The Foldinterface Editor: A Visual Tool for Designing User Interfaces for Foldable Displays

Jean Vanderdonckt
Université catholique de Louvain
Louvain Research Institute in Management and Organizations
Louvain-la-Neuve, Belgium
jean.vanderdonckt@uclouvain.be

Iyad Khaddam
Université catholique de Louvain
Louvain Research Institute in Management and Organizations
Louvain-la-Neuve, Belgium
iyad.khaddam@uclouvain.be

Radu-Daniel Vatavu
MintViz Lab,
MANSiD Research Center
University Ştefan cel Mare of Suceava
Suceava 720229, Romania
radu.vatavu@usm.ro

ABSTRACT

Designing foldable user interfaces ("foldinterfaces") is complex and time-consuming because of the multiple technologies involved in the process, from the hardware details for folding pixels to design requirements regarding efficiency and ease of use. To assist this process, we introduce a visual editor for designers to specify foldinterfaces by implementing an extension of the Yoshizawa-Randlett diagramming system for origami and Event-Condition-Action rules from event-driven software architecture. The outcome is rendered as a 3-D foldable surface in a virtual environment.

CCS CONCEPTS

• **Human-centered computing → Interactive systems and tools**; *Graphical user interfaces*; *Virtual reality*; • **Software and its engineering → Integrated and visual development environments**;

KEYWORDS

Foldable displays; Foldable user interfaces; Foldinterfaces; Graphical User Interfaces; Origami; Kirigami; Visual editor; Prototyping; Tool.

ACM Reference Format:
Jean Vanderdonckt, Iyad Khaddam, and Radu-Daniel Vatavu. 2020. The Foldinterface Editor: A Visual Tool for Designing User Interfaces for Foldable Displays. In *ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '20 Companion), June 23–26, 2020, Sophia Antipolis, France.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3393672.3398490

EICS '20 Companion, June 23–26, 2020, Sophia Antipolis, France

ACM ISBN 978-1-4503-7984-7/20/06…$15.00
https://doi.org/10.1145/3393672.3398490

1 INTRODUCTION

Unlike conventional screens, foldable displays present the attractive feature that they can fold into small form factors when not used at full display capacity and, consequently, they optimize the display area and physical space to accommodate a variety of desktop, mobile, and wearable contexts of use [2, 10, 12, 14, 18, 22, 24]. Computer displays have improved considerably by incorporating new technologies, including liquid crystals, LEDs, organic LEDs, and active-matrix organic LEDs [23]. Moreover, the recent availability of commercial foldable smartphones[1] suggests a realistic future for thinner, lighter, and reconfigurable computer displays that can accommodate a diversity of form factors and contexts of use where screen space is defined *relative* to the task and is no longer absolute.

The theoretical concept of a "foldable display" has an important role in this process by enabling on-the-fly adaptation of the display form factor, such as by means of gesture input [5], and by supporting new interactions besides conventional input techniques [2]. Foldable displays act as both observation and action surfaces according to the classification of Coutaz *et al.* [4] as users interact with the graphical user interface (GUI) by touch and/or gesture input and the effects are rendered on the same surface. Foldable displays offer a large interaction surface when unfolded and reduce to a minimum size when stored away. They also offer new opportunities for rich interactions, such as by means of "folding gestures," and greater adaptability to the context of use.

However, designing user interfaces for foldable displays is conditioned by the availability of the physical hardware, which hinders prototyping and evaluation. To address this aspect, we introduce the "Foldinterface Editor," a visual editor that enables specification of the GUI of foldable interactive displays and surfaces and visualizes the result in the form of a virtual surface that unfolds in 3-D. The Foldinterface Editor makes prototyping less expensive and less dependent on the availability of hardware and, thus, fosters new explorations of design possibilities for user foldinterfaces.

[1] https://www.samsung.com/global/galaxy/galaxy-fold

2 RELATED WORK

Probably the first manifestation of a folding digital surface is identifiable in the "Fold & Drop" interaction technique of Dragicevic [5] from 2004, where flipping between the windows of the GUI was achieved through "folding" and "dropping" gestures instead of pressing keys on the keyboard to imitate real-world flipping of pages from a book. Soon after, two families of foldable GUIs emerged: *projection-based systems*, where the elements of the GUI are projected onto a foldable, uninstrumented surface that is tracked in space, and *foldable devices*, where the surface and display are integrated in the device itself. For example, GUIs can be video projected to various surfaces, such as scrolls, fans, and even umbrellas as demonstrated by Lee *et al.* [14]; PaperWindows [11] introduced digital paper by means of video projections onto a sheet of paper tracked by a motion capture system; and, in FlexPad [22], interactive content was displayed onto a foldable surface tracked by a Microsoft Kinect sensor mounted on the ceiling. MigriXML [17] graphically rendered user interfaces across multiple interactive surfaces in Virtual Reality (VR) and enabled migration with adaptation of UIs from one surface to another. FoldMe [12] projected a GUI on predefined hinges where IR markers served as reference points. Instead of projecting GUIs on foldable surfaces, devices were used to control a virtual scene in which the GUI was rendered: in Gallant *et al.* [6], a paper sheet with IR reflectors was tracked by a video camera and folding gestures were recognized; by means of the augmented sheet of paper, the user could control the GUI displayed on a conventional screen.

Other approaches to foldable displays considered extensions of display devices or their integration into complex form factors. For example, WristOrigami [26] was designed to extend the interaction surface of a smartwatch by connecting foldable displays on its four sides with rotary hinges and rubber strings. Various 2-D and 3-D configurations were enabled to support interactive tasks, such as the "cushion" fold, "cupboard" fold, "book" fold, and the "cube" fold [26]. The SurfaceConstellations technique and system [15] adopted a complementary approach by assembling individual displays into complex form factors, such as a prism.

Yet other approaches have considered tools for prototyping such interfaces. For example, Tangrami [25] provided a digital design toolkit in which the shape and functionality of a foldable UI could be specified with building blocks designed in a visual editor and subsequently printed and assembled. The connection between the foldable UIs and the prototype application was managed via the Open Sound protocol [25]. Foldios [18] are foldable interactive objects with embedded input sensing and output capabilities made out of thin-film printed electronics. Foldable maps [16] imitate real-world physical maps on which buildings are recreated in 3-D.

3 A VISUAL NOTATION SYSTEM FOR PROTOTYPING USER FOLDINTERFACES

In this section, we introduce a visual notation for foldinterfaces inspired by the practice of origami [13]. First, we present our definition for foldinterfaces.

Definition

In this work, we use the term "foldinterfaces" to denote all user interfaces designed for foldable interactive displays and surfaces, no matter the technology employed to display the user interface (*e.g.*, video projection), the capabilities embedded in the surface itself (*e.g.*, thin-film printed electronics), or the structure employed to compose multi-display systems that can change shape (*e.g.*, constellations of displays).

The Yoshizawa-Randlett diagramming system

We rely in this work on the Yoshizawa-Randlett diagramming system [13], a standard notation for making origami that specifies several basic operations; see Figure 1:[2] *valley fold* (a part of the model is folded onto another to form a "valley" shape; the valley fold is represented by a dashed line with an arrow pointing in the direction of the fold); *mountain fold* (one part of the model is folded onto the back of another to form a "mountain" shape; the mountain fold is represented by a dashed dotted line and with an empty arrow pointing in the direction of the fold); *fold and unfold* (one part of the model is folded and then unfolded using the valley fold; the same operation can be performed with a mountain fold); *turn over* (the invisible lines behind the model are represented with dotted lines); *rotate* (the model is rotated up to a specified angle in the direction indicated by the arrows).

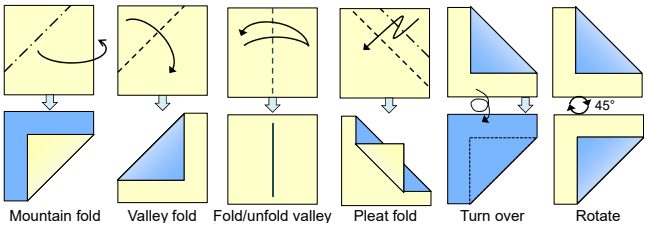

Figure 1: The Yoshizawa-Randlett notation for origami [13].

A Visual Notation for Foldinterfaces

The Yoshizawa-Randlett notation takes into account folding operations on surfaces that follow predetermined lines, as shown in Figure 1. However, foldinterfaces also present bending capabilities. To accommodate the latter, we introduce an extension of the Yoshizawa-Randlett notation for the scope

[2]Images redrawn from Yrogirg (Sarnitskiy Grigory), released into public domain; https://en.wikipedia.org/wiki/Yoshizawa-Randlett_system

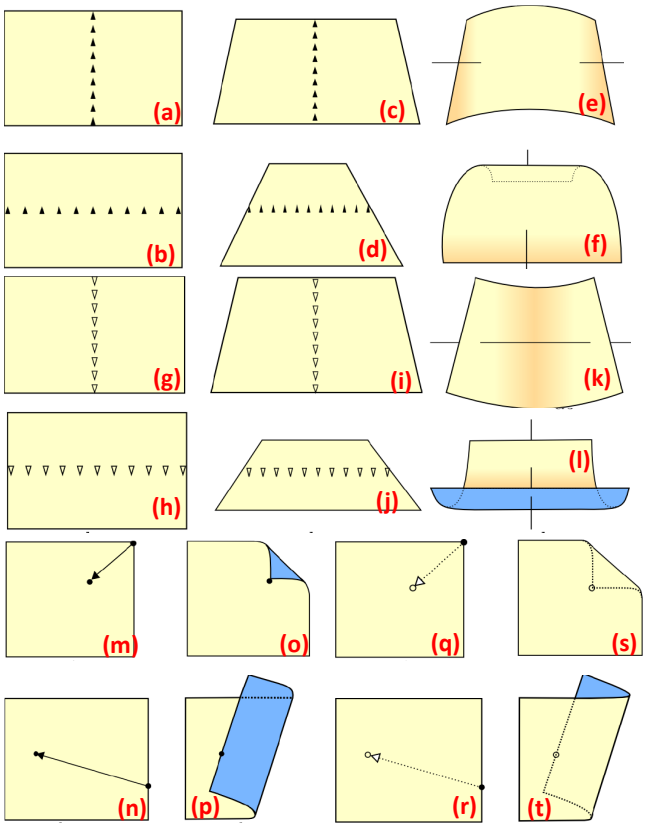

Figure 2: Our visual system for foldinterfaces: *mountain bending* (a and b denote flat states; c and d add perspective; e and f show the result of the bending); *valley bending* (g and h are flat states; i and j show perspective; k and l show the result); *valley corner-bending* (m, n flat state; o, p the result); *mountain corner-bending* (q, r flat state; s and t the result).

and purpose of foldinterfaces (see Figure 2) that consists of the following operations:

- *Mountain bending*: the surface bends upwards to form a "mountain"-like shape. The corresponding visual notation is a series of arrows pointing upwards.
- *Valley bending*: the surface bends downwards.
- *Valley corner-bending*: the valley-bending operation is applied to any part of the surface. The corresponding visual notation is a straight arrow (as opposed to a curved arrow for the Yoshizawa-Randlett valley fold). The starting point is denoted by a small circle that indicates where the surface that must be picked up to perform the bending operation. The ending point is also denoted with a circle and corresponds to the point on the surface where dragging must end.
- *Mountain corner-bending*: the mountain-bending operation is applied to any part of the surface. The corresponding notation is a dotted straight empty arrow.

4 THE FOLDINTERFACE EDITOR

Based on the visual notation system illustrated in Figure 2, we developed the Foldinterface Editor that enables prototyping of user foldinterfaces in four steps:

(1) *Surface creation.* The surface is specified using direct manipulation, *e.g.*, two rectangular surfaces can be joined to form a newspaper-like shape; see Figure 3a.
(2) *Definition of content.* Graphical content is imported and graphical widgets (*e.g.*, dials, radio buttons, push buttons, etc.) are positioned on top of the surface; see Figure 3b.
(3) *Definition of the interaction.* A set of Event-Condition-Action (ECA) rules [27] are specified; see Figure 3c. Each rule is composed of an *event* (which defines the signal that triggers the application of the rule), a *condition* (specifies when the corresponding action is executed), and an *action* (denotes the command to be executed when the condition is evaluated to true).
(4) *Automated generation of the foldinterface.* Once the previous steps are completed, the Foldinterface Editor automatically generates X3D code[3] for a virtual foldable surface corresponding to the specified foldinterface and renders it in a X3D-compliant browser for further examination and evaluation; see Figure 3d.

Conceptual Modeling in the Foldinterface Editor

The conceptual modeling of a foldinterface consists of two meta-models expressed as UML Class Diagrams V2.5 [3] and stored as XML files in the Foldinterface Editor. Transformation templates [1] are used to convert the XML files into X3D code via one-to-one mappings. Figure 4 illustrates the conceptual model of a a foldable display (Foldable-Display), which is composed of a set of Surfaces following the definition of interaction surfaces of Coutaz *et al.* [4]. Multiple Foldable-Displays link to each other through Fold-Links, a super-class for all the types of foldable operations illustrated in Figure 2. The Content rendered on a Foldable-Display is subject to the User-Interaction model that implements a set of ECA-Rules. For each rule, Events are decomposed into Event-Types and Event-Sources. A Condition is decomposed into three parts, as follows: the Condition-Object specifying the object for which the condition is evaluated, the Condition-Value specifying the value that is compared against, and the Condition-Operator denoting the operator used to compare the condition object against the expected value. An Action is decomposed into the Action-Object, Action-Type, and the Action-Value. Figure 3 shows the steps for creating a newspaper foldinterface based on the "fold-unfold" gestures from Figure 6. When the rules are specified, the foldinterface is automatically rendered in X3D and becomes actionable by

[3]https://www.web3d.org/x3d/what-x3d/

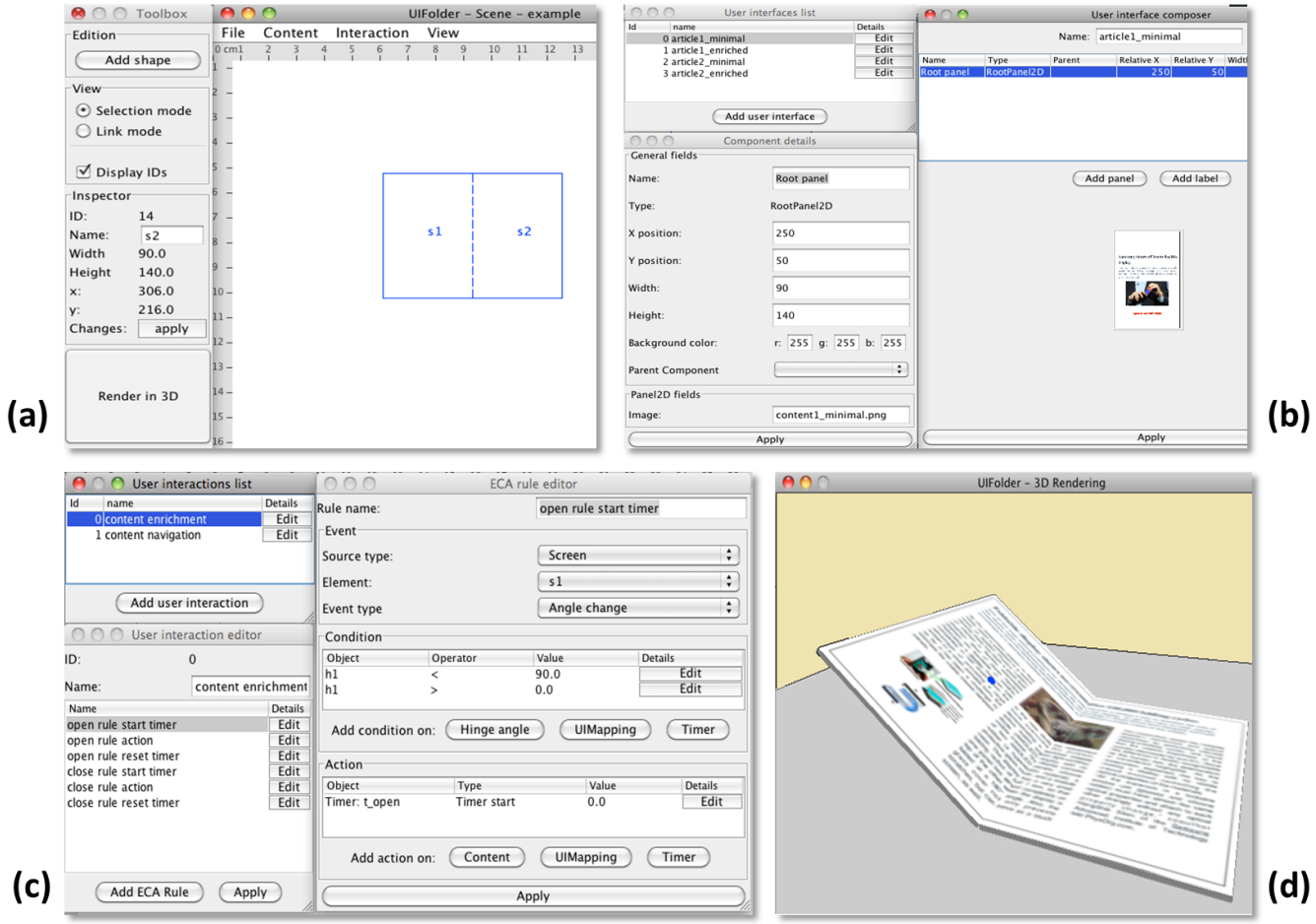

Figure 3: Example of a scenario for a foldable GUI; see also Figure 7.

means of the ECA rules. Figure 7 shows the same fold-unfold behavior, but with standard GUI widgets, such as text fields, spin buttons, and dials.

Implementation

We developed the Foldinterface Editor using Java 7 SDK with a total number of 186 expandable classes, of which 115 classes dedicated to GUI widgets and content for the foldable surfaces. At the moment, only rectangular surfaces can be defined with respect to the fold links and composed by joining their edges by means of direct manipulation. The current version of the Foldinterface is available on GitHub at the web address https://github.com/jeanvdd/UIFolder.

5 CONCLUSION AND FUTURE WORK

We introduced the Foldinterface Editor, a visual tool for prototyping designs of foldable graphical user interfaces that renders the results in X3D for visualization in a web browser. There are several directions for future development: building a catalog of foldable patterns that emerge from the practice

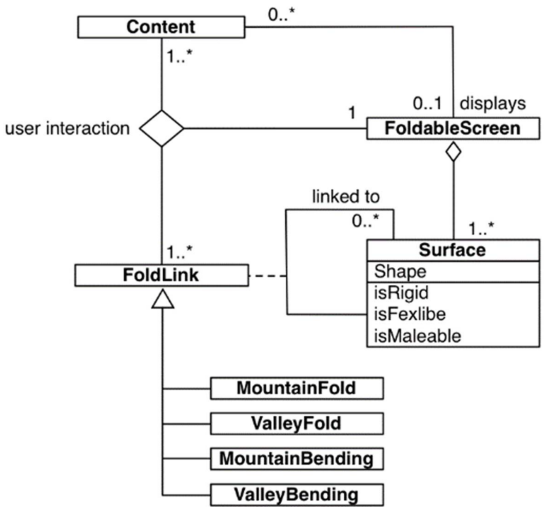

Figure 4: Conceptual model for a foldinterface.

of using foldinterfaces and extending our visual notation system and the rendering engine of the Foldinterface Editor to support bendable and rollable user interfaces. For

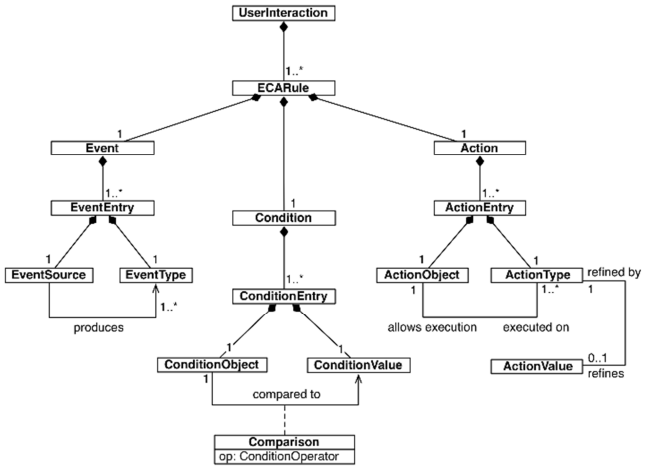

Figure 5: Model for interaction with foldinterfaces.

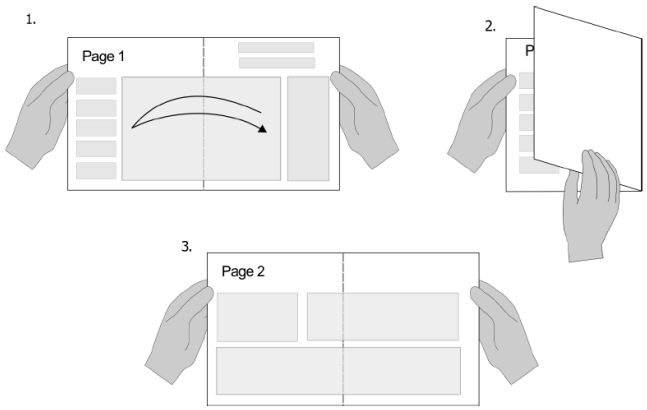

Figure 6: Content navigation on a newspaper-like shape foldinterface using a bimanual fold-unfold gesture.

instance, we would like to relax the current restriction in the Editor regarding rectangular surfaces by considering simple quadrilaterals, both concave (*e.g.*, a dart) and convex (*e.g.*, a trapezoid), but also complex quadrilaterals, such as antiparallelograms. The ultimate goal is to reach the capability to render any GUI on any surface or set of connected surfaces. Exploration of various types of gestures for interacting with foldinterfaces, such as hand and finger gestures sensed by smart rings [7], including bimanual gestures [8], and integrating foldable displays into multi-device architectures for smart environments [19, 20] represent interesting directions for future exploration as well.

Another direction for future work will explore more connections with the practice of Origami and Kirigami, respectively. The original art of Origami forbids cutting, gluing, or marking. When this constraint is relaxed, the Japanese word "Kirigami" is used instead. With the advent of shape-changing UIs, future work will investigate to what extent a Kirigami-like GUI could be prototyped toward novel foldinterfaces, such as cell-composed touchscreen surfaces [9] and modular devices that afford lending [21]. Also, the origami practice indicates a catalog of basic folds, of which some have direct and meaningful application to foldinterface as we showed in this work, while others are purely decorative. Therefore, we plan to compile an inventory of folds that are meaningful for user foldinterfaces inspired by origami galleries[4] and catalogs of foldable structures.

ACKNOWLEDGMENTS

The authors would like to thank Noé Phan for developing the Foldinterface Editor.

REFERENCES

[1] Nathalie Aquino, Jean Vanderdonckt, and Oscar Pastor. 2010. Transformation Templates: Adding Flexibility to Model-Driven Engineering of User Interfaces. In *Proceedings of the 2010 ACM Symposium on Applied Computing (SAC '10)*. ACM, New York, NY, USA, 1195–1202. https://doi.org/10.1145/1774088.1774340

[2] Jesse Burstyn, Amartya Banerjee, and Roel Vertegaal. 2013. FlexView: An Evaluation of Depth Navigation on Deformable Mobile Devices. In *Proceedings of the 7th International Conference on Tangible, Embedded and Embodied Interaction (TEI '13)*. ACM, New York, NY, USA, 193–200. https://doi.org/10.1145/2460625.2460655

[3] Ashley Bush and Sandeep Purao. 2011. Mapping UML Techniques to Design Activities. In *Information Modeling in the New Millennium*, Matti Rossi and Keng Siau (Eds.). IDEA Publishing Group, Chapter 12, 199–217.

[4] Joëlle Coutaz, Christophe Lachenal, and Sophie Dupuy-Chessa. 2003. Ontology for Multi-surface Interaction. In *Proc. of IFIP TC13 Int. Conf. on Human-Computer Interaction, 1st-5th Sept. 2003, Zurich (INTERACT'03)*. IOS Press, 447–454.

[5] Pierre Dragicevic. 2004. Combining Crossing-based and Paper-based Interaction Paradigms for Dragging and Dropping Between Overlapping Windows. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology (UIST '04)*. ACM, New York, NY, USA, 193–196. https://doi.org/10.1145/1029632.1029667

[6] David T. Gallant, Andrew G. Seniuk, and Roel Vertegaal. 2008. Towards More Paper-like Input: Flexible Input Devices for Foldable Interaction Styles. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST '08)*. ACM, New York, NY, USA, 283–286. https://doi.org/10.1145/1449715.1449762

[7] Bogdan-Florin Gheran, Jean Vanderdonckt, and Radu-Daniel Vatavu. 2018. Gestures for Smart Rings: Empirical Results, Insights, and Design Implications. In *Proc. of the 2018 Designing Interactive Systems Conference (DIS '18)*. 623–635. https://doi.org/10.1145/3196709.3196741

[8] Bogdan-Florin Gheran, Radu-Daniel Vatavu, and Jean Vanderdonckt. 2018. Ring X2: Designing Gestures for Smart Rings Using Temporal Calculus. In *Proceedings of the 2018 ACM Conference Companion Publication on Designing Interactive Systems (DIS '18 Companion)*. 117–122. https://doi.org/10.1145/3197391.3205422

[9] Alix Goguey, Cameron Steer, Andrés Lucero, Laurence Nigay, Deepak Ranjan Sahoo, Céline Coutrix, Anne Roudaut, Sriram Subramanian, Yutaka Tokuda, Timothy Neate, Jennifer Pearson, Simon Robinson, and Matt Jones. 2019. PickCells: A Physically Reconfigurable Cell-Composed Touchscreen. In *Proceedings of the 2019 CHI*

[4] https://www.giladorigami.com/origami-galleries.html

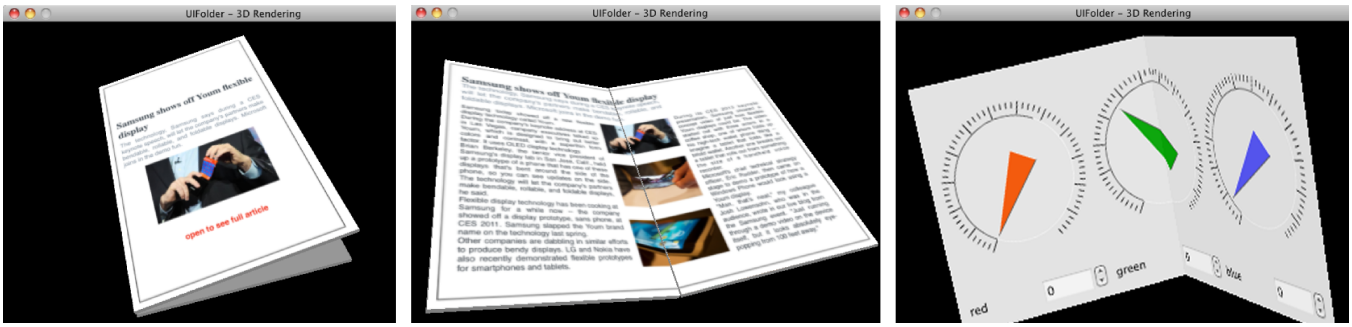

Figure 7: X3D representation of the foldinterface from Figure 6 in our Foldinterface Editor.

Conference on Human Factors in Computing Systems (CHI '19). Association for Computing Machinery, New York, NY, USA, Article Paper 273, 14 pages. https://doi.org/10.1145/3290605.3300503

[10] Kim Gustafson, Oyuna Angatkina, and Aimy Wissa. 2019. Model-based design of a multistable origami-enabled crawling robot. *Smart Materials and Structures* 29, 1 (nov 2019), 015013. https://doi.org/10.1088/1361-665x/ab52c5

[11] David Holman, Roel Vertegaal, Mark Altosaar, Nikolaus Troje, and Derek Johns. 2005. Paper Windows: Interaction Techniques for Digital Paper. In *Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI '05)*. ACM, New York, NY, USA, 591–599. https://doi.org/10.1145/1054972.1055054

[12] Mohammadreza Khalilbeigi, Roman Lissermann, Wolfgang Kleine, and Jürgen Steimle. 2012. FoldMe: Interacting with Double-sided Foldable Displays. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction (TEI '12)*. ACM, New York, NY, USA, 33–40. https://doi.org/10.1145/2148131.2148142

[13] Robert J. Lang. 2011. Origami Diagramming Conventions - Diagramming, Part 1. Web site. Retrieved November 26, 2018 from http://langorigami.com/article/origami-diagramming-conventions/.

[14] Johnny C. Lee, Scott E. Hudson, and Edward Tse. 2008. Foldable Interactive Displays. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST '08)*. ACM, New York, NY, USA, 287–290. https://doi.org/10.1145/1449715.1449763

[15] Nicolai Marquardt, Frederik Brudy, Can Liu, Ben Bengler, and Christian Holz. 2018. SurfaceConstellations: A Modular Hardware Platform for Ad-Hoc Reconfigurable Cross-Device Workspaces. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 354, 14 pages. https://doi.org/10.1145/3173574.3173928

[16] S. Martedi, H. Uchiyama, G. Enriquez, H. Saito, T. Miyashita, and T. Hara. 2010. Foldable augmented maps. In *2010 IEEE International Symposium on Mixed and Augmented Reality*. 65–72. https://doi.org/10.1109/ISMAR.2010.5643552

[17] José Pascual Molina Massó, Jean Vanderdonckt, and Pascual González López. 2006. Direct Manipulation of User Interfaces for Migration. In *Proceedings of the 11th International Conference on Intelligent User Interfaces (IUI '06)*. ACM, New York, NY, USA, 140–147. https://doi.org/10.1145/1111449.1111483

[18] Simon Olberding, Sergio Soto Ortega, Klaus Hildebrandt, and Jürgen Steimle. 2015. Foldio: Digital Fabrication of Interactive and Shape-Changing Objects With Foldable Printed Electronics. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. Association for Computing Machinery, New York, NY, USA, 223–232. https://doi.org/10.1145/2807442.2807494

[19] Ovidiu-Andrei Schipor, Radu-Daniel Vatavu, and Jean Vanderdonckt. 2019. Euphoria: A Scalable, Event-Driven Architecture for Designing Interactions Across Heterogeneous Devices in Smart Environments. *Information and Software Technology* 109 (2019), 43–59. https://doi.org/10.1016/j.infsof.2019.01.006

[20] Ovidiu-Andrei Schipor, Radu-Daniel Vatavu, and Wenjun Wu. 2019. SAPIENS: Towards Software Architecture to Support Peripheral Interaction in Smart Environments. *Proceedings of the ACM on Human-Computer Interaction* 3, EICS, Article Article 11 (June 2019), 24 pages. https://doi.org/10.1145/3331153

[21] Teddy Seyed, Xing-Dong Yang, and Daniel Vogel. 2017. A Modular Smartphone for Lending. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. Association for Computing Machinery, New York, NY, USA, 205–215. https://doi.org/10.1145/3126594.3126633

[22] Jürgen Steimle, Andreas Jordt, and Pattie Maes. 2013. Flexpad: Highly Flexible Bending Interactions for Projected Handheld Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 237–246. https://doi.org/10.1145/2470654.2470688

[23] Miriam Sturdee and Jason Alexander. 2018. Analysis and Classification of Shape-Changing Interfaces for Design and Application-Based Research. *ACM Comput. Surv.* 51, 1, Article Article 2 (Jan. 2018), 32 pages. https://doi.org/10.1145/3143559

[24] Aneesh P. Tarun, Peng Wang, Audrey Girouard, Paul Strohmeier, Derek Reilly, and Roel Vertegaal. 2013. PaperTab: An Electronic Paper Computer with Multiple Large Flexible Electrophoretic Displays. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 3131–3134. https://doi.org/10.1145/2468356.2479628

[25] Michael Wessely, Nadiya Morenko, Jürgen Steimle, and Michael Schmitz. 2018. Interactive Tangrami: Rapid Prototyping with Modular Paper-folded Electronics. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings (UIST '18 Adjunct)*. ACM, New York, NY, USA, 143–145. https://doi.org/10.1145/3266037.3271630

[26] Kening Zhu, Morten Fjeld, and Ayça Ünlüer. 2018. WristOrigami: Exploring Foldable Design for Multi-Display Smartwatch. In *Proc. of the 2018 Designing Interactive Systems Conference (DIS '18)*. ACM, New York, NY, USA, 1207–1218. https://doi.org/10.1145/3196709.3196713

[27] Michael Zoumboulakis, George Roussos, and Alex Poulovassilis. 2004. Active Rules for Wireless Networks of Sensors & Actuators. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*. ACM, New York, NY, USA, 263–264. https://doi.org/10.1145/1031495.1031527