14, Issue 5, pages 7967–7981, April 2023. DOI: 10.1007/s12652-023-04606-9.

- Arthur Sluÿters, Sébastien Lambot, Jean Vanderdonckt, and Radu-Daniel Vatavu. "RadarSense: Accurate Recognition of Mid-Air Hand Gestures with Radar Sensing and Few Training Examples". In *ACM Transactions*

on Interactive Intelligent Systems, Vol. 13, Issue 3, pages 1-45, September 2023. DOI: 10.1145/3589645.

- Stefano Chioccarello, Arthur Sluÿters, Alberto Testolin, Jean Vanderdonckt, and Sébastien Lambot. "FORTE: Few Samples for Recognizing Hand Gestures with a Smartphone-attached Radar". In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 7, Issue EICS, Article 179, pages 1-25, June 2023. DOI: 10.1145/3593231.

- Paul Brie, Nicolas Burny, Arthur Sluÿters, and Jean Vanderdonckt. "Evaluating a Large Language Model on Searching for GUI Layouts". In *Proceedings of the ACM on Human-Computer Interaction*, Vol. 7, Issue EICS, Article 178, pages 1-37, June 2023. DOI: 10.1145/3593230.

- Santiago Villarreal-Narvaez, Arthur Sluÿters, Jean Vanderdonckt, and Radu-Daniel Vatavu. "Brave New GES World: A Systematic Literature Review of Gestures and Referents in Gesture Elicitation Studies". In *ACM Computing Surveys*, Vol. 56, Issue 5, Article 128, pages 1–55, January 2024. DOI: 10.1145/3636458.

- Quentin Sellier and Arthur Sluÿters and Jean Vanderdonckt and Ingrid Poncin. "Evaluating gesture user interfaces: Quantitative measures, qualitative scales, and method". In *International Journal of Human-Computer Studies*, pages 1-24, February 2024. DOI: 10.1016/j.ijhcs.2024.103242.

- Arthur Sluÿters, Sébastien Lambot, Jean Vanderdonckt, and Santiago Villarreal-Narvaez. "Analysis of User-defined Radar-based Hand Gestures Sensed through Multiple Materials". In *IEEE Access*, Vol. 12, February 2024. DOI: 10.1109/ACCESS.2024.3366667

- Sébastien Lambot, Kaijun Wu, Arthur Sluÿters, and Jean Vanderdonckt. "The full-wave radar equation for wave propagation in multilayered media and its applications". In *Ground Penetrating Radar: From Theoretical Endeavors to Computational Electromagnetics Signal, Processing, Antenna Design and Field Applications*, pages 123-160, In press.

Contents

1 Introduction **1**
- 1.1 Context . . . 1
- 1.2 Research . . . 3
  - 1.2.1 Statement and Research Questions . . . 3
  - 1.2.2 Scope . . . 4
  - 1.2.3 Contributions . . . 5
- 1.3 Overview of the Thesis . . . 7

2 State of the Art **11**
- 2.1 Overview of Gesture-based Interaction . . . 13
  - 2.1.1 Gesture Elicitation . . . 13
  - 2.1.2 Challenges . . . 14
  - 2.1.3 Applications . . . 22
  - 2.1.4 Summary . . . 24
- 2.2 Analysis of a Mature Sensing Technology . . . 24
  - 2.2.1 Algorithms . . . 25
  - 2.2.2 Summary . . . 27
- 2.3 A Deep Dive into Radar-based Gesture Interaction . . . 27
  - 2.3.1 Applications . . . 27
  - 2.3.2 Algorithms . . . 29
  - 2.3.3 Gestures . . . 31
  - 2.3.4 Radars . . . 32
  - 2.3.5 Summary . . . 37
- 2.4 Positioning this Work in the Literature . . . 37

3 Facilitating the Development of Gesture-based Applications **39**
- 3.1 A Framework for Building Gesture-Based Applications . . . 41
  - 3.1.1 Dataflow Architecture . . . 41
  - 3.1.2 Configuration . . . 47
  - 3.1.3 Application Programming Interface . . . 48
- 3.2 Evaluation . . . 49
  - 3.2.1 Experiment Protocol . . . 49
  - 3.2.2 Results and Discussion . . . 52
  - 3.2.3 Changes . . . 57

3.3 Sensor Integration and Applications . . . 58
3.4 Discussion . . . 59
3.4.1 Strengths . . . 59
3.4.2 Limitations . . . 60
3.5 Conclusion . . . 61

4 Streamlining the Evaluation of Gesture Recognizers 63
4.1 A Tool for Evaluating Gesture Recognition Algorithms . . . 65
4.1.1 Integration into QuantumLeap . . . 65
4.1.2 Testing procedures . . . 67
4.2 Discussion . . . 68
4.2.1 Strengths . . . 68
4.2.2 Limitations . . . 70
4.3 Conclusion . . . 70

5 Designing Highly Usable Gesture-based Applications 73
5.1 The LUI Application . . . 75
5.1.1 Overview . . . 75
5.1.2 Requirements . . . 75
5.1.3 Iterative Design . . . 78
5.2 User-centered Development Method . . . 79
5.2.1 Gesture Set Design . . . 79
5.2.2 Software Architecture and Implementation . . . 84
5.3 Evaluation . . . 88
5.3.1 Experiment Protocol . . . 88
5.3.2 Results and Discussion . . . 93
5.4 Discussion . . . 95
5.4.1 Strengths . . . 95
5.4.2 Limitations . . . 97
5.5 Conclusion . . . 98

6 Addressing the Challenges of Radar-based Gesture Recognition 101
6.1 Mathematical Theory . . . 103
6.1.1 Full-wave Radar Model . . . 103
6.1.2 Determination of the Radar-antenna Characteristic Functions . . . 104
6.1.3 Full-Wave Signal Inversion . . . 106
6.2 Radar Data Processing Strategy . . . 107
6.2.1 Raw Data Capture . . . 107
6.2.2 Radar Source and Antenna Effects Removal . . . 108
6.2.3 Background Scene Removal . . . 109
6.2.4 Time Gating . . . 109
6.2.5 Full-wave Inversion . . . 111

6.2.6 Filtering . . . 112
6.3 Implementation . . . 113
6.4 Limitations . . . 115
6.5 Conclusion . . . 115

7 Experimenting with Radar-based Gestures 117
7.1 Data Collection . . . 119
7.1.1 Sensors . . . 119
7.1.2 Datasets . . . 121
7.1.3 Data Pre-processing . . . 128
7.2 Experiment 1: Sensors and Antenna Subsets . . . 130
7.2.1 Aims . . . 130
7.2.2 Protocol . . . 130
7.2.3 Results . . . 133
7.2.4 Summary . . . 142
7.3 Experiment 2: Gestures Subsets . . . 143
7.3.1 Aims . . . 143
7.3.2 Protocol . . . 143
7.3.3 Results . . . 145
7.3.4 Summary . . . 147
7.4 Experiment 3: Gestures through Materials . . . 150
7.4.1 Aims . . . 150
7.4.2 Protocol . . . 150
7.4.3 Results . . . 151
7.4.4 Summary . . . 157
7.5 Discussion and Implications for Design . . . 158
7.6 Conclusion . . . 162

8 Conclusion 165
8.1 Summary . . . 165
8.2 Future Works . . . 169

A QuantumLeap Modules 227
A.1 Sensors . . . 227
A.1.1 Leap Motion Controller . . . 227
A.2 Filters . . . 227
A.2.1 1€ . . . 227
A.3 Static and Dynamic Datasets . . . 227
A.3.1 QLDynamic . . . 227
A.3.2 QLStatic . . . 228
A.4 Static Recognizers . . . 228
A.4.1 $P^3+ . . . 228
A.4.2 GPSDa . . . 228

A.5 Analyzers . . . 228
A.5.1 Basic Analyzer . . . 228
A.6 Segmenters . . . 229
A.6.1 Zoning Segmenter . . . 229
A.6.2 Threshold Segmenter . . . 229
A.6.3 Window Segmenter . . . 229
A.7 Dynamic Recognizers . . . 229
A.7.1 P^3 . . . 229
A.7.2 Q^3 . . . 230
A.7.3 P^3+ . . . 231
A.7.4 P^3+X . . . 231
A.7.5 3 cent . . . 231
A.7.6 Jackknife . . . 231
A.7.7 $F . . . 231
A.7.8 FreeHandUni . . . 232
A.7.9 Rubine3D . . . 232
A.7.10 Rubine-Sheng . . . 232

B Tasks **235**
B.1 Configure QuantumLeap . . . 235
B.1.1 Initial State . . . 235
B.1.2 Steps . . . 235
B.2 Add Gesture Recognition to a Small Application . . . 236
B.2.1 Steps . . . 236

C LUI Gesture Elicitation Studies **239**
C.1 Photo- and Video-Browsing Gesture Elicitation Studies . . . 239
C.2 Multimedia Gesture Elicitation Study . . . 239
C.2.1 Experimental Protocol . . . 239
C.2.2 Results . . . 241

D Experimenting with Radar-based Gestures (Additional Results) **249**
D.1 Experiment 2: Gestures Subsets . . . 249
D.2 Experiment 3: Gestures through Materials . . . 252

Chapter 1

Introduction

1.1 Context

Since the advent of computers, significant effort has been invested in improving how they interface with humans, fostered by continuous technological improvements through the 20th and 21st centuries. In the mid-20th century, computers were still large, costly, and highly specialized tools mostly reserved for big institutions and companies. The emergence of high-level programming languages vastly simplified the work of researchers at these institutions by enabling them to create complex algorithms without having to manipulate individual bits. From the 1970s onward, the user base of computers gradually expanded to cover a broader segment of society. As they became accessible to more people, more intuitive methods of interaction were required, leading to the emergence of graphical user interfaces (GUIs). Thanks to GUIs, many tasks became accessible to less tech-savvy users, such as writing text and manipulating images. The early 21st century saw an explosion in the use of computers and small portable electronic devices thanks in part to massive improvements in efficiency, performance, size, and cost. This rapid digitalization of our world could leave vulnerable people, such as the elderly, at risk of being shut out from the rest of society. Now more than ever, it is essential to design computer systems with accessibility in mind so that no one is left behind: technology can, and should, be at the service of everyone, regardless of their age, ethnicity, and whether they suffer from a disability. Only that way could society truly benefit from the new opportunities offered by the digitization of many of its facets, including healthcare, education, culture, and governance.

Fortunately, the increase in computing power and improvements in sensing devices opened up a wealth of new possibilities for human-computer interaction (HCI). In particular, mid-air gesture interaction, *i.e.*, relying on body postures and movements to control computer interfaces [136], was proposed as a natural and intuitive alternative to traditional input methods [302, 112]. Mid-air gestures have applications in various domains, such as aiding patient

rehabilitation in healthcare [202], facilitating the teaching of complex subjects, assisting people with disabilities [57, 202], and offering new ways for artists to express their creativity [57, 139]. A myriad of sensors are currently available to capture user motion, each with distinct advantages and limitations in terms of cost, performance, privacy, and practicality. The characteristics of a sensor impact its suitability for different applications. For instance, vision-based sensors like the UltraLeap Leap Motion Controller (LMC) and Microsoft Kinect combine cameras with computer vision to accurately track key points of the user's body (*e.g.*, full body skeleton, hand joints) from a distance [324, 61] but often feature a limited field of view, potential privacy concerns, and a sensitivity to occlusion and lighting conditions. Inertial Measurement Units (IMUs) combine gyroscopes and accelerometers to measure data about the object to which they are attached, such as its acceleration and angular rate. They are often embedded in wearable devices like smart rings [88], smartwatches, and armbands. IMU data can serve to compute positioning information and identify user gestures. Despite being largely immune to the challenges faced by vision-based sensors, IMUs are susceptible to drift (*i.e.*, an increase in the difference between the system-estimated position and the actual position of the sensor over time), and are more intrusive as they must be worn by the user during the interaction, making them less suitable for public spaces.

Radar sensors [338] have emerged as a promising alternative to these technologies for gesture interaction. By emitting electromagnetic waves towards the user and capturing their reflections, they gather valuable information (*e.g.*, amplitude and frequency of the reflected signal) for identifying user gestures. Radars combine a compact form factor and low power consumption with the ability to capture gestures through opaque surfaces and in challenging lighting conditions, all while preserving user privacy. These characteristics could make them well-suited for gestural interaction in public spaces. Radars were successfully applied in various domains, including virtual reality [113], activity recognition [19, 20], material recognition [83, 338], and tangible interaction [339]. However, research on radar-based gestural interfaces is still in its early stages and presents several challenges. In particular, the lack of consensus on a few affordable and efficient off-the-shelf radar sensors has resulted in plenty of (custom) radar systems, often incompatible with each other, which is especially problematic as existing works typically rely on sophisticated deep-learning algorithms for gesture recognition to cope with the complexity of radar signals. These algorithms are usually tailored to a specific sensor and gesture set, thus complicating their (re)use in actual applications.

In light of this context, this thesis aims to advance the state of radar-based gesture interaction by providing tools and methods that bridge the gap between researchers and practitioners to facilitate the development of highly usable radar-based gesture interfaces.

1.2 Research

1.2.1 Statement and Research Questions

Radars have many advantages over other devices, such as vision-based sensors. Provided that developers and practitioners can overcome their limitations, radar sensors may become one of the keys to building ubiquitous gesture interaction, such as envisioned by Şiean *et al.* [257]. However, research on radar-based gestural interfaces is in its infancy and many challenges that hinder their seamless integration and widespread adoption still have to be solved.

The central objective of this thesis is thus twofold: (1) explore the use of radar for gesture recognition and (2) provide tools and methods to facilitate the development of highly usable (radar-based) gesture interfaces. We further break down this objective into five research questions (RQs):

RQ1 *What are the main challenges of mid-air gesture recognition?* Developers and practitioners may face multiple challenges in the development of gesture-based applications. It thus is crucial to identify these challenges as a first step towards fostering the development of gestural interaction.

RQ2 *What is the performance of radar sensors for mid-air gesture recognition?* Evaluating the efficacy and efficiency of radar-based gesture recognition will enable us to identify the scenarios in which radars could be a relevant alternative to other types of sensors. In addition, it would help us identify avenues for improving their performance, *e.g.*, by applying signal processing.

RQ3 *What types of gesture-based applications would benefit from radar sensors?* Despite their limitations, radars have many advantages over other sensors, such as their ability to sense gestures through opaque surfaces and in various lighting conditions, which could them more suitable for specific applications depending on the kind of performance that they offer. The answer to RQ2 will help us answer this research question as well.

RQ4 *How can we foster collaboration between researchers and practitioners working on (radar) gesture recognition?* Fostering collaboration, *e.g.*, by facilitating the sharing of data and algorithms, has many benefits for both the scientific community and practitioners. It reduces the necessary steps to repeat and replicate experiments, which are major pillars of the scientific method, encourages building upon previous research (*e.g.*, to improve an existing algorithm), and makes this research more readily available to use in real-world applications.

RQ5 *How can tools and methods aid in designing gesture-based applications that operate independently of gesture recognition logic?* Separating the gesture

recognition logic from the user interface of an application enables developers to focus on polishing the actual user experience and facilitates future modifications to the gesture recognition logic, *e.g.*, to support new sensors, change the gesture set, or improve the gesture recognition algorithm. Developing appropriate tools and methods is thus paramount to encourage the development of highly usable gesture-based interfaces.

1.2.2 Scope

We define the scope of this thesis as follows:

- *S1* *Use off-the-shelf sensors*: relying on off-the-shelf sensors ensures that our findings can be easily reproduced and replicated by other researchers.
- *S2* *Use existing gesture recognizers*: relying on existing gesture recognizers allows us to follow a more generic approach, with the goal of providing useful tools and methods that are compatible with many techniques for gesture recognition.
- *S3* *Focus on mid-air arm and hand gestures*: mid-air gestures promise natural interaction [326] with a very large degree of freedom, but can be challenging to recognize with radar systems due to the small surface area of the hand. Focusing on mid-air gestures is a good starting point for supporting more types of gestures, such as breathing patterns or even tongue gestures.
- *S4* *Focus on non-critical applications*: as radar-based gesture recognition is still in its infancy, we will mostly focus on non-critical applications, such as multimedia applications, which do not require infallible gesture recognition.
- *S5* *Support user customization*: our tools and methods should support user-defined gestures, as they can result in higher memorability and an overall more enjoyable experience compared to pre-defined gesture sets [203]. In addition, custom gestures can make applications more accessible for users with disabilities.

We will also strive to produce software that satisfies at least four of the ISO/IEC 25010 software quality properties [116], namely:

- *P1* *Compatibility*: defined as the "degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions while sharing the same hardware or software environment" [116].

P2 *Maintainability*: defined as the "degree of effectiveness and efficiency with which a product or system can be modified to improve it, correct it or adapt it to changes in environment, and in requirements" [116].

P3 *Portability*: defined as the "degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another" [116].

P4 *Usability*: defined as the "degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" [116].

1.2.3 Contributions

This thesis makes several contributions to the field of gesture recognition in the form of literature reviews, tools, methods, applications, datasets, and experiments.

Literature Reviews

A targeted literature review (TLR) [146] and two systematic reviews of the literature (SLR) [132] were conducted to gather insight into the state of research in the field of (radar-based) gesture interaction:

- *TLR on gestural interaction*: a summary of the literature on the topic of gesture-based interaction in general, including gesture elicitation studies, the challenges of gesture recognition, and examples of gesture-based interfaces.
- *SLR on LMC-based gesture interaction*: a look into how the LMC, a relatively mature vision-based sensor for gesture interaction, was used by researchers and practitioners to implement gesture recognition.
- *SLR on radar-based gesture recognition*: a deep dive into the state of radar-based gesture recognition in the literature that focuses on applications, radar systems, gestures, and algorithms.

Tools and Methods

We provide two tools and one method that aims at fostering the development of highly usable (radar-based) gesture interfaces:

- *QuantumLeap framework*: a tool that addresses most of the challenges usually faced when creating gestural interfaces and provides an easy way to evaluate and compare gesture recognition algorithms. Its modular architecture featuring standardized modules facilitates the sharing of algorithms, sensors, and datasets across researchers and practitioners.
- *User-centered development method*: an iterative development method and evaluation protocol for gestural interfaces that guide developers through the design process of highly usable gesture-based applications.
- *Radar data pre-processing pipeline*: an electromagnetic model and inversion-based approach that normalizes radar signal to become independent from the radar system and background scene.

Applications

We developed one gesture-based application in the context of this thesis, but more were built by other developers using QuantumLeap (see Section 3.3):

- *Large User Interface*: a gesture-based application that enables the manipulation of multimedia content with hand gestures and serves as a demonstration of the QuantumLeap framework and our user-centered development method.

Datasets and Experiments

Three experiments were conducted, each with its corresponding dataset, to evaluate the performance of our radar data pre-processing pipeline on a wide range of gestures performed by multiple users in different contexts. These datasets allow the testing and comparison of gesture recognition techniques under various conditions and will thus be made publicly available to foster research and development in the field of gesture recognition.

- *Sensors comparison*: a dataset featuring 16 gesture classes from one participant recorded with three different devices, including two radars and an LMC. The corresponding experiment compares the performance of the three sensors in various testing configurations.
- *20 gestures*: a dataset featuring 20 gesture classes from six participants recording with a Walabot sensor. The corresponding experiment evaluates the suitability of radar sensors for recognizing large sets of gestures and identifies smaller subsets of highly-differentiable gestures.
- *Gestures through materials*: a dataset featuring nine gesture classes from 20 participants recorded with a Walabot through three types of materials

(wood, PVC, and glass). The corresponding experiment investigates the performance of the Walabot when sensing gestures through materials and the ability of our pre-processing pipeline to normalize radar data sensed through different materials.

1.3 Overview of the Thesis

Fig. 1.1 provides a graphical overview of this thesis. Subsequent tasks are represented from top to bottom, along with their related chapter(s), contribution(s), and research question(s). Each chapter may cover one or more research question(s), contribution(s), and task(s).

Chapter 2: State of the Art

This chapter positions this work in the literature by exploring the state of the art. We first provide an overview of gesture-based interaction, including gesture elicitation studies, the technical challenges of building gestural interaction, and examples of applications. We then conduct a first SLR focusing on the Leap Motion Controller (LMC), a relatively mature vision-based sensing technology, to help contextualize the current state of radar-based gesture recognition. With this context in mind, we conduct a second SLR that covers radar-based gesture interaction in general, including applications, sensors, gestures, and algorithms for gesture recognition. We then conclude this chapter by relating the current state of the art to this work's context and objectives.

Chapter 3: Facilitating the Development of Gesture-based Applications

In this chapter, we introduce QuantumLeap, a software tool that attempts to address most of the challenges usually faced when creating gestural interfaces, to foster the development of new gesture-based applications. We first provide an overview of the tool's modular architecture, designed to enable the reuse of components across applications. We then conduct an experiment to evaluate the usability of QuantumLeap among seven software developers of various experience levels. Finally, we explore recent projects that utilized and/or extended the QuantumLeap framework and we discuss its advantages and limitations.

Chapter 4: Streamlining the Evaluation of Gesture Recognizers

In this chapter, we introduce the QuantumLeap testing tool, an extension to QuantumLeap that allows researchers and practitioners to evaluate the performance of gesture recognizers, *e.g.*, to compare them against the current state

of the art or to select an adequate recognizer for a specific application. The tool reuses the standardized components of QuantumLeap to facilitate sharing modules, testing procedures, and results. We explain how we integrated the tool into QuantumLeap, present its main features, including its supported testing procedures, configuration user interface, and results format, and discuss its advantages and limitations.

Chapter 5: Designing Highly Usable Gesture-based Applications

In this chapter, we propose a user-centered iterative development method and evaluation protocol for gesture-based interfaces, which aims to support the development of highly usable gestural interfaces. We detail how we applied the method and evaluation protocol to a real use-case scenario: the development of an application for manipulating multimedia objects on large displays with hand gestures.

Chapter 6: Addressing the Challenges of Radar-based Gesture Recognition

In this chapter, we introduce a full-wave electromagnetic model and inversion-based approach for radar signal processing that aims to overcome the challenges of radar sensors for gesture recognition so that developers and practitioners can take full advantage of their capabilities. We first lay the theoretical groundwork for the data processing pipeline and then provide information about each of its six stages and their effect on the radar signal.

Chapter 7: Experimenting with Radar-based Gestures

In this chapter, we investigate the efficacy of our new radar signal processing approach by conducting three different experiments: (1) comparing the performance of radar-based and vision-based sensors on the same set of gestures, (2) evaluating the efficacy of our pipeline on a large set of gestures performed by multiple users and on smaller, highly differentiable subsets, and (3) investigating whether gestures can be accurately recognized when captured through three different types of materials and the ability of our pipeline to normalize their signal. We first describe each sensor, as well as the recording procedure and pre-processing applied to each dataset. We then provide information about the testing procedure of each experiment, discuss their results, and consider their implications for the design of radar-based gestural interfaces.

Chapter 8: Conclusion

In this last chapter, we conclude this thesis by summarizing our work and discussing future avenues for research.

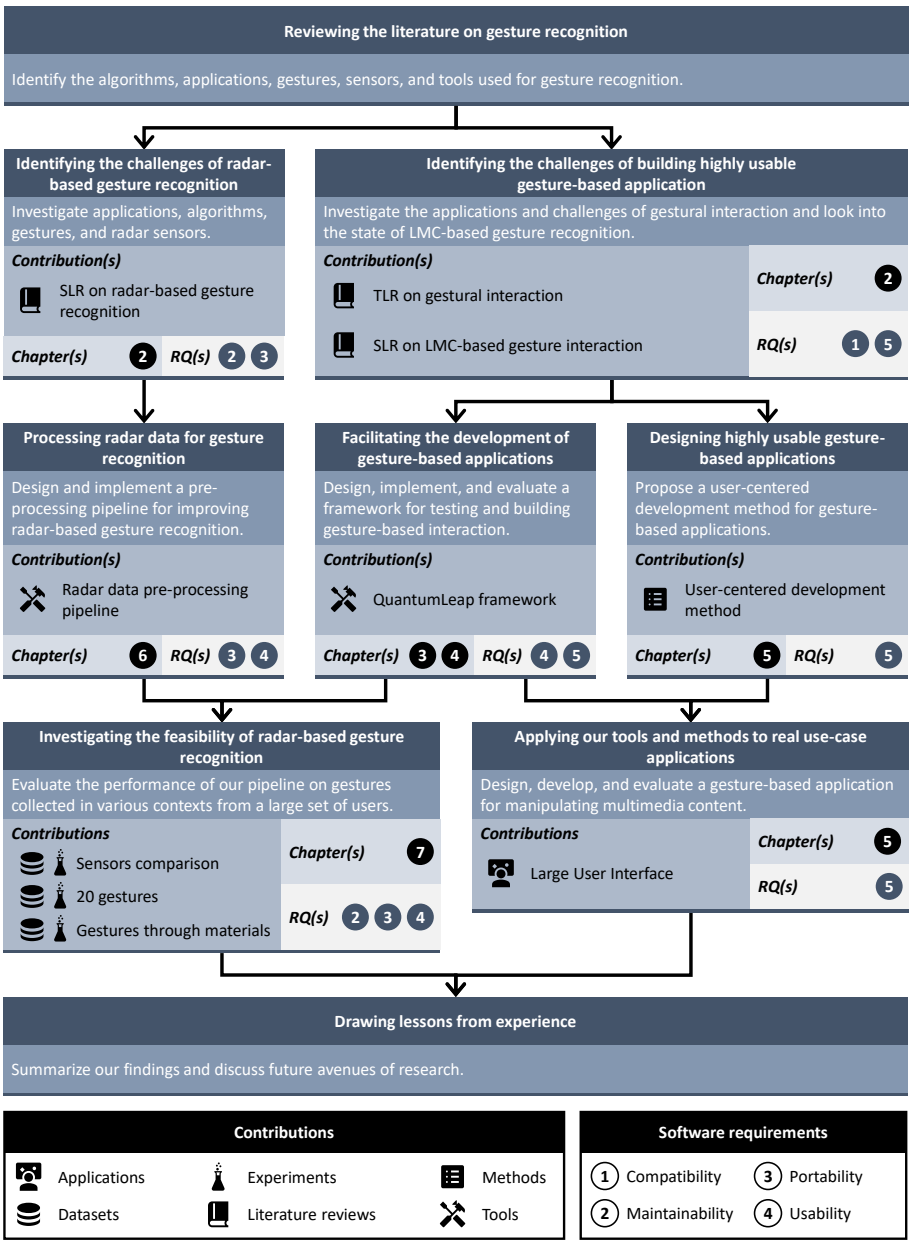


FIGURE 1.1: Overview of our research method. For each task, we provide the related contribution(s), chapter(s), and research question(s) (RQ).

Chapter 2

State of the Art

In this chapter, we explore the state of the art of gesture recognition in an attempt to answer the following research questions, defined in Section 1.2.1:

RQ1 *What are the main challenges of mid-air gesture recognition?*

RQ2 *What is the performance of radar sensors for mid-air gesture recognition?*

RQ3 *What types of gesture-based applications would benefit from radar sensors?*

RQ5 *How can tools and methods aid in designing gesture-based applications that operate independently of gesture recognition logic?*

Fig. 2.1 highlights the main contributions of this chapter and how it fits into this thesis. The rest of the chapter is divided into four main parts. Section 2.1 first provides an overview of gesture based-interaction. Section 2.2 then takes a look at a relatively mature vision-based technology for hand gesture interaction, called the Leap Motion Controller (LMC), to contextualize the state of the research on radar-based gesture interaction, which is explored in Section 2.3. Finally, Section 2.4 positions this work within the literature.

Publications. This chapter is based on three papers published in the EICS 2022 conference proceedings [268], Taylor and Francis IJHCI journal [269], and ACM TiiS journal [266].

Resources. The list of references from the two SLRs conducted in this chapter is available in an OSF repository at `https://osf.io/q43j8/?view_only=20b39a8a3d994ae4bd9ce52bfa2ded28`.

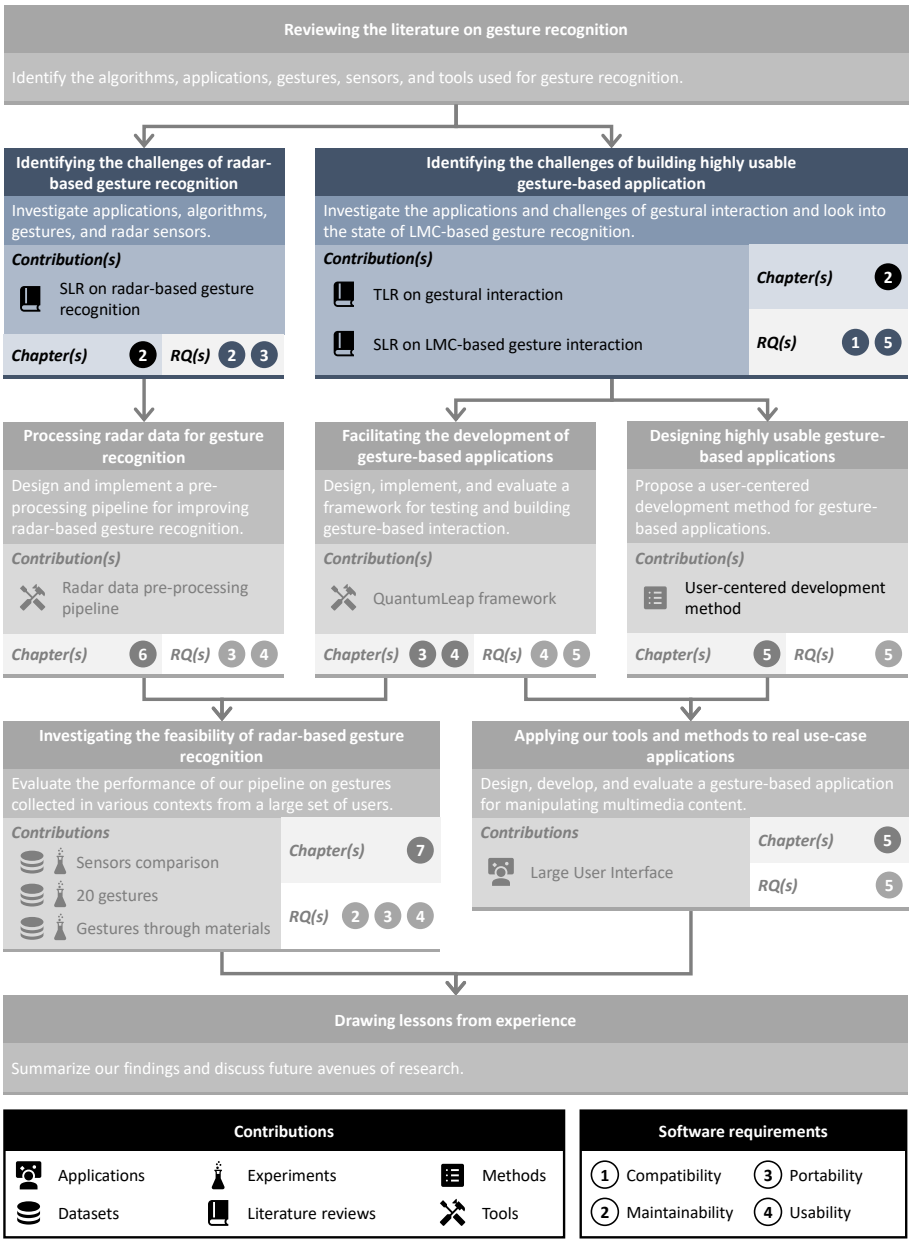


FIGURE 2.1: Main contributions of this chapter.

2.1 Overview of Gesture-based Interaction

We conducted a Targeted Literature Review (TLR) to provide a summary of the literature on the topic of gesture-based interaction. TLRs are non-systematic, in-depth, and informative reviews of the literature that keep only the references maximizing rigorousness and relevance while minimizing selection bias [146]. Section 2.1.1 first looks into gesture elicitation studies (GES) [329], which are a great way to involve end users in the development process of gesture-based applications. Section 2.1.2 then discusses the main challenges faced when building gesture interaction and Section 2.1.3 provides examples of gesture-based interfaces in the literature. Finally, Section 2.1.4 summarizes our findings.

2.1.1 Gesture Elicitation

Involving end users in the development process of gesture-based applications, *e.g.*, by conducting a GES to identify a set of highly-guessable gestures, is crucial to provide a good user experience. GESs were presented by Wobbrock *et al.* [329] as a participatory design method for understanding, collecting, exploring, and analyzing users preferences for gestures suitable to a particular context of use [33], which is made up of a user, a device or computing platform, an environment, and some related tasks. The GES outcome consists of a characterization of users' gesture input behavior in terms of user-defined gestures [95] with valuable information for designers, practitioners, and end users regarding the consensus levels between participants, computed as agreement or co-agreement scores [302], or more recently agreement rates [301]. Vatavu and Zaiti [302] suggested that a correlation exists between the agreement rate and the recall rate, *i.e.*, a higher agreement usually means that participants recall their gesture propositions better. This method has been widely applied to a large array of individual contexts of use (see [303] for a systematic literature review), such as in Augmented Reality [228], smartphones [241], TVs [302, 200, 65], and holograms [227].

The vast majority of these GESs target a single context of use, *i.e.*, a single category of users issuing gestures with a dedicated device in a given environment, thus posing the question of their transferability and applicability to other contexts. Among the few GESs targeting multiple contexts of use, Anthony *et al.* [13] observed that stroke gestures issued by a stylus *vs.* finger were inconsistent. In a systematic literature review of mid-air gestures, Groenewald *et al.* [96] observed that gesture sets were largely inconsistent with each other and that no holistic study existed to address this problem. Wittorf and Jakobsen [327] elicited mid-air gestures for interacting with elements on wall displays. They noted that users' hand poses were often irrelevant to the meaning of the proposed gestures (*e.g.*, a *swipe* gesture can be performed with any hand pose).

Instead, the meaning was conveyed through the direction or expression of the gesture. When referents resembled actions associated with touch interactions, the proposed gestures were often larger-scale versions of common touch gestures. Users tended to favor gestures resembling the physical manipulation of objects. Overall, compared to other studies on smaller displays, the participants of this study proposed larger and more physically-based gestures. Similarly, Ortega *et al.* [214] conducted a GES for eliciting multi-touch and mid-air gestures for 3D navigation in a procedurally generated universe. They found significant overlap between the proposed touch and mid-air gestures, and observed that most participants found mid-air gestures more immersive and intuitive than touch gestures.

Cross-context consistency has been initially addressed in three studies. Dingler *et al.* [64] designed a gesture set for reading via RSVP (Rapid Serial Visual Presentation) that ensures consistency across three devices, *i.e.*, a smartwatch, a smartphone, and smart glasses. A first GES collected gesture proposals for all three device types at once and considered the gesture proposals for all devices when constructing the final gesture set. They defined two new measures: the *transferability score* (*i.e.*, how well a set of commands learned on one device can be ported to another) and the *consistency score* (*i.e.*, the mean of all transferability scores between all devices). A second experiment with 18 participants validated the transferability of their gesture sets across devices. More recently, Vogiatzidakis and Koutsabasis [308] computed a consistency rate $CR(C)$ by computing the agreement for one user performing one of 14 commands on seven home devices (*i.e.*, a smart television, speakers, video player, audio player, air conditioner, lights, and blinds). For this purpose, they computed the agreement rate for a single command and a single user, but for different devices. In the same home context of use [309], they introduced a consistency routine process that checks gesture proposals for consistency before incorporating them into a real system.

2.1.2 Challenges

This section highlights the main challenges that developers and practitioners may face in the development of gesture-based applications and discusses potential workarounds.

Sensor Limitations

Depending on their technology, sensors may be subject to a series of limitations. For example, vision-based sensors such as the LMC are sensitive to environmental conditions [340], limited field of view [104], occlusion [27], and may raise privacy concerns [19]. On the other hand, radar-based systems preserve

privacy and can function in darkness, but are sensitive to clutter, noise, and interference. Some wearable sensors (*e.g.*, smart rings, watches, data gloves) may provide more accurate data, but this higher accuracy comes at the cost of practicality. Besides the limitations intrinsic to their technology, most sensors result in a compromise between a series of factors, including cost, field of view, framerate, and resolution.

Practitioners should consider these limitations into account in the design of their gesture-based user interfaces. Gesture sets should be designed such that the sensor(s) can differentiate between each gesture and the selected sensor(s) should suit the context of use of the UI. When no single sensor adequately fulfills the requirements of an application, integrating multiple sensors through sensor fusion should be considered to combine their strengths and enhance overall system performance. For instance, Marin *et al.* [191] fed features extracted from two vision-based sensors, an LMC and a Microsoft Kinect, into a Support Vector Machine (SVM) classifier to improve gesture recognition accuracy. Kalman filters are a powerful tool for sensor fusion, thanks to their ability to estimate the state of a system (*e.g.*, the position of individual hand joints) in real time from a series of noisy measurements from one or more sensors. They can help improve the reliability of gesture recognition in challenging conditions, like occlusion or varying lighting conditions. For example, Ovur *et al.* [217] used Kalman filters to merge data from two LMCs positioned at different angles. This approach enabled reliable hand pose tracking, even when one part of the hand obscured another from the viewpoint of one sensor (*i.e.*, self-occlusion). Similarly, Ponraj and Ren [229] leveraged Kalman filters to combine LMC data with two flex sensors integrated into a glove. The addition of flex sensors enabled more accurate hand pose recognition, particularly when parts of the hand were not visible to the LMC.

Jitter

Jitter can be caused by tracking errors, sensor noise, or even a slight tremor in the user's hands. Too much jitter can significantly hinder interactions with gesture-based applications [226], as it may induce recognition errors, as well as a lack of precision when pointing and selecting, especially on larger screens. It is thus important to perform at least some filtering in a gesture recognition pipeline to increase the precision of pointing and provide cleaner data to the gesture recognizers.

Depending on the sensors and recognition algorithms used, some filtering may already occur at several points in the pipeline. For instance, the LMC applies some filtering on the data that it provides [49], and the sampling performed by some recognizers such as 3 cent [36], Jackknife [284], or $P+ [295]

can act as a low pass filter. However, additional filtering may be applied to further smoothen the data.

Filters should be carefully selected and configured to provide reasonable smoothing while keeping latency low [226] and minimizing the risk of introducing inaccuracies. In [38], Casiez *et al.* propose the 1€ filter, a simple and fast filter based on double exponential smoothing that promises good smoothing and low latency for both fast and slow motion. They compare it to other filtering methods, namely, moving average, single and double exponential [156], and Kalman, and show that it manages *state-of-the-art* performance while being faster to execute.

Gesture Production Variability

Some variability in gesture production is usually observed both in gestures performed by a single user (intra-user variability) and by several users (inter-user variability). This variability can originate from differences in execution (*e.g.*, speed, trajectory) or in anatomy (*e.g.*, hand size, posture). If ignored, it can reduce gesture recognition accuracy and thus negatively impact the usability of gesture-based applications.

Several techniques exist to deal with gesture production variability. For instance, increasing the size of the training set by including more gesture samples produced by a greater range of users often results in higher recognition accuracy [297]. However, large training sets may increase execution time and are time-consuming to create. Synthetic data generation has been shown to increase accuracy while requiring little to no effort from developers [282, 161]. A series of tools, such as MAGIC [18, 134, 133] and Gesture à Go Go [161], have been developed to quickly augment gesture datasets with synthetic gestures. In [185], Maghoumi and LaViola used synthetic gesture generation in their DeepGRU recognizer to avoid overfitting.

To increase accuracy for single users, recognizers could be trained only on gestures performed by this specific user. Indeed, template-matching algorithms have been shown to perform better in user-dependent settings [297], *i.e.*, where the recognizers are trained and tested with samples produced by the same user. Coupled with synthetic data generation, one could achieve high accuracy with little input from the target user.

Finally, gesture recognition algorithms can be designed to accommodate some gesture production variability: for example, techniques such as Dynamic Time Warping (DTW) and spatial sampling can help deal with differences in gesture articulation speed [284, 297].

Static vs. Dynamic Gestures

Two main gesture types emerge from popular taxonomies: static and dynamic gestures [7, 300, 228, 46]. Inspired by [300], we propose the following definitions: a *static* gesture is a particular configuration of the body or body parts at one instant of time, whereas a *dynamic* gesture corresponds to a particular motion of the body or body parts through time. Different techniques may be leveraged to recognize each type of gesture. For instance, static gestures can be recognized at each frame to provide real-time recognition, while dynamic gestures are usually extracted from the raw stream of data (*i.e.*, gesture segmentation) before attempting to recognize them.

Gesture Segmentation

One of the greatest challenges in continuous gesture recognition is gesture segmentation, *i.e.*, extracting user gestures from a continuous stream of data. To maximize user satisfaction, a good gesture recognition pipeline should recognize all intentional gestures with minimal input from the user, while ignoring parasitic gestures (*e.g.*, transitions between gestures or unintentional motions) [18].

There are many different approaches to segmentation. A sliding window segmenter feeds the input data into a fixed-length buffer. The content of the buffer is continuously sent to the recognizer for evaluation. While this technique does not require input from the user, it makes assumptions about the length of gestures and relies on the recognizer to distinguish gestures from noise. Sliding windows have been used in combination with the 3 cent and Jackknife recognizers [34, 284]. They have also been ML-augmented to distinguish intentional gestures from parasitic movements [138]. Segmentation may also be triggered by user input, *e.g.*, pressing a button performing a simple motion or hand pose to delimit gesture boundaries, or performing the gesture in a specific "active" zone in space. While accurate, these techniques require active input from the user and are thus likely to feel less natural to the user. Some publications have focused on using some properties of input data to trigger segmentation, such as speed and acceleration. For instance, in [40], Chen *et al.* used the energy and root mean square (RMS) of raw EMG signals to properly segment gestures (energy-based segmentation). Other techniques such as Machete [283] rely on simple recognizers that continuously evaluate input data to determine the possible boundaries of gestures.

The variability in gesture production discussed previously also impacts gesture segmentation. Consequently, DTW has been employed to accommodate differences in gesture articulation speed. For example, Hernández-Vela *et al.* [108] assumed that two consecutive motions were always separated by an idle gesture. They utilized DTW to detect this idle gesture, making it possible

to extract user gestures from the continuous stream of data. Instead of relying on idle gestures, Tang *et al.* [281] opted for a different approach combining an SVM classifier with DTW for gesture segmentation. The SVM uses velocity information to detect "slow-down zones" occuring after a gesture has been performed and the DTW algorithm then backtracks to find the starting point of the gesture.

As gesture segmentation techniques are still far from perfect, some measures should be taken to limit the number of false positives. Examples include training recognition algorithms to recognize parasitic gestures (*e.g.*, by generating synthetic "noise" gestures) [284] and rejecting recognized gestures if their score falls below a threshold. This threshold may be arbitrarily defined or automatically computed based on the training set [34].

Overlapping Gestures

Two gestures are overlapping when part of one gesture is identical to part of the other gesture. While well-designed overlapping gestures may lead to increased memorability [238], they are challenging to recognize as they rely on near-perfect gesture segmentation. Overlapping may occur between two gestures of one gesture set or between a gesture from the gesture set and a parasitic gesture (*e.g.*, typing on a keyboard or reaching for an object). In the first case, one can design the gesture set in such a way that each gesture is easily distinguishable from all other gestures. In the second case, the designer should take into account as many potential parasitic gestures as possible to reduce the risk of false positives. Tools such as Ashbrook *et al.*'s MAGIC [18] assist designers in the creation of gesture sets with as little overlap as possible. GEsture Clustering toolKit (GECKo) [13] also helps investigate how end users articulate stroke gestures in terms of stroke number, order, and direction, thus assisting the designer in keeping or discarding a particular gesture to avoid overlapping.

Arm Fatigue

Arm fatigue poses a significant challenge for mid-air gesture interaction, as badly designed gesture sets may reduce the time users can interact with a system before needing to rest [102] or even lead to systems going unused [256].

This phenomenon, often dubbed the "gorilla arm effect", has led to various metrics being proposed to quantify user arm fatigue. For instance, Hincapie *et al.* [109] introduced the consumed endurance (CE) metric (Equation 2.1), calculated as the ratio between the interaction time and the endurance time of a participant (*i.e.*, the maximum duration they could sustain this type of

interaction before having to rest their arm).

$$CE(T, TotalTime) = 100\frac{TotalTime}{E(T)} \tag{2.1}$$

The authors demonstrated the applicability of this metric for assessing various design alternatives for gesture-based interfaces in a series of experiments. For instance, they found that extending the arm and holding it higher consumed more endurance than holding it closer to the body and at a lower position. Jang *et al.* [119] identified several limitations of the CE metric, including its oversight of rest periods and their impact on cumulative fatigue, its reliance on general assumptions regarding users' shoulder strength, and not applying well to interaction with low exertion levels. In response, they devised the three-compartment muscle (TCM) model, a more sophisticated model of muscle fatigue that addresses the limitations of the CE metric to quantify cumulative muscle fatigue with greater accuracy.

Hansberger *et al.* [102] illustrated that well-designed mid-air gestures could achieve fatigue levels comparable to those experimented with traditional interaction methods, like a keyboard. Conversely, they found that gestures designed without considering arm fatigue could lead to high exhaustion levels, with many users unable to complete a 30-minute game session in an experiment they conducted. Their study also suggested that the type of sensor used for gesture recognition could influence user fatigue. For instance, vision-based sensors like the LMC require users to place their hands within the field of view of the sensor, potentially leading to suboptimal hand position and thus increased fatigue, depending on the sensor placement. Wrist-worn IMUs, on the other hand, give users more flexibility in performing gestures. For example, Siddhpuria *et al.* [256] demonstrated that "at-your-side" gestures recorded with a smartwatch enabled users to interact with public displays in a way that felt socially acceptable and natural while minimizing arm fatigue. Similarly, Liu *et al.*'s Gunslinger [179] relied on two LMCs mounted on users' tights, enabling them to adopt a more relaxed posture during interaction.

Discoverability

Due to their novelty, mid-air gesture interfaces have yet to be standardized [210]. Variations among users, whether personal or cultural, pose challenges in creating gesture-based UIs that feel natural and can be used seamlessly without prior introduction.

Given the considerable freedom afforded by gestures, designers must establish a clear conceptual model of the application, perhaps even more so than with traditional interfaces [210]. Possible actions should be easily discoverable, and clear tutorials and feedback mechanisms should be implemented to ensure

that users can understand the physical constraints of the system (*e.g.*, limited field of view) [209] and learn the proper gestures without feeling overwhelmed or frustrated [210]. Signifiers, *i.e.*, deliberate or incidental cues about how an object is supposed to be used [211], should be placed by application designers in the UI to inform users about the potential actions available, and how to perform them. These cues should be subtle and seamlessly integrated into the interface to avoid distracting experienced users while remaining easy to notice and simple to understand for new users [237, 311]. Designers of gesture-based UI should also rely on existing conventions, such as those learned from touch-based UIs [209]. For instance, swiping left to display the next page of a document may resonate with users familiar with touchscreens, as they have already adopted this convention for 2D gestures. While leveraging existing conventions may lower the entry barrier of mid-air gesture interfaces, designers must be cautious, as not all users share the same conventions (*e.g.*, liking content with a thumbs-up gesture *vs.* by drawing a checkmark). Similarly, designers should be careful when introducing new conventions to avoid overwhelming users with many unfamiliar gestures or violating existing conventions, which could lead to confusion. GESs can be a useful tool to identify existing conventions and define new conventions.

Discoverability is particularly important in public displays, where capturing and retaining user attention is challenging given that user patience can be low in such scenarios [311]. As such, researchers have proposed qualitative scales to measure discoverability [252] and explored various techniques for teaching gestures to users. For example, Bau and Mackay [24] and Kristensson and Denby [140] investigated teaching 2D gestures on touchscreens by recognizing incomplete trajectories drawn by users and displaying possible continuations on the screen. These systems were later extended to 3D mid-air interaction, covering simple 3D trajectories [79] and gestures involving multiple body parts [237, 8]. Walter *et al.* [311] explored three strategies for revealing initial gestures on interactive public displays: spatial division (*i.e.*, displaying an explanation of the gesture on part of the screen), temporal division (interrupting the application to show the gesture in full screen), and integration (integrating visual cues in the application). They found that spatial division worked well, whereas temporal division caused users to leave while the cue was being shown because it interrupted their interaction with the application. White *et al.* [325] compared different methods for demonstrating to users how to interact with objects in AR. These methods included text (*i.e.*, a description of the gesture), diagrams (*i.e.*, an image of the gesture), and ghosting (*i.e.*, ghost images representing the action of the gesture). They found that combining two methods, such as text and ghosting, was preferred by users over using a single method.

Gesture Set Customization

Even a well-designed gesture set may not cater to the needs and preferences of every user, as factors such as motricity and personal preferences vary between individuals. Allowing users to customize the gesture set of an application can lead to higher memorability and usability compared to pre-designed gestures [213, 203], better accuracy, as users can submit their own gesture samples to train the system, and better inclusion of users with disabilities. However, enabling gesture set customization presents several challenges.

First, the gesture recognition pipeline must be able to incorporate changes to the gesture set with minimal inconvenience to the end user. The synthetic data generation techniques discussed previously may be leveraged to reduce the time required for end-users to record new gesture samples. Some techniques for gesture recognition, such as template matching algorithms like $P+ [295], Jackknife [284], or 3 cent [36], inherently support changes to their training set, making them particularly suitable for customizable gesture-based interfaces. Approaches based on deep learning can also accommodate new user-defined gestures [6] but come with their own challenges, such as catastrophic forgetting, a phenomenon where neural networks lose previously acquired knowledge when trained with new classes [249, 170].

Second, the gesture customization UI should be designed to be accessible to all users, including those with a disability who may not be able to perform the pre-designed gestures. Ideally, this UI should seamlessly integrate into the gesture-based application. Researchers have explored various methods for gesture customizations. For instance, Ouyang and Li [216] introduced a way to define custom shortcuts on touchscreens quickly and efficiently. Besides supporting arbitrary user-defined shortcuts, the system allows users to explore gestures proposed by others and can even identify the potential intent of a user's shortcut by comparing it with gestures from other users. Xu *et al.* [334] implemented gesture recognition into a smartwatch in a way that allowed users to record new gestures. The tool can assess whether a new gesture is sufficiently different from other gestures and activities (*e.g.*, cooking or walking) known by the system, whether the produced samples are consistent, and whether the gesture can be recognized accurately, providing feedback accordingly. While users can name their new gestures, the tool lacks a visual representation of the samples performed by the user.

Lastly, representing gesture samples to users, while rather straightforward for 2D trajectories, becomes more complex for 3D mid-air gestures. For instance, while vision-based sensors can record videos or extract skeleton data from the images [61, 172], visualizing data from other sensors, such as radars or IMUs, can be more challenging.

2.1.3 Applications

In the field of 3D gesture interaction [155, 136], hand gesture interaction (see [43] for a survey) is particularly attractive as the hand is one of the most mobile human limbs. As such, many hand gesture-based interfaces have been proposed, in particular for interacting with large screens, such as TVs and public displays.

Mid-air Gesture Interaction Techniques

The research on mid-air gesture interaction revealed some of its shortcomings, including arm fatigue [97], lack of consistency [96], and a need for short, pleasing, and easy-to-recognize gestures [133]. Several factors were identified that influence the quality of mid-air gestures on large displays, such as position and direction [84], sideways hand extension [135], dimension and bit cardinality [297], and continuity [133]. Beyond these factors, [204] identified three factors for designing mid-air pan and zoom navigation on large displays: uni- *vs.* bimanual interaction, linear *vs.* circular motion, and level of guidance (from high guidance with a 1-dimension motion on an input device to low guidance with a free 3D motion in mid-air). They studied the impact of these factors on users' performance and subjective preference by comparing 12 mid-air pan-and-zoom techniques with various factor combinations. Despite its appeal, free mid-air motion was less efficient and more tiring than other techniques. Overall, users preferred linear motion and bimanual interaction over other techniques. Various techniques for mid-air interaction with large screens have been studied in the literature to try to alleviate these shortcomings, but papers often focus on a few actions only, such as point-and-click or pan-and-zoom [203], or a limited set of gestures [96].

For instance, Yoo *et al.* [341] compared two techniques for interacting with large displays: point and dwell and a combination of push and grab-and-pull. When navigating Twitter feeds on a public display using a Microsoft Kinect for gesture recognition, the combination of push and grab-and-pull gestures was deemed more fun, faster, and less tiring than point-and-dwell.

Vogel and Balakrishnan [306] compared different techniques for freehand pointing and selecting a target on a large display, including two clicking techniques (airtap and thumb trigger) and three pointing techniques (ray casting, relative pointing with clutching, and hybrid ray-to-relative pointing). Auditory and visual feedback was implemented to compensate for the lack of tactile feedback inherent to mid-air gestural interfaces. Although clicking techniques were equally performant and accurate, airtap induced less fatigue than thumb trigger. Participants preferred relative pointing with clutching for pointing, as it felt faster and easier to perform.

Interaction with Multimedia Objects

The consumption and manipulation of multimedia objects, such as audio, video, pictures, and documents, is popular in public and private spaces, both on large displays (*e.g.*, TVs, public displays) and small screens (*e.g.*, smartphones, tablets). As such, enabling gesture-based interaction with multimedia objects is a relevant research subject in HCI, with works usually focusing on only one or two media types at a time.

For example, Ackad *et al.* [1] designed and evaluated a *media ribbon*, a large public display that shows information about events for theaters and concerts. Designing an intuitive and easy-to-learn gestural interface is crucial for such displays, as people are less inclined to take time to learn how to interact. They evaluated the interface by letting passers-by freely interact with the large display and interviewing them afterward. They suggested keeping gesture sets small and easy to learn, *e.g.*, by always displaying the most important gestures and by providing context-dependent tutorials for additional gestures.

Vlaming *et al.* [305] designed a system for interacting with documents, videos, and pictures on a large screen through a combination of pointing and pinch gestures. A modified Wiimote and an infrared light source tracked two retro-reflective points per hand (index and thumb). Multiple objects could be displayed simultaneously and manipulated through one- and two-handed gestures. Active objects could be controlled in various ways by interacting with media-dependent widgets. The authors observed that the system caused arm fatigue after prolonged use and that the use of two-handed gestures was sometimes misunderstood.

Zigelbaum *et al.*'s G-stalt [366] relied on an advanced motion capture system to track retro-reflective dots on each hand. A set of 21 gestures allowed users to manipulate and re-organize multimedia content in a 3D space. The authors recommended designing gestural interfaces to be neither too complex nor too simple and taking advantage of the hands' capabilities while keeping gestures easy to learn and perform.

Jakobsen *et al.* [118] compared touch and mid-air gestures for object selection on large displays. They observed that touch was usually faster and more accurate than mid-air gestures, but that this advantage decreased with the distance between objects, as mid-air gestures allowed users to see and interact with targets anywhere on the screen with minimal motion. The poor performance of mid-air gestures could be explained in part by an imperfect implementation of the system. They also showed that users mostly favored mid-air gestures when regularly forced to move away from the screen (*e.g.*, to interact with something else), despite the loss in accuracy incurred.

Siddhpuria *et al.* [256] designed *at-your-side* gestures, *i.e.*, gestures performed with the arm at rest along the side of the body, to minimize the effort associated with typical mid-air gesture interaction. *At-your-side* were considered as low-effort and low embarrassment by users, and thus suited for interaction with (semi-)public displays.

Drossis *et al.* [66] proposed MAGIC, a system for browsing a collection of pictures and videos on a large display in public spaces with mid-air gestures captured by a Microsoft Kinect. They observed that fine-grained gestures suffered from low recognition accuracy at a distance and thus replaced them with a combination of buttons and point-and-dwell.

2.1.4 Summary

Designing good gesture-based interfaces is a challenging endeavor in which user preferences often find themselves at odds with the technical challenges of gesture-based interaction. Finding a balance between usability and addressing these technical challenges should be possible, although it requires a high level of expertise in UX design, application development, and gesture recognition. This high barrier of entry and the lack of tools available to aid developers in this process partly explain why 3D gesture-based recognition is still relatively niche, despite its potential benefits to many facets of society.

2.2 Analysis of a Mature Sensing Technology

Looking at more mature sensing technologies, such as vision-based sensors, can help us better grasp the challenges of radar-based gesture interaction. In this section, we chose to focus on the Leap Motion Controller (LMC): an innovative, 3D motion-capturing device designed to track two hands and their fingers [287] with high precision and robustness [324]. Its API transforms the raw data from its two near-infrared cameras into a hand skeleton to be used in any application. The LMC reaches sub-millimeter (0.2 mm) accuracy for static gestures and falls to 1.2 mm for dynamic gestures [324]. It natively recognizes a handful of system-defined gestures but custom gesture recognition algorithms are often preferred as they enable the support of a wider range of gestures [27]. The affordability, off-the-shelf availability, and accuracy of the LMC make it relatively popular among researchers and practitioners.

We undertook a Systematic Literature Review (SLR) [132] to explore the algorithms employed for LMC-based gesture recognition (Fig. 2.2). The query `RQ="leap motion" AND "gesture recognition"` was run on September 20, 2020 in the ACM Digital Library[1] (ACM DL), which was highlighted

[1] Accessible at `https://dl.acm.org`

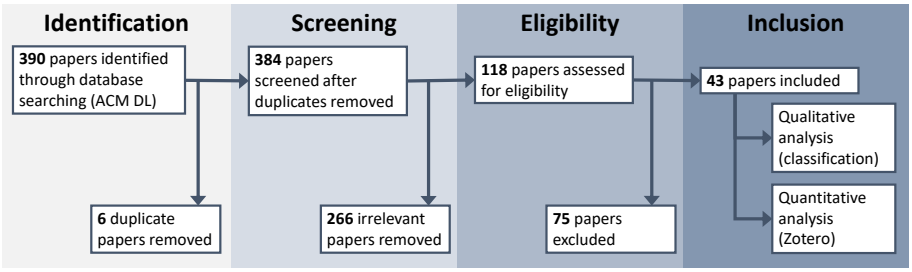


FIGURE 2.2: PRISMA diagram [218] of our SLR on LMC gesture recognition.

as a good option for computer science literature reviews by Gusenbauer and Haddaway [99]. ACM DL is recognized for its scientific references in computer science, primarily regarding software aspects, which are particularly relevant to this SLR, but not only. Other libraries can be considered although they have a different focus of interest, such as IEEE Xplore for technological and hardware aspects or ScienceDirect for human and sociological aspects. Focusing on one extensive database of papers provided us with a good overview of the literature on LMC-based gesture recognition while allowing us to channel most of our efforts into our subsequent SLR on radar-based gesture recognition (Section 2.3). The query yielded 390 publications, from which $N{=}43$ were identified as relevant.

2.2.1 Algorithms

22 types of algorithms for gesture recognition were identified in the papers, spanning across seven categories (Table 2.1).

A major part of the analyzed publications ($\frac{16}{43}{=}37\%$) implemented gestural support opportunistically, *i.e.*, by relying on the few system-defined gestures natively supported by the LMC or by recognizing gestures based on hard-coded thresholds (*e.g.*, [31, 86, 362]). While this opportunistic approach may be appropriate in some cases (*e.g.*, for quick prototyping[14] or very simple UIs), it often forces developers to make significant compromises that could be avoided with more advanced recognition techniques. For instance, Liang *et al.* [175] designed a storytelling system in which children can interact with hand gestures, but these gestures were hard-coded and could not be modified easily to better fit the motor abilities and preferences of each child. Zocco *et al.* [367] showed that an LMC-based system could be more efficient than the traditional trackpad-keyboard combination to interact with a Command and Control system. However, as they were limited to the few gestures natively supported by the LMC, they did not study whether more suitable types of

gestures could result in even greater efficiency and better usability, such as those obtained with user-defined gestures [95].

Found in nine references ($\frac{9}{43}$=21%), K-Nearest Neighbors (KNN) classifiers [69] were also a popular choice for gesture recognition. These algorithms are a simple yet powerful alternative to the opportunistic approach as they are easy to implement and train while being reasonably fast and accurate. Some are presented as all-purpose recognizers [36, 284] while others target more specific applications such as handwritten numerals recognition [45] and Sign Language recognition, which remains a key topic for any language: American [80, 142, 189, 250], Greek [260], and Thai [285].

The remainder of the papers featured more advanced ML techniques such as Neural Networks (NN), SVMs, Hidden Markov Models (HMM), or Ensemble Learning. These techniques were used in all kinds of applications. For

Type	**Algorithm**	**References**
Opportunistic	Hard-Coded Thresholds	[31, 35, 85, 86, 243, 286, 289, 336, 337, 342, 362]
	Leap Gestures	[35, 39, 58, 175, 195, 243, 286, 289, 337, 367]
	GameWAVE Software	[175]
Nearest Neighbor	K-Nearest Neighbors	[36, 45, 48, 58, 82, 92, 164, 284, 285, 274]
Support Vector Machines	Support Vector Machine	[80, 82, 274, 120, 131, 182, 190, 220, 260]
Neural Networks	Multilayer Perceptron	[189, 287]
	Deep Feedforward Neural Network	[250]
	Feedforward Neural Network	[60]
	Gated Recurring Units	[285]
	Neural Network	[120]
	Radial Basis Function Network	[345]
Hidden Markov Models	Hidden Markov Model	[144, 143, 310, 368]
	Coupled Hidden Markov Model	[142]
Ensemble Learning	Random Forest	[190, 250]
	Bagging Trees	[120]
	Gradient Tree Boosting	[131]
Other AI/ML Techniques	Decision Tree	[82, 274, 362]
	Decision Table	[171]
	Linear Discriminant Analysis	[120]
	Fuzzy Integral	[171]
	Multinomial Logistic Regression	[131]
	Naive Bayes	[231]

TABLE 2.1: Summary of the gesture recognition algorithms identified in the SLR.

instance, Simos and Nikolaos [260] used an SVM to classify the 24 letters of the Greek Sign Language alphabet with high accuracy, and Kumar *et al.* [144] trained an HMM to recognize words from single-stroke Latin sentences.

Among the 27 publications relying exclusively on non-opportunistic algorithms for gesture recognition, only four ($\frac{4}{27}$=14.8%) targeted both static and dynamic gestures, while the rest focused exclusively on one type of gesture.

Finally, some papers combined algorithms and/or sensors. Daniels *et al.* [58] combined the LMC's native gestures with Wobbrock *et al.*'s $1 recognizer [330] to recognize a larger set of gestures for the manipulation of protein structures. Jiang *et al.* [120] combined an LMC with myography data, Marin *et al.* augment the LMC with an MS Kinect [191] or a depth sensor [190] to increase accuracy.

2.2.2 Summary

Despite the popularity and ease of use of the LMC, we observe that none of the papers discussed in the previous section attempted to propose an approach that is reusable across gesture sets, gestural UIs, and applications towards their mainstream development. Reusability, interoperability, and maintainability, three important (sub-)factors for software quality defined in the ISO/IEC 25010 [116], are not explicitly addressed.

2.3 A Deep Dive into Radar-based Gesture Interaction

To get a comprehensive look at the state of the research on radar-based gesture interaction, an SLR was conducted to identify the types of radar, algorithms, and gesture sets used for radar-based gesture recognition (Fig. 2.3). The query `RQ="radar" AND "gesture recognition"` was run on December 21, 2020 in five digital libraries: (1) ACM Digital Library, (2) IEEE Xplore, (3) MDPI (Sensors), (4) SpringerLink, and (5) Elsevier ScienceDirect and resulted in 1,515 articles, from which $N = 118$ were identified as relevant. Table 2.2 groups these papers based on their year of publication. 91.5% of the papers were published between 2018 and 2020, indicating that this field of research has recently become quite active.

2.3.1 Applications

While all papers provided some evaluation of the system performance (see Fig. 2.4), only 14 papers ($\frac{14}{118}$=12%) demonstrated potential use cases of their

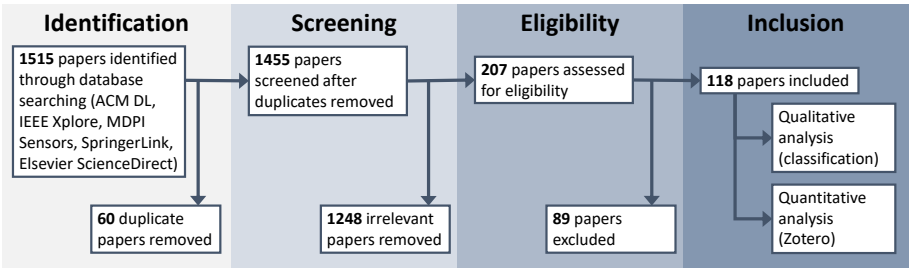


FIGURE 2.3: PRISMA diagram [218] of our SLR on radar-based gesture interaction.

Year	#Refs	Reference(s)
2009	1	[173]
2014	1	[312]
2015	1	[199]
2016	6	[130, 176, 188, 221, 316, 354]
2017	11	[153, 165, 193, 235, 245, 304, 351, 128, 129, 62, 126]
2018	21	[106, 115, 154, 152, 167, 166, 183, 206, 207, 225, 242, 244, 270, 275, 279, 335, 356, 358, 360, 355, 364]
2019	35	[5, 10, 11, 16, 17, 25, 41, 47, 51, 67, 71, 73, 78, 81, 87, 89, 93, 105, 107, 121, 141, 177, 262, 276, 319, 320, 314, 317, 323, 321, 359, 350, 357, 352, 361]
2020	41	[4, 9, 23, 68, 72, 98, 124, 127, 157, 159, 158, 168, 169, 174, 178, 180, 197, 222, 233, 234, 239, 247, 246, 263, 261, 277, 278, 290, 293, 318, 315, 313, 322, 332, 333, 343, 344, 347, 346, 349, 353]
2021	1	[3]

TABLE 2.2: Distribution of papers from our systematic literature review, based on their year of publication.

system with a prototype and one paper ($\frac{1}{118}$=1%) evaluated the user experience of the proposed system.

Eight of the 14 papers focused on multimedia applications, including controlling media playback (*e.g.*, music, videos) [67, 157, 332, 270, 178] or triggering specific keystrokes (*e.g.*, Enter, Alt+F4) [206, 207], thus enabling the control of classical PC applications without modifications to their source code. Liu *et al.* [178] introduced *mHomeGes*, a radar-based gesture recognition system for smart homes that allowed users to control lighting, music, and a home monitor. After evaluating the user experience of the system, the authors observed that 70% of the users found *mHomeGes* more convenient than voice commands. Smith *et al.* [270] implemented in-vehicle interaction for the driver and passengers of a car to control music and answer phone calls. Lien *et al.* [176] introduced *virtual tookits*, *i.e.*, small sets of gestures complex enough to enable the control of UIs while being memorable and easy for the system to recognize.

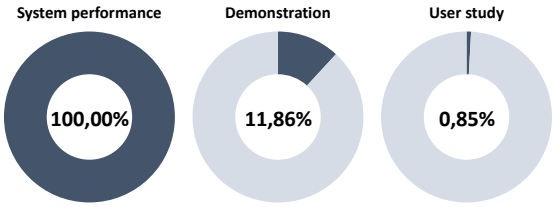


FIGURE 2.4: Types of validation.

They built a pipeline based on the Google Soli radar to recognize gestures in real-time and trigger actions in applications accordingly. Wu *et al.* [332] proposed to embed radar antennas into fabric to augment everyday objects (*e.g.*, furniture, clothing) with gesture-based interfaces. The authors developed prototypes to demonstrate various use cases, including controlling media playback with an interactive sofa armrest, checking fitness goals with interactive sports clothing, and creating interactive plush toys.

Other applications included air-writing recognition [315], user authentication based on 2D hand trajectories and breathing patterns [159], making a home-assistant for people with hearing impairements [247], manipulating medical images in an operating room [197], and controlling a robot with hand gestures [353, 173].

2.3.2 Algorithms

Fig. 2.5 presents the algorithms used in our corpus. 151 algorithms for radar-based gesture recognition were identified, the majority representing deep learning approaches ($\frac{81}{151} = 54\%$), of which Convolutional Neural Networks (CNN) were the most frequently used. CNNs were often combined with other models, such as Long Short-Term Memory (LSTM) or SVMs, in $\frac{23}{158} = 15\%$ of the cases. Classical machine learning algorithms were still used, as follows: K Nearest Neighbors ($\frac{23}{151} = 15\%$), Support Vector Machines ($\frac{16}{151} = 11\%$), ensemble learning ($\frac{8}{151} = 5\%$), Decision Trees ($\frac{6}{151} = 4\%$), Hidden Markov Models ($\frac{3}{151} = 2\%$), and Bayesian networks ($\frac{2}{151} = 1\%$).

From the above analysis, we make the following observations

1. *Predominance of CNN algorithms*: CNNs, often coupled with other models like LSTMs, are predominant for recognizing radar-based signals. They are particularly appropriate for image processing, but also they benefit from their intrinsic capability to select the discriminant radar features without any human supervision. This ability proves particularly useful when dealing with gesture sets that are very large in size and complexity, as radar images are very challenging to discriminate by a non-expert. An

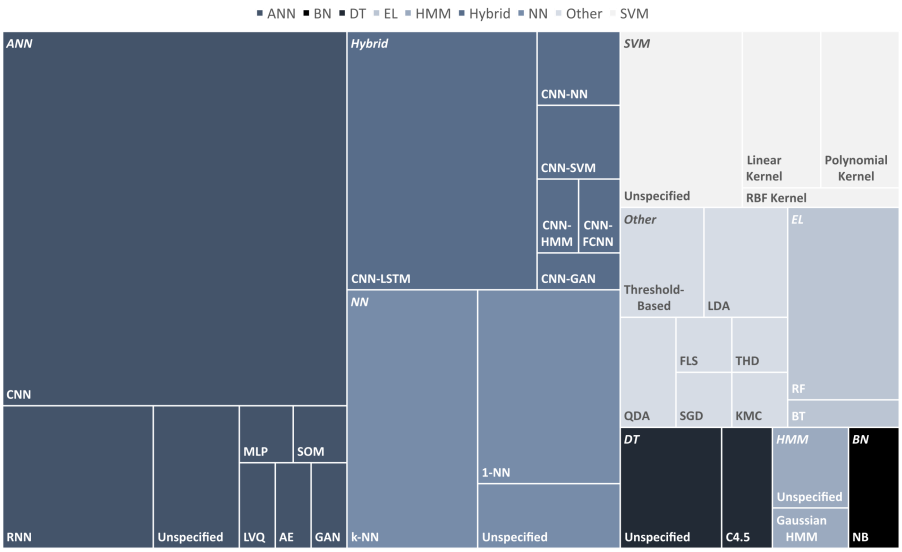


FIGURE 2.5: Tree map [255] of the algorithms for radar-based gesture recognition identified in the corpus. The color of a cell indicates the type of algorithm (*e.g.*, ANN, HMM, or SVM) while the size of a cell is proportional to the number of papers featuring this algorithm.

alternative to relying on CNNs for feature selection involves extracting point clouds from raw radar signals and directly inputting them into a classification algorithm, such as an LSTM [219]. This approach can be more efficient thanks to the smaller size of point clouds compared to raw radar data.

2. *Dependence of CNN with respect to the radar*: since CNNs automatically select features directly from radar images, these key features are not pre-trained and only learned when the gesture set is examined, thereby making the CNN very appropriate, but also very specific to the radar images produced. The CNNs therefore suffer from a high dependence on radar type and possibly radar unit for a given model.

3. *No modeling of the radar functioning*: since CNNs ensure most of the work for a specific radar, we did not identify any work in gesture recognition aimed at modeling the radar functioning in some way to ensure some radar independence or some mechanism to reason about its processing.

2.3.3 Gestures

Fig. 2.6 gives an overview of the types of gestures used in the identified papers. Most papers ($\frac{107}{118} = 91\%$) featured mid-air arm and hand gestures, which are relatively easy to recognize thanks to their scale that is neither too small, thus not requiring extremely fine range resolution, nor too large, which makes it possible for the radar to focus exclusively on the current user, thus simplifying signal processing. Almost half of the papers ($\frac{54}{118} = 46\%$) featured mid-air finger gestures, which require higher frequency radars (capable of finer range resolution) to accurately capture the fine-grained finger motion. Other papers featured whole body gestures [173, 168, 293], touch input [51, 332], or even head gestures [312]. We identified only seven publicly available datasets out of the 118 papers from our SLR; see Table 2.3. The mHomeGes dataset [178] comprises 10 types of arm gestures collected from 25 participants: arm up/down, push, pull, draw a circle, draw a zigzag, clap hands, mimic knocking on a table, yawn, and lift both arms. Each gesture was recorded in seven environments, including a bedroom and a parlor. [247] collected 50 different ASL signs from 15 participants in three environments (classroom, laboratory, and conference room) and five different scenarios for single and multiple users. Data were recorded simultaneously with a radar, a Kinect, and an RGB camera. Deep-soli is a dataset introduced by [316] that comprises 11 hand gestures performed by 10 users above Google Soli: pinch index/pinky, finger slide, finger rub, slow/fast swipe, push, pull, palm tilt, circle, and palm hold. Deep-Soli was also used by [25] to evaluate GestureVLAD, their proposed framework for radar gesture recognition. [234] introduce Dop-net, a dataset of four gestures recorded with two different radar sensors (CW and FMCW). Only the FMCW dataset was released in the form of a challenge. Dop-net was used in another paper to compare the performance of CW and FMCW radars for gesture recognition [23]. Finally, HARrad consists of two datasets of six gestures of arm and hand gestures collected from nine participants with a 77GHz radar [293]: drumming, shaking, swiping left/right, and thumb up/down. HARrad events contain six different actions, including entering or leaving a room, sitting down, standing up, getting dressed, and undressing.

From the above analysis, we make the following observations:

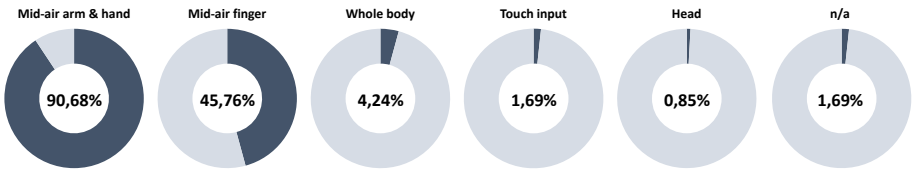


FIGURE 2.6: Types of gestures.

Name	Classes	Users	Samples	Sensor(s)	Base paper	Other papers
mHomeGes	10	25	>22000	TI IWR1443	[178]	/
mmASL (wake-words)	2	3	3700	NI multi-FPGA platform (w/ NI PXIe7902, 7976, 3610/3630) and SiBeam phased antenna array	[247]	/
mmASL (signs)	50	15	12236	NI multi-FPGA platform (w/ NI PXIe7902, 7976, 3610/3630) and SiBeam phased antenna array + Kinect + RGB camera	[247]	/
deep-soli	11	11	5500	Google Soli	[316]	[25]
dop-net	4	6	3052	Ancortek SDR-KIT 2400AD2	[234]	[23]
HARrad (gestures)	6	9	2347	77GHz FMCW radar	[293]	/
HARrad (events)	6	9	1505	77GHz FMCW radar	[293]	/

TABLE 2.3: Summary of the publicly available datasets identified in our systematic literature review.

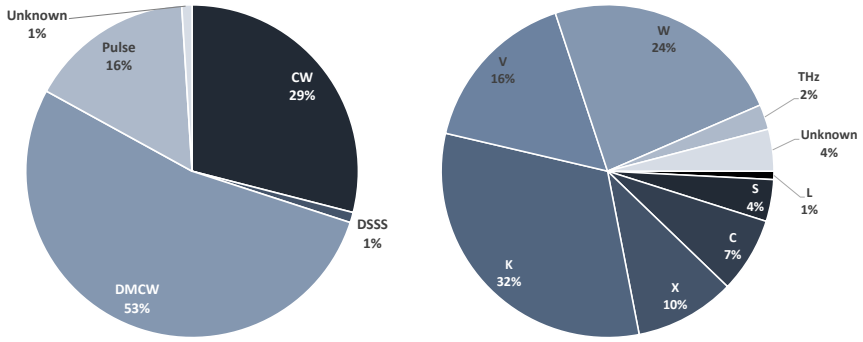
4. *Limited availability of datasets*: very little datasets are available for benchmarking.

5. *Huge size of datasets*: the size of the raw data captured for a single gesture is very high, making the entire dataset huge in size and challenging to process. Little or no dimension reduction has been observed.

2.3.4 Radars

We identified 123 radar systems and extracted their parameters, including the radar type, the frequency band in which they operate, the model, and their position. We identified four types of radars (Fig. 2.7a):

- *Frequency-Modulated Continuous-Wave* (FMCW) radars ($\frac{65}{123}$=53%), which continuously transmit a signal that varies in frequency, *i.e.*, modulated in frequency. These radars can resolve both the range and Doppler [111], enabling the recognition of static poses and dynamic gestures.



(A) Radar type. (B) Frequency band [75].

FIGURE 2.7: Distribution of the radars identified in our systematic literature review.

- *Unmodulated Continuous-Wave* (CW) radars ($\frac{36}{123}$=29%), which continuously transmit a signal at a constant frequency and amplitude [212]. These radars are simple and inexpensive. They can measure the Doppler shift of the return signal caused by moving objects but cannot provide range information or identify stationary targets, as the latter do not cause a Doppler shift. Consequently, radars of this type are not suitable for detecting static poses of the hand or body.

- *Pulse* radars ($\frac{21}{123}$=17%) transmit short radar pulses and listen for the returning echo. These radars can measure the range of targets, stationary and moving, making them suitable for the recognition of static poses and dynamic gestures. The pulse length affects the radar detection range (a longer pulse length increases the maximum range but may prevent the detection of very close targets) as well as its range resolution (a shorter pulse length allows for better discrimination between two spatially separated targets) [26]. The smaller the pulse length in the time domain, the larger the bandwidth in the frequency domain counterpart.

- *Direct-Sequence Spread Spectrum* (DSSS) radars ($\frac{1}{123}$=1%) transmit signals modulated by a random bit sequence. These radars can measure the range of stationary and moving targets, and their architecture is simpler than FMCW radars [280].

Most of the papers ($\frac{98}{118}$=83%) relied on a single radar sensor to capture gestures, while the rest ($\frac{20}{118}$=17%) used two or more radars. Among the latter, four papers introduced different radar models. Some papers introduced other types of sensors, such as [247], where gestures were collected using a

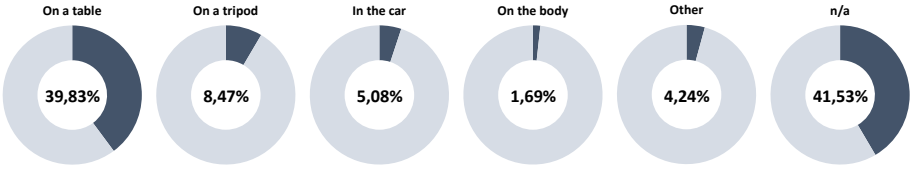


FIGURE 2.8: Positions of the radar.

radar, a Kinect, and an RGB camera for comparison purposes. [89] synthesized micro-Doppler radar signatures of gestures from the skeleton provided by a Kinect. Other papers performed data fusion with RGB cameras [199, 293], depth cameras [290, 199], and thermal cameras [263], respectively.

Fig. 2.8 summarizes the position of the radar systems identified in the user environment. Almost half of the identified papers ($\frac{49}{118} = 42\%$) did not provide enough information about the position of the radar(s) in the user's environment. From the papers that provided this information, 47 ($\frac{47}{118} = 40\%$) placed the radar(s) on a table, while 10 ($\frac{10}{118} = 8\%$) placed them on a tripod, and six ($\frac{6}{118} = 5\%$) in the car (*e.g.*, on the center console [199], near the steering wheel [5, 279], or on the roof console [276]). Other positions include on the body [197, 51], on fabric [332], and on a robot [173].

We also classified radars according to their frequency band according to the IEEE Standard Letter Designations for Radar-Frequency Bands [75] (Fig. 2.7b). The larger the frequency band, the better resolution the radar provides, and, in principle, the finer the gestures can be acquired. Fig. 2.7b shows that most of the radars ($\frac{88}{123} = 72\%$) operate in the K (18 to 27 GHz, 32%), V (40 to 75 GHz, 16%), and W (75 to 110 GHz, 24%) bands. The V-band encompasses the Google Soli [176], which operates at 60 GHz and was used in five papers [176, 316, 51, 25, 47] identified by our SLR. Three papers used very high-frequency THz-band radars (300 to 1000 GHz) to achieve better resolution [314, 313, 364]. Only $\frac{27}{123} = 22\%$ of the radars operated at frequencies below 18 GHz: 10% in the X-band (8 to 12 GHz), 7% in the C-band (4 to 8 GHz), 4% in the S-band (2 to 4 GHz), and 1% in the L-band (1 to 2 GHz). The benefit of lower frequencies is the larger detection range. For a given application, a trade-off has therefore to be chosen between resolution and range. The detection range can also be increased by increasing the transmitted power, but this increases energy consumption and can lead to adverse health effects. Fig. 2.10 summarizes the distribution of frequency bands by radar type.

Finally, we extracted the radar models used in the papers that provided this information; see Fig. 2.9. More than a third of the papers ($\frac{46}{123} = 37\%$) did not provide enough information to identify the exact models. Of the 77 remaining publications, we identified 29 unique models with the most popular being the Texas Instruments AWR1642 ($\frac{13}{123} = 11\%$) and IWR1443 ($\frac{5}{123} = 4\%$), the

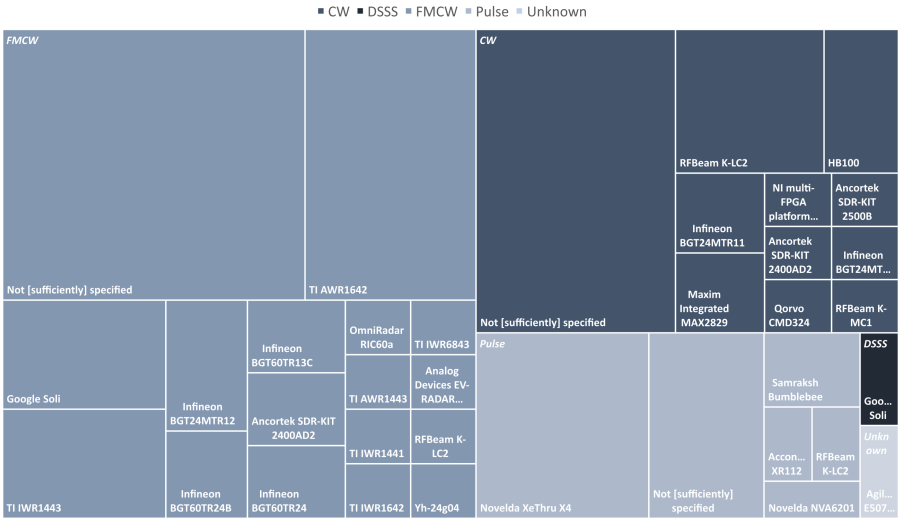


FIGURE 2.9: Tree map [255] of the radar models identified in our systematic literature review. The color of a cell indicates the type of radar (CW, DSSS, FMCW, Pulse, or unknown) while the size of a cell is proportional to the number of papers featuring this model of radar.

Novelda XeThru X4 ($\frac{9}{123} = 7\%$), the RFBeam K-LC2 ($\frac{8}{123} = 7\%$), and Google Soli ($\frac{6}{123} = 5\%$), respectively. Except for Google Soli, which was designed for integration into smartphones and smart speakers, most of the radar systems from the SLR were not meant for everyday use. However, some of the papers introduce prototypes of radars that could be integrated into furniture [193] or even stitched into clothing [332].

From the above analysis, we make the following observations:

6. *Predominance of FMCW over other types of radar (Fig. 2.7a and 2.10)*: these radars are the most frequently exploited due to their ability to sense both static and dynamic gestures but also because several models are commercially available. The frequency bands covered by these two categories are high enough to acquire raw data with high resolution.

7. *Limited coverage of very low/high-frequency bands* (Fig. 2.11): lower frequencies result in worse range resolution, making it more difficult for the radar to distinguish between two targets. Such radars are thus less interesting for smaller-scale gestures, such as hand and finger gestures, which explains the lower number of works covering these systems. On the contrary, higher frequencies enable better range resolution and thus are better suited for small-scale hand and finger gestures (*e.g.*, Google

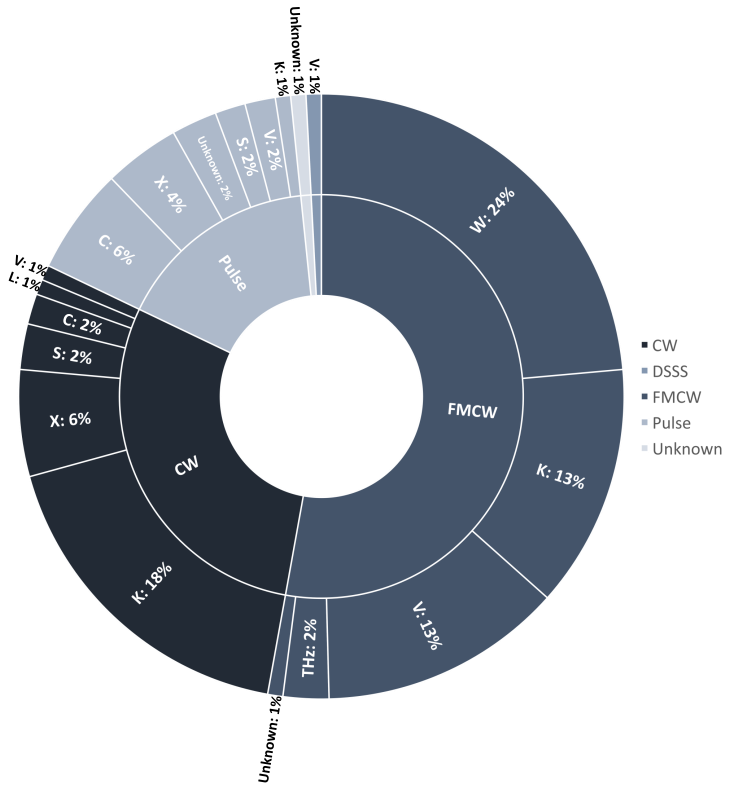


FIGURE 2.10: Sunburst chart of the radars identified in the SLR. The inner ring and color show the proportion of each radar type, while the outer ring indicates the proportion of each frequency band for each radar type according to the IEEE standard [75].

Soli [176, 316]), at the cost of higher complexity and lower range as the frequency increases. Most systems operate at frequencies between 4 GHz and 100 GHz, which can provide sufficient range resolution with relatively low complexity and power consumption. In addition, some off-the-shelf radar systems used in HCI research were originally developed for other applications and operate specific frequencies, such as collision avoidance systems in cars, which often operate in the K-band.

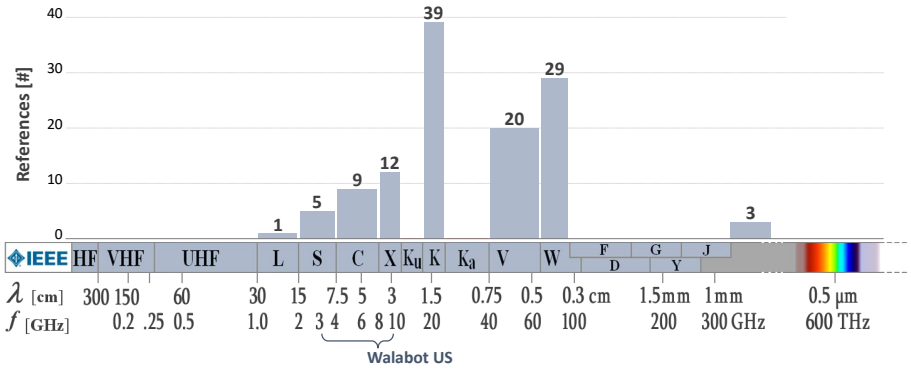


FIGURE 2.11: Number of radar systems according to their frequency band [75] distributed along the electromagnetic continuum (Bottom image of waves and frequency ranges used by radar used with permission from Christian Wolff [331]), from a total of 118 radars. Five radars did not mention their frequency band.

2.3.5 Summary

Radar-based gesture recognition is still in its infancy. Current research mostly focuses on radar systems, signal processing, and algorithms for gesture recognition. As a result, few papers demonstrate a potential use case of their system with a prototype application. In addition, most of the papers rely on deep learning techniques tailored to a specific radar system and gesture set. The combination of a wide variety of sensors, often custom-designed, and a lack of publicly available datasets makes it difficult to reproduce and compare results.

2.4 Positioning this Work in the Literature

This chapter provided an overview of gesture-based interaction in the literature. We looked at how to involve users in the development process of gesture-based applications through GESs and identified the main challenges faced by developers when implementing real-time gesture recognition. These challenges include taking into account sensor limitations, extracting gestures from a continuous flow of data, and supporting some variability in the gestures produced by end users. We then had an in-depth look at the LMC, an affordable off-the-shelf device for vision-based gesture recognition. Despite the popularity of this device, most of the papers identified implemented gesture recognition in an opportunistic manner, which could not be adapted easily to support other devices, gestures, or applications. This highlights a need for

a tool that would enable the efficient reuse of components designed by, *e.g.*, the scientific community, in the development of gesture-based applications. We made several observations regarding three of the four software quality properties defined in Section 1.2.1. Regarding *maintainability*, we observed a reliance of many papers on non-reusable, often opportunistic, approaches to gesture recognition. As such, supporting other gestures may be impossible without completely re-writing the gesture recognition logic. In addition, we noticed a lack of clear separation of concerns between user interfaces and their gesture recognition logic, which often unnecessarily complicates future modifications. Similarly, the reliance on these opportunistic approaches hinders software *portability*, as they are often tied to the exact sensor used. Supporting other sensors would thus require substantial work. As for *usability*, the scarcity of tools available to developers presents significant entry barriers for creating highly usable gesture-based applications. Indeed, developing such applications currently requires expertise in gesture recognition, UX design, and application development.

We also looked at radar-based gesture recognition, to identify any application, algorithm, sensor, and dataset described in the literature. Regarding the *compatibility* software quality property, we observed a scarcity of publicly available datasets and noted that most gesture recognition techniques were tailored for one radar system in particular, making it difficult to reuse datasets across different systems. From a *maintainability* point of view, a majority of the analyzed papers relied on deep learning techniques for gesture recognition. While effective, they posed a challenge for future updates, as changing the sensor or the gesture set could require extensive re-training to achieve the same level of performance. In terms of *portability*, we did not observe any consensus on an affordable and efficient off-the-shelf radar sensor, resulting in a plethora of different (custom) radar systems. Making matters worse, a large part of the proposed gesture recognition techniques were specific to one sensor and gesture set, which limited their applicability to other applications. Regarding *usability*, we noticed that very few papers demonstrated potential use cases of their system with prototype applications. In addition, most of the proposed techniques could not support user-submitted gestures without extensive re-training, which could negatively impact usability, should they be used in real applications [203].

In conclusion, most of the papers that we analyzed in this chapter did not properly address four of the ISO/IEC 25010 software quality properties [116], namely compatibility, maintainability, portability, and usability. Having identified the main challenges of (radar-based) gesture recognition, the rest of this thesis will focus on the development of tools and methods that streamline the development of highly usable gesture interfaces.

Chapter 3

Facilitating the Development of Gesture-based Applications

The landscape of contactless UIs is constantly evolving. Sensors such as the LMC are paving the way towards cheap and accurate gesture recognition, while researchers are continuously working on improving gesture recognition algorithms. However, the lack of a common framework for gesture recognition is making it increasingly harder to keep track of this evolution outside of the research community, resulting in only a handful of real-life applications (see Sections 2.1.3 and 2.2). As such, this chapter attempts to answer two of the five research questions defined in Section 1.2.1:

RQ4 *How can we foster collaboration between researchers and practitioners working on (radar) gesture recognition?*

RQ5 *How can tools and methods aid in designing gesture-based applications that operate independently of gesture recognition logic?*

To that end, it introduces QuantumLeap, a software tool that aims at bridging the gap between researchers and developers by providing a common framework to (1) help researchers share the results of their work in a way that is easily reusable and (2) allow developers to create gesture-based applications without spending time to solve the numerous challenges of gesture recognition (see Section 2.1.2). The main contribution of this chapter and how it fits into the rest of this thesis are summarized in Fig. 3.1.

The rest of this chapter is organized as follows. Section 3.1 first provides an in-depth description of QuantumLeap, including its dataflow architecture, configuration, and API. Section 3.2 discusses the results of an experiment aimed at evaluating the overall usability of QuantumLeap, which involved seven developers with different levels of expertise. Section 3.3 then explores how

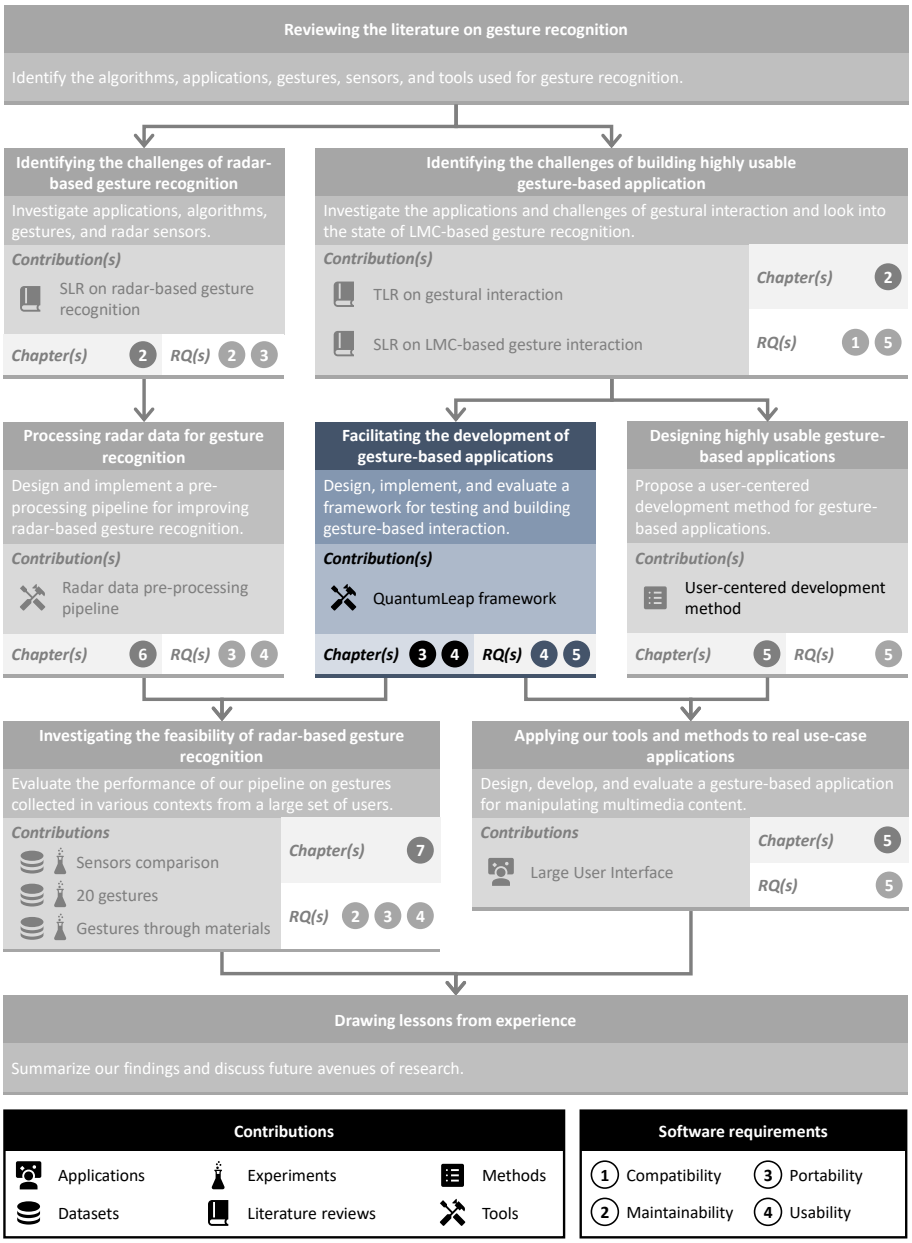


FIGURE 3.1: Main contributions of this chapter.

some recent projects utilized and extended the framework. Finally, Section 3.4 discusses the advantages and limitations of QuantumLeap, and Section 3.5 concludes this chapter.

Publications. This chapter is adapted from a paper published in the EICS 2022 conference proceedings [268].

Resources. The QuantumLeap framework is available on GitHub at `https://github.com/sluyters/QuantumLeap`. Demonstration videos of applications built with QuantumLeap are available at `https://www.youtube.com/playlist?list=PLj1KcEqIa0fHyDyeW_NpiLHCP6A8j7mS3`.

3.1 A Framework for Building Gesture-Based Applications

QuantumLeap is designed as an intermediate layer between gesture-based applications and gesture sensors. It contains all of the logic necessary for gesture recognition and only exposes a simple JavaScript API that allows developers to map gestures to actions without focusing on acquiring, segmenting, and recognizing gestures. Implemented in JavaScript (ECMAScript 2019) [114], the QuantumLeap framework is divided into two parts according to Pree's notation [230]: *frozen spots*, which are "fixed" parts which remain constant for all of its instantiations, and *hot spots*, which are generic parts to be modified depending on the context of use [33]. QuantumLeap is a *gray box* software framework [223], as it provides developers with two options: pick modules from a list of predefined components (black box) or write new components that meet their needs better (white box). The rest of this section describes the overall software architecture of QuantumLeap.

3.1.1 Dataflow Architecture

The dataflow architecture of QuantumLeap (Fig. 3.2) has been designed to be versatile and simple to configure (Section 3.1.2). It consists of eight modules (*i.e.*, hot spots), each stemming from the challenges of gesture recognition discussed in Section 2.1.2: a sensor module, filtering module, loaders for static and dynamic datasets, static and dynamic gesture recognizers, gesture segmentation module, and an analyzer module. Various implementations are provided for each module to suit the needs of a specific application (see Appendix A). When these implementations are not suited for a particular application, the framework is extensible and developers are thus encouraged to

Id	Description
1	`Frame` object.
2	`Sample` object.
3	JavaScript object with three properties: the gesture type (*i.e.*, static or dynamic), the gesture name, and the data from the analyzer if the gesture is static.
4	WebSocket message of type `Data` containing frames and static/dynamic gestures if detected.
5	WebSocket message of type `Operation`.

TABLE 3.1: Data structures for inter-module communication.

propose new implementations. The modules communicate together using various data structures described in Table 3.1. The rest of this section provides information about each type of module.

Sensor

This module serves as an interface between QuantumLeap and any physical sensor, such as an LMC, a radar sensor, or a Microsoft Kinect. It transforms raw, sensor-specific data into a standardized format usable throughout the framework. This module is called a set number of times per second by QuantumLeap to retrieve tracking data at one instant in time. Depending on the context of use, multiple sensor modules may be combined, *e.g.*, to increase recognition accuracy. A sensor module should implement three methods:

- `getPoints(timestamp)`: get a list of points from the sensor at this instant in time.
- `connect()`: connect to the sensor.
- `disconnect()`: disconnect from the sensor.

One sensor module is provided with QuantumLeap. Its description can be found in Appendix A.1.

Filter

As discussed in Section 2.1.2, noise can negatively impact the user experience by reducing pointing and gesture recognition accuracy. Before it can be used by the other modules, each frame is first processed by the filter module, which computes a new frame with reduced noise. Filtering should be carefully tuned to fit the sensor and the target application, as too much noise reduction runs the risk of introducing some lag. A filter module should implement one method:

- `filter(frame)`: compute a new frame with less noise.

Four different modules are provided and described in Appendix A.2.

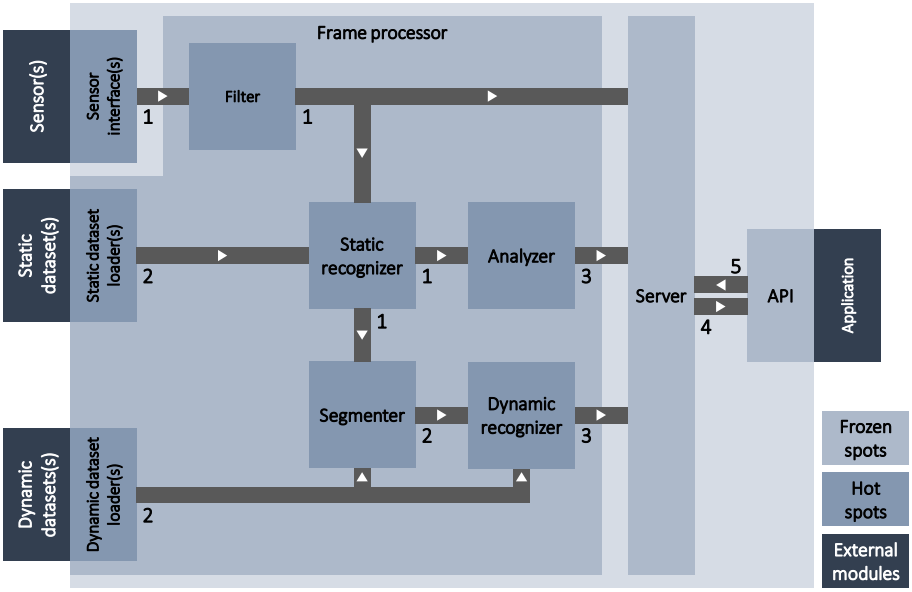


FIGURE 3.2: QuantumLeap dataflow architecture.

Static and Dynamic Datasets

These modules transform a gesture dataset into a `GestureSet` object, *i.e.*, a standardized format compatible with the framework. Datasets are then used to train the static and dynamic recognizers. Fig. 3.3 depicts the UML V2.5 [30] Class diagram of the modular data structure used in the framework. The `GestureSet` object can contain samples of two types: (1) `PoseData`, for static gestures consisting of a single frame that represents a hand pose; and (2) `StrokeData`, for dynamic gestures, which comprise multiple paths, where a path corresponds to the trajectory [37] of a specific joint (*e.g.*, the tip of the right index). Paths are in turn decomposed into a series of strokes. Only one method is required:

- `loadDataset(name, path, identifier)`: load a gesture dataset into memory as a `GestureSet` object.

Developers have two possibilities for selecting a gesture set. They can create a custom gesture set with any specialized software, such as MAGIC2 [134] or GESTMAN [187], or via an LMC gesture recorder[1]. Another option is to

[1]We release the code of the LMC Gesture Recorder on GitHub at `https://github.com/sluyters/LeapGesturePlayback`.

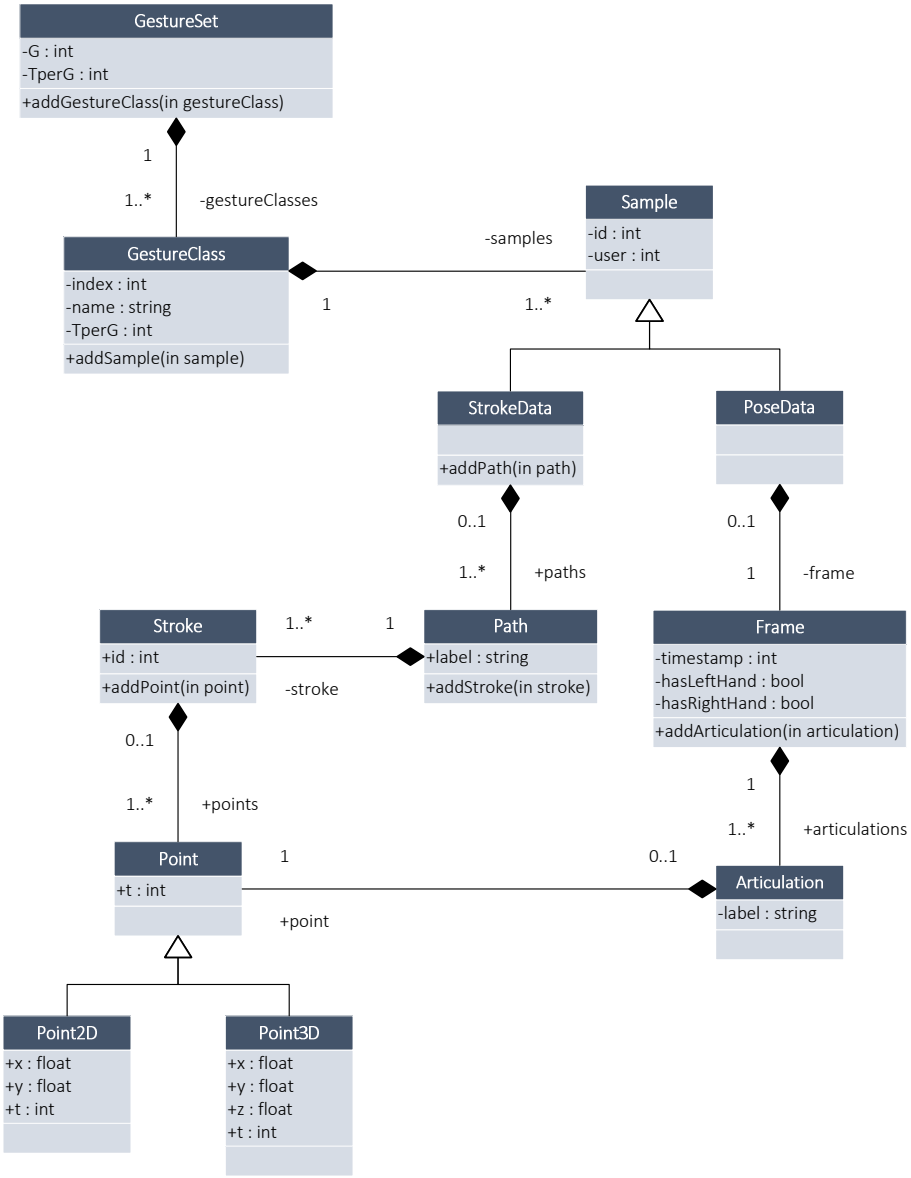


FIGURE 3.3: UML V2.5 class diagram of the gesture sets.

pick one of the many publicly available datasets, such as SHREC2019 [34], and convert it into the standardized format to be considered in QuantumLeap.

Static Recognizer

The static recognizer module attempts to recognize static gestures at each frame. For each recognized gesture, the recognizer should compute a confidence score ranging from 0 to 1, where 1 indicates that it is absolutely confident that it has correctly identified the gesture. If the score is lower than a manually set threshold, the gesture is rejected. In the future, this threshold could be automatically computed by QuantumLeap for each gesture class based on the training dataset(s). Having a dedicated module for static gesture recognition allows using more efficient recognizers that are specialized for this task. This should thus result in increased accuracy and lower CPU usage. A static recognizer module should implement three methods:

- `addGesture(name, frame)`: add a static gesture to the training set of this recognizer.
- `removeGesture(name)`: remove a static gesture from the training set of this recognizer (including all of its samples).
- `recognize(frame)`: determine the name of the static gesture performed in the frame given as argument.

To avoid accidentally triggering dynamic gestures while the user is performing a static gesture, the recognition of dynamic gestures is stopped when a static gesture is recognized. Two static recognizer modules are provided and described in Appendix A.4.

Analyzer

These modules extract additional data from the sensor frames, which can then be used inside applications. They can serve to offload heavy computation from the front-end to the QuantumLeap dataflow, to implement more advanced feature extraction techniques that would otherwise clutter the front-end application, or to remove sensor-dependent code from the application. Any analyzer module should implement two methods:

- `analyze(frame)`: extract relevant information from a frame.
- `reset()`: reset the internal state of the analyzer.

As analyzer modules are highly specific to the application, we expect developers to create their own modules. However, one simple module is provided and described in Appendix A.5.

Segmenter

The role of this module is to identify intentional dynamic gestures from a continuous stream of data. It analyzes frames one by one until it identifies a gesture. It then returns one or more lists of frames (*e.g.*, if it has multiple sliding windows), which are subsequently transformed into `StrokeData` objects that are sent to the dynamic gesture recognizer for recognition. If multiple lists of frames are returned at the same time, only the gesture with the highest confidence score is kept. The most appropriate method for gesture segmentation varies depending on the application, environment, and sensor(s). For instance, a simple sliding window segmenter could be appropriate for a multimedia application, while a more accurate technique may be required in environments where mistakes are costly, such as operating rooms. It is thus important to leave the choice of segmenter to the designer. A segmenter module should implement four methods:

- `addGesture(name, sample)`: add a dynamic gesture to the training set of the segmenter.
- `removeGesture(name)`: remove a dynamic gesture from the training set of the segmenter (including all of its samples).
- `computeSegments(frame)`: process a frame and return one or more lists of frames (*i.e.*, segments) if the frame marks the ending of a gesture.
- `notifyRecognition()`: notify the segmenter that a gesture has been recognized. The segmenter may use this information to take the coarticulation between two subsequent gestures into account (*e.g.*, by temporarily pausing gesture segmentation after a gesture is recognized).

Three different segmenters are provided and described in Appendix A.6.

Dynamic Recognizer

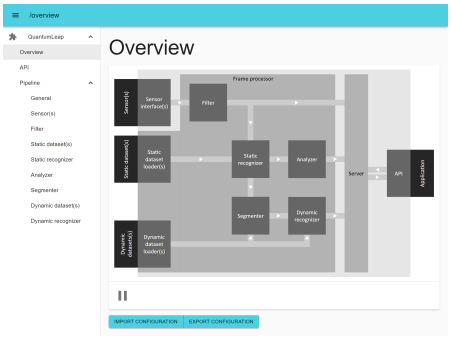
Distinct from the static recognizers, dynamic recognizers analyze sequences of frames instead of individual frames. They are thus called only when the segmenter detects an intended gesture, which limits CPU usage and reduces the number of false positives. As for the static recognizers, they should return a confidence score for each recognized gesture and implement three methods:

- `addGesture(name, sample)`: add a gesture to the training set of this recognizer.
- `removeGesture(name)`: remove a gesture from the training set of this recognizer (including all of its samples).
- `recognize(sample)`: determine the name of the gesture that corresponds to the candidate sample given as argument.

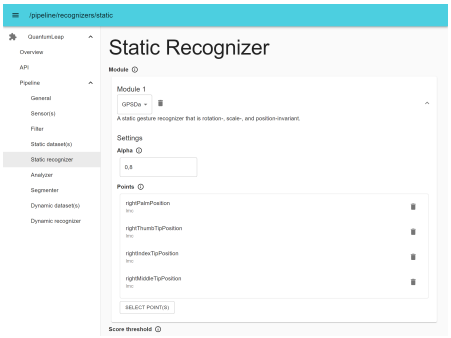
Ten dynamic recognizers are provided that cover multiple properties of gesture sets, such as uni- *vs.* multi-stroke and properties of invariance (see Appendix A.7).

3.1.2 Configuration

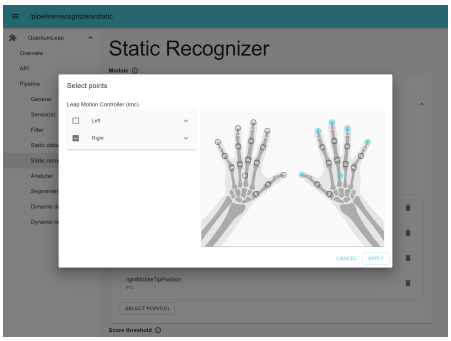
QuantumLeap features a configuration graphical user interface (GUI) that allows developers to quickly set it up for their needs. It is divided into a series of pages: an "overview" page which shows the complete dataflow architecture and allows users to navigate to the page of each module (Fig. 3.4a), an "API" page that contains the complete API documentation (Fig. 3.4d), and dedicated "module" pages for each type of module (Fig. 3.4b and 3.4c). The "module" pages let developers select a specific implementation of the module and configure it to their needs (*e.g.*, select the points used by a recognizer). The settings of each module can be defined in their corresponding "config-template.json"



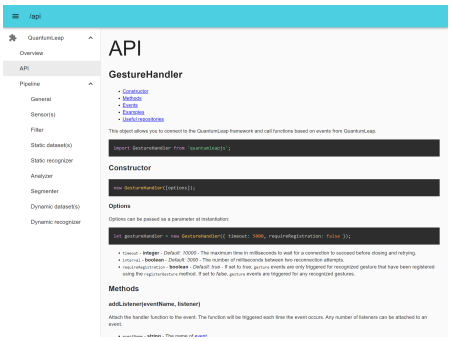
(A) Overview.



(B) Static recognizer settings.



(C) Point selection.



(D) API documentation.

FIGURE 3.4: Screenshots of the QuantumLeap GUI.

file. Framework configurations can be imported and exported as JSON files from the GUI.

The GUI provides limited guidance in the form of short descriptions for each implemented module to help developers select the best set of modules for their application. Further guidance, such as recommending the most appropriate modules (*e.g.*, gesture recognizers) for a particular application, is currently outside of the scope of this paper. However, resources like the comparative studies of Magrofuoco *et al.* [186] or Erazo and Pérez-Medina [76] may prove extremely valuable when choosing, *e.g.*, a gesture recognizer.

3.1.3 Application Programming Interface

The QuantumLeap API exposes the `GestureHandler` class, a member of the `EventEmitter` class of the Node.js core API. A `GestureHandler` object attempts to connect to a running instance of the QuantumLeap framework. It then emits events based on the data received from QuantumLeap. The reliance on a standard event-driven architecture should flatten the learning curve for front-end developers and facilitate the transition to gestural UIs, as events are already an integral part of modern UIs. The methods provided by the `GestureHandler` class include:

- `connect([address])`: connect to a running instance of QuantumLeap.
- `disconnect()`: disconnect from QuantumLeap.
- `registerGestures(type, names)`: register gestures to QuantumLeap, to enable their recognition.
- `unregisterGestures(type, names)`: unregister gestures to stop them from being recognized.
- `addListener(eventName, listener)`: attach a listener to a particular event. Possible events include:
  - `frame`: a frame has been received from QuantumLeap.
  - `gesture`: a gesture (static or dynamic) has been recognized.
  - `connect`: the application has connected to QuantumLeap.
  - `disconnect`: the application has disconnected from QuantumLeap.
- `removeListener(eventName, listener)`: remove a listener attached to an event.

3.2 Evaluation

The ISO 9241-Part 11 [117] defines the "Usability" quality factor appearing in the ISO 25010 [116] as "*the effectiveness, the efficiency, and the satisfaction with which specified users achieve specified goals in particular environments*". Therefore, to evaluate the overall usability of QuantumLeap, an experiment was conducted to quantitatively evaluate the task completion rate (to cover effectiveness), the task completion time, and the cognitive workload based on NASA-TLX [103] (to cover efficiency), and the System Usability Scale (SUS) [28] to assess its overall usability (to also cover satisfaction). This section describes the evaluation procedure and discusses its results.

3.2.1 Experiment Protocol

Environment

Participants were sitting in front of a computer in a quiet room. An LMC was already set up and placed in front of them. Each participant was provided with two elements:

- A small React web application consists of four buttons linked to four gestures ("left swipe", "right swipe", "thumb", and "point index"). Clicking on a button displays the type, name, and image representation of the gesture on the screen.
- The QuantumLeap framework, including the documentation of its API. The configuration of the framework was reset before each experiment.

Participants

A purposive and snowball sampling was used to recruit 7 participants (all males), aged between 21 and 27 years old (M=23.3, SD=1.9), who volunteered for the experiment. They reported between 1 and 7 years of experience in the field of computer science (M=4.9, SD=2.2), and expertise levels ranging from 5 to 9 (M=7.6, SD=1.8). Education levels included bachelor's degree (4 participants) and master's degree (3 participants). Our criteria for inclusion in this experiment required participants to have at least one year of professional development experience.

Procedure and Tasks

The experiment consisted of one session taking around 80 minutes per participant. A session consisted of six phases:

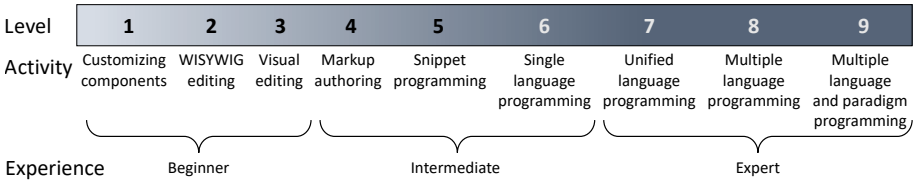


FIGURE 3.5: The nine development expertise levels [54].

1. *Consent form and demographic information.* Participants were introduced to the procedure and informed that they could leave the experiment at any time. They were then invited to sign a GDPR-compliant consent form and fill in a demographic survey with their age, highest education level, current occupation, estimated number of years of experience in the field of computer science, and estimated expertise in development based on the nine expertise levels defined by Costabile *et al.* [54] (Fig. 3.5).

2. *Introduction phase.* Participants were introduced to the application and the QuantumLeap framework. They were given a few minutes to try them out and analyze the source code of the application before continuing the experiment.

3. *Configuration phase.* Participants configured the QuantumLeap framework for the application with its configuration GUI. This task comprised seven steps, detailed in Appendix B.1.

4. *Development phase.* Participants modified the source code of the application to make it react to static and dynamic hand gestures. This task comprised 13 steps, detailed in Appendix B.2.

5. *Questionnaires.* Participants were asked to complete two questionnaires: the System Usability Scale (SUS) questionnaire [28] and the NASA-Task Load Index (NASA-TLX) questionnaire [103]. SUS comprises ten questions evaluating the usability of a system (in this case, QuantumLeap) and has been selected for its proven reliability [22] and efficient administration. NASA-TLX assesses the estimated workload of the *configuration* and *development* tasks and comprises six subscales: mental demand, physical demand, temporal demand, effort, performance, and frustration level. Each subscale is assigned a weight determined by asking the participant to select the subscale that contributes the most to the workload in 15 pairwise comparisons. In this way, each participant was able to express which subscale is perceived as the most important with respect to others, as opposed to maintaining the same weight for all subscales.

Pandian and Tuleri's NASA-TLX software [253] was used to simplify the process of data collection and analysis (Fig. 3.6).

6. *Open-ended questions and comments.* Participants answered four questions to provide insights on their overall feelings:

   - If you did not complete all the tasks, what was the reason?
   - What did you think of QuantumLeap?
   - What did you like?
   - What would you change?

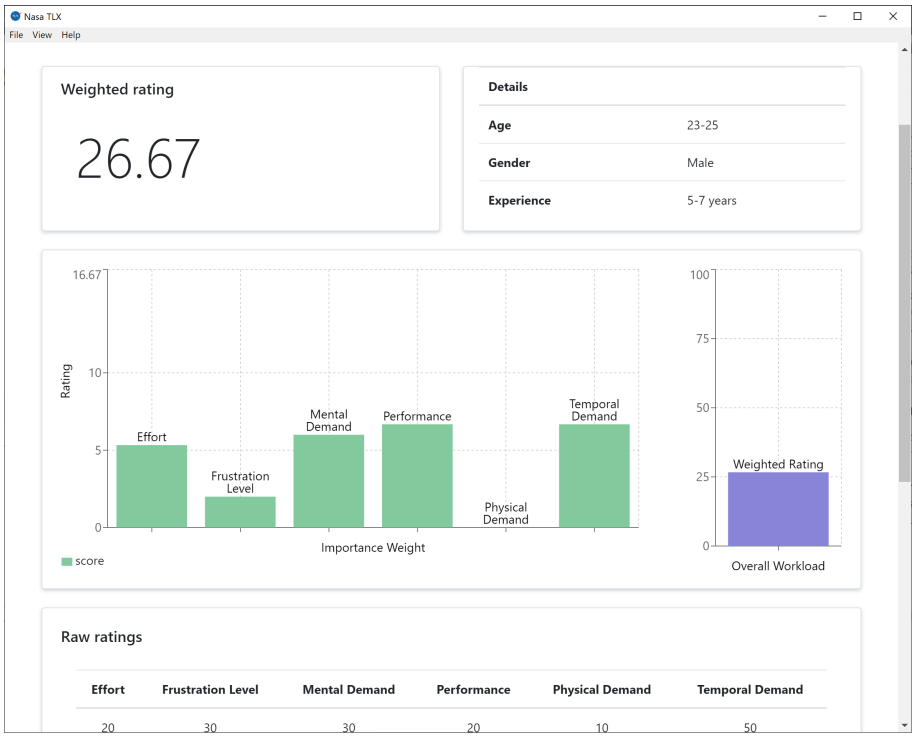


FIGURE 3.6: NASA-TLX software [253] screenshot showing results of a participant: the overall weighted rating, the rating per subscale, and the individual weights for each subscale.

Metrics and Data Collection

We collected a set of metrics for the configuration and development phases of the experiment to provide insight into the efficiency and effectiveness of QuantumLeap:

1. The TASK SUCCESS RATE is the ratio between the number of steps completed successfully by a participant and the total number of steps required to complete the procedure.
2. The TASK COMPLETION TIME measures the time taken by the participant for each of the configuration and development tasks.
3. The STEP COMPLETION TIME measures the time taken to perform each step of each of the two aforementioned tasks.

In addition to these metrics, we collected demographic information, as well as answers to the SUS questionnaire, NASA-TLX questionnaire, and open-ended questions.

3.2.2 Results and Discussion

Task Success Rate

Despite their differences in terms of years of experience and level of expertise, six of the seven participants ($\frac{6}{7}$=86%) completed all the steps of each task without any issue on the first run. Only one participant initially performed the fifth step of the *Development* task incorrectly. However, this participant immediately realized his mistake and subsequently corrected it after completing the next step, ultimately resulting in a task success rate of 100%. This suggests that the effectiveness is well covered by QuantumLeap.

Task and Step Completion Times

As a potential optimal performance, the first author of this paper (23 years, 6 years of experience, level 8) completed the *Configuration* and *Development* tasks of the experiment in 2.1 minutes and 10.2 minutes, respectively. On average, participants took 6.6 minutes to complete the first task (SD=1.2), with the slowest participant completing it in 9.1 minutes and the fastest in 5.2 minutes. For the second task, the completion time was between 22.2 and 65.1 minutes (M=37.4, SD=18.2). Fig. 3.8 shows that participants spent a more significant amount of time on the fifth step of this task (*i.e.*, adding an event listener that reacts to the "swipe left" gesture), which is normal since this is the most complex step.

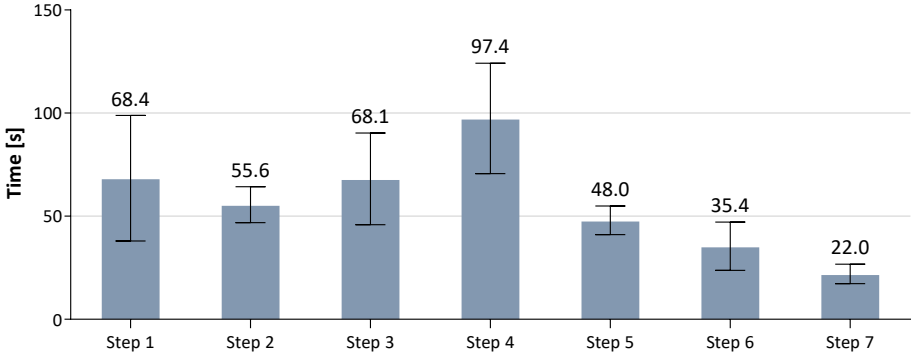


FIGURE 3.7: Average time for each step of the *Configuration* task. Error bars show a confidence interval with α=0.5.

While steps five and eight are similar, participants took seven times longer to perform the former than the latter. A plausible explanation could be that the first time they encountered the problem of reacting to a gesture, participants took some time to check the API before modifying the source code of the application. When they mastered the API, participants could then quickly complete the remaining steps.

To investigate the relationship between participants' expertise and their task completion time, we compute a Spearman's rank correlation between participants' reported level of expertise in development and their completion time for the *Configuration* and *Development* tasks. While no significant correlation was observed between the completion time of the first task and the expertise level, thus suggesting that the configuration GUI is perceived with a similar easiness regardless of the skill level, the completion time of the second task significantly correlates with expertise ($r = -.93$, p=.002**). This result was expected, as the second task of the experiment requires participants to leverage their programming skills to modify the source code of the application.

SUS Questionnaire

Since the answers to the SUS questionnaire were not normally distributed based on a Shapiro-Wilk test (*e.g.*, for question 3, W=0.45, p<.001***), we investigated the overall appreciation of the participants by conducting a one-sample Wilcoxon test comparing their answers to the midpoint of the scale (*i.e.*, 3.0). For odd-numbered, resp. for even-numbered questions, an answer is considered positive if it is significantly higher, resp. lower than the midpoint value. All odd-numbered, resp. even-numbered, questions received a mean higher, resp. lower than the midpoint of the scale, thus representing an

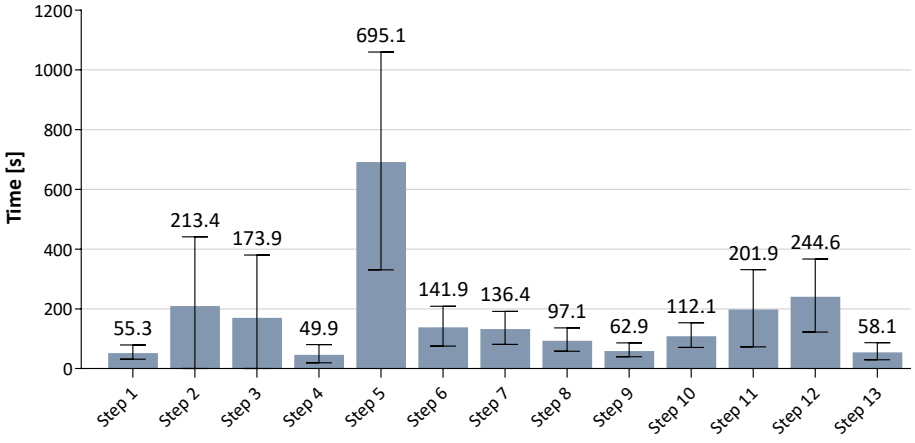


FIGURE 3.8: Average time for each step of the *Development* task. Error bars show a confidence interval with α=0.5. Note that one participant performed tasks 8, 9, and 10 in parallel. The participant's time for each of these tasks was thus inferred from the total time to perform the three tasks and the ratio between the average time for these tasks.

already positive assessment. Furthermore, six of the ten questions are statistically significantly different from the midpoint value, suggesting an even more positive answer for these questions: question 1 (W=21.0, p=.031*), question 3 (W=28.0, p=.016*), question 5 (W=21.0, p=.031*), question 6 (W=−21.0, p=.031*), question 7 (W=28.0, p=.016*), and question 10 (W=−21.0, p=.031*). As recommended by Brooke [28] and Bangor *et al.* [22], we also analyzed the SUS questionnaire as a whole. SUS scores range between 62.5 and 85.0 out of 100 (M=77.9, SD=7.8), resulting in a rating of B+ ("good") on Sauro's rating scale [248], which denotes a reasonable level of usability for such an environment. According to Lewis and Sauro [163], the overall SUS score can be further divided into two subcomponents, namely learnability and usability. Learnability is computed by summing the scores of questions 4 and 10 of the questionnaire and multiplying the result by 12.5. Usability is computed by summing the scores of the other questions and multiplying the result by 3.125. QuantumLeap obtains a mean learnability score of 78.6 and a usability score of 77.7.

To investigate the internal consistency reliability among participants, we also computed Guttman's λ_4=.91 coefficient [100] with n=2,000 iterations. This value is above $\lambda_4 \geq .90$ which represents the highest rank of consistency, usually required for high-stakes decisions like school admissions [32], thus suggesting that participants consistently answered the SUS questionnaire and

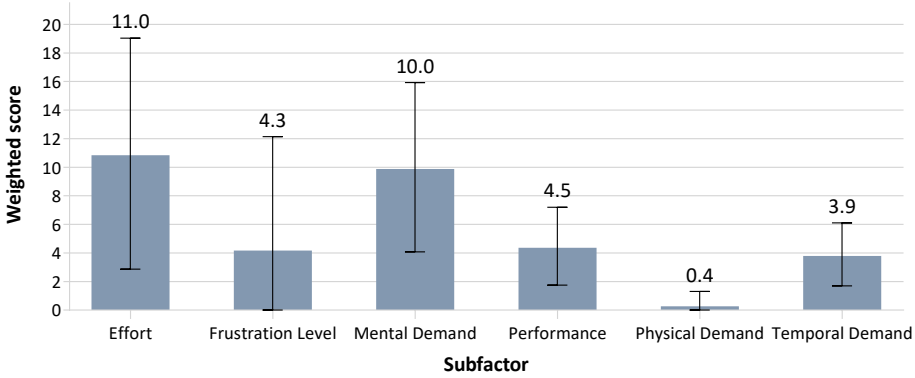


FIGURE 3.9: Average weighted score for each subscale of the NASA-TLX questionnaire. Error bars show a confidence interval with α=0.5.

that their answers were reliable enough. Guttman's λ_4 coefficient represents a good measure of reliability that produces a value higher than the usual Cronbach's α, but with a positive bias likely to be small if the estimated reliability is larger than 0.85 (which is the case here) when the sample is small (n=7). After computing the Spearman's rank correlation between participants' SUS scores and their reported level of expertise, we observe a positive correlation (r=.82, p=.033*), which suggests that the system feels more usable to experienced users than to intermediary ones.

NASA-TLX Questionnaire

Workload scores range from 8.7 to 65.3 out of 100 (M=34.0, SD=19.5). According to Grier [94], who analyzed the NASA-TLX scores of 37 computer activities, the measured workload for QuantumLeap falls into the second quartile (score lower than 54.0). Each participant managed to complete all steps of the experiment and some even reported being impressed by the results, considering that they would not have been able to do the same without the help of the framework. This could explain the low impact of the "performance subfactor". Contrarily to the SUS score, no significant correlation was observed between participants' workload scores and their reported level of expertise after computing Spearman's rank correlation.

Subscale	Score	
	M	SD
Mental Demand	10.0	6.4
Physical Demand	0.4	1.0
Temporal Demand	3.9	2.4
Effort	11.0	8.7
Performance	4.5	2.9
Frustration level	4.3	8.5
Overall	34.0	19.5

TABLE 3.2: Average weighted score and standard deviation for each subscale of the NASA-TLX questionnaire.

Open-Ended Questions

Overall, participants found that adding gesture support to an application was surprisingly quick and easy. However, some participants expressed some concern about not being able to configure the dataflow correctly without external help. Their remarks and suggestions are summarized as follows:

1. *Configuration*. Comments targeting the configuration process, the GUI, and the usability.
   - There should be some feedback to indicate to the user that the framework is starting/stopping ($\frac{6}{7}$=86%).
   - There should be some feedback to notify the user that the settings were successfully saved ($\frac{5}{7}$=71%).
   - After selecting a module, its settings should be displayed by default, instead of having to click to display them ($\frac{5}{7}$=71%).
   - A lot of steps are required to configure the dataflow ($\frac{2}{7}$=29%).
   - There should be a button to add a module, instead of just selecting the module in a drop-down menu ($\frac{2}{7}$=29%).
   - There should be a tutorial that provides guidance on dataflow configuration ($\frac{1}{7}$=14%).
   - There are some inconsistencies in the GUI ($\frac{1}{7}$=14%).
   - There should be a simple way to switch from right-handed gestures to left-handed ($\frac{1}{7}$=14%).
2. *Development*. Comments targeting the development of a gesture-based application using the QuantumLeap API.
   - There should be a method to directly attach a listener to a specific gesture, instead of having to listen to the generic "gesture" event and then check the gesture name ($\frac{2}{7}$=29%).
   - The API documentation should provide the data types of event properties ($\frac{1}{7}$=14%).

- The API documentation should clearly show the link between listeners and events ($\frac{1}{7}$=14%).
- The distinction between static and dynamic gestures may be confusing ($\frac{1}{7}$=14%).

3. *Experiment* Comments targeting the experiment itself (*e.g.*, unclear steps).
   - The order of some steps was confusing. In particular, the call to `registerGesture` should have been introduced before the call to `addListener` ($\frac{2}{7}$=29%).
   - In step three of the second task, the address and port of the running instance of QuantumLeap should have been provided ($\frac{2}{7}$=29%).

Limitations of the Study

This study is subject to two main limitations, which could influence the validity of its outcomes. First, the substantial guidance provided during the "configuration" phase of the study poses a threat to its external validity [77], as our findings may not generalize to a real-life scenario in which no such guidance is available. Second, the study would have benefited from incorporating an A/B comparison between the development of a gesture-based application with and without QuantumLeap to better highlight its impact in terms of time, effort, and the overall quality of the resulting work.

3.2.3 Changes

Following this experiment, we are planning on improving QuantumLeap based on the aforementioned participants' comments.

Short-term Changes

The most requested changes have already been implemented in the latest version of QuantumLeap. They include:

- Displaying a loading icon in the GUI when QuantumLeap is starting/stopping.
- Displaying a message in the GUI when the settings have been saved.
- Displaying the settings of modules by default in the GUI, instead of requiring the user to click on a button to display them.

Long-term Changes

We believe that these long-term changes would prove especially useful for less experienced developers:

- Add the possibility to attach listeners to specific gestures in one step, instead of requiring developers to listen to the generic "gesture" event.
- Propose a set of default configurations to allow developers to quickly prototype gesture-enabled applications without having to build a complete dataflow configuration.
- Create an interactive tutorial that guides users through the configuration of the QuantumLeap dataflow.
- Automatically configure the dataflow based on the selected gesture set and target usage. For instance, QuantumLeap could suggest a segmenter that is well-suited to the environment and determine the most accurate recognizers for the gesture set.

3.3 Sensor Integration and Applications

Through the years, additional sensors were implemented into QuantumLeap by students for use in their master's thesis, showing that QuantumLeap can be easily extended with new modules. These sensors and some of their applications are summarized in Table 3.3.

In this thesis, we used QuantumLeap to build an LMC-based multimedia application called Large User Interface (LUI, Fig. 3.10a [269]. This application, its user-centered development method, and its evaluation are described in Chapter 5. With the help of my friends and colleagues Quentin Sellier, Victor Sluÿters, and Jean-Martin Vlaeminck, we also used QuantumLeap during the Citizens of Wallonia hackathon in March 2023 to create Wall'ON (Fig. 3.10b), an interactive kiosk that provides information about works in the city in a fun and intuitive way [232].

Nothomb *et al.* [291] introduced a gesture-based DICOM image viewer (Fig. 3.10c) that allowed surgeons to navigate patient images in the operating room more hygienically and efficiently. Chauvaux and Steeman [273] integrated the Kinemic sensor, an armband that uses Inertial Measurement Units (IMUs) to track arm gestures, into QuantumLeap and built a gesture-based presentation application in which users could change slides with simple hand gestures (Fig. 3.10d). Gosselin and Moinnet [198] used QuantumLeap to build a touch-based application that enabled users to record and assign (combinations of) stroke gestures to actions in a smart house, such as turning the lights on (Fig. 3.10e).

Finally, Neuville [205], Lahousse [147], and Cornet [53] integrated three additional sensors into QuantumLeap: (1) the 3D Touchpad, a device that supports touch and 3D gestures performed above its surface, (2) the Phidget ring, which captures 3D hand motion with an accelerometer, and (3) the Myo Armband, an EMG sensor that fits around the arm. They did not develop any application but used QuantumLeap to compare various recognizer modules on their gesture set.

3.4 Discussion

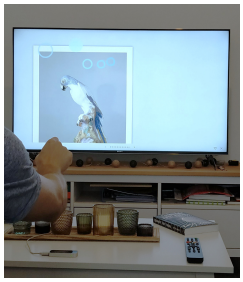
In this section, we discuss the main strengths and limitations of the QuantumLeap framework.

3.4.1 Strengths

The modular approach to gesture recognition promoted by QuantumLeap offers many advantages compared to typical non-reusable and/or opportunistic approaches. First, QuantumLeap provides a clear separation of concerns between the UI design and the gesture recognition dataflow, which greatly facilitates updates to the UI, the supported set of gestures, and the dataflow. This separation of concerns also allows developers to ignore the specifics of gesture recognition when working on the application, thus lowering the barriers to entry into gesture-based UI development. In addition, the standardized modules

Sensor(s)	Type	Application Domain	Validation	Reference
Leap Motion Controller	Vision	Multimedia	S, D, U	This work
		Healthcare	S, D	[291]
		Participatory governance	D	[232]
Kinemic	IMUs	Multimedia	S, D	[273]
3D Touchpad	Touch + proximity sensor	N/A	S	[205]
Gheran Ring Device	IMUs	N/A	S	[147]
Myo Armband	EMG	N/A	S	[53]
Touchscreen	Touch	Smart home	S, D	[198]

TABLE 3.3: Summary of the sensors included into QuantumLeap. Types of validation include system performance (S), demonstration (D), and user study (U).



(A) LUI.



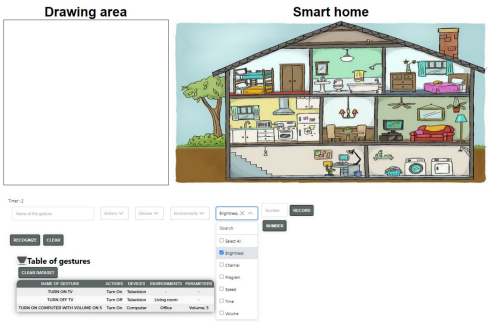
(B) Wall'ON.



(C) DICOM viewer [291].



(D) Slides application [273].



(E) Smart home application [198].

FIGURE 3.10: Examples of applications created with QuantumLeap.

and datasets supported by QuantumLeap enable researchers and practitioners to efficiently reuse them across applications, share them, and seamlessly replace one module with another in a dataflow, *e.g.*, to improve performance. Finally, the QuantumLeap GUI allows developers to visualize and change the settings of the dataflow and its modules in a few clicks without having to access and modify their source code. Dataflow configurations can be exported from the GUI and shared with other researchers and practitioners.

3.4.2 Limitations

There is, however, still plenty of room for improvement. First, the QuantumLeap GUI only provides limited guidance, in the form of short textual descriptions of modules and settings, for configuring the gesture recognition dataflow (Section 3.1.2). As such, while QuantumLeap reduces the effort required for developing gesture-based applications, some understanding of the modules (*e.g.*, gesture recognizers) is still required to build an efficient dataflow. In addition, the dataflow architecture of QuantumLeap combines modularity with

some rigidity. For instance, supporting more than one user, adding new module types (*e.g.*, a third type of gesture recognizer), or changing their order in the dataflow cannot be done easily without modifying the source code of the framework. This rigidity makes the framework more accessible to less experienced developers but may prevent some developers from experimenting with more advanced gesture recognition techniques. Finally, QuantumLeap can only support one application at a time. Switching between two gesture-based applications thus requires users to manually switch their configurations within the QuantumLeap GUI and restart the framework.

3.5 Conclusion

In this chapter, we introduced QuantumLeap, a framework for engineering gestural user interfaces based on the Leap Motion Controller by leveraging several barriers encountered in the development life cycle of gesture-based interactive applications. To facilitate the development of such LMC gesture-based UIs, manual development is replaced by configuring existing modules implemented in QuantumLeap in a dataflow way that can be saved in a configuration file for future usage. We demonstrated its usability in an experiment with seven participants of varying levels of expertise and showcased different use cases of QuantumLeap proposed by students and colleagues throughout the years. They actively contributed to the project by implementing new modules and crafting unique gesture-based applications.

We remained mindful of the four software quality properties mentioned in Section 1.2.1 throughout the design and development of QuantumLeap. Regarding *compatibility*, the standardized modules and datasets of QuantumLeap can be seamlessly shared and reused across applications. In particular, these components are interoperable with the testing tool described in Chapter 4. This interoperability would extend in the future to other applications and tools that use the same dataset format and module interfaces. From a *maintainability* perspective, updates to a module are isolated, thus minimizing the impact on other modules. The clear separation of concerns between an application and its gesture recognition logic further enables developers to modify the dataflow without altering its application code. It also provides the flexibility to transition to a different framework than QuantumLeap, should a more suitable alternating become available. In terms of *portability*, QuantumLeap configurations, modules, and datasets can be reused in other applications. Moreover, modules and datasets used in a dataflow can be replaced to support a different environment in a matter of seconds with minimal impact on applications, requiring only a quick restart of the dataflow. As for *usability*, leveraging QuantumLeap enables developers to prioritize user experience instead of spending

time to implement gesture recognition from scratch. In addition, applications built with QuantumLeap can also easily accommodate user-defined gestures, resulting in enhanced usability [203].

Future work can be envisioned at increasing levels of granularity. At the framework itself, improvements could focus on three key aspects. First, we could refine the architecture of QuantumLeap, introducing greater flexibility to accommodate more complex dataflows. Changes could include supporting gestures from multiple users simultaneously, changing the order of modules in the dataflow, or integrating custom module types. Enhancing the flexibility of QuantumLeap would also increase its complexity. As such, it will be essential to provide more guidance to developers when configuring the dataflow, *e.g.*, by automatically suggesting suitable configurations based on the selected sensor(s) and gesture set. In addition, we could enable applications to submit their dataflow configuration upon connecting to QuantumLeap, automatically downloading missing modules in the process. This feature would allow users to seamlessly switch between applications without having to manually change the configuration each time. In addition, this would enable users to replace default gestures with custom ones directly from within an application. At a finer level (*i.e.*, the framework modules), future work could focus on gesture segmentation, gesture recognition, and motion tracking sensors. We could explore novel methods for segmentation, such as Taranta *et al.*'s work [284] which trains a recognizer with automatically generated examples of parasitic gestures to distinguish them from those that are intended by the user. Equally interesting is Leiva *et al.*'s approach [162] that automatically generates synthetic gestures based on a template to train a recognizer, thus avoiding asking several participants to acquire gestures. In addition, we could study the impact of combining multiple sensors and/or gesture recognizers on recognition accuracy.

Chapter 4

Streamlining the Evaluation of Gesture Recognizers

Evaluating the performance and efficiency of gesture recognizers is an important step both in the design of gesture-based interfaces, to select the most appropriate gesture recognition algorithm for a specific application (Section 5.2.2), and in the development of new sensors, signal processing techniques, or gesture recognition algorithms, to compare them against the current state of the art or to assess how they perform in a particular context (Chapter 7). This chapter thus attempts to answer the following research question, defined in Section 1.2.1:

RQ5 *How can tools and methods aid in designing gesture-based applications that operate independently of gesture recognition logic?*

With this objective in mind, this chapter introduces the QuantumLeap testing tool, an extension of QuantumLeap (Chapter 3) that takes advantage of its standardized modules to provide a simple yet efficient way for researchers and practitioners to evaluate and compare gesture recognition algorithms. This tool facilitates the sharing of testing procedures and their results, thus opening the door for future attempts to reproduce or replicate them. As such, it was used extensively in this thesis, in some of our publications [269, 265, 266], and by master's students in their theses [205, 273, 147, 53, 90]. Fig. 4.1 illustrates the main contribution of this chapter and how it fits with the rest of this thesis.

The rest of this chapter is divided into three sections. Section 4.1 first explains how the testing tool was integrated into QuantumLeap, its main features, and how it can be used to evaluate gesture recognition algorithms. Section 4.2 then discusses its main advantages and limitations and Section 4.3 concludes this chapter.

Resources. The QuantumLeap testing tool is available on GitHub at `https://github.com/sluyters/QuantumLeap`.

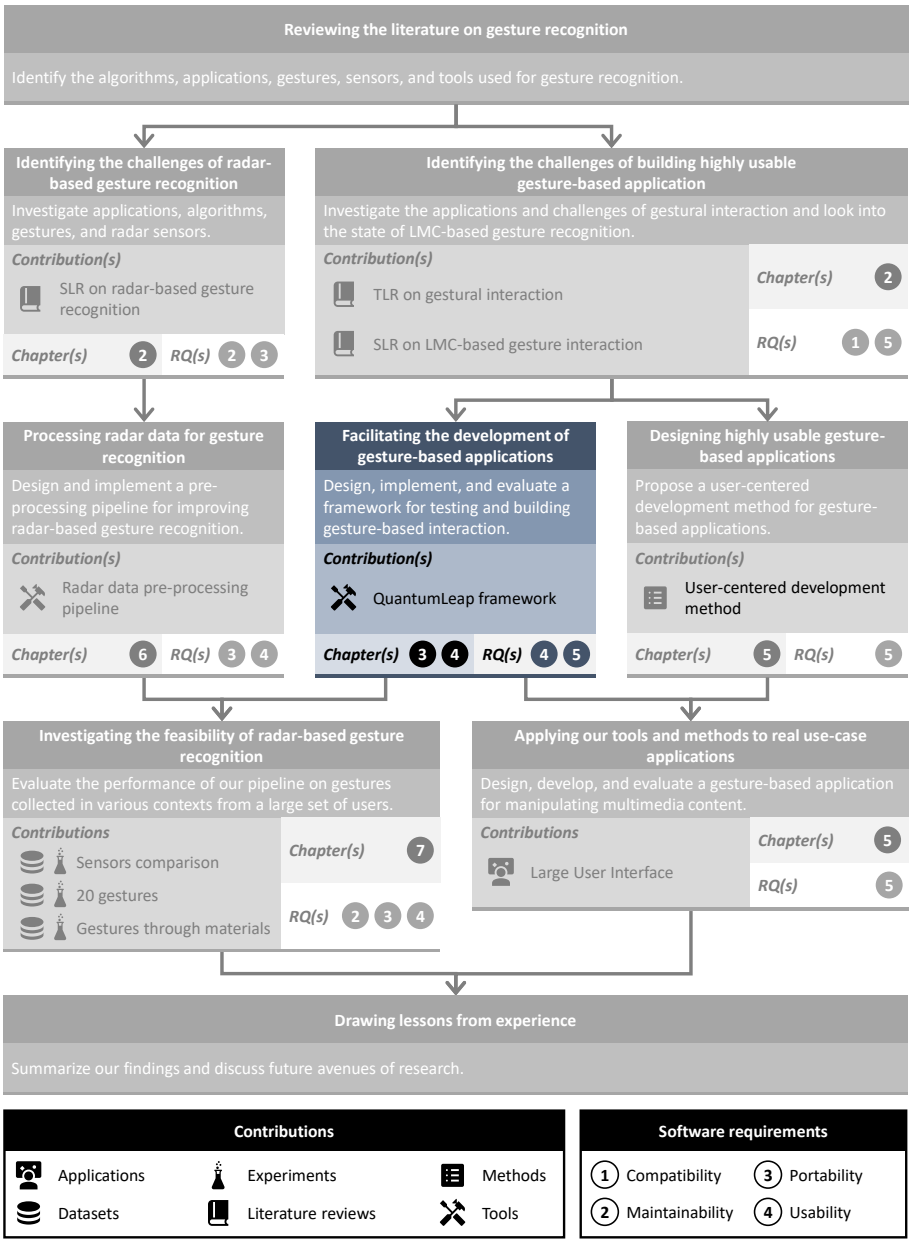


FIGURE 4.1: Main contributions of this chapter.

4.1 A Tool for Evaluating Gesture Recognition Algorithms

The QuantumLeap testing tool is based on the same architecture as the gesture recognition dataflow and is thus also written in JavaScript (ECMAScript 2019). The reuse of many QuantumLeap software components, from its backend to its frontend, vastly simplified its development and allowed this tool to work directly with all of the existing recognizer and dataset modules. Fig. 4.2 illustrates the testing tool architecture and data structures for inter-module communication are described in Table 3.1. The rest of this section discusses the integration of this tool into QuantumLeap and the various testing procedures that it supports.

4.1.1 Integration into QuantumLeap

Modules

This tool relies on the same dataset and recognizer modules as the QuantumLeap gesture recognition dataflow. However, it only supports the testing of static and dynamic recognizers. Testing other types of modules (*e.g.*, a segmenter) or combinations of modules (*e.g.*, a segmenter and a dynamic recognizer) would require modifications to its source code.

Configuration

The testing configuration was integrated into the QuantumLeap configuration UI and features two pages: one for dynamic recognizers and another for static

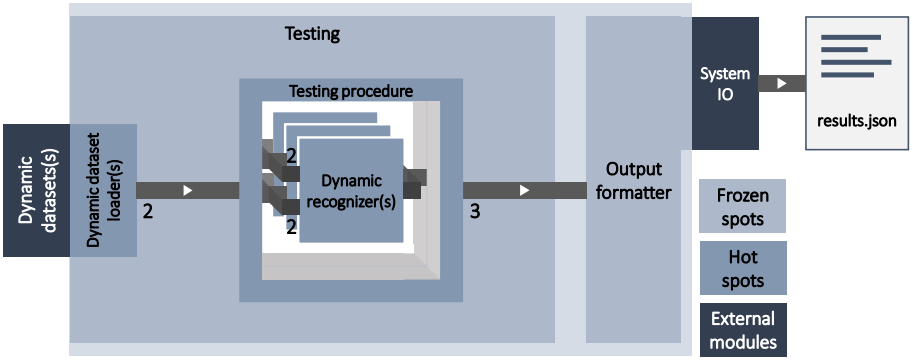
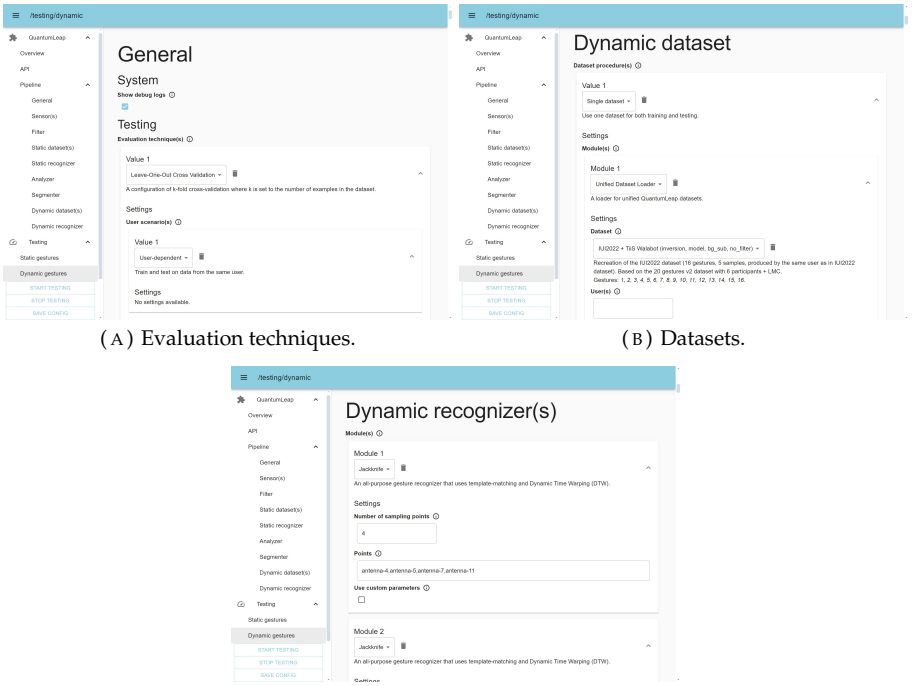


FIGURE 4.2: QuantumLeap testing tool architecture.



(A) Evaluation techniques. (B) Datasets.

(C) Recognizers.

FIGURE 4.3: Screenshots of the QuantumLeap UI for testing.

recognizers testing. For each type of recognizer, users can select and configure the evaluation technique(s), dataset(s), and recognizer(s) to test (Fig. 4.3). Testing parameters are displayed dynamically based on the "config-template.json" files of the testing and any selected module. Their assigned values are read from and saved to the "config.json" file of the testing. This file can be easily shared with, and modified by, other users so that they can attempt to reproduce or even replicate testing results. Buttons allow users to save the configuration and start the testing. The testing status (progress, time remaining), as well as potential warnings and errors, are displayed in the terminal.

Results

The *recognition rate* (*i.e.*, the ratio of correctly recognized gestures divided by the number of trials) and the *execution time* (*i.e.*, the time for recognizing the class of a candidate gesture) are computed for each tested recognizer configuration. In addition, individual trial results are summarized in a *confusion matrix* (*i.e.*, a matrix representation of the predictions from the recognizer).

Testing results are exported to JSON files, which can be parsed manually or with specialized tools to extract data and generate graphical representations. Each file features global testing parameters (*e.g.*, number of repetitions per gesture class, evaluation technique, dataset procedure) and aggregate results for each tested recognizer configuration (recognition rate, execution time, and confusion matrix). The format of the results is described below:

```
[
  {
    "r": 100,
    "evaluationTechnique": "LOOCV"
    "datasetProcedure": "singleDataset|crossDataset",
    "userScenario": "userDependent"
    "datasets": [
      "Training dataset",
      "Testing dataset (if crossDataset)"
    ],
    "gestures": ["Gesture 1", "Gesture 2"],
    "data": [
      {
        "name": "Name of the recognizer",
        "options": {
          "Option 1": "Value",
          "Option 2": "Value"
        },
        "data": [
          {
            "accuracy": 0.375,
            "time": 10.0,
            "confusionMatrix": [
              [50 50],
              [75 25]
            ]
          }
        ]
      }
    ]
  }
]
```

4.1.2 Testing procedures

QuantumLeap features ten different testing procedures for evaluating the performance of gesture recognizers. Support for new procedures may be added

relatively easily in the future should the need arise. The ten currently supported procedures are listed in Table 4.1 along with a detailed description.

Evaluation Techniques

QuantumLeap supports two evaluation techniques: (1) train-test split (TTS), in which testing and training samples are randomly selected from the dataset(s) and (2) leave-one-out cross-validation (LOOCV), an extreme case of k-fold cross-validation (with k=1) in which recognizers are systematically tested with each sample of the testing dataset while training with all other valid samples of the training dataset [363].

Dataset Procedures

Two dataset procedures are also supported: (1) single-dataset, in which training and testing samples are selected from the same dataset, and (2) cross-dataset, where one dataset is used for training and the other for testing.

User Scenarios

Three user scenarios are supported by QuantumLeap: (1) user-dependent (UD), in which testing and training samples are produced by the same user [297], (2) user-independent (UI), in which the testing sample is produced by a different user than the training samples [297], and (3) mixed, in which testing and training samples may or may not be produced by the same user. The UD and UI scenarios are supported by both TTS and LOOCV, while only LOOCV supports the "mixed" scenario.

4.2 Discussion

4.2.1 Strengths

The QuantumLeap testing tool has multiple advantages over using custom, opportunistic scripts for evaluating gesture recognition algorithms. It allows developers and practitioners to use the same modules in an evaluation and in a final application, which is more efficient and reduces the risk of errors when switching from testing to implementation. It also proposes ten predefined evaluation protocols and saves the results in a standard format, that can be parsed to produce figures and extract relevant measures. Its GUI enables users to select one or more modules to evaluate and configure them without having to modify their source code. Comparing the performance of multiple gesture

Technique	**Users**	**Dataset(s)**	**Procedure description** For each combination of independent variables [...]
TTS	UD	Single	For each gesture class in the dataset, repeat R times: (1) select a random sample of that class for testing, (2) train the recognizer with N samples per class randomly selected from the same user, and (3) attempt recognition.
		Cross	For each gesture class in the testing set, repeat R times: (1) select a random sample of that class for testing, (2) train the recognizer with N samples per class randomly selected from the same user in the training set, and (3) attempt recognition.
	UI	Single	For each gesture class in the dataset, repeat R times: (1) select a random sample of that class for testing, (2) train the recognizer with N samples per class randomly selected from different users than the user of the testing samples, and (3) attempt recognition.
		Cross	For each gesture class in the testing set, repeat R times: (1) select a random sample of that class for testing, (2) train the recognizer with N samples per class randomly selected from different users than the user of the testing samples in the training set, and (3) attempt recognition.
LOOCV	UD	Single	For each gesture sample in the dataset: (1) select the sample for testing, (2) use all other samples from the same user for training, and (3) attempt recognition.
		Cross	For each gesture sample in the testing set: (1) select the sample for testing, (2) train the recognizer with all samples in the training set from the same user, and (3) attempt recognition.
	UI	Single	For each gesture sample in the dataset: (1) select the sample for testing, (2) train the recognizer with all other samples from other users, and (3) attempt recognition.
		Cross	For each gesture sample in the testing set: (1) select the sample for testing, (2) train the recognizer with all samples in the training set from other users, and (3) attempt recognition.
	Mixed	Single	For each gesture sample in the dataset: (1) select the sample for testing, 2) train the recognizer with all other samples, and (3) attempt recognition.
		Cross	For each gesture sample in the testing dataset: (1) select the sample for testing, (2) train the recognizer with all samples in the training set, and (3) attempt recognition.

TABLE 4.1: Testing procedures implemented in QuantumLeap.

recognizers in the same scenario thus becomes a painless and extremely efficient process. Modules, testing configurations, and results can be shared with other researchers, providing them with all the necessary tools to attempt to repeat, replicate, and reproduce these results. By encouraging cooperation between researchers, this tool could foster the development of better algorithms for gesture recognition.

4.2.2 Limitations

However, this testing tool suffers from a few limitations which could be addressed in the future. For instance, evaluating a more complex gesture recognition pipeline (*e.g.*, a combination of filter, segmenter, and recognizer), while very relevant, is currently impossible without modifying the source code of the tool. In addition, this tool is aimed at evaluating template-matching algorithms written in JavaScript. This currently limits its impact on the HCI community, although supporting more languages and other types of gesture recognition algorithms (*e.g.*, based on deep learning) may be possible in the future. Finally, testing results are aggregated before being exported to a file, with no way to access individual trials. While these aggregated results may be easier to parse, they may not provide sufficient granularity for more in-depth analyses.

4.3 Conclusion

In this chapter, we introduced the QuantumLeap testing tool, an extension of QuantumLeap (Chapter 3) that utilizes the same modules and datasets for evaluating the performance of gesture recognizers. It supports ten testing procedures and features standardized configuration files and results, allowing researchers to share all of the elements required to repeat, replicate, and reproduce their experiments.

Like QuantumLeap, this tool addresses the four software quality properties [116] mentioned in Section 1.2.1: compatibility, maintainability, portability, and usability. As mentioned in Section 3.5, QuantumLeap and its testing tool support the *compatibility* property by relying on the same modules and datasets. In terms of *maintainability*, the QuantumLeap testing tool helps developers adapt their applications to different environments (sensor, gesture set), by making it easy to compare gesture recognizers and select the most suitable one for a specific environment. Regarding *portability*, QuantumLeap testing configurations, modules, and datasets can be shared by researchers so that other algorithms can be evaluated in the same conditions, thus contributing to the advancement of research on gesture recognition techniques. In addition,

a testing procedure can be applied to many different modules in a matter of seconds without any modification to the source code. With regards to *usability*, the ability to evaluate and compare modules before employing them in an application enables developers to build more accurate gesture recognition, which can result in more usable applications. Overall, our extensive use of the tool throughout the rest of this thesis highlights the benefits of designing software that satisfies these quality properties.

Despite its advantages, the QuantumLeap testing tool still has limitations that could be addressed in future work. One avenue for improvement involves supporting the evaluation of more complex gesture recognition dataflows, beyond gesture recognizers, such as combinations of filters, segmenters, and recognizers. In addition, we could work on improving the granularity of the results output by the tool, to enable more in-depth analyses. Finally, supporting other languages than JavaScript would vastly expand the possibilities of the tool, *e.g.*, by allowing researchers to evaluate deep learning algorithms written in Python or R.

Chapter 5

Designing Highly Usable Gesture-based Applications

Effective gesture-based interfaces should feature a concise set of gestures that are consistent across the application and align with user expectations, as hand gestures can be challenging to learn [84], produce [297], and remember [203]. However, as we concluded in Section 2.1.4, designing such interfaces is a complex process that often confronts user preferences with the technical challenges of gesture recognition. Active involvement of designers and end users throughout the whole development process is thus paramount to achieving an optimal outcome, *e.g.*, by collecting user preferences through GESs (Section 2.1.1) and by conducting usability evaluations. In an attempt to lower the barriers to entry for the development of intuitive gestural interfaces, this chapter addresses one of the five research questions defined in Section 1.2.1:

RQ5 *How can tools and methods aid in designing gesture-based applications that operate independently of gesture recognition logic?*

To do so, we propose a user-centered iterative development method and evaluation protocol for gesture-based interfaces, which aims to support the development of highly usable gestural interfaces. We apply this method to LUI, a gesture-based application for browsing and manipulating multimedia content. Fig. 4.1 illustrates these contributions and how they fit within this thesis.

The rest of this chapter is organized as follows. Section 5.1 first overviews the LUI application, its concepts, software architecture, and design requirements, and summarizes the differences between its three iterations. Section 5.2 then details the stages of the development method used for LUI, including the design of a consistent gesture set, and the use of QuantumLeap for testing and implementing gesture interaction. Section 5.3 evaluates the overall quality of the interface, as well as the memorability and efficacy of the implemented gestures. Finally, Section 5.4 discusses the main strengths and limitations of LUI and Section 5.5 concludes this chapter.

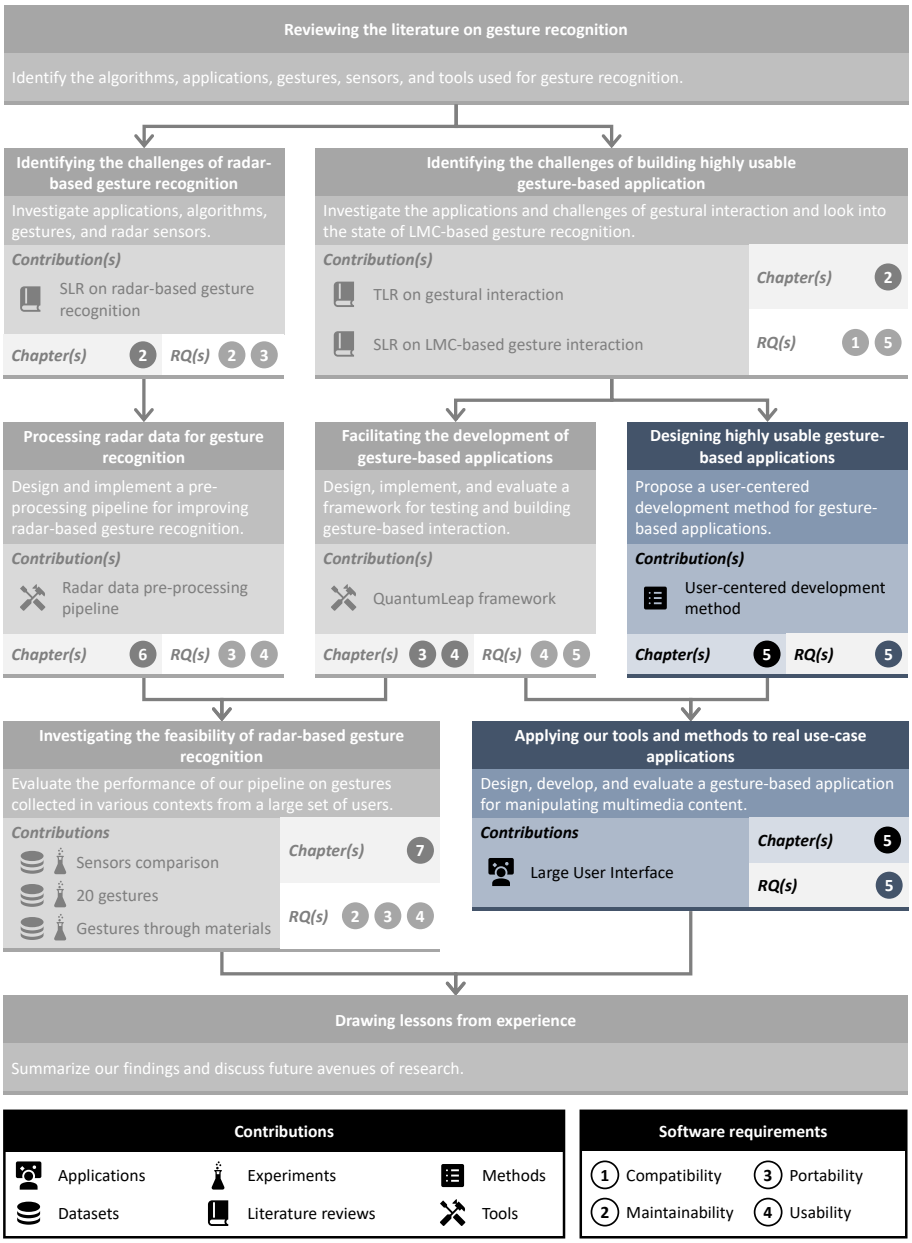


FIGURE 5.1: Main contributions of this chapter.

Publications. This chapter is based on two papers published in the IJHCI journal [269] and IJHCS journal [252].

Resources. The LUI application source code and its QuantumLeap configuration are available on GitHub at `https://github.com/sluyters/LUI`. Demonstration videos of LUI are available at `https://www.youtube.com/playlist?list=PLj1KcEqIa0fGRb0Z5E7f0UoL4LaxiKbm4`.

5.1 The LUI Application

5.1.1 Overview

LUI is a gesture-based interactive application for manipulating multimedia objects on large displays, such as TVs and wall displays, in a semi-public environment. It currently focuses on interaction with photos, videos, documents, and maps, for which different sets of functions are desired (Table 5.1). LUI is designed to be widely distributed and easily installed by being cheap to set up and easy to use. As such, we settled on using the Leap Motion Controller (LMC)[1] for hand tracking, a cheap off-the-shelf sensor (less than 100$) that can be plugged via USB to the computer. We selected the LMC for mid-air gesture interaction because it tracks two hands and their ten fingers [49] with high precision and robustness [324], including the palm vector and hand radius. This sensor is comprised of two cameras and three infrared (IR) LEDs. The interaction space is .6m wide by .6m long by .6m deep, resulting in an .22 cubic meter volumetric space which takes the shape of an inverted pyramid above the sensor. The LMC API [272] enables to natively recognize a handful of system-defined gestures [27], but custom gesture recognition algorithms often support a wider range of gestures [21]. The LMC benefits from better accuracy and precision than other devices, such as an MS Kinect [101] or an Intel RealSense [125]. It is also widely used in many domains of applications such as augmented reality [182] and activity recognition [190]. Several LMC-based gesture recognition algorithms exist [21]. No study compares them on the same basis, apart from [82] with three classical machine learning algorithms, but without any recent recognizer such as DeepGru [185].

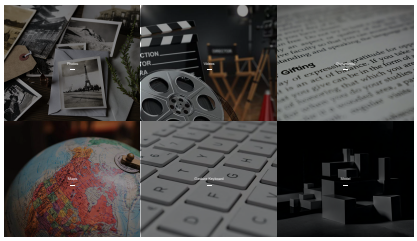
5.1.2 Requirements

To address the quality of the gesture interaction, LUI should fulfill three requirements:

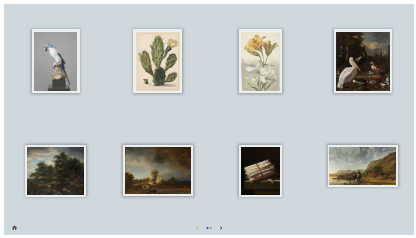
[1] See `https://www.ultraleap.com/product/leap-motion-controller/`



(A) Welcome screen.



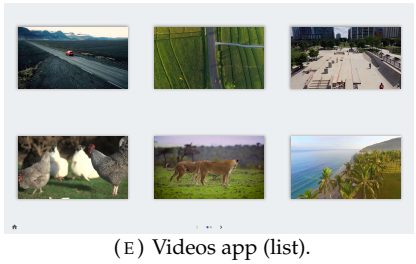
(B) Application menu.



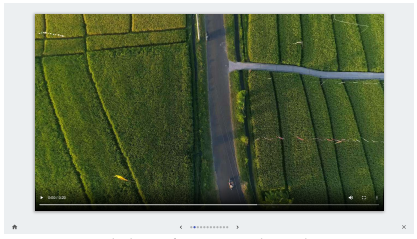
(C) Photos app (list).



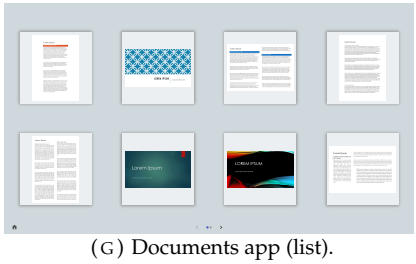
(D) Photos app (item).



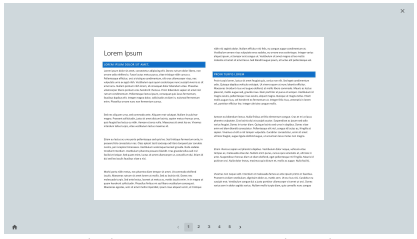
(E) Videos app (list).



(F) Videos app (item).



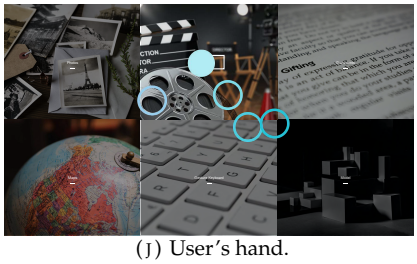
(G) Documents app (list).



(H) Documents app (item).



(I) Maps app.



(J) User's hand.

FIGURE 5.2: Screenshots of the third prototype of the LUI application.

Function	Media type			
	Photos	Videos	Documents	Maps
Fullscreen on/off	●	●	●	●
Like/dislike	●	●	●	●
Previous/Next	●	●	●	○
Zoom in/out	●	○	●	●
Rotate	●	○	●	●
Pan	●	○	●	●
Change brightness	●	○	○	○
Change contrast	●	○	○	○
Play/pause	○	●	○	○
Show/hide subtitles	○	●	○	○
Volume up/down	○	●	○	○
Fast-forward/backward	○	●	○	○
Highlight text	○	○	●	○
Place marker	○	○	○	●
Change layer	○	○	○	●

TABLE 5.1: List of functions for each media type.

1. *Consistency.* Similar gestures should be assigned consistently to similar functions, independently of the media type, to foster knowledge transfer [64], to maximize gesture memorability, and to reduce the learning curve [63]. Distinguishable gestures should be assigned to other specific functions [133].
2. *Continuity.* End users should be able to perform gestures continuously, without needing to notify the system when they are performing a gesture, *i.e.*, the system should continuously recognize gestures and distinguish intended gestures from parasitic motion with little to no user intervention. End users should be able to manipulate media objects directly with a reasonable system response time (*e.g.*, 1 sec for direct manipulation according to [208]) and continuous feedback to the user [7] (*e.g.*, gradually zooming in a picture by increasing the distance between the index and the thumb). Whether a gesture is discrete for a 0D/1D function or continuous for a 2D/3D function, this should not induce any difference for the end user.
3. *Customizability.* The system should support users in including their own gestures without requiring any modification to the LUI core [55]. User-defined gesture sets [95] usually result in higher satisfaction and memorability than system- or designer-defined gestures [203]. Some gesture recognizers perform better when the gestures used for training are recorded by the target user of the system [297].

5.1.3 Iterative Design

The LUI application went through two iterations before the final one described in this chapter. This section summarizes the three iterations of LUI and its gesture set (Fig. 5.3).

First Iteration (v1)

In this first prototype [224], the basics of LUI were laid out. Users could navigate through lists of photos and videos, switch to full-screen mode, and play, pause, or change the volume of a video. Gesture recognition was developed opportunistically based on the LMC API [272], which made adding new functions or gestures to LUI a complex endeavor and resulted in a rather small set of gestures. End users were thus forced to use the predefined gestures, with no possibility of assigning their own gestures to specific functions.

Second Iteration (v2)

A usability testing was conducted with 20 participants to evaluate their subjective satisfaction regarding the first prototype [224, 70]. It highlighted multiple issues, including a high learning curve, unpredictable gesture recognition, and fatigue, in particular for the *splay* gesture, which participants found especially difficult to perform and reproduce accurately. Some participants expressed their intention to define their own preferred gestures, instead of using the predefined ones. This second prototype replaced the *splay* gesture with the *flick*

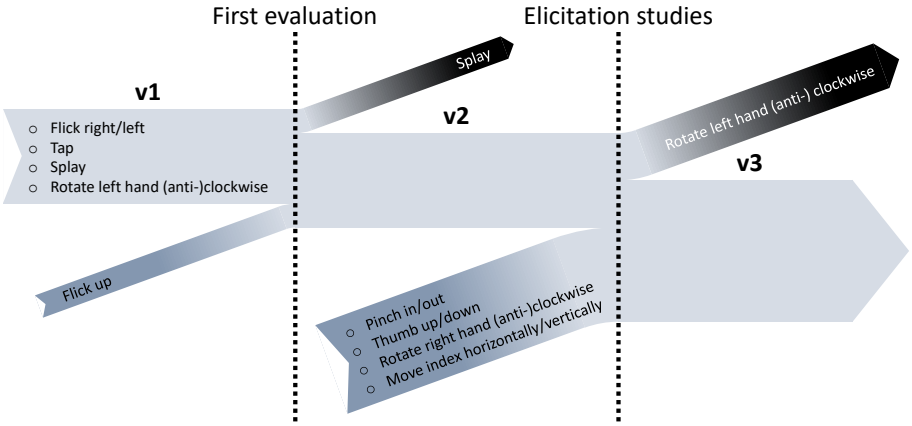


FIGURE 5.3: Evolution of the gesture set during the iterative design.

up gesture, which is easier to perform. Other issues highlighted by the testing were left unaddressed in this version, as they required more substantial modifications to the LUI core.

Third Iteration (v3)

The current version of LUI was developed based on the assets of the first two prototypes but features significant changes to its gesture set and gesture recognition pipeline (Section 5.2). It now relies on the QuantumLeap pipeline for gesture recognition, which enables end users to add their own preferred gestures to predefined ones without having to modify the LUI core. The use of QuantumLeap also vastly simplified the process of extending LUI with new applications and gestures, which enabled us to add new functions to the Photos and Videos applications (*e.g.*, rotating a photo and browsing through a video), and implement new Documents and Maps applications.

5.2 User-centered Development Method

This section details the three-stage development process that we followed for the third iteration of LUI. We first designed an intuitive set of gestures, consistent across the entire LUI application, based on the results of a GES. Then, we ran a comparative testing of static and dynamic gesture recognizers using the QuantumLeap testing tool (Chapter 4) to identify a suitable dataflow configuration for this gesture set. Finally, we instantiated a gesture recognition dataflow based on the results of our testing using the QuantumLeap framework (Chapter 3).

5.2.1 Gesture Set Design

Gesture Identification

As the number of functions increased, it was decided to conduct a GES [329] to explore a consensus set of consistent gestures that would optimize agreement among end users. Based on the classical method [329] and some extensions [88], we initially conducted two GESs (see Appendix C.1): one for photo browsing and another for video browsing. Prior work [13, 96] pointed out that gesture sets for different devices could be inconsistent. Our two studies confirm this observation as they led to inconsistent gestures tailored to one media type at a time. Rather than conducting a separate GES for each media type, we conducted a single GES (see Appendix C.2) integrating the various media with both generic and specific functions.

The result of our GES consists of a characterization of users' gesture input behaviors with valuable information for designers, practitioners, and end users regarding the consensus among participants (which is computed as an *agreement* and *co-agreement rate* [301]), most frequent (thus, generalizable across users) gestures for a specified task, and insights into users' conceptual models for performing tasks. [64] computed the agreement score [329] for their multi-device GES. Instead, we computed the agreement rate $AR(r)$ [301], which is smaller and more restrictive than the agreement score $A(r)$, as follows:

$$A(r) = \sum_{P_i \subseteq P} \left(\frac{|P_i|}{|P|} \right)^2 \geq AR(r) = \frac{\sum_{P_i \subseteq P} \frac{1}{2}|P_i| \, (|P_i|-1)}{\frac{1}{2}|P| \, (|P|-1)} \tag{5.1}$$

where r denotes the referent to elicit a gesture, $|P|$ denotes the number of elicited gestures, and $|P_i|$ denotes the number of gestures elicited for the i^{th} subgroup of P.

Table C.3 summarizes the two most agreed gestures for each referent of the GES. From these results, we observe that the majority ($\frac{16}{18}$=88.9%) of the most agreed gestures were one-handed. One-handed gestures let users interact with the system while holding an object in the other hand (*e.g.*, a glass of water or a paper document). In addition, the low occurrence of two-handed gestures in our study may be explained by a lower ease of execution compared to one-handed gestures [201]. Some referents, such as enable/disable subtitles, have a very low agreement rate. For these referents, user-elicited gestures may not be the most appropriate, as the low consensus may result in lower guessability and memorability [328]. Instead, it may be preferable to favor other solutions, such as prompting users to assign their own gesture to these actions at the start of the application (*i.e.*, user-defined gestures [203]), and/or providing an indirect way of triggering these actions (*e.g.*, by navigating through a menu with highly agreed gestures). For referents with a low to medium agreement rate, such as play and pause, aliasing (*i.e.*, assign multiple gestures, such as the most and second most agreed gestures, to one referent) may greatly increase guessability [328, 329]. This would enable users to use the gesture they are most familiar with, and thus flatten the learning curve and increase the memorability of the gesture set. However, attentive care should be given to avoid overloading gestures.

Gestures Selection

The LUI gesture set (Fig. 5.4 and Table 5.2) resulting from the contextual GES consists of gestures selected following four guidelines inspired by the literature [201, 328, 329, 64]:

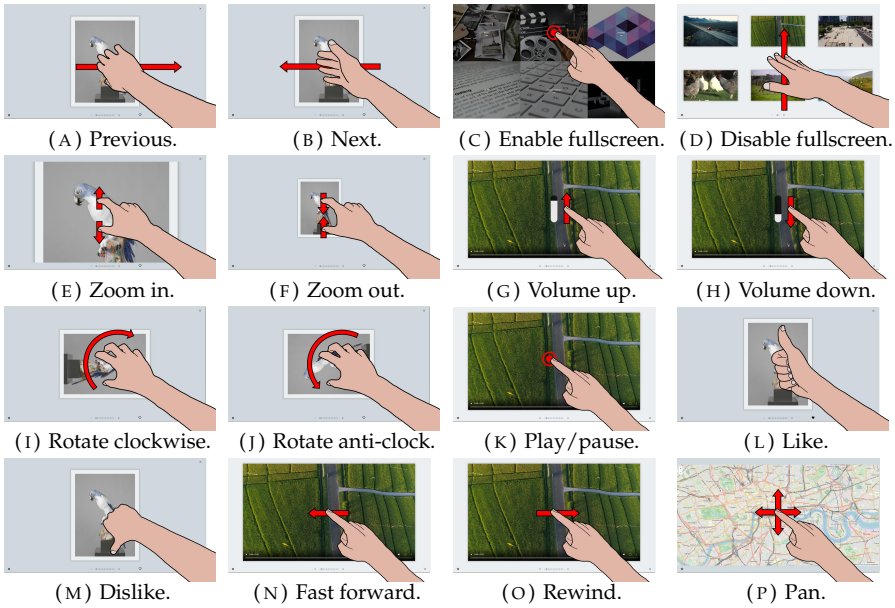


FIGURE 5.4: Illustrations of gestures mapped to each function in the LUI application.

1. Favor the most agreed gestures, as this usually results in a flatter learning curve and higher memorability [328, 329, 64]. For instance, we selected "flick left" instead of "flick right" for the "next" function, as the former was proposed by more participants.
2. Prefer gestures that are less physically and mentally demanding to limit frustration and fatigue [201]. For example, we favored one-handed over two-handed gestures.
3. Map similar gestures to similar functions, independently of the media to increase memorability and flatten the learning curve [64]. For instance, we selected the same gesture for zooming in a photo/in a map and used symmetrical gestures for going to the previous/next item in a list.
4. Map unique gestures to different functions occurring in the same context to avoid conflicts where one gesture would trigger multiple functions [328, 64].

Together, these guidelines enabled us to build a conflict-free, highly guessable, and memorable gesture set that minimizes end users' mental and physical effort. The "pan" gesture was added later as part of the Maps application but was not part of the elicitation study.

Gesture Classification

After designing a consistent and memorable gesture set, the next step was to determine how to efficiently recognize these gestures. As such, we first attempted to classify each gesture according to Aigner *et al.*'s HCI-oriented gesture taxonomy [7]. While several multidisciplinary taxonomies exist in general for classifying gestures, including Kendon's taxonomy [123] and McNeil's taxonomy [194], we rely on Aigner's classification as it identifies gesture classes suitable to the specific field of HCI by empirically validating 5,500 collected gestures on interactive functions without accompanying them with speech. Indeed, [74] characterize human gestures according to three fundamental considerations, *i.e.*, origin, usage, and coding, and structure them into idiosyncratic, interactive, informative, and communicative classes. Kendon [123] structures gestures along a continuum based on speech: gesticulation (beat, cohesives), language-like (iconic), pantomimes, emblems (deictic), and sign language (symbolic). McNeill [194] structures gestures into iconics, metaphorics, beats, cohesives, and deictics. By analyzing each gesture and its effect on the application, this step helped us identify the most appropriate configuration of QuantumLeap in the next section. Table 5.2 summarizes the results of our classification and details how each gesture should be recognized.

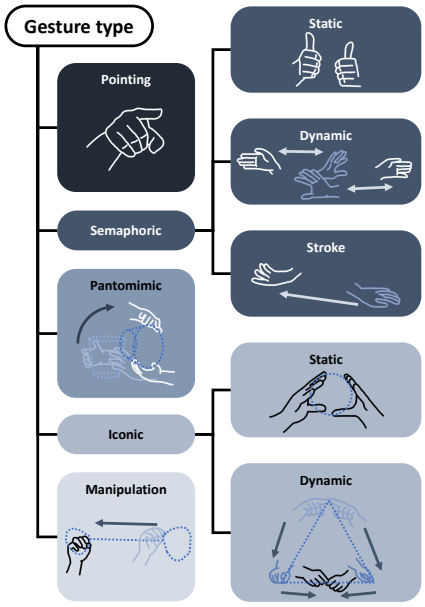


FIGURE 5.5: Aigner *et al.*'s taxonomy [7].

	Elicitation		Recognition	
Action	Gesture	Category	Name	Type
Previous	Flick right	Stroke semaphoric	Flick right	Dynamic
Next	Flick left	Stroke semaphoric	Flick left	Dynamic
Enable fullscreen	Tap	Pointing	Tap	Dynamic
Disable fullscreen	Flick up	Stroke semaphoric	Flick up	Dynamic
Zoom in	Pinch out	Manipulation	Pinch	Static
Zoom out	Pinch in	Manipulation	Pinch	Static
Volume up	Swipe up	Manipulation	Point index	Static
Volume down	Swipe down	Manipulation	Point index	Static
Rotate clockwise	Rotate a knob clockwise	Manipulation	Grab	Static
Rotate anti-clockwise	Rotate a knob anti-clockwise	Manipulation	Grab	Static
Play	Tap	Stroke semaphoric	Tap	Dynamic
Pause	Tap	Stroke semaphoric	Tap	Dynamic
Like	Thumbs up	Static semaphoric	Thumb	Static
Dislike	Thumbs down	Static semaphoric	Thumb	Static
Fast-forward	Swipe left	Manipulation	Point index	Static
Rewind	Swipe right	Manipulation	Point index	Static
Pan	–	–	Point index	Static

TABLE 5.2: Classification of LUI gestures according to Aigner *et al.*'s taxonomy [7].

Eight of the 16 elicited gestures are classified in the *manipulation* category, which are gestures for manipulating media objects in a short feedback loop. The gesture effect on the object is perceived in real-time while the user is performing it, with low latency [7] (*e.g.*, a response time of 0.1 sec to perceive interaction as direct manipulation and 1 sec to keep it seamless [208]). Other authors estimate the response time as follows: [42] situates the best compromise around 500 ms with a recognition rate above 90%, while [12] report that end users perceive a response time of 50 ms for inking tasks on a touch-sensitive surface. As an example of direct manipulation, the size of an image should change proportionally to the thumb-index distance while the user is performing a *pinch in* or *pinch out* gesture. For these gestures, the hand pose conveys the meaning of the gesture. They can thus be recognized by analyzing the user's hand pose in each frame in real-time. Six gestures are considered as *stroke semaphoric* and two are considered as *static semaphoric*. *Semaphoric gestures* are hand poses and movements used to convey a specific meaning, often unrelated to the gesture. They can be further divided into three categories, including *static* (if the meaning is entirely conveyed by the hand pose) and *strokes* (which consist of single stroke hand motions) [7]. The *stroke semaphoric* gestures could be recognized by analyzing the hand trajectory over time, while

the *static semaphoric* gestures could be recognized by analyzing the user's hand pose. One gesture belongs to the *pointing* category, *i.e.*, gestures which indicate objects and directions [7]. Indeed, to enable fullscreen, the *tap* gesture acts as a selection method, as users first need to hover their finger above the selected item before performing the gesture. This gesture could be recognized similarly to the *stroke semaphoric* gestures.

5.2.2 Software Architecture and Implementation

We used the QuantumLeap framework (Chapter 3) to implement gesture recognition into LUI, instead of developing a pipeline from scratch. Using the QuantumLeap API, gestures were assigned effects on the UI to be triggered each time they were recognized. For instance, the *pinch* static gesture rotates a picture depending on the angle of the user's hand, and the *tap* dynamic gesture opens an application, displays content in fullscreen, or plays/pauses a video, depending on the context. The use of QuantumLeap has multiple benefits. For instance, changes to the gesture set or updates to the gesture recognition pipeline do not require any modification to the LUI source code. In addition, the implementation of gesture recognition into LUI was extremely quick, as we could just select the most appropriate modules among those already supported by the framework. In the rest of this section, we motivate our choice of modules by conducting a comparative testing of static and dynamic gesture recognizers (see [186] for a survey in 2D) and discussing the use of a sliding window for gesture segmentation.

Static Gestures

Two static gesture recognizers were tested on a custom gesture set containing six types of poses (grab, pinch, point index, flat hand, thumb, and rest) acquired from one participant. Each pose was performed in various orientations and scales, resulting in more than 1000 templates per class. The study is within-factors with three independent variables:

Recognizer		**Gesture class**						**Global accuracy**		**Time**
Name	A	Flat	Point	Pinch	Grab	Thumb	Idle	Mean	SD	
$\$P^3$+	6	99.7%	100.0%	100.0%	100.0%	100.0%	100.0%	99.9%	0.1%	0.15ms
$\$P^3$+	16	100.0%	99.9%	99.7%	100.0%	100.0%	100.0%	99.9%	0.1%	0.46ms
GPSDa	6	99.6%	99.0%	98.3%	99.9%	98.2%	100.0%	99.2%	0.8%	0.04ms
GPSDa	16	99.6%	99.8%	99.4%	100.0%	99.5%	100.0%	99.7%	0.3%	0.22ms

TABLE 5.3: Recognition rate of each static recognizer, for T=128. Accuracy higher than 98% is highlighted in blue.

- `Recognizer`: nominal variable with two conditions, representing the static recognizers tested (Appendix A.4): GPSDa, a custom recognizer with α=0.65) and $P^3+, a implementation of Vatavu's $P+ [295] that supports 3D gestures.

- `Number of Joints`: numerical variable with two conditions, representing the number of joints used by the recognizers: $A \in \{6, 16\}$.

- `Number of Templates`: numerical variable with eight conditions, representing the number of gesture templates used to train the recognizers: $T \in \{1, 2, 4, 8, 16, 32, 64, 128\}$.

The testing was conducted using the QuantumLeap testing tool (Chapter 4) following a train-test-split (TTS) procedure with 1000 repetitions in a user-dependent and single-dataset scenario (see Table 4.1 for a detailed description of the procedure).

This experiment shows that GPSDa exhibits a different behavior than $P^3+ (Table 5.3): GPSDa performs better with a larger number of joints, while the opposite occurs for $P^3+. GPSDa performs better than $P^3+ with a small number of templates (*e.g.*, with 4 templates, GPSDa (A=16) reaches 91.0% accuracy against 87.1% for $P^3+ ($A$=6)), but $P^3+ starts performing better as the number of templates increases. Both recognizers achieved $>99\%$ accuracy with 128 training templates. Despite $P^3+ being very slightly better in the benchmark, GPSDa was selected for the LUI application as its rotation invariance gives it an edge in real-world situations (*e.g.*, when the user's hand is rotated relative to the training samples).

Dynamic Gestures

Similarly to the static gesture recognizers, eight dynamic gesture recognizers were tested on the LUI gesture set to determine the best-suited recognizer. The gesture set contains four types of pre-segmented dynamic gestures, *i.e.*, *flick left*, *flick right*, *flick up*, and *tap*, acquired from four participants. Gestures were performed four times by each participant, giving 16 templates per class. The study is within-factors with three independent variables:

1. `Recognizer`: nominal variable with five conditions, representing the dynamic recognizers tested: 3 cent [36], P^3 (adapted from $P [299]), $P^3+ (adapted from $P+ [295]), Q^3 (adapted from $Q [298]), and Jackknife [284]. The recognizers are detailed in Appendix A.7.

2. `Number of Joints`: numerical variable with three conditions, representing the number of joints taken into account by the recognizers: $A \in \{1, 6, 16\}$.

Recognizer		Gesture class				Global accuracy		Time
Name	A	Flick up	Flick left	Flick right	Tap	Mean	SD	
3¢	1	100.0%	97.2%	100.0%	100.0%	99.3%	1.4%	0.78ms
$P^3	1	100.0%	67.7%	70.1%	100.0%	84.5%	18.0%	0.33ms
$P^3	6	94.3%	91.5%	80.3%	99.9%	91.5%	8.2%	0.38ms
$P^3	16	91.9%	99.9%	90.8%	100.0%	95.7%	5.0%	0.28ms
$P^3+	1	100.0%	58.9%	53.5%	100.0%	78.1%	25.4%	0.05ms
$P^3+	6	99.9%	85.8%	78.1%	100.0%	91.0%	10.9%	0.06ms
$P^3+	16	100.0%	94.2%	83.0%	100.0%	94.3%	8.0%	0.10ms
$Q^3	1	100.0%	65.9%	36.1%	100.0%	75.5%	30.8%	0.91ms
$Q^3	6	100.0%	82.1%	71.6%	100.0%	88.4%	14.0%	1.03ms
$Q^3	16	99.2%	90.8%	75.2%	100.0%	91.3%	11.5%	0.97ms
Jackknife	1	100.0%	100.0%	100.0%	100.0%	100.0%	0.0%	0.04ms
Jackknife	6	100.0%	100.0%	100.0%	100.0%	100.0%	0.0%	0.10ms
Jackknife	16	100.0%	100.0%	100.0%	100.0%	100.0%	0.0%	0.19ms

TABLE 5.4: Recognition rate of each dynamic recognizer, for T=8. Recognizers with a global accuracy higher than 98% are highlighted in blue.

3. `Number of Templates`: numerical variable with four conditions, representing the number of gesture templates used to train the recognizers: $T \in \{1, 2, 4, 8\}$.

The testing procedure was similar to the one used for static recognizers, but each gesture template was resampled to 16 points.

Because the dataset includes only four simple gestures, the recognition rates are very high for most recognizers even with very few training templates (Table 5.4, T=8). Jackknife already reaches 100% with one training template and accuracy for all recognizers generally increases with the number of training templates. Jackknife (A=1) was selected as the dynamic recognizer for LUI, as it maximizes accuracy and minimizes the execution time on this dataset.

Gesture Segmentation

A simple sliding window was selected for gesture segmentation. Sliding windows are easy to implement, fast to execute, require no specific user input, and have been combined with recognizers such as Jackknife [284] and 3¢ [34]. However, they are sensitive to parasitic gestures, which may hinder the user experience. To address this issue, it was combined with two other techniques: (1) a motion rejection threshold to reject small hand motions such as hand tremors, and (2) a rejection threshold based on the score returned by the recognizers, to reject poorly performed gestures. In the future, more advanced segmentation techniques [137, 40, 283] could be incorporated into a new module for QuantumLeap without requiring any code modification to LUI.

Final Configuration

Fig. 5.6 shows the final QuantumLeap pipeline configuration for the LUI application. The Window segmenter and Jackknife recognizer identify users' hand gestures from the continuous flow of data. The Basic analyzer and GPSDa recognizer identify static hand gestures and allow the manipulation of images, videos, and documents with continuous feedback.

In addition to simplifying future modifications to the LUI gesture recognition pipeline, the use of QuantumLeap reduced the size of its source code by around 1280 lines of code (LOCs), distributed as follows:

- 101 LOCs to connect to the LMC and extract data in the correct format.
- 110 LOCs to load the dynamic gesture dataset.
- 63 LOCs to load the static gesture dataset.
- 143 LOCs for the GPSDa static recognizer.
- 60 LOCs for the analyzer.
- 78 LOCs for the window segmenter.
- 725 LOCs for the JackKnife dynamic gesture recognizer.

If we factor out the time required to conceive, design, and develop an adequate pipeline for gesture recognition from scratch, and to integrate it into a working environment, the benefits of QuantumLeap in terms of time-saving become more obvious than in terms of LOCs saving.

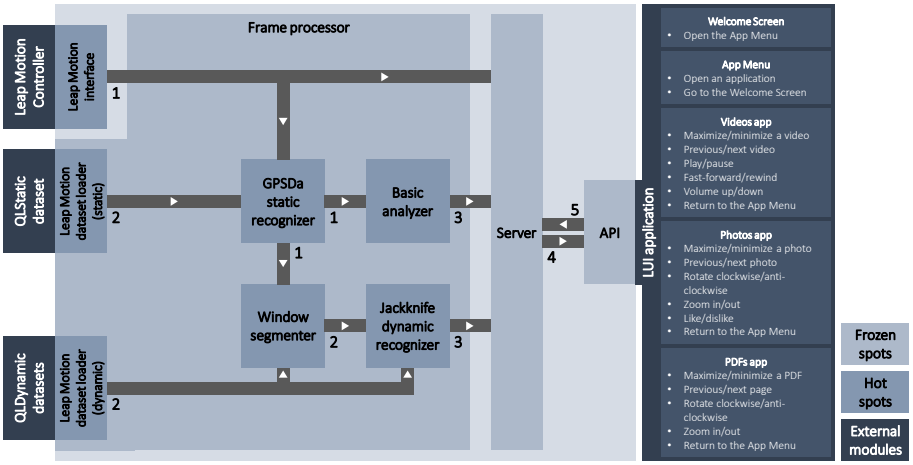


FIGURE 5.6: Instantiation of the QuantumLeap framework for LUI.

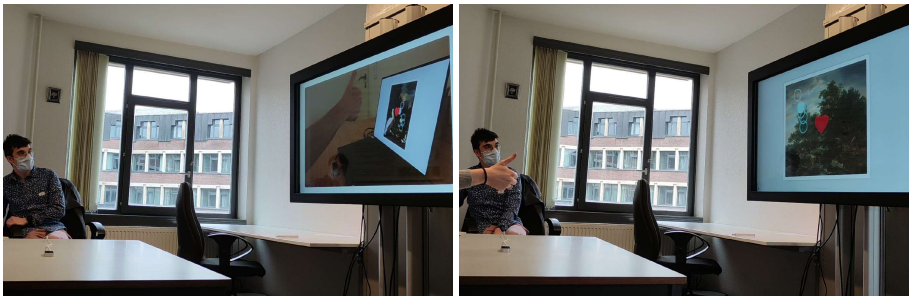
5.3 Evaluation

This section reports the results of an experiment performed on the third LUI version, according to a procedure inspired by [307].

5.3.1 Experiment Protocol

Environment

The experiment involved multiple devices, including a large TV screen for displaying the LUI application and tutorial videos (Fig. 5.7), an LMC for capturing the participants' gestures, and a laptop for running the LUI application and controlling the experiment with a custom software. A camera was positioned to record both the participants' hands and the TV screen. The LUI application was configured to record the logs of the gestures recognized by the system. Participants were installed at a desk and faced a large screen situated around two meters on the opposite side of the desk. The LMC was positioned in front of them.



(A) Tutorial video playing on the screen. (B) Participant performing a gesture.

FIGURE 5.7: The setup of the experiment.

Participants

A stratified sampling of 17 participants (8 females and 9 males), aged between 22 and 65 years old (M=37.5, SD=15.3 years) was randomly selected from a list of volunteers to balance between genders, backgrounds, and ages (Table 5.5). All participants were right-handed and all of them used smartphones daily. All but one participant ($\frac{16}{17}$=94.1%) used computers at least once a week, and only two participants ($\frac{2}{17}$=11.8%) frequently used gestural interaction. Occupations included student, researcher, employee, computer scientist, physiotherapist, and accountant.

Gender	Female				Male			
Background	Technical		Non-technical		Technical		Non-technical	
Age [years]	≤ 30	> 30	≤ 30	> 30	≤ 30	> 30	≤ 30	> 30
#participants	2	2	2	2	2	2	2	3

TABLE 5.5: Distribution of the participants in our study.

Id	Task	Target(s)	Gestures
A1	Previous	Photo, video, document, page	Flick right
A2	Next	Photo, video, document, page	Flick left
A3	Enable fullscreen	Photo, video, document, app	Tap
A4	Disable fullscreen	Photo, video, document, app	Flick up
A5	Zoom in	Photo, document, map	Pinch out
A6	Zoom out	Photo, document, map	Pinch in
A7	Volume up	Video	Swipe up
A8	Volume down	Video	Swipe down
A9	Rotate clockwise	Photo, document	Rotate a knob clockwise
A10	Rotate anti-clockwise	Photo, document	Rotate a knob anti-clockwise
A11	Play	Video	Tap
A12	Pause	Video	Tap
A13	Like	Photo	Thumbs up
A14	Dislike	Photo	Thumbs down
A15	Fast-forward	Video	Swipe left
A16	Rewind	Video	Swipe right
A17	Pan	Map	Move hand while pointing index

TABLE 5.6: The 17 atomic tasks, target content, and associated gestures.

Procedure and Tasks

The experiment consisted of two sessions conducted on separate days. The interval between the first and the second sessions of a participant ranged from 3 to 14 days (M=7, SD=2.2 days).

Session 1 (Discovery and Short-term Memorability). The first session was aimed at introducing LUI to participants and evaluating its user experience and the short-term memorability of its gesture set. It consisted of six phases:

1. *Consent form and demographic information.* The participants were introduced to the procedure and informed that they could choose to leave the experiment at any time. They were then invited to sign a GDPR-compliant consent form and fill in a demographic survey with information such as their age, gender, level of education, occupation, digital

Id	Task	Instructions	Required actions
C1	Maps	In the Maps app, move and enlarge the map so that the two blue markers are situated at the left and right sides of the screen.	Enable fullscreen, zoom in, pan
C2	Photos	In the Photos app, manipulate the windmill painting so that the author's name is the right way up and large enough to be easily readable.	Enable fullscreen, next, zoom in, rotate clockwise/anticlockwise
C3	Videos	In the Videos app, find the words pronounced halfway through the "sea turtles" video. Fast-forward through the video to get to the right part.	Enable fullscreen, next, play, fast-forward

TABLE 5.7: The three compound tasks and the minimum set of actions required to complete each task.

skills [236], and previous experience with technologies such as smartphones, computers, and gestural interaction.

2. *Discovery phase.* Participants interacted freely with the LUI application for three minutes, starting from the app menu. They did not receive any specific instructions or information about the interface, except that it was controllable using the movements of the right hand above the sensor, without touching any surface. This phase served as a first introduction of LUI to the participants.

3. *Learning phase.* This phase served to teach the gestures to the participants and let them familiarize themselves with the system. The participants were given 17 tasks in a random order (Table 5.6). Each task was read aloud by the experimenter, who then played a short tutorial video of the associated gesture (Fig. 5.7a). The use of videos ensured that all participants received the same instructions. Once the tutorial finished playing, the experimenter loaded the associated page of the LUI application and prompted the participants to perform the gesture in order to complete the task (Fig. 5.7b). Participants could re-watch the tutorial videos as many times as necessary. If a participant accidentally left the current page (*e.g.*, by performing the wrong gesture), the experimenter informed them and re-loaded the correct page.

4. *Resting phase.* Participants were given five minutes to rest in a separate room.

5. *Testing phase.* This phase aimed to evaluate the short-term memorability of the gesture set by asking participants to perform 20 tasks with no external help. The participants first completed 17 atomic tasks (Table 5.6) in a random order, following a similar procedure to the learning phase: the experimenter first read the task aloud, loaded the associated page

of the LUI application, and prompted the participants to perform the gesture. The target content was randomly selected for each task (*e.g.*, for the "zoom in" task, the target could be a picture, a document, or a map) and the tasks were grouped by target content. The participants then performed three compound tasks (Table 5.7). For each compound task, the experimenter read the task aloud, loaded the application menu, and prompted the participants to perform the task. Participants could request the experimenter to repeat the instructions or to go back to the application menu.

Participants were instructed to fill in at the end of each session a UEQ+ questionnaire (User Experience Questionnaire) [251], a modular extension of UEQ in which we selected 7 scales (*i.e.*, attractiveness, efficiency, perspicuity, dependability, stimulation, usefulness, and intuitive use) among 20 scales to focus on evaluating the user experience of participants interacting by gestures with LUI. Each scale is, in turn, decomposed into four subscales to be evaluated (*e.g.*, attractiveness is decomposed into four subscales: annoying *vs.* enjoyable, bad *vs.* good, unpleasant *vs.* pleasant, and unfriendly *vs.* friendly), each subscale being a differential scale with 7 points between items of each pair (*e.g.*, annoying o o o o o o o enjoyable).

6. *Questionnaire.* Participants were asked to complete a UEQ+ questionnaire (User Experience Questionnaire) [251], a modular extension of UEQ. We selected 7 UEQ+ scales (*i.e.*, attractiveness, efficiency, perspicuity, dependability, stimulation, usefulness, and intuitive use) among the 20 available scales to focus on evaluating the user experience of participants interacting by gestures with LUI. Each scale consists of four subscales to be evaluated (*e.g.*, attractiveness is decomposed into four subscales: annoying *vs.* enjoyable, bad *vs.* good, unpleasant *vs.* pleasant, and unfriendly *vs.* friendly).

Session 2 (Long-term Memorability). The second session was much shorter and consisted of only three phases:

1. *Introduction.* Participants were welcomed and reminded of the procedure, in particular, that they could leave the experiment at any time.

2. *Testing phase.* This second testing phase served to evaluate the long-term memorability of the LUI gesture set. We followed the same procedure as in the testing phase of the first session.

3. *Questionnaire.* At the end of the session, participants were asked to fill in the same UEQ+ questionnaire as in the first session.

Metrics and Data Collection

We collected a set of metrics for the learning and testing phases of both sessions to give us insight into the performance and usability of the system:

1. The `Task Success Rate` measures the percentage between the number of successful tasks (*i.e.*, the gesture is correctly remembered, produced, and recognized) and the total number of executed tasks. An atomic task was considered successful if it was correctly executed within a maximum of four trials. Regarding compound tasks, the number of trials was not limited since they required participants to perform several gestures in an undefined order. We thus considered a compound task unsuccessful only if it was skipped by the participant.

2. The `Number of Trials` measures the total number of gestures executed for each task, whether successful, incorrectly recalled, incorrectly executed, or incorrectly recognized by the system.

In addition to the metrics, demographic information, and questionnaires, we captured video footage of the discovery, learning, and testing phases.

Id	Task	Learning phase	Testing phase 1	Testing phase 2
A1	Previous	100%	100%	100%
A2	Next	100%	94%	94%
A3	Enable fullscreen	88%	82%	100%
A4	Disable fullscreen	76%	88%	100%
A5	Zoom in	100%	88%	94%
A6	Zoom out	94%	94%	82%
A7	Volume up	94%	88%	82%
A8	Volume down	82%	100%	94%
A9	Rotate clockwise	71%	71%	71%
A10	Rotate anti-clockwise	82%	82%	71%
A11	Play	82%	88%	100%
A12	Pause	94%	88%	82%
A13	Like	100%	100%	100%
A14	Dislike	100%	100%	94%
A15	Fast-forward	88%	88%	88%
A16	Rewind	82%	82%	71%
A17	Pan	88%	88%	88%
C1	Photo	–	100%	94%
C2	Video	–	82%	94%
C3	Map	–	76%	65%

TABLE 5.8: Success rate for the 17 atomic and 3 compound tasks.

5.3.2 Results and Discussion

Task Success Rate

Table 5.8 gives the success rate for the atomic and compound tasks. Only two atomic tasks have a success rate below 80% in the learning phase: tasks A4 (disable full screen) and A9 (rotate clockwise). In the first testing phase, only task A9 has a success rate below 80%, while in the second testing phase, the overall success rate decreases slightly, and three tasks have a success rate lower than 80%: tasks A9, A10 (rotate anticlockwise), and A16 (rewind). Overall, the lower success rate of the rotating (anti-)clockwise tasks may be explained by the high sensitivity of the system to the users' hand pose. For instance, if the users' fingers were too far apart when performing the "grab" gesture, it was not recognized by the system. To a lesser extent, this high sensitivity may have affected the recognition of other static gestures (see Table 5.2), such as in the "fast-forward", "rewind", and "pan" tasks.

Regarding compound tasks, tasks C1 (photo) and C2 (video) had a higher success rate than task C3 (map) in both the first and second testing phases. The "pan" gesture used in the Maps application was not elicited, which may have impacted the usability of the application. In addition, this task required more precision than the other two, as it required participants to accurately manipulate the map so that two markers were situated on each side of the screen.

Id	Task	Learning phase		Testing phase 1		Testing phase 2	
		Mean	Median	Mean	Median	Mean	Median
A1	Previous	1,2	1	1,6	1	1,7	1
A2	Next	1,1	1	1,3	1	1,6	2
A3	Enable fullscreen	1,5	1	1,5	1	1,1	1
A4	Disable fullscreen	1,2	1	1,7	2	1,5	1
A5	Zoom in	1,4	1	1,3	1	1,5	1
A6	Zoom out	1,4	1	1,1	1	1,2	1
A7	Volume up	1,3	1	1,3	1	1,1	1
A8	Volume down	2,0	2	1,5	1	1,3	1
A9	Rotate clockwise	2,1	1,5	1,6	1,5	1,9	1,5
A10	Rotate anti-clockwise	1,8	1,5	2,1	1,5	2,4	2,5
A11	Play	1,3	1	2,0	1	1,6	1
A12	Pause	1,4	1	1,5	1	1,8	1,5
A13	Like	1,0	1	1,2	1	1,3	1
A14	Dislike	1,0	1	1,1	1	1,0	1
A15	Fast-forward	1,7	1	1,7	1	1,9	2
A16	Rewind	1,4	1	1,7	1	1,4	1
A17	Pan	1,3	1	1,3	1	1,5	1

TABLE 5.9: Number of trials for successful completion of each atomic task.

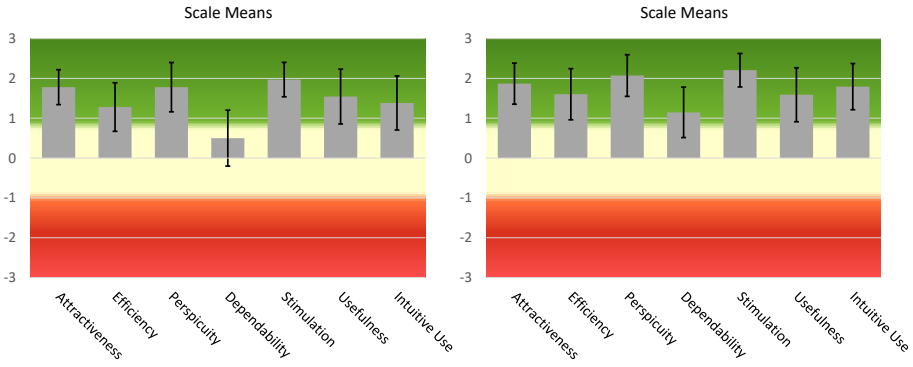
Number of Trials

Table 5.9 gives the average and median number of tries needed to successfully complete the atomic tasks. These metrics are presented for the learning session and the two testing sessions, allowing in particular to study the memorability of gestures.

As presented, most tasks were completed with a single try. The most problematic tasks were the rotations, whether clockwise or anticlockwise. We can also observe that more attempts were needed to pass the tasks in session 2, especially for A1 (previous), A2 (next), A12 (pause), and A15 (fast forward). On the contrary, the tasks of increasing or decreasing the volume (A7 and A8) required fewer tries over time.

Results of the UEQ+ Questionnaire

Users' answers to the 7 selected UEQ+ scales were computed and interpreted with the UEQ data analysis tool. Fig. 5.8 represents the mean score for each selected scale on an interval [-3,...,+3], where a value of -3 expresses the weakest value and +3 expresses the strongest value for both sessions (first in Fig. 5.8a and second in Fig. 5.8b). If we stick to the interpretation that a value greater than or equal to 0.8 represents a positive evaluation [251], then all mean scores for session 1 are considered positive (ranging from M=1.28 for efficiency to M=1.97 for stimulation), except dependability (M=0.50, SD=2.22) considered as a neutral value. This low value can be explained by the cognitive destabilization of participants producing these kinds of gestures for the first time, and being puzzled by them. However, this phenomenon was observed transiently insofar as the gestures, initially felt to be difficult to produce because



(A) Session 1. (B) Session 2.

FIGURE 5.8: Mean score for each scale of the UEQ+ questionnaire.

they were unknown, then became easier to produce. Indeed, dependability increased by 130% (M=1.15, SD=1.80) in session 2. Mean scores for session 2 are all superior to those obtained for session 1 and range from M=1.15 for dependability to M=2.21 for stimulation. The mean score for some scales even largely increased, such as efficiency (from M=1.28 to M=1.60), perspicuity (from M=1.78 to M=2.07), and intuitive use (from M=1.38 to M=1.79). The KPI (Key Performance Indicator) computed by UEQ+ [110] increased from 1.44 for session 1 to 1.74 for session 2, confirming the overall positive evolution.

5.4 Discussion

In this section, we discuss the main strengths and limitations of the LUI application, which we designed following a user-centered iterative development method.

5.4.1 Strengths

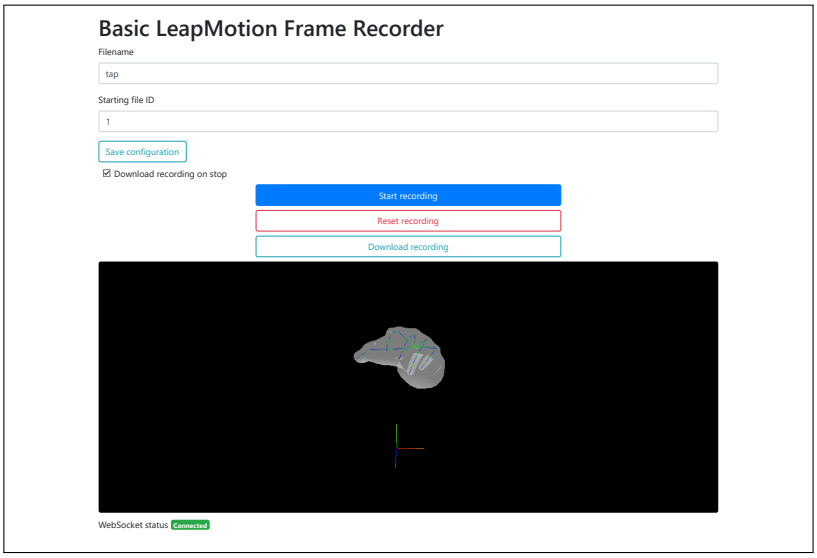
The main strengths of LUI lie in the three requirements defined in Section 5.1.2, which we now revisit in the light of this work:

1. *Consistency.* The gesture set of LUI was designed to be consistent across the various types of media contents, by assigning similar gestures to similar functions, regardless of the media type. In addition, it is expected to be compliant with end users' expectations as the gesture set was based on a contextual gesture elicitation study.

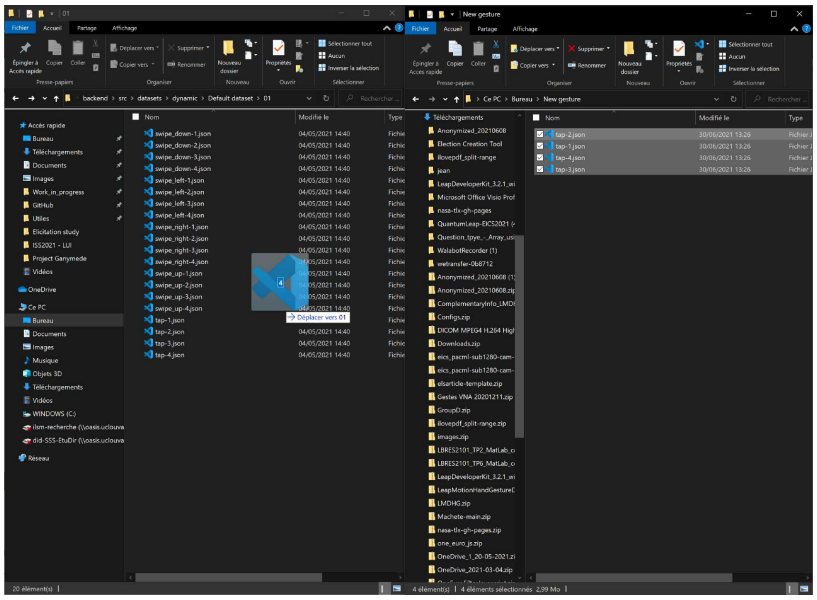
2. *Continuity.* The LUI application supports continuous interaction with media contents thanks to QuantumLeap. The segmentation module allows the continuous recognition of gestures while they are being issued, without input from the user, and the static recognizer enables the manipulation of media objects with immediate and continuous feedback.

3. *Customizability.* Thanks to QuantumLeap, the LUI application supports customizability by enabling users with no coding skills to include their own gestures in three steps (Fig. 5.9): (1) record new gesture samples using custom software, *e.g.*, our LeapGesturePlayback tool[2] (Fig. 5.9a), (2) overwrite the old gesture files with the new samples (Fig. 5.9b), and (3) restart QuantumLeap to propagate the changes, which takes less than ten seconds on a modern computer.

[2]The code of the gesture recording tool is available on GitHub at `https://github.com/sluyters/LeapGesturePlayback`.



(A) Recording new gesture samples.



(B) Replacing old gesture files with new samples.

FIGURE 5.9: Two main steps for customizing the LUI gesture set.

5.4.2 Limitations

The LUI application has a few limitations, some of which arise from its use of the QuantumLeap framework. First, while utilizing simple template-matching recognizers allows users to quickly and easily customize the LUI gesture set with their own gestures, employing properly trained state-of-the-art machine learning recognizers might accommodate a broader range of gestures and enable better gesture recognition accuracy. Moreover, taking into account modifications to the gesture set requires a restart of QuantumLeap. Although this process is swift thanks to the use of template-matching recognizers [29], as opposed to neural network-based recognizers which may require extensive retraining before recognizing the updated gesture set [29], the need for a restart could be avoided altogether. In addition, the customization process could be further streamlined and simplified by integrating it into the LUI application, enabling users to modify the gesture set without leaving the application. Furthermore, gesture segmentation is not perfect, and parasitic motion may be mistaken for a gesture intended by the user, despite the workarounds that we implemented (*e.g.*, rejection thresholds). Hence, users must remain careful while interacting with the application to avoid triggering actions inadvertently. Lastly, LUI does not support concurrent interaction from multiple users due to limitations of QuantumLeap. As a result, concurrent gestures from two users will interfere with each other.

Other limitations of LUI are related to our development method, specifically, the design of the gesture set and the proposed evaluation procedure. First, user fatigue may become a problem during long interactions with LUI, as mid-air gestures can be more tiring than traditional interaction methods [256, 118, 102] (Section 2.1.2). Minimizing the physical effort of mid-air gestures is critical to supporting prolonged gesture interaction [102]. Then, while user-elicited gesture sets have been shown to improve the immediate usability of an application [328], the GES procedure followed in this chapter is not without flaws. Indeed, the small sample size of our study may also not accurately represent the majority of the population, as gesture propositions are affected by participants' cultural background [15]. Our choice of illustrations for the referents featured notable differences from the final LUI application and may have influenced participants' gesture propositions. In addition, GESs usually rely on subjective criteria for gesture clustering, which may further impact the validity of their findings [296]. Fortunately, the customizability of LUI contributes to mitigating these issues by enabling end users to define their own set of gestures. Finally, our evaluation procedure focused on collecting quantitative data. While this approach is a great way to evaluate the usability of an application, it provides little insight into the actual reasons behind user challenges and potential avenues for improvement. Collecting additional

qualitative data, *e.g.*, by incorporating thinking-aloud techniques [271] into our evaluation procedure, would provide us with a more comprehensive understanding of users' thoughts and reasoning.

5.5 Conclusion

In this chapter, we introduced an iterative and participatory process for designing gesture-based applications, comprising three stages: (1) designing an intuitive and consistent gesture set based on user preferences identified from the literature or through a GES, (2) implementing a gesture recognition dataflow and identifying the most optimal algorithms for gesture recognition, and (3) evaluating the final gestural interface with potential end users of the system. We applied this process in the development of LUI, a gesture-based application designed for browsing and manipulating multimedia content (images, videos, documents, and maps) on large displays. The use of the QuantumLeap framework (Chapter 3) and its testing tool (Chapter 4) greatly facilitated the implementation of consistent, continuous, and customizable mid-air gesture interaction into LUI. After evaluating LUI in an experiment with 17 participants, we noted an increase in its perceived usability between the two experimental sessions, despite a few participants having trouble remembering some of the gestures. We also observed some unpredictability in gesture recognition for certain gestures, attributed in part to slight variations in the way they were performed by the participants. This underscores the importance of supporting gesture set customization.

The LUI application inherits multiple ISO/IEC 25010 software quality properties [116] from its use of QuantumLeap and its testing tool (Chapters 3 and 4), including *maintainability* and *portability*. Our methodology also supports the *usability* property, by encouraging developers to involve end users in the development process of their application, by conducting GESs to identify user preferences, and by evaluating their application among potential users.

This chapter opens up promising avenues for future work. First, the current iteration of LUI relies exclusively on single-handed gestures, which aligns with user preferences observed in our GES. However, because these preferences may change depending on the sensor and context of use, it would be relevant to implement and evaluate the performance of the system on a gesture set of two-handed, or even whole-body gestures. In addition, we focused on a single-user scenario where one person controls the system and others are spectators. Future work could investigate multiuser scenarios of two types: (1) a collaborative scenario, where multiple users cooperate on the same screen simultaneously, and (2) a "talking stick" scenario where control can switch between users (*e.g.*, in a classroom or an office meeting). Implementing these

scenarios might involve one device capable of sensing multiple users concurrently (*e.g.*, MS Kinect) or multiple devices, each capturing gestures from one or more users (*e.g.*, one LMC in front of each seat at a conference table). Next, we could look into incorporating speech recognition into LUI, to complement mid-air gestures by providing a simple way to execute complex commands, such as searching for specific content by keywords. Finally, if we seek to deploy the system on public displays, several challenges will certainly arise, including dealing with users' privacy concerns and a less forgiving environment. As such, we could investigate other types of sensors, such as radars, that address most privacy issues related to vision-based sensors and are less sensitive to environmental conditions. UI elements would also have to be carefully designed to attract users, retain their attention, and effectively communicate the supported gestures.

Chapter 6

Addressing the Challenges of Radar-based Gesture Recognition

Radar sensors provide a unique alternative to other input devices (*e.g.*, vision-based sensors, IMUs) for gesture recognition. They combine a low sensitivity to lighting conditions, an ability to see through surfaces, and user privacy preservation with a small form factor and low power consumption. However, radar signals can be noisy, complex to analyze due to their high dimensionality [184], and usually do not transpose well from one radar to another. Our SLR on radar-based gesture interaction (Section 2.3) highlighted that most analyzed papers relied on deep learning techniques tailored to a specific radar and set of gestures. While these highly tailored solutions can achieve extremely accurate gesture recognition, they are difficult to adapt to other use cases (*e.g.*, sensors, gestures, and environments) and hinder the ability of end users to customize applications by defining their own set of gestures. Consequently, few real applications have been developed, despite the remarkable potential of radar-based systems for gesture interaction. In this context, this chapter addresses two of the five research questions defined in Section 1.2.1:

RQ3 *What types of gesture-based applications would benefit from radar sensors?*

RQ4 *How can we foster collaboration between researchers and practitioners working on (radar) gesture recognition?*

To this end, we introduce a signal processing pipeline that employs full-wave electromagnetic modeling and inversion [149, 148] and satisfies three properties to address the main challenges of radar-based gesture recognition: (1) radar system invariance, *i.e.*, the output signal is normalized to become independent from the radar, (2) background scene invariance, *i.e.*, the output signal

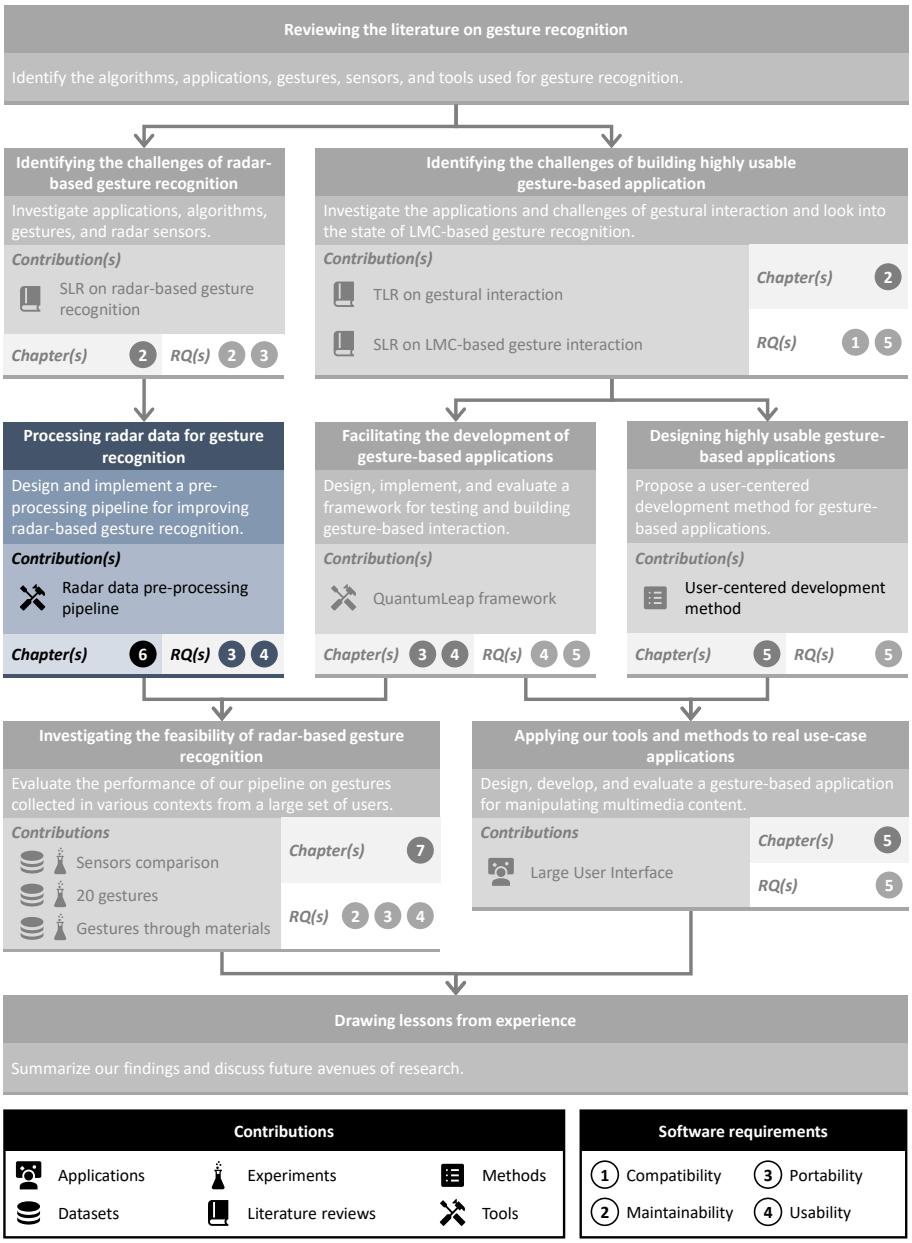


FIGURE 6.1: Main contributions of this chapter.

is no longer affected by the background scene, and (3) one-shot calibration, *i.e.*, the system needs to be calibrated only once.

The rest of this chapter consists of four sections. Section 6.1 first explains the mathematical theory for full-wave electromagnetic modeling and inversion. Section 6.2 then describes how we applied this theory in the six stages of our model-based approach for radar signal processing. Section 6.3 discusses the implementation of the pipeline and applies it to a simple application. Finally, Section 6.5 summarizes this section.

Publications. This chapter is based on two papers published in the IUI 2022 conference proceedings (Best paper award) [265] and ACM TiiS journal [266], and a book chapter [151].

Resources. Two demonstration videos of the prototype application built with the pipeline are accessible at `https://youtu.be/qmHcBuxU1ig` and `https://youtu.be/YlD-808gimw`.

6.1 Mathematical Theory

6.1.1 Full-wave Radar Model

The far-field full-wave radar equation [149] assumes a homogeneous field distribution for the back-scattered field over the antenna aperture when the target is far enough from the antenna. The linear block diagram in Fig. 6.2 describes the antenna using four characteristic global reflection (R) and transmission (T) coefficients: $R_i(\omega)$, $T_i(\omega)$, $T_s(\omega)$, and $R_s(\omega)$, where ω is the angular frequency and $a(\omega)$ and $b(\omega)$ are the source and returned signals in the frequency domain, respectively. Wave propagation from the radar antenna to the target is described using the 3D multilayered media Green function, defined as the x-directed backscattered electric field for a unit-strength electric source, both located at the antenna phase center [44, 196, 264, 149]. This equivalent medium is used for our hand gesture application, as an analytical solution of Maxwell's equations exists, while for real hands a 3D numerical solution would be needed, but would be impractical in terms of computing resources and too complex to use. The equivalent linear block diagram shown in Fig. 6.2 is solved as follows:

$$b(\omega) = a(\omega)R_i(\omega) + \frac{a(\omega)T_i(\omega)G_{xx}(\omega)}{1 - R_s(\omega)G_{xx}(\omega)}T_s(\omega) \tag{6.1}$$

Defining the ratio $S(\omega) = b(\omega)/a(\omega)$, like the so-called S_{11} quantity measured by a Vector Network Analyzer (VNA), and the product $T(\omega) = T_i(\omega)T_s(\omega)$,

Eq. (6.1) becomes [149]:

$$S(\omega) = R_i(\omega) + \frac{T(\omega)G_{xx}(\omega)}{1 - R_s(\omega)G_{xx}(\omega)} \tag{6.2}$$

For a pulse radar, the quantity $s(t)$ is measured, with t being time, and $b(\omega)$ is obtained using the Fourier transform:

$$b(\omega) = \mathcal{F}\{s(t)\} = \int_{-\infty}^{\infty} s(t)e^{-2\pi jt\omega}dt \tag{6.3}$$

Then, defining $H_i(\omega) = a(\omega)R_i(\omega)$, $H(\omega) = a(\omega)T(\omega)$ and $H_s(\omega) = R_s(\omega)$, Eq. (6.1) becomes:

$$\mathcal{F}\{s(t)\} = H_i(\omega) + \frac{H(\omega)G_{xx}(\omega)}{1 - H_s(\omega)G_{xx}(\omega)} \tag{6.4}$$

Whether the radar provides the ratio $b(\omega)/a(\omega)$ or the signal $b(\omega)$, the radar-antenna system is fully described by only three independent functions. $R_i(\omega)$ or $H_i(\omega)$ corresponds to a free-space measurement, for which $G_{xx}(\omega) = 0$, while $R_s(\omega)$ is responsible for the multiple reflections between the antenna and the target medium. Eq. (6.2) is analytically inverted to derive Green's function from the measurements, thereby filtering out the effects of the antenna and resulting in a normalized quantity that is independent of the radar in a given frequency range:

$$G_{xx}(\omega) = \frac{S(\omega) - R_i(\omega)}{T(\omega) + S(\omega)R_s(\omega) - R_i(\omega)R_s(\omega)} \tag{6.5}$$

A practical application of the far-field radar equation is to improve radar imaging, *e.g.*, for landmine detection [181].

6.1.2 Determination of the Radar-antenna Characteristic Functions

The radar characteristic functions $R_i(\omega)$, $T(\omega)$ and $R_s(\omega)$ (or $H_i(\omega)$, $H(\omega)$ and $H_s(\omega)$) can be determined by solving a system of equations (6.2) containing at least three different model configurations (denoted k). For our practice, the antenna is positioned at varying distances from a metal sheet, representing an infinite Perfect Electric Conductor (PEC). Knowing the properties of the medium, Green's functions are computed and the corresponding $S_k(\omega)$ are

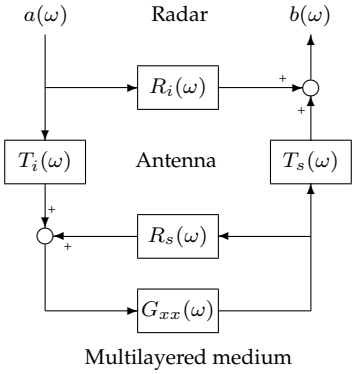


FIGURE 6.2: Equivalent linear block diagram of the radar-antenna-medium corresponding to the far-field radar equation [149].

measured. The system of equations can be expressed in a linear form by reformulating Eq. (6.2) assuming frequency dependence ω:

$$S_k = R_i + S_k G_{xx,k} R_s + G_{xx,k} \left(T - R_i R_s\right) \tag{6.6}$$

Hence, the resulting linear system of equations can be written in matrix form as:

$$\mathbf{b} = \mathbf{A}\mathbf{x} \tag{6.7}$$

where

$$\mathbf{b} = \begin{bmatrix} S_1 \\ \vdots \\ S_k \\ \vdots \\ S_n \end{bmatrix}, \mathbf{A} = \begin{bmatrix} 1 & S_1 G_{xx,1} & G_{xx,1} \\ \vdots & \vdots & \vdots \\ 1 & S_k G_{xx,k} & G_{xx,k} \\ \vdots & \vdots & \vdots \\ 1 & S_n G_{xx,n} & G_{xx,n} \end{bmatrix}, \mathbf{x} = \begin{bmatrix} R_i \\ R_s \\ T - R_i R_s \end{bmatrix} \tag{6.8}$$

The vector of unknowns is then determined using the least squares approach:

$$\mathbf{x} = \left(\mathbf{A}^{\mathrm{H}}\mathbf{A}\right)^{-1} \mathbf{A}^{\mathrm{H}}\mathbf{b} \tag{6.9}$$

where symbol H denotes the Hermitian or conjugate transpose ($\mathbf{A}^{\mathrm{H}} \equiv \overline{\mathbf{A}^T}$). For the Green's function model to apply, which assumes infinite layers, the metal sheet should be larger than five times the maximum wavelength, depending on the antenna radiation pattern. For smaller sizes, the edge effects can affect the results.

6.1.3 Full-Wave Signal Inversion

Full-wave inversion can be used to determine the properties of the layered medium, by minimizing an objective function formulated in the least squares sense, expressed as:

$$\phi(\mathbf{b}) = |\mathbf{G}^*_{\mathbf{xx}} - \mathbf{G}_{\mathbf{xx}}|^T \mathbf{C}^{-1} |\mathbf{G}^*_{\mathbf{xx}} - \mathbf{G}_{\mathbf{xx}}| \tag{6.10}$$

where $\mathbf{G}^*_{\mathbf{xx}} = G_{xx}(\omega)$ and $\mathbf{G}_{\mathbf{xx}} = G_{xx}(\omega, \mathbf{b})$ are the vectors containing the measured and simulated radar signals, respectively. The vector $\mathbf{b} = [\varepsilon_n, \sigma_n, h_n]$ with $n = 1, ..., N$ contains the unknowns, while $\mathbf{C}$ is the measurement error covariance matrix. The discrepancy between observed and modeled data is expressed by the amplitude of the errors in the complex plane, where the radar data vectors are arranged by frequency since the radar data are complex-valued. To focus on a particular range window, containing for example the medium surface only [150], inversion is performed in the time domain. The reflections occurring after the reflection of interest are thereby disregarded and the inverse model configuration can be simplified, e.g., reducing it to a half-space medium. To convert the frequency domain data into the time domain, the Inverse Fast Fourier Transform (IFFT) is applied:

$$g_{xx}(t) = \mathcal{F}^{-1}\{G_{xx}(\omega)\} = \int_{-\infty}^{\infty} G_{xx}(\omega) e^{2\pi j \omega t} d\omega \tag{6.11}$$

Then, a maximum time t_{max} can be set to focus on the initial reflectors of interest, such as the surface reflection (Fig. 6.3). The objective function becomes:

$$\phi(\mathbf{b}) = (\mathbf{g}^*_{\mathbf{xx}} - \mathbf{g}_{\mathbf{xx}})^T \mathbf{C}^{-1} (\mathbf{g}^*_{\mathbf{xx}} - \mathbf{g}_{\mathbf{xx}}) \tag{6.12}$$

where

$$\mathbf{g}^*_{\mathbf{xx}} = g^{\uparrow *}_{xx}(t)\big|_0^{t_{max}} \quad \text{and} \quad \mathbf{g}_{\mathbf{xx}} = g_{xx}(t, \mathbf{b})\big|_0^{t_{max}} \tag{6.13}$$

are the vectors containing, respectively, the observed and simulated time-domain windowed Green functions. Depending on the number of unknowns to estimate and the information available from the radar data, the topography of the objective function to be optimized can be quite complex, and in some cases, the inverse problem can be ill-posed. When dealing with one or two unknowns, local optimization using a method such as the Levenberg-Marquardt Algorithm (LMA) [192] or a Look-Up Table (LUT) approach, which calculates the objective function across the entire parameter space, is reliable and efficient. However, for more complex problems, global optimization is usually required.

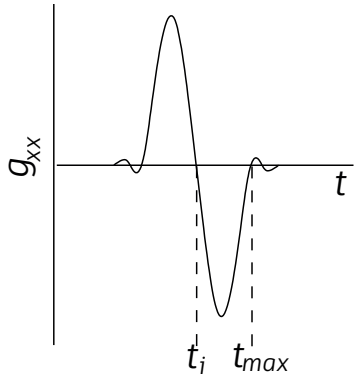


FIGURE 6.3: Sketch of the radar signal in the time domain, $g_{xx}(t)$, with antenna effects removed. The time window for the inversion, denoted as t_{max}, is chosen to focus, as an example, on the surface reflection. t_i denotes the approximate time corresponding to the surface interface for the Green's function.

6.2 Radar Data Processing Strategy

Based on the full-wave electromagnetic modeling and inversion (Section 6.1), a six-stage pipeline processes the radar signals for gesture recognition (Fig. 6.4). Data from each stage can serve as input to gesture recognition: data from stages (b), (c), and (d) can be better suited for image processing algorithms, such as DL [262], while data from stages (e) and (f) could be used with template-matching algorithms [215].

6.2.1 Raw Data Capture

Data is retrieved from one or more receiving radar antennas, either in the time or frequency domain, depending on the radar used (Fig. 6.4a). Raw data consists of the relevant signal with noise and parasitic reflections, which vary depending on the environment and the radar system. To perform subsequent processing stages, the time-domain signal should first be converted to the frequency domain using a Fast Fourier Transform (FFT) Eq. 6.3) in order to apply the normalizing radar equation 6.5. The FFT yields a spectrum spanning from 0 to the Nyquist frequency. Within this spectrum, a specific sub-range is strategically chosen to strike an ideal balance between bandwidth and signal-to-noise ratio. In the context of pulse radar applications, it is common to narrow down this sub-range to a range approximately one-fourth below the center frequency up to twice the center frequency.

6.2.2 Radar Source and Antenna Effects Removal

This second stage removes the effects of the radar source and antenna (except radiation pattern) by applying the radar equation (Eq. 6.5) [149, 148] to the raw signal (Fig. 6.4b). These effects include internal reflections and transmissions between antennas, as well as antenna-target interactions. In the case of antenna arrays, this stage requires the generation of a far-field model of each emitter/receiver pair of antennas by computing their respective characteristic functions. Generating the model is a one-shot process (property P_3) that consists of performing several measurements at different distances from a metal sheet (Section 6.1.2). The far-field version of the radar model is chosen as it is much less resource-intensive to compute than the near-field one and is valid as long as the users stay sufficiently far away from the radar, *i.e.*, the distance to the target should be longer than the radar aperture size [288]. This stage

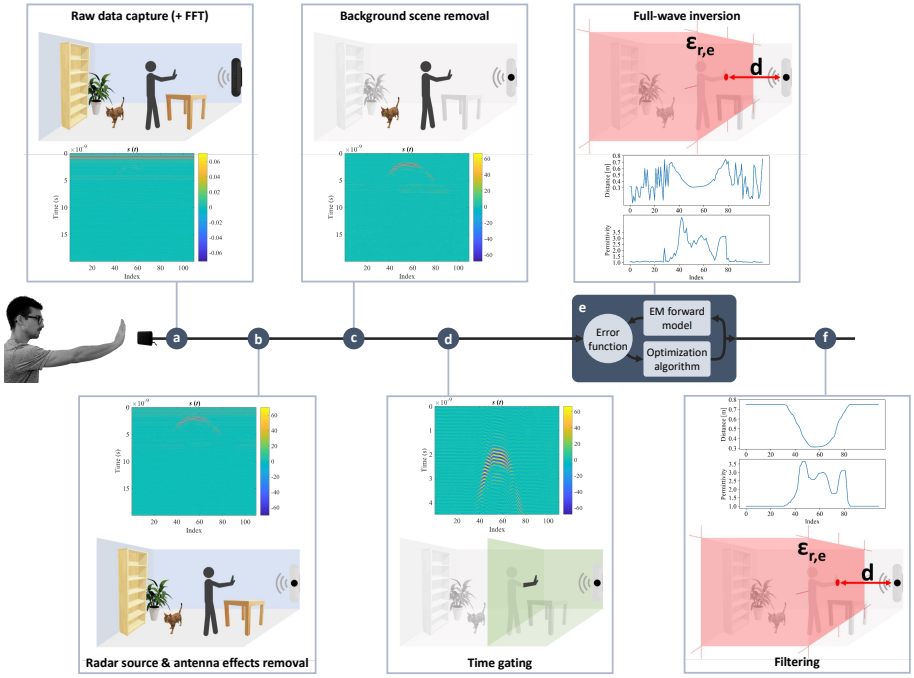


FIGURE 6.4: Overview of the radar data processing strategy. For each stage, a visualization of its effect on the radar data and an example of the corresponding radar signal are provided. The radar signal is represented in the time domain for stages (a) to (d) and as time-distance and time-permittivity plots for stages (e) and (f).

produces a Green's function that is independent of the radar and thereby normalized in the operating frequency range of interest (property P_1) [59]. Fig. 6.5 illustrates the normalization of signals from two distinct radar systems (a Walabot and a custom radar with a BBHA 9120 A horn antenna, both operating from 0.8 to 5 GHz) using the radar equation to filter out source and antenna effects. This process yields nearly identical Green's functions for both systems, despite their different origins, by focusing on the common frequency components. The slight differences observed are attributed to measurement errors. Users' reflections now stand out as undesired echoes are removed. This preliminary experiment suggests that data from this stage could be seamlessly transferred between two distinct radars operating at the same frequencies, or at least with sufficient frequency overlap. The resulting Green's functions can serve as input for various processing stages, such as full wave inversion (Section 6.2.5), feature extraction using CNNs, point cloud generation, or range-Doppler map generation. The data normalization applied by this stage could thus facilitate the reuse of datasets, features, and trained neural networks across different radar systems. We provide the computational complexity of each stage for processing one radar frame based on the variable n, which is the size of the radar signal from a single receiving antenna. The signal size is impacted by the sampling rate of the radar. The computational complexity of this stage is $O(n)$.

6.2.3 Background Scene Removal

Based on the superposition principle, the reflections from static objects in the surrounding environment (*e.g.*, walls, plants, furniture) are subtracted from the normalized signal (Fig. 6.4. This leaves only the reflections of moving objects and persons, including the end user, in the resulting signal (property P_2), resulting in higher accuracy and efficiency of full-wave inversion. It is important to acknowledge that while the interactions between the static objects and the hand are not eliminated with the application of the superposition principle, they are anticipated to be minimal or negligible when the static objects are positioned farther away from the endpoint, as detailed in the following section. The computational complexity of this stage for processing one radar frame is $O(n)$, where n is the size of the radar signal.

6.2.4 Time Gating

An IFFT (Eq. 6.11) is first applied to the signal to convert it to the time domain (Fig. 6.4d), which simplifies subsequent computations. The time-domain signal is then truncated (Fig. 6.4e) to exclude all reflections occurring outside some distance range of the radar (Fig. 6.3 and Eq. 6.13). The distance range

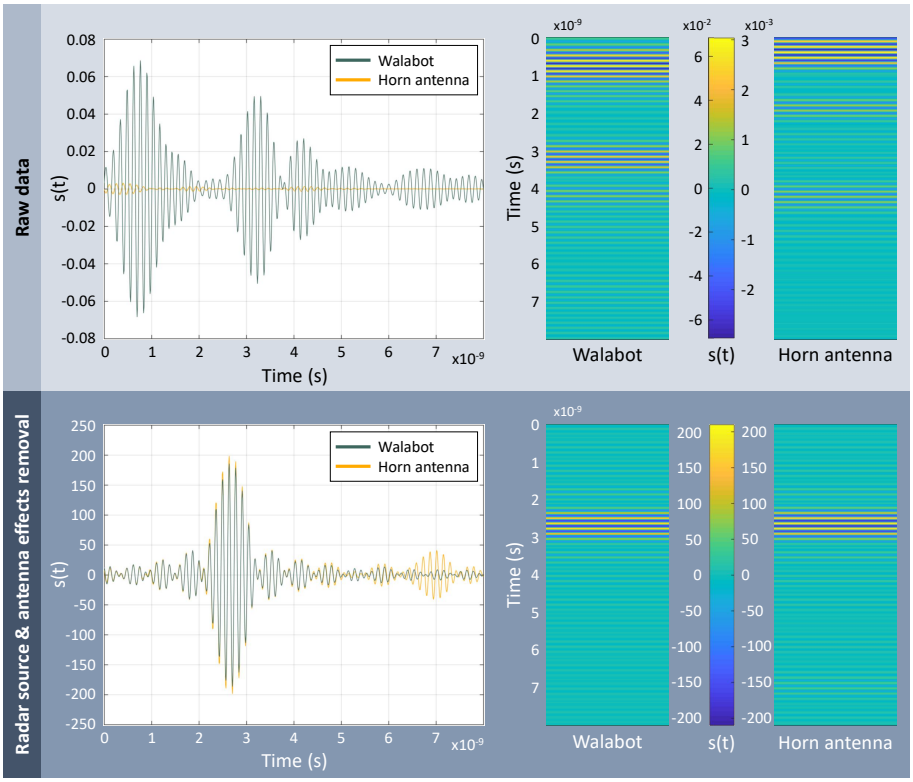


FIGURE 6.5: Radar signal from two radars (Walabot and vector network analyzer + horn antenna) at 40 cm from a large copper plate. The top plots show two representations of the raw time-domain signal. The signals from both radars are at different scales and feature radar source and antenna effects, including internal reflections, making them difficult to compare. The bottom plots show how the signal from both radars is normalized after removing the radar source and antenna effects.

is carefully selected to maximize the signal-to-noise ratio while maintaining enough distance between the user and the radar to ensure the validity of our far-field antenna model (Eq. 6.5). Consequently, movements occurring behind the user are ignored, which improves the efficiency of the next stages, and fewer data need to be processed in the full-wave inversion, thus increasing the efficiency of the next stage and enabling faster real-time processing without compromising on accuracy. Because the truncated radar signal is copied into a new frame, the computational complexity of this stage for processing one radar frame is $O(n)$, where n is the size of the truncated radar signal. The computational complexity could be reduced to a constant value ($O(1)$) if the signal was truncated without copying it into a new frame.

6.2.5 Full-wave Inversion

The process of full-wave inversion reduces the radar signal to two physically meaningful dimensions (Fig. 6.4e): the hand-radar distance d and the relative permittivity $\epsilon_{r,e}$ of an infinite medium equivalent to the hand. The interface between the air and this medium is situated at a distance d from the radar and is orthogonal to the radar-hand direction. If the hand is not perfectly aligned with the radar, however, the distance determined by the system will increase as the reflections from the hand take longer to return towards the radar (see Fig. 6.7). The obtained permittivity is an apparent value that remains unaffected by variations in the hand-radar distance. This apparent permittivity is determined by the quantity of signal reflected by the hand and should not be confused with its true permittivity. As such, it provides information on the pose of the hand, whereas the distance gives the dynamic position of the hand. The inversion algorithm iteratively adjusts the distance and apparent relative permittivity parameters of the model to find the best fit between the modeled and measured signals (Eq. 6.12). The computations are sped up by the use of a pre-calculated LUT for real-time objective function minimization. This drastic data reduction keeps essential information from the radar signal in the form of distance and permittivity values, thus enabling to use of less sophisticated algorithms, such as template-matching algorithms for gesture recognition. The computational complexity of this step is $O(n \cdot k)$, where n is the size of the truncated radar signal and k is the size of the LUT. Despite the use of an LUT, this is the most expensive stage of the pipeline.

We conducted a small experiment to evaluate the sensitivity of the full-wave inversion process to three main variables: (1) the presence of strong reflectors in the background, (2) the distance between the hand and the radar, from 30 cm to 110 cm, and (3) the angle between the hand and the radar, from 0°to 60°at a distance of 30 cm and from 0°to 45°at a distance of 50 cm. A custom radar (vector network analyzer combined with the BBHA 9120 A horn

antenna, 0.8 to 5 GHz, high directivity of 45°3 dB beamwidth in the E-plane and 30°in the H-plane at 1 GHz) was attached to a 3D positioning system and faced a hollow aluminum box of dimensions 21.6 cm (height) $\times$ 10 cm (width) $\times$ 8 cm (depth) attached to a PVC pole. A 3 m $\times$ 3 m copper plate placed on the wall 127.5 cm behind the aluminum box acted as a strong background reflector. Fig. 6.6 and 6.7 illustrate the experimental setup and the results of this experiment. The copper plate in the background did not have an impact on the inversion process as its reflections were removed in the time gating stage. Similarly, the inversion process correctly estimated the distance between the radar and the aluminum box up to 110 cm. Worse results should be expected when using radars with a lower signal-to-noise ratio and/or estimating the distance of less reflective objects. The maximum interaction distance should thus be adjusted accordingly. The results of the inversion process when increasing the angle between the aluminum box and the radar antenna were consistent with its high directivity of 45°: the inversion process failed to correctly estimate the distance from 40°at 30 cm and 46°at 50 cm. The permittivity estimations changed drastically depending on the distance and angle between the aluminum box and the radar antenna, suggesting that this measure should not be used in scenarios where users are expected to move a lot around the radar. Although the electromagnetic model accounts for the decrease of signal strength with respect to distance (3D spherical divergence), these results make sense as the inversion process considers the hand to be an infinite medium to simplify computations. Because the hand is a finite medium, some of the radar signal does not interact with it, resulting in less signal being reflected than anticipated, which corresponds to a lower estimated relative permittivity. This phenomenon is emphasized when the hand is moved further away and/or more offset from the radar antenna. We believe that the model could be improved to account for the finite size of the hand, thus providing an estimation of relative permittivity that would be independent of the hand-radar distance.

6.2.6 Filtering

The result of the full-wave inversion depends on the signal-to-noise ratio, *i.e.*, on the quality of the radar data. A low signal-to-noise ratio and moving objects behind or in front of the user may produce an incorrect estimation of the distance and apparent permittivity. Two strategies are employed to clean up the signal produced by the inversion stage (Fig. 6.4f). First, the distance and permittivity values are discarded and replaced with default values (d=75cm, $\epsilon_{r,e}$=1.0) when the estimated permittivity is close to the relative permittivity of the air, *i.e.*, when no hand is detected in front of the radar. In addition, a moving median filter is applied to smooth out too-fast variations in the estimated distance and permittivity. The moving median filter was selected for its

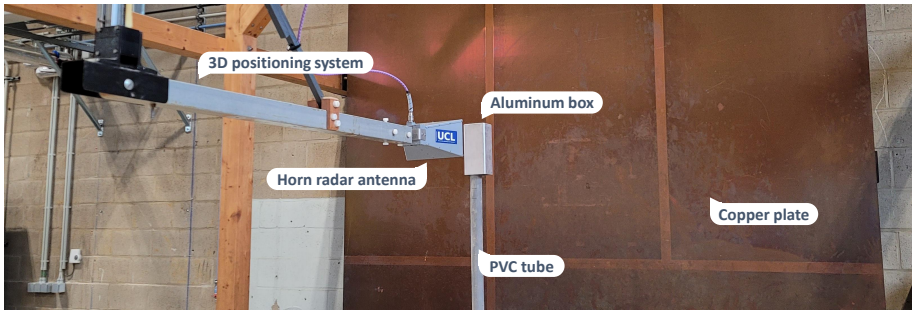


FIGURE 6.6: Experimental setup for evaluating the sensitivity of the full-wave inversion process with respect to distance, angle, and background scene.

ability to filter out large and sudden variations in the data that would not be physically possible. Employing radars with a large dynamic range and leading to a higher signal-to-noise ratio and a wider field of view could decrease the need for filtering, as they would better dissociate the hand signal from the noise and hand reflections, even when the hand is not perfectly in front of the radar. The computational complexity of this stage for processing one radar frame is constant ($O(1)$) as the signal after inversion from one receiving antenna is always reduced to 2 dimensions regardless of its initial size.

6.3 Implementation

The radar data processing strategy was initially implemented based on MATLAB scripts provided by Prof. Sébastien Lambot that were adapted to automatically process an entire gesture dataset. However, these scripts were slow and could not be adapted easily for real-time gesture recognition. Prof. Sébastien Lambot and his team later converted parts of the MATLAB scripts to C++, which we assembled and optimized for real-time gesture recognition and faster batch processing of gestures. Both the MATLAB and C++ versions of the batch processing pipeline compile all processed gestures into the QuantumLeap dataset format. The C++ real-time version of the pipeline initializes a WebSocket server through which it can send processed radar data to other applications, such as the QuantumLeap framework (Chapter 3).

This version of the pipeline was used in a simple prototype application as part of the RadarSense project in Suceava, Romania, which combined touch- and radar-based gestures for manipulating videos on a large screen (Fig. 6.8). Users could interact with videos using touch gestures and switch between categories by pushing their palm towards a radar placed under the table or attached to the screen.

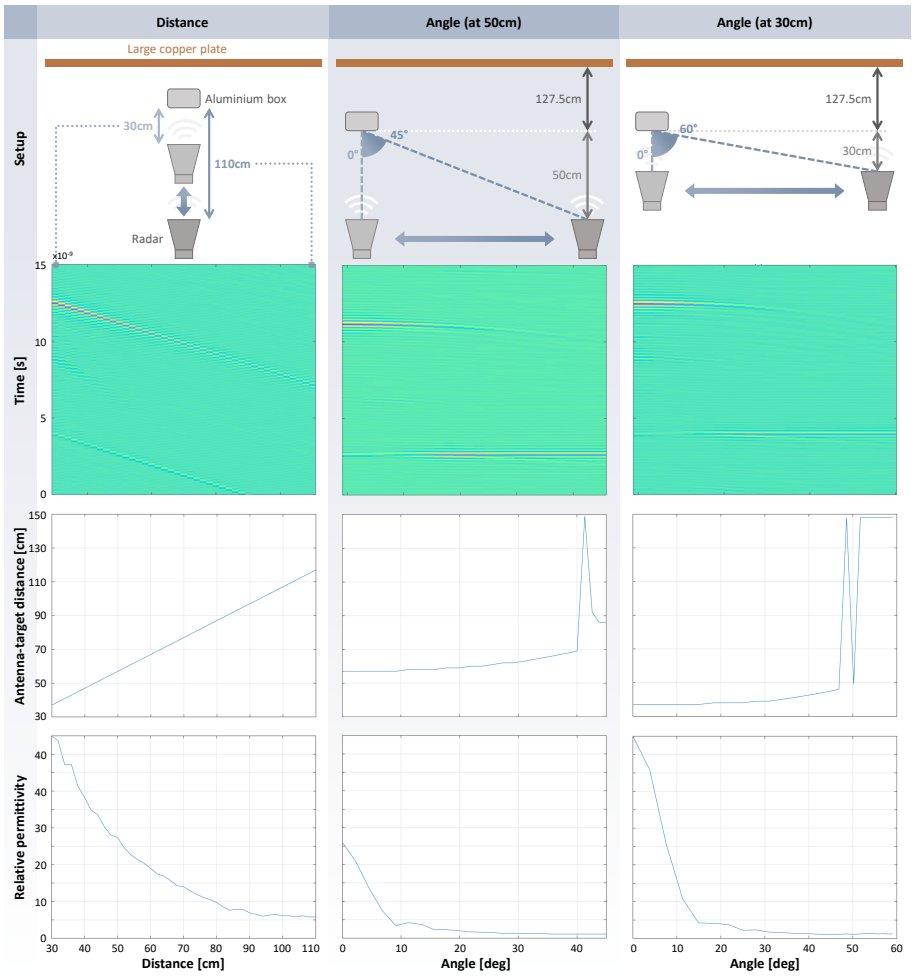
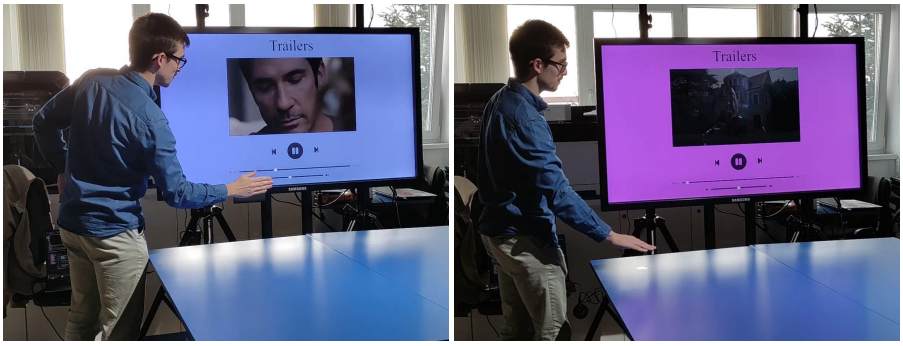


FIGURE 6.7: Effect of radar-target angle and distance on the signal after radar source and antenna effects removal and after inversion.



(A) Radar placed on the screen. (B) Radar placed under the table.

FIGURE 6.8: Screenshots of the prototype application in two configurations.

6.4 Limitations

This section discusses the main limitations of our approach. First, our pipeline does not take into account variations in position with respect to the radar, as shown in Section 6.1.3 and Fig. 6.7, resulting in inaccurate permittivity estimations that change drastically depending on the distance and angle between the user and the radar. Second, our signal processing pipeline neglects anatomical differences between users (*e.g.*, hand size, arm length), which may lower the effectiveness of gesture recognition algorithms in user-independent scenarios (Section 4.1.2). For example, a larger hand will reflect more signal towards the radar, resulting in a greater relative permittivity measured for the same gesture. Taking into account these variations would require at least three changes to the pipeline: (1) normalize the amplitude of user reflections with respect to their hand and body size, (2) extract the hand-body distance instead of the hand-radar distance in the inversion stage, and (3) normalize the hand-body distance with respect to the user's arm length.

6.5 Conclusion

The use of microwave radar sensors for gesture recognition has several advantages compared to vision-based sensors, including less sensitivity to environmental conditions. However, because of the complexity of radar signals, most of the existing work on this topic has relied on complex machine-learning approaches. In this chapter, we introduced a full-wave electromagnetic model and inversion-based approach for radar signal processing, which removes

radar source and antenna effects from raw radar signals, eliminates the reflections of static objects in the vicinity of the radar and moving objects behind the user, and reduces these signals to two physically meaningful features, the hand-radar distance and the effective permittivity of the hand. Two small experiments were conducted to investigate the efficacy of our pipeline. The first experiment showed that the pipeline could effectively normalize signals from two distinct radars, including a Walabot and a custom radar equipped with a horn antenna. The second investigated the sensitivity of our pipeline to variations in radar-target distance and angle and showed that signals from our processing pipeline changed with both radar-target distance and angle. Multiple versions of the pipeline were implemented in MATLAB and C++ for batch and real-time signal processing, and a prototype application was created to showcase the system.

This chapter addresses the four software quality properties outlined in Section 1.2.1. Regarding *compatibility*, our pipeline introduces a degree of interoperability between radar systems by normalizing their signals. Datasets produced with one radar could thus be reused with another one, provided they operate at comparable frequency ranges. This interoperability was illustrated in Section 6.2.2, which compared the normalized signals from two different radar systems. Similarly, this normalization of radar signals contributes to *maintainability* and *portability*. For instance, a radar and its corresponding gesture set can be used in a new environment without having to re-record all of the gestures, requiring only a new snapshot of the background scene. Similarly, the pipeline can accommodate new radar systems after they have been properly calibrated. In terms of *usability*, the normalization applied by our pipeline could improve gesture recognition accuracy, thus resulting in more usable gestural interfaces.

We identify three main avenues for future research. First, the pipeline could be modified to take into account two elements: the finite size of the hand, to provide an estimation of relative permittivity independent of the hand-radar distance, and anatomical differences between users. Future work could also explore the possibility of employing trilateration with data from the full-wave inversion stage produced by a set of spatially separated antennas to compute a 3D position of the hand. This would have multiple benefits, including being able to display a cursor on the screen and enabling more accurate recognition of 3D hand trajectories. Finally, we could look into implementing techniques for gesture segmentation to facilitate complex real-time gesture interaction.

Chapter 7

Experimenting with Radar-based Gestures

In this chapter, we attempt to validate the radar processing pipeline designed in Chapter 6 through experimentation and address three of the five research questions defined in Section 1.2.1:

RQ2 *What is the performance of radar sensors for mid-air gesture recognition?*

RQ3 *What types of gesture-based applications would benefit from radar sensors?*

RQ4 *How can we foster collaboration between researchers and practitioners working on (radar) gesture recognition?*

To this end, we collected three datasets that emulate different contexts (environment, sensor(s)) and feature a wide variety of gestures. These contributions are illustrated in Fig. 7.1. The rest of the chapter is organized as follows. Section 7.1 provides information about our data collection, including the three sensors that we employed, the three datasets, and the pre-processing applied to radar data. A first experiment is conducted in Section 7.2, in which the performance of three different sensors (one vision-based and two radar-based) is compared on a dataset of 16 gestures. A second experiment is carried out in Section 7.3, in which the performance of the Walabot is evaluated in user-dependent, user-independent, and mixed scenarios on an extended dataset of 20 gestures and well-differentiated subsets of gestures. A third experiment is conducted in Section 7.4 to analyze the impact of occlusion by three different types of materials (wood, PVC, and glass) on the Walabot and confirm whether our pre-processing pipeline can successfully normalize its signal for accurate gesture recognition. Section 7.5 then discusses the results of these three experiments and their implications on the design of radar-based gesture recognition. Finally, Section 7.6 concludes this chapter.

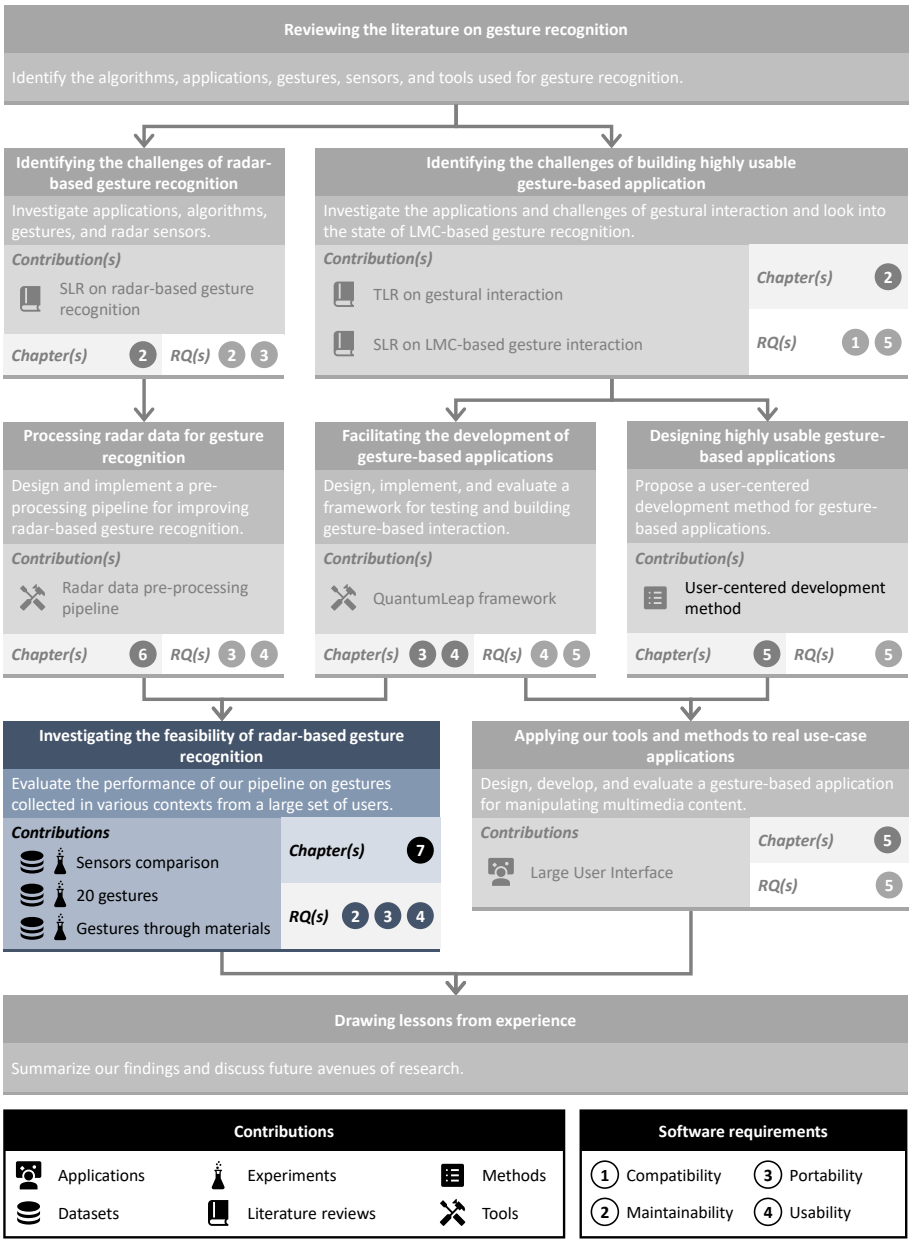


FIGURE 7.1: Main contributions of this chapter.

Publications. This chapter is based on three papers published in the IUI 2022 conference proceedings (Best paper award) [265], ACM TiiS journal [266], and IEEE Access journal [267].

Resources. The datasets, testing configurations, and results from the experiments conducted in this chapter are available on OSF at `https://osf.io/q43j8/?view_only=20b39a8a3d994ae4bd9ce52bfa2ded28`.

7.1 Data Collection

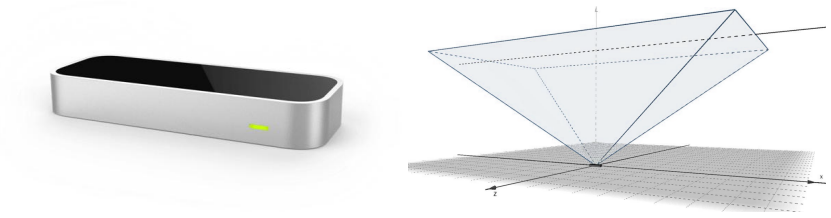
This section provides information about our data collection, consisting of three different datasets collected using three different sensors. Section 7.1.1 describes the three sensors while Section 7.1.2 goes over the three datasets, including their gestures and recording procedure. Section 7.1.3 provides details on the pre-processing applied to each gesture based on the pipeline proposed in Chapter 6.

7.1.1 Sensors

This section describes the three sensors that we used for data collection, including a vision-based sensor (the LMC) and two radar sensors (a Walabot and a custom horn antenna).

LMC

The LMC (Fig. 7.2) is a small, relatively inexpensive, and popular vision-based sensor for gesture recognition. It was described in detail in Section 2.2 when we looked at the state of LMC-based gesture recognition in the literature. The LMC API provides hand skeletons that can be given as input to gesture recognition algorithms.



(A) Sensor. (B) Field of view.

FIGURE 7.2: The Leap Motion Controller.

Walabot

The Walabot Developer is a commercially available, ultra-wideband (UWB) radar sensor employing frequency-modulated continuous-wave (FMCW) technology. It operates in the 6.3-8 GHz range in its EU/CE version and features an array of 18 bowtie radar antennas, including four used as transmitters and 15 as receivers (Fig. 7.3). The advantage of using an antenna array, as opposed to a monostatic or bistatic radar configuration, is its capability to furnish not only distance information but also direction information. The compact form factor of the Walabot (72×140 mm board) and its USB connectivity make it suitable for use in (semi-)mobile and stationary contexts of use. The Walabot Developer SDK enables the retrieval of raw time-domain radar data for use in custom applications. Three profiles are provided for distant scanning, defining the frequency range, the set of antenna pairs (right part of Fig. 7.3), the number of fast-time samples per frame, and the frame rate, as follows:

- Profile 1 (`PROF_SENSOR`): 6.3-8 GHz range, 40 antenna pairs, 8192 fast-time samples/frame, ~20 fps.
- Profile 2 (`PROF_SENSOR_NARROW`): 6.3-8 GHz range, 12 antenna pairs, 4096 fast-time samples/frame, ~41 fps.
- Profile 3 (`PROF_WIDE_BAND`): 3.3-10 GHz range, 40 antenna pairs, 8192 fast time samples/frame, ~15 fps. This profile is only available with the US/FCC version of the Walabot.

The Walabot has been used in various application domains, including material identification [2], activity recognition [19, 365], and gesture recognition [348].

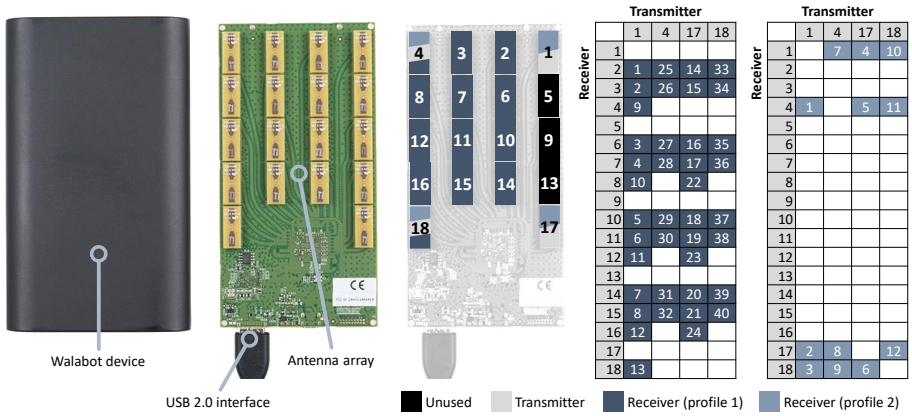


FIGURE 7.3: Walabot enclosure and PCB, antennas IDs, and antenna pairs IDs (profiles 1 and 2).

Custom Radar

The "horn" (Fig. 7.4) is a custom radar equipped with an antenna system consisting of a linearly polarized, double-ridged broadband horn antenna (BBHA 9120 A, Schwarzbeck Mess-Elektronik, Germany). The antenna dimensions are 22 cm (8.66 in.) long and 14×24 cm^2 (5.51×9.45 in.^{2}) aperture area. The antenna's nominal frequency range is 0.8 to 5 GHz, and its isotropic gain ranges from 6 to 18 dBi. The high directivity of the antenna (45°3 dB beamwidth in the E-plane and 30°in the H-plane at 1 GHz) makes it suitable for gesture recognition. This radar system provides very high range resolution, but low angular resolution as it features only one antenna.

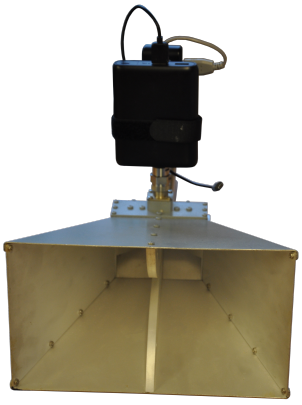


FIGURE 7.4: Custom radar.

7.1.2 Datasets

Three datasets were designed to evaluate the radar data processing strategy in different conditions (Table 7.1). Their gestures are compiled in Fig. 7.6, 7.7, and 7.8. Each gesture is classified according to Aigner *et al.*'s taxonomy [7] and our taxonomy, inspired by Vatavu's iFad gestures [294] (Fig. 7.5).

Name	Classes	Users	Samples	Sensor(s)
Sensors comparison	16	1	320	Walabot, Custom radar, LMC
20-gestures	20	6	1200	Walabot
Through-materials	9	20	2700	Walabot with wood, PVC, and glass

TABLE 7.1: Summary of the datasets used in the three experiments.

Sensors Comparison

A set of 16 gestures was designed by consolidating results obtained in [219, 104, 176] and in a GES: (a) open hand, (b) close hand, (c) open then close hand, (d) extend one finger, (e) extend two fingers, (f) extend three fingers, (g) extend four fingers, (h) swipe right, (i) swipe left, (j) swipe up, (k) swipe down, (l) draw infinity, (o) push fist, (p) push palm, (r) hello, and (u) barrier gesture.

Each gesture was performed by one right-handed participant (male, 55 years old, with no prior experience with radar gestures) and recorded with three different devices, namely an LMC, a Walabot, and a custom radar (horn). Because the overlap between the frequency ranges of the two radars risked producing interference, we decided to record the dataset in two sessions: (1) simultaneously with the LMC and the Walabot, and (2) simultaneously with the LMC and the custom radar. In both sessions, the radar was positioned at a distance of about 70 cm in front of the subject, enough to ensure that the far-field hypothesis of the antenna model was valid at all times (see Section 6.1.1). It was separated from the table by an empty cardboard box, a material relatively transparent to RF waves, with a height of 26 cm. The LMC was placed on the desk, under the participant's hand. The setup for both sessions is shown in Fig. 7.9a and 7.9b.

The participant produced five repetitions of each gesture type in a controlled environment, resulting in a total of 16 (gestures) $\times$ 1 (participant) $\times$ 5 (repetitions) $=$ 80 samples per sensor. The recording procedure was as follows (Fig. 7.10): show the gesture to the participant, then, for each sample (1) the participant starts with their hands on their laps, (2) the recording is started, (3) the participant produces the gesture, (4) the participant places their hands back on their laps, (5) the recording is stopped and the data is saved to a text file. This sequence of actions was repeated for each repetition of each gesture and

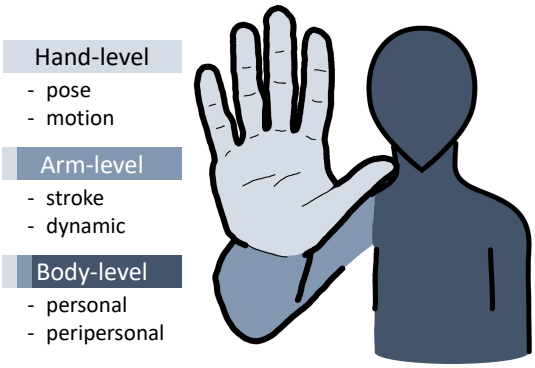


FIGURE 7.5: Taxonomy inspired by Vatavu's iFAD gestures [294].

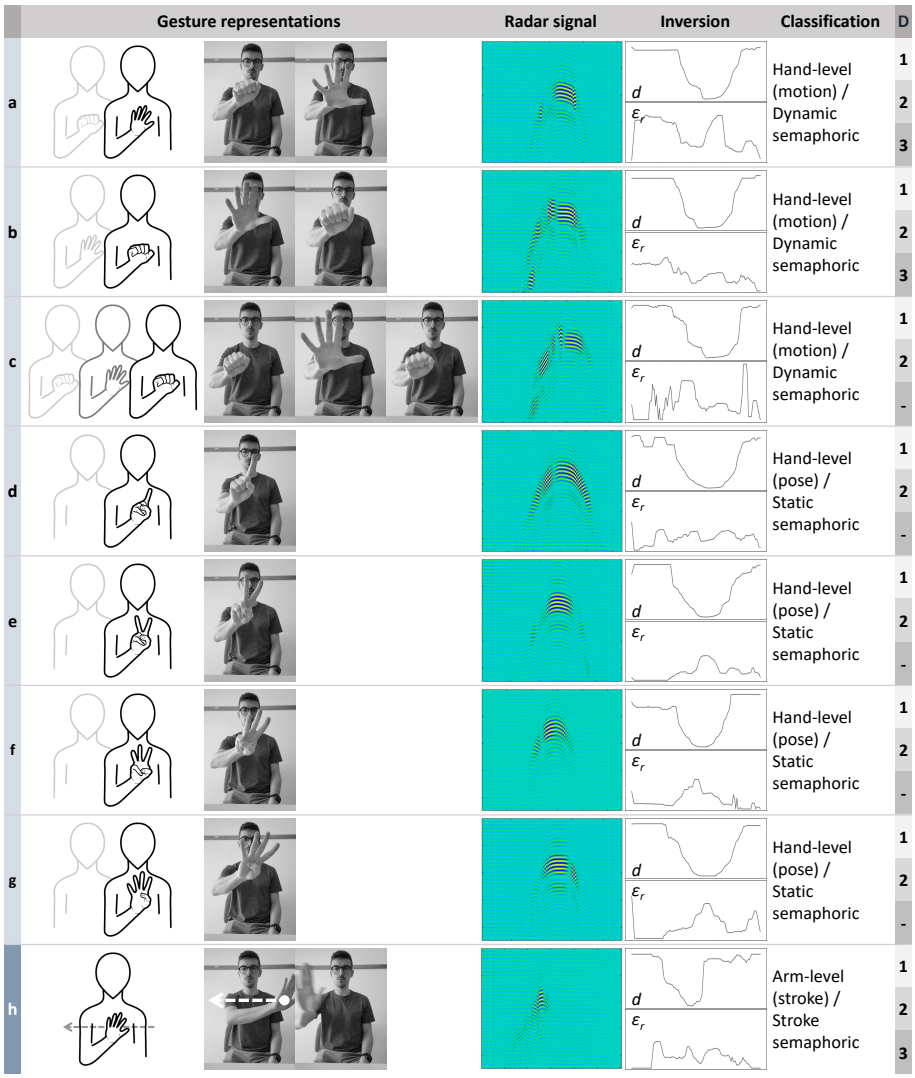


FIGURE 7.6: Consolidation of all gestures from the three radar datasets and their classification according to our gesture taxonomy (Fig. 7.5) and the taxonomy of Aigner *et al.* [7].

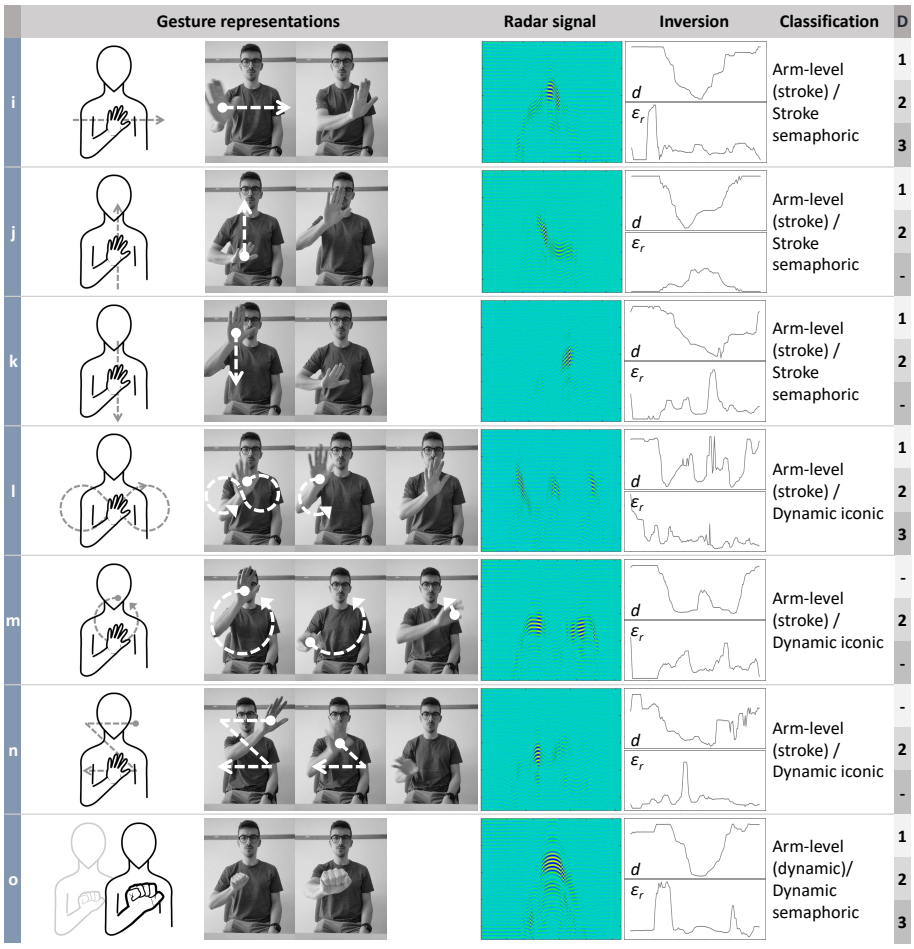


FIGURE 7.7: Consolidation of all gestures from the three radar datasets and their classification according to our gesture taxonomy (Fig. 7.5) and the taxonomy of Aigner *et al.* [7] (cont.).

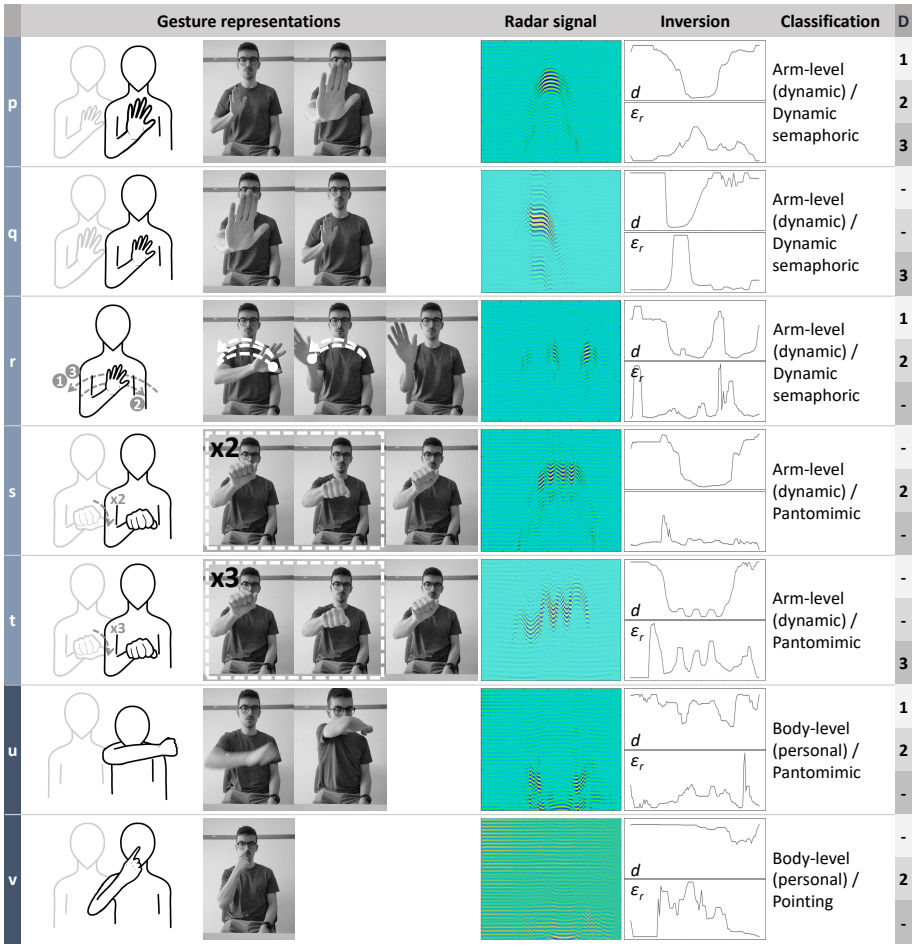
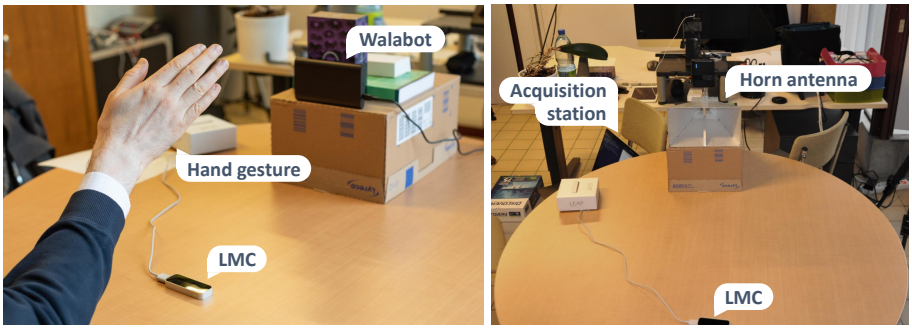


FIGURE 7.8: Consolidation of all gestures from the three radar datasets and their classification according to our gesture taxonomy (Fig. 7.5) and the taxonomy of Aigner *et al.* [7] (cont.).



(A) Walabot and LMC. (B) Custom radar and LMC.

FIGURE 7.9: Setup used for acquiring the "sensors comparison" dataset.

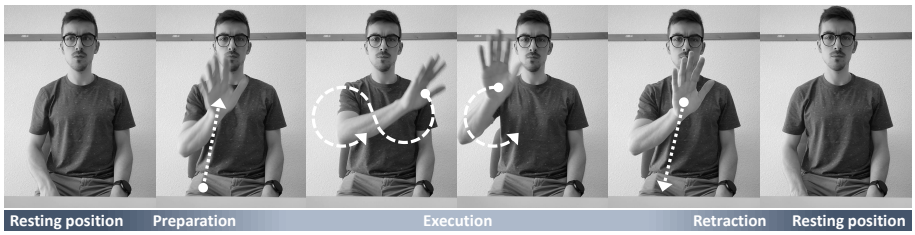


FIGURE 7.10: The five phases of a gesture, based on [122].

each sensor. LMC data was recorded simultaneously with the Walabot/custom radar data at around 100 frames per second using custom software. The Walabot was configured to record with 12 antenna pairs at around 40 frames per second (`PROF_SENSOR_NARROW` profile). It produced time-domain data that was truncated to keep only the first 1024 fast-time samples out of 4096 to maximize the recording framerate. The custom radar produced frequency-domain data at about ten frames per second.

20-gestures Dataset

The second dataset is an extension of the previous dataset with four additional gestures, totaling 20 gestures: (a) open hand, (b) close hand, (c) open then close hand, (d) extend one finger, (e) extend two fingers, (f) extend three fingers, (g) extend four fingers, (h) swipe right, (i) swipe left, (j) swipe up, (k) swipe down, (l) draw infinity, (m) draw circle, (n) draw Z, (o) push fist, (p) push palm, (r) hello, (s) knock twice, (u) barrier gesture, and (v) touch nose.

Each gesture was performed ten times by six participants in a controlled environment and recorded with a Walabot device, resulting in a total of 20

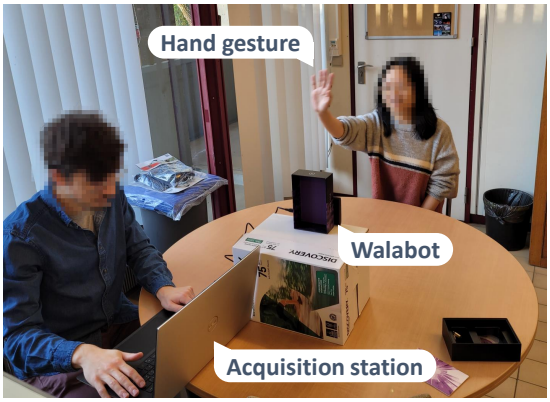


FIGURE 7.11: Setup used for acquiring the "20-gestures" dataset.

(gestures) $\times$ 6 (participants) $\times$ 10 (repetitions) $=$ 1200 samples. The setup (Fig. 7.11) and recording procedure were similar to the first dataset, with the difference being that we only used a Walabot for recording. On average, a recording session (*i.e.*, record ten samples for each of the 20 gestures) took around 30 minutes.

Through-materials

The third dataset consists of nine frequent gestures selected based on our experience with the previous datasets: (a) open hand, (b) close hand, (h) swipe right, (i) swipe left, (l) draw infinity, (o) push fist, (p) push palm, (q) pull palm, and (t) knock three times.

All gestures were recorded with three Walabot devices, each attached to the back of a 1 m $\times$ 1 m plate of a different material (see Fig 7.12): (1) wood (1.7 cm thick), (2) PVC (0.9 cm thick), and (3) glass (0.5 cm thick). While not as extensive as in the work of Pucihar *et al.* [52], this set of materials covers a wide variety of real-world contexts of use and many potential scenarios in which hand gesture recognition could be deployed. For instance, all three materials are used in furniture and as construction materials while some toys are made of wood and PVC. Participants performed gestures while standing up in front of the material. Each material was placed on an easel with its height adjusted so that the participant's right hand faced the Walabot with their arm extended. The distance between the participant's hand and the material was 20 cm. The Walabot configuration was the same as in the other two datasets.



FIGURE 7.12: Setup used for acquiring the "through-materials" datasets.

Each gesture was performed five times by 20 participants in a controlled environment, resulting in a total of 9 (gestures) $\times$ 20 (participants) $\times$ 5 (repetitions) $=$ 900 samples per type of material. The recording procedure was as follows: show the gesture to the participant, then, for each sample, (1) the participant starts with their arms alongside their body, (2) the recording is started, (3) the participant produces the gesture, (4) the participant places their arms back alongside their body, (5) the recording is stopped and the data saved to a text file. This action sequence was repeated for each repetition of each gesture and each material.

7.1.3 Data Pre-processing

Sensor Calibration

All radar sensors, *i.e.*, the horn and the Walabot with and without material, were calibrated in the GPRLouvain facilities, featuring a 3D positioning system and a 3 m $\times$ 3 m copper plane specifically designed for radar calibration. Each radar system was placed in front of the copper plate and measurements were taken from at least five different distances to compute the characteristic functions of each emitter/receiver pair of antennas (see Section 6.1.2). Using more than five distances results in an overdetermined system of equations, thus enhancing the calibration precision. The custom radar and the Walabot



(A) Horn calibration. (B) Walabot calibration. (C) Walabot+wood calibration.

FIGURE 7.13: Calibration process of the different radar systems.

were attached to the 3D positioning system to simplify accurate distance measurements (Fig. 7.13a and 7.13b). The Walabot with wood, PVC, and glass were moved manually as they were too heavy for the system (Fig. 7.13c).

Pipeline Stages

We processed the radar signals from each dataset with our processing pipeline described in Chapter 6, which we configured as follows:

(a) *Raw data capture*: all twelve antenna pairs of the Walabot recorded data at about 40 frames per second.

(b) *Radar source & antenna effects removal*: each antenna pair of the Walabot was calibrated as described in the previous section. The resulting model was applied to each antenna pair to remove the effects of the radar source and antennas.

(c) *Background scene removal*: the second frame of each gesture was subtracted from its other frames.

(d) *Time gating*: the pipeline ignored all reflections received more than 5 ns after the radar signal was emitted, corresponding to approximately 75 cm.

(e) *Full-wave inversion*: the inversion process used a LUT containing 351 permittivity values (between 1 and 8) and 750 distance values (from 1 mm to 75 cm).

(f) *Filtering*: distance and permittivity values were discarded and replaced with default values (d = 75 cm, $\epsilon_{r,e}$=1.0) when the estimated permittivity was lower than 1.1, indicating a hand was not detected in front of the radar. A moving median filter was then applied to the resulting data with a window of size 8. Compared to a moving average, the moving median filter is less sensitive to outliers.

Data from stages a, b, c, and d could be exported in the frequency and time domains.

7.2 Experiment 1: Sensors and Antenna Subsets

7.2.1 Aims

This first experiment investigates the performance of three sensors on the "sensors comparison" dataset (Section 7.1.2), a diverse set of 16 hand gestures produced by one participant. This experiment has three main goals:

1. Compare the performance of the Walabot with the LMC, a popular vision-based sensor, and the horn, a custom high-end radar.
2. Evaluate the efficacy of four stages from our signal-processing pipeline (Chapter 6), namely raw data capture, background scene removal, full-wave inversion, and filtering.
3. Identify the best-performing sets of pairs of Walabot antennas.

7.2.2 Protocol

This experiment consisted of four studies, each following a similar protocol, albeit with slight variations. The QuantumLeap testing tool (Chapter 4) was configured with Taranta *et al.*'s Jackknife [284], a 1-NN gesture recognizer that can accommodate different data types, works with as few as one training template per class, and is fast to train and execute. We chose to work with an existing recognizer, as implementing the best-performing algorithm for (radar) gesture recognition was outside of the scope of this thesis. However, other algorithms (*e.g.*, deep-learning-based) could be used in conjunction with QuantumLeap and our data processing pipeline to achieve better performance. Jackknife was set up with its default settings for the inner product distance measure. The performance of Jackknife on data from each sensor was evaluated following a leave-one-out cross-validation (LOOCV) procedure in a user-dependent and single-dataset scenario (see Table 4.1 for a detailed description of the procedure). The total number of training templates per trial is 79 (four to five per

gesture class), as one of the 80 templates is put aside for testing. The *recognition rate*, *execution time*, and a *confusion matrix* were computed by QuantumLeap for each combination of the studied variables (*e.g.*, the number of training templates). All the tests were run on a laptop with an Intel i7-10875H CPU and 32GB of DDR4 RAM running Windows 10.

We conducted a similar experiment in a previous paper [265] using a train-test-split (TTS) procedure. This experiment addresses some of the limitations of this paper, including the different recording procedures between the LMC+Walabot and LMC+Horn sessions and the fact that Walabot gestures were recorded with only half of the available antenna pairs.

LMC

This first study evaluates the accuracy of Jackknife on the raw LMC data from the LMC+Horn and LMC+Walabot sessions, following the protocol described at the start of Section 7.2.2. The study is within-factors with three independent variables:

1. `Recording Session`: nominal variable with two conditions, representing the recording session of the training data: LMC+Walabot, and LMC+Horn.

2. `Number of Sampling Points`: numerical variable with 37 values representing the number of points per gesture template: $N{=}\{x \in \mathbb{N} \mid 4 \leq x \leq 40\}$.

3. `Number of Joints`: numerical variable with three conditions (Fig. 7.14), representing the number of joints used to train the recognizer: $J{\in}\{6, 11, 16\}$.

Each gesture sample consists of a sequence of frames, where each frame has two elements: a timestamp and a vector of length $3 \times J$ resulting from the concatenation of the 3D coordinates (x,y,z) of the selected joints.

Horn

This second study evaluates the accuracy of Jackknife on the Horn data from four stages of the radar data processing pipeline. The study is within-factors with two independent variables:

1. `Processing Stage`: nominal variable with four conditions, representing the processing stage of the training data: raw data capture, background scene removal, full-wave inversion, and filtering.

2. `Number of Sampling Points`: same as the first study.

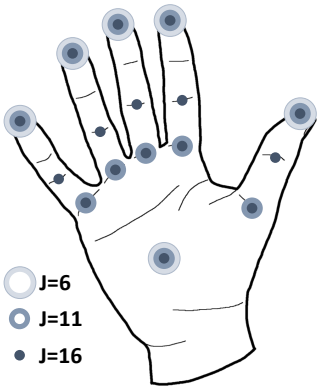


FIGURE 7.14: Joints used in each condition $J \in \{6, 11, 16\}$.

Each gesture template consists of a sequence of frames, where each frame contains two elements: (1) a timestamp and (2) a vector containing the radar data at that instant. In the inversion and filtering stages, the vector contains two elements: the distance and apparent permittivity evaluated at that instant. In the other stages, the vector (length 2×89) contains the real and imaginary parts of the frequency-domain radar signal (89 frequencies).

Walabot

The last two studies were conducted on the Walabot data. The third study evaluates the impact of the selected set of antenna pairs on the accuracy of Jackknife at four stages of the radar data processing pipeline. The number of sampling points is fixed to N=16. The study is within-factors with two independent variables:

1. `Processing Stage`: same as the second study.

2. `Antenna Pairs`: nominal variable with 127 conditions, representing all the possible combinations of non-redundant antenna pairs (excluding the empty set) and the set of all antenna pairs: $AP = \{(1), (2), (3), (6), (8), (9), (1, 2), ..., (2, 3, 6, 8, 9), (1, 2, 3, 6, 8, 9)\} \cup \{(4), (5), (7), (10), (11), (12), (4, 5), ..., (5, 7, 10, 11, 12), (4, 5, 7, 10, 11, 12)\} \cup \{(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)\}$.

The fourth study investigates the impact of the number of sampling points on the accuracy of Jackknife for the three best-performing sets of antenna pairs of each processing stage (determined in the previous study). This study is also within-factors, with three independent variables:

1. `Processing Stage`: same as the first study.
2. `Number of Sampling Points`: same as the first study.
3. `Antenna Pairs`: nominal variable with 18 conditions, representing the three best-performing sets of antenna pairs determined in the first study, all single antenna pairs, the two largest sets of non-redundant antenna pairs, and the set of all antenna pairs. For instance, for the "filtering" processing stage, $AP = \{(1, 2, 6, 8, 9), (1, 2, 8, 9), (4, 7, 10, 11, 12)\} \cup \{(1), (2), ..., (11), (12)\} \cup \{(1, 2, 3, 6, 8, 9), (4, 5, 7, 10, 11, 12), (1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)\}$.

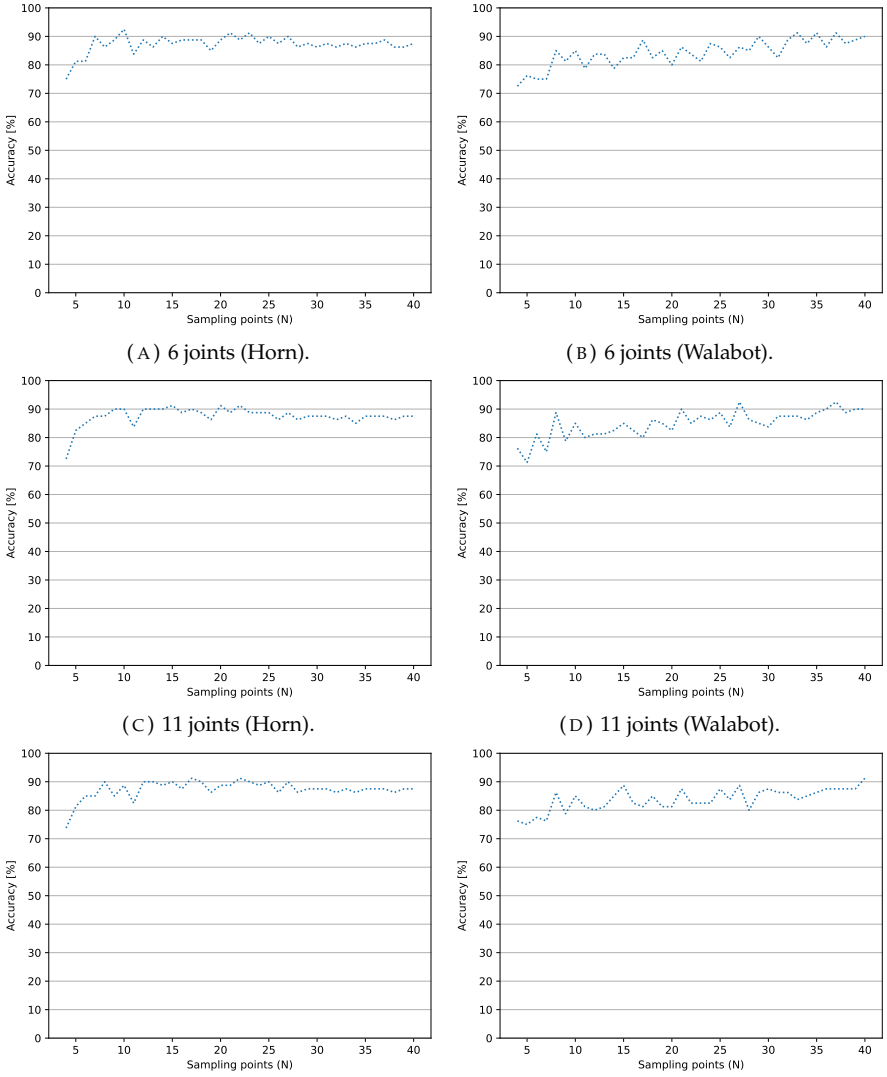
In both studies, a gesture template consists of a sequence of frames, each containing two elements: (1) a timestamp and (2) a vector with the radar data at that instant. In the inversion and filtering steps, the vector (length $2 \times AP$) is the concatenation of the distance and permittivity values retrieved from all the antenna pairs. In the other stages, the vector (length $2 \times 34 \times AP$) results from the concatenation of the real and imaginary parts of the frequency-domain radar signal (34 frequencies) from the selected pairs of antennas.

7.2.3 Results

LMC

Fig. 7.15 depicts the evolution of accuracy with respect to the number of sampling points on data from the LMC+Horn and LMC+Walabot sessions. The best configurations achieved 92.5% accuracy for both the LMC+Horn (N=10, J=6) and LMC+Walabot (N=27, J=11) sessions. Execution time was short at 0.3 and 0.8 ms, respectively, and stayed under 1 ms in most configurations. Overall, the execution time increased with the number of sampling points and joints. In general, average accuracy dropped sharply with N<7 and plateaued around 90% for higher values of N. However, accuracy was less stable with respect to N for the LMC+Walabot session.

Fig. 7.16 shows the confusion matrices for the best configurations of Jackknife on data from the LMC+Horn and LMC+Walabot sessions. In general, smaller-scale hand gestures, including "extend one/two/three/four fingers" (gestures d, e, f, g) were less accurately identified by the LMC, which could have been caused by self-occlusion, where some part of the hand (*e.g.*, the fingers) was occluded by another part of the hand (*e.g.*, the hand palm). The placement of the LMC on a desk under the participant's hands meant that it could not always correctly identify the exact position of the fingers, thus causing confusion between these gestures. We also observed some confusion between the "swipe left" and "barrier" gestures (gestures i and u), between



(A) 6 joints (Horn).

(B) 6 joints (Walabot).

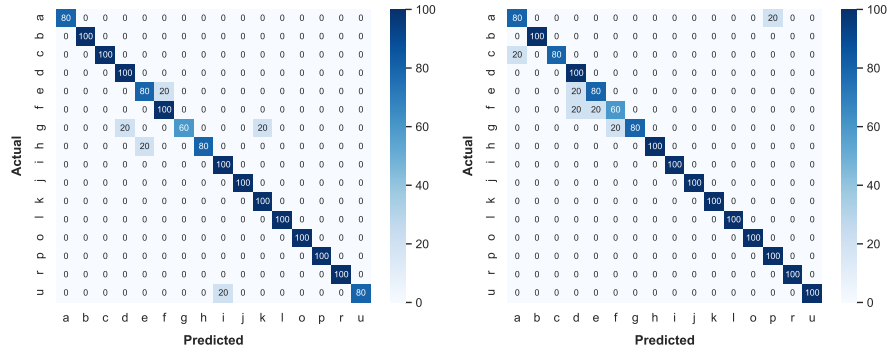
(C) 11 joints (Horn).

(D) 11 joints (Walabot).

(E) 16 joints (Horn).

(F) 16 joints (Walabot).

FIGURE 7.15: The accuracy of Jackknife with respect to the number of sampling points on the LMC data of the LMC+Horn (left) and LMC+Walabot (right) sessions.



(A) LMC+Horn session (N=10, J=6). (B) LMC+Walabot session (N=27, J=11).

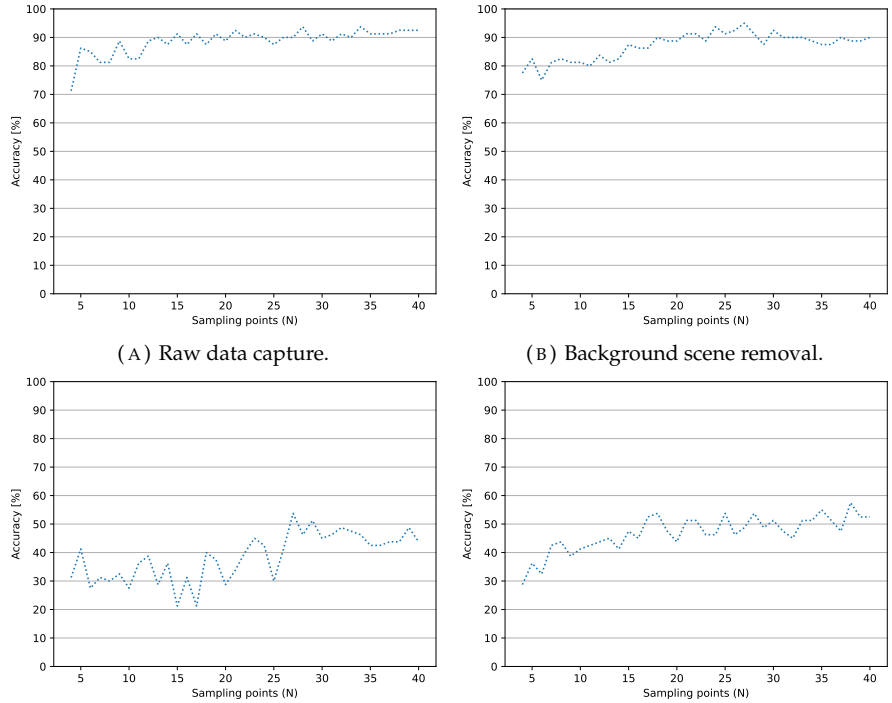
FIGURE 7.16: Normalized confusion matrices for the best configuration with LMC data on the LMC+Horn and LMC+Walabot sessions. The values are represented as percentages.

"open hand" and "push palm" (gestures a and p), and between "extend four fingers" and "swipe down" (gestures g and k). Confusion in all of these cases can be explained by some similarity in the motion of the gestures. For instance, the end of "extend four fingers" looks similar to swiping down. Compared to a similar experiment described in our previous paper [265], we did not observe confusion between the "open hand", "close hand", and "open then close hand" gestures from the LMC+Walabot session, which seems to confirm our hypothesis that the confusion was caused by the fixed length of the recordings.

Horn

Fig. 7.17 shows, for the four processing stages, the accuracy of Jackknife with respect to the number of sampling points and training templates. In addition, Fig. 7.18 shows the confusion matrix of the best-performing configuration for each stage.

Raw data capture. The average accuracy on the raw radar data was already very high and hovered around 90% with $N \geq 12$ sampling points, reaching up to 93.8% for N=28 (Fig. 7.17a), with an average execution time of 6.0 ms. Overall, the execution time increased with the number of sampling points. The corresponding confusion matrix (Fig. 7.18a) shows that 11 of the 16 gesture classes were correctly recognized 100% of the time. The remaining gestures were "extend three fingers", "swipe right", "swipe up", "draw infinity", and "barrier gesture" (f, h, j, l, u). In particular, gestures j and l were confused with gesture k (swipe down). They all featured similar hand poses with different



(A) Raw data capture.

(B) Background scene removal.

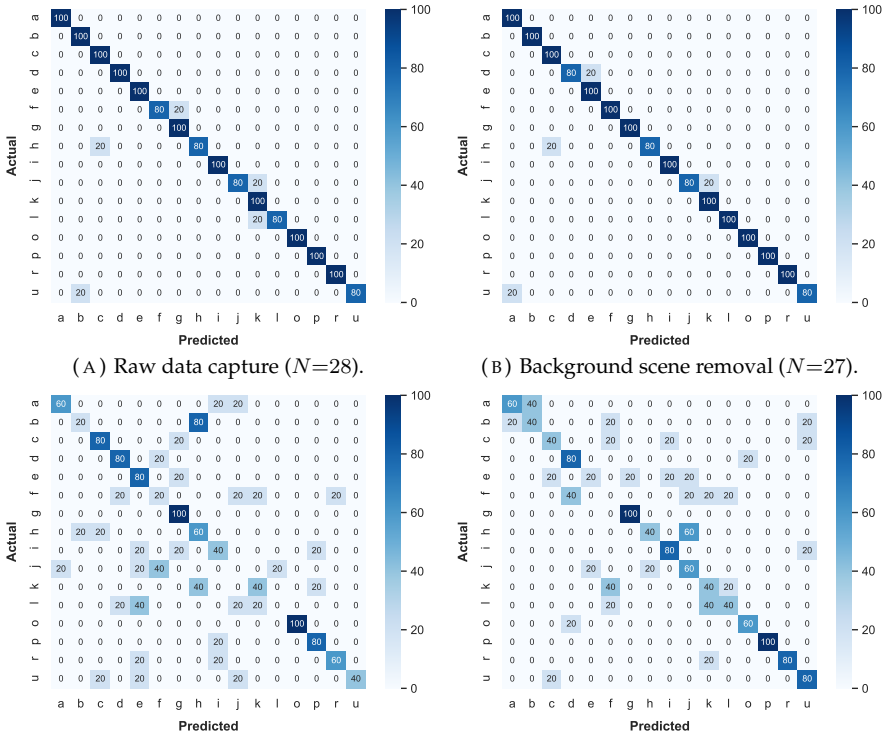
(C) Inversion.

(D) Filtering.

FIGURE 7.17: The accuracy of Jackknife with respect to the number of sampling points with data acquired from the Horn at four different stages of the pipeline.

trajectories in the plane orthogonal to the direction of the emitted radar signal, which can be difficult to distinguish from a single radar antenna. In that respect, an antenna array such as the Walabot, may be preferable. Similarly, gesture h may have been confused with gesture c (open then close hand) because an open hand positioned off-center from the radar antenna reflects less of the signal and can thus be interpreted as a closed fist right in front of the radar. Finally, the confusion between gestures f and g (extend four fingers) is probably caused by their similar radar cross-sections.

Background scene removal. The removal of antenna effects and background subtraction enabled even higher recognition rates, reaching 95.0% for N=27 (Fig. 7.17b), with an average execution time of 5.8 ms. The resulting confusion matrix (Fig. 7.18b) is similar to the confusion matrix obtained on the raw data, with some minor differences that can be explained in the same way. 100%



(A) Raw data capture (N=28).

(B) Background scene removal (N=27).

(C) Inversion (N=27).

(D) Filtering (N=38).

FIGURE 7.18: Normalized confusion matrices for the best configuration with data acquired from the Horn at four different stages of the pipeline. The values in each cell are represented as percentages.

recognition rate was achieved for 12 of the 16 gesture classes, with the four other gestures being "extend one finger", "swipe right", "swipe up", and "barrier gesture" (d, h, j, u).

Full-wave inversion. The recognition rate decreased significantly after the inversion stage and reached a maximum of 53.8% for N=27 (Fig. 7.17c). Average execution time was very short at 0.5 ms. Such a drop was expected, as the size of each frame was divided by 89 compared to the background scene removal stage. We observed that the recognition rate did not steadily increase with the number of sampling points and could change by more than 10% between two consecutive values of N. This behavior may have been caused by errors in the inversion process, where the distance and apparent permittivity values corresponding to one frame were incorrectly estimated due to the low quality

of the radar data (*e.g.*, if the received signal was barely above noise level). The filtering stage should help alleviate this problem. Fig. 7.18c shows an average accuracy equal to or higher than 80% for only six gestures: "open, then close hand", "extend one/two/four finger(s)", and "push fist/palm" (c, d, e, g, o, p). The high accuracy for some of these gestures can be explained by the nature of the permittivity and distance metrics. The estimated permittivity increases when the hand opens (larger radar cross-section) and decreases when it closes (smaller radar cross-section), which helped distinguish between gestures c, d, e, and g, as they mostly differed in their hand pose. Gestures o and p (push fist/palm) were easy to differentiate from other gestures using the distance metric and the permittivity measure allowed to distinguish between the two. As in the previous stages, the "swipe" and "draw infinity" gestures (h, i, j, k, l) were more difficult to recognize due to our reliance on data from a single radar antenna.

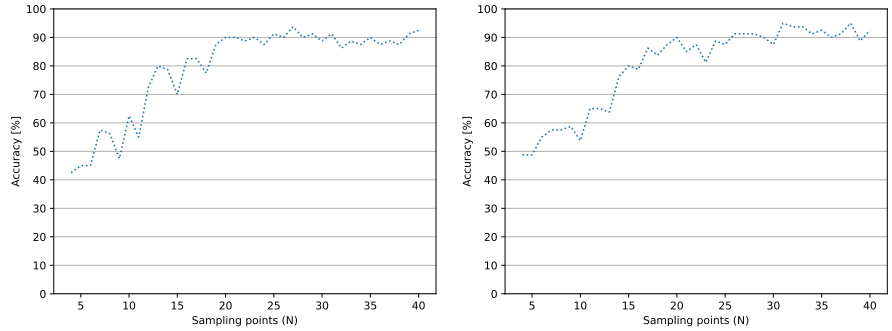
Filtering. As expected, the recognition rate was more stable with respect to the number of sampling points after filtering. However, the best configuration still only reached 57.5% accuracy for N=30 (Fig. 7.17d), far too low for accurate gesture recognition. Execution time was still very short at 0.4 ms. The confusion matrix for this configuration (Fig. 7.18d) shows that the recognition rate exceeded 80% for only six gestures: "extend one/four fingers", "swipe left", "push palm", "wave hand", and "barrier gesture" (d, g, i, p, r, u). Some gestures were recognized less accurately after filtering, including "push fist" (gesture o), which the recognizer sometimes even failed to match with any other gesture. This could indicate that we applied too much filtering, thus erasing some relevant information from the signal in the process.

Walabot

Raw data capture. The recognition rate differed widely depending on the set of antenna pairs used for gesture recognition, with larger sets usually achieving better accuracy (Table 7.2).

In the third study, the three best configurations all achieved $>86\%$ accuracy on the raw Walabot data: 88.8% with AP=(4, 5, 7, 11), 86.3% with AP=(4, 7, 11), and 86.3% with AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12). Execution time increased with the number of antenna pairs and reached up to 10.5 ms with the set of all antenna pairs.

The three best-performing antenna pairs of the third study were selected for the fourth study and tested along with other sets of antennas. Fig. 7.19a plots the evolution of accuracy with respect to the number of sampling points and for the best-performing set of antenna pairs (AP=(1, 2, 3, 6, 8, 9)). Accuracy generally increased with the number of sampling points and



(A) Raw data capture AP=$(1, 2, 3, 6, 8, 9)$.

(B) Background scene removal AP=$(4, 5, 7, 10, 11, 12)$.

(C) Inversion AP=$(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)$.

(D) Filtering AP=$(1, 2, 3, 6, 8, 9)$.

FIGURE 7.19: The accuracy of Jackknife with respect to the number of sampling points for the best-performing set of antenna pairs with data acquired from the Walabot at four different stages of the pipeline.

reached up to 93.8% for N=27, similar to results from the horn session, but slightly higher than the best configuration on the corresponding LMC data. The average execution time for this configuration was 10.1 ms. Other sets of antennas performed worse, with single pairs of antennas achieving between 76.3% (N=28, AP=(5)) and 88.8% accuracy (N=37, AP=(9)). The confusion matrix for the best configuration (Fig. 7.20a) shows that 12 of the 16 gesture classes were correctly recognized 100% of the time. The remaining gestures were “close hand”, “open then close hand”, “extend two fingers”, and “swipe down” (b, c, e, k). Our observations are similar to those we made for the horn data from the LMC+Horn session. In particular, the confusion between gestures e and f (extend three fingers) was likely caused by their similar radar cross-section. The confusion between gestures k and a (open hand), as well as between gestures b, c, and h (swipe right), is similar to what we observed

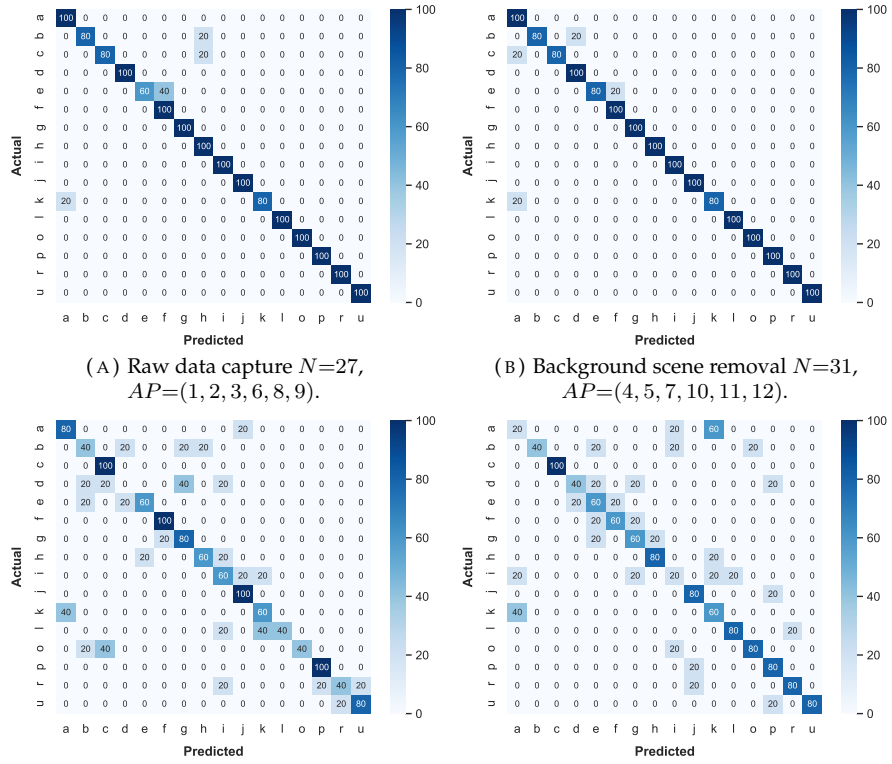
with the Horn. This suggests that the spacing between Walabot antennas may be too small to accurately distinguish between moving the hand in the plane orthogonal to the radar beam and changing its radar cross-section (*e.g.*, fist *vs.* palm).

Background scene removal. Similarly to the Horn, the removal of antenna effects and background subtraction enabled higher recognition rates, with top three configurations achieving $>86\%$ accuracy: 90.0% with $AP{=}(4,\ 7,\ 11)$, 88.8% with $AP{=}(4, 7, 11, 12)$, and 86.3% with $AP{=}(4, 5, 7, 11)$. Execution time increased with the number of antenna pairs and reached up to 10.6 ms with the set of all antenna pairs.

The three best-performing antenna pairs of the third study were selected for the fourth study and tested along with other sets of antennas. Fig. 7.19b plots the evolution of accuracy with respect to the number of sampling points and for the best-performing set of antenna pairs ($AP{=}(4, 5, 7, 10, 11, 12)$). Accuracy also increased with the number of sampling points, reaching up to 95.0% for $N{=}31$, which is higher than the raw data and better than the LMC. The average execution time for this configuration was 11.0 ms. The resulting confusion matrix (Fig. 7.20b) is extremely similar to the confusion matrix obtained on the raw data. Only four gestures were not correctly recognized 100% of the time: "close hand", "open then close hand", "extend two fingers", and "swipe down" (b, c, e, k).

Full-wave inversion. The recognition rate for all sets of antenna pairs after the inversion stage was noticeably lower than after the first two stages (Table 7.2). Such a drop was expected, as the size of each frame was divided by 34 compared to the first two processing stages. The best-performing combinations of antenna pairs were: $AP{=}(4, 7, 10, 11, 12)$ with 45.0% accuracy, $AP{=}(2, 3, 8, 9)$ with 45.0% accuracy, and $AP{=}(1, 2, 3, 6, 8)$ with 43.8% accuracy. The average execution time was very short but increased with the number of antenna pairs, reaching up to 0.5 ms with the set of all antenna pairs.

As previously, the three best antenna pairs were selected for the fourth study. The accuracy reached 65.0% with the set of all antenna pairs for $N{=}34$. The execution time was very short, at about 1.3 ms. Similar to the Horn, accuracy did not steadily increase with the number of sampling points and could change by nearly 20% between two consecutive values of N. This may be caused by errors in the inversion process and amplified by the lower signal-to-noise ratio of the Walabot. Fig. 7.20c shows that the recognition rate is higher than 80% for only seven gestures: "open hand", "open then close hand", "extend three/four fingers", "swipe up", "push palm", and "barrier gesture" (a, c, f, g, j, p, u). Gestures h, i, j, k, and l (swipe right/left/up/down and draw infinity) were better recognized than with the Horn, indicating that the array



(A) Raw data capture N=27, AP=(1, 2, 3, 6, 8, 9).

(B) Background scene removal N=31, AP=(4, 5, 7, 10, 11, 12).

(C) Inversion N=34, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(D) Filtering N=35, AP=(1, 2, 3, 6, 8, 9).

FIGURE 7.20: Normalized confusion matrices for the best configuration with data acquired from the Walabot at four different stages of the pipeline. The values in each cell are represented as percentages.

of antennas of the Walabot managed to pick up more information to accurately differentiate these similar gestures.

Filtering. The best-performing combinations of antenna pairs after filtering were: 51.3% with AP=(1, 2, 6, 8, 9), 50.0% with AP=(1, 2, 8, 9), and 50.0% with AP=(4, 7, 10, 11, 12).

The three best antenna pairs were selected for the fourth study. The accuracy reached 63.8% with AP=(1, 2, 3, 6, 8, 9) and N=35. While it is slightly lower than before filtering, the recognition rate was also more stable with respect to the number of sampling points (Fig. 7.19d). The execution time is still extremely fast at 0.8 ms. The confusion matrix in Fig. 7.20d shows that

AP	Raw data			Background subtraction			Full-wave inversion			Filtering		
	RR M (SD) [%]	ET [ms]	N	RR M (SD) [%]	ET [ms]	N	RR M (SD) [%]	ET [ms]	N	RR M (SD) [%]	ET [ms]	N
1st best	92.5 (17.7)	5.6	22	91.3 (14.6)	6.1	28	53.8 (34.0)	0.5	23	56.3 (29.4)	0.4	19
2nd best	90.0 (16.3)	5.2	27	93.8 (12.0)	7.1	28	47.5 (27.2)	0.8	35	50.0 (29.2)	0.3	16
3rd best	93.8 (14.1)	16.7	23	92.5 (12.4)	10.5	40	53.8 (28.0)	0.9	38	60.0 (25.3)	0.6	29
1	82.5 (20.5)	3.4	38	77.5 (27.2)	2.8	34	30.0 (23.1)	0.3	22	28.8 (31.0)	0.0	4
2	77.5 (20.5)	3.4	37	76.3 (24.5)	1.7	24	18.8 (22.5)	0.9	39	26.3 (30.7)	0.3	36
3	81.3 (18.5)	2.7	33	78.8 (21.3)	2.9	39	26.3 (23.9)	0.9	40	30.0 (24.2)	0.1	10
4	76.3 (22.2)	3.1	35	81.3 (18.6)	2.8	37	20.0 (14.6)	0.6	35	22.5 (23.0)	0.1	8
5	76.3 (18.2)	2.1	28	75.0 (27.8)	3.1	40	21.3 (22.5)	0.2	16	25.0 (29.7)	0.0	4
6	81.3 (15.4)	2.2	29	82.5 (19.2)	2.6	32	28.8 (25.3)	0.7	37	31.3 (23.1)	0.3	27
7	78.8 (28.7)	3.1	40	80.0 (25.3)	3.2	40	33.8 (26.0)	0.2	16	30.0 (28.3)	0.2	16
8	81.3 (20.0)	2.3	26	80.0 (27.3)	3.1	36	27.5 (25.3)	0.4	29	26.3 (28.0)	0.2	17
9	88.8 (17.8)	2.9	37	87.5 (16.1)	2.7	31	22.5 (25.2)	0.0	4	30.0 (24.2)	0.3	26
10	83.8 (22.2)	2.6	33	80.0 (20.7)	3.1	34	23.8 (22.2)	0.5	28	32.5 (33.4)	0.0	4
11	86.3 (21.6)	3.0	37	87.5 (23.0)	2.9	31	23.8 (22.2)	0.1	13	23.8 (20.9)	0.1	7
12	77.5 (17.7)	2.2	29	80.0 (19.3)	2.7	31	28.8 (30.1)	0.5	36	31.3 (27.3)	0.3	26
1, 2, 3, 6, 8, 9	93.8 (12.0)	10.1	27	93.8 (9.6)	9.6	26	61.3 (31.4)	0.8	39	63.8 (23.4)	0.8	35
4, 5, 7, 10, 11, 12	92.5 (12.4)	14.4	38	95.0 (8.9)	11.0	31	51.3 (31.0)	0.6	23	56.3 (31.2)	0.4	15
All 12 pairs	93.8 (14.1)	16.7	23	93.8 (12.0)	28.7	38	65.0 (28.8)	1.3	34	58.8 (32.2)	0.5	18

TABLE 7.2: Recognition rates and execution times obtained for the Walabot data at four steps of the processing pipeline for the first, second, and third best sets of antenna pairs in the third study, all single antenna pairs, the two largest sets of non-redundant antenna pairs, and the set of all antenna pairs. AP=antenna pairs, RR=recognition rate [%], ET=execution time [ms], N=number of sampling points. The best-performing sets of antenna pairs are highlighted in blue.

eight gestures were correctly recognized more than 80% of the time, with only one being correctly recognized 100% of the time: "open then close hand", "swipe right/up", "draw infinity", "push fist/palm", "wave hand", and "barrier gesture" (c, h, j, l, o, p, r, u). The low signal-to-noise ratio of the Walabot compared to the Horn caused some confusion between gestures d, e, f, g (extend one/two/three/four finger(s)) as they all feature small variations in radar cross-section. Similarly to the previous stage, gestures h, j, k, l, and r (swipe right/up/down, draw infinity, wave hand) were recognized relatively accurately, thanks to the antenna array of the Walabot. However, the spacing between the Walabot antennas is too small to prevent confusion between gesture i (swipe left) and gestures a and b (open/close hand).

7.2.4 Summary

Removing the antenna effect and background scene provided minor gains in accuracy compared to relying on the raw radar data. We could see more meaningful gains when using the data from different radars for training and testing. Conversely, the average accuracy dropped after the full-wave inversion stage, reaching a maximum of 65%, too low for use in actual gesture-based interfaces.

The best performance was achieved with the set of all antenna pairs (AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)) and the two largest sets of non-redundant antenna pairs (AP=(1, 2, 3, 6, 8, 9) and AP=(4, 5, 7, 10, 11, 12)). The next experiments will thus not evaluate other sets of antenna pairs. Overall, the Walabot achieved better accuracy than the LMC and the Horn, indicating that inexpensive radars could be used successfully for gesture interaction, at least in a controlled environment. The twelve antenna pairs of the Walabot gave it an edge on the Horn's single antenna, despite their lower quality. However, more spaced-out antennas could achieve even better angular resolution, which would benefit some types of gestures, such as "swipe right/left/up/down", "wave hand", and "draw infinity". Using three or more spread-out antennas could also enable us to use trilateration to identify the 3D position of the hand based on data from the full-wave inversion and filtering stages. This could improve the recognition of 3D hand trajectories and be used to display a cursor indicating the position of the user's hand in an application. Finally, sensor fusion techniques such as Kalman filters (Section 2.1.2) could be leveraged to combine the strengths of radar and vision-based sensors, thus improving gesture recognition accuracy and robustness in challenging environmental conditions.

7.3 Experiment 2: Gestures Subsets

7.3.1 Aims

This second experiment evaluates the performance of the Walabot on the "20-gestures" dataset, a set of 20 hand gestures produced by six participants (Section 7.1.2). It has two main goals:

1. Evaluate the efficacy of our signal-processing pipeline on a large set of gestures from multiple users in user-dependent, user-independent, and mixed scenarios.

2. Explore how smaller, highly differentiable subsets of the gesture set affect the performance of our approach in a user-dependent scenario.

7.3.2 Protocol

This experiment was divided into two studies, both following a similar protocol to the first experiment (Section 7.2.2). The QuantumLeap testing tool ran on the same computer and used the Jackknife recognizer with the same default settings. The performance of Jackknife was evaluated following a leave-one-out cross-validation (LOOCV) procedure in three user scenarios (see Table 4.1 for a

detailed description). Both studies used the same format for gesture templates as the Walabot studies in the first experiment (Section 7.2.2).

All Gestures

The first study of this experiment evaluates the accuracy of Jackknife on the whole dataset at two stages of the radar data processing pipeline in three user scenarios. The study is within-factors with four independent variables:

1. `Processing Stage`: nominal variable with two conditions, representing the processing stage of the training data: time gating, filtering.
2. `User Scenario`: nominal variable with three conditions, representing the type of user scenario: user-dependent, user-independent, mixed.
3. `Number of Sampling Points`: numerical variable with 37 values representing the number of points per gesture template: $N{=}\{x \in \mathbb{N} \mid 4 \leq x \leq 40\}$.
4. `Antenna Pairs`: nominal variable with three conditions, representing the selected combination of antenna pairs: $AP = \{(1,2,3,6,8,9), (4,5,7,10,11,12), (1,2,3,4,5,6,7,8,9,10,11,12)\}$.

The total number of training templates per trial is 199 (nine to ten per gesture class) in the user-dependent scenario, 1000 (50 per gesture class) in the user-independent scenario, and 1199 (59 to 60 per gesture class) in the mixed scenario.

Gesture Subsets

This second study evaluates the accuracy of Jackknife on subsets of our "20-gestures" dataset at two stages of the radar data processing pipeline in a user-dependent scenario. Three of those subsets were created based on our taxonomy (Fig. 7.5): hand-level, arm-level, and body-level gestures. The other subsets, consisting of highly differentiable gesture classes, were created as follows: (1) identify all subsets with zero (resp. one) confusion between pairs of gesture classes in the results from the previous user-dependent testing, (2) order them by number of gesture classes and total number of confusions, (3) select the largest subset(s) with zero (resp. one) confusion between pairs of gesture classes. We thus have both larger subsets with low confusion between gesture classes and smaller subsets with little to no confusion. The study is within-factors with four independent variables:

1. `Processing Stage`: same as the first study.

Id	Description	Gestures
S1	Hand-level	a b c d e f g
S2	Arm-level	h i j k l m n o p r s
S3	Body-level	u v
S4	Filtering, C=0	b g l
S5	Filtering, C=1	b g h l
S6	Time gating, C=0 (a)	a e h k m n o r
S7	Time gating, C=0 (b)	a e h k n o r v
S8	Time gating, C=1 (a)	b f h i j l m n p r s v
S9	Time gating, C=1 (b)	b f h i k l m n p r s v

TABLE 7.3: The nine selected subsets of gestures. C represents the maximum number of confusion between pairs of gestures. Gestures are colored based on the taxonomy in Fig. 7.5.

2. `Number of Sampling Points`: same as the first study.
3. `Antenna Pairs`: same as the first study.
4. `Gesture Subset`: nominal variable with nine conditions, representing the gesture subset used for training and testing (see Table 7.3).

The selected subsets are summarized in Table 7.3 and highlighted in Fig. 7.21. The time-gating subsets (S6 to S9) consist mostly of arm-level gesture classes, with only two hand-level gesture classes per subset. This indicates that the larger motion of arm-level gestures was easier to differentiate than the smaller-scale hand-level gestures. The filtering subsets (S4 and S5) contain fewer gesture classes than the time-gating subsets, which reflects the high confusion between gesture classes when using data from the filtering stage. The total number of training templates per trial varies based on the size of the gesture subset but the training sets always feature nine or ten templates per gesture class.

7.3.3 Results

All Gestures

Accuracy reaches 90.6% over the whole dataset in the user-dependent scenario using data from the time gating stage, with an average execution time of 34ms (N=38, AP=(4, 5, 7, 10, 11, 12)). We observe more confusion between gesture classes d, e, f, g, and p (extend one/two/three/four finger(s), push palm), most of which fall under 90% accuracy. These gestures feature small changes in hand pose (and thus, slight variations in the amplitude of the reflected signal)

and similar trajectories, making them tricky to differentiate. Some confusion is also observed between gestures j and k (swipe up and down), whose opposite trajectories result in very similar radar signatures. Conversely, seven gesture classes reach over 95% accuracy, including one hand-level gesture (j - swipe up), five arm-level gestures (h, i, l, m, and n - swipe right/left, draw infinity/circle/Z), and one body-level gesture (v - barrier gesture). Accuracy drops to 45.2% with data from the filtering stage (N=26, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)), with a much shorter execution time at 2ms. Only seven gesture classes are recognized with more than 50% accuracy, namely b, k, l, m, n, p, and s (close hand, swipe down, draw infinity/circle/Z, push palm, knock twice). All but two of these classes (b and s) feature different motions in space with similar hand poses. The distance measurement is thus crucial for distinguishing between these gesture classes, while the relative permittivity is less relevant. We also observe some confusion between gesture classes d, e, f, g, o, and p (extend one/two/three/four finger(s), push fist/palm), which feature different hand poses but similar trajectories. In particular, gestures d and o are recognized with less than 30% accuracy. The relative permittivity is more relevant than the distance measure to identify hand poses. As such, the permittivity measures may lack the required precision to accurately distinguish between these slight variations in hand poses, thus explaining the lower accuracy on these gesture classes. In this context, it is noteworthy to mention that the obtained permittivity values exhibit high sensitivity to minor shifts in hand orientation, rendering them susceptible to recognition inaccuracies. Fig. 7.21 shows the confusion matrices corresponding to these two configurations.

Accuracy decreased sharply in the user-independent scenario, dropping to 24.6% using data from the time gating stage (N=38, AP=(1, 2, 3, 6, 8, 9)). Looking at the performance for each gesture class shows similar patterns to the user-dependent scenario, albeit with much lower accuracy. There was a lot of confusion between gesture classes a, b, c, d, e, f, g, and p (open/close hand, open then close hand, extend one/two/three/four finger(s), push palm). Only five gesture classes reached more than 30% accuracy, namely k, m, p, u, and v (swipe down, draw circle, push palm, barrier gesture, touch nose). In particular, class k, which had a very different radar signature than most other gestures, was accurately recognized 77% of the time. Accuracy was only 15.8% with data from the filtering stage (N=29, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)). Patterns are very hard to distinguish in this scenario because of the very low accuracy. Only two gesture classes reached 30% accuracy, namely, classes m and n (draw circle/Z). Some form of normalization of the signal with respect to the users could be applied to improve the performance of the system. Fig. D.1 shows the confusion matrices corresponding to these two configurations.

In the mixed scenario, accuracy was only slightly lower than in the user-dependent scenario, reaching 90.3% with data from the time gating stage (N=39, AP=(1, 2, 3, 6, 8, 9)), and 40.2% with data from the filtering stage (N=25, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)). Execution time was longer, reaching 203ms (resp. 12ms) with time-domain (resp. inversion) data. Confusion matrices in this scenario are similar to the user-dependent scenario, with similar observations (Fig. D.2).

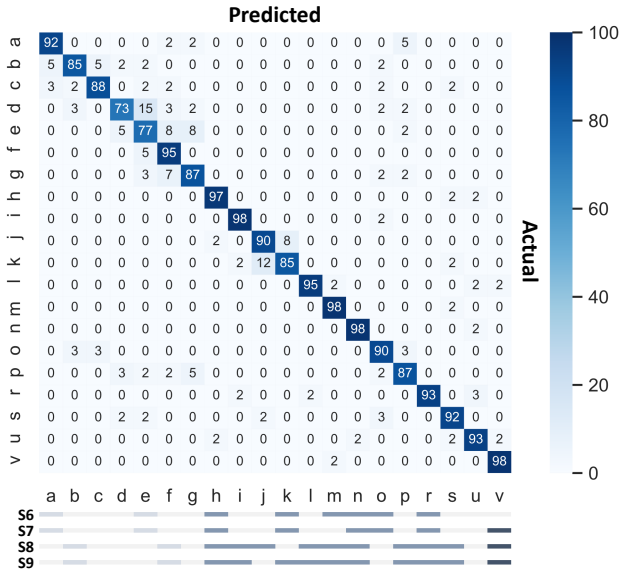
Gesture Subsets

Despite containing fewer gestures, the classification accuracy on subset S1 (hand-level) was lower than on the full set of gestures, at 87.9% (N=35, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)). Conversely, the accuracy on the larger S2 subset (arm-level) was higher, at 96.2% (N=37, AP=(1, 2, 3, 6, 8, 9)), indicating that arm-level gestures may be easier to differentiate with our technique than smaller-scale gestures. Accuracy on body-level gestures (S3) was even higher at 99.2% (N=20, AP=(4, 5, 7, 10, 11, 12)), but the small size of this subset left less room for confusion between gestures. As expected, classification accuracy was lower on data from the filtering stage, with the accuracy for hand-, arm-, and body-level gestures reaching 51.9% (N=26, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)), 55.5% (N=34, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)), and 80.0% (N=28, AP=(4, 5, 7, 10, 11, 12)), respectively. A similar pattern emerged, however, with arm-level gestures being classified more accurately than hand-level ones despite the greater size of the subset.

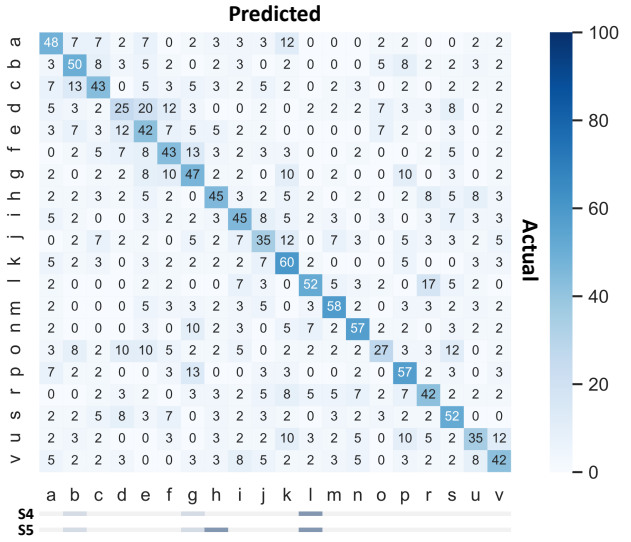
The accuracy on subsets S4 and S5 with data from the filtering stage was relatively low considering the size of the subsets: 86.1% (N=22, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)) and 77.5% (N=23, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)) for three and four gesture classes, respectively. Conversely, Jackknife managed a high accuracy of 98.5% on subsets S6 (N=38, AP=(4, 5, 7, 10, 11, 12)) and S7 (N=38, AP=(4, 5, 7, 10, 11, 12)) using data from the time gating stage. Accuracy was still high at 97.8% and 97.5% on the larger S8 (N=38, AP=(4, 5, 7, 10, 11, 12)) and S9 (N=37, AP=(4, 5, 7, 10, 11, 12)) subsets, respectively. Other subsets could perform even better but testing them all would be unrealistic due to the large number of subsets.

7.3.4 Summary

Overall, the best performance on the whole dataset was achieved in user-dependent and mixed scenarios, using data from the time-gating stage of our processing strategy. For real-time execution in a mixed scenario, the number of training samples should be carefully selected to maximize accuracy while minimizing the response time of the system, which depends mostly on the



(A) Time gating stage N=38, AP=(4, 5, 7, 10, 11, 12).



(B) Filtering stage N=26, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

FIGURE 7.21: Confusion matrices of the best-performing configurations in the user-dependent scenario on the complete dataset.

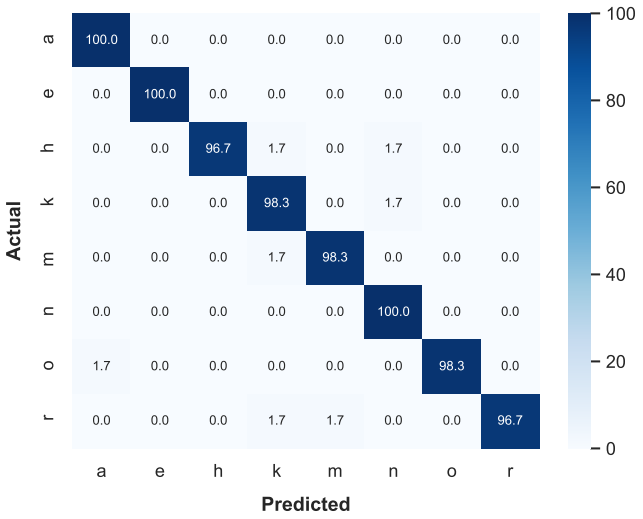


FIGURE 7.22: Confusion matrix of the best-performing configuration on gesture subset S6 (time gating stage).

execution time of the recognizer. The good performance of the system in the mixed scenario showed that it could accurately recognize the gestures of one user, even when the training set featured other users' samples. This property makes the combination of our pipeline and the Jackknife recognizer ideal for building gestural interfaces that are trained and used by a small set of users, such as a gesture-based TV remote. In addition, the use of a template-matching recognizer enables quick and easy customization of the gesture set.

The evaluation of nine gesture subsets showed that the Walabot had some trouble differentiating between hand gestures that featured small differences in radar cross-section, while arm-level gestures were easier to differentiate. In addition, it suggested that using smaller, well-differentiated sets of gestures could be the key to building reliable radar-based gesture recognition. Our method for subset selection (Section 7.3.2) managed to effectively produce highly-differentiable sets of gestures. Although better subsets may be identified by testing all possible combinations, this becomes unrealistic as the pool of potential gestures grows.

7.4 Experiment 3: Gestures through Materials

7.4.1 Aims

Radar sensing can penetrate non-conducting materials, such as glass, wood, and plastic, thus making it appropriate for gesture recognition in environments with poor visibility, *e.g.*, due to occlusion, which are often incompatible with other types of sensors. This third and last experiment investigates the performance of the Walabot on the "through-materials" dataset, a set of nine hand gestures performed by 20 participants sensed through three types of materials (wood, PVC, and glass). It has two main goals:

1. Evaluate the performance of the Walabot for sensing gestures through wood, PVC, and glass.
2. Evaluate the ability of our signal-processing pipeline to normalize radar data collected through different materials.

The experiment described in this section is similar to a study conducted by Leiva *et al.* [160], that analyzed the impact on gesture recognition accuracy of different fabrics covering a Google Soli radar.

7.4.2 Protocol

This experiment consists of two studies that follow a similar protocol to the first two experiments (Sections 7.2.2 and 7.3). The QuantumLeap testing tool ran on the same computer and used the Jackknife recognizer with the same default settings. The performance of Jackknife was evaluated following a leave-one-out cross-validation (LOOCV) procedure (see Table 4.1 for a detailed description). The gesture template format was similar to the first two experiments but featured time-domain data instead of frequency-domain data for the background subtraction stage, which should be less sensitive to the number of sampling points and result in better accuracy across different users and materials.

Intra-material

This first study evaluates the accuracy of Jackknife on data recorded through three different materials, at two stages of the radar data processing pipeline and in two user scenarios. Data from the same material is used for training and testing. The study is within-factors with six independent variables:

1. `Processing Stage`: nominal variable with two conditions, representing the processing stage of the training data: time gating, filtering.

2. `User Scenario`: nominal variable with two conditions, representing the type of user scenario: user-dependent, user-independent.
3. `Material`: nominal variable with three conditions, representing the type of material used for both training and testing: wood, PVC, glass.
4. `Number of Sampling Points`: numerical variable with 37 values representing the number of points per gesture template: $N=\{x \in \mathbb{N} \mid 4 \leq x \leq 40\}$.
5. `Antenna Pairs`: nominal variable with three conditions, representing the selected combination of antenna pairs: $AP = \{(1,2,3,6,8,9), (4,5,7,10,11,12), (1,2,3,4,5,6,7,8,9,10,11,12)\}$.

The total number of training templates per trial is 44 (four to five per gesture class) in the user-dependent scenario and 855 (95 per gesture class) in the user-independent scenario.

Inter-material

This second study evaluates the accuracy of Jackknife when tested on data recorded through a different type of material than its training data, at two stages of the radar data processing pipeline in a user-dependent and cross-dataset scenario. The study is within-factors with four independent variables:

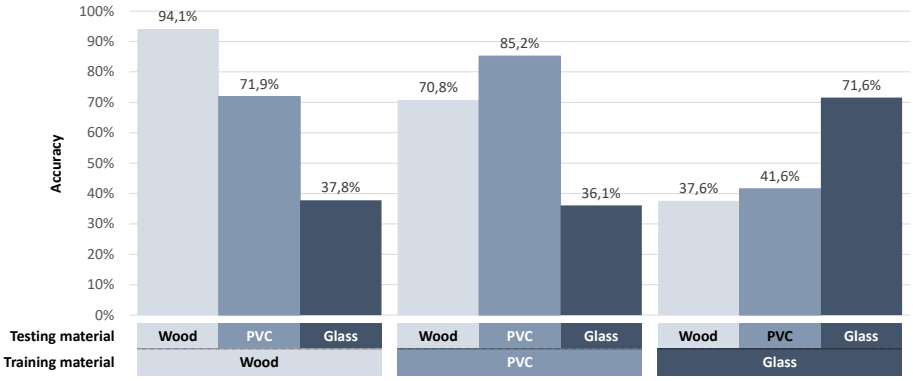
1. `Processing Stage`: same as the first study.
2. `Materials`: nominal variable with six conditions, representing the combination of materials used for training and testing the recognizer (*training+testing*): wood+PVC, wood+glass, PVC+wood, PVC+glass, glass+wood, and glass+PVC.
3. `Number of Sampling Points`: same as the first study.
4. `Antenna Pairs`: same as the first study.

The total number of training templates per trial is 45 (five per gesture class).

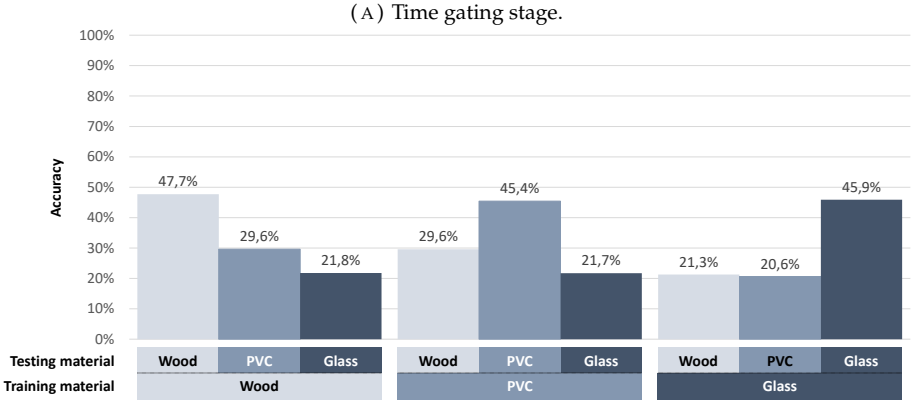
7.4.3 Results

Intra-material

Intra-material testing was conducted in three scenarios: user-dependent, user-independent, and mixed. Fig. 7.24, D.3, and D.5 depict the evolution of accuracy with respect to the number of sampling points in user-dependent and user-independent scenarios for the wood, PVC, and glass training/testing materials,

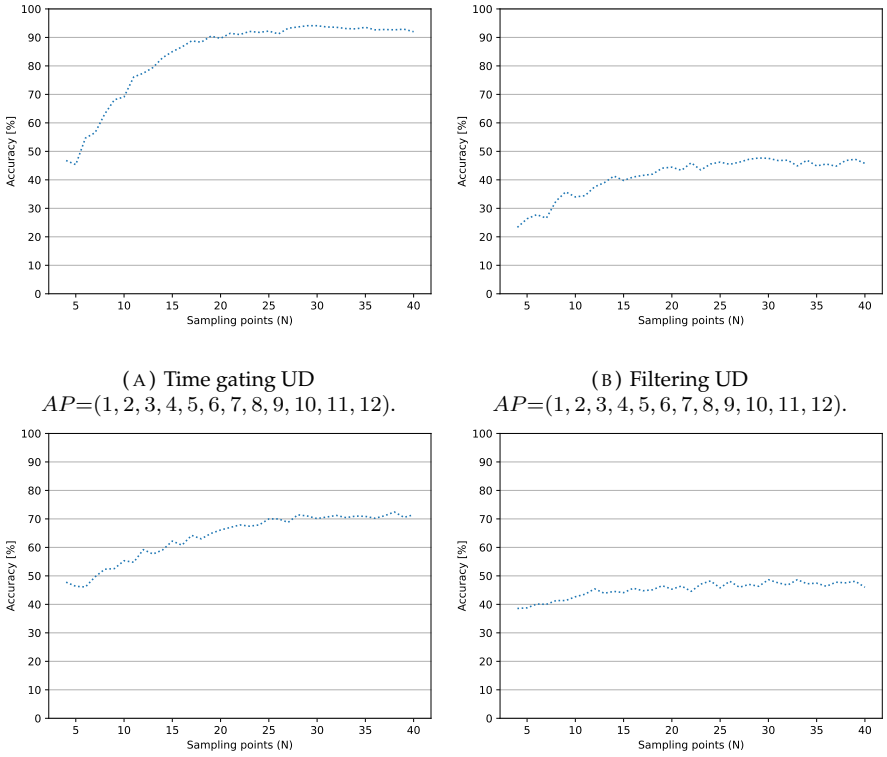


(A) Time gating stage.



(B) Filtering stage.

FIGURE 7.23: Average gesture recognition accuracy in the best configuration for each combination of training and testing materials in the user-dependent scenario at two stages of the pipeline.



(A) Time gating UD
AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(B) Filtering UD
AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(C) Time gating UI
AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(D) Filtering UI
AP=(1, 2, 3, 6, 8, 9).

FIGURE 7.24: The accuracy of Jackknife with respect to the number of sampling points for the best-performing set of antenna pairs when using wood data for training and testing in a user-dependent (top) and user-independent scenario (bottom).

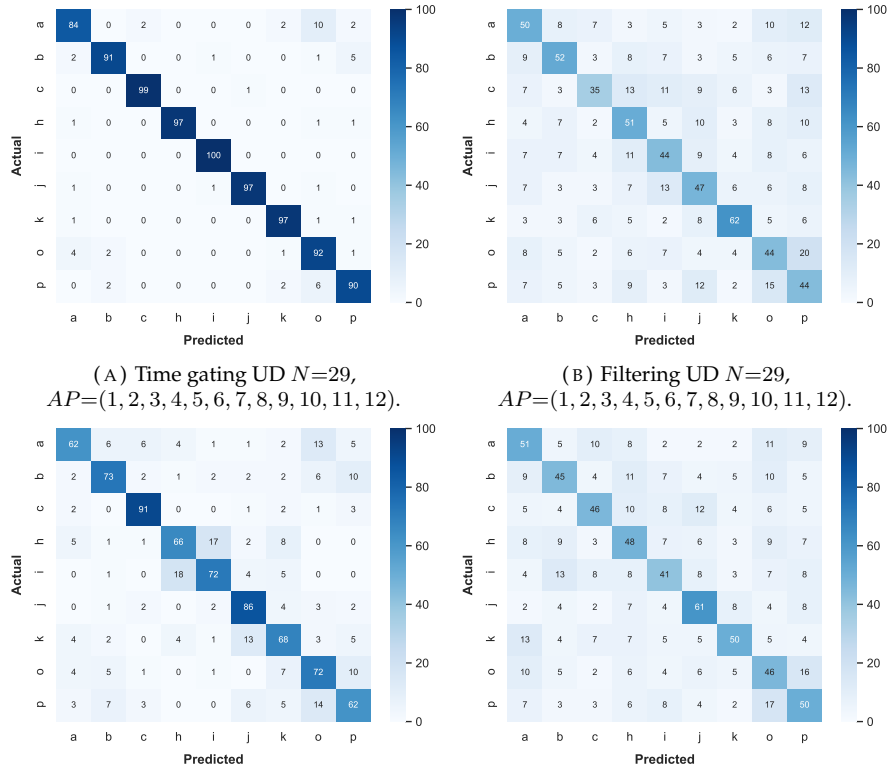
respectively. Similarly, Fig. 7.25, D.4, and D.6 show the confusion matrices of the best configurations in user-dependent and user-independent scenarios for the wood, PVC, and glass training/testing materials, respectively.

Time gating. Fig. 7.23a shows the average accuracy achieved by the best configuration for each combination of training and testing material with data from the time-gating stage in the user-dependent scenario. Overall, we observed that the average accuracy increased rather smoothly with the number of sampling points, plateauing at around 93%, 84%, and 70% for the wood, PVC, and glass, respectively. The best configuration achieved 94.1% accuracy for the

wood (N=29, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)), with all but one gesture correctly recognized more than 90% of the time. Average execution time was rather short, at 16.3 ms. We noticed some confusion between gestures a and o (open hand and push fist), as well as between b and p (close hand and push palm). In both cases, the gestures start with a similar hand pose and motion. For instance, "open hand" starts with the hand closed and moving slightly towards the radar before opening the hand, which is similar to the "push fist" gesture. Average accuracy dropped for the PVC and glass, at 85.2% (N=40, AP=(4, 5, 7, 10, 11, 12)), and 71.6% (N=38, AP=(1, 2, 3, 6, 8, 9)), respectively. As for the wood, gesture a (resp. b) was often confused with gesture o (resp. p). In addition, we noticed some confusion between gestures a, b, and c (open/close/open then close hand), and between gestures o and p (push fist/palm), especially for the glass. This indicates that differences in signal amplitude become harder to identify as the signal-to-noise ratio decreases. With the glass, we also observed some confusion between swipes (h, i, j) and "open/close/open then close hand" (a, b, c). This was also observed in the first experiment (Section 7.2.3) and probably stems from an inability of the radar to accurately distinguish between moving the hand in the plane orthogonal to the radar beam and changing its radar cross-section.

Average accuracy in the user-independent scenario was lower but behaved similarly to the user-dependent scenario, plateauing at around 70%, 63%, and 55% for the wood, PVC, and glass, respectively. This is better than in the second experiment (Section 7.3.3), which may be explained by the smaller set of gestures, the larger number of participants, and the use of time-domain data. The best configuration achieved 72.4% accuracy for the wood (N=38, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)). The average execution time was very long, at 342.2 ms. Our observations are mostly similar to the user-dependent scenario, with some confusion between gestures a and o (open hand and push fist), b and p (close hand and push palm), o and p (push fist/palm), j and k (swipe up/down), as well as between h and i (swipe right/left). Average accuracy also dropped for the best configurations of the PVC and glass, at 64.9% (N=34, AP=(4, 5, 7, 10, 11, 12)), and 55.7% (N=34, AP=(1, 2, 3, 6, 8, 9)), respectively. The corresponding confusion matrices are very similar to the wood, albeit with lower accuracy.

Filtering. Fig. 7.23b depicts the average accuracy achieved by the best configuration for each combination of training and testing material with data from the filtering stage in the user-dependent scenario. Accuracy dropped significantly compared to time-domain data from the time gating stage but still increased with the number of sampling points, plateauing at around 46% for the wood and 43% for the PVC and glass. The best configurations for the wood, PVC, and glass all performed similarly, reaching 47.7% (N=38, AP=(1, 2, 3, 4,



(A) Time gating UD N=29, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(B) Filtering UD N=29, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(C) Time gating UI N=38, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(D) Filtering UI N=30, AP=(1, 2, 3, 6, 8, 9).

FIGURE 7.25: Normalized confusion matrices for the best configuration at two stages of the pipeline in user-dependent (top) and user-independent scenarios (bottom) when using wood data for training and testing. The values in each cell are represented as percentages.

5, 6, 7, 8, 9, 10, 11, 12)), 45.4% (N=32, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)), and 45.9% (N=23, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)), respectively. Execution time was very short at 0.6 ms for the wood.

Accuracy in the user-independent scenario was slightly higher and similar across all materials, reaching 48.7% (N=30, AP=(1, 2, 3, 6, 8, 9)), 49.1% (N=38, AP=(1, 2, 3, 6, 8, 9)), and 51.1% (N=36, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)), for the wood, PVC, and glass, respectively. The average execution time was longer at 7.7 ms for the wood.

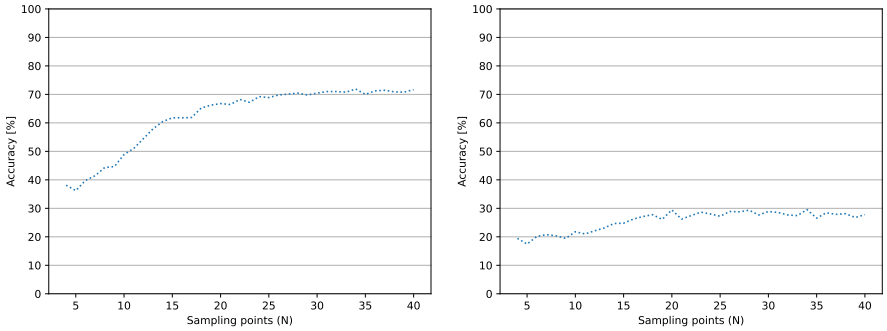
The confusion matrices in both scenarios are very similar and our observations from the time gating stage are also valid for this stage.

Inter-material

The inter-material testing was only conducted in a user-dependent scenario. Fig. 7.26, D.7, D.9, D.11,D.13, and D.15 depict the evolution of accuracy with respect to the number of sampling points for the wood+PVC, PVC+wood, wood+glass, glass+wood, PVC+glass, and glass+PVC training/testing materials, respectively. Similarly, Fig. 7.27, D.8, D.10, D.12,D.14, and D.16 show the confusion matrices of the best configurations for the wood+PVC, PVC+wood, wood+glass, glass+wood, PVC+glass, and glass+PVC conditions, respectively.

Time gating. Fig. 7.23b depicts the average accuracy achieved by the best configuration for each combination of training and testing material with data from the time gating stage. The results for the wood+PVC and PVC+wood combinations of training and testing materials are similar to those of the intra-material testing in the user-dependent scenario but with slightly lower accuracy. The best configurations reached 71.9% (N=34, AP=$(4, 5, 7, 10, 11, 12)$) and 70.8% accuracy (N=35, AP=$(4,\ 5,\ 7,\ 10,\ 11,\ 12)$) for wood+PVC and PVC+wood, respectively. The corresponding confusion matrices are very similar to those of the intra-dataset testing, and our observations are thus still valid here. Accuracy for wood+glass, glass+wood, pvc+glass and glass+pvc combinations of materials dropped drastically, at 37.8% (N=40, AP=$(1,\ 2,\ 3,\ 6,\ 8,\ 9)$), 37.6% (N=29, AP=$(1,\ 2,\ 3,\ 6,\ 8,\ 9)$), 36.1% (N=37, AP=$(1,\ 2,\ 3,\ 6,\ 8,\ 9)$), and 41.6% (N=37, AP=$(4,\ 5,\ 7,\ 10,\ 11,\ 12)$), respectively. This drop in accuracy seems to indicate that, despite the normalization applied by our processing pipeline, radar signals recorded through the glass were not comparable to those recorded through the wood and PVC. This may have been a consequence of the high relative permittivity of the glass which caused more energy to be reflected at the interface between air and glass. This, in turn, resulted in a lower signal-to-noise ratio. The smaller relative permittivity of wood and PVC resulted in a higher signal-to-noise ratio and thus more comparable radar signatures.

Filtering. Fig. 7.23b depicts the average accuracy achieved by the best configuration for each combination of training and testing material with data from the filtering stage. The results are similar to those obtained with data from the time gating stage, but with even lower accuracy, reaching only 29.6% accuracy with the wood+pvc (N=34, AP=$(1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ 11,\ 12)$) and pvc+wood materials (N=26, AP=$(1,\ 2,\ 3,\ 4,\ 5,\ 6,\ 7,\ 8,\ 9,\ 10,\ 11,\ 12)$). Because of the very low accuracy, it is difficult to draw any conclusion from the corresponding confusion matrices. Accuracy for wood+glass, glass+wood, pvc+glass and glass+pvc combinations of materials dropped by nearly 10%, at 21.8% (N=21, AP=$(1,\ 2,\ 3,\ 6,\ 8,\ 9)$), 21.3% (N=21, AP=$(1,\ 2,\ 3,\ 6,\ 8,\ 9)$),



(A) Time gating AP=$(4, 5, 7, 10, 11, 12)$.

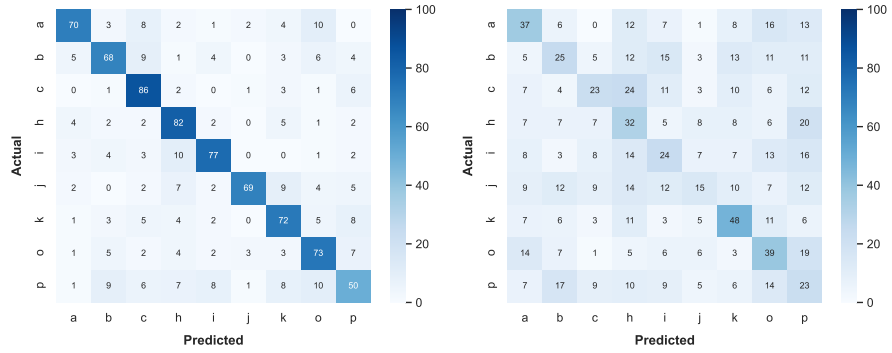
(B) Filtering AP=$(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)$.

FIGURE 7.26: The accuracy of Jackknife with respect to the number of sampling points for the best-performing set of antenna pairs when using wood data for training and PVC data for testing in a user-dependent scenario.

21.7% (N=20, AP=$(1,\ 2,\ 3,\ 6,\ 8,\ 9)$), and 20.6% (N=36, AP=$(1,\ 2,\ 3,\ 6,\ 8, 9)$), respectively. Similarly to the time gating stage, the low accuracy could be explained by the lower signal-to-noise ratio of the signals recorded through glass, which may have caused the inversion process to fail more often than for the other materials.

7.4.4 Summary

This experiment demonstrated that a relatively inexpensive radar like the Walabot could perform relatively well through some types of materials, including wood and PVC. In addition, our signal-processing pipeline was able to normalize the signals from the two materials, achieving a fairly high accuracy of around 71% across these materials. The results of Leiva *et al.*'s study about radar sensing through fabric [160] suggest that training the gesture recognition algorithm on a larger set of samples without material obstruction could further enhance accuracy across all tested materials. This avenue would be worth investigating in future research, as combining this technique with the data normalization applied by our pipeline could lead to even greater gesture recognition accuracy through different materials. The higher relative permittivity of the glass compared to the other materials resulted in a lower accuracy. While the average accuracy still reached 71.6% in the user-dependent scenario when using glass data for training and testing, it dropped to around 38% when mixing glass with another material. This indicates that our processing pipeline could not properly normalize signals recorded through the glass, probably



(A) Time gating N=34, AP=(4, 5, 7, 10, 11, 12).

(B) Filtering N=34, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

FIGURE 7.27: Normalized confusion matrices for the best configuration at two stages of the pipeline in the user-dependent scenario when using wood data for training and PVC data for testing. The values in each cell are represented as percentages.

because of the lower signal-to-noise ratio. Other radar systems operating at a different power level and/or frequency range may alleviate this issue. Finally, our results were more stable with respect to the number of sampling points than in the previous experiments, suggesting that the time-domain data we used in this experiment may be better suited to gesture recognition than frequency-domain data.

7.5 Discussion and Implications for Design

In this section, we draw ten implications from our three experiments for the design of hand gesture interaction with radar sensors. These implications are categorized into four key areas: (1) radar systems (R1), (2) gesture sets (G1-G5), (3) signal processing (S1-S2), and (4) environment of interaction (E1-E2).

R1. Use multiple antennas with enough separation. Our first experiment (Section 7.2.3) showed that taking advantage of multiple pairs of antennas could improve gesture recognition accuracy, as more information was captured that could help differentiate between similar gestures. Overall, the number of antennas seemed to be a more relevant parameter than the quality of the antennas, especially when using data from the full-wave inversion and filtering stages.

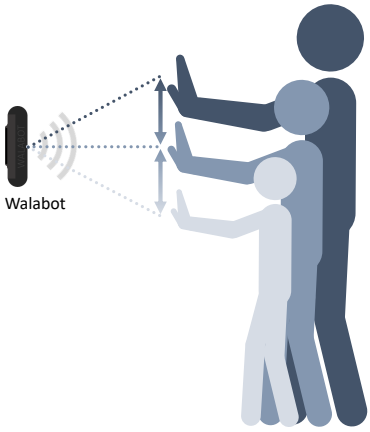


FIGURE 7.28: Examples of variations in hand position and surface.

G1. Collect gestures from multiple users. Hands can vary widely in size and surface area between users. The length of the hand can range from 15 to 30 cm (5.9 to 11.8 in) and the circumference of the hand from 15 to 28 cm (5.9 to 11.02 in), giving an idea of the potential surface range. Hand and palm surface areas vary with gender, age, and morphology. For example, [91] reported that the average hand surface area for Turkish women and men is M=158.34 cm^2 (SD=14.76 cm^2) and M=127.87 cm^2 (SD=14.75 cm^2), respectively, and palm surface area M=63.91 cm^2 (M=11.84 cm^2) and M=82.98 cm^2 (SD=10.74 cm^2), respectively. Variations in the hand surface area result in variations in the radar signal that should be taken into account to ensure high recognition accuracy, especially for the user-independent scenario. Other parameters can equally vary between different users, such as arm length and position relative to the radar (*e.g.*, elevation, distance), which may also result in signal variations (Fig. 7.28). Gesture training sets for user-independent recognition should include samples produced by a variety of users.

G2. Support user customization. Our testing showed that gesture recognition accuracy in user-independent scenarios was too low to be usable in real applications. Conversely, our system performed very well in user-dependent and even mixed scenarios. Using small gesture sets that contain training samples produced by the current user of the system thus seems to be the key to achieving high gesture recognition accuracy. The ability of our system to work with very few training samples facilitates this, as it takes only a few minutes for users to generate their own gesture set. In addition, user-defined gestures usually result in higher satisfaction and memorability [203].

G3. Favor gestures with motion parallel to the radar beam. As the angular resolution of a radar is significantly lower than its range resolution, it is easier to recognize gestures that are performed in a parallel plane with respect to the radar than in other configurations, such as the orthogonal plane. As a negative example, the "swipe left/right/up/down" and "open/close hand" gestures are more difficult to differentiate, especially with only one pair of antennas (Section 7.2.3). As a positive example, "push with palm/fist" will be easier to differentiate from other gestures (*e.g.*, "pull"), even with one antenna pair due to the relatively good range resolution of the radar.

G4. Favor gestures with a highly differentiable surface of exposure. For example, in the user-dependent scenario, there was little to no confusion between the "push with palm" and "push with fist" gestures (Fig. 7.18b, 7.20b, 7.21a) because the palm has a larger exposure surface than the fist, resulting in a larger amplitude of the received radar signal. This difference in amplitude can be used by a recognizer to differentiate between the two gestures. On the other hand, gestures such as "extend one/two/three/four fingers" were less accurately recognized, especially in the user-independent scenario (Fig. D.1a) because the changes in their surfaces of exposure were relatively small.

G5. Select gestures based on the level of criticality of your application. For non-critical applications (*e.g.*, interacting with a TV at home), gestures with lower accuracy, such as "open/close/open then close hand" and "extend one/two/three/four fingers", may be selected if they are well suited to perform a specific action. In such applications, an incorrect result from the recognizer would only impact the user experience of the application. For safety-critical applications, such as performing high-precision medical procedures, developers should rely on small gesture sets composed exclusively of gestures that can be recognized with very high accuracy (> 99%). To guarantee high accuracy, it is strongly advised to train the system with data from the same person(s) that would use the system, as an incorrectly recognized gesture in these applications could have life-threatening consequences.

S1. Use only the steps of the recognition pipeline that are necessary for a given application. Our experiments showed that each stage of the pipeline could provide some benefits for different applications. For instance, trilateration can be used to compute the position of the hand in space based on the signal produced by the full-wave inversion and filtering stages (Section 7.2.4). This has many applications, such as displaying a cursor to facilitate UI interactions, improving the recognition of 2D/3D paths, or segmenting gestures from the continuous flow of data based on the position of the hand in space.

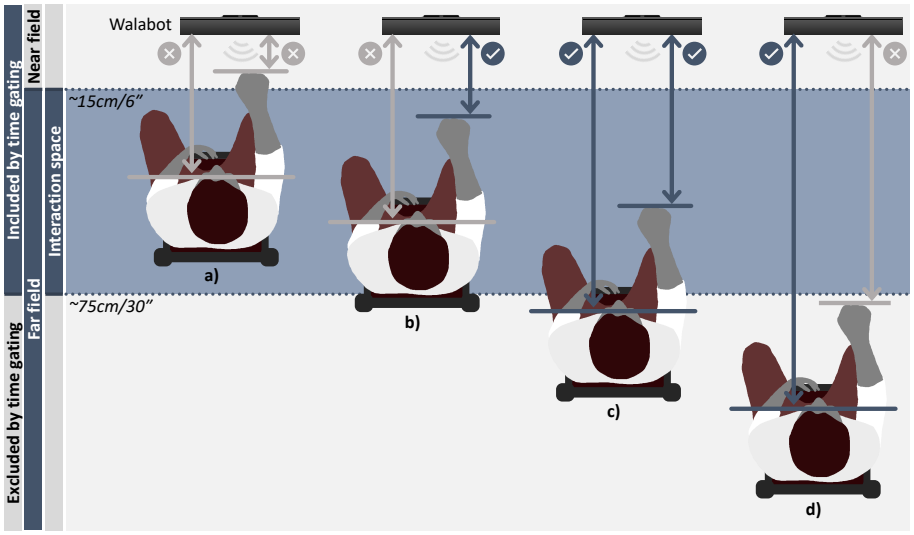


FIGURE 7.29: Distance constraints for radar-based gestures.

Time-domain data from the time-gating stage seems better suited for recognizing more complex gestures, especially in user-dependent and mixed scenarios (Section 7.3.4). Because the radar signals output by this step are normalized and thus free of antenna effects, noise, and clutter, they would also work great with image classification algorithms, such as Convolutional Neural Networks (CNNs), especially if their training set features gestures captured with different sensors.

S2. Prefer time-domain to frequency-domain data. Compared to the first two experiments, our third experiment (Section 7.4.3) showed that gesture recognition accuracy was more stable with respect to the number of sampling points when using time-domain data, as temporal information is better preserved when the signal is resampled to a lower number of points. In addition, time-domain data is relatively easy to interpret by humans, which can be an advantage when designing a highly differentiable gesture set.

E1. Avoid materials with high relative permittivity. Our third experiment showed that we could still achieve relatively accurate gesture recognition when the radar was obstructed by some material like wood or PVC (Section 7.4.3). However, materials with a higher relative permittivity (*e.g.*, glass) will reflect a larger fraction of the radar signal, resulting in a lower signal-to-noise ratio which will impact gesture recognition accuracy. As such, materials with a high

relative permittivity should be avoided to maximize the accuracy of the system, and thus, its usability.

E2. Ensure that hand gestures are performed within the appropriate range of the radar sensor. Hand gestures should be performed neither too close to the radar (otherwise, a more complicated near-field algorithm should be employed to generate the model of the radar and remove radar source and antenna effects [148]) nor too far away (otherwise, the reflection may be too weak, reaching the noise level of the radar, and past the time gating threshold). Fig. 7.29 shows both correct and wrong examples of distances: (a) the hand and body are both too close to the radar, (b) the hand is far enough from the radar (within 15-75 cm) but the body is not excluded by the time gating, (c) the hand and body are both within an appropriate range, and (d) the hand and body are both out of range. This is especially visible in Fig. 7.8, which shows that most of the reflections from gestures u and v (barrier gesture and touch nose) are outside of the time gating range. In particular, the captured radar signal from gesture u barely sits above the noise level. If too many gestures are performed outside of the range of the sensor, it could result in lower recognition accuracy.

7.6 Conclusion

In this chapter, we evaluated the efficacy of the radar processing pipeline proposed in Chapter 6 by conducting three experiments. The first experiment analyzed the performance of two radar sensors alongside the LMC, a popular vision-based sensor, on a dataset featuring 16 gestures. Our findings revealed that inexpensive off-the-shelf radars like the Walabot could achieve better accuracy than a higher-end single-antenna radar and even outperformed the LMC. The second experiment evaluated the Walabot in three user scenarios on a large set of 20 gestures and a selection of well-differentiated gesture subsets. Its results suggested that employing smaller, well-differentiated sets of gestures, and supporting user-defined gestures could be the key to building reliable radar-based gesture recognition. The third experiment analyzed the impact of occlusion by three different types of materials (wood, PVC, and glass) on gesture recognition accuracy. Its results indicated that our pipeline could successfully normalize radar signals if the signal-to-noise ratio was high enough, like with the wood and PVC, which reflected less of the signal due to their low relative permittivity. Based on our observations from the three experiments, we drew ten implications for the design of hand gesture interaction with radar sensors. These observations related to four areas of radar-based gesture recognition: (1) radar systems, (2) gesture sets, (3) signal processing, and (4) environment of interaction.

This chapter combined multiple tools introduced in the previous chapters, such as QuantumLeap and its testing tool (Chapters 3 and 4), and our radar signal processing pipeline (Chapter 6), serving as a great example of interoperability and reusability, which are sub-properties of *compatibility* and *maintainability*, respectively [116]. In addition, our use of the QuantumLeap testing tool for conducting these experiments allows researchers to easily reproduce (to check if they can get the same results in the same conditions) and replicate our results (to check if they get similar results under different conditions). Finally, while the outcomes of this chapter are less tangible than the LUI application developed in Chapter 5 or the QuantumLeap framework from Chapter 3, the experiments conducted in this section are a necessary step towards the development of practical, highly usable, radar-based interfaces.

This work has multiple potential avenues for future research. First, we could analyze the performance of other radar systems. Radars featuring multiple antennas separated by a large distance or phased arrays may be prime candidates, as they could enable better angular resolution. Another interesting direction for future work would be to investigate the impact of radar placement at different locations [258] on gesture recognition accuracy. This chapter focused on template matching techniques for gesture recognition, in particular, the Jackknife recognizer [284]. While it performed great in some of our tests with relatively few training samples, it would be interesting to investigate the performance of other techniques, such as CNNs or LSTMs, at different stages of our pipeline. Finally, we could investigate other gesture sets, either by expanding our list of hand gestures or by focusing on different types of gestures, such as breathing patterns, whole-body gestures, or even tongue gestures. We could also record gestures in a less controlled environment featuring more clutter and people passing by in the background.

Chapter 8

Conclusion

This thesis had two key objectives: (1) explore the use of radar for gesture recognition and (2) provide tools and methods to facilitate the development of highly usable (radar-based) gesture interfaces. This chapter concludes our thesis in two sections. Section 8.1 summarizes our findings and contributions, and Section 8.2 discusses potential avenues for future research.

8.1 Summary

In this section, we summarize the findings and contributions from the six main chapters of this thesis.

Chapter 2: State of the Art

This chapter explored the state of the art of gesture recognition through targeted and systematic literature reviews.

We first conducted a TLR to gain a general understanding of gesture-based interaction. We recognized the efficacy of GESs for involving end users in the development of highly usable gesture-based applications, as user-elicited gestures with high agreement have been associated with higher memorability [302]. We also identified the key challenges of gesture interaction and potential solutions. These challenges included sensor limitations, jitter caused by tracking errors, noise, or hand tremors, intra- and inter-user variability of gesture production, gesture segmentation, and overlapping gestures. Effectively addressing these challenges while factoring in user preferences requires a high level of expertise in UX design, gesture recognition, and application development. This high entry barrier and the lack of accessible tools for developers might explain why we identified relatively few gesture-based applications in the literature, despite their broad appeal across many facets of society.

We then carried out an SLR on LMC-based gesture recognition. This choice to analyze a relatively mature vision-based technology like the LMC aimed to

provide insights for contextualizing the current state of the research in radar-based gesture interaction. We observed that, despite the popularity and ease of use of the LMC, none of the papers identified in the SLR proposed an approach that was reusable across gesture sets, gestural UIs, and applications. On the contrary, a large proportion of these papers resorted to opportunistic techniques for gesture recognition, such as hard-coded thresholds and sensor-specific gestures, which are difficult to maintain and lack portability.

With this context in mind, we undertook a second SLR to gain a comprehensive understanding of the current state of research in radar-based gesture interaction. We noticed that most of the research focused on radar systems, signal processing techniques, and algorithms for gesture recognition, with only a handful of papers demonstrating potential use cases of their systems with prototype applications. In addition, the scarcity of publicly available datasets in these papers hinders collaboration among researchers, while the plethora of sensors used, often custom-designed, and the reliance on deep learning algorithms for gesture recognition, makes it difficult to apply these results to a different environment.

Finally, we concluded this chapter by positioning our work within the existing literature in the light of our findings.

Chapter 3: Facilitating the Development of Gesture-based Applications

This chapter introduced QuantumLeap, a modular framework designed to bridge the gap between researchers and practitioners to foster the development of gesture-based applications.

Its modular architecture encapsulates all of the gesture recognition logic and only exposes a simple API to the application, which allows developers to map gestures to actions without focusing on acquiring, segmenting, and recognizing gestures. The configuration GUI of QuantumLeap makes it easy for developers to set up the gesture recognition dataflow for their needs without having to modify its source code. Additionally, its modular architecture facilitates future updates to the dataflow, such as an improved gesture recognizer or a new gesture set.

We evaluated the framework among seven developers with varying levels of expertise, which resulted in a good usability rating. Despite the limitations of this study, such as the excessive guidance provided to the participants, it offered valuable insights for future improvements to QuantumLeap. They include providing more guidance to users when configuring the dataflow (*e.g.*, module recommendations) and addressing the rigidity of the QuantumLeap dataflow architecture to support more complex gesture recognition techniques.

Despite these limitations, the QuantumLeap framework was used by many students over the years, who contributed to the project by implementing new modules and developing unique gesture-based applications.

Chapter 4: Streamlining the Evaluation of Gesture Recognizers

This chapter introduced the QuantumLeap testing tool, an extension of QuantumLeap designed for evaluating the performance of gesture recognizers.

The tool relies on the same modules and datasets as QuantumLeap and supports ten distinct testing procedures. In addition, its testing configurations and results can be exported in a standardized format, which is extremely convenient as researchers can now easily share all of the elements required to attempt to repeat, replicate, and reproduce their results.

The tool inherits some of the limitations of QuantumLeap, including its relatively rigid architecture, which complicates the evaluation of more complex gesture recognition pipelines. Despite its limitations, it has been used extensively in the rest of the thesis.

Chapter 5: Designing Highly Usable Gesture-based Applications

This chapter proposed a user-centered iterative development method and evaluation protocol for gesture-based applications, designed to facilitate the creation of highly usable gestural interfaces.

The proposed method comprises three key steps. First, a consistent set of gestures is designed, incorporating user preferences derived from the literature or a GES. Then, a gesture recognition dataflow is implemented, *e.g.*, utilizing our QuantumLeap framework. An optimal dataflow configuration may be identified through a benchmarking of gesture recognizers on the selected gesture set. Finally, an experiment is conducted to assess the usability of the resulting application. Depending on the experiment's outcome, a new iteration of this process may be started to improve the application based on these new insights.

We successfully applied this method to LUI, a gesture-based application for browsing and manipulating multimedia content, such as images, videos, documents, and maps. Implementing gesture recognition through QuantumLeap proved instrumental in supporting complex gestures with relatively low effort, such as the manipulation of multimedia objects with immediate and continuous feedback. In addition, the separation of concerns provided by the framework enables users to customize the gesture set of LUI without having to delve into its source code. Finally, the consistent set of gestures featured in LUI resulted in a relatively effective and usable interface, as indicated by our usability evaluation conducted among 17 participants. Some limitations

were also highlighted, including an unpredictability in gesture recognition for certain gestures, attributed in part to slight variations in the way they were performed by the participants. This emphasizes the importance of supporting user-defined gestures.

Chapter 6: Addressing the Challenges of Radar-based Gesture Recognition

This chapter introduced a full-wave electromagnetic model and inversion-based [149, 148] radar signal processing pipeline, aimed at addressing some of the challenges of radar-based gesture recognition. In particular, the variance between radar systems and their sensitivity to objects in the environment.

The pipeline consists of six stages: (1) raw data capture, (2) radar source and antenna effects removal, to remove internal reflections and transmissions between antennas, as well as antenna-target interactions, (3) background scene removal, to remove the reflections of static objects in the vicinity of the radar, (4) time gating, to ignore remaining reflections from objects situated past a certain distance from the radar, such as people moving behind the user, (5) full-wave inversion, to reduce the radar signal to only two physically-meaningful metrics (distance and permittivity), and (6) filtering, to smooth out errors in the inversion process and discard unrealistic values.

Two small experiments were conducted to (1) analyze whether the pipeline could effectively normalize signals from two distinct radars, and (2) investigate the sensitivity of our pipeline to variations in radar-target distance and angle. Although the results of our first experiment were conclusive, the second experiment showed that signals from our processing pipeline changed with both radar-target distance and angle.

Different implementations of the pipeline were provided for batch and real-time signal processing, and a prototype application was created to demonstrate potential use cases of the pipeline.

Chapter 7: Experimenting with Radar-based Gestures

This chapter investigated the efficacy of our radar signal processing pipeline in three experiments.

The first experiment compared the performance of three different sensors (an LMC, a custom radar, and a Walabot) on a dataset of 16 gestures. Its results revealed that the Walabot could achieve better accuracy than the LMC and the custom radar, suggesting that inexpensive antenna arrays could be used successfully for gesture interaction.

The second experiment evaluated the Walabot in user-dependent, user-independent, and mixed scenarios on an extended dataset of 20 gestures and well-differentiated subsets of gestures. The results suggested that employing

smaller, well-differentiated sets of gestures, coupled with supporting user-defined gestures, could be the key to building reliable radar-based gesture recognition.

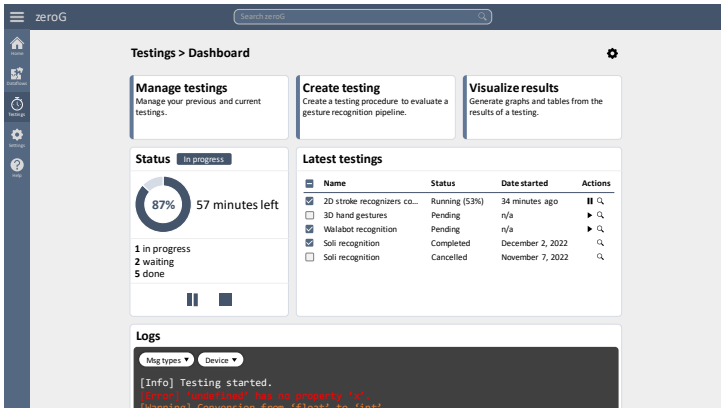
The third experiment analyzed the impact of occlusion by three different types of materials (wood, PVC, and glass) on a dataset of nine gestures. It showed that our pipeline could successfully normalize radar signals, provided that the signal-to-noise ratio was high enough. Indeed, the system still achieved high accuracy when obstructed by materials with a low relative permittivity like wood and PVC.

We drew ten implications for radar-based gesture recognition from our experiments, which we categorized into four key areas. The first implication relates to the radar systems, recommending the use of multiple spatially separated radar antennas when possible. Several implications address gesture sets, in particular, how they can be designed to maximize gesture recognition accuracy. They underscore the importance of collecting gestures from a wide range of users and supporting user-defined gestures. They also suggest favoring gestures with motion parallel to the radar beam and with highly differentiable surfaces of exposure. Two implications relate to the signal processing pipeline, advising developers to carefully select the necessary stages of the pipeline for their specific needs and suggesting the use of time-domain data instead of frequency-domain. The last two implications consider the environment of interaction, cautioning against the use of materials with high relative permittivity and stressing the importance of performing gestures within the appropriate range of the radar. These implications can serve as a valuable guide for developers when designing radar-based gestural interfaces.

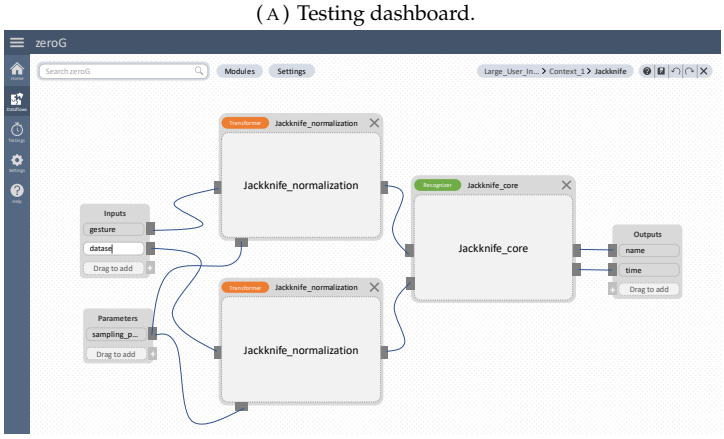
8.2 Future Works

We envision three key avenues for future research to address the main limitations of this thesis.

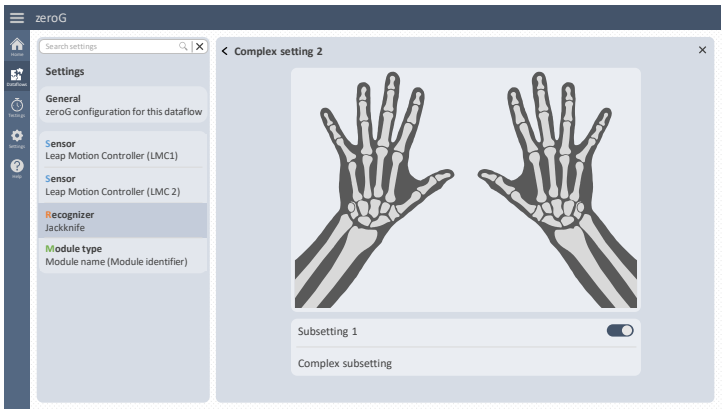
Regarding tools for the development of gesture-based applications, we are currently working on a new framework for gesture recognition, named *zeroG*, that aims to address the main limitations of QuantumLeap and its testing tool through profound architectural enhancements. The *zeroG* framework (Fig. 8.1) would rely on a flexible architecture based on standardized modules that could be assembled to construct complex dataflows for gesture recognition (Fig. 8.1b). This architecture would cater to users of varying expertise levels: (1) experts could manually create dataflows from scratch by freely combining modules, (2) advanced users could adapt predefined dataflow templates for their applications by filling placeholders with suitable module implementations, similarly to QuantumLeap, and (3) novice users could ask *zeroG* to suggest (or even



(A) Testing dashboard.



(B) Dataflow creation.



(C) Dataflow configuration.

FIGURE 8.1: Mockups of the zeroG UI.

generate) functioning dataflows based on some criteria like the type of application, sensor(s), and gestures. Overall, the configuration GUI (Fig. 8.1c) should provide ample guidance to users in the form of tutorials, tips, and module suggestions. To expedite the development process of gesture-based applications, *zeroG* could generate scaffolding code to perform, among other, initialization steps like connecting to the framework and submitting a dataflow configuration. In line with the *compatibility* software quality property, *zeroG* should seamlessly support the use of multiple applications on the same platform, without requiring user involvement. As such, it should enable applications to submit their dataflow configuration to the framework upon connection. Later developments could extend to building an entire ecosystem around the framework, including databases of modules, gestures, and dataflows. This would facilitate access to components for gesture recognition and would allow *zeroG* to automatically download missing components required by an application.

Future research could also focus on refining our pipeline for radar signal processing. This includes adding some form of user normalization to account for anatomical differences between users, as well as modifying the full-wave inversion stage to take into account the finite size of the hand, thus improving gesture recognition accuracy when the distance between the user and the radar is not fixed. In addition, we could add a stage at the end of the pipeline to compute the 3D position of the hand with trilateration, which would enable applications to display a cursor on the screen, and could improve the recognition of 3D hand trajectories. Similarly, we could apply sensor fusion to improve the robustness of a gesture recognition system under challenging environmental conditions. We could also investigate the possibility of recognizing gestures performed by more than one user concurrently with a single radar system and look into gesture segmentation techniques for radar to facilitate real-time gesture interaction, *e.g.*, using the QuantumLeap framework.

Finally, future work could continue the exploration of radar-based gesture recognition through experimentation. For instance, we could assess the evaluate the performance of alternative radar systems, such as phased arrays or radars featuring more spaced-out antennas, and look into the impact of radar placement (*e.g.*, on the wall, under a desk, or worn by the user) on gesture recognition. We could also investigate other types of gestures, including breathing patterns (an interesting alternative to the "sip-and-puff" assistive technology), whole body gestures, or tongue gestures, and record them in more challenging environments, such as public spaces. Additionally, we could compare the performance of Jackknife for gesture recognition with other techniques, including other template-matching algorithms and deep learning techniques, like CNNs or LSTMs.

Bibliography

[1] Christopher Ackad, Andrew Clayphan, Martin Tomitsch, and Judy Kay. "An In-the-Wild Study of Learning Mid-Air Gestures to Browse Hierarchical Information at a Large Interactive Public Display". In: *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (Osaka, Japan). UbiComp '15. New York, NY, USA: Association for Computing Machinery, 2015, pp. 1227–1238. ISBN: 9781450335744. DOI: 10.1145/2750858.2807532. URL: https://doi.org/10.1145/2750858.2807532.

[2] Gianluca Agresti and Simone Milani. "Material Identification Using RF Sensors and Convolutional Neural Networks". In: *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing* (Brighton, UK). ICASSP 2019. Piscataway, NJ, USA: IEEE, 2019, pp. 3662–3666. DOI: 10.1109/ICASSP.2019.8682296.

[3] Misbah Ahmad, Milind Ghawale, Sakshi Dubey, Ayushi Gupta, and Poonam Sonar. "GigaHertz: Gesture Sensing Using Microwave Radar and IR Sensor with Machine Learning Algorithms". en. In: *Image Processing and Capsule Networks*. Ed. by Joy Iong-Zong Chen, João Manuel R. S. Tavares, Subarna Shakya, and Abdullah M. Iliyasu. Vol. 1200. Series Title: Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2021, pp. 422–434. ISBN: 978-3-030-51858-5 978-3-030-51859-2. DOI: 10.1007/978-3-030-51859-2_39. URL: http://link.springer.com/10.1007/978-3-030-51859-2_39 (visited on 12/21/2020).

[4] Shahzad Ahmed and Sung Ho Cho. "Hand Gesture Recognition Using an IR-UWB Radar with an Inception Module-Based Classifier". In: *Sensors* 20.2 (2020), pp. 1–18. ISSN: 1424-8220. DOI: 10.3390/s20020564. URL: https://www.mdpi.com/1424-8220/20/2/564.

[5] Shahzad Ahmed, Faheem Khan, Asim Ghaffar, Farhan Hussain, and Sung Ho Cho. "Finger-Counting-Based Gesture Recognition within Cars Using Impulse Radar with Convolutional Neural Network". In: *Sensors* 19.6 (2019), pp. 1–14. ISSN: 1424-8220. DOI: 10.3390/

s19061429. URL: https://www.mdpi.com/1424-8220/19/6/1429.

[6] Shubhra Aich, Jesus Ruiz-Santaquiteria, Zhenyu Lu, Prachi Garg, K. J. Joseph, Alvaro Fernandez Garcia, Vineeth N. Balasubramanian, Kenrick Kin, Chengde Wan, Necati Cihan Camgoz, Shugao Ma, and Fernando De La Torre. "Data-Free Class-Incremental Hand Gesture Recognition". In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2023, pp. 20901–20910. DOI: 10.1109/ICCV51070.2023.01916. URL: https://doi.ieeecomputersociety.org/10.1109/ICCV51070.2023.01916.

[7] Roland Aigner, Daniel Wigdor, Hrvoje Benko, Michael Haller, David Lindbauer, Alexandra Ion, Shengdong Zhao, and Jeffrey Tzu Kwan Valino Koh. *Understanding Mid-Air Hand Gestures: A Study of Human Preferences in Usage of Gesture Types for HCI*. Tech. rep. MSR-TR-2012-111. Nov. 2012. URL: https://www.microsoft.com/en-us/research/publication/understanding-mid-air-hand-gestures-a-study-of-human-preferences-in-usage-of-gesture-types-for-hci/.

[8] Florian Alt, Sabrina Geiger, and Wolfgang Höhl. "ShapelineGuide: Teaching Mid-Air Gestures for Large Interactive Displays". In: *Proceedings of the 7th ACM International Symposium on Pervasive Displays*. PerDis '18. Munich, Germany: Association for Computing Machinery, 2018. ISBN: 9781450357654. DOI: 10.1145/3205873.3205887. URL: https://doi.org/10.1145/3205873.3205887.

[9] Moeness G. Amin, Zhengxin Zeng, and Tao Shan. "Arm Motion Classification Using Curve Matching of Maximum Instantaneous Doppler Frequency Signatures". In: *Proceedings of the IEEE International Radar Conference*. RADAR '20. ISSN: 2640-7736. Apr. 2020, pp. 303–308. DOI: 10.1109/RADAR42522.2020.9114779.

[10] Moeness G. Amin, Zhengxin Zeng, and Tao Shan. "Hand Gesture Recognition based on Radar Micro-Doppler Signature Envelopes". In: *Proceedings of the IEEE Int. Radar Conference* (Boston, MA, USA). RADAR '19. ISSN: 2375-5318. Piscataway, NJ, USA: IEEE, Apr. 2019, pp. 1–6. DOI: 10.1109/RADAR.2019.8835661.

[11] Moeness G. Amin, Zhengxin Zeng, Tao Shan, and R.G. Guendel. "Automatic Arm Motion Recognition Using Radar for Smart Home Technologies". In: *Proceedings of the IEEE International Radar Conference* (Boston, MA, USA). RADAR '19. ISSN: 2640-7736. Piscataway, NJ, USA: IEEE, Sept. 2019, pp. 1–4. DOI: 10.1109/RADAR41533.2019.171318.

[12] Michelle Annett, Albert Ng, Paul Dietz, Walter F. Bischof, and Anoop Gupta. "How Low Should We Go? Understanding the Perception of Latency While Inking". In: *Proceedings of Graphics Interface 2014* (Montreal, Quebec, Canada). GI '14. CAN: Canadian Information Processing Society, 2014, pp. 167–174. ISBN: 9781482260038.

[13] Lisa Anthony, Radu-Daniel Vatavu, and Jacob O. Wobbrock. "Understanding the Consistency of Users' Pen and Finger Stroke Gesture Articulation". In: *Proceedings of Graphics Interface 2013* (Regina, Sascatchewan, Canada). GI '13. CAN: Canadian Information Processing Society, 2013, pp. 87–94. ISBN: 9781482216806.

[14] Lisa Anthony and Jacob O. Wobbrock. "$N-ProTractor: A Fast and Accurate Multistroke Recognizer". In: *Proceedings of Graphics Interface 2012* (Toronto, Ontario, Canada). GI '12. Toronto, Ont., Canada: Canadian Information Processing Society, 2012, pp. 117–120. ISBN: 978-1-4503-1420-6. URL: http://dl.acm.org/citation.cfm?id=2305276.2305296.

[15] Dane Archer. "Unspoken Diversity: Cultural Differences in Gestures". In: *Qualitative Sociology* 20 (1997), pp. 79–105. DOI: 10.1023/A:1024716331692. URL: https://api.semanticscholar.org/CorpusID:142578394.

[16] Muhammad Arsalan and Avik Santra. "Character Recognition in Air-Writing Based on Network of Radars for Human-Machine Interface". In: *IEEE Sensors Journal* 19.19 (Oct. 2019), pp. 8855–8864. ISSN: 1558-1748. DOI: 10.1109/JSEN.2019.2922395.

[17] Kongphum Arthamanolap, Somprasong Gabbualoy, and Pattarapong Phasukkit. "Doppler Radar for Dynamic Hand Gesture Recognition based on Signal Image Processing". In: *Proceedings of 16th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology* (Pattaya, Thailand). ECTI-CON'19. Piscataway, NJ, USA: IEEE, July 2019, pp. 931–934. DOI: 10.1109/ECTI-CON47248.2019.8955217.

[18] Daniel Ashbrook and Thad Starner. "MAGIC: A Motion Gesture Design Tool". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA). CHI '10. New York, NY, USA: Association for Computing Machinery, 2010, pp. 2159–2168. ISBN: 9781605589299. DOI: 10.1145/1753326.1753653. URL: https://doi.org/10.1145/1753326.1753653.

[19] Daniel Avrahami, Mitesh Patel, Yusuke Yamaura, and Sven Kratz. "Below the Surface: Unobtrusive Activity Recognition for Work Surfaces Using RF-Radar Sensing". In: *Proc. of the 23rd ACM International Conference on Intelligent User Interfaces* (Tokyo, Japan). IUI '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 439–451. ISBN: 9781450349451. DOI: 10.1145/3172944.3172962. URL: https://doi.org/10.1145/3172944.3172962.

[20] Daniel Avrahami, Mitesh Patel, Yusuke Yamaura, Sven Kratz, and Matthew Cooper. "Unobtrusive Activity Recognition and Position Estimation for Work Surfaces Using RF-Radar Sensing". In: *ACM Trans. Interact. Intell. Syst.* 10.1 (Aug. 2019). ISSN: 2160-6455. DOI: 10.1145/3241383. URL: https://doi.org/10.1145/3241383.

[21] Daniel Bachmann, Frank Weichert, and Gerhard Rinkenauer. "Review of Three-Dimensional Human-Computer Interaction with Focus on the Leap Motion Controller". In: *Sensors* 18.7 (July 2018), p. 2194. ISSN: 1424-8220. DOI: 10.3390/s18072194. URL: http://dx.doi.org/10.3390/s18072194.

[22] Aaron Bangor, Philip Kortum, and James Miller. "An Empirical Evaluation of the System Usability Scale". In: *International Journal of Human–Computer Interaction* 24.6 (2008), pp. 574–594. DOI: 10.1080/10447310802205776. URL: https://doi.org/10.1080/10447310802205776.

[23] Alan Bannon, Richard Capraru, and Matthew Ritchie. "Exploring gesture recognition with low-cost CW radar modules in comparison to FMCW architectures". In: *Proceedings of the IEEE International Radar Conference* (Washington, DC, USA). RADAR'20. ISSN: 2640-7736. Piscataway, NJ, USA: IEEE, Apr. 2020, pp. 744–748. DOI: 10.1109/RADAR42522.2020.9114650.

[24] Olivier Bau and Wendy E. Mackay. "OctoPocus: a dynamic guide for learning gesture-based command sets". In: *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology*. UIST '08. Monterey, CA, USA: Association for Computing Machinery, 2008, pp. 37–46. ISBN: 9781595939753. DOI: 10.1145/1449715.1449724. URL: https://doi.org/10.1145/1449715.1449724.

[25] Abel Díaz Berenguer, Meshia Cédric Oveneke, Habib-Ur-Rehman Khalid, Mitchel Alioscha-Perez, André Bourdoux, and Hichem Sahli. "GestureVLAD: Combining Unsupervised Features Representation and Spatio-Temporal Aggregation for Doppler-Radar Gesture Recognition". In: *IEEE Access* 7 (2019), pp. 137122–137135. DOI: 10.1109/ACCESS.2019.2942305.

[26] Alan Bole, Alan Wall, and Andy Norris. "Chapter 2 - The Radar System – Technical Principles". In: *Radar and ARPA Manual (Third Edition)*. Ed. by Alan Bole, Alan Wall, and Andy Norris. Third Edition. Oxford: Butterworth-Heinemann, 2014, pp. 29–137. ISBN: 978-0-08-097752-2. DOI: https://doi.org/10.1016/B978-0-08-097752-2.00002-7. URL: https://www.sciencedirect.com/science/article/pii/B9780080977522000027.

[27] Sanders Brandon. *Mastering Leap Motion*. Birmingham: Packt Publishing, 2014. ISBN: 978-1-783-55139-2.

[28] John Brooke. "SUS-A quick and dirty usability scale". In: *Usability evaluation in industry* (1996), pp. 189–194.

[29] Roberto Brunelli. *Template Matching Techniques in Computer Vision: Theory and Practice*. New York: John Wiley & Sons, 2009. ISBN: 978-0-470-51706-2.

[30] Ashley Bush and Sandeep Purao. "Mapping UML Techniques to Design Activities". In: *Information Modeling in the New Millennium*. Ed. by Matti Rossi and Keng Siau. IDEA Publishing Group, 2011. Chap. 12, pp. 199–217.

[31] Minghao Cai and Jiro Tanaka. "Go Together: Providing Nonverbal Awareness Cues to Enhance Co-Located Sensation in Remote Communication". In: *Hum.-Centric Comput. Inf. Sci.* 9.1 (Dec. 2019). ISSN: 2192-1962. DOI: 10.1186/s13673-019-0180-y. URL: https://doi.org/10.1186/s13673-019-0180-y.

[32] John C. Callender and H. G. Osburn. "An Empirical Comparison of Coefficient Alpha, Guttman's Lambda - 2, and MSPLIT Maximized Split-Half Reliability Estimates". In: *Journal of Educational Measurement* 16.2 (1979), pp. 89–99. ISSN: 00220655, 17453984. URL: http://www.jstor.org/stable/1434452.

[33] Gaëlle Calvary, Joëlle Coutaz, David Thevenin, Quentin Limbourg, Laurent Bouillon, and Jean Vanderdonckt. "A Unifying Reference Framework for multi-target user interfaces". In: *Interact. Comput.* 15.3 (2003), pp. 289–308. DOI: 10.1016/S0953-5438(03)00010-9. URL: https://doi.org/10.1016/S0953-5438(03)00010-9.

[34] Fabio Marco Caputo, S. Burato, Gianni Pavan, Théo Voillemin, Hazem Wannous, Jean-Philippe Vandeborre, Mehran Maghoumi, Eugene M. Taranta II, Alaleh Razmjoo, Joseph J. LaViola Jr., Fabio Manganaro, Stefano Pini, Guido Borghi, Roberto Vezzani, Rita Cucchiara, Hai-Dang Nguyen, Minh-Triet Tran, and Andrea Giachetti. "Online Gesture Recognition". In: *Eurographics Workshop on 3D Object Retrieval*. Ed.

by Silvia Biasotti, Guillaume Lavoué, and Remco Veltkamp. The Eurographics Association, 2019, pp. 93–102. ISBN: 978-3-03868-077-2. DOI: 10.2312/3dor.20191067.

[35] Fabio Marco Caputo and Andrea Giachetti. "Evaluation of Basic Object Manipulation Modes for Low-Cost Immersive Virtual Reality". In: *Proceedings of the 11th Biannual Conference on Italian SIGCHI Chapter* (Rome, Italy). CHItaly 2015. New York, NY, USA: Association for Computing Machinery, 2015, pp. 74–77. ISBN: 9781450336840. DOI: 10.1145/2808435.2808439. URL: https://doi.org/10.1145/2808435.2808439.

[36] Fabio Marco Caputo, Pietro Prebianca, Alessandro Carcangiu, Lucio Davide Spano, and Andrea Giachetti. "A 3 Cent Recognizer: Simple and Effective Retrieval and Classification of Mid-Air Gestures from Single 3D Traces". In: *Proceedings of the Conference on Smart Tools and Applications in Computer Graphics* (Catania, Italy). STAG '17. Goslar, DEU: Eurographics Association, 2017, pp. 9–15. DOI: 10.2312/stag.20171221. URL: https://doi.org/10.2312/stag.20171221.

[37] Fabio Marco Caputo, Pietro Prebianca, Alessandro Carcangiu, Lucio Davide Spano, and Andrea Giachetti. "Comparing 3D trajectories for simple mid-air gesture recognition". In: *Computers & Graphics* 73 (2018), pp. 17–25. ISSN: 0097-8493. DOI: https://doi.org/10.1016/j.cag.2018.02.009. URL: http://www.sciencedirect.com/science/article/pii/S0097849318300335.

[38] Géry Casiez, Nicolas Roussel, and Daniel Vogel. "1€ filter: a simple speed-based low-pass filter for noisy input in interactive systems". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '12. New York, NY, USA: Association for Computing Machinery, May 2012, pp. 2527–2530. ISBN: 978-1-4503-1015-4. DOI: 10.1145/2207676.2208639. URL: https://doi.org/10.1145/2207676.2208639 (visited on 12/29/2020).

[39] Ishan Chatterjee, Robert Xiao, and Chris Harrison. "Gaze+Gesture: Expressive, Precise and Targeted Free-Space Interactions". In: *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction* (Seattle, Washington, USA). ICMI '15. New York, NY, USA: Association for Computing Machinery, 2015, pp. 131–138. ISBN: 9781450339124. DOI: 10.1145/2818346.2820752. URL: https://doi.org/10.1145/2818346.2820752.

[40] Yineng Chen, Xiaojun Su, Feng Tian, Jin Huang, Xiaolong (Luke) Zhang, Guozhong Dai, and Hongan Wang. "Pactolus: A Method for Mid-Air Gesture Segmentation within EMG". In: *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San

Jose, California, USA). CHI EA '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 1760–1765. ISBN: 9781450340823. DOI: 10.1145/2851581.2892492. URL: https://doi.org/10.1145/2851581.2892492.

[41] Zhaoxi Chen, Gang Li, Francesco Fioranelli, and Hugh Griffiths. "Dynamic Hand Gesture Classification Based on Multistatic Radar Micro-Doppler Signatures Using Convolutional Neural Network". In: *Proceedings of the IEEE International Radar Conference* (Boston, MA, USA). RADAR '19. ISSN: 2375-5318. Piscataway, NJ, USA: IEEE, Apr. 2019, pp. 1–5. DOI: 10.1109/RADAR.2019.8835796.

[42] Zhaoxin Chen. "Recognition and interpretation of multi-touch gesture interaction." PhD thesis. INSA de Rennes, France, 2017. URL: https://tel.archives-ouvertes.fr/tel-01578068.

[43] Hong Cheng, Lu Yang, and Zicheng Liu. "Survey on 3D Hand Gesture Recognition". In: *IEEE Transactions on Circuits and Systems for Video Technology* 26.9 (2016), pp. 1659–1673. DOI: 10.1109/TCSVT.2015.2469551. URL: https://ieeexplore.ieee.org/document/7208833.

[44] Weng Cho Chew. *Waves and Fields in Inhomogeneous Media*. New York: Van Nostrand Reinhold, 1990.

[45] Cheng-Chin Chiang, Ren-Hong Wang, and Bo-Ruei Chen. "Recognizing Arbitrarily Connected and Superimposed Handwritten Numerals in Intangible Writing Interfaces". In: *Pattern Recogn.* 61.C (Jan. 2017), pp. 15–28. ISSN: 0031-3203. DOI: 10.1016/j.patcog.2016.07.018. URL: https://doi.org/10.1016/j.patcog.2016.07.018.

[46] Eunjung Choi, Heejin Kim, and Min K. Chung. "A taxonomy and notation method for three-dimensional hand gestures". en. In: *International Journal of Industrial Ergonomics* 44.1 (Jan. 2014), pp. 171–188. ISSN: 0169-8141. DOI: 10.1016/j.ergon.2013.10.011. URL: http://www.sciencedirect.com/science/article/pii/S0169814113001285 (visited on 01/06/2021).

[47] J. Choi, S. Ryu, and J. Kim. "Short-Range Radar Based Real-Time Hand Gesture Recognition Using LSTM Encoder". In: *IEEE Access* 7 (2019), pp. 33610–33618. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2903586.

[48] Andrew Clark and Deshendran Moodley. "A System for a Hand Gesture-Manipulated Virtual Reality Environment". In: *Proceedings of the Annual Conference of the South African Institute of Computer Scientists and Information Technologists* (Johannesburg, South Africa). SAICSIT '16. New York, NY, USA: Association for Computing Machinery, 2016. ISBN: 9781450348058. DOI: 10.1145/2987491.2987511. URL: https://doi.org/10.1145/2987491.2987511.

[49] Alex Colgan. *How Does the Leap Motion Controller Work?* Jan. 2017. URL: https://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work/.

[50] Harisson Cook, Quang Vinh Nguyen, Simeone Simoff, and Mao Lin Huang. "Enabling Gesture Interaction with 3D Point Cloud". In: *Proceedings of the 24th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*. Vol. 2602. Computer Science Research Notes CSRN, June 2016, pp. 59–68. ISBN: 978-80-86943-58-9. URL: https://opus.lib.uts.edu.au/handle/10453/45407.

[51] Klen Copič Pucihar, Christian Sandor, Matjaž Kljun, Wolfgang Huerst, Alexander Plopski, Takafumi Taketomi, Hirokazu Kato, and Luis A. Leiva. "The Missing Interface: Micro-Gestures on Augmented Objects". In: *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI EA '19. New York, NY, USA: Association for Computing Machinery, May 2019, pp. 1–6. ISBN: 978-1-4503-5971-9. DOI: 10.1145/3290607.3312986. URL: https://doi.org/10.1145/3290607.3312986 (visited on 12/21/2020).

[52] Klen Čopič Pucihar, Nuwan T. Attygalle, Matjaz Kljun, Christian Sandor, and Luis A. Leiva. "Solids on Soli: Millimetre-Wave Radar Sensing through Materials". In: *Proc. ACM Hum.-Comput. Interact.* 6.EICS (June 2022). DOI: 10.1145/3532212. URL: https://doi.org/10.1145/3532212.

[53] Erwin Cornet. "Human-Computer Interaction with an Armband by Electromyography". Anglais. PhD thesis. UCL - Ecole polytechnique de Louvain, 2023. URL: http://hdl.handle.net/2078.1/thesis:43237.

[54] Maria Francesca Costabile, Piero Mussio, Loredana Parasiliti Provenza, and Antonio Piccinno. "End Users as Unwitting Software Developers". In: *Proceedings of the 4th International Workshop on End-User Software Engineering* (Leipzig, Germany). WEUSE '08. New York, NY, USA: Association for Computing Machinery, 2008, pp. 6–10. ISBN: 9781605580340. DOI: 10.1145/1370847.1370849. URL: https://doi.org/10.1145/1370847.1370849.

[55] Adrien Coyette, Sascha Schimke, Jean Vanderdonckt, and Claus Vielhauer. "Trainable Sketch Recognizer for Graphical User Interface Design". In: *Proc. of 11th IFIP TC 13 International Conference on Human-Computer Interaction, INTERACT '07* (Rio de Janeiro, Brazil). Ed. by Cécilia Baranauskas, Philippe Palanque, Julio Abascal, and Simone Diniz Junqueira Barbosa. Berlin, Heidelberg: Springer, 2007, pp. 124–135. ISBN: 978-3-540-74796-3.

[56] Elena-Gina Craciun, Ionela Rusu, and Radu-Daniel Vatavu. *Free-Hand Gesture Recognizer Pseudocode*. http://www.eed.usv.ro/mintviz/projects/GIVISIMP/data/Pseudocode2.pdf. [Online; accessed 09-August-2020]. 2016.

[57] Chris Creed. "Assistive technology for disabled visual artists: exploring the impact of digital technologies on artistic practice". In: *Disability & Society* 33.7 (2018), pp. 1103–1119. DOI: 10.1080/09687599.2018.1469400. eprint: https://doi.org/10.1080/09687599.2018.1469400. URL: https://doi.org/10.1080/09687599.2018.1469400.

[58] Zachary A. Daniels, Steven R. Stinson, Shenchi Tian, Evan Mulbry, and Brian Y. Chen. "A Gesture-Based Interface for the Exploration and Classification of Protein Binding Cavities". In: *Proceedings of the 2014 Workshop on Mobile Augmented Reality and Robotic Technology-Based Systems* (Bretton Woods, New Hampshire, USA). MARS '14. New York, NY, USA: Association for Computing Machinery, 2014, pp. 47–50. ISBN: 9781450328234. DOI: 10.1145/2609829.2609838. URL: https://doi.org/10.1145/2609829.2609838.

[59] Albéric De Coster and Sébastien Lambot. "Fusion of Multifrequency GPR Data Freed From Antenna Effects". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 11.2 (2018), pp. 664–674. DOI: 10.1109/JSTARS.2018.2790419.

[60] Roberto De Prisco, Delfina Malandrino, Gianluca Zaccagnino, and Rocco Zaccagnino. "Natural User Interfaces to Support and Enhance Real-Time Music Performance". In: *Proceedings of the International Working Conference on Advanced Visual Interfaces* (Bari, Italy). AVI '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 204–211. ISBN: 9781450341318. DOI: 10.1145/2909132.2909249. URL: https://doi.org/10.1145/2909132.2909249.

[61] Quentin De Smedt, Hazem Wannous, Jean-Philippe Vandeborre, Joris Guerry, Bertrand Le Saux, and David Filliat. "3D Hand Gesture Recognition Using a Depth and Skeletal Dataset: SHREC'17 Track". In: *Proceedings of the Workshop on 3D Object Retrieval* (Lyon, France). 3Dor

'17. Goslar, DEU: Eurographics Association, 2017, pp. 33–38. DOI: 10.2312/3dor.20171049. URL: https://doi.org/10.2312/3dor.20171049.

[62] Bastiaan Dekker, Sebastiaan Jacobs, A. S. Kossen, Maarten C. Kruithof, Albert G. Huizing, and M. Geurts. "Gesture recognition with a low power FMCW radar and a deep convolutional neural network". In: *Proceedings of the European Radar Conference*. EURAD '17. Oct. 2017, pp. 163–166. DOI: 10.23919/EURAD.2017.8249172.

[63] William Delamare, Chaklam Silpasuwanchai, Sayan Sarcar, Toshiaki Shiraki, and Xiangshi Ren. "On Gesture Combination: An Exploration of a Solution to Augment Gesture Interaction". In: *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces* (Daejeon, Republic of Korea). ISS '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 135–146. ISBN: 9781450368919. DOI: 10.1145/3343055.3359706. URL: https://doi.org/10.1145/3343055.3359706.

[64] Tilman Dingler, Rufat Rzayev, Alireza Sahami Shirazi, and Niels Henze. "Designing Consistent Gestures Across Device Types: Eliciting RSVP Controls for Phone, Watch, and Glasses". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada). CHI '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 1–12. ISBN: 9781450356206. DOI: 10.1145/3173574.3173993. URL: https://doi.org/10.1145/3173574.3173993.

[65] Haiwei Dong, Nadia Figueroa, and Abdulmotaleb El Saddik. "An Elicitation Study on Gesture Attitudes and Preferences Towards an Interactive Hand-Gesture Vocabulary". In: *Proceedings of the 23rd ACM International Conference on Multimedia* (Brisbane, Australia). MM '15. New York, NY, USA: Association for Computing Machinery, 2015, pp. 999–1002. ISBN: 9781450334594. DOI: 10.1145/2733373.2806385. URL: https://doi.org/10.1145/2733373.2806385.

[66] Giannis Drossis, Dimitris Grammenos, Chryssi Birliraki, and Constantine Stephanidis. "MAGIC: Developing a Multimedia Gallery Supporting mid-Air Gesture-Based Interaction and Control". In: *HCI International 2013 - Posters' Extended Abstracts*. Ed. by Constantine Stephanidis. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 303–307. ISBN: 978-3-642-39473-7.

[67] Chong Yang Du, Xin Huai Wang, Zuan Xing Yuan, and Yin Xu. "Design of Gesture Recognition System Based on 77GHz Millimeter Wave

Radar". In: *2019 International Conference on Microwave and Millimeter Wave Technology (ICMMT)*. May 2019, pp. 1–3. DOI: 10.1109/ICMMT45702.2019.8992849.

[68] Chuan Du, Lei Zhang, Xiping Sun, Junxu Wang, and Jialian Sheng. "Enhanced Multi-Channel Feature Synthesis for Hand Gesture Recognition Based on CNN With a Channel and Spatial Attention Mechanism". In: *IEEE Access* 8 (2020), pp. 144610–144620. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3010063.

[69] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley & Sons, New York, 2000.

[70] Catherine Dumas, Sanda Erdelez, Jeffrey Pomerantz, and Vik Parthiban. "Usability testing of LUI: A new human-computer interface for large displays". In: *Proceedings of the Association for Information Science and Technology* 56.1 (2019), pp. 645–647. DOI: https://doi.org/10.1002/pra2.118. eprint: https://asistdl.onlinelibrary.wiley.com/doi/pdf/10.1002/pra2.118. URL: https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/pra2.118.

[71] Manuel Eggimann, Jonas Erb, Philipp Mayer, Michele. Magno, and Luca Benini. "Low Power Embedded Gesture Recognition Using Novel Short-Range Radar Sensors". In: *2019 IEEE SENSORS*. ISSN: 2168-9229. Oct. 2019, pp. 1–4. DOI: 10.1109/SENSORS43011.2019.8956617.

[72] Matthias G. Ehrnsperger, Thomas Brenner, Uwe Siart, and Thomas F. Eibert. "Real-Time Gesture Recognition with Shallow Convolutional Neural Networks Employing an Ultra Low Cost Radar System". In: *2020 German Microwave Conference (GeMiC)*. ISSN: 2167-8022. Mar. 2020, pp. 88–91.

[73] Matthias G. Ehrnsperger, Henri L. Hoese, Uwe Siart, and Thomas F. Eibert. "Performance Investigation of Machine Learning Algorithms for Simple Human Gesture Recognition Employing an Ultra Low Cost Radar System". In: *2019 Kleinheubach Conference*. Sept. 2019, pp. 1–4.

[74] Paul Ekman and Wallace V. Friesen. "The Repertoire of Nonverbal Behavior: Categories, Origins, Usage, and Coding". In: *Semiotica* 1.1 (1969), pp. 49–98. DOI: doi:10.1515/semi.1969.1.1.49. URL: https://doi.org/10.1515/semi.1969.1.1.49.

[75] Institute of Electrical and Electronics Engineers. "IEEE Standard Letter Designations for Radar-Frequency Bands". In: *IEEE Std 521-2019 (Revision of IEEE Std 521-2002)* (2020), pp. 1–15. DOI: 10.1109/IEEESTD.2020.8999849. URL: https://ieeexplore.ieee.org/servlet/opac?punumber=8999827.

[76] Ana Belén Erazo and Jorge Luis Pérez Medina. "Algorithmic Efficiency of Stroke Gesture Recognizers: a Comparative Analysis". In: *International Journal on Advanced Science, Engineering and Information Technology* 10.2 (2020), pp. 438–446. ISSN: 2088-5334. DOI: 10.18517/ijaseit.10.2.10807. URL: http://ijaseit.insightsociety.org/index.php?option=com_content%5C&view=article%5C&id=9%5C&Itemid=1%5C&article_id=10807.

[77] Robert Feldt and Ana Magazinius. "Validity Threats in Empirical Software Engineering Research - An Initial Survey". In: *Proceedings of the 22nd International Conference on Software Engineering & Knowledge Engineering (SEKE'2010), Redwood City, San Francisco Bay, CA, USA, July 1 - July 3, 2010*. Knowledge Systems Institute Graduate School, 2010, pp. 374–379.

[78] Xiang Feng, Qun Song, Qingfang Guo, Duo Liu, Zhanfeng Zhao, and Yinan Zhao. "Hand Gesture Recognition with Ensemble Time-Frequency Signatures Using Enhanced Deep Convolutional Neural Network". In: *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. ISSN: 2640-0103. Nov. 2019, pp. 1602–1605. DOI: 10.1109/APSIPAASC47483.2019.9023254.

[79] Katherine Fennedy, Jeremy Hartmann, Quentin Roy, Simon Tangi Perrault, and Daniel Vogel. "OctoPocus in VR: Using a Dynamic Guide for 3D Mid-Air Gestures in Virtual Reality". In: *IEEE Transactions on Visualization and Computer Graphics* 27.12 (Dec. 2021), pp. 4425–4438. ISSN: 1077-2626. DOI: 10.1109/TVCG.2021.3101854. URL: https://doi.org/10.1109/TVCG.2021.3101854.

[80] Pedro M. Ferreira, Jaime S. Cardoso, and Ana Rebelo. "On the Role of Multimodal Learning in the Recognition of Sign Language". In: *Multimedia Tools Appl.* 78.8 (Apr. 2019), pp. 10035–10056. ISSN: 1380-7501. DOI: 10.1007/s11042-018-6565-5. URL: https://doi.org/10.1007/s11042-018-6565-5.

[81] Lars Ohlsson Fhager, Sebastian Heunisch, Hannes Dahlberg, Anton Evertsson, and Lars-Erik Wernersson. "Pulsed Millimeter Wave Radar for Hand Gesture Sensing and Classification". In: *IEEE Sensors Letters* 3.12 (Dec. 2019), pp. 1–4. ISSN: 2475-1472. DOI: 10.1109/LSENS.2019.2953022.

[82] Ivo Aluízio Stinghen Filho, Estevam Nicolas Chen, Jucimar Maia da Silva Junior, and Ricardo da Silva Barboza. "Gesture Recognition Using Leap Motion: A Comparison between Machine Learning Algorithms". In: *ACM SIGGRAPH 2018 Posters* (Vancouver, British Columbia, Canada). SIGGRAPH '18. New York, NY, USA: Association for Computing Machinery, 2018. ISBN: 9781450358170. DOI: 10.

1145/3230744.3230750. URL: https://doi.org/10.1145/3230744.3230750.

[83] Zak Flintoff, Bruno Johnston, and Minas Liarokapis. "Single-Grasp, Model-Free Object Classification using a Hyper-Adaptive Hand, Google Soli, and Tactile Sensors". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 1943–1950. DOI: 10.1109/IROS.2018.8594166.

[84] Bruno Fruchard, Eric Lecolinet, and Olivier Chapuis. "How Memorizing Positions or Directions Affects Gesture Learning?" In: *Proceedings of the 2018 ACM International Conference on Interactive Surfaces and Spaces* (Tokyo, Japan). ISS '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 107–114. ISBN: 9781450356947. DOI: 10.1145/3279778.3279787. URL: https://doi.org/10.1145/3279778.3279787.

[85] Zeqing Fu, Yiyi Deng, Xin Jia, Bin Gao, Xiaoming Zhu, and Yanlin Luo. "Automated Brain Extraction and Associated 3D Inspection Layers for the Rhesus Macaque MRI Datasets". In: *Proceedings of the 15th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry - Volume 1* (Zhuhai, China). VRCAI '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 261–269. ISBN: 9781450346924. DOI: 10.1145/3013971.3013984. URL: https://doi.org/10.1145/3013971.3013984.

[86] Claire Galea and Chris Porter. "Accessible Choral Ensembles for Visually Impaired Singers". In: *Proceedings of the 32nd International BCS Human Computer Interaction Conference* (Belfast, United Kingdom). HCI '18. Swindon, GBR: BCS Learning & Development Ltd., 2018. DOI: 10.14236/ewic/HCI2018.37. URL: https://doi.org/10.14236/ewic/HCI2018.37.

[87] Asim Ghaffar, Faheem Khan, and Sung Ho Cho. "Hand Pointing Gestures Based Digital Menu Board Implementation Using IR-UWB Transceivers". In: *IEEE Access* 7 (2019), pp. 58148–58157. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2914410.

[88] Bogdan-Florin Gheran, Jean Vanderdonckt, and Radu-Daniel Vatavu. "Gestures for Smart Rings: Empirical Results, Insights, and Design Implications". In: *Proc. of the 2018 Designing Interactive Systems Conference* (Hong Kong, China). Ed. by Ilpo Koskinen, Youn-Kyung Lim, Teresa Cerratto-Pargman, Kenny K. N. Chow, and William Odom. DIS '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 623–635. ISBN: 9781450351980. DOI: 10.1145/3196709.3196741. URL: https://doi.org/10.1145/3196709.3196741.

[89] Andrew Gigie, Smriti Rani, Arijit Chowdhury, Tapas Chakravarty, and Arpan Pal. "An agile approach for human gesture detection using synthetic radar data". In: *Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2019 ACM International Symposium on Wearable Computers*. UbiComp/ISWC '19 Adjunct. New York, NY, USA: Association for Computing Machinery, Sept. 2019, pp. 558–564. ISBN: 978-1-4503-6869-8. DOI: 10.1145/3341162.3349332. URL: https://doi.org/10.1145/3341162.3349332 (visited on 12/21/2020).

[90] Emile Giot. "Radar-based gesture interaction: from sensing to recognition". Anglais. PhD thesis. UCL - Ecole polytechnique de Louvain, 2023. URL: http://hdl.handle.net/2078.1/thesis:43330.

[91] Pınar Göker and Memduha Gülhal Bozkir. "Determination of hand and palm surface areas as a percentage of body surface area in Turkish young adults". In: *Trauma and Emergency Care* 2.4 (May 2017), pp. 1–4. DOI: 10.15761/TEC.1000135. URL: https://www.oatext.com/pdf/TEC-2-135.pdf.

[92] Francisco Gomez-Donoso, Sergio Orts-Escolano, Alberto Garcia-Garcia, Jose Garcia-Rodriguez, John Alejandro Castro-Vargas, Sergiu Ovidiu-Oprea, and Miguel Cazorla. "A Robotic Platform for Customized and Interactive Rehabilitation of Persons with Disabilities". In: *Pattern Recogn. Lett.* 99.C (Nov. 2017), pp. 105–113. ISSN: 0167-8655. DOI: 10.1016/j.patrec.2017.05.027. URL: https://doi.org/10.1016/j.patrec.2017.05.027.

[93] Piyali Goswami, Sandeep Rao, Sachin Bharadwaj, and Amanda Nguyen. "Real-Time Multi-Gesture Recognition using 77 GHz FMCW MIMO Single Chip Radar". In: *2019 IEEE International Conference on Consumer Electronics (ICCE)*. ISSN: 2158-4001. Jan. 2019, pp. 1–4. DOI: 10.1109/ICCE.2019.8662006.

[94] Rebecca A. Grier. "How High is High? A Meta-Analysis of NASA-TLX Global Workload Scores". en. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 59.1 (Sept. 2015). Publisher: SAGE Publications Inc, pp. 1727–1731. ISSN: 2169-5067. DOI: 10.1177/1541931215591373. URL: https://doi.org/10.1177/1541931215591373 (visited on 02/08/2021).

[95] Daniela Grijincu, Miguel A. Nacenta, and Per Ola Kristensson. "User-Defined Interface Gestures: Dataset and Analysis". In: *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces* (Dresden, Germany). ITS '14. New York, NY, USA: Association for Computing Machinery, 2014, pp. 25–34. ISBN: 9781450325875. DOI:

10.1145/2669485.2669511. URL: https://doi.org/10.1145/2669485.2669511.

[96] Celeste Groenewald, Craig Anslow, Junayed Islam, Chris Rooney, Peter Passmore, and William Wong. "Understanding 3D Mid-Air Hand Gestures with Interactive Surfaces and Displays: A Systematic Literature Review". In: *Proceedings of the 30th International BCS Human Computer Interaction Conference: Fusion!* (Poole, United Kingdom). HCI '16. Swindon, GBR: BCS Learning & Development Ltd., 2016. DOI: 10.14236/ewic/HCI2016.43. URL: https://doi.org/10.14236/ewic/HCI2016.43.

[97] Aakar Gupta, Thomas Pietrzak, Cleon Yau, Nicolas Roussel, and Ravin Balakrishnan. "Summon and Select: Rapid Interaction with Interface Controls in Mid-Air". In: *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces* (Brighton, United Kingdom). ISS '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 52–61. ISBN: 9781450346917. DOI: 10.1145/3132272.3134120. URL: https://doi.org/10.1145/3132272.3134120.

[98] Sevgi Z. Gurbuz, Ali C. Gurbuz, Evie A. Malaia, Darrin J. Griffin, Chris Crawford, Emre Kurtoglu, Mohammad Mahbubur Rahman, Ridvan Aksu, and Robiulhossain Mdrafi. "ASL Recognition Based on Kinematics Derived from a Multi-Frequency RF Sensor Network". In: *2020 IEEE Sensors*. ISSN: 2168-9229. Oct. 2020, pp. 1–4. DOI: 10.1109/SENSORS47125.2020.9278864.

[99] Michael Gusenbauer and Neal R. Haddaway. "Which academic search systems are suitable for systematic reviews or meta-analyses? Evaluating retrieval qualities of Google Scholar, PubMed, and 26 other resources". In: *Research Synthesis Methods* 11.2 (2020), pp. 181–217. DOI: https://doi.org/10.1002/jrsm.1378. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jrsm.1378. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/jrsm.1378.

[100] Louis Guttman. "A basis for analyzing test-retest reliability". In: *Psychometrika* 10 (Dec. 1945), pp. 255–282. DOI: 10.1007/BF02288892. URL: https://link.springer.com/article/10.1007/BF02288892.

[101] Tibor Guzsvinecz, Veronika Szucs, and Cecilia Sik-Lanyi. "Suitability of the Kinect Sensor and Leap Motion Controller—A Literature Review". In: *Sensors* 19.5 (2019). ISSN: 1424-8220. DOI: 10.3390/s19051072. URL: https://www.mdpi.com/1424-8220/19/5/1072.

[102] Jeffrey T. Hansberger, Chao Peng, Shannon L. Mathis, Vaidyanath Areyur Shanthakumar, Sarah C. Meacham, Lizhou Cao, and Victoria R. Blakely. "Dispelling the Gorilla Arm Syndrome: The Viability of Prolonged Gesture Interactions". In: *Virtual, Augmented and Mixed Reality*. Ed. by Stephanie Lackey and Jessie Chen. Cham: Springer International Publishing, 2017, pp. 505–520. ISBN: 978-3-319-57987-0.

[103] Sandra G. Hart and Lowell E. Staveland. "Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research". In: *Human Mental Workload*. Ed. by Peter A. Hancock and Najmedin Meshkati. Vol. 52. Advances in Psychology. North-Holland, 1988, pp. 139–183. DOI: https://doi.org/10.1016/S0166-4115(08)62386-9. URL: http://www.sciencedirect.com/science/article/pii/S0166411508623869.

[104] Eiji Hayashi, Jaime Lien, Nicholas Gillian, Leonardo Giusti, Dave Weber, Jin Yamanaka, Lauren Bedal, and Ivan Poupyrev. "RadarNet: Efficient Gesture Recognition Technique Utilizing a Miniature Radar Sensor". In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan). CHI '21. New York, NY, USA: Association for Computing Machinery, 2021. ISBN: 9781450380966. DOI: 10.1145/3411764.3445367. URL: https://doi.org/10.1145/3411764.3445367.

[105] Souvik Hazra and Avik Santra. "Radar Gesture Recognition System in Presence of Interference using Self-Attention Neural Network". In: *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. Dec. 2019, pp. 1409–1414. DOI: 10.1109/ICMLA.2019.00230.

[106] Souvik Hazra and Avik Santra. "Robust Gesture Recognition Using Millimetric-Wave Radar System". In: *IEEE Sensors Letters* 2.4 (Dec. 2018), pp. 1–4. ISSN: 2475-1472. DOI: 10.1109/LSENS.2018.2882642.

[107] Souvik Hazra and Avik Santra. "Short-Range Radar-Based Gesture Recognition System Using 3D CNN With Triplet Loss". In: *IEEE Access* 7 (2019), pp. 125623–125633. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2938725.

[108] Antonio Hernández-Vela, Miguel Ángel Bautista, Xavier Perez-Sala, Víctor Ponce-López, Sergio Escalera, Xavier Baró, Oriol Pujol, and Cecilio Angulo. "Probability-based Dynamic Time Warping and Bag-of-Visual-and-Depth-Words for Human Gesture Recognition in RGB-D". In: *Pattern Recognition Letters* 50 (2014). Depth Image Analysis, pp. 112–121. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.

patrec.2013.09.009. URL: https://www.sciencedirect.com/science/article/pii/S0167865513003450.

[109] Juan David Hincapié-Ramos, Xiang Guo, Paymahn Moghadasian, and Pourang Irani. "Consumed endurance: a metric to quantify arm fatigue of mid-air interactions". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '14. Toronto, Ontario, Canada: Association for Computing Machinery, 2014, pp. 1063–1072. ISBN: 9781450324731. DOI: 10.1145/2556288.2557130. URL: https://doi.org/10.1145/2556288.2557130.

[110] Andreas Hinderks, Martin Schrepp, Francisco José Domínguez Mayo, M.J. Escalona, and Jörg Thomaschewski. "Developing a UX KPI based on the User Experience Questionnaire". In: *Computer Standards & Interfaces* 65 (July 2019), pp. 38–44. DOI: 10.1016/j.csi.2019.01.007.

[111] Akram Al-Hourani, Robin J. Evans, Peter M. Farrell, Bill Moran, Marco Martorella, Sithamparanathan Kandeepan, Stan Skafidas, and Udaya Parampalli. "Chapter 7 - Millimeter-wave integrated radar systems and techniques". In: *Academic Press Library in Signal Processing, Volume 7*. Ed. by Rama Chellappa and Sergios Theodoridis. Academic Press, 2018, pp. 317–363. ISBN: 978-0-12-811887-0. DOI: https://doi.org/10.1016/B978-0-12-811887-0.00007-9. URL: https://www.sciencedirect.com/science/article/pii/B9780128118870000079.

[112] Jinmiao Huang, Prakhar Jaiswal, and Rahul Rai. "Gesture-based system for next generation natural and intuitive interfaces". In: *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 33.1 (2019), pp. 54–68. DOI: 10.1017/S0890060418000045.

[113] Cloe Huesser, Simon Schubiger, and Arzu Çöltekin. "Gesture Interaction in Virtual Reality". In: *Human-Computer Interaction – INTERACT 2021*. Ed. by Carmelo Ardito, Rosa Lanzilotti, Alessio Malizia, Helen Petrie, Antonio Piccinno, Giuseppe Desolda, and Kori Inkpen. Cham: Springer International Publishing, 2021, pp. 151–160. ISBN: 978-3-030-85613-7.

[114] ECMA International. *Standard ECMA-404 – The JSON Data Interchange Syntax*. Dec. 2017. URL: https://www.ecma-international.org/publications/standards/Ecma-404.htm.

[115] Karim Ishak, Nils Appenrodt, Jrgen Dickmann, and Christian Waldschmidt. "Human Motion Training Data Generation for Radar Based Deep Learning Applications". In: *2018 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*. Apr. 2018, pp. 1–4. DOI: 10.1109/ICMIM.2018.8443559.

[116] ISO. *ISO/IEC 25010 - Software Quality Product Standard*. standard. Geneva: International Standard Organization, 2019. URL: https://iso25000.com/index.php/en/iso-25000-standards/iso-25010?limit=3%5C&limitstart=0.

[117] ISO. *ISO/IEC 9241, Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts*. standard. Geneva: International Standard Organization, 2018. URL: https://www.iso.org/obp/ui/%5C#iso:std:iso:9241:-11:en.

[118] Mikkel R. Jakobsen, Yvonne Jansen, Sebastian Boring, and Kasper Hornbæk. "Should I Stay or Should I Go? Selecting Between Touch and Mid-Air Gestures for Large-Display Interaction". In: *Human-Computer Interaction – INTERACT 2015*. Berlin, Heidelberg: Springer-Verlag, 2015, pp. 455–473. ISBN: 978-3-319-22697-2. DOI: 10.1007/978-3-319-22698-9_31. URL: https://doi.org/10.1007/978-3-319-22698-9_31.

[119] Sujin Jang, Wolfgang Stuerzlinger, Satyajit Ambike, and Karthik Ramani. "Modeling Cumulative Arm Fatigue in Mid-Air Interaction based on Perceived Exertion and Kinetics of Arm Motion". In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI '17. Denver, Colorado, USA: Association for Computing Machinery, 2017, pp. 3328–3339. ISBN: 9781450346559. DOI: 10.1145/3025453.3025523. URL: https://doi.org/10.1145/3025453.3025523.

[120] Xianta Jiang, Zhen Gang Xiao, and Carlo Menon. "Virtual Grasps Recognition Using Fusion of Leap Motion and Force Myography". In: *Virtual Real.* 22.4 (Nov. 2018), pp. 297–308. ISSN: 1359-4338. DOI: 10.1007/s10055-018-0339-2. URL: https://doi.org/10.1007/s10055-018-0339-2.

[121] Zhang Kang, Lan Shengchang, and Zhang Guiyuan. "Mining Spatio-Temporal Features from mmW Radar echoes for Hand Gesture Recognition". In: *2019 IEEE Asia-Pacific Microwave Conference (APMC)*. Dec. 2019, pp. 93–95. DOI: 10.1109/APMC46564.2019.9038713.

[122] Adam Kendon. "Gesticulation and Speech: Two Aspects of the Process of Utterance". In: *The Relationship of Verbal and Nonverbal Communication*. Berlin, New York: De Gruyter Mouton, 1980, pp. 207–228. ISBN: 9783110813098. DOI: doi:10.1515/9783110813098.207. URL: https://doi.org/10.1515/9783110813098.207.

[123] Adam Kendon. "How gestures can become like words". In: *Crosscultural Perspectives in Nonverbal Communication* (Jan. 1988), pp. 131–141.

[124] Nicolai Kern, Maximilian Steiner, Ramona Lorenzin, and Christian Waldschmidt. "Robust Doppler-Based Gesture Recognition With Incoherent Automotive Radar Sensor Networks". In: *IEEE Sensors Letters* 4.11 (Nov. 2020), pp. 1–4. ISSN: 2475-1472. DOI: 10.1109/LSENS.2020.3033586.

[125] Ahmed S. Khalaf, Sultan A. Alharthi, Igor Dolgov, and Z O. Toups. "A Comparative Study of Hand Gesture Recognition Devices in the Context of Game Design". In: *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces* (Daejeon, Republic of Korea). ISS '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 397–402. ISBN: 9781450368919. DOI: 10.1145/3343055.3360758. URL: https://doi.org/10.1145/3343055.3360758.

[126] Faheem Khan, Seong Kyu Leem, and Sung Ho Cho. "Hand-Based Gesture Recognition for Vehicular Applications Using IR-UWB Radar". In: *Sensors* 17.4 (2017), pp. 1–18. ISSN: 1424-8220. DOI: 10.3390/s17040833. URL: https://www.mdpi.com/1424-8220/17/4/833.

[127] Faheem Khan, Seong Kyu Leem, and Sung Ho Cho. "In-Air Continuous Writing Using UWB Impulse Radar Sensors". In: *IEEE Access* 8 (2020), pp. 99302–99311. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2994281.

[128] Seo Yul Kim, Hong Gul Han, Jin Woo Kim, Sanghoon Lee, and Tae Wook Kim. "A Hand Gesture Recognition Sensor Using Reflected Impulses". In: *IEEE Sensors Journal* 17.10 (May 2017), pp. 2975–2976. ISSN: 1558-1748. DOI: 10.1109/JSEN.2017.2679220.

[129] Youngwook Kim and Brian Toomajian. "Application of Doppler radar for the recognition of hand gestures using optimized deep convolutional neural networks". In: *2017 11th European Conference on Antennas and Propagation (EUCAP)*. Mar. 2017, pp. 1258–1260. DOI: 10.23919/EuCAP.2017.7928465.

[130] Youngwook Kim and Brian Toomajian. "Hand Gesture Recognition Using Micro-Doppler Signatures With Convolutional Neural Network". In: *IEEE Access* 4 (2016), pp. 7125–7130. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2016.2617282.

[131] Vasilij Kiselev, Maxim Khlamov, and Kirill Chuvilin. "Hand Gesture Recognition with Multiple Leap Motion Devices". In: *Proceedings of the 24th Conference of Open Innovations Association FRUCT* (Moscow, Russia). FRUCT'24. Helsinki, Uusimaa, FIN: FRUCT Oy, 2019.

[132] Barbara Kitchenham, Rialette Pretorius, David Budgen, O. Pearl Brereton, Mark Turner, Mahmood Niazi, and Stephen Linkman. "Systematic literature reviews in software engineering – A tertiary study". In: *Information and Software Technology* 52.8 (2010), pp. 792–805. ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2010.03.006. URL: http://www.sciencedirect.com/science/article/pii/S0950584910000467.

[133] Daniel Kohlsdorf and Thad E. Starner. "MAGIC summoning: towards automatic suggesting and testing of gestures with low probability of false positives during use". In: *The Journal of Machine Learning Research* 14.1 (Jan. 2013), pp. 209–242. ISSN: 1532-4435.

[134] Daniel Kohlsdorf, Thad E. Starner, and Daniel Ashbrook. "MAGIC 2.0: A web tool for false positive prediction and prevention for gesture recognition systems". In: *Face and Gesture 2011*. 2011, pp. 1–6.

[135] Panayiotis Koutsabasis and Chris K. Domouzis. "Mid-Air Browsing and Selection in Image Collections". In: *Proceedings of the International Working Conference on Advanced Visual Interfaces* (Bari, Italy). AVI '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 21–27. ISBN: 9781450341318. DOI: 10.1145/2909132.2909248. URL: https://doi.org/10.1145/2909132.2909248.

[136] Panayiotis Koutsabasis and Panagiotis Vogiatzidakis. "Empirical Research in Mid-Air Interaction: A Systematic Review". In: *International Journal of Human–Computer Interaction* 35.18 (2019), pp. 1747–1768. DOI: 10.1080/10447318.2019.1572352. URL: https://doi.org/10.1080/10447318.2019.1572352.

[137] Sven Kratz and Maribeth Back. "Towards Accurate Automatic Segmentation of IMU-Tracked Motion Gestures". In: *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* (Seoul, Republic of Korea). CHI EA '15. New York, NY, USA: Association for Computing Machinery, 2015, pp. 1337–1342. ISBN: 9781450331463. DOI: 10.1145/2702613.2732922. URL: https://doi.org/10.1145/2702613.2732922.

[138] Sven Kratz and Jason Wiese. "GestureSeg: Developing a Gesture Segmentation System Using Gesture Execution Phase Labeling by Crowd Workers". In: *Proceedings of the 8th ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Brussels, Belgium). EICS '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 61–72. ISBN: 9781450343220. DOI: 10.1145/2933242.2933261. URL: https://doi.org/10.1145/2933242.2933261.

[139] Vinayak Krishnamurthy and Karthik Ramani. "A gesture-free geometric approach for mid-air expression of design intent in 3D virtual pottery". In: *Computer-Aided Design* 69 (2015), pp. 11–24. ISSN: 0010-4485. DOI: https://doi.org/10.1016/j.cad.2015.06.006. URL: https://www.sciencedirect.com/science/article/pii/S001044851500086X.

[140] Per Ola Kristensson and Leif C. Denby. "Continuous Recognition and Visualization of Pen Strokes and Touch-Screen Gestures". In: *Sketch Based Interfaces and Modeling, Vancouver, BC, Canada, 5-7 August 2011. Proceedings*. Ed. by Tracy Hammond and Andrew Nealen. Eurographics Association, 2011, pp. 95–102. DOI: 10.2312/SBM/SBM11/095-102. URL: https://doi.org/10.2312/SBM/SBM11/095-102.

[141] Hovannes Kulhandjian, Prakshi Sharma, Michel Kulhandjian, and Claude D'Amours. "Sign Language Gesture Recognition Using Doppler Radar and Deep Learning". In: *2019 IEEE Globecom Workshops (GC Wkshps)*. Dec. 2019, pp. 1–6. DOI: 10.1109/GCWkshps45667.2019.9024607.

[142] Pradeep Kumar, Himaanshu Gauba, Partha Pratim Roy, and Debi Prosad Dogra. "Coupled HMM-Based Multi-Sensor Data Fusion for Sign Language Recognition". In: *Pattern Recogn. Lett.* 86.C (Jan. 2017), pp. 1–8. ISSN: 0167-8655.

[143] Pradeep Kumar, Partha Pratim Roy, and Debi Prosad Dogra. "Independent Bayesian Classifier Combination Based Sign Language Recognition Using Facial Expression". In: *Inf. Sci.* 428.C (Feb. 2018), pp. 30–48. ISSN: 0020-0255. DOI: 10.1016/j.ins.2017.10.046. URL: https://doi.org/10.1016/j.ins.2017.10.046.

[144] Pradeep Kumar, Rajkumar Saini, Partha Pratim Roy, and Debi Prosad Dogra. "3D Text Segmentation and Recognition Using Leap Motion". In: *Multimedia Tools Appl.* 76.15 (Aug. 2017), pp. 16491–16510. ISSN: 1380-7501. DOI: 10.1007/s11042-016-3923-z. URL: https://doi.org/10.1007/s11042-016-3923-z.

[145] Gordon Kurtenbach, George Fitzmaurice, Thomas Baudel, and Bill Buxton. "The Design of a GUI Paradigm Based on Tablets, Two-hands, and Transparency". In: *Proceedings of the ACM International Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA). CHI '97. New York, NY, USA: ACM, 1997, pp. 35–42. ISBN: 0-89791-802-9. DOI: 10.1145/258549.258574. URL: http://doi.acm.org/10.1145/258549.258574.

[146] Lynn Kysh. "Difference between a systematic review and a literature review". In: (Aug. 2013). DOI: 10.6084/m9.figshare.766364.v1. URL: https://figshare.com/articles/Difference_between_a_systematic_review_and_a_literature_review/766364.

[147] Arnaud Lahousse. "Ring-based gestural interaction: from elicitation to recognition". Anglais. PhD thesis. UCL - Ecole polytechnique de Louvain, 2022. URL: http://hdl.handle.net/2078.1/thesis:36013.

[148] Sébastien Lambot and Frédéric André. "Full-Wave Modeling of Near-Field Radar Data for Planar Layered Media Reconstruction". In: *IEEE Transactions on Geoscience and Remote Sensing* 52.5 (2014), pp. 2295–2303. DOI: 10.1109/TGRS.2013.2259243.

[149] Sébastien Lambot, Evert C. Slob, Idesbald van den Bosch, Benoit Stockbroeckx, and Marnik Vanclooster. "Modeling of ground-penetrating Radar for accurate characterization of subsurface electric properties". In: *IEEE Transactions on Geoscience and Remote Sensing* 42.11 (2004), pp. 2555–2568. DOI: 10.1109/TGRS.2004.834800.

[150] Sébastien Lambot, L. Weihermüller, J. A. Huisman, H. Vereecken, M. Vanclooster, and Evert C. Slob. "Analysis of air-launched ground-penetrating radar techniques to measure the soil surface water content". In: *Water Resources Research* 42 (2006). DOI: 10.1029/2006WR005097.

[151] Sébastien Lambot, Kaijun Wu, Arthur Sluÿters, and Jean Vanderdonckt. "The full-wave radar equation for wave propagation in multilayered media and its applications". In: *Ground Penetrating Radar: From Theoretical Endeavors to Computational Electromagnetics Signal, Processing, Antenna Design and Field Applications*. Ed. by Mohammed Serhir and Dominique Lesselier. ISTE Ltd, in press, pp. 123–160. ISBN: 9781789451573.

[152] Shengchang Lan, Zonglong He, Weichu Chen, and Lijia Chen. "Hand Gesture Recognition Using Convolutional Neural Networks". In: *2018 USNC-URSI Radio Science Meeting (Joint with AP-S Symposium)*. July 2018, pp. 147–148. DOI: 10.1109/USNC-URSI.2018.8602809.

[153] Shengchang Lan, Zonglong He, Haoyu Tang, Kai Yao, and Wenshuang Yuan. "A hand gesture recognition system based on 24GHz radars". In: *2017 International Symposium on Antennas and Propagation (ISAP)*. Oct. 2017, pp. 1–2. DOI: 10.1109/ISANP.2017.8228827.

[154] Shengchang Lan, Zonglong He, Kai Yao, and Weichu Chen. "Hand Gesture Recognition using a Three-dimensional 24 GHz Radar Array". In: *2018 IEEE/MTT-S International Microwave Symposium - IMS*. ISSN: 2576-7216. June 2018, pp. 138–140. DOI: 10.1109/MWSYM.2018.8439658.

[155] Joseph J. LaViola. "3D Gestural Interaction: The State of the Field". In: *International Scholarly Research Notices* 2013 (2013). DOI: 10.1155/2013/514641. URL: https://www.hindawi.com/journals/isrn/2013/514641/.

[156] Joseph J. LaViola. "Double exponential smoothing: an alternative to Kalman filter-based predictive tracking". In: *Proceedings of the workshop on Virtual environments 2003*. EGVE '03. New York, NY, USA: Association for Computing Machinery, May 2003, pp. 199–206. ISBN: 978-1-58113-686-9. DOI: 10.1145/769953.769976. URL: https://doi.org/10.1145/769953.769976 (visited on 12/29/2020).

[157] Hyo Ryun Lee, Jihun Park, and Young-Joo Suh. "Improving Classification Accuracy of Hand Gesture Recognition Based on 60 GHz FMCW Radar with Deep Learning Domain Adaptation". In: *Electronics* 9.12 (2020), pp. 1–24. ISSN: 2079-9292. DOI: 10.3390/electronics9122140. URL: https://www.mdpi.com/2079-9292/9/12/2140.

[158] Seong Kyu Leem, Faheem Khan, and Sung Ho Cho. "Detecting Mid-Air Gestures for Digit Writing With Radio Sensors and a CNN". In: *IEEE Transactions on Instrumentation and Measurement* 69.4 (Apr. 2020), pp. 1066–1081. ISSN: 1557-9662. DOI: 10.1109/TIM.2019.2909249.

[159] Seong Kyu Leem, Faheem Khan, and Sung Ho Cho. "Remote Authentication Using an Ultra-Wideband Radio Frequency Transceiver". In: *2020 IEEE 17th Annual Consumer Communications Networking Conference (CCNC)*. ISSN: 2331-9860. Jan. 2020, pp. 1–8. DOI: 10.1109/CCNC46108.2020.9045438.

[160] Luis A. Leiva, Matjaz Kljun, Christian Sandor, and Klen Copic Pucihar. "The Wearable Radar: Sensing Gestures Through Fabrics". In: *22nd International Conference on Human-Computer Interaction with Mobile Devices and Services* (Oldenburg, Germany). MobileHCI '20. New York, NY, USA: Association for Computing Machinery, 2020. ISBN: 9781450380522. DOI: 10.1145/3406324.3410720. URL: https://doi.org/10.1145/3406324.3410720.

[161] Luis A. Leiva, Daniel Martín-Albo, and Réjean Plamondon. "Gestures à Go Go: Authoring Synthetic Human-Like Stroke Gestures Using the Kinematic Theory of Rapid Movements". In: *ACM Transactions on Intelligent Systems Technology* 7.2 (Nov. 2015). ISSN: 2157-6904. DOI: 10.1145/2799648. URL: https://doi.org/10.1145/2799648.

[162] Luis A. Leiva, Daniel Martín-Albo, and Radu-Daniel Vatavu. "GATO: Predicting Human Performance with Multistroke and Multitouch Gesture Input". In: *Proceedings of the 20th ACM International Conference on Human-Computer Interaction with Mobile Devices and Services* (Barcelona, Spain). MobileHCI '18. New York, NY, USA: Association for Computing Machinery, 2018. ISBN: 9781450358989. DOI: 10.1145/3229434.3229478. URL: https://doi.org/10.1145/3229434.3229478.

[163] James R. Lewis and Jeff Sauro. "The Factor Structure of the System Usability Scale". In: *Human Centered Design*. Ed. by Masaaki Kurosu. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 94–103. ISBN: 978-3-642-02806-9.

[164] Feifei Li, Yujun Li, Baozhen Du, Hongji Xu, Hailiang Xiong, and Min Chen. "A Gesture Interaction System Based on Improved Finger Feature and WE-KNN". In: *Proceedings of the 2019 4th International Conference on Mathematics and Artificial Intelligence* (Chegndu, China). ICMAI 2019. New York, NY, USA: Association for Computing Machinery, 2019, pp. 39–43. ISBN: 9781450362580. DOI: 10.1145/3325730.3325759. URL: https://doi.org/10.1145/3325730.3325759.

[165] Gang Li, Rui Zhang, Matthew Ritchie, and Hugh Griffiths. "Sparsity-based dynamic hand gesture recognition using micro-Doppler signatures". In: *2017 IEEE Radar Conference (RadarConf)*. ISSN: 2375-5318. May 2017, pp. 0928–0931. DOI: 10.1109/RADAR.2017.7944336.

[166] Gang Li, Rui Zhang, Matthew Ritchie, and Hugh Griffiths. "Sparsity-Driven Micro-Doppler Feature Extraction for Dynamic Hand Gesture Recognition". In: *IEEE Transactions on Aerospace and Electronic Systems* 54.2 (Apr. 2018), pp. 655–665. ISSN: 1557-9603. DOI: 10.1109/TAES.2017.2761229.

[167] Gang Li, Shimeng Zhang, Francesco Fioranelli, and Hugh Griffiths. "Effect of sparsity-aware time–frequency analysis on dynamic hand gesture classification with radar micro-Doppler signatures". In: *IET Radar, Sonar Navigation* 12.8 (2018), pp. 815–820. ISSN: 1751-8792. DOI: 10.1049/iet-rsn.2017.0570.

[168] Hao-Yang Li, Han-Ting Zhao, Meng-Lin Wei, Heng-Xin Ruan, Ya Shuang, Tie Jun Cui, Philipp del Hougne, and Lianlin Li. "Intelligent Electromagnetic Sensing with Learnable Data Acquisition and Processing". In: *Patterns* 1.1 (2020), p. 100006. ISSN: 2666-3899. DOI: https://doi.org/10.1016/j.patter.2020.100006. URL: http://www.sciencedirect.com/science/article/pii/S2666389920300064.

[169] Haobo Li, Xiangpeng Liang, Aman Shrestha, Yuchi Liu, Hadi Heidari, Julien Le Kernec, and Francesco Fioranelli. "Hierarchical Sensor Fusion for Micro-Gesture Recognition With Pressure Sensor Array and Radar". In: *IEEE Journal of Electromagnetics, RF and Microwaves in Medicine and Biology* 4.3 (Sept. 2020), pp. 225–232. ISSN: 2469-7257. DOI: 10.1109/JERM.2019.2949456.

[170] Mingxue Li, Yang Cong, Yuyang Liu, and Gan Sun. "Class-Incremental Gesture Recognition Learning with Out-of-Distribution Detection". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 1503–1508. DOI: 10.1109/IROS47612.2022.9981167.

[171] Qiming Li, Chen Huang, Shengqing Lv, Zeyu Li, Yimin Chen, and Lizhuang Ma. "An Human-Computer Interactive Augmented Reality System for Coronary Artery Diagnosis Planning and Training". In: *J. Med. Syst.* 41.10 (Oct. 2017), pp. 1–11. ISSN: 0148-5598. DOI: 10.1007/s10916-017-0805-5. URL: https://doi.org/10.1007/s10916-017-0805-5.

[172] Tianxing Li, Xi Xiong, Yifei Xie, George Hito, Xing-Dong Yang, and Xia Zhou. "Reconstructing Hand Poses Using Visible Light". In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1.3 (Sept. 2017). DOI: 10.1145/3130937. URL: https://doi.org/10.1145/3130937.

[173] Tuanjie Li and Mengmeng Ge. "Human motion recognition using ultra-wideband radar and cameras on mobile robot". en. In: *Transactions of Tianjin University* 15.5 (Oct. 2009), pp. 381–387. ISSN: 1006-4982, 1995-8196. DOI: 10.1007/s12209-009-0067-5. URL: http://link.springer.com/10.1007/s12209-009-0067-5 (visited on 12/21/2020).

[174] Huaiyuan Liang, Xiangrong Wang, Maria S. Greco, and Fulvio Gini. "Enhanced Hand Gesture Recognition using Continuous Wave Interferometric Radar". In: *2020 IEEE International Radar Conference (RADAR)*. ISSN: 2640-7736. Apr. 2020, pp. 226–231. DOI: 10.1109/RADAR42522.2020.9114807.

[175] Hui Liang, Jian Chang, Ismail K. Kazmi, Jian J. Zhang, and Peifeng Jiao. "Hand Gesture-Based Interactive Puppetry System to Assist Storytelling for Children". In: *Vis. Comput.* 33.4 (Apr. 2017), pp. 517–531. ISSN: 0178-2789. DOI: 10.1007/s00371-016-1272-6. URL: https://doi.org/10.1007/s00371-016-1272-6.

[176] Jaime Lien, Nicholas Gillian, M. Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev. "Soli: ubiquitous gesture sensing with millimeter wave radar". In: *ACM Transactions on Graphics* 35.4 (July 2016), 142:1–142:19. ISSN: 0730-0301. DOI: 10.1145/2897824.2925953. URL: https://doi.org/10.1145/2897824.2925953 (visited on 12/21/2020).

[177] Changjiang Liu, Yuanhao Li, Dongyang Ao, and Haiyan Tian. "Spectrum-Based Hand Gesture Recognition Using Millimeter-Wave Radar Parameter Measurements". In: *IEEE Access* 7 (2019), pp. 79147–79158. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2923122.

[178] Haipeng Liu, Yuheng Wang, Anfu Zhou, Hanyue He, Wei Wang, Kunpeng Wang, Peilin Pan, Yixuan Lu, Liang Liu, and Huadong Ma. "Real-time Arm Gesture Recognition in Smart Home Scenarios via Millimeter Wave Sensing". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4.4 (Dec. 2020), 140:1–140:28. DOI: 10.1145/3432235. URL: https://doi.org/10.1145/3432235 (visited on 12/21/2020).

[179] Mingyu Liu, Mathieu Nancel, and Daniel Vogel. "Gunslinger: Subtle Arms-down Mid-air Interaction". In: *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. UIST '15. Charlotte, NC, USA: Association for Computing Machinery, 2015, pp. 63–71. ISBN: 9781450337793. DOI: 10.1145/2807442.2807489. URL: https://doi.org/10.1145/2807442.2807489.

[180] Yu Liu, Yuheng Wang, Haipeng Liu, Anfu Zhou, Jianhua Liu, and Ning Yang. "Long-Range Gesture Recognition Using Millimeter Wave Radar". en. In: *Green, Pervasive, and Cloud Computing*. Ed. by Zhiwen Yu, Christian Becker, and Guoliang Xing. Vol. 12398. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 30–44. ISBN: 978-3-030-64242-6 978-3-030-64243-3. DOI: 10.1007/978-3-030-64243-3_3. URL: http://link.springer.com/10.1007/978-3-030-64243-3_3 (visited on 12/21/2020).

[181] Olga Lopera, Evert C. Slob, Nada Milisavljevic, and Sébastien Lambot. "Filtering Soil Surface and Antenna Effects From GPR Data to Enhance Landmine Detection". In: *IEEE Transactions on Geoscience and Remote*

Sensing 45.3 (Mar. 2007), pp. 707–717. DOI: 10.1109/TGRS.2006.888136. URL: https://ieeexplore.ieee.org/document/4106057.

[182] Gustavo López, Luis Quesada, and Luis A. Guerrero. "A Gesture-Based Interaction Approach for Manipulating Augmented Objects Using Leap Motion". In: *Ambient Assisted Living. ICT-based Solutions in Real Life Situations*. Ed. by Ian Cleland, Luis Guerrero, and José Bravo. Cham: Springer International Publishing, 2015, pp. 231–243. ISBN: 978-3-319-26410-3.

[183] Xinye Lou, Zhiwen Yu, Zhu Wang, Kaijie Zhang, and Bin Guo. "Gesture-Radar: Enabling Natural Human-Computer Interactions with Radar-Based Adaptive and Robust Arm Gesture Recognition". In: *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. ISSN: 2577-1655. Oct. 2018, pp. 4291–4297. DOI: 10.1109/SMC.2018.00726.

[184] Laurens van der Maaten, Eric Postma, and Jaap van den Herik. "Dimensionality Reduction: A Comparative Review". In: *Journal of Machine Learning Research* 10 (Jan. 2009), pp. 66–71. URL: https://members.loria.fr/moberger/Enseignement/AVR/Exposes/TR_Dimensiereductie.pdf.

[185] Mehran Maghoumi and Joseph J. LaViola. "DeepGRU: Deep Gesture Recognition Utility". In: *Advances in Visual Computing*. Ed. by George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Daniela Ushizima, Sek Chai, Shinjiro Sueda, Xin Lin, Aidong Lu, Daniel Thalmann, Chaoli Wang, and Panpan Xu. Cham: Springer International Publishing, 2019, pp. 16–31. ISBN: 978-3-030-33720-9.

[186] Nathan Magrofuoco, Paolo Roselli, and Jean Vanderdonckt. "Two-Dimensional Stroke Gesture Recognition: A Survey". In: *ACM Comput. Surv.* 54.7 (July 2021). ISSN: 0360-0300. DOI: 10.1145/3465400. URL: https://doi.org/10.1145/3465400.

[187] Nathan Magrofuoco, Paolo Roselli, Jean Vanderdonckt, Jorge Luis Pérez-Medina, and Radu-Daniel Vatavu. "GestMan: A Cloud-Based Tool for Stroke-Gesture Datasets". In: *Proceedings of the ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Valencia, Spain). EICS '19. New York, NY, USA: Association for Computing Machinery, 2019. ISBN: 9781450367455. DOI: 10.1145/3319499.3328227. URL: https://doi.org/10.1145/3319499.3328227.

[188] Greg Malysa, Dan Wang, Lorin Netsch, and Murtaza Ali. "Hidden Markov model-based gesture recognition with FMCW radar". In: *2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. Dec. 2016, pp. 1017–1021. DOI: 10.1109/GlobalSIP.2016.7905995.

[189] Rajesh B. Mapari and Govind Kharat. "American Static Signs Recognition Using Leap Motion Sensor". In: *Proc. of the Second International Conference on Information and Communication Technology for Competitive Strategies* (Udaipur, India). ICTCS '16. New York, NY, USA: Association for Computing Machinery, 2016. ISBN: 9781450339629. DOI: 10.1145/2905055.2905125. URL: https://doi.org/10.1145/2905055.2905125.

[190] Giulio Marin, Fabio Dominio, and Pietro Zanuttigh. "Hand Gesture Recognition with Jointly Calibrated Leap Motion and Depth Sensor". In: *Multimedia Tools Appl.* 75.22 (Nov. 2016), pp. 14991–15015. ISSN: 1380-7501. DOI: 10.1007/s11042-015-2451-6. URL: https://doi.org/10.1007/s11042-015-2451-6.

[191] Giulio Marin, Fabio Dominio, and Pietro Zanuttigh. "Hand gesture recognition with Leap Motion and Kinect devices". In: *2014 IEEE International Conference on Image Processing, ICIP 2014* (2014), pp. 1565–1569. DOI: 10.1109/ICIP.2014.7025313.

[192] Donald W. Marquardt. "An Algorithm for Least-Squares Estimation of Nonlinear Parameters". In: *Journal of the Society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441. DOI: 10.1137/0111030. eprint: https://doi.org/10.1137/0111030. URL: https://doi.org/10.1137/0111030.

[193] Jess McIntosh, Mike Fraser, Paul Worgan, and Asier Marzo. "DeskWave: Desktop Interactions using Low-cost Microwave Doppler Arrays". In: *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. CHI EA '17. New York, NY, USA: Association for Computing Machinery, May 2017, pp. 1885–1892. ISBN: 978-1-4503-4656-6. DOI: 10.1145/3027063.3053152. URL: https://doi.org/10.1145/3027063.3053152 (visited on 12/21/2020).

[194] David McNeill. "Hand and Mind: What Gestures Reveal About Thought". In: *The University of Chicago Press* 27 (1995). DOI: 10.2307/1576015.

[195] Roi Méndez, Julián Flores, Enrique Castelló, and José Ramón Viqueira. "New Distributed Virtual TV Set Architecture for a Synergistic Operation of Sensors and Improved Interaction between Real and Virtual Worlds". In: *Multimedia Tools Appl.* 77.15 (Aug. 2018), pp. 18999–19025.

ISSN: 1380-7501. DOI: 10.1007/s11042-017-5353-y. URL: https://doi.org/10.1007/s11042-017-5353-y.

[196] Krzysztof A. Michalski and Dalian Zheng. "Electromagnetic scattering and radiation by subsurfaces of arbitrary shape in layered media: Parts I and II". In: *IEEE Transactions on Antennas and Propagation* 38 (1990), pp. 335–352.

[197] Elishiah Miller, Zheng Li, Helena Mentis, Adrian Park, Ting Zhu, and Nilanjan Banerjee. "RadSense: Enabling one hand and no hands interaction for sterile manipulation of medical images using Doppler radar". In: *Smart Health* 15 (2020), p. 100089. ISSN: 2352-6483. DOI: https://doi.org/10.1016/j.smhl.2019.100089. URL: http://www.sciencedirect.com/science/article/pii/S2352648319300534.

[198] Adrien Moinnet and Benoît Gosselin. "Composition of strokes gestures into system instructions". Anglais. PhD thesis. UCL - Ecole polytechnique de Louvain, 2022. URL: http://hdl.handle.net/2078.1/thesis:37872.

[199] Pavlo Molchanov, Shalini Gupta, Kihwan Kim, and Kari Pulli. "Multi-sensor system for driver's hand-gesture recognition". In: *2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG)*. Vol. 1. May 2015, pp. 1–8. DOI: 10.1109/FG.2015.7163132.

[200] Meredith Ringel Morris. "Web on the Wall: Insights from a Multimodal Interaction Elicitation Study". In: *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces* (Cambridge, Massachusetts, USA). ITS '12. New York, NY, USA: Association for Computing Machinery, 2012, pp. 95–104. ISBN: 9781450312097. DOI: 10.1145/2396636.2396651. URL: https://doi.org/10.1145/2396636.2396651.

[201] Meredith Ringel Morris, Jacob O. Wobbrock, and Andrew D. Wilson. "Understanding Users' Preferences for Surface Gestures". In: *Proceedings of Graphics Interface 2010* (Ottawa, Ontario, Canada). GI '10. CAN: Canadian Information Processing Society, 2010, pp. 261–268. ISBN: 9781568817125.

[202] Christopher Munroe, Yuanliang Meng, Holly Yanco, and Momotaz Begum. "Augmented reality eyeglasses for promoting home-based rehabilitation for children with cerebral palsy". In: *2016 11th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. 2016, pp. 565–565. DOI: 10.1109/HRI.2016.7451858.

[203] Miguel A. Nacenta, Yemliha Kamber, Yizhou Qiang, and Per Ola Kristensson. "Memorability of Pre-Designed and User-Defined Gesture Sets". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France). CHI '13. New York, NY, USA: Association for Computing Machinery, 2013, pp. 1099–1108. ISBN: 9781450318990. DOI: 10.1145/2470654.2466142. URL: https://doi.org/10.1145/2470654.2466142.

[204] Mathieu Nancel, Julie Wagner, Emmanuel Pietriga, Olivier Chapuis, and Wendy Mackay. "Mid-Air Pan-and-Zoom on Wall-Sized Displays". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada). CHI '11. New York, NY, USA: Association for Computing Machinery, 2011, pp. 177–186. ISBN: 9781450302289. DOI: 10.1145/1978942.1978969. URL: https://doi.org/10.1145/1978942.1978969.

[205] Romain Neuville. "Air+Touch gesture recognition : algorithms, software, and experiment". Anglais. PhD thesis. UCL - Ecole polytechnique de Louvain, 2021. URL: http://hdl.handle.net/2078.1/thesis:33024.

[206] Minh Q. Nguyen, Anthony Flores-Nigaglioni, and Changzhi Li. "Range-gating technology for millimeter-wave radar remote gesture control in IoT applications". In: *2018 IEEE MTT-S International Wireless Symposium (IWS)*. May 2018, pp. 1–4. DOI: 10.1109/IEEE-IWS.2018.8400811.

[207] Minh Q. Nguyen and Changzhi Li. "Radar and ultrasound hybrid system for human computer interaction". In: *2018 IEEE Radar Conference (RadarConf18)*. ISSN: 2375-5318. Apr. 2018, pp. 1476–1480. DOI: 10.1109/RADAR.2018.8378783.

[208] Jakob Nielsen. *Usability Engineering*. Interactive Technologies. Elsevier Science, 1994. ISBN: 9780125184069. URL: https://books.google.be/books?id=95As2OF67f0C.

[209] Donald A. Norman. "Affordance, conventions, and design". In: *Interactions* 6.3 (May 1999), pp. 38–43. ISSN: 1072-5520. DOI: 10.1145/301153.301168. URL: https://doi.org/10.1145/301153.301168.

[210] Donald A. Norman. "Natural user interfaces are not natural". In: *Interactions* 17.3 (May 2010), pp. 6–10. ISSN: 1072-5520. DOI: 10.1145/1744161.1744163. URL: https://doi.org/10.1145/1744161.1744163.

[211] Donald A. Norman. "Signifiers, not affordances". In: *Interactions* 15.6 (Nov. 2008), pp. 18–19. ISSN: 1072-5520. DOI: 10.1145/1409040.1409044. URL: https://doi.org/10.1145/1409040.1409044.

[212] Joachim Oberhammer, Nutapong Somjit, Umer Shah, and Zargham Baghchehsaraei. "RF MEMS for automotive radar". In: *Handbook of Mems for Wireless and Mobile Applications*. Ed. by Deepak Uttamchandani. Woodhead Publishing Series in Electronic and Optical Materials. Woodhead Publishing, 2013, pp. 518–549. ISBN: 978-0-85709-271-7. DOI: https://doi.org/10.1533/9780857098610.2.518. URL: https://www.sciencedirect.com/science/article/pii/B9780857092717500165.

[213] Uran Oh and Leah Findlater. "The challenges and potential of end-user gesture customization". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: Association for Computing Machinery, 2013, pp. 1129–1138. ISBN: 9781450318990. DOI: 10.1145/2470654.2466145. URL: https://doi.org/10.1145/2470654.2466145.

[214] Francisco R. Ortega, Alain Galvan, Katherine Tarre, Armando Barreto, Naphtali Rishe, Jonathan Bernal, Ruben Balcazar, and Jason-Lee Thomas. "Gesture elicitation for 3D travel via multi-touch and mid-Air systems for procedurally generated pseudo-universe". In: *2017 IEEE Symposium on 3D User Interfaces (3DUI)*. 2017, pp. 144–153. DOI: 10.1109/3DUI.2017.7893331.

[215] Mehdi Ousmer, Arthur Sluÿters, Nathan Magrofuoco, Paolo Roselli, and Jean Vanderdonckt. "Recognizing 3D Trajectories as 2D Multi-Stroke Gestures". In: *Proc. ACM Hum.-Comput. Interact.* 4.ISS (Nov. 2020). DOI: 10.1145/3427326. URL: https://doi.org/10.1145/3427326.

[216] Tom Ouyang and Yang Li. "Bootstrapping personal gesture shortcuts with the wisdom of the crowd and handwriting recognition". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '12. Austin, Texas, USA: Association for Computing Machinery, 2012, pp. 2895–2904. ISBN: 9781450310154. DOI: 10.1145/2207676.2208695. URL: https://doi.org/10.1145/2207676.2208695.

[217] Salih Ertug Ovur, Hang Su, Wen Qi, Elena De Momi, and Giancarlo Ferrigno. "Novel Adaptive Sensor Fusion Methodology for Hand Pose Estimation With Multileap Motion". In: *IEEE Transactions on Instrumentation and Measurement* 70 (2021), pp. 1–8. DOI: 10.1109/TIM.2021.3063752.

[218] Matthew J Page, David Moher, Patrick M Bossuyt, Isabelle Boutron, Tammy C Hoffmann, Cynthia D Mulrow, Larissa Shamseer, Jennifer M Tetzlaff, Elie A Akl, Sue E Brennan, Roger Chou, Julie Glanville, Jeremy M Grimshaw, Asbjørn Hróbjartsson, Manoj M Lalu, Tianjing Li, Elizabeth W Loder, Evan Mayo-Wilson, Steve McDonald, Luke A McGuinness, Lesley A Stewart, James Thomas, Andrea C Tricco, Vivian A Welch, Penny Whiting, and Joanne E McKenzie. "PRISMA 2020 explanation and elaboration: updated guidance and exemplars for reporting systematic reviews". In: *BMJ* 372 (2021), pp. 1–36. DOI: 10.1136/bmj.n160. eprint: https://www.bmj.com/content/372/bmj.n160.full.pdf. URL: https://www.bmj.com/content/372/bmj.n160.

[219] Sameera Palipana, Dariush Salami, Luis A. Leiva, and Stephan Sigg. "Pantomime: Mid-Air Gesture Recognition with Sparse Millimeter-Wave Radar Point Clouds". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 5.1 (Mar. 2021), 27:1–27:27. DOI: 10.1145/3448110. URL: https://doi.org/10.1145/3448110 (visited on 05/25/2021).

[220] Chanho Park, Hyunwoo Cho, Sangheon Park, Young-Suk Yoon, and Sung-Uk Jung. "HandPoseMenu: Hand Posture-Based Virtual Menus for Changing Interaction Mode in 3D Space". In: *Proceedings of the 2019 ACM International Conference on Interactive Surfaces and Spaces* (Daejeon, Republic of Korea). ISS '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 361–366. ISBN: 9781450368919. DOI: 10.1145/3343055.3360752. URL: https://doi.org/10.1145/3343055.3360752.

[221] Junbum Park and Sung Ho Cho. "IR-UWB Radar Sensor for Human Gesture Recognition by Using Machine Learning". In: *2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. Dec. 2016, pp. 1246–1249. DOI: 10.1109/HPCC-SmartCity-DSS.2016.0176.

[222] Jungwoon Park, Junyoung Jang, Geunhaeng Lee, Hyunmin Koh, Changhwan Kim, and Tae Wook Kim. "A Time Domain Artificial Intelligence Radar System Using 33-GHz Direct Sampling for Hand Gesture Recognition". In: *IEEE Journal of Solid-State Circuits* 55.4 (Apr. 2020), pp. 879–888. ISSN: 1558-173X. DOI: 10.1109/JSSC.2020.2967547.

[223] David Parsons, Awais Rashid, Andreas Speck, and Alexandru C. Telea. "A "framework" for object oriented frameworks design". In: *Proceedings Technology of Object-Oriented Languages and Systems. TOOLS 29 (Cat. No.PR00275)*. 1999, pp. 141–151. DOI: 10.1109/TOOLS.1999.779007. URL: https://ieeexplore.ieee.org/author/37268563800.

[224] Vik Parthiban and Ashley Jieun Lee. "LUI: A Multimodal, Intelligent Interface for Large Displays". In: *The 17th International Conference on Virtual-Reality Continuum and Its Applications in Industry* (Brisbane, QLD, Australia). VRCAI '19. New York, NY, USA: Association for Computing Machinery, 2019. ISBN: 9781450370028. DOI: 10.1145/3359997.3365743. URL: https://doi.org/10.1145/3359997.3365743.

[225] Avishek Patra, Philipp Geuer, Andrea Munari, and Petri Mähönen. "mm-Wave Radar Based Gesture Recognition: Development and Evaluation of a Low-Power, Low-Complexity System". In: *Proceedings of the 2nd ACM Workshop on Millimeter Wave Networks and Sensing Systems*. mmNets '18. New York, NY, USA: Association for Computing Machinery, Oct. 2018, pp. 51–56. ISBN: 978-1-4503-5928-3. DOI: 10.1145/3264492.3264501. URL: https://doi.org/10.1145/3264492.3264501 (visited on 12/21/2020).

[226] Andriy Pavlovych and Wolfgang Stuerzlinger. "The tradeoff between spatial jitter and latency in pointing tasks". In: *Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems*. EICS '09. New York, NY, USA: Association for Computing Machinery, July 2009, pp. 187–196. ISBN: 978-1-60558-600-7. DOI: 10.1145/1570433.1570469. URL: https://doi.org/10.1145/1570433.1570469 (visited on 01/07/2021).

[227] Tran Pham, Jo Vermeulen, Anthony Tang, and Lindsay MacDonald Vermeulen. "Scale Impacts Elicited Gestures for Manipulating Holograms: Implications for AR Gesture Design". In: *Proceedings of the 2018 Designing Interactive Systems Conference* (Hong Kong, China). DIS '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 227–240. ISBN: 9781450351980. DOI: 10.1145/3196709.3196719. URL: https://doi.org/10.1145/3196709.3196719.

[228] Thammathip Piumsomboon, Adrian Clark, Mark Billinghurst, and Andy Cockburn. "User-Defined Gestures for Augmented Reality". en. In: *Human-Computer Interaction – INTERACT 2013*. Ed. by Paula Kotzé, Gary Marsden, Gitte Lindgaard, Janet Wesson, and Marco Winckler. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2013, pp. 282–299. ISBN: 978-3-642-40480-1. DOI: 10.1007/978-3-642-40480-1_18.

[229] Godwin Ponraj and Hongliang Ren. "Sensor Fusion of Leap Motion Controller and Flex Sensors Using Kalman Filter for Human Finger Tracking". In: *IEEE Sensors Journal* 18.5 (2018), pp. 2042–2049. DOI: 10.1109/JSEN.2018.2790801.

[230] Wolfgang Pree. "Meta Patterns - A Means For Capturing the Essentials of Reusable Object-Oriented Design". In: *Proceedings of the 8th European Conference on Object-Oriented Programming*. ECOOP '94. Berlin, Heidelberg: Springer-Verlag, 1994, pp. 150–162. ISBN: 3540582029.

[231] Alexandros Preventis, Kostas Stravoskoufos, Stelios Sotiriadis, and Euripides G. M. Petrakis. "Interact: Gesture Recognition in the Cloud". In: *Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*. UCC '14. USA: IEEE Computer Society, 2014, pp. 501–502. ISBN: 9781479978816. DOI: 10.1109/UCC.2014.71. URL: https://doi.org/10.1109/UCC.2014.71.

[232] Isabelle Rawart. *Citizens of Wallonia 2023. 9 projets innovants récompensés*. https://www.digitalwallonia.be/fr/publications/citizens-of-wallonia-2023-9-projets-innovants/. [Online; accessed 20-November-2023]. 2023.

[233] Aihu Ren, Yong Wang, Xiaobo Yang, and Mu Zhou. "A Dynamic Continuous Hand Gesture Detection and Recognition Method with FMCW Radar". In: *2020 IEEE/CIC International Conference on Communications in China (ICCC)*. ISSN: 2377-8644. Aug. 2020, pp. 1208–1213. DOI: 10.1109/ICCC49849.2020.9238935.

[234] Matthew Ritchie, Richard Capraru, and Francesco Fioranelli. "DopNET: a micro-Doppler radar data challenge". In: *Electronics Letters* 56.11 (2020), pp. 568–570. ISSN: 0013-5194. DOI: 10.1049/el.2019.4153.

[235] Matthew Ritchie, A. Jones, J. Brown, and Hugh D. Griffiths. "Hand gesture classification using 24 GHz FMCW dual polarised radar". In: *International Conference on Radar Systems (Radar 2017)*. Oct. 2017, pp. 1–6. DOI: 10.1049/cp.2017.0482.

[236] Susan Rose, Moira Clark, Phillip Samouel, and Neil Hair. "Online Customer Experience in e-Retailing: An empirical model of Antecedents and Outcomes". In: *Journal of Retailing* 88 (June 2012), pp. 308–322. DOI: 10.1016/j.jretai.2012.03.001.

[237] Gustavo Rovelo, Donald Degraen, Davy Vanacken, Kris Luyten, and Karin Coninx. "Gestu-Wan - An Intelligible Mid-Air Gesture Guidance System for Walk-up-and-Use Displays". In: *Human-Computer Interaction – INTERACT 2015*. Ed. by Julio Abascal, Simone Barbosa, Mirko Fetter, Tom Gross, Philippe Palanque, and Marco Winckler. Cham:

Springer International Publishing, 2015, pp. 368–386. ISBN: 978-3-319-22668-2.

[238] Quentin Roy, Sylvain Malacria, Yves Guiard, Eric Lecolinet, and James Eagan. "Augmented Letters: Mnemonic Gesture-Based Shortcuts". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2013, pp. 2325–2328. ISBN: 9781450318990. URL: https://doi.org/10.1145/2470654.2481321.

[239] Josip Rožman, Hani Hagras, Javier Andreu Perez, Damien Clarke, Beate Müller, and Steve Fitz Data. "Privacy-Preserving Gesture Recognition with Explainable Type-2 Fuzzy Logic Based Systems". In: *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. ISSN: 1558-4739. July 2020, pp. 1–8. DOI: 10.1109/FUZZ48607.2020.9177768.

[240] Dean Rubine. "Specifying Gestures by Example". In: *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '91. New York, NY, USA: ACM, 1991, pp. 329–337. ISBN: 0-89791-436-8. DOI: 10.1145/122718.122753. URL: http://doi.acm.org/10.1145/122718.122753.

[241] Jaime Ruiz, Yang Li, and Edward Lank. "User-Defined Motion Gestures for Mobile Interaction". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada). CHI '11. New York, NY, USA: Association for Computing Machinery, 2011, pp. 197–206. ISBN: 9781450302289. DOI: 10.1145/1978942.1978971. URL: https://doi.org/10.1145/1978942.1978971.

[242] Si-Jung Ryu, Jun-Seuk Suh, Seung-Hwan Baek, Songcheol Hong, and Jong-Hwan Kim. "Feature-Based Hand Gesture Recognition Using an FMCW Radar and its Temporal Feature Analysis". In: *IEEE Sensors Journal* 18.18 (Sept. 2018), pp. 7593–7602. ISSN: 1558-1748. DOI: 10.1109/JSEN.2018.2859815.

[243] Ahmed R. Sadik, Bodo Urban, and Omar Adel. "Using Hand Gestures to Interact with an Industrial Robot in a Cooperative Flexible Manufacturing Scenario". In: *Proceedings of the 3rd International Conference on Mechatronics and Robotics Engineering* (Paris, France). ICMRE 2017. New York, NY, USA: Association for Computing Machinery, 2017, pp. 11–16. ISBN: 9781450352802. DOI: 10.1145/3068796.3068801. URL: https://doi.org/10.1145/3068796.3068801.

[244] Takuya Sakamoto, Xiaomeng Gao, Ehsan Yavari, Ashikur Rahman, Olga Boric-Lubecke, and Victor M. Lubecke. "Hand Gesture Recognition Using a Radar Echo I–Q Plot and a Convolutional Neural Network". In: *IEEE Sensors Letters* 2.3 (Sept. 2018), pp. 1–4. ISSN: 2475-1472. DOI: 10.1109/LSENS.2018.2866371.

[245] Takuya Sakamoto, Xiaomeng Gao, Ehsan Yavari, Ashikur Rahman, Olga Boric-Lubecke, and Victor M. Lubecke. "Radar-based hand gesture recognition using I-Q echo plot and convolutional neural network". In: *2017 IEEE Conference on Antenna Measurements Applications (CAMA)*. Dec. 2017, pp. 393–395. DOI: 10.1109/CAMA.2017.8273461.

[246] Panneer Selvam Santhalingam, Yuanqi Du, Riley Wilkerson, Al Amin Hosain, Ding Zhang, Parth Pathak, Huzefa Rangwala, and Raja Kushalnagar. "Expressive ASL Recognition using Millimeter-wave Wireless Signals". In: *2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. ISSN: 2155-5494. June 2020, pp. 1–9. DOI: 10.1109/SECON48991.2020.9158441.

[247] Panneer Selvam Santhalingam, Al Amin Hosain, Ding Zhang, Parth Pathak, Huzefa Rangwala, and Raja Kushalnagar. "mmASL: Environment-Independent ASL Gesture Recognition Using 60 GHz Millimeter-wave Signals". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4.1 (Mar. 2020), 26:1–26:30. DOI: 10.1145/3381010. URL: https://doi.org/10.1145/3381010 (visited on 12/21/2020).

[248] Jeff Sauro. *A practical guide to measuring usability: 72 answers to the most common questions about quantifying the usability of websites and software*. Measuring Usability LCC, 2010.

[249] Monika Schak and Alexander Gepperth. "A Study on Catastrophic Forgetting in Deep LSTM Networks". In: *Artificial Neural Networks and Machine Learning – ICANN 2019: Deep Learning*. Ed. by Igor V. Tetko, Věra Kůrková, Pavel Karpov, and Fabian Theis. Cham: Springer International Publishing, 2019, pp. 714–728. ISBN: 978-3-030-30484-3.

[250] Jacob Schioppo, Zachary Meyer, Diego Fabiano, and Shaun Canavan. "Sign Language Recognition: Learning American Sign Language in a Virtual Environment". In: *Extended Abstracts of the ACM Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk). CHI EA '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1–6. ISBN: 9781450359719. DOI: 10.1145/3290607.3313025. URL: https://doi.org/10.1145/3290607.3313025.

[251] Martin Schrepp, Andreas Hinderks, and Jörg Thomaschewski. "Construction of a Benchmark for the User Experience Questionnaire (UEQ)". In: *International Journal of Interactive Multimedia and Artificial Intelligence* 4 (June 2017), pp. 40–44. DOI: 10.9781/ijimai.2017.445.

[252] Quentin Sellier, Arthur Sluÿters, Jean Vanderdonckt, and Ingrid Poncin. "Evaluating gesture user interfaces: Quantitative measures, qualitative scales, and method". In: *International Journal of Human-Computer Studies* (2024), p. 103242. ISSN: 1071-5819. DOI: https://doi.org/10.1016/j.ijhcs.2024.103242. URL: https://www.sciencedirect.com/science/article/pii/S1071581924000260.

[253] Vinoth Pandian Sermuga Pandian and Sarah Suleri. *NASA-TLX Web App: An Online Tool to Analyse Subjective Workload*. 2020. arXiv: 2001.09963 [cs.HC].

[254] Jia Sheng. *A Study of AdaBoost in 3D Gesture Recognition*. technical report CSC2515. Department of Computer Science, University of Toronto, 2004. URL: http://www.dgp.toronto.edu/~jsheng/doc/CSC2515/Report.pdf.

[255] Ben Shneiderman. "Tree Visualization with Tree-Maps: 2-d Space-Filling Approach". In: *ACM Trans. Graph.* 11.1 (Jan. 1992), pp. 92–99. ISSN: 0730-0301. DOI: 10.1145/102377.115768. URL: https://doi.org/10.1145/102377.115768.

[256] Shaishav Siddhpuria, Keiko Katsuragawa, James R. Wallace, and Edward Lank. "Exploring At-Your-Side Gestural Interaction for Ubiquitous Environments". In: *Proceedings of the 2017 Conference on Designing Interactive Systems* (Edinburgh, United Kingdom). DIS '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 1111–1122. ISBN: 9781450349222. DOI: 10.1145/3064663.3064695. URL: https://doi.org/10.1145/3064663.3064695.

[257] Alexandru-Ionut Siean, Cristian Pamparău, Arthur Sluÿters, Radu-Daniel Vatavu, and Jean Vanderdonckt. "Flexible gesture input with radars: systematic literature review and taxonomy of radar sensing integration in ambient intelligence environments". In: *Journal of Ambient Intelligence and Humanized Computing* (2023). DOI: 10.1007/s12652-023-04606-9. URL: https://link.springer.com/article/10.1007/s12652-023-04606-9.

[258] Alexandru-Ionut Siean, Cristian Pamparău, and Radu-Daniel Vatavu. "Scenario-Based Exploration of Integrating Radar Sensing into Everyday Objects for Free-Hand Television Control". In: *Proceedings of ACM*

International Conference on Interactive Media Experiences (Aveiro, JB, Portugal). IMX '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 357–362. ISBN: 9781450392129. DOI: 10.1145/3505284.3532982. URL: https://doi.org/10.1145/3505284.3532982.

[259] Beat Signer, Ueli Kurmann, and Moira C. Norrie. "iGesture: A General Gesture Recognition Framework". In: *9th International Conference on Document Analysis and Recognition (ICDAR 2007), 23-26 September, Curitiba, Paraná, Brazil*. IEEE Computer Society, 2007, pp. 954–958. DOI: 10.1109/ICDAR.2007.4377056. URL: https://doi.org/10.1109/ICDAR.2007.4377056.

[260] Merkourios Simos and Nikolaos Nikolaidis. "Greek Sign Language Alphabet Recognition Using the Leap Motion Device". In: *Proceedings of the 9th Hellenic Conference on Artificial Intelligence* (Thessaloniki, Greece). SETN '16. New York, NY, USA: Association for Computing Machinery, 2016. ISBN: 9781450337342. DOI: 10.1145/2903220.2903249. URL: https://doi.org/10.1145/2903220.2903249.

[261] Sruthy Skaria, Akram Al-Hourani, and Robin J. Evans. "Deep-Learning Methods for Hand-Gesture Recognition Using Ultra-Wideband Radar". In: *IEEE Access* 8 (2020), pp. 203580–203590. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3037062.

[262] Sruthy Skaria, Akram Al-Hourani, Margaret Lech, and Robin J. Evans. "Hand-Gesture Recognition Using Two-Antenna Doppler Radar With Deep Convolutional Neural Networks". In: *IEEE Sensors Journal* 19.8 (Apr. 2019), pp. 3041–3048. ISSN: 1558-1748. DOI: 10.1109/JSEN.2019.2892073.

[263] Sruthy Skaria, Da Huang, Akram Al-Hourani, Robin J. Evans, and Margaret Lech. "Deep-Learning for Hand-Gesture Recognition with Simultaneous Thermal and Radar Sensors". In: *2020 IEEE Sensors*. ISSN: 2168-9229. Oct. 2020, pp. 1–4. DOI: 10.1109/SENSORS47125.2020.9278683.

[264] Evert Slob and Jacob Fokkema. "Coupling effects of two electric dipoles on an interface". In: *Radio Science* 37.5 (2002), pp. 1–10. DOI: 10.1029/2001RS2529.

[265] Arthur Sluÿters, Sébastien Lambot, and Jean Vanderdonckt. "Hand Gesture Recognition for an Off-the-Shelf Radar by Electromagnetic Modeling and Inversion". In: *Proceedings of 27th International ACM Conference on Intelligent User Interfaces* (Helsinki, Finland). IUI '22. New York, NY, USA: Association for Computing Machinery, 2022, pp. 506–

522. ISBN: 9781450391443. DOI: 10.1145/3490099.3511107. URL: https://doi.org/10.1145/3490099.3511107.

[266] Arthur Sluÿters, Sébastien Lambot, Jean Vanderdonckt, and Radu-Daniel Vatavu. "RadarSense: Accurate Recognition of Mid-Air Hand Gestures with Radar Sensing and Few Training Examples". In: *ACM Transactions on Interactive Intelligent Systems* (Mar. 2023). ISSN: 2160-6455. DOI: 10.1145/3589645. URL: https://doi.org/10.1145/3589645.

[267] Arthur Sluÿters, Sébastien Lambot, Jean Vanderdonckt, and Santiago Villarreal-Narvaez. "Analysis of User-defined Radar-based Hand Gestures Sensed through Multiple Materials". In: *IEEE Access* 12 (Feb. 2024). DOI: 10.1109/ACCESS.2024.3366667. URL: https://doi.org/10.1109/ACCESS.2024.3366667.

[268] Arthur Sluÿters, Mehdi Ousmer, Paolo Roselli, and Jean Vanderdonckt. "QuantumLeap, a Framework for Engineering Gestural User Interfaces based on the Leap Motion Controller". In: *Proc. ACM Human-Computer Interaction* 6.EICS (June 2022), pp. 1–47. DOI: 10.1145/3532211. URL: https://doi.org/10.1145/3532211.

[269] Arthur Sluÿters, Quentin Sellier, Jean Vanderdonckt, Vik Parthiban, and Pattie Maes. "Consistent, Continuous, and Customizable Mid-Air Gesture Interaction for Browsing Multimedia Objects on Large Displays". In: *International Journal of Human–Computer Interaction* (2022), pp. 1–32. DOI: 10.1080/10447318.2022.2078464. eprint: https://doi.org/10.1080/10447318.2022.2078464. URL: https://doi.org/10.1080/10447318.2022.2078464.

[270] Karly A. Smith, Clément Csech, David Murdoch, and George Shaker. "Gesture Recognition Using mm-Wave Sensor for Human-Car Interface". In: *IEEE Sensors Letters* 2.2 (June 2018), pp. 1–4. ISSN: 2475-1472. DOI: 10.1109/LSENS.2018.2810093.

[271] Maarten van Someren, Yvonne Barnard, and Jacobijn Sandberg. *The Think Aloud Method - A Practical Guide to Modelling Cognitive Processes*. London: Academic Press, 1994.

[272] Mischa Spiegelmock. *Leap Motion Development Essentials*. 1st ed. Packt Publishing, Oct. 2013. ISBN: 978-1849697729. URL: https://www.packtpub.com/eu/hardware-and-creative/leap-motion-development-essentials.

[273] Sarah Steeman and Chloé Chauvaux. "Uni/Bi-manual wrist gesture interaction for one/two users : application to Kinemic". Anglais. PhD thesis. UCL - Ecole polytechnique de Louvain, 2022. URL: http://hdl.handle.net/2078.1/thesis:35604.

[274] Ivo Stinghen and Bernardo Gatto. "Gesture Recognition Using Leap Motion: A Machine Learning-based Controller Interface". In: *17th Brazilian Symposium on Computer Games and Digital Entertainment* (Foz do Iguaçu, Brazil). SBGames '18. IEEE Computer Society, Nov. 2018. ISBN: 978-1-5386-9605-7.

[275] Jun Seuk Suh, Siiung Ryu, Bvunghun Han, Jaewoo Choi, Jong-Hwan Kim, and Songcheol Hong. "24 GHz FMCW Radar System for Real-Time Hand Gesture Recognition Using LSTM". In: *2018 Asia-Pacific Microwave Conference (APMC)*. Nov. 2018, pp. 860–862. DOI: 10.23919/APMC.2018.8617375.

[276] Yuliang Sun, Tai Fei, Shangyin Gao, and Nils Pohl. "Automatic Radar-based Gesture Detection and Classification via a Region-based Deep Convolutional Neural Network". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. ISSN: 2379-190X. May 2019, pp. 4300–4304. DOI: 10.1109/ICASSP.2019.8682277.

[277] Yuliang Sun, Tai Fei, Xibo Li, Alexander Warnecke, Ernst Warsitz, and Nils Pohl. "Multi-Feature Encoder for Radar-Based Gesture Recognition". In: *2020 IEEE International Radar Conference (RADAR)*. ISSN: 2640-7736. Apr. 2020, pp. 351–356. DOI: 10.1109/RADAR42522.2020.9114664.

[278] Yuliang Sun, Tai Fei, Xibo Li, Alexander Warnecke, Ernst Warsitz, and Nils Pohl. "Real-Time Radar-Based Gesture Detection and Recognition Built in an Edge-Computing Platform". In: *IEEE Sensors Journal* 20.18 (Sept. 2020), pp. 10706–10716. ISSN: 1558-1748. DOI: 10.1109/JSEN.2020.2994292.

[279] Yuliang Sun, Tai Fei, Frank Schliep, and Nils Pohl. "Gesture Classification with Handcrafted Micro-Doppler Features using a FMCW Radar". In: *2018 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*. Apr. 2018, pp. 1–4. DOI: 10.1109/ICMIM.2018.8443507.

[280] Adrian Tang, Robert Carey, Gabriel Virbila, Yan Zhang, Rulin Huang, and Mau-Chung Frank Chang. "A Delay-Correlating Direct-Sequence Spread-Spectrum (DS/SS) Radar System-on-Chip Operating at 183–205 GHz in 28 nm CMOS". In: *IEEE Transactions on Terahertz Science and Technology* 10.2 (2020), pp. 212–220. DOI: 10.1109/TTHZ.2020.2969105.

[281] Jingren Tang, Hong Cheng, Yang Zhao, and Hongliang Guo. "Structured dynamic time warping for continuous hand trajectory gesture recognition". In: *Pattern Recognition* 80 (2018), pp. 21–31. ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2018.02.011. URL: https://www.sciencedirect.com/science/article/pii/S0031320318300621.

[282] Eugene M. Taranta, Mehran Maghoumi, Corey R. Pittman, and Joseph J. LaViola. "A Rapid Prototyping Approach to Synthetic Data Generation for Improved 2D Gesture Recognition". In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. UIST '16. New York, NY, USA: Association for Computing Machinery, Oct. 2016, pp. 873–885. ISBN: 978-1-4503-4189-9. DOI: 10.1145/2984511.2984525. URL: https://doi.org/10.1145/2984511.2984525 (visited on 12/23/2020).

[283] Eugene M. Taranta II, Corey R. Pittman, Mehran Maghoumi, Mykola Maslych, Yasmine M. Moolenaar, and Joseph J. Laviola Jr. "Machete: Easy, Efficient, and Precise Continuous Custom Gesture Segmentation". In: *ACM Trans. Comput.-Hum. Interact.* 28.1 (Jan. 2021). ISSN: 1073-0516. DOI: 10.1145/3428068. URL: https://doi.org/10.1145/3428068.

[284] Eugene M. Taranta II, Amirreza Samiei, Mehran Maghoumi, Pooya Khaloo, Corey R. Pittman, and Joseph J. LaViola Jr. "Jackknife: A Reliable Recognizer with Few Samples and Many Modalities". In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA). CHI '17. New York, NY, USA: ACM, 2017, pp. 5850–5861. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3026002. URL: http://doi.acm.org/10.1145/3025453.3026002.

[285] Pakpoom Thaweesitthichat, Thuchchai Jiamsorn, Proadpran Punyabukkana, Ekapol Chuangsuwanich, and Atiwong Suchato. "Thai Sign Language Recognition with Leap Motion". In: *Proceedings of the 12th International Convention on Rehabilitation Engineering and Assistive Technology* (Shanghai, China). i-CREATe 2018. Midview City, SGP: Singapore Therapeutic, Assistive & Rehabilitative Technologies (START) Centre, 2018, pp. 47–50.

[286] Michail Theofanidis, Saif Iftekar Sayed, Alexandros Lioulemes, and Fillia Makedon. "VARM: Using Virtual Reality to Program Robotic Manipulators". In: *Proceedings of the 10th International Conference on PErvasive Technologies Related to Assistive Environments* (Island of Rhodes, Greece). PETRA '17. New York, NY, USA: Association for Computing Machinery, 2017, pp. 215–221. ISBN: 9781450352277. DOI: 10.1145/

3056540.3056541. URL: https://doi.org/10.1145/3056540.3056541.

[287] Enkhtogtokh Togootogtokh, Timothy K. Shih, W. G. Kumara, Shih-Jung Wu, Shih-Wei Sun, and Hon-Hang Chang. "3D Finger Tracking and Recognition Image Processing for Real-Time Music Playing with Depth Sensors". In: *Multimedia Tools Appl.* 77.8 (Apr. 2018), pp. 9233–9248. ISSN: 1380-7501. DOI: 10.1007/s11042-017-4784-9. URL: https://doi.org/10.1007/s11042-017-4784-9.

[288] Anh Phuong Tran, Frederic André, Christophe Craeye, and Sébastien Lambot. "Near-field or far-field full-wave ground penetrating radar modeling as a function of the antenna height above a planar layered medium". In: *Progress in Electromagnetics Research-Pier* 141 (2013), pp. 415–430. ISSN: 1559-8985. DOI: 10.2528/pier13053106.

[289] Van Thanh Tran, Jaewoon Lee, Dongho Kim, and Young-Sik Jeong. "Easy-to-Use Virtual Brick Manipulation Techniques Using Hand Gestures". In: *J. Supercomput.* 72.7 (July 2016), pp. 2752–2766. ISSN: 0920-8542. DOI: 10.1007/s11227-015-1588-4. URL: https://doi.org/10.1007/s11227-015-1588-4.

[290] Asaf Tzadok, Alberto Valdes-Garcia, Petar Pepeljugoski, J.-O. Plouchart, Mark Yeck, and Huijuan Liu. "AI-driven Event Recognition with a Real-Time 3D 60-GHz Radar System". In: *2020 IEEE/MTT-S International Microwave Symposium (IMS)*. ISSN: 2576-7216. Aug. 2020, pp. 795–798. DOI: 10.1109/IMS30576.2020.9224112.

[291] Harry Usuwiel, Jean Wilberz, Tancrède d'Aspremont Lynden, and Bastien Nothomb. "Opportunity study for the implementation of gesture recognition technology in the hospital environment - Commercialization through a university spin-off". Anglais. PhD thesis. UCL UCL UCL UCL - Louvain School of Management Louvain School of Management Faculté des sciences économiques, sociales, politiques et de communication Ecole polytechnique de Louvain, 2020. URL: http://hdl.handle.net/2078.1/thesis:24615.

[292] Jean Vanderdonckt, Paolo Roselli, and Jorge Luis Pérez-Medina. "!FTL, an Articulation-Invariant Stroke Gesture Recognizer with Controllable Position, Scale, and Rotation Invariances". In: *Proceedings of the ACM International Conference on Multimodal Interaction* (Boulder, CO, USA). Ed. by Sidney K. D'Mello, Panayiotis G. Georgiou, Stefan Scherer, Emily Mower Provost, Mohammad Soleymani, and Marcelo Worsley. ICMI '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 125–134. DOI: 10.1145/3242969.3243032. URL: https://doi.org/10.1145/3242969.3243032.

[293] Baptist Vandersmissen, Nicolas Knudde, Azarakhsh Jalalvand, Ivo Couckuyt, Tom Dhaene, and Wesley De Neve. "Indoor human activity recognition using high-dimensional sensors and deep neural networks". en. In: *Neural Computing and Applications* 32.16 (Aug. 2020), pp. 12295–12309. ISSN: 0941-0643, 1433-3058. DOI: 10.1007/s00521-019-04408-1. URL: http://link.springer.com/10.1007/s00521-019-04408-1 (visited on 12/21/2020).

[294] Radu-Daniel Vatavu. "IFAD Gestures: Understanding Users' Gesture Input Performance with Index-Finger Augmentation Devices". In: *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany). CHI '23. New York, NY, USA: Association for Computing Machinery, 2023. ISBN: 9781450394215. DOI: 10.1145/3544548.3580928. URL: https://doi.org/10.1145/3544548.3580928.

[295] Radu-Daniel Vatavu. "Improving Gesture Recognition Accuracy on Touch Screens for Users with Low Vision". In: *Proceedings of the ACM Conference on Human Factors in Computing Systems* (Denver, Colorado, USA). CHI '17. New York, NY, USA: ACM, 2017, pp. 4667–4679. ISBN: 978-1-4503-4655-9. DOI: 10.1145/3025453.3025941. URL: http://doi.acm.org/10.1145/3025453.3025941.

[296] Radu-Daniel Vatavu. "The Dissimilarity-Consensus Approach to Agreement Analysis in Gesture Elicitation Studies". In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk). CHI '19. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1–13. ISBN: 9781450359702. DOI: 10.1145/3290605.3300454. URL: https://doi.org/10.1145/3290605.3300454.

[297] Radu-Daniel Vatavu. "The impact of motion dimensionality and bit cardinality on the design of 3D gesture recognizers". In: *International Journal of Human-Computer Studies* 71.4 (2013), pp. 387–409. ISSN: 1071-5819. DOI: https://doi.org/10.1016/j.ijhcs.2012.11.005. URL: http://www.sciencedirect.com/science/article/pii/S1071581912002108.

[298] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O. Wobbrock. "$Q: A Super-Quick, Articulation-Invariant Stroke-Gesture Recognizer for Low-Resource Devices". In: *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Barcelona, Spain). MobileHCI '18. New York, NY, USA: Association for Computing Machinery, 2018. ISBN: 9781450358989. DOI: 10.1145/3229434.3229465. URL: https://doi.org/10.1145/3229434.3229465.

[299] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O. Wobbrock. "Gestures As Point Clouds: A $P Recognizer for User Interface Prototypes". In: *Proceedings of the 14th ACM International Conference on Multimodal Interaction* (Santa Monica, California, USA). ICMI '12. New York, NY, USA: ACM, 2012, pp. 273–280. ISBN: 978-1-4503-1467-1. DOI: 10.1145/2388676.2388732. URL: http://doi.acm.org/10.1145/2388676.2388732.

[300] Radu-Daniel Vatavu and Stefan Pentiuc. "Multi-Level Representation of Gesture as Command for Human Computer Interaction." In: *Computing and Informatics* 27 (Jan. 2008), pp. 837–851.

[301] Radu-Daniel Vatavu and Jacob O. Wobbrock. "Formalizing Agreement Analysis for Elicitation Studies: New Measures, Significance Test, and Toolkit". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea). Ed. by Bo Begole, Jinwoo Kim, Kori Inkpen, and Woontack Woo. CHI 2015. ACM, 2015, pp. 1325–1334. DOI: 10.1145/2702123.2702223. URL: https://doi.org/10.1145/2702123.2702223.

[302] Radu-Daniel Vatavu and Ionut-Alexandru Zaiti. "Leap Gestures for TV: Insights from an Elicitation Study". In: *Proceedings of the ACM International Conference on Interactive Experiences for TV and Online Video* (Newcastle Upon Tyne, United Kingdom). TVX '14. New York, NY, USA: Association for Computing Machinery, 2014, pp. 131–138. ISBN: 9781450328388. DOI: 10.1145/2602299.2602316. URL: https://doi.org/10.1145/2602299.2602316.

[303] Santiago Villarreal-Narvaez, Jean Vanderdonckt, Radu-Daniel Vatavu, and Jacob O. Wobbrock. "A Systematic Review of Gesture Elicitation Studies: What Can We Learn from 216 Studies?" In: *Proceedings of the 2020 ACM Designing Interactive Systems Conference* (Eindhoven, Netherlands). DIS '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 855–872. ISBN: 9781450369749. DOI: 10.1145/3357236.3395511. URL: https://doi.org/10.1145/3357236.3395511.

[304] Oleh Viunytskyi and Alexander Totsky. "Novel bispectrum-based wireless vision technique using disturbance of electromagnetic field by human gestures". In: *2017 Signal Processing Symposium (SPSympo)*. Sept. 2017, pp. 1–4. DOI: 10.1109/SPS.2017.8053684.

[305] Luc Vlaming, Jasper Smit, and Tobias Isenberg. "Presenting using two-handed interaction in open space". In: *2008 3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems*. 2008, pp. 29–32. DOI: 10.1109/TABLETOP.2008.4660180.

[306] Daniel Vogel and Ravin Balakrishnan. "Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays". In: *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology* (Seattle, WA, USA). UIST '05. New York, NY, USA: Association for Computing Machinery, 2005, pp. 33–42. ISBN: 1595932712. DOI: 10.1145/1095034.1095041. URL: https://doi.org/10.1145/1095034.1095041.

[307] Panagiotis Vogiatzidakis and Panayiotis Koutsabasis. "'Address and Command': Two-Handed Mid-Air Interactions with Multiple Home Devices". In: *Int. J. Hum.-Comput. Stud.* 159.C (Mar. 2022). ISSN: 1071-5819. DOI: 10.1016/j.ijhcs.2021.102755. URL: https://doi.org/10.1016/j.ijhcs.2021.102755.

[308] Panagiotis Vogiatzidakis and Panayiotis Koutsabasis. "Frame-Based Elicitation of Mid-Air Gestures for a Smart Home Device Ecosystem". In: *Informatics* 6.2 (2019). ISSN: 2227-9709. DOI: 10.3390/informatics6020023. URL: https://www.mdpi.com/2227-9709/6/2/23.

[309] Panagiotis Vogiatzidakis and Panayiotis Koutsabasis. "Mid-Air Gesture Control of Multiple Home Devices in Spatial Augmented Reality Prototype". In: *Multimodal Technologies and Interaction* 4.3 (2020). ISSN: 2414-4088. DOI: 10.3390/mti4030061. URL: https://www.mdpi.com/2414-4088/4/3/61.

[310] Christina Volioti, Sotiris Manitsaris, Edgar Hemery, Stelios Hadjidimitriou, Vasileios Charisis, Leontios Hadjileontiadis, Eleni Katsouli, Fabien Moutarde, and Athanasios Manitsaris. "A Natural User Interface for Gestural Expression and Emotional Elicitation to Access the Musical Intangible Cultural Heritage". In: *J. Comput. Cult. Herit.* 11.2 (Apr. 2018). ISSN: 1556-4673. DOI: 10.1145/3127324. URL: https://doi.org/10.1145/3127324.

[311] Robert Walter, Gilles Bailly, and Jörg Müller. "StrikeAPose: revealing mid-air gestures on public displays". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '13. Paris, France: Association for Computing Machinery, 2013, pp. 841–850. ISBN: 9781450318990. DOI: 10.1145/2470654.2470774. URL: https://doi.org/10.1145/2470654.2470774.

[312] Qian Wan, Yiran Li, Changzhi Li, and Ranadip Pal. "Gesture recognition for smart home applications using portable radar sensors". In: *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. ISSN: 1558-4615. Aug. 2014, pp. 6414–6417. DOI: 10.1109/EMBC.2014.6945096.

[313] Liying Wang, Zongjie Cao, Zongyong Cui, Changjie Cao, and Yiming Pi. "Negative Latency Recognition Method for Fine-Grained Gestures Based on Terahertz Radar". In: *IEEE Transactions on Geoscience and Remote Sensing* 58.11 (Nov. 2020), pp. 7955–7968. ISSN: 1558-0644. DOI: 10.1109/TGRS.2020.2985421.

[314] Liying Wang, Zongyong Cui, Zongjie Cao, Shengping Xu, and Rui Min. "Fine-Grained Gesture Recognition Based on High Resolution Range Profiles of Terahertz Radar". In: *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*. ISSN: 2153-7003. July 2019, pp. 1470–1473. DOI: 10.1109/IGARSS.2019.8900601.

[315] Pengcheng Wang, Junyang Lin, Fuyue Wang, Jianping Xiu, Yue Lin, Na Yan, and Hongtao Xu. "A Gesture Air-Writing Tracking Method that Uses 24 GHz SIMO Radar SoC". In: *IEEE Access* 8 (2020), pp. 152728–152741. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3017869.

[316] Saiwen Wang, Jie Song, Jaime Lien, Ivan Poupyrev, and Otmar Hilliges. "Interacting with Soli: Exploring Fine-Grained Dynamic Gesture Recognition in the Radio-Frequency Spectrum". In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan). UIST '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 851–860. ISBN: 9781450341899. DOI: 10.1145/2984511.2984565. URL: https://doi.org/10.1145/2984511.2984565.

[317] Yong Wang, Xiuqian Jia, Mu Zhou, Liangbo Xie, and Zengshan Tian. "A novel F-RCNN based hand gesture detection approach for FMCW systems". en. In: *Wireless Networks* (July 2019), pp. 1–14. ISSN: 1022-0038, 1572-8196. DOI: 10.1007/s11276-019-02096-2. URL: http://link.springer.com/10.1007/s11276-019-02096-2 (visited on 12/21/2020).

[318] Yong Wang, Aihu Ren, Mu Zhou, Wen Wang, and Xiaobo Yang. "A Novel Detection and Recognition Method for Continuous Hand Gesture Using FMCW Radar". In: *IEEE Access* 8 (2020), pp. 167264–167275. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3023187.

[319] Yong Wang, Shasha Wang, Mu Zhou, Qing Jiang, and Zengshan Tian. "TS-I3D Based Hand Gesture Recognition Method With Radar Sensor". In: *IEEE Access* 7 (2019), pp. 22902–22913. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2019.2897060.

[320] Yong Wang, Shasha Wang, Mu Zhou, Wei Nie, Xiaolong Yang, and Zengshan Tian. "Two-Stream Time Sequential Network Based Hand Gesture Recognition Method Using Radar Sensor". In: *2019 IEEE Globecom Workshops (GC Wkshps)*. Dec. 2019, pp. 1–6. DOI: 10.1109/GCWkshps45667.2019.9024691.

[321] Yong Wang, Zedong Zhao, Mu Zhou, and Jinjun Wu. "Two Dimensional Parameters Based Hand Gesture Recognition Algorithm for FMCW Radar Systems". en. In: *Wireless and Satellite Systems*. Ed. by Min Jia, Qing Guo, and Weixiao Meng. Vol. 280. Series Title: Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Cham: Springer International Publishing, 2019, pp. 226–234. ISBN: 978-3-030-19152-8 978-3-030-19153-5. DOI: 10.1007/978-3-030-19153-5_23. URL: http://link.springer.com/10.1007/978-3-030-19153-5_23 (visited on 12/21/2020).

[322] Zetao Wang, Gang Li, and Le Yang. "Dynamic Hand Gesture Recognition Based on Micro-Doppler Radar Signatures Using Hidden Gauss-Markov Models". In: *IEEE Geoscience and Remote Sensing Letters* 18.2 (2020), pp. 291–295. ISSN: 1558-0571. DOI: 10.1109/LGRS.2020.2974821.

[323] Zhu Wang, Xinye Lou, Zhiwen Yu, Bin Guo, and Xingshe Zhou. "Enabling non-invasive and real-time human-machine interactions based on wireless sensing and fog computing". en. In: *Personal and Ubiquitous Computing* 23.1 (Feb. 2019), pp. 29–41. ISSN: 1617-4909, 1617-4917. DOI: 10.1007/s00779-018-1185-7. URL: http://link.springer.com/10.1007/s00779-018-1185-7 (visited on 12/21/2020).

[324] Frank Weichert, Daniel Bachmann, Bartholomäus Rudak, and Denis Fisseler. "Analysis of the Accuracy and Robustness of the Leap Motion Controller". In: *Sensors* 13 (May 2013), pp. 6380–6393. DOI: 10.3390/s130506380. URL: https://www.mdpi.com/1424-8220/13/5/6380.

[325] Sean White, Levi Lister, and Steven Feiner. "Visual Hints for Tangible Gestures in Augmented Reality". In: *2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2007, pp. 47–50. DOI: 10.1109/ISMAR.2007.4538824.

[326] Daniel Wigdor and Dennis Wixon. *Brave NUI world: designing natural user interfaces for touch and gesture*. 1st ed. The Morgan Kaufmann Series in Computer Architecture and Design. Burlington, Massachusetts, United States: Morgan Kaufmann Publishers, 2011. ISBN: 978-0-12-382231-4. DOI: https://doi.org/10.1016/C2009-0-64091-5. URL: https://www.sciencedirect.com/book/9780123822314/brave-nui-world.

[327] Markus L. Wittorf and Mikkel R. Jakobsen. "Eliciting Mid-Air Gestures for Wall-Display Interaction". In: *Proceedings of the 9th Nordic Conference on Human-Computer Interaction* (Gothenburg, Sweden). NordiCHI

'16. New York, NY, USA: Association for Computing Machinery, 2016. ISBN: 9781450347631. DOI: 10.1145/2971485.2971503. URL: https://doi.org/10.1145/2971485.2971503.

[328] Jacob O. Wobbrock, Htet Htet Aung, Brandon Rothrock, and Brad A. Myers. "Maximizing the Guessability of Symbolic Input". In: *CHI '05 Extended Abstracts on Human Factors in Computing Systems* (Portland, OR, USA). CHI EA '05. New York, NY, USA: Association for Computing Machinery, 2005, pp. 1869–1872. ISBN: 1595930027. DOI: 10.1145/1056808.1057043. URL: https://doi.org/10.1145/1056808.1057043.

[329] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. "User-defined Gestures for Surface Computing". In: *Proceedings of the ACM Conference on Human Factors in Computing Systems* (Boston, MA, USA). CHI '09. New York, NY, USA: ACM, 2009, pp. 1083–1092. ISBN: 978-1-60558-246-7. DOI: 10.1145/1518701.1518866. URL: http://doi.acm.org/10.1145/1518701.1518866 (visited on 02/23/2018).

[330] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. "Gestures Without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes". In: *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (Newport, Rhode Island, USA). UIST '07. New York, NY, USA: ACM, 2007, pp. 159–168. ISBN: 978-1-59593-679-0. DOI: 10.1145/1294211.1294238. URL: http://doi.acm.org/10.1145/1294211.1294238.

[331] Christian Wolff. *Waves and Frequency Ranges*. Jan. 2022. URL: https://www.radartutorial.eu/07.waves/Waves%20and%20Frequency%20Ranges.en.html.

[332] Te-Yen Wu, Shutong Qi, Junchi Chen, MuJie Shang, Jun Gong, Teddy Seyed, and Xing-Dong Yang. "Fabriccio: Touchless Gestural Input on Interactive Fabrics". In: *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI '20. New York, NY, USA: Association for Computing Machinery, Apr. 2020, pp. 1–14. ISBN: 978-1-4503-6708-0. DOI: 10.1145/3313831.3376681. URL: https://doi.org/10.1145/3313831.3376681 (visited on 12/21/2020).

[333] Zhaoyang Xia, Yixiang Luomei, Chenglong Zhou, and Feng Xu. "Multidimensional Feature Representation and Learning for Robust Hand-Gesture Recognition on Commercial Millimeter-Wave Radar". In: *IEEE Transactions on Geoscience and Remote Sensing* 59.6 (2020), pp. 4749–4764. ISSN: 1558-0644. DOI: 10.1109/TGRS.2020.3010880.

[334] Xuhai Xu, Jun Gong, Carolina Brum, Lilian Liang, Bongsoo Suh, Shivam Kumar Gupta, Yash Agarwal, Laurence Lindsey, Runchang Kang, Behrooz Shahsavari, Tu Nguyen, Heriberto Nieto, Scott E Hudson, Charlie Maalouf, Jax Seyed Mousavi, and Gierad Laput. "Enabling Hand Gesture Customization on Wrist-Worn Devices". In: *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*. CHI '22. New Orleans, LA, USA: Association for Computing Machinery, 2022. ISBN: 9781450391573. DOI: 10.1145/3491102.3501904. URL: https://doi.org/10.1145/3491102.3501904.

[335] Li Yang and Gang Li. "Sparsity aware dynamic gesture recognition using radar sensors with angular diversity". In: *IET Radar, Sonar Navigation* 12.10 (2018), pp. 1114–1120. ISSN: 1751-8792. DOI: 10.1049/iet-rsn.2018.5206.

[336] Lin Yang, Long Ye, Wei Zhong, Yingying Zhang, and Qin Zhang. "A Real-Time Interactive System Based on Hand Gesture Recognition in Virtual Fitting". In: *LNCS on Transactions on Edutainment XIII - Volume 10092*. Berlin, Heidelberg: Springer-Verlag, 2017, pp. 86–96. ISBN: 9783662543948. DOI: 10.1007/978-3-662-54395-5_8. URL: https://doi.org/10.1007/978-3-662-54395-5_8.

[337] Yuan Yao, Po-Tsung Chiu, and Wai-Tat Fu. "A Gestural Interface for Practicing Children's Spatial Skills". In: *Proceedings of the 22nd International Conference on Intelligent User Interfaces Companion* (Limassol, Cyprus). IUI '17 Companion. New York, NY, USA: Association for Computing Machinery, 2017, pp. 43–47. ISBN: 9781450348935. DOI: 10.1145/3030024.3038265. URL: https://doi.org/10.1145/3030024.3038265.

[338] Hui-Shyong Yeo, Gergely Flamich, Patrick Schrempf, David Harris-Birtill, and Aaron Quigley. "RadarCat: Radar Categorization for Input & Interaction". In: *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan). UIST '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 833–841. ISBN: 9781450341899. DOI: 10.1145/2984511.2984515. URL: https://doi.org/10.1145/2984511.2984515.

[339] Hui-Shyong Yeo, Ryosuke Minami, Kirill Rodriguez, George Shaker, and Aaron Quigley. "Exploring Tangible Interactions with Radar Sensing". In: *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2.4 (Dec. 2018). DOI: 10.1145/3287078. URL: https://doi.org/10.1145/3287078.

[340] Hui-Shyong Yeo and Aaron Quigley. "Radar Sensing in Human-Computer Interaction". In: *Interactions* 25.1 (Dec. 2017), pp. 70–73. ISSN: 1072-5520. DOI: 10.1145/3159651. URL: https://doi.org/10.1145/3159651.

[341] Soojeong Yoo, Callum Parker, Judy Kay, and Martin Tomitsch. "To Dwell or Not to Dwell: An Evaluation of Mid-Air Gestures for Large Information Displays". In: *Proceedings of the Annual Meeting of the Australian Special Interest Group for Computer Human Interaction* (Parkville, VIC, Australia). OzCHI '15. New York, NY, USA: Association for Computing Machinery, 2015, pp. 187–191. ISBN: 9781450336734. DOI: 10.1145/2838739.2838819. URL: https://doi.org/10.1145/2838739.2838819.

[342] Ge Yu, Ji Liang, and Lili Guo. "Multi-Modal Interaction for Space Telescience of Fluid Experiments". In: *Proceedings of the 2018 International Conference on Artificial Intelligence and Virtual Reality* (Nagoya, Japan). AIVR 2018. New York, NY, USA: Association for Computing Machinery, 2018, pp. 35–41. ISBN: 9781450366410. DOI: 10.1145/3293663.3293672. URL: https://doi.org/10.1145/3293663.3293672.

[343] Jih-Tsun Yu, Li Yen, and Po-Hsuan Tseng. "mmWave Radar-based Hand Gesture Recognition using Range-Angle Image". In: *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*. ISSN: 2577-2465. May 2020, pp. 1–5. DOI: 10.1109/VTC2020-Spring48590.2020.9128573.

[344] Myoungseok Yu, Narae Kim, Yunho Jung, and Seongjoo Lee. "A Frame Detection Method for Real-Time Hand Gesture Recognition Systems Using CW-Radar". In: *Sensors* 20.8 (2020), pp. 1–17. ISSN: 1424-8220. DOI: 10.3390/s20082321. URL: https://www.mdpi.com/1424-8220/20/8/2321.

[345] Wei Zeng, Cong Wang, and Qinghui Wang. "Hand Gesture Recognition Using Leap Motion via Deterministic Learning". In: *Multimedia Tools Appl.* 77.21 (Nov. 2018), pp. 28185–28206. ISSN: 1380-7501.

[346] Zhengxin Zeng, Moeness G. Amin, and Tao Shan. "Arm Motion Classification Using Time-Series Analysis of the Spectrogram Frequency Envelopes". In: *Remote Sensing* 12.3 (2020), pp. 1–20. ISSN: 2072-4292. DOI: 10.3390/rs12030454. URL: https://www.mdpi.com/2072-4292/12/3/454.

[347] Zhengxin Zeng, Moeness G. Amin, and Tao Shan. "Automatic Arm Motion Recognition Based on Radar Micro-Doppler Signature Envelopes". In: *IEEE Sensors Journal* 20.22 (Nov. 2020), pp. 13523–13532. ISSN: 1558-1748. DOI: 10.1109/JSEN.2020.3004581.

[348] Bo Zhang, Lei Zhang, Mojun Wu, and Yan Wang. "Dynamic Gesture Recognition Based on RF Sensor and AE-LSTM Neural Network". In: *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*. 2021, pp. 1–5. DOI: 10.1109/ISCAS51556.2021.9401065.

[349] Guiyuan Zhang, Shengchang Lan, Kang Zhang, and Linting Ye. "Temporal-Range-Doppler Features Interpretation and Recognition of Hand Gestures Using mmW FMCW Radar Sensors". In: *2020 14th European Conference on Antennas and Propagation (EuCAP)*. Mar. 2020, pp. 1–4. DOI: 10.23919/EuCAP48036.2020.9135694.

[350] Guiyuan Zhang, Kang Zhang, Yihan Yun, Gang Lu, and Shengchang Lan. "Implementation of C4.5 decision tree in Human Gesture Recognition based on Doppler radars". In: *2019 International Symposium on Antennas and Propagation (ISAP)*. Oct. 2019, pp. 1–3.

[351] Jiajun Zhang and Zhiguo Shi. "Deformable deep convolutional generative adversarial network in microwave based hand gesture recognition system". In: *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*. ISSN: 2472-7628. Oct. 2017, pp. 1–6. DOI: 10.1109/WCSP.2017.8170976.

[352] Jiajun Zhang, Jinkun Tao, and Zhiguo Shi. "Doppler-Radar Based Hand Gesture Recognition System Using Convolutional Neural Networks". en. In: *Communications, Signal Processing, and Systems*. Ed. by Qilian Liang, Jiasong Mu, Min Jia, Wei Wang, Xuhong Feng, and Baoju Zhang. Vol. 463. Series Title: Lecture Notes in Electrical Engineering. Singapore: Springer Singapore, 2019, pp. 1096–1113. ISBN: 978-981-10-6570-5 978-981-10-6571-2. DOI: 10.1007/978-981-10-6571-2_132. URL: http://link.springer.com/10.1007/978-981-10-6571-2_132 (visited on 12/21/2020).

[353] Kaijie Zhang, Zhiwen Yu, Dong Zhang, Zhu Wang, and Bin Guo. "RaCon: A gesture recognition approach via Doppler radar for intelligent human-robot interaction". In: *2020 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. Mar. 2020, pp. 1–6. DOI: 10.1109/PerComWorkshops48775.2020.9156109.

[354] Shimeng Zhang, Gang Li, Matthew Ritchie, Francesco Fioranelli, and Hugh Griffiths. "Dynamic hand gesture classification based on radar micro-Doppler signatures". In: *2016 CIE International Conference on Radar (RADAR)*. Oct. 2016, pp. 1–4. DOI: 10.1109/RADAR.2016.8059518.

[355] Xuhao zhang, Qisong Wu, and Dixian Zhao. "Dynamic Hand Gesture Recognition Using FMCW Radar Sensor for Driving Assistance". In: *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*. ISSN: 2472-7628. Oct. 2018, pp. 1–6. DOI: 10.1109/WCSP.2018.8555642.

[356] Zhenyuan Zhang, Zengshan Tian, Zhou Mu, and Yi Liu. "Application of FMCW Radar for Dynamic Continuous Hand Gesture Recognition". In: *Proceedings of the 11th EAI International Conference on Mobile Multimedia Communications*. MOBIMEDIA'18. Brussels, BEL: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), Sept. 2018, pp. 298–303. ISBN: 978-1-63190-164-5. (Visited on 12/21/2020).

[357] Zhenyuan Zhang, Zengshan Tian, Ying Zhang, Mu Zhou, and Bang Wang. "u-DeepHand: FMCW Radar-Based Unsupervised Hand Gesture Feature Learning Using Deep Convolutional Auto-Encoder Network". In: *IEEE Sensors Journal* 19.16 (Aug. 2019), pp. 6811–6821. ISSN: 1558-1748. DOI: 10.1109/JSEN.2019.2910810.

[358] Zhenyuan Zhang, Zengshan Tian, and Mu Zhou. "Latern: Dynamic Continuous Hand Gesture Recognition Using FMCW Radar Sensor". In: *IEEE Sensors Journal* 18.8 (Apr. 2018), pp. 3278–3289. ISSN: 1558-1748. DOI: 10.1109/JSEN.2018.2808688.

[359] Zhenyuan Zhang, Zengshan Tian, and Mu Zhou. "SmartFinger: A Finger-Sensing System for Mobile Interaction via MIMO FMCW Radar". In: *2019 IEEE Globecom Workshops (GC Wkshps)*. Dec. 2019, pp. 1–5. DOI: 10.1109/GCWkshps45667.2019.9024578.

[360] Zhenyuan Zhang, Zengshan Tian, Mu Zhou, Wei Nie, and Ze Li. "Riddle: Real-Time Interacting with Hand Description via Millimeter-Wave Sensor". In: *2018 IEEE International Conference on Communications (ICC)*. ISSN: 1938-1883. May 2018, pp. 1–6. DOI: 10.1109/ICC.2018.8422765.

[361] Zedong Zhao, Yong Wang, Mu Zhou, Xiaolong Yang, and Liangbo Xie. "Interference Suppression Based Gesture Recognition Method with FMCW Radar". In: *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*. ISSN: 2472-7628. Oct. 2019, pp. 1–6. DOI: 10.1109/WCSP.2019.8928108.

[362] Andrew Jie Zhou and Grace Hui Yang. "Minority Report by Lemur: Supporting Search Engine with Virtual Reality". In: *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval* (Ann Arbor, MI, USA). SIGIR '18. New York, NY, USA: Association for Computing Machinery, 2018, pp. 1329–1332. ISBN:

9781450356572. DOI: 10.1145/3209978.3210179. URL: https://doi.org/10.1145/3209978.3210179.

[363] Feifei Zhou, Xiangyu Li, and Zhihua Wang. "Efficiently User-Independent Ultrasonic-Based Gesture Recognition Algorithm". In: *2019 IEEE SENSORS*. 2019, pp. 1–4. DOI: 10.1109/SENSORS43011.2019.8956774.

[364] Zhi Zhou, Zongjie Cao, and Yiming Pi. "Dynamic Gesture Recognition with a Terahertz Radar Based on Range Profile Sequences and Doppler Signatures". In: *Sensors* 18.1 (2018), pp. 1–15. ISSN: 1424-8220. DOI: 10.3390/s18010010. URL: https://www.mdpi.com/1424-8220/18/1/10.

[365] Shangyue Zhu, Junhong Xu, Hanqing Guo, Qiwei Liu, Shaoen Wu, and Honggang Wang. "Indoor Human Activity Recognition Based on Ambient Radar with Signal Processing and Machine Learning". In: *2018 IEEE International Conference on Communications (ICC)*. 2018, pp. 1–6. DOI: 10.1109/ICC.2018.8422107.

[366] Jamie Zigelbaum, Alan Browning, Daniel Leithinger, Olivier Bau, and Hiroshi Ishii. "G-Stalt: A Chirocentric, Spatiotemporal, and Telekinetic Gestural Interface". In: *Proceedings of the Fourth International Conference on Tangible, Embedded, and Embodied Interaction* (Cambridge, Massachusetts, USA). TEI '10. New York, NY, USA: Association for Computing Machinery, 2010, pp. 261–264. ISBN: 9781605588414. DOI: 10.1145/1709886.1709939. URL: https://doi.org/10.1145/1709886.1709939.

[367] Alessandro Zocco, Matteo D. Zocco, Antonella Greco, Salvatore Livatino, and Lucio Tommaso Paolis. "Touchless Interaction for Command and Control in Military Operations". In: *Proceedings of the Second International Conference on Augmented and Virtual Reality - Volume 9254*. Berlin, Heidelberg: Springer-Verlag, 2015, pp. 432–445. ISBN: 9783319228877. DOI: 10.1007/978-3-319-22888-4_32. URL: https://doi.org/10.1007/978-3-319-22888-4_32.

[368] Yi-Bo Zou, Yi-Min Chen, Ming-Ke Gao, Quan Liu, Si-Yu Jiang, Jia-Hui Lu, Chen Huang, Ze-Yu Li, and Dian-Hua Zhang. "Coronary Heart Disease Preoperative Gesture Interactive Diagnostic System Based on Augmented Reality". In: *J. Med. Syst.* 41.8 (Aug. 2017), pp. 1–18. ISSN: 0148-5598. DOI: 10.1007/s10916-017-0768-6. URL: https://doi.org/10.1007/s10916-017-0768-6.

Appendix A

QuantumLeap Modules

This appendix provides a short description of the modules provided with the QuantumLeap framework. If necessary, developers may write new modules to better fit their needs.

A.1 Sensors

A.1.1 Leap Motion Controller

A module that connects to the LMC and fetches frames using its JavaScript API. It returns the position of a set of hand joints at one instant in time.

A.2 Filters

A.2.1 1€

A simple and efficient low-pass filter inspired by the $-family of recognizers [38]. Designed for interactive systems, it relies on an adaptive cutoff frequency that allows it to greatly reduce jitter at low speeds while limiting lag at high speeds.

A.3 Static and Dynamic Datasets

A.3.1 QLDynamic

A dataset composed of 17 dynamic gestures performed by four participants. Each participant performed each gesture four times above the LMC. This dataset was used in the LUI application (Chapter 5), as well as the DICOM image viewer and Wall'ON application (Section 3.3).

A.3.2 QLStatic

A dataset composed of 5 static poses performed by only one participant above the LMC. It was also used in the LUI and Wall'ON applications.

A.4 Static Recognizers

QuantumLeap comes with two modules for static gesture recognition. Table A.1 summarizes their invariance properties.

Recognizer	Type of invariance			
	Position	Scale	Rotation	Direction
$P^3+	●	●	○	●
GPSDa	●	●	●	●

TABLE A.1: Invariance properties of the static recognizers.

A.4.1 $P^3+

A 3D version of *$P+* algorithm [295]. It is position-, direction-, and scale-invariant [145], but not rotation-invariant. It is described in more detail in Appendix A.7.

A.4.2 GPSDa

A novel position-, scale-, direction-, and rotation-invariant recognizer for hand poses regardless of the position of the hand. Its rotation-invariance makes it well-suited to situations where hand poses should be recognized regardless of their direction (*e.g.*, rotate a picture by performing a "grab" static gesture and gradually rotating the hand).

A.5 Analyzers

A.5.1 Basic Analyzer

This module extracts four values from each frame: (1) the rotation of the hand compared to the previous frame; (2) the ratio of the distances between the index and the thumb of the current and the previous frames; (3) the displacement of the palm compared to the previous frame; and (4) a 3D vector that gives the orientation of the thumb.

A.6 Segmenters

Three simple segmentation methods have been implemented to demonstrate the possibilities of QuantumLeap. We are planning on implementing more advanced segmentation techniques such as Taranta *et al.*'s Machete [283] and Chen *et al.*'s Pactolus [40]. However, support for Pactolus will only be relevant once EMG sensors are supported by QuantumLeap.

A.6.1 Zoning Segmenter

A segmenter that triggers gesture recognition when one or more selected points are within a bounding box above the sensor. After all the selected points have left, the recorded frames are sent to the recognizer.

A.6.2 Threshold Segmenter

A sensor that triggers gesture recognition when one or more parameters meet a pre-defined condition. The conditions (*e.g.*, the value of a parameter exceeds/falls below some threshold) can be manually configured from the UI. Once the conditions are no longer met, the recorded frames are sent to the recognizer.

A.6.3 Window Segmenter

A segmenter that triggers gesture recognition at regular intervals by relying on fixed-size buffers called *windows*. To adapt to varying gesture duration, it can be configured with any number of windows. Each time a new frame is received, it is pushed in the window(s), while the oldest frame is removed. For every fixed number of frames, the buffered gesture data are sent to the recognizer.

A.7 Dynamic Recognizers

QuantumLeap comes with ten modules for dynamic gesture recognition. Table A.2 compares their invariance properties.

A.7.1 P^3

A generalization of *$P* [299] towards supporting 3D multi-stroke gestures which is similar to the $P3D recognizer implemented by [50]. However, unlike

Recognizer	Type of invariance			
	Position	Scale	Rotation	Direction
$P^3	●	●	○	●
$Q^3	●	●	○	●
$P^3+	●	●	○	●
$P^3+X	●	●	○	◐
3 cent	●	●	○	○
Jackknife	●	●	○	○
$F	●	●	○	●
FreeHandUni	●	●	○	●
Rubine3D	●	●	○	○
Rubine-Sheng	●	●	○	○

TABLE A.2: Invariance properties of the dynamic recognizers.

$P3D the recognizer $P^3 does not support 3D static poses and 2D dynamic gestures recognition. To keep memory usage and execution time low, gestures are represented as unordered sets of points, called *point clouds*. Gesture recognition happens in two phases: normalization and cloud-matching. The normalization process is similar to other $-family algorithms and is divided into three steps: (1) resample the gesture to a fixed number of equidistant points; (2) scale the gesture uniformly to keep its shape; (3) translate the centroid of the point cloud to the origin, *i.e.*, $(0, 0, 0)$. The cloud-matching phase matches a gesture's point cloud to the point cloud of each template by associating each point from the template to exactly one point from the gesture. It then computes the resulting distance, as the sum of Euclidean distances between all pairs of matching points, and returns the template that minimizes it. This recognizer is position-, direction-, and scale-invariant, but not rotation-invariant.

A.7.2 $Q^3

A 3D variant of *$Q* [298], which achieved a 46X speedup on average over *$P* with no loss of accuracy. It brings two key changes to *$P*^{3}: (1) early abandonment of templates, as soon as the distance exceeds the current shortest distance; (2) each point cloud has a $16\times16\times16$ 3D look-up table (LUT), where each location (x, y, z) refers to the closest point. These LUTs allow *$Q*^{3} to compute a lower bound of the distance between a gesture and a template in $\mathcal{O}(n)$. If it exceeds the current shortest distance, the template can be rejected without computing the exact distance. It has the same invariance properties as *$P*^{3}.

A.7.3 P^3+

A more accurate version of *P^3*, adapted from Vatavu's *P+* [295]. It brings three key improvements to *P^3*: (1) each point from the template can now be matched to more than one point of the gesture; (2) the distance between a gesture and a template takes into account the connections between consecutive points (in the form of their turning angles); (3) early abandonment of templates. This recognizer has the same invariance properties as *P^3*.

A.7.4 P^3+X

A variant of *P^3*+, which supports partial direction-invariance by keeping track of conflicting templates (*i.e.*, templates that represent the same gesture but drawn in different directions). If a gesture matches a conflicting template, its direction is compared with the direction of each conflicting template and the closest one is returned.

A.7.5 3 cent

An optimized 3D port of Wobbrock *et al.*'s *$1* [330] by Caputo *et al.* [36], which recognizes uni-stroke gestures in two phases: the gesture is first normalized (similarly to *$P*) and its distance to each template is then computed as the sum of the squared distances of corresponding points. The template with the shortest distance is returned. *3 cent* is position- and scale-invariant, but neither rotation- nor direction-invariant.

A.7.6 Jackknife

A general-purpose 3D gesture recognizer [284] that supports any number of joints. Unlike most $-family recognizers, it represents gestures as time series. It uses the nearest-neighbor approach, where it compares a gesture to each template and returns the closest one. It uses *Dynamic Time Warping* as a distance measure but applies a series of correction factors to take into account differences in gesture scale and span. As a result, this recognizer is both position- and scale-invariant, but neither direction- nor rotation-invariant.

A.7.7 $F

A new recognizer that adds *$P*+'s flexible cloud matching [295] to *P^3*. As for most $-recognizers, the points of the candidate and the templates are resampled to equidistantly-spaced points, scaled within a unit box (isometric [292]), and translated so their centroid is at the origin $(0, 0, 0)$. The template with

the lowest dissimilarity score is considered the best matching template for the candidate. As opposed to *P^3*, *$F*'s cloud matching is flexible, as points can be matched to more than one point. It consists of matching the points from the first cloud with their closest point from the second cloud, then matching the points from the second cloud that have not been matched yet with their closest point in the first cloud.

A.7.8 FreeHandUni

A recognizer derived from Vatavu *et al.*'s Free-Hand recognizer [56], which extends *$P* [299] to support 3D hand gestures. It replaces the hand pose structure with a 3D point structure (x, y, z). With this modification, *FreehandUni* improves *P^3* using a flexible cloud matching based on a one-to-many alignment between points [295]. The pre-processing stays the same but the matching process is more flexible: each point of the template cloud is matched with the closest point from the candidate cloud, then the remaining points from the candidate cloud are matched with the closest point from the template cloud. It returns the gesture class with the lowest dissimilarity score. *FreeHandUni* is different from *$F* in that the early abandoning is not implemented, to align the computational complexity to *P^3*.

A.7.9 Rubine3D

Inspired by the iGesture framework [259], Rubine3D is a feature-based recognizer that combines a set of three individual 2D Rubine recognizers [240], one for each plane XY, YZ, and ZX. Before it can recognize 3D trajectories, Rubine3D pre-processes the training templates to compute weights for each feature of each gesture class. Then, it can recognize gestures in four steps: (1) it pre-processes the gesture by scaling and filtering its points; (2) it projects it onto each plane and extracts a feature vector $(f_1, .., f_{13})$ from each of the three projections; (3) for each projection, it selects the gesture class that maximizes the similarity score (computed by combining the weights of the gesture class with the feature vector of the gesture); (4) it determines the resulting class by merging the results from the three projections.

A.7.10 Rubine-Sheng

Rubine-Sheng (*RS*) is a 3D recognizer inspired by Rubine. It supports 3D gestures by adding three new features proposed in the AdaBoost recognizer [254] to Rubine's existing 13 features. Aside from this difference, it is extremely similar to its 2D counterpart.

Note. The implemented multipath recognizers include an optimization that avoids comparing the candidate gesture performed with one hand against the gestures performed with both hands in the training set.

Appendix B

Tasks

B.1 Configure QuantumLeap

B.1.1 Initial State

The QuantumLeap framework is running in the background and its UI is opened on the "Overview" page.

B.1.2 Steps

1. Go to the "Sensor(s)" page and select the "Leap Motion Controller" module. Rename it as "lmc" and save the modifications.
2. Go to the "Static dataset(s)" page and select the "Leap Motion Dataset Loader" module. Select the "Simple dataset (static)" dataset and give it the "lmc" identifier. Select 100 templates per class. Save the modifications.
3. Go to the "Static recognizer" page and select the "GPSDa" module. Select all the points of the right hand. Save the modifications.
4. Go to the "Segmenter" page and select the "Sliding window" module. Configure it with one (1) window with a length of 20 frames. Select the right hand palm. Save the modifications.
5. Go to the "Dynamic dataset(s)" page and select the "Leap Motion Dataset Loader" module. Select the "Simple dataset (dynamic)" dataset and give it the "lmc" identifier. Select 8 templates per class. Save the modifications.
6. Go to the "Dynamic recognizer" page and select the "Jackknife" module. Configure it with the right hand palm. Save the modifications.
7. Restart QuantumLeap by toggling the play/pause button in the "Overview" page.

B.2 Add Gesture Recognition to a Small Application

Initial State

Visual Studio Code is opened on the App.js file. Only this file should be modified by the participant. The application is running in development mode in a Google Chrome tab. The application refreshes automatically when the modifications to App.js are saved. The QuantumLeap framework is running in the background.

B.2.1 Steps

1. To be able use the QuantumLeap API, first import `GestureHandler` from the quantumleapjs module.
2. Instantiate a new `GestureHandler` object without any option in the constructor of the `App` class. Assign it to a new instance variable of `App` (*e.g.*, `this.gestureHandler`). In the next steps, you will use the methods of `GestureHandler` to implement gesture recognition.
3. In `componentDidMount`, add a call to the `connect` method of `GestureHandler` in order to connect to the QuantumLeap framework after the page has loaded.
4. Make sure to disconnect from the QuantumLeap framework when the page is closed by adding a call to the `disconnect` method of `GestureHandler` in `componentWillUnmount`.
5. Let's add the first gesture. The application will display the "swipe left" image each time a "swipe left" gesture is recognized. In `componentDidMount`, add an event listener for the "gesture" event that calls `this.onLeftSwipe` each time "swipe left" is recognized. You should use the `addEventListener` method of `GestureHandler`.
6. You may notice that nothing happens when performing a "swipe left" gesture. To fix this, register the gesture to QuantumLeap by adding a call to the `registerGestures` method of `GestureHandler` with the type ("dynamic") and name ("swipe left") of the gesture. This will notify QuantumLeap that you want it to recognize the "swipe left" gesture.

7. Now, modify the listener that you added at step 5 to display the "swipe right" image (by calling `this.onRightSwipe`) each time the corresponding gesture is performed. Add the "swipe right" gesture to the call to `registerGestures`.

8. Modify the listener again to display the "point index" image (by calling `this.onPoint`) every time the user points his index.

9. The "point index" gesture is static. You should thus add a new call to the `registerGestures` method of `GestureHandler` with the type ("static") and name ("point index") of the gesture.

10. Now, modify the listener to display the "thumb" image (by calling `this.onThumb`) each time the corresponding gesture is performed. Add the "thumb" gesture to the call to `registerGestures` with the "point index" gesture.

11. Modify the listener one last time to display the name and type of each gesture when they are recognized. You can do it by adding a call to `this.onGesture` (with the type and name of gesture as arguments) in the listener.

12. Let's now add a text at the bottom of the screen to show if the application is connected to QuantumLeap. In `componentDidMount`, use the `addEventListener` method of `GestureHandler` to call `this.setConnected` with "true" every time a "connect" event is emitted.

13. To reset the text when the application is disconnected from QuantumLeap, use the `addEventListener` method of `GestureHandler` to call `this.setConnected` with "false" every time a "disconnect" event is emitted.

Appendix C

LUI Gesture Elicitation Studies

C.1 Photo- and Video-Browsing Gesture Elicitation Studies

Two experiments were conducted to determine users' preferred gestures for browsing photos and videos. 30 participants (16 female) aged between 18 and 77 years old (M=34.5, SD=17.5) took part in the photo-browsing experiment while 22 participants (8 females) aged between 15 and 54 years old (M=28.9, SD=12.6) took part in the video-browsing experiment. Agreement rates ranged from .064 to .690 (M=.205, SD=.147) in the photo-browsing experiment (Table C.1) and from .052 to .593 (M=.229, SD=.162) in the video-browsing experiment (Table C.2).

C.2 Multimedia Gesture Elicitation Study

A third GES was conducted to elicit mid-air gestures for multimedia interaction (Section 5.2.1). Section C.2.1 provides details about the experimental protocol and Section C.2.2 presents its results.

C.2.1 Experimental Protocol

Participants

23 participants (6 females), aged from 18 to 56 years old (M=27.1, SD=10.2 years), volunteered for the study. Their occupations include students and employees, mostly in IT, art, and engineering. All but two of the participants never used an LMC, but all of them regularly used a computer. Only one

participant did not frequently use a smartphone. 95.7% of the participants (22/23) were right-handed.

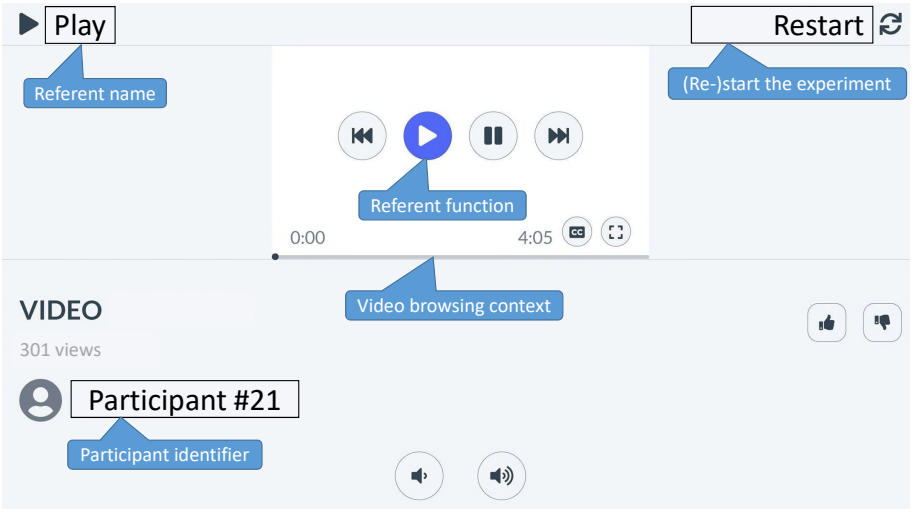


FIGURE C.1: Screenshot of the gesture elicitation software.

Apparatus

The experiment involved three devices: the LMC to capture participants' mid-air gesture proposals, a smartphone to record their touch gesture proposals, and a large screen to display the referents. A laptop controlled the display and recorded the LMC gestures using custom software suited for gesture elicitation C.1.

Design

The experiment manipulates one main independent variable: REFERENT, a within-subject nominal variable with 18 conditions. Each condition represents a common task to perform in multimedia environments, such as in a music player, a video gallery, or a PowerPoint presentation: (1) previous, (2) next, (3) enable fullscreen, (4) disable fullscreen, (5) zoom in, (6) zoom out, (7) volume up, (8) volume down, (9) rotate 90°clockwise, (10) rotate 90°counter-clockwise, (11) play, (12) pause, (13) like, (14) dislike, (15) fast forward 5 seconds, (16) rewind 5 seconds, (17) enable subtitles, and (18) disable subtitles. The number of conditions has been purposefully kept small by combining similar referents

for different contexts together, *e.g.*, there is no distinction between going to the next picture and going to the next video.

The following measures were employed to evaluate and understand users' preferences for gestures captured by the LMC [88]:

1. The agreement rate $AR(r)$ was computed for each REFERENT.
2. The THINKING-TIME measures the time, in seconds, needed by participants to propose a gesture for a given referent.
3. The GOODNESS-OF-FIT represents participants' subjective assessment, as a rating between 1 and 5, of their confidence about how well a gesture fits a referent.

Task

Participants were installed at a desk and faced a large screen situated a few meters on the opposite side of the desk. The LMC was positioned in front of them, and a smartphone was placed on their right. Each participant was presented with a series of referents in a randomized order. For each referent, participants were asked to propose a mid-air gesture that fits the referent, in an order determined randomly. Participants' thinking time was measured for each gesture proposal. No constraint was imposed on the participants' choices. After each gesture proposal, the participants estimated its goodness-of-fit on a Likert scale from 1 (very unsatisfied) to 5 (very satisfied).

C.2.2 Results

414 mid-air gesture proposals were collected from 23 (participants) $\times$ 18 (referents) conditions. These gesture proposals were then clustered into 112 groups of similar gestures. For this clustering, we distinguished between gestures according to their direction (*i.e.*, similar gestures performed in different directions are considered different) but grouped similar gestures performed with different poses together (*e.g.*, swiping with a flat hand is considered the same as swiping while pointing the index finger). Table C.3 summarizes the most agreed-upon gestures for each referent.

Consensus

Overall, agreement rates are quite high, between .028 and .83 (M=.366, SD=.268). These results fit within the reported agreement rates in the literature of gesture elicitation, see [301] (p. 1332) that summarize agreement rates of 18 studies. According to the recommendations of [301] to interpret the magnitudes of agreement rates, most of our results fall inside the high consensus (.300−.500) category. These results were confirmed by Kendall's coefficient of concordance

for estimating the inter-rater reliability: W=.564 ($\chi^2(26)$=220.454, p<.0001, large effect).

Gestures' Goodness of Fit

Participants rated their gesture proposals with numbers from 1 (poor fit) to 5 (excellent fit) to denote their confidence in the goodness of fit of their proposals. Overall, participants were satisfied with their gesture propositions as shown by the high average goodness of fit (M=4.21, SD=0.88). GOODNESS-OF-FIT correlated significantly with the AGREEMENT-RATE (Pearson's $r_{(N=18)}$=.711, R^2=.505, p=.00094): referents that reached high agreement rates were assigned gestures that were rated a good fit (Fig. C.2).

Thinking Time

The average thinking time was usually quite low (M=6.15 s, SD=9.12 s). THINKING-TIME correlated significantly with AGREEMENT-RATE (Pearson's $r_{(N=18)}$=−.620, R^2=.384, p=.006) (Fig. C.3): the agreement rate decreased for longer thinking times. A few referents showed considerably higher than average thinking times: disable subtitles, enable subtitles, and skip backward 5 seconds. These referents represent tasks that participants are less likely to perform regularly on their devices.

Referent	AR	1st most agreed gesture	2nd most agreed gesture
1. *Pan photo gallery right*	.251 (M)	1 hand flick	Multiple fingers swipe
2. *Pan photo gallery left*	.269 (M)	Multiple fingers swipe	1 finger swipe
3. *Pan photo gallery up*	.170 (M)	1 finger swipe	Hand rotation
4. *Pan photo gallery down*	.175 (M)	1 hand flick	1 finger swipe
5. *Go to next page of photo gallery*	.205 (M)	Multiple fingers swipe	1 hand flick
6. *Go to previous page of photo gallery*	.168 (M)	Multiple fingers swipe	1 hand flick
7. *Zoom in a photo*	.338 (H)	Splay the hand	Move hands apart
8. *Zoom out a photo*	.299 (M)	Clench the hand	Move hands apart
9. *Maximize a photo from normal view to full screen*	.363 (H)	Move hands apart	Splay the hand
10. *Unmaximize a photo from full screen to normal view*	.193 (M)	Clench the hand	Move hands closer together
11. *Open photo information*	.308 (H)	1 finger tap	Splay the hand
12. *Close photo information*	.170 (M)	1 finger tap	1 finger swipe
13. *Set layout to carousel*	.099 (L)	Hand rotation	1 finger tap
14. *Start slideshow*	.126 (M)	1 finger tap	Hand pose
15. *Stop slideshow*	.156 (M)	1 finger tap	Move flat hand forward/backward
16. *Take a selfie*	.099 (L)	Hand pose	1 finger tap
17. *Insert a new photo*	.092 (L)	Hand pose	1 finger tap
18. *Duplicate a photo*	.097 (L)	1 finger swipe	1 finger tap
19. *Delete a photo*	.094 (L)	1 finger swipe	Hand pose
20. *Crop a photo*	.090 (L)	Draw	Move hands apart
21. *Resize a photo*	.333 (H)	Move hands apart	Splay the hand
22. *Rotate a photo 90°CW*	.634 (V)	Hand rotation	N/A
23. *Rotate a photo 90°CCW*	.690 (V)	Hand rotation	N/A
24. *Increase constrast*	.071 (L)	1 finger swipe	Open arms (with flat hands)
25. *Decrease contrast*	.064 (L)	1 finger swipe	Multiple fingers swipe
26. *Dock a photo*	.097 (L)	1 finger swipe	1 hand flick
27. *Undock a photo*	.152 (M)	1 finger swipe	Take and throw
28. *Like a photo*	.218 (M)	1 finger tap	Hand pose
29. *Search by criteria*	.131 (M)	Hand pose	Draw
30. *Share a photo with somebody (vocal email)*	.087 (L)	Hand pose	1 finger swipe
31. *Convert a photo into another format*	.115 (M)	Move hands apart	1 finger tap

TABLE C.1: First and second most agreed gesture proposals for each referent (photo-browsing experiment). Referents with above-average agreement rates are highlighted in blue. Notation for the Agreement Rate (AR): L=low, M=medium, H=high, V=very high [301].

Referent	AR	1st most agreed gesture	2nd most agreed gesture
1. *Pan video gallery right*	.433 (H)	1 hand drag	1 finger drag
2. *Pan video gallery left*	.433 (H)	1 hand drag	1 finger drag
3. *Pan video gallery up*	.390 (H)	1 hand drag	1 finger drag
4. *Pan video gallery down*	.390 (H)	1 hand drag	1 finger drag
5. *Go to next page of video gallery*	.407 (H)	1 hand drag	1 finger drag
6. *Go to previous page of video gallery*	.329 (H)	1 hand drag	1 finger drag
7. *Zoom in a video*	.182 (M)	1 hand drag	Close one hand
8. *Zoom out a video*	.242 (M)	1 hand drag	Close one hand
9. *Set a video in full screen*	.165 (M)	1 finger tap	Drag 1 finger from each hand
10. *Unmaximize a video from full screen to normal view*	.095 (L)	Drag 1 finger from each hand	2 hands drag
11. *Play a video*	.333 (H)	1 finger tap	1 finger drag
12. *Stop playing a video*	.294 (M)	Push/pull 1 hand	1 finger tap
13. *Move to the beginning of video*	.169 (M)	1 hand drag	1 finger tap
14. *Move to the end of a video*	.169 (M)	1 hand drag	1 finger tap
15. *Open video information*	.117 (M)	1 finger tap	Push/pull 1 hand
16. *Close video information*	.117 (M)	1 hand drag	1 finger drag
17. *Set layout to carousel*	.117 (M)	1 hand drag	Tap and drag 1 finger
18. *Insert a new video*	.078 (L)	Push/pull and drag 1 hand	Push/pull 1 hand
19. *Duplicate a video*	.052 (L)	2 hands drag	Tap and drag 1 finger from each hand
20. *Delete a video*	.074 (L)	1 hand drag	1 finger drag
21. *Rotate a video 90°CW*	.593 (V)	1 hand drag	Push/pull 1 hand
22. *Rotate a video 90°CCW*	.593 (V)	1 hand drag	Push/pull 1 hand
23. *Dock a video*	.078 (L)	Push/pull and drag 1 hand	Tap and drag 1 finger
24. *Undock a video*	.078 (L)	Push/pull and drag 1 hand	Close, push/pull and drag 1 hand
25. *Like a video*	.091 (L)	1 finger drag	Push/pull 1 hand
26. *Search for a video by criteria*	.082 (L)	1 finger drag	1 hand drag
27. *Share a video with somebody*	.082 (L)	1 hand drag	Close, push/pull and drag 1 hand

TABLE C.2: First and second most agreed gesture proposals for each referent (video-browsing experiment). Referents with above-average agreement rates are highlighted in blue. Notation for the Agreement Rate (AR): L=low, M=medium, H=high, V=very high [301].

Referent	AR	1st most agreed gesture	2nd most agreed gesture
1. *Previous*	.498 (H)	Flick right	Flick left
2. *Next*	.498 (H)	Flick left	Flick right
3. *Enable fullscreen*	.273 (M)	Move hands apart	Pinch out
4. *Disable fullscreen*	.146 (M)	Move hands closer together	Pinch in
5. *Zoom in*	.320 (H)	Pinch out	Move hands apart
6. *Zoom out*	.300 (M)	Pinch in	Move hands closer together
7. *Volume up*	.478 (H)	Swipe up	Rotate a knob CW
8. *Volume down*	.316 (H)	Swipe down	Rotate a knob CCW
9. *Rotate 90°CW*	.830 (V)	Rotate a knob CW	N/A
10. *Rotate 90°CCW*	.830 (V)	Rotate a knob CCW	N/A
11. *Play*	.190 (M)	Tap with the index finger	Draw the "play" symbol
12. *Pause*	.166 (M)	Tap with the index finger	Swipe down with two fingers
13. *Like*	.751 (V)	Thumb up	N/A
14. *Dislike*	.751 (V)	Thumbs down	N/A
15. *Fast-forward 5 seconds*	.087 (L)	Flick right twice	Swipe left
16. *Rewind 5 seconds*	.095 (L)	Swipe right	Flick left twice
17. *Enable subtitles*	.032 (L)	Swipe up	Push
18. *Disable subtitles*	.028 (L)	Swipe down	Pull

TABLE C.3: First and second most agreed gesture proposals for each referent. Referents with above-average agreement rates are highlighted. Magnitude of the Agreement Rate (*AR*) [301]: L=low, M=medium, H=high, V=very high.

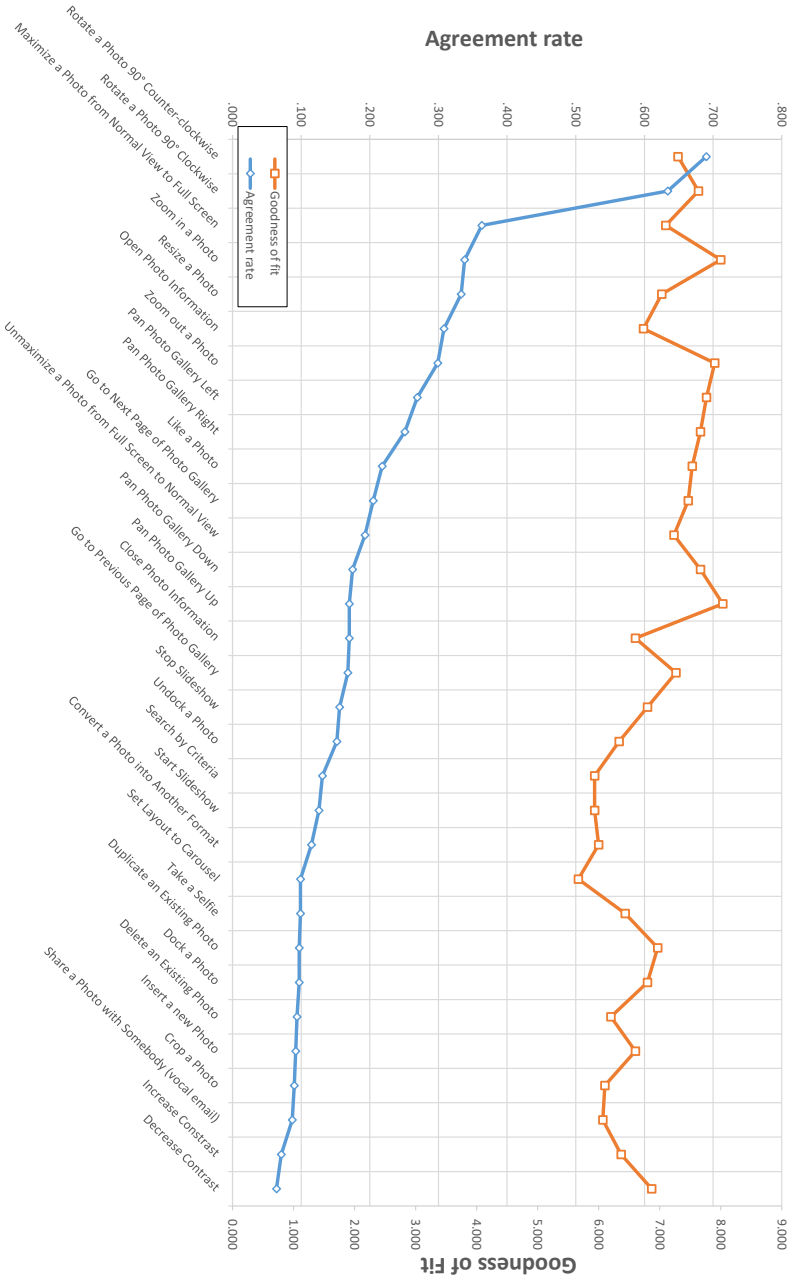


FIGURE C.2: Relationship between AGREEMENT-RATE and GOODNESS-OF-FIT.

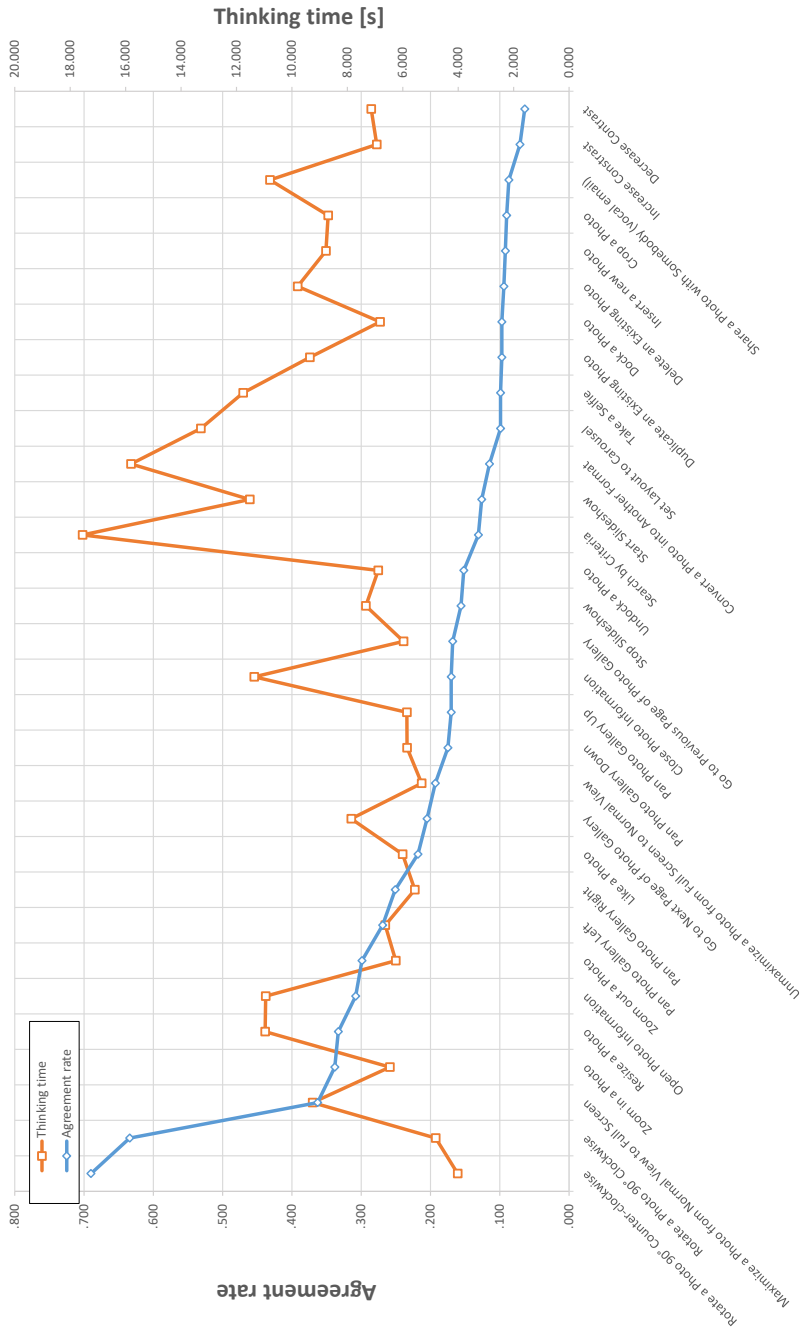
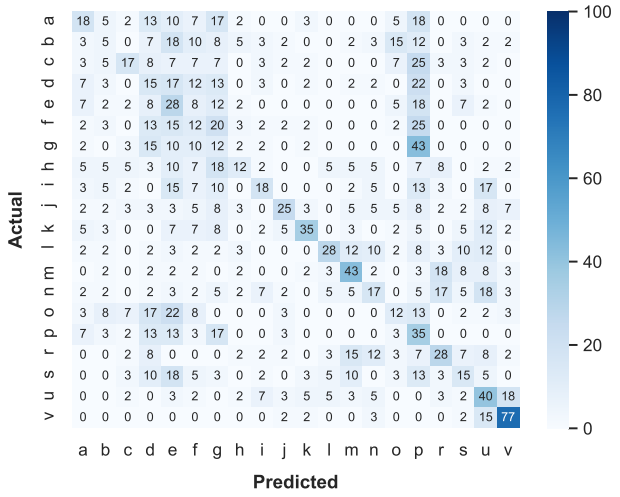


FIGURE C.3: Relationship between AGREEMENT-RATE and THINKING-TIME.

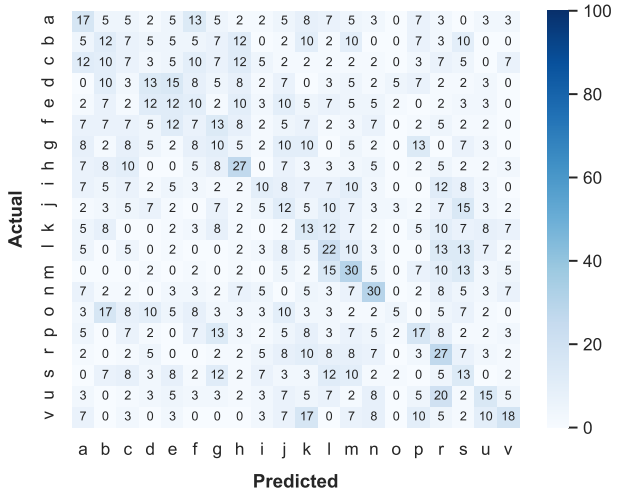
Appendix D

Experimenting with Radar-based Gestures (Additional Results)

D.1 Experiment 2: Gestures Subsets

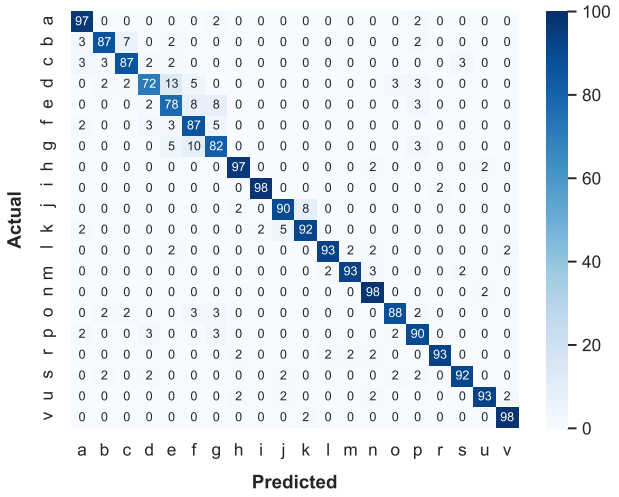


(A) Time gating N=38,
AP=(1, 2, 3, 6, 8, 9).

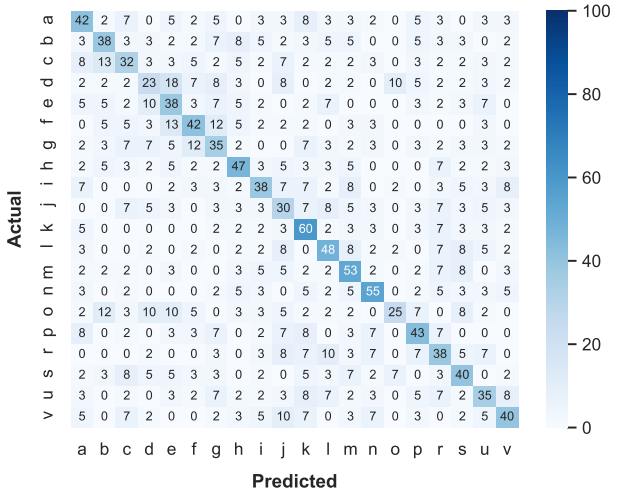


(B) Filtering N=29,
AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

FIGURE D.1: Confusion matrices of the best-performing configurations in the user-independent scenario on the complete dataset.



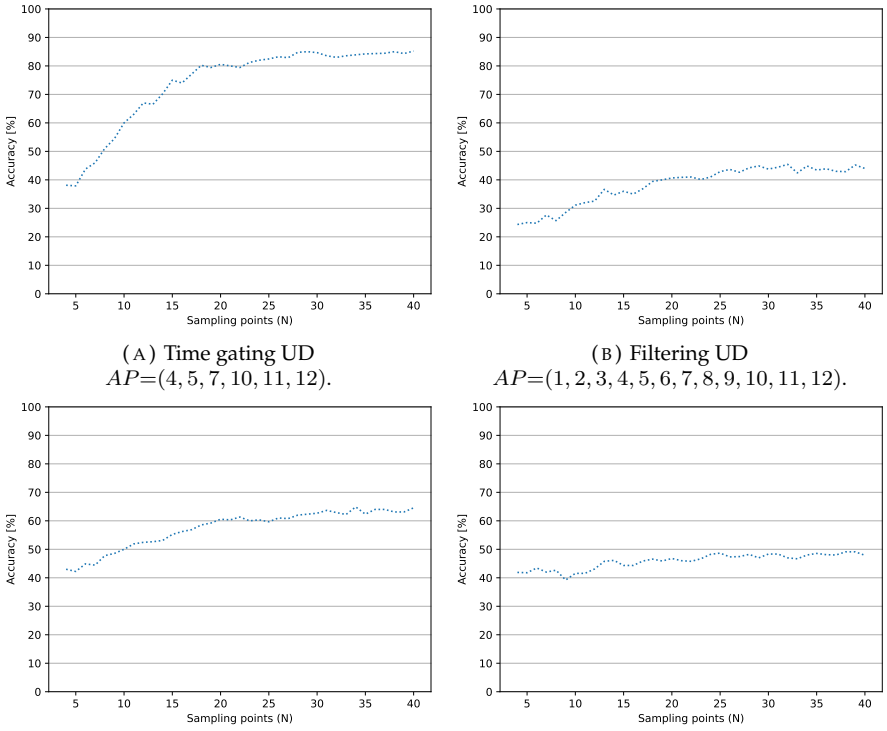
(A) Time gating N=39,
AP=(1, 2, 3, 6, 8, 9).



(B) Filtering N=25,
AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

FIGURE D.2: Confusion matrices of the best-performing configurations in the mixed scenario on the complete dataset.

D.2 Experiment 3: Gestures through Materials



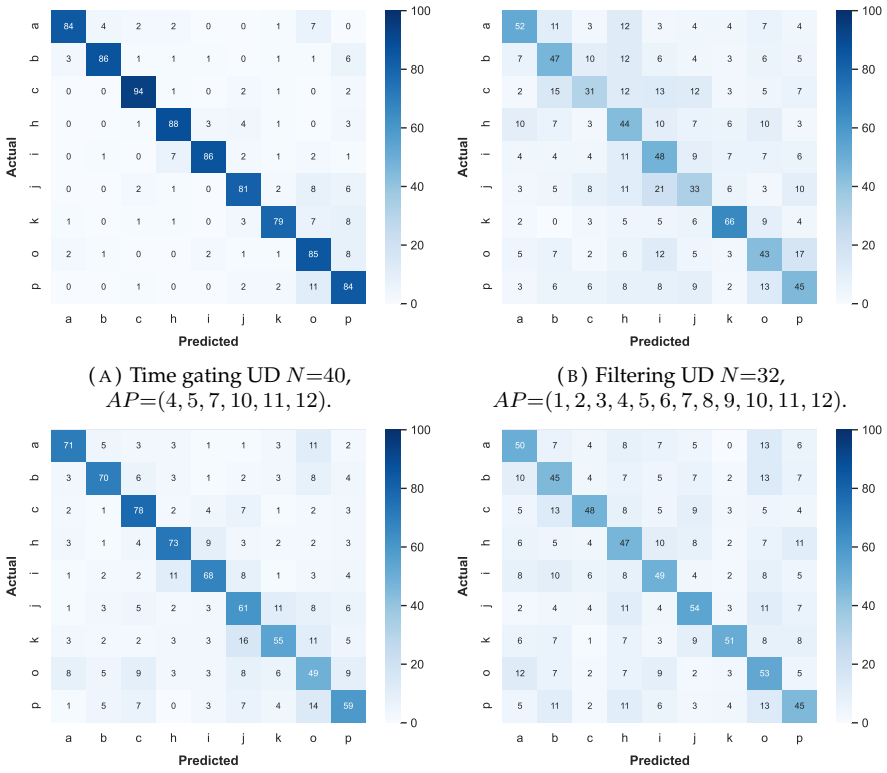
(A) Time gating UD AP=(4, 5, 7, 10, 11, 12).

(B) Filtering UD AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(C) Time gating UI AP=(4, 5, 7, 10, 11, 12).

(D) Filtering UI AP=(1, 2, 3, 6, 8, 9).

FIGURE D.3: The accuracy of Jackknife with respect to the number of sampling points for the best performing set of antenna pairs when using PVC data for training and testing in a user-dependent (top) and user-independent scenario (bottom).



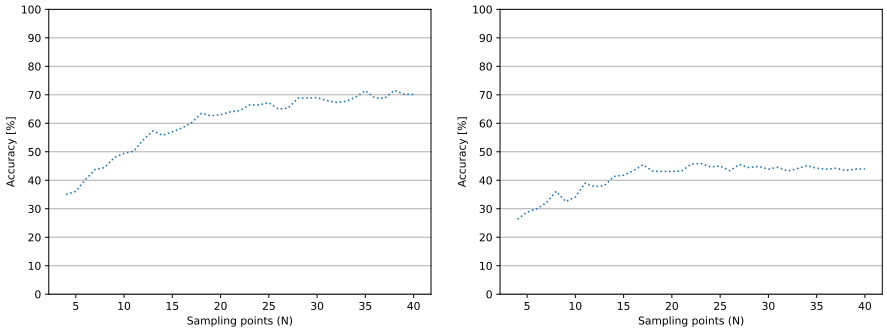
(A) Time gating UD N=40, AP=(4, 5, 7, 10, 11, 12).

(B) Filtering UD N=32, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

(C) Time gating UI N=34, AP=(4, 5, 7, 10, 11, 12).

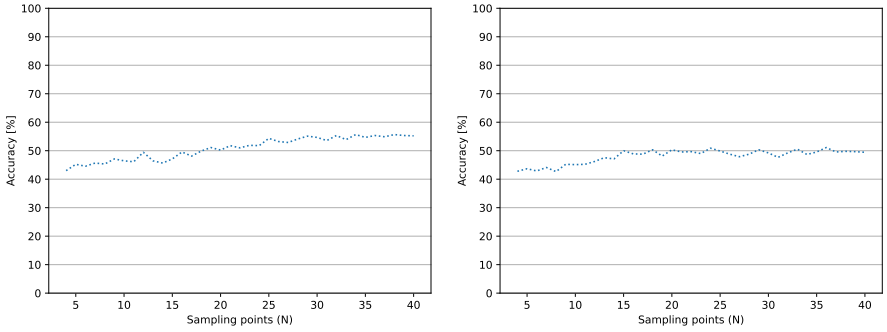
(D) Filtering UI N=38, AP=(1, 2, 3, 6, 8, 9).

FIGURE D.4: Normalized confusion matrices for the best configuration at two stages of the pipeline in user-dependent (top) and user-independent scenarios (bottom) when using PVC data for training and testing. The values in each cell are represented as percentages.



(A) Time gating UD
AP=(1, 2, 3, 6, 8, 9).

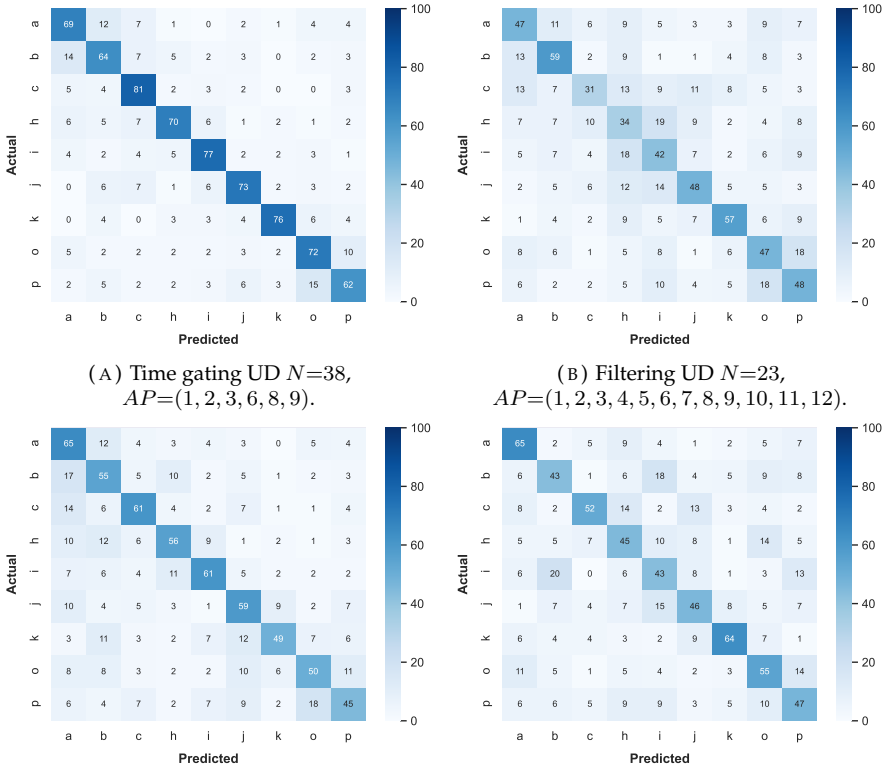
(B) Filtering UD
AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).



(C) Time gating UI
AP=(1, 2, 3, 6, 8, 9).

(D) Filtering UI
AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

FIGURE D.5: The accuracy of Jackknife with respect to the number of sampling points for the best performing set of antenna pairs when using glass data for training and testing in a user-dependent (top) and user-independent scenario (bottom).



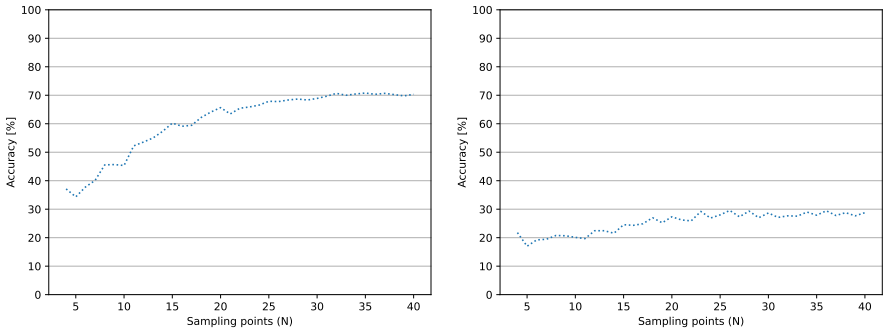
(A) Time gating UD N=38, AP=$(1, 2, 3, 6, 8, 9)$.

(B) Filtering UD N=23, AP=$(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)$.

(C) Time gating UI N=34, AP=$(1, 2, 3, 6, 8, 9)$.

(D) Filtering UI N=36, AP=$(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)$.

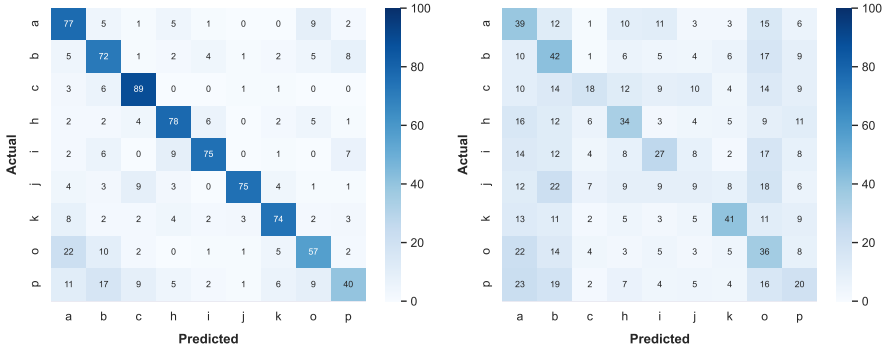
FIGURE D.6: Normalized confusion matrices for the best configuration at two stages of the pipeline in user-dependent (top) and user-independent scenarios (bottom) when using glass data for training and testing. The values in each cell are represented as percentages.



(A) Time gating AP=(4, 5, 7, 10, 11, 12).

(B) Filtering AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

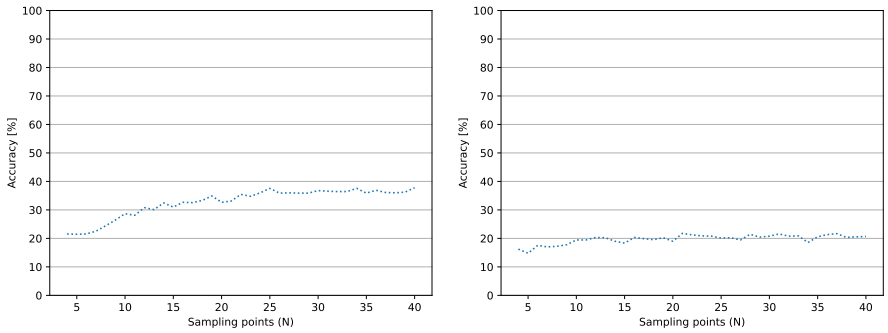
FIGURE D.7: The accuracy of Jackknife with respect to the number of sampling points for the best performing set of antenna pairs when using PVC data for training and wood data for testing in a user-dependent scenario.



(A) Time gating N=35, AP=(4, 5, 7, 10, 11, 12).

(B) Filtering N=26, AP=(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12).

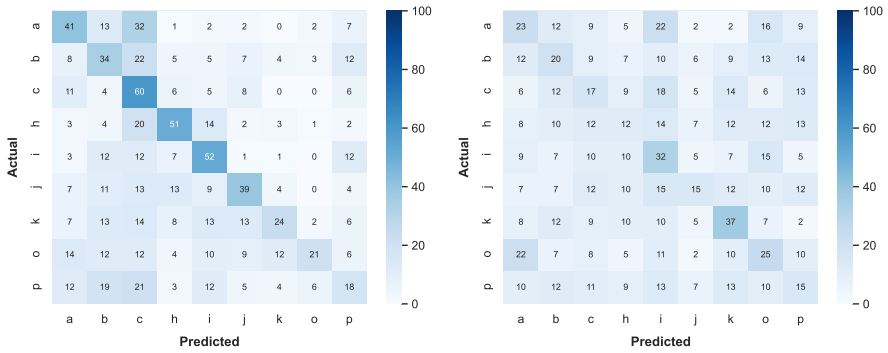
FIGURE D.8: Normalized confusion matrices for the best configuration at two stages of the pipeline in the user-dependent scenario when using PVC data for training and wood data for testing. The values in each cell are represented as percentages.



(A) Time gating AP=(1, 2, 3, 6, 8, 9).

(B) Filtering AP=(1, 2, 3, 6, 8, 9).

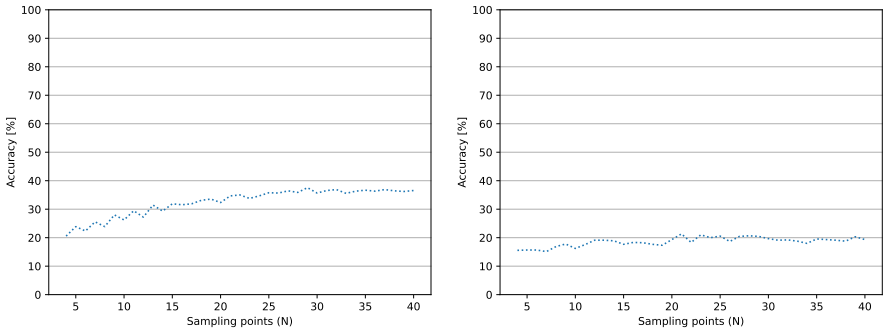
FIGURE D.9: The accuracy of Jackknife with respect to the number of sampling points for the best performing set of antenna pairs when using wood data for training and glass data for testing in a user-dependent scenario.



(A) Time gating N=40, AP=(1, 2, 3, 6, 8, 9).

(B) Filtering N=21, AP=(1, 2, 3, 6, 8, 9).

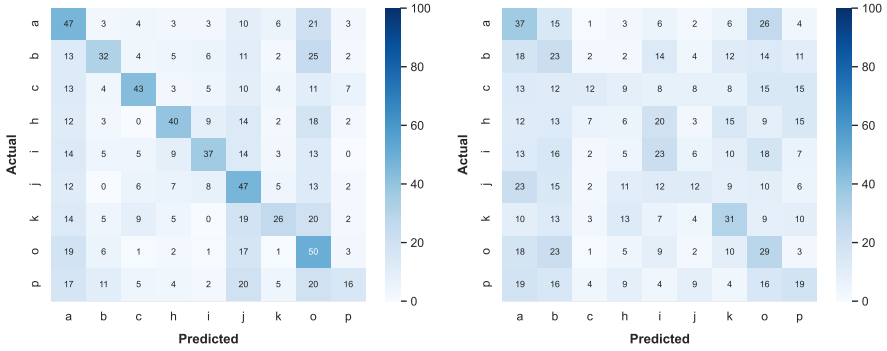
FIGURE D.10: Normalized confusion matrices for the best configuration at two stages of the pipeline in the user-dependent scenario when using wood data for training and glass data for testing. The values in each cell are represented as percentages.



(A) Time gating AP=(1, 2, 3, 6, 8, 9).

(B) Filtering AP=(1, 2, 3, 6, 8, 9).

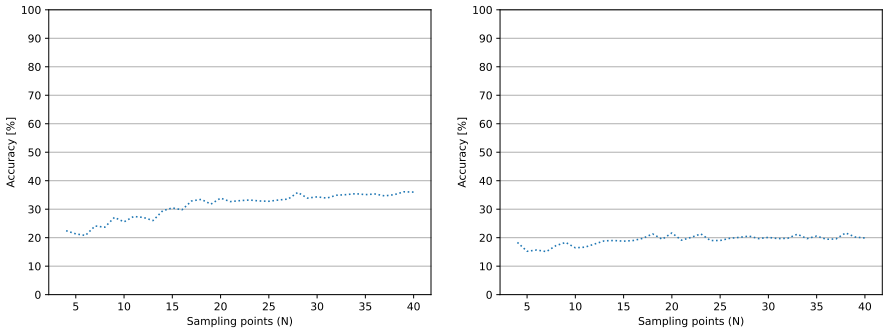
FIGURE D.11: The accuracy of Jackknife with respect to the number of sampling points for the best performing set of antenna pairs when using glass data for training and wood data for testing in a user-dependent scenario.



(A) Time gating N=29, AP=(1, 2, 3, 6, 8, 9).

(B) Filtering N=21, AP=(1, 2, 3, 6, 8, 9).

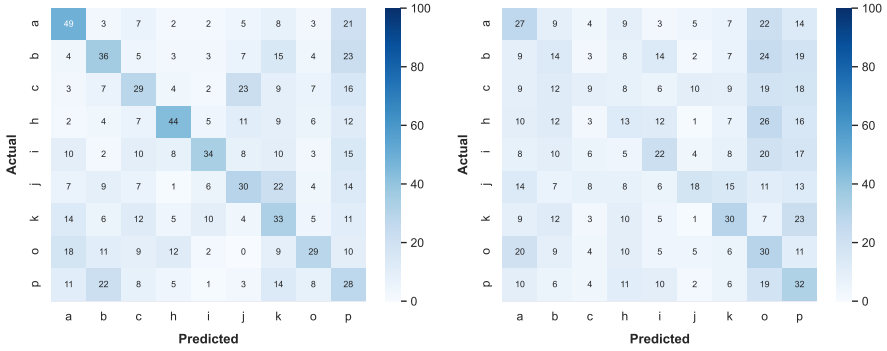
FIGURE D.12: Normalized confusion matrices for the best configuration at two stages of the pipeline in the user-dependent scenario when using glass data for training and wood data for testing. The values in each cell are represented as percentages.



(A) Time gating AP=(1, 2, 3, 6, 8, 9).

(B) Filtering AP=(1, 2, 3, 6, 8, 9).

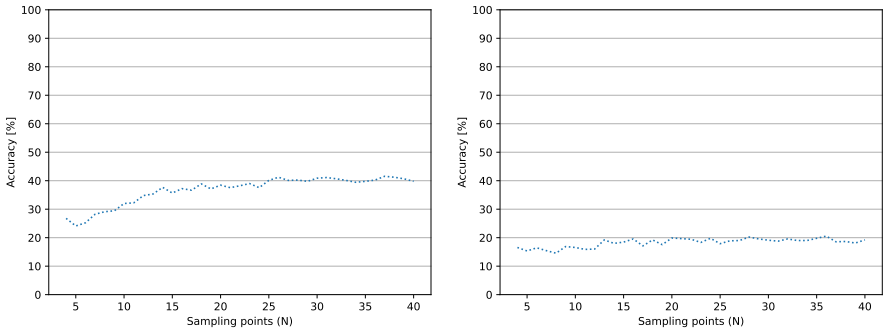
FIGURE D.13: The accuracy of Jackknife with respect to the number of sampling points for the best performing set of antenna pairs when using PVC data for training and glass data for testing in a user-dependent scenario.



(A) Time gating N=37, AP=(1, 2, 3, 6, 8, 9).

(B) Filtering N=20, AP=(1, 2, 3, 6, 8, 9).

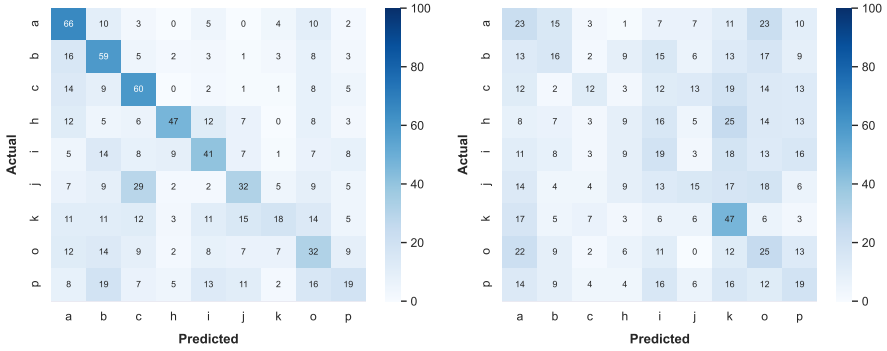
FIGURE D.14: Normalized confusion matrices for the best configuration at two stages of the pipeline in the user-dependent scenario when using PVC data for training and glass data for testing. The values in each cell are represented as percentages.



(A) Time gating AP=(4, 5, 7, 10, 11, 12).

(B) Filtering AP=(1, 2, 3, 6, 8, 9).

FIGURE D.15: The accuracy of Jackknife with respect to the number of sampling points for the best performing set of antenna pairs when using glass data for training and PVC data for testing in a user-dependent scenario.



(A) Time gating N=37, AP=(4, 5, 7, 10, 11, 12).

(B) Filtering N=36, AP=(1, 2, 3, 6, 8, 9).

FIGURE D.16: Normalized confusion matrices for the best configuration at two stages of the pipeline in the user-dependent scenario when using glass data for training and PVC data for testing. The values in each cell are represented as percentages.