Dynamic Decals: Pervasive Freeform Interfaces Using Constrained Deformable Graphical Elements

Aziz Niyazov, Nicolas Mellado, Loic Barthe, Marcos Serrano

Dynamic Decals: Pervasive Freeform Interfaces Using Constrained Deformable Graphical Elements

AZIZ NIYAZOV, University of Toulouse - IRIT, France
NICOLAS MELLADO, University of Toulouse - IRIT, France
LOIC BARTHE, University of Toulouse - IRIT, France
MARCOS SERRANO, University of Toulouse - IRIT, France

Pervasive interfaces can present relevant information anywhere in our environment, and they are thus challenged by the non rectilinearity of the display surface (e.g. circular table) and by the presence of objects that can partially occlude the interface (e.g. a book or cup on the table). To tackle this problem, we propose a novel solution based on two core contributions: the decomposition of the interface into deformable graphical units, called Dynamic Decals, and the control of their position and behaviour by a constraint-based approach. Our approach dynamically deforms the interface when needed while minimizing the impact on its visibility and layout properties. To do so, we extend previous work on implicit deformations to propose and experimentally validate functions defining different decal shapes and new deformers modeling decal deformations when they collide. Then, we interactively optimize the decal placements according to the interface geometry and their interrelations. Relations are modeled as constraints and the interface evolution results from an easy and efficient to solve minimization problem. Our approach is validated by a user study showing that, compared to two baselines, Dynamic decals is an aesthetically pleasant interface that preserves visibility, layout and aesthetic properties.

CCS Concepts: • **Human-centered computing → Interactive systems and tools**.

Additional Key Words and Phrases: Projected interfaces; Contact deformation; Constraints; Optimization.

ACM Reference Format:
Aziz Niyazov, Nicolas Mellado, Loic Barthe, and Marcos Serrano. 2021. Dynamic Decals: Pervasive Freeform Interfaces Using Constrained Deformable Graphical Elements . *Proc. ACM Hum.-Comput. Interact.* 1, 1 (October 2021), 27 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Pervasive interfaces (e.g. projection-based, embedded displays, tabletops [40]) are relevant in several contexts, such as smart buildings, university classrooms or museums, to provide situated access to contextual data, online resources and interactive content [19, 31, 51]. However, the deployment of such interfaces is challenged by the non-regularity of the display surface (e.g. a circular table), as well as the presence of objects that can occlude the projection (e.g. a cup or book on a table). Adapting current interfaces to better fit and take profit of the available space is complex as traditional GUIs windows and widgets are mostly rectangular and designed for rectangular screens. Besides, in such pervasive contexts the interface should be able to dynamically adapt to the available display space (e.g. when the user moves a cup over the interface), which poses an additional problem.

Most of the previous solutions in HCI addressing this problem of content occlusion consist in presenting the occluded content or an icon-sized miniature around the physical objects [14, 27], which brings additional limitations (content visibility of icon-sized miniatures, maximum number and size of elements). The few approaches trying to develop real freeform projections that would

Authors' addresses: Aziz Niyazov, University of Toulouse - IRIT, Toulouse, France, aziz.niyazov@irit.fr; Nicolas Mellado, University of Toulouse - IRIT, Toulouse, France, nicolas.mellado@irit.fr; Loic Barthe, University of Toulouse - IRIT, Toulouse, France, loic.barthe@irit.fr; Marcos Serrano, University of Toulouse - IRIT, Toulouse, France, marcos.serrano@irit.fr.

2021. 2573-0142/2021/10-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

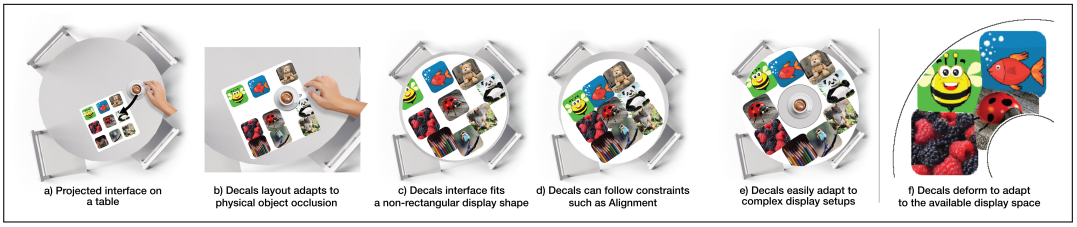


Fig. 1. In this work, we model pervasive freeform interfaces as dynamic arrangements of deformable UI widgets, called Dynamic Decals. (a) Our approach is fully dynamic and it automatically adapts the placement and shape of UI widgets w.r.t. (b) user interaction and (c) changes on the screen shape. The relations between the decals are modeled as constraints on their placement and are modified and optimized interactively to (d) enforce specific relations (e.g., alignment), or (e) to adapt the placement to complex display setup. We model decals as deformable objects (f) that are constrained by the screen boundaries and surrounding decals.

adapt to this available display space either are intended to augment tangibles instead of projecting regular 2D UI layouts [41] or deform the overall interface using content warping [8]. However, warping the entire interface has a negative impact on content visibility, since all objects on the vicinity of the deformed area are either hidden or severely deformed.

We present Dynamic Decals, denoted *decals* subsequently, a novel type of deformable UI widget that can overcome the aforementioned issues. This concept is inspired by previous work in computer graphics introducing repetitive patterns composing 2D textures [10]. As illustrated in Figure 1, in our approach all the elements of the graphical interface are represented as decals. Decals can be automatically deformed when colliding with other decals, with physical objects or with the boundaries of the display window. This shape deformation allows the interface content to become freeform and maximize space occupation. Furthermore, our approach allows us to define the position of the group of decals composing the interface to respect certain layout properties, such as alignment. This repositioning of the content at the decal level, preserves content visibility and readability. We thus focus on the following research challenges: (1) how to define the decal deformation according to its initial shape and content, and (2) how to define the placement constraints that rule the behaviour of an interface made of several decals.

Our technical contribution to solve the first challenge consists in the use of field function deformers: we introduce four new deformers for modeling UI objects when they collide based on two inside behaviors (squash or overlap) and two boundary behaviors (union or blending). We also describe the implementation of three different initial shapes for the decals: circular, rectangular and rectangular with rounded corners. We studied three types of decal content (icons, images and text), however the general approach is valid for any other content. To validate this implementation, we conducted a user study where we asked participants to rate the deformations that resulted from applying the four deformers to the aforementioned decals shape and content according to different visibility and aesthetics metrics.

To solve the second challenge, we propose a constraint-based approach. We contribute a novel set of constraints to ensure that the overall layout of decals complies with adequate layout properties. Constraints are defined as cost functions measuring certain properties of the interface, e.g., decals must stay in the display area and not overlap with physical objects, decals should stay at a given minimal distance. When the user interacts with the UI or when the display area is changed, we optimize the placement of the decals by minimizing the cost introduced by this change, as measured by the constraints. We validate these constraints through a controlled study where we implemented various interfaces and emulated either a physical object occluding its content or a change in the

overall display shape. Finally, we validate the Dynamic Decals approach though a user study comparing our approach with two baselines, when a physical object occludes the content and when the interface fits a freeform display area. Results show that compared to the baselines, users found our system to be aesthetically pleasant while preserving content visibility, layout properties (alignment and grouping) and usability.

To summarize, our contributions are at the intersection of HCI and Computer Graphics and can be detailed a follows: (1) the definition of Dynamic Decals; (2) the design and implementation of the graphical behaviour of Dynamic Decals, validated through a first user study; (3) the design and implementation of a constraint-based approach to control the placement of decals, validated through a controlled automatic evaluation; and (4) the validation of our approach through a final user study evaluating aesthetics, content visibility, layout properties and usability. Beyond these findings, we believe that this research opens new promising directions and is a source of inspiration for enriching HCI research with Computer Graphics methods and optimisation models for the design and implementation of new GUIs (e.g. immersive interfaces [13]).

2 RELATED WORK

Our goal is to define dynamic UIs where individual elements (e.g., icons, images) are moved and deformed depending on the display shape and its surroundings. We first review current approaches to deal with content occlusion in pervasive interfaces. We review methods to adapt traditional GUIs to non-rectangular displays, as well as graphical layout optimization.

2.1 Occlusion-aware interfaces

The pervasive display of user interfaces on the environment (either using tabletop or projected systems) raises the prevailing challenge of how to overcome virtual content occlusion by physical objects such as books, cups or plants [22, 51]. Vogel and Balakrishnan [52] defined the concept of Occlusion-aware interfaces, i.e. interaction techniques which know what area of the display is occluded to counteract potential problems or use the hidden area. The usual approach in these occlusion-aware interfaces consists on presenting the occluded content or an icon-sized miniature around the physical objects [14, 27], which are used as anchors. For instance, SnapRail [14] creates a circular widget rail around the physical objects, where the interface elements are attached. This approach has the disadvantage of not preserving the original layout of the interface, and adopting a circular layout can make it difficult to handle large virtual elements, or too many elements. ObjecTop [27] presents an icon-sized miniature representation of the occluded content (which remains in its original location) around the physical object. However, this approach does not preserve the content visibility, and requires the user to perform an explicit action to access the content.

Instead, our approach is to decompose the interface into smaller elements, and then propose a novel solution to reorganize these elements using constraints to preserve the desired UI properties, and if needed deform the content to fit into the available space.

2.2 Adapting GUI layouts to non-rectangular displays

Layout adaptation has first been studied for standard rectangular displays, such as desktop or mobile displays, to define one GUI that can adapt to multiple target platforms [34, 48]. Waldner et al. [53] developed a display-adaptive window management for irregular surfaces. Their approach semi-automatically placed the rectangular windows in the available display space. To best exploit the available freeform display space in pervasive environments, others have explored the use of projected interfaces combined with content mapping or deformation. For instance, the Illumiroom and RoomAlive concepts [24, 25] are room-sized systems where the virtual content is projected onto the walls. This content distortion can also be used to create perspective-aware interfaces [36].

The combination of the previous considerations, i.e. object occlusion management and content deformation, has been scarcely explored. Cotting et al. [8] developed the Display Bubbles system, a projected interface that is deformed to adapt to the environment. However, in this work the whole window is warped, which impacts the visibility of the content, particularly for UI elements that are close to the edges of the window (and which are more severely deformed). Controlling the deformation of the interface leads to a tradeoff between efficient space coverage and visibility/manipulability of the content. FlowPut [41] is an environment-aware framework that projects output on and around tangible objects: the projection is optimized by environmental constraints to avoid interferences between the projection and real-world objects. FlowPut proposes two presentation techniques: either to place visual elements based on environmental constraints, or to project as much visual content in the surrounding of the tangible object as possible. However, this work does not study how to optimize the layout of multiple UI elements on a freeform space to respect the desired UI properties.

Instead, our approach is based on decomposing the interface into deformable freeform widgets than can be dynamically rearranged, preserving the content visibility at the widget level. To ensure visibility, we also prioritize virtual objects arrangement over deformation. Our approach is related to previous attempts to define interfaces with particle systems applying some type of physics [1, 26, 29, 54]. For instance, [54] used an advanced game physics engine combined with a touch surface to add real-worl dynamics to interactive surfaces. This approach uses the physical simulation to handle collisions and frictions of virtual objects. Overall, physics-based approaches are well-adapted to develop playful interfaces and real-world metaphors. However, they have a limited expressive power and hence are less adapted to define gui layout properties than constraint-based approaches.

2.3 Constraint-based layout optimization

Standard user interfaces are usually designed manually, which limits their use in dynamic contexts. Several approaches have been proposed to optimize the interface layout and speeding up the creation process. For instance, Scout [50] is a system helping designers to explore alternatives through high-level constraints based on design concepts such as semantic structure or emphasis. Dayama et al. [9] propose an optimization approach to generate grid-based layouts. These methods can also integrate a modelization of the user's performance. For instance, Duan et al. [12] present an automatic method to optimize the layout of mobile UIs using a predictive model of task performance. Constraints have also been applied to generate web interfaces according to predefined requirements, such as image placement depending on the page width [5]. All these approaches are similar in that they generate static versions of the interface, either during the design process, or before rendering (as for web interfaces).

Closer to our contributions, automatic methods have been proposed to dynamically adapt interfaces to user actions. In 1962 Sutherland [49] already used atomic constraints in the seminal Sketchpad to control lines organization: vertical, horizontal, parallel, or perpendicular. Constraints were then used to assist direct manipulation techniques in interactive geometric modelling [17, 21, 47]. For instance, Gleicher [16, 17] used constraints to establish and preserve relationships between geometric objects positioned with direct manipulation. Most of these approaches rely on linear constraints, which do not scale well to real-world applications. Instead, Hosobe [20] proposed a constraint solver handling non-linear geometric constraints such as Euclidean geometric, non-overlapping, and graph layout constraints. When applied to UIs, constraints help to define the state of an interface according to user input, as in ConstraintJS [38], or to dynamically adjust layouts when the window dimensions change. For instance, Cassowary [2] is a constraint solver algorithm used in windows management in some commercial systems, such as Apple's Auto Layout. However,

most of these systems perform on rectangular content distributed on linear grids. Riemann et al. [41] use environmental constraints (projection quality, proximity to desired location, orientation, scale) to project content on and around tangible objects. However, they did not apply such constraints to reorganize traditional UI layouts. To our knowledge no previous approach tackles the challenge of the organization of non-rectangular UI elements in a freeform display space.

3 DYNAMIC DECALS

Before detailing our contribution, we describe the problems we intend to solve with a usage scenario.

3.1 Problem Statement

Bob is a student at a smart campus, where classrooms have been augmented by projecting interfaces on tables. As the HCI class starts, the system projects the file navigation showing a grid of folders. The table is circular and the projected interface is square, hence not fully exploiting the available display space (1-space coverage). The teacher starts providing the session instructions, and Bob needs to take some notes and opens his notebook, covering a part of the interface. Then Bob needs to open a specific folder, which is hidden by his notebook (2-object occlusion). He moves the book and selects the folder to open. The folder contains a set of images. Bob decides to rescale his window to avoid any object occlusion. The window now covers only half of the table, which leaves large parts of the table unused and makes image preview icons too small (3-content visibility). Students need to carry out a 5 minutes brainstorming session using the post-it application. Bob starts adding and linking post-its. He needs more space so he decides to enlarge the projected window. Then he needs to manually displace the post-its to the available spaces that are not covered by physical objects, which is quite tedious (4-content positioning).

3.2 Overview of the proposed approach

To solve these challenges, and achieve dynamic layout optimization, we divide the interface into graphical elements and cast our problem as an interactive dynamic placement problem. Our approach is composed of three main contributions: (1) the definition of a novel type of deformable GUI element, the Dynamic Decals (denoted *decals*), ; (2) a dynamic layout optimisation model, which moves the decals according to predefined constraints and user interaction; and (3) a deformation model, which changes the shapes of colliding decals. The approach of decomposing the interface into multiple small graphical elements is well adapted to pervasive scenarios applications, such as post-its for brainstorming, gallery of images, or grids of files and folders (see Illustrative Applications section in the Discussion). In this paper we focus on such scenarios, although we further discuss how our approach could be extended to consider more complex interface layouts in the Future Work section.

3.3 Decal definition and possible shapes

To define the dynamic graphical elements of our interface, we introduce a new contribution: the use of field function deformers with their formulation. Our dynamic decals are inspired by previous work in computer graphics, namely Implicit Decals. Implicit Decals are small tiles used to texture the surface of a free form 3D object and were introduced with an implicit formulation by de Groot et al. [10]. An Implicit Decal D_i is defined by a field function $f_i(p) : \mathbb{R}^2 \rightarrow [0, 1]$ whose value is 1 at the decal center, 0 at all points p along its influence limit and $\frac{1}{2}$ at points on the decal boundary (see Appendix A). A fundamental advantage of the field function formulation is to enable the automatic generation of contact deformations between n decals by combining their field functions f_i with a composition operator. In this paper we define three different decal shapes corresponding to the most common element shapes in GUIs (see Figure 2): square, rounded square and circle (all shape

equations are provided in Appendix B). Each decal content, e.g. text, icons or images, can then be stored as a 2D texture.

3.4 Layout constraint properties

In this section we introduce our formulation for the interactive layout adaptation. We propose to optimize the position of the decals according to desired layout properties, e.g. spacing, alignment. Each time the display property changes (e.g., the shape of the display changes, a physical object is moved on top of the display area), our system updates the decal positions.

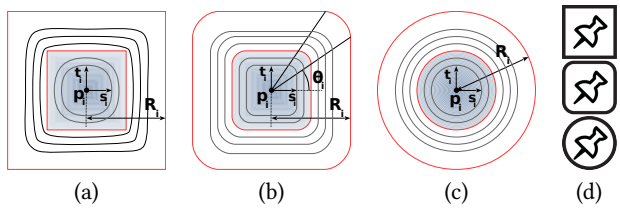


Fig. 2. (a) Square, (b) rounded square and (c) a circle decal shapes. In red, iso-lines $f_i = \frac{1}{2}$ and $f_i = 0$ defining respectively the decal boundary and its limit of influence. (d) Icon decal adapted to the three shapes

Our contribution is to extend the use of *arbitrary* constraints expressed as cost functions [33], which are minimized in the least squares sense to find a compromise between them. This enables the design of new constraints specific to GUI layout optimisation. Also, this approach allows to define a gamut constraint to force the decals to stay within the display area.

We now present the new constraints we have defined to preserve the desired GUI layout properties: GUI content visibility, layout simplicity, which refers to the number of lines and columns on the layout grid (i.e. if items are aligned, the layout is simpler), and content grouping. We designed these constraints empirically, following the layout simplicity guidelines introduced by Galitz [15], Ngo et al. [37]. In these constraints, coordinates and distances are expressed in pixels. All constraints equations are detailed in Appendix C.

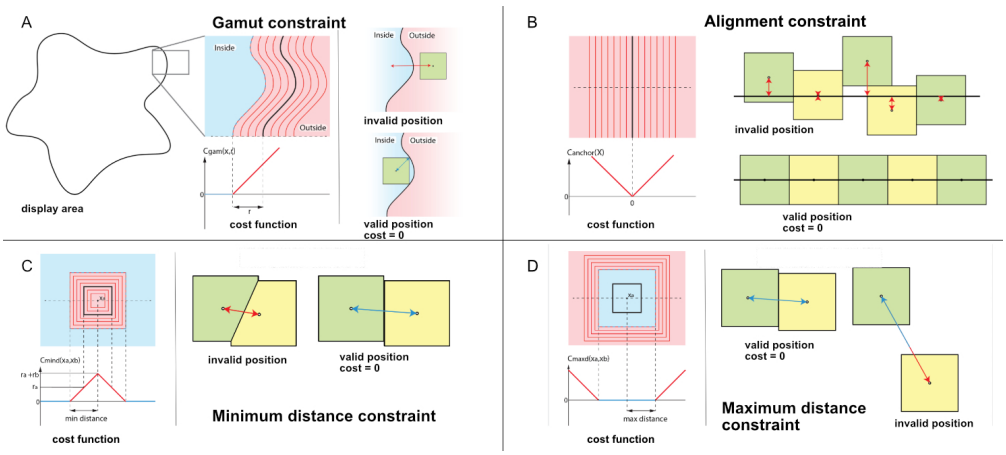


Fig. 3. Our layout constraints. For each one of them, we illustrate the cost function and the value of the cost according to the decals positions.

3.4.1 Gamut constraint. The goal of this constraint is to force the decals to stay within the display area (i.e. the gamut). A decal having a radius $r_i = n$ pixels is penalized if it lays *inside* the display area but at less than n pixels from the boundary (see Figure 3-A).

3.4.2 Distance constraints. We propose two distance constraints in order to (1) prevent decals overlap to preserve content visibility and (2) favor decals grouping. The **minimum distance** (see Figure 3-C) constraint penalizes overlapping decals. The **maximum distance** (see Figure 3-D) constraint ensures that decals belonging to the same group remain in close proximity to each other. We look for proximity and do not want to favor distances in x and y directions. Thus, we penalize decals when their L2 distance goes higher than a given threshold.

3.4.3 Alignment. The anchor line constraint ensures that decals belonging to the same vertical/horizontal line remain aligned even when one of the decals moves (see Figure 3-B). The cost is computed w.r.t. the distance between the anchor line l and each decal $\mathbf{x}_k$ belonging to a group. Any group of decals can be associated with multiple lines, either vertical, horizontal or both (e.g. to form a grid). Depending on the context, the position of the line can be either fixed (ie. coordinates are defined when creating the interface) or updated when the display properties change.

3.5 Deformation properties

When modeling the contact interaction of decals, there are two aspects to define : the way the content (i.e. the inside part) deforms and the way the boundary deforms. Concerning the content, our goal is to preserve its visibility during deformation. For the boundaries, the deformer could tend to preserve its shape to accentuate the separation between decals, or on the contrary, fuse them to indicate that decals belong to a same group of GUI elements (e.g. such as toolbar icons).

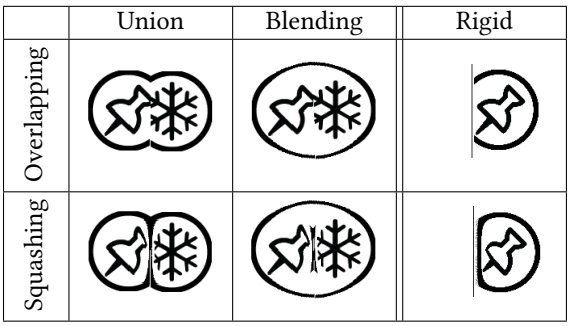


Fig. 4. The different types of deformers modeling the different contact behaviors for our study. By row, the inside content behavior (overlapping or squashing) and by column, the decal boundary behavior (union, blending or rigid).

In this paper, we thus introduce four new deformers for modeling UI objects interactions when they collide (equations are detailed in Appendix D). They are based on two inside behaviors - squashing or overlapping (Figure 4-rows) - and two decal boundary behaviors - union or blending (Figure 4-left and middle columns). The spacial case of the display boundary that remains rigid when a decal collides - i.e. only the decal is deformed - is handled with the same two inside decal behaviors (squashing or overlapping). Note that these two additional deformers, illustrated in Figure 4-right, are not part of the study (Section 5) as the squashing or overlapping behaviors of the decals is already evaluated.

4 IMPLEMENTATION

To achieve real-time performances, we use the versatile solver proposed by Mellado et al. [33], which allows to define arbitrary constraints between optimized elements. After each layout optimization step, we deform the colliding GUI elements with other elements or with display boundaries.

Interactive processing pipeline. Our approach is designed to dynamically update a GUI when the display area or its content change: we call such event an *update event*. Our prototype takes as input an augmented GUI, where GUI elements are modelled as decals and their relations described as constraints. Each time an update event is triggered, the system starts by updating the gamut: a 2D binary texture (black inside, white outside) caches the gamut of the current frame. Similarly, we compute and store in a 2D texture the associated signed distance field. For the studies, we combine

a static binary image representing the display area, modified by adding the footprint of the object at a given location. With this approach, the system can easily handle multiple physical objects, since no additional computation is needed to integrate each object. Then the solver is run, taking as input the decals, their relations, and the gamut distance field. The decals are deformed, and the GUI is rendered using the Qt library. Our prototype achieves interactive frame-rate on a standard laptop with a CPU Intel(R) Core(TM) i7-8850H 2.60GHz and 32GB RAM.

Optimization. As Mellado et al. [33], we minimize the constraint costs with the single thread implementation of Levenberg-Marquardt (LM) algorithm provided by Eigen [18]. As LM is a non-linear algorithm, it takes as input starting decal positions that are modified to minimize the energy. In order to get stable optimization results, we set the starting positions as the initial decal positions. When a user grabs and moves a decal in the GUI, the position of all the other decals are optimized, but not the selected one.

5 STUDY OF DECAL SHAPE AND DEFORMATION

To evaluate the previously presented decal shapes and deformers, we chose to conduct an online pairwise experiment in which participants had to rate pairs of conditions (in our case, pairs of animations showing two decals colliding). Such experiments are typical to gather Quality of Experience (QoE) feedback and have been previously used to compare freeform interface layouts [45].

5.1 Study description

We adopted an approach similar to [45] by conducting an online survey and using the Bradley-Terry-Luce [6] mathematical model to analyse the results, as detailed below.

5.1.1 Task. We conducted an online survey, which was divided into two parts. For the first part, the task consisted of comparing pairs of deformers. Each deformer was illustrated using an animated gif showing two decals colliding. For each pair of deformers, participants had to answer three questions to say which deformer was nicer (i.e. visually pleasing), clearer (i.e. with more visible/readable content), and which one better represented a group of items (association). The first two questions correspond to aesthetic terms proposed in the literature [30]. We focused on aesthetics because previous work has shown a strong correlation between aesthetics and usability [28]. The last question focused on whether, when two decals collide, the deformation makes it clear that the two decals are grouped, since GUI items are usually grouped (such as toolbar items). For each question the participant could provide three answers: pair A, pair B or both (i.e. equality). For the second part of the survey, the participants were asked to rank the decal shapes in order of preference for each type of content (icon, image and text) and motivate their choice.

5.1.2 Evaluated conditions. We tested the three decal shapes and the four deformers (Overlapping-Union, Overlapping-Blending, Squashing-Union and Squashing-Blending, see Figure 5). Concerning the decals content, we decided to study three common types of UI content: icons, images or text. Notice that we decided to only study collisions between decals having the same shape and content. As for the text, we used dummy Lorem Ipsum text. There were eleven words in both square and rounded square decal shapes, whereas only nine words fit in circular shape decal having the same font size. Regarding images and icons, we chose them so that the contact behaviour was noticeable enough on decals collision. All deformers were shown using the same collision trajectory, the speed was empirically set to 54 pixels per second so that the deformation was visible but not too fast. The speed of decals in the final application depends on the solver, and we do not apply any speed constraint (e.g. decals can move slower or faster, according to the user actions). The maximum

overlapping area of the decals was around 40%, which is the maximum overlap that would occur in our final system. In total we generated 36 animations corresponding to 4 deformers x 3 shapes x 3 content types.

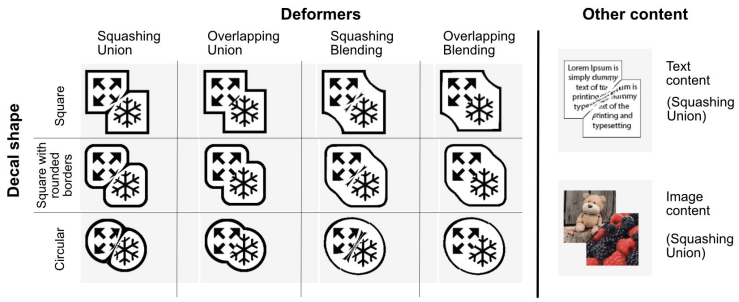


Fig. 5. In this survey we evaluated three decal shapes and four deformers (Left, illustrated with the Icon content). We also investigated these conditions with Text and Image content (Right, illustrated with Squashing-Union on a rectangular decal).

5.1.3 Participants. We had 31 complete answers to our study, but we removed results from 4 participants due to their low consistency (Section 5.2.1). The final 27 participants (13 female) were aged 28.1 on average (SD=6.9). 14 of them were university students, while the rest were professional workers from various areas: 3 researchers, 3 office workers, a teacher, a photograph, an illustrator, a computer scientist, a flight attendant, a surgeon and an architect.

5.1.4 Study Design. This study followed a 4x3x3 within-subject design, with Deformation operators (Overlapping-Union, Overlapping-Blending, Squashing-Union and Squashing-Blending), Shape (Square, Square with rounded borders or Circular) and Content type (Icon, Image or Text) as factors. For the first part of the study participants evaluated 54 pairs of animations (6 pairs combining the 4 deformers x 3 content types x 3 shapes). We used a 3x3 Latin square to counterbalance the order of the Shape factor. For the second part of the study, we asked participants to rank the three shapes for each content type.

5.2 Results

We first report on the consistency of the collected data before describing the results of the pairwise comparisons of the deformers and on the shape preference.

5.2.1 Data consistency. Our analysis of the results is based on the three steps described in [7] which validate the consistency of the collected data. The first step consists in computing the Transitivity Satisfaction Rate (TSR) to analyse the individual consistency judgement over multiple questions. After measuring the TSR for each participant, we removed 4 of them due to a low value (between 0.7 and 0.8). For the resting 27 participants, the TSR was above 0.8, meaning that they paid full attention to the study [7]. Second, we calculated the group consistency judgement using Kendall's tau coefficient [7]. The Kendall tau coefficient was greater than 0.5 for all answers to the 'nicer/clearer' questions, meaning that the overall judgement was consistent enough [7]. The answers to the 'grouping' question provided a Kendall tau coefficient above 0.5 only for some conditions (shapes having text as content, as well as circular icons). Therefore, we provide the 'grouping' results only for these conditions. Finally, we used the Bradley-Terry-Luce model [6] to

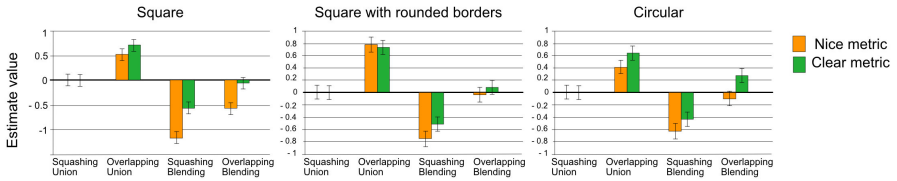


Fig. 6. Results for the pair comparisons between deformers for each decal shape (the y-axis represents the mean rating with respect to Squashing-Union, whose value is always equal to 0). Orange bars show the results for the Nice metric and green bars for the Clear metric.

get an "ability" metric for each condition as well as statistical differences between conditions with p-values. For sake of clarity, we add all details regarding these p-values in Appendix E.

5.2.2 Results on visibility and aesthetics. Figure 6 shows the results representing the Nice and Clear estimate values as computed via the Bradley-Terry-Luce model. For all figures, the Squashing-Union operator was used as reference. Hence, the estimates for the other deformation operators are provided with respect to Squashing-Union.

Results on the Nice and Clear metrics are consistent across decal shapes and contents. Overall, the Overlapping-Union operator was rated nicer and clearer than the other deformation operators (see Figure 6). Squashing-Union and Overlapping-Blending were perceived to be the most clear and nice after Overlapping-Union: we found no statistical difference between Squashing-Union and Overlapping-Blending except for rectangular icons and images, where Squashing-Union was deemed nicer. There were only two conditions where Overlapping-Union values were not significantly different than Squashing-Union and Overlapping-Blending: for icons presented on rectangular or rounded decals. The Squashing-Blending operator was always rated the least nicer and clearer.

5.2.3 Grouping. As said earlier, we only found significant results for the grouping value for Text content and Circular Shapes. Results reveal that Overlapping-Blending and Squashing-Blending are the deformers that better represent a grouping of multiple decals, except for text content on rectangular decals. Results reveal no significant difference between these two operators. While the fact that these two operators better represent a grouping behaviour was predictable, given that they apply a blending deformation, we discovered that this blending effect was stronger on textual decals.

5.2.4 Shape Preference and subjective feedback. When asked which decal shape they prefer, participants answers varied according to the content type. Rectangular shapes with rounded borders were preferred for icons (ranked first by 16 participants) and images (13). It is interesting to note that the rectangular shape was the least preferred for icons and images (ranked last 16 and 15 times respectively), even though we are used to display images on rectangular windows. Instead, the rectangular shape is preferred for text (12).

Participants commented on their shape preference choice. Concerning the overall preference for Rectangular shapes with Rounded borders (for icons and images), participants stated that the shape was more "pleasant" (P3, P22), and that "the rounded rectangle seems to allow the morphing in a more smooth fashion" (P6). Concerning the preference for rectangular shapes for textual content, participants referred to their habits and reading performance: "text in rectangles is more familiar and natural" (p7). Participants commented on circular shapes. Those that did not like the shape mostly referred to it being space inefficient: "circular is the least optimized" (P9). Instead, those that

liked it referred to the fact that the circular shape seems to be better for deformation: "changing the shape of a circle seems more natural than warping the rectangles" (P2).

5.3 Summary

To summarize, our study reveals that Overlapping-Union is the deformation operator that is perceived to be clearer and nicer. This result holds true for all decal shapes and content. To represent a group of decals, both deformer operators using blending seem to show better results but only for textual and circular content. Concerning the shapes, overall participants preferred the rectangular rounded shape for icons and images, and the rectangular shape for text. These results complement previous work on the perception of non-rectangular interfaces [44–46], which mostly focused on static content .

6 STUDY OF DECAL CONSTRAINTS

To validate our constraints, we conduct a simulation-based study on different application interfaces.

6.1 Study Method

6.1.1 Choice of experiment. Our study consists of simulating, for each interface, multiple object occlusions at predefined positions or changes in the overall display shape, then adapting the interface using different conditions, and automatically computing relevant metrics on the final interface layout. We chose to conduct such automatic testing to obtain a more systematic approach to compare the different optimization approaches (e.g. control the exact position of the physical object), thereby removing all possible biases introduced by object tracking flaws or user behaviour in a real setup.

6.1.2 Interfaces, constraints and conditions. We implement four different interfaces to test the effects of our constraints: an Image Viewer, a Mind Map, a 3x3 Grid of Images and a 5x5 Grid of folders (see Figure 7). We implement our decals with different constraint conditions. Either with only the Minimum distance constraint (namely Decals_Min) or with an additional constraint varying according to the application (namely Decals_Combined): we apply the alignment constraint on the toolbar icons of the Image Viewer and on the grids of images and folders (both on the lines and columns) ; we apply the maximum distance constraint on groups of post-its of the Mind Map. For each interface, we also implement two baselines: No_Deformation and Warping. The No_Deformation condition corresponds to an interface that would simply be occluded without any kind of deformation or positioning behaviour. The Warping condition is similar to those proposed in the literature [8] and consists of deforming the whole interface.

Concerning the disruption conditions, we either emulate a physical object covering the interface or change the overall display shape. Both are implemented as part of the gamut, i.e. through a gamut constraint. We simulate two different physical objects: a circular cup and a rectangular book (slightly rotated). We define 9 predefined positions of the object on the interface, corresponding to different levels of occlusion on the right and top side of the interface. For the change in display shape, we test 10 different shapes: four geometrical shapes (e.g. circle, octagon, hexagon and pentagon), four freeform shapes and two particular shapes: a donut (i.e. with an inner hole) and a 'separated' shape (i.e. composed of two separate parts).

6.1.3 Metrics and collected data. We automatically compute two metrics: content preservation and layout simplicity. The content preservation is calculated as the average ratio between the GUI elements size after deformation and their initial size. For the layout simplicity we use the formula provided by [37]: the simplicity is a function of the number of vertical and horizontal alignment points. We compute the difference between the initial simplicity and the value after

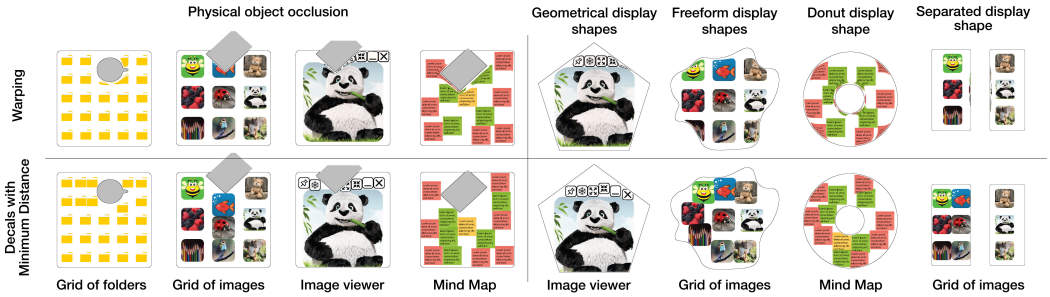


Fig. 7. Study results reveal that, compared to regular warping (top interfaces), our decals approach (bottom interfaces) improves content preservation when adapting the layout to an object occlusion or a non-rectangular display shape.

deformation. Note that the Simplicity metric is not directly optimized by the solver. Hence our constraints and metrics are independent.We do not compute the simplicity for the Mind Maps application, given that the post-its are not aligned, nor for the No_Deformation condition, as it does not change the layout of the interface. In total, we generated 448 interfaces: we compute the content preservation metric for all 448 interfaces, and the layout simplicity for 252 interfaces. For each one of the generated conditions we also captured a screenshot of the interface illustrating our results.

6.1.4 Data analysis. We analyze the results with a Shapiro-Wilk test determining the normality of the collected data. For all the reported data, the Shapiro-Wilk test reveals that data did not follow a normal distribution: we performed a Box-Cox transformation, but the data could not always be transformed to be normally distributed. Hence we conducted a Kruskall Wallis test to verify if there is a significant difference of the layout adaptation technique and a pairwise Wilcoxon test with Bonferroni correction to calculate pairwise comparisons between adaptation techniques.

6.2 Results

6.2.1 Content preservation. The Kruskall-Wallis test reveals a significant effect of the Layout adaptation technique ($p<.001$) on content preservation. The pairwise Wilcoxon test shows that Warping and No_Deformation are significantly different than both the Decals_Min ($p<.001$ for both) and Decals_Combined ($p<.001$ for both) conditions. These results hold true for conditions involving a layout adaptation to object occlusion and a non-rectangular display area. They indicate that the use of constraints significantly improves content preservation.

When adapting the layout to an object occlusion, content preservation is on average 96% for the Decal conditions, 89% for Warping, and 89% for No_Deformation. When adapting the interface to a non-rectangular area, this difference is even more pronounced in favor of decals: overall decals provide a content preservation of 95%, while Warping and No_deformation only 76%. This is in part due to the tested physical objects that only occlude a small part of the interface, which shows that the advantages of our approach are more evident as more content is occluded.

6.2.2 Layout simplicity. Concerning the simplicity metric, the Kruskall Wallis test reveals a significant effect of the Layout adaptation technique ($p<.001$). The pairwise Wilcoxon test shows that Decals_Combined is significantly different than both Decals_Min ($p<.001$) and Warping ($p<.001$) conditions in terms of layout simplicity. The results did not reveal any difference between Decals_Min and Warping ($p>.05$). Again, these results hold true for conditions involving a layout

adaptation to a physical object occlusion and to a non-rectangular display area. Overall the use of the Alignment constraint on Decals_Combined improves layout simplicity, as illustrated on Figure 8.

When occluding the interface with a physical object, layout simplicity average is 93% for Decals_Combined (i.e. with Alignment), 84% for Decals_Min (i.e. without Alignment), and 85% for Warping. When adapting the layout to a non-rectangular display, the simplicity with Decals_Combined is 93%, with Decals_Min 75% and with Warping 81%.

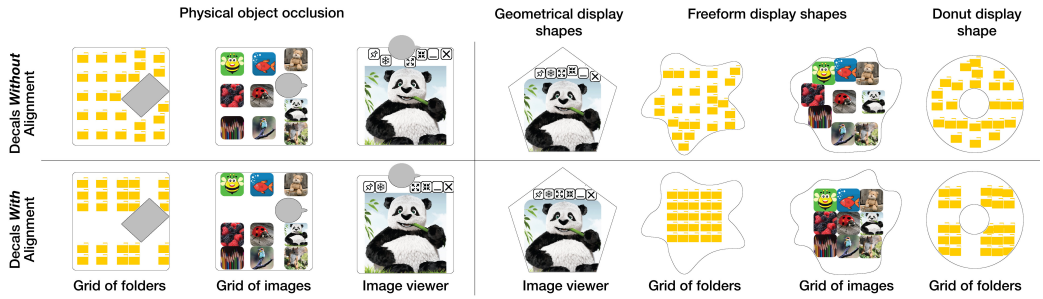


Fig. 8. The use of an Alignment constraint (bottom interfaces) improves layout simplicity compared to using a Minimum distance constraint only (top interfaces).

6.3 Summary

Our results reveal that, overall, the use of decals preserves the interface content and the layout simplicity compared to not deforming the interface or warping the entire interface. These results hold true when the interface is occluded by an object or when the interface adapts to a non-rectangular display area.

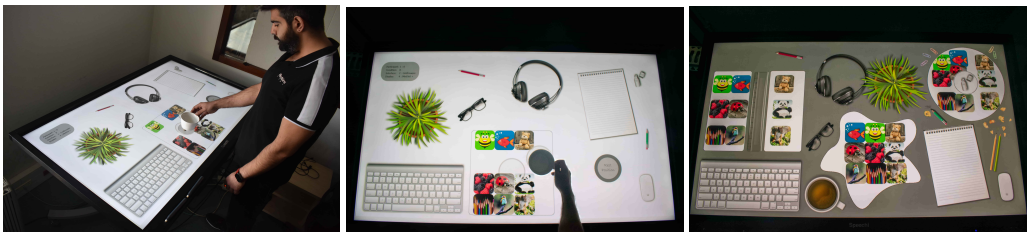
7 USER EVALUATION

We conducted a final study with the goal to collect feedback from users regarding aesthetics, content visibility and usability of our system in a real physical setup.

7.1 Study Description

7.1.1 Study task and instructions. The user study consisted of two parts corresponding to the two disruption conditions: object occlusion and change on the overall display shape. In the first part (Figure 9-a and -b), we asked participants to grasp a physical object (a mug in our study), displace it over two predefined positions on the interface to occlude it and rate the interface on three different metrics: interface aesthetics, content visibility, and content alignment or grouping (depending on the interface). In the second part (Figure 9-c), the participants had to evaluate, using the same metrics, the interfaces presented in three non-rectangular display areas: a freeform shape, a donut (with inner hole) and a 'separated' shape (composed of two parts).

7.1.2 Conditions. For the first part, we tested the same four interfaces as in our previous study, namely the Image Viewer, the 3x3 Grid of Images, the 5x5 Grid of Folders and the Mind Map. We compared two baselines (No_Deformation and Warping) against two Decals conditions (Decals_Min and Decals_Combined). As in the previous study, the Decals_Combined consisted of applying the Alignment constraint for all interfaces except the Mind Map, for which we applied a Maximum Distance constraint (which had the effect of grouping post-its of the same color). For the second



(a) User Evaluation Part 1 (b) User Evaluation Part 1 - top view (c) User Evaluation Part 2 - top view

Fig. 9. Two parts of the study with the effect of Alignment on 3x3 Grid of Images. (a) and (b) Physical Object Occlusion. (c) Static Freeform Display Areas

part of the experiment, we evaluated three interfaces (Image Viewer, 3x3 Grid of Images and 5x5 Grid of Folders). We did not include the Mind Maps as we posited that changing the display shape would have minimal impact on the Maximum distance constraint. We presented each interface on three freeform display areas positioned next to each other (Figure 9-c).

7.1.3 Setup. We chose to conduct the experiment using a tabletop instead of a projection-based system to remove any bias due to inefficient tracking of the physical object. We used a 65 inches multitouch tabletop (1920x1080px), at a height of 80 cm allowing a comfortable grasp of the physical object when standing in front of it. The mug (5cm radius, 6cm height) had three soft pads (1.5cm height) attached to its bottom to be detected as a touch pattern on the touchscreen. To make participants more immersed in the environment, we imitated the presence of other physical objects around our interfaces. Our system was running on a desktop PC (i7-4790K CPU @ 4.00GHz; 32GB; GeForce GTX 980).

7.1.4 Participants. Twelve participants (2 female, 10 male) took part in this study (average age of 27.75, SD = 3.2). Ten of them were local university students (1 Master Degree and 9 PhD students) and 2 were researchers (in physics and computer science).

7.1.5 Study Design. The study followed a 4x4x2 within-subjects design with Interface (Mind Map, Image Viewer, 3x3 Grid of Images and 5x5 Grid of Folders), Display Condition (No_Deformation, Warping, Decals_Min and Decals_Combined) and Type of disruption (Physical object occlusion, Freeform display area) as factors. As said earlier, we did not evaluate the Mind Map interface on the freeform display shape . We counterbalanced the Interfaces and Display Conditions across participants. In total we had 12 trials per Type of disruption.

7.1.6 Data collection and analysis. After each condition, participants used a 5-points Likert scale to rank their preferences on the metrics mentioned above (nice, visible content, alignment and grouping). At the end of the study we measured the usability of each display condition with a System Usability Scale (SUS). To analyze the Likert scale results, we treated data as nominal non-parametric variables. We used a Kruskal-Wallis test to obtain the degree to which one group has higher ranks than the others. After that, we made a post-hoc analysis using a pairwise Mann-Whitney-Wilcoxon test [11].

7.2 Results

For both parts of the study the Kruskal-Wallis test showed a significant difference between display conditions ($p<.001$). In this section, for each metric and condition we will report the percentage

of participants that gave a positive score on the 5-points Likert scale, i.e. that answered *Agree* or *Strongly agree*.

7.2.1 Physical Object Occlusion. Figure 10-left illustrates the results of the Likert scale for the first part of the study, where participants moved a physical object over the interfaces. For the Nice and Visible metrics, the Wilcoxon pairwise comparison shows that there is a significant difference between the Decals conditions and the baseline conditions ($p < .001$ for all). Decals_Min and Decals_Combined were evaluated as nicer and more visible compared to No_Deformation and Warping techniques. The percentage of positive scores for the Nice metric was respectively of 71% for Decals_Min, 85% for Decals_Combined, 25% for No_Deformation and 25% for Warping. The percentage of positives scores for the Visible metric was respectively of 79% for Decals_Min, 88% for Decals_Combined, 6% for No_Deformation and 10% for Warping.

Concerning the content Alignment, results only revealed a significant difference between Decals_Min and the other conditions ($p < .001$ for all). All conditions were perceived to offer a better content alignment (Decals_Combined: 89%, No_Deformation: 83%, and Warping: 89%) than Decals_Min (30%).

Results were similar regarding the Grouping metric for the MindMaps application, where we only found a significant difference between Decals_Min and the other conditions ($p < .001$ for all). All conditions were perceived to offer a better content grouping (Decals_Combined: 100%, No_Deformation: 92%, and Warping: 84%) than Decals_Min (25%).

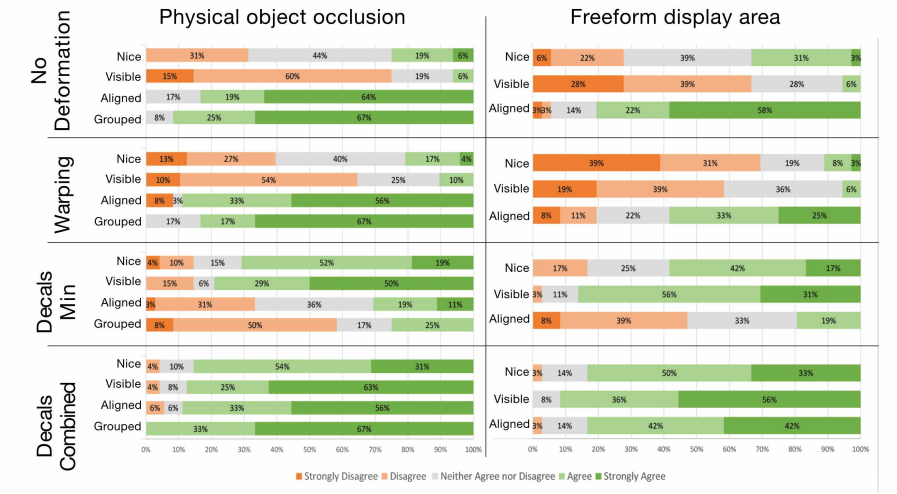


Fig. 10. Likert scale results for the two parts of the study: Physical object occlusion (left) and Static freeform display area (right).

7.2.2 Static Freeform Display Areas. Figure 10-right illustrates the results of the Likert scale for the second part of the study, where interfaces were presented in freeform display areas. For the Nice metric, the Wilcoxon pairwise comparison reveals a significant difference between all conditions (p=0.015 between Decals_Combined and Decals_Min, $p < .001$ for all others) except between No_Deformation and Decals_Min. Overall, Decals_Combined was the nicer condition (83%), followed by Decals_Min and No_Deformation (59% and 31% respectively), with Warping being considered the less nice (11%).

For the Visible metric, the Wilcoxon pairwise comparison reveals a significant difference between all conditions (p=0.048 between Decals_Combined and Decals_Min, $p < .001$ for all others) except between No_Deformation and Warping. Decals_Combined and Decals_Min were considered to offer a better content visibility (92% and 87% respectively) than No_Deformation and Warping (6% each).

Finally, regarding the content alignment, the Wilcoxon pairwise comparison reveals a significant difference between all conditions ($p < .05$ for all) except between No_Deformation and Decals_Combined. Overall, the content alignment was thought to be better when applying No_Deformation and Decals_Combined (80% and 84% respectively) than with Decals_Min (19%) or Warping (58%).

7.2.3 System Usability Scale. Regarding the SUS results, No_Deformation had an average score of 78.1, which corresponds to a system between "Good" and "Excellent", Warping had a score of 60.0, which equals "OK" and Decals_Min had a score of 71.9, which is "Good". Participants evaluated separately the Decals_Combined using the Alignment constraint and the one using the Maximum distance constraint (only on the Mind Maps interface): the Decals_Combined with alignment scored 80.2, whereas Decals_Combined with maximum distance constraint had a score of 84.2, which are both "Excellent".

7.2.4 Qualitative Feedback. The qualitative feedback for each technique helps us to better understand the previous results. Regarding the two baseline conditions, participants preferred No_Deformation over Warping because they felt the interface was more natural with no additional features, even though the content was occluded by the cup. The Warping was considered as cool but not useful or usable. Regarding Decals_Min, participants mentioned that it was hard to track the distribution of the UI elements, especially for the 5x5 Grid of Folders. Considering the Decals_Combined, they noted that it was easy to predict the behaviour of the UI elements.

7.3 Summary

To sum up, our user study confirms that displaying interfaces using our approach offers several benefits compared to the two baselines. With our approach the interface content is nicer and more visible than the baselines, and Decals_Combined preserves content alignment and grouping of elements. Overall, Decals_Combined is the only technique with very positive scores across all the metrics at the same time. This is valid both when occluding the interface with a physical object, as well as when adapting the interface to a freeform display area.

8 DISCUSSION

Our studies validate that adopting an optimization approach to place GUI elements, combined with graphical deformation, helps in solving the issues of content occlusion and the non-rectilinearity of the display area on pervasive interfaces. Our approach improves content visibility and layout properties (i.e. alignment and grouping) compared to previous works warping the interface [8]. We further discuss the possible applications of our approach, its performance, limitations and future directions.

Illustrative applications. Our approach can have many applications beyond our initial augmented classroom user scenario. We showcase in Figure 11 different scenarios where we applied Dynamic Decals: from left to right and top to bottom, casual image gallery on freeform coffee table, magic desk application [3], wall projection combining virtual post-its with physical documents, floor projection of images around users in a museum exhibit, personal augmented workspace and augmented restaurant dishware.

Fig. 11. Six illustrative applications for the Dynamic Decals.

Performances of the layout solver. We designed our prototype using a general-purpose constraint-based solver [33]. Regarding computation times, the optimization is fast-enough to enable interactive framerates, however we believe that it can be improved in several ways, for instance using analytical derivative computations and parallel constraint evaluations. Regarding accuracy, the LM algorithm finds local minima of the energy function, which in some cases might lead to sub-optimal results, e.g. when no valid solution can be found (over-constrained problem: display area too small, conflicting constraints), or several optimal solutions when they exist (under-constrained problem: very large screen, missing constraints). These situations might be detected when designing the interface (to adapt the constraints) and at runtime by analyzing the shape of the display area. Visual hints could also be shown to the user to highlight parts of the GUI where the solver struggles to find a solution, by analysing the optimization residuals.

Limitations. This research is a first step into combining graphical deformations with layout constraints to produce freeform interfaces dynamically adapting to content occlusion and non-rectangular display shapes. As such, we implemented a set of fundamental constraints for freeform GUIs, with layout simplicity in mind. While the SUS results are very positive, given the task of the study (with limited interaction) the evaluated usability results needs to be interpreted with caution. Obviously, other constraints can be added to address other layout properties, such as balance or symmetry [15, 37], or anchors to the initial items location (e.g. minimizing displacement of decals from their original positions). Besides layout properties, constraints could also address other relevant considerations, such as higher-level semantic or task requirements. With more constraints will come the question of the scalability of the optimization. We considered a maximum combination of three constraints at the same time. Obviously, finding an optimal solution to the minimization problem becomes more difficult as we add constraints, and further studies should validate this scalability question. In our studies, we focus on adapting the decal positions. The integration of decal scale and orientation as well as the scale of parts or even the whole interface in the solver constraints is an interesting line of research. This would increase the space of solutions and constraints for the solver. This also requires new studies and considerations that should be the topic of new dedicated investigations.

Future work. Beyond addressing the previous limitations, in the future we plan to investigate other approaches to define constraints: in this work we implemented position-based constraints,

but we could also consider to use other types of external constraints, such as user preference. Another important aspect of GUI interfaces, beyond layout properties, is the fact that GUI elements can be part of a hierarchy. Implementing constraints on a hierarchy of objects poses a series of challenges but can also allow for a more efficient optimization by clustering the problem resolution to a subset of the decals, as already demonstrated for position-based dynamics [35]. We also plan to further investigate the impact of presenting distorted content on Decals: while distortion has been used in HCI for a long time [32], it may be better suited for certain types of content. For instance, presenting distorted text will most certainly affect its readibility: the question on how to best present text on freeform surfaces has already been explored before [44], underlining the need for proper scrolling techniques.

We also plan to ease the integration of our approach into pervasive interfaces. The first step is the creation of an API to program decals and their constraints. With such toolkit, we will implement a demonstrator with a projection-based approach [23, 39, 55] and a depth camera detecting the environment [8]. Depth camera images will then have to be efficiently processed to extract the gamut and its distance field before being used as input in the pipeline presented in this paper. Then, we would like to explore the combination of our constrained-based approach with multimodal interfaces, which are particularly well suited to such pervasive contexts. Applying constraints to a component-based model of the multimodal interaction [42, 43] could decide which modalities to combine according to the interface state.

9 CONCLUSION

In this paper we present a novel approach tackling challenges resulting from content occlusion and non-rectangular display areas on pervasive interfaces. Our approach is composed of two main components: (1) a dynamic layout optimisation model moving the GUI elements according to predefined constraints and user interactions and (2) a deformation model changing the shape of colliding GUI elements. We implement the deformation model with an extension of Implicit Decals [10] defining different decal shapes and new deformers. An online pairwise study reveals that overall the Overlapping-Union deformer is perceived to be nicer and clearer than the others. Our optimisation model is built upon the solver by Mellado et al. [33] combining and minimizing constraints to update the decal positions. We validate the constraints through a study where we implement different interfaces on which we apply content occlusion and display area changes. The results show that compared to the baselines (i.e. warping the entire interface or covering the content), our solution ensures both content preservation and layout simplicity. We confirm these results with a user study showing that, compared to the baselines, users found our system to be aesthetically pleasant while preserving content visibility, layout properties (alignment and grouping) and usability.

10 ACKNOWLEDGMENTS

This work was supported by the ANR JCJC PERFIN grant (ANR-18-CE33-0009). We thank the anonymous reviewers of this article for their relevant critiques and recommendations. Many thanks to all study participants for their valuable time.

REFERENCES

[1] Anand Agarawala and Ravin Balakrishnan. 2006. Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada) *(CHI '06)*. Association for Computing Machinery, New York, NY, USA, 1283–1292. https://doi.org/10.1145/1124772.1124965

[2] Greg J. Badros, Alan Borning, and Peter J. Stuckey. 2001. The Cassowary Linear Arithmetic Constraint Solving Algorithm. *ACM Trans. Comput.-Hum. Interact.* 8, 4 (Dec. 2001), 267–306. https://doi.org/10.1145/504704.504705

[3] Xiaojun Bi, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2011. Magic Desk: Bringing Multi-Touch Surfaces into Desktop Work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC, Canada) *(CHI '11)*. Association for Computing Machinery, New York, NY, USA, 2511–2520. https://doi.org/10.1145/1978942.1979309

[4] James F. Blinn. 1982. A Generalization of Algebraic Surface Drawing. *ACM Trans. Graph.* 1, 3 (July 1982), 235–256. https://doi.org/10.1145/357306.357310

[5] Alan Borning, Richard Lin, and Kim Marriott. 1997. Constraints for the Web. In *Proceedings of the Fifth ACM International Conference on Multimedia* (Seattle, Washington, USA) *(MULTIMEDIA '97)*. Association for Computing Machinery, New York, NY, USA, 173–182. https://doi.org/10.1145/266180.266361

[6] Ralph Allan Bradley and Milton E. Terry. 1952. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika* 39, 3/4 (1952), 324–345. http://www.jstor.org/stable/2334029

[7] Kuan-Ta Chen, Chen-Chi Wu, Yu-Chun Chang, and Chin-Laung Lei. 2009. A Crowdsourceable QoE Evaluation Framework for Multimedia Content. In *Proceedings of the 17th ACM International Conference on Multimedia* (Beijing, China) *(MM '09)*. Association for Computing Machinery, New York, NY, USA, 491–500. https://doi.org/10.1145/1631272.1631339

[8] Daniel Cotting, Markus Gross, and Markus Gross. 2006. Interactive Environment-aware Display Bubbles. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology* (Montreux, Switzerland) *(UIST '06)*. ACM, New York, NY, USA, 245–254. https://doi.org/10.1145/1166253.1166291

[9] Niraj Ramesh Dayama, Kashyap Todi, Taru Saarelainen, and Antti Oulasvirta. 2020. GRIDS: Interactive Layout Design with Integer Programming. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376553

[10] Erwin de Groot, Brian Wyvill, Loïc Barthe, Ahmad Nasri, and Paul Lalonde. 2014. Implicit Decals: Interactive Editing of Repetitive Patterns on Surfaces. *Comput. Graph. Forum* 33, 1 (Feb. 2014), 141–151. https://doi.org/10.1111/cgf.12260

[11] Joost de Winter and Dimitra Dodou. 2010. Five-Point Likert Items: t Test Versus Mann–Whitney–Wilcoxon. *Practical Assessment, Research and Evaluation* 15 (01 2010).

[12] Peitong Duan, Casimir Wierzynski, and Lama Nachman. 2020. Optimizing User Interface Layouts via Gradient Descent. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3313831.3376589

[13] Barrett Ens, Benjamin Bach, Maxime Cordeil, Ulrich Engelke, Marcos Serrano, Wesley Willett, Arnaud Prouzeau, Christoph Anthes, Wolfgang Büschel, Cody Dunne, Tim Dwyer, Jens Grubert, Jason H. Haga, Nurit Kirshenbaum, Dylan Kobayashi, Tica Lin, Monsurat Olaosebikan, Fabian Pointecker, David Saffo, Nazmus Saquib, Dieter Schmalstieg, Danielle Albers Szafir, Matt Whitlock, and Yalong Yang. 2021. Grand Challenges in Immersive Analytics. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 459, 17 pages. https://doi.org/10.1145/3411764.3446866

[14] Genki Furumi, Daisuke Sakamoto, and Takeo Igarashi. 2012. SnapRail: A Tabletop User Interface Widget for Addressing Occlusion by Physical Objects. In *Proceedings of the 2012 ACM International Conference on Interactive Tabletops and Surfaces* (Cambridge, Massachusetts, USA) *(ITS '12)*. Association for Computing Machinery, New York, NY, USA, 193–196. https://doi.org/10.1145/2396636.2396666

[15] Wilbert O. Galitz. 2007. *The Essential Guide to User Interface Design: An Introduction to GUI Design Principles and Techniques*. John Wiley & Sons, Inc., USA.

[16] Michael Gleicher. 1992. Briar: A Constraint-Based Drawing Program. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Monterey, California, USA) *(CHI '92)*. Association for Computing Machinery, New York, NY, USA, 661–662. https://doi.org/10.1145/142750.143074

[17] Michael Gleicher. 1992. Integrating constraints and direct manipulation. In *Proceedings of the 1992 symposium on Interactive 3D graphics - SI3D '92*. ACM Press, New York, New York, USA, 171–174. https://doi.org/10.1145/147156.147194

[18] Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org.

[19] Alina Hang, Enrico Rukzio, and Andrew Greaves. 2008. Projector Phone: A Study of Using Mobile Phones with Integrated Projector for Interaction with Maps. In *Proceedings of the 10th International Conference on Human Computer Interaction with Mobile Devices and Services* (Amsterdam, The Netherlands) *(MobileHCI '08)*. Association for Computing Machinery, New York, NY, USA, 207–216. https://doi.org/10.1145/1409240.1409263

[20] Hiroshi Hosobe. 2001. A modular geometric constraint solver for user interface applications. In *Proceedings of the 14th annual ACM symposium on User interface software and technology - UIST '01*. ACM Press, New York, New York, USA, 91. https://doi.org/10.1145/502348.502362

[21] Scott E. Hudson and Ian Smith. 1996. Ultra-Lightweight Constraints. In *Proceedings of the 9th Annual ACM Symposium on User Interface Software and Technology* (Seattle, Washington, USA) *(UIST '96)*. Association for Computing Machinery,

New York, NY, USA, 147–155. https://doi.org/10.1145/237091.237112

[22] Waqas Javed, KyungTae Kim, Sohaib Ghani, and Niklas Elmqvist. 2011. Evaluating Physical/Virtual Occlusion Management Techniques for Horizontal Displays. In *Human-Computer Interaction – INTERACT 2011*, Pedro Campos, Nicholas Graham, Joaquim Jorge, Nuno Nunes, Philippe Palanque, and Marco Winckler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 391–408.

[23] Tyler Johnson and Henry Fuchs. 2007. Real-Time Projector Tracking on Complex Geometry Using Ordinary Imagery. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*. 1–8. https://doi.org/10.1109/CVPR.2007.383460

[24] Brett Jones, Hrvoje Benko, Eyal Ofek, and Andrew Wilson. 2013. IllumiRoom: peripheral projected illusions for interactive experiences. https://doi.org/10.1145/2503368.2503375

[25] Brett Jones, Rajinder Sodhi, Michael Murdock, Ravish Mehra, Hrvoje Benko, Andrew Wilson, Eyal Ofek, Blair MacIntyre, Nikunj Raghuvanshi, and Lior Shapira. 2014. RoomAlive: Magical Experiences Enabled by Scalable, Adaptive Projector-Camera Units. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology* (Honolulu, Hawaii, USA) *(UIST '14)*. Association for Computing Machinery, New York, NY, USA, 637–644. https://doi.org/10.1145/2642918.2647383

[26] Brett R. Jones, Rajinder Sodhi, Roy H. Campbell, Guy Garnett, and Brian P. Bailey. 2010. Build your world and play in it: Interacting with surface particles on complex objects. In *2010 IEEE International Symposium on Mixed and Augmented Reality*. 165–174. https://doi.org/10.1109/ISMAR.2010.5643566

[27] Mohammadreza Khalilbeigi, Jürgen Steimle, Jan Riemann, Niloofar Dezfuli, Max Mühlhäuser, and James D. Hollan. 2013. ObjecTop: Occlusion Awareness of Physical Objects on Interactive Tabletops. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces* (St. Andrews, Scotland, United Kingdom) *(ITS '13)*. Association for Computing Machinery, New York, NY, USA, 255–264. https://doi.org/10.1145/2512349.2512806

[28] Masaaki Kurosu and Kaori Kashimura. 1995. Apparent Usability vs. Inherent Usability: Experimental Analysis on the Determinants of the Apparent Usability. In *Conference Companion on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '95)*. Association for Computing Machinery, New York, NY, USA, 292–293. https://doi.org/10.1145/223355.223680

[29] Ricardo Langner, John Brosz, Raimund Dachselt, and Sheelagh Carpendale. 2010. PhysicsBox: Playful Educational Tabletop Games. In *ACM International Conference on Interactive Tabletops and Surfaces* (Saarbrücken, Germany) *(ITS '10)*. Association for Computing Machinery, New York, NY, USA, 273–274. https://doi.org/10.1145/1936652.1936712

[30] Talia Lavie and Noam Tractinsky. 2004. Assessing dimensions of perceived visual aesthetics of web sites. *International Journal of Human-Computer Studies* 60, 3 (2004), 269 – 298. https://doi.org/10.1016/j.ijhcs.2003.09.002

[31] Johnny C. Lee, Scott E. Hudson, Jay W. Summet, and Paul H. Dietz. 2005. Moveable Interactive Projected Displays Using Projector Based Tracking. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology* (Seattle, WA, USA) *(UIST '05)*. Association for Computing Machinery, New York, NY, USA, 63–72. https://doi.org/10.1145/1095034.1095045

[32] Y. K. Leung and M. D. Apperley. 1994. A Review and Taxonomy of Distortion-Oriented Presentation Techniques. *ACM Trans. Comput.-Hum. Interact.* 1, 2 (June 1994), 126–160. https://doi.org/10.1145/180171.180173

[33] Nicolas Mellado, David Vanderhaeghe, Charlotte Hoarau, Sidonie Christophe, Mathieu Brédif, and Loic Barthe. 2017. Constrained Palette-Space Exploration. *ACM Trans. Graph.* 36, 4, Article 60 (July 2017), 14 pages. https://doi.org/10.1145/3072959.3073650

[34] Jan Meskens, Jo Vermeulen, Kris Luyten, and Karin Coninx. 2008. Gummy for Multi-Platform User Interface Designs: Shape Me, Multiply Me, Fix Me, Use Me. In *Proceedings of the Working Conference on Advanced Visual Interfaces* (Napoli, Italy) *(AVI '08)*. Association for Computing Machinery, New York, NY, USA, 233–240. https://doi.org/10.1145/1385569.1385607

[35] M. Müller. 2008. Hierarchical Position Based Dynamics. In *VRIPHYS*.

[36] Miguel A. Nacenta, Satoshi Sakurai, Tokuo Yamaguchi, Yohei Miki, Yuichi Itoh, Yoshifumi Kitamura, Sriram Subramanian, and Carl Gutwin. 2007. E-Conic: A Perspective-Aware Interface for Multi-Display Environments. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (Newport, Rhode Island, USA) *(UIST '07)*. Association for Computing Machinery, New York, NY, USA, 279–288. https://doi.org/10.1145/1294211.1294260

[37] David Chek Ling Ngo, Lian Seng Teo, and John G. Byrne. 2003. Modelling Interface Aesthetics. *Inf. Sci.* 152, 1 (June 2003), 25–46. https://doi.org/10.1016/S0020-0255(02)00404-8

[38] Stephen Oney, Brad Myers, and Joel Brandt. 2012. ConstraintJS. In *Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST '12*. ACM Press, New York, New York, USA, 229. https://doi.org/10.1145/2380116.2380146

[39] Ramesh Raskar, Michael S. Brown, Ruigang Yang, Wei-Chao Chen, Greg Welch, Herman Towles, Brent Seales, and Henry Fuchs. 1999. Multi-Projector Displays Using Camera-Based Registration. In *Proceedings of the Conference on Visualization '99: Celebrating Ten Years* (San Francisco, California, USA) *(VIS '99)*. IEEE Computer Society Press, Washington, DC, USA, 161–168.

[40] Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin, and Henry Fuchs. 1998. The Office of the Future: A Unified Approach to Image-Based Modeling and Spatially Immersive Displays. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98)*. Association for Computing Machinery, New York, NY, USA, 179–188. https://doi.org/10.1145/280814.280861

[41] Jan Riemann, Martin Schmitz, Alexander Hendrich, and Max Mühlhäuser. 2018. FlowPut: Environment-Aware Interactivity for Tangible 3D Objects. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 1, Article 31 (March 2018), 23 pages. https://doi.org/10.1145/3191763

[42] Marcos Serrano and Laurence Nigay. 2009. Temporal Aspects of CARE-Based Multimodal Fusion: From a Fusion Mechanism to Composition Components and WoZ Components. In *Proceedings of the 2009 International Conference on Multimodal Interfaces* (Cambridge, Massachusetts, USA) *(ICMI-MLMI '09)*. Association for Computing Machinery, New York, NY, USA, 177–184. https://doi.org/10.1145/1647314.1647346

[43] Marcos Serrano and Laurence Nigay. 2010. A wizard of oz component-based approach for rapidly prototyping and testing input multimodal interfaces. *Journal on Multimodal User Interfaces, Springer Publ.* 3, 3 (2010), 215–225. http://www.springerlink.com/content/f116502x844117t6

[44] Marcos Serrano, Anne Roudaut, and Pourang Irani. 2016. Investigating Text Legibility on Non-Rectangular Displays. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. ACM, New York, NY, USA, 498–508. https://doi.org/10.1145/2858036.2858057

[45] Marcos Serrano, Anne Roudaut, and Pourang Irani. 2017. Visual Composition of Graphical Elements on Non-Rectangular Displays. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. ACM, New York, NY, USA, 4405–4416. https://doi.org/10.1145/3025453.3025677

[46] Florine Simon, Anne Roudaut, Pourang Irani, and Marcos Serrano. 2019. Finding Information on Non-Rectangular Interfaces. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. ACM, New York, NY, USA, Article 102, 8 pages. https://doi.org/10.1145/3290605.3300332

[47] S. Sistare. 1991. Graphical Interaction Techniques in Constraint-Based Geometric Modeling. In *Proceedings of Graphics Interface '91* (Calgary, Alberta, Canada) *(GI '91)*. Canadian Man-Computer Communications Society, Toronto, Ontario, Canada, 85–92. http://graphicsinterface.org/wp-content/uploads/gi1991-12.pdf

[48] Wolfgang Stuerzlinger, Olivier Chapuis, Dusty Phillips, and Nicolas Roussel. 2006. User Interface FaçAdes: Towards Fully Adaptable User Interfaces. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology* (Montreux, Switzerland) *(UIST '06)*. Association for Computing Machinery, New York, NY, USA, 309–318. https://doi.org/10.1145/1166253.1166301

[49] Ivan E. Sutherland. 1964. Sketch Pad a Man-Machine Graphical Communication System. In *Proceedings of the SHARE Design Automation Workshop (DAC '64)*. Association for Computing Machinery, New York, NY, USA, 6.329–6.346. https://doi.org/10.1145/800265.810742

[50] Amanda Swearngin, Chenglong Wang, Alannah Oleson, James Fogarty, and Amy J. Ko. 2020. Scout: Rapid Exploration of Interface Layout Alternatives through High-Level Design Constraints. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376593

[51] Aurelien Tabard, Simon Gurn, Andreas Butz, and Jakob Bardram. 2013. A Case Study of Object and Occlusion Management on the ELabBench, a Mixed Physical/Digital Tabletop. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces* (St. Andrews, Scotland, United Kingdom) *(ITS '13)*. Association for Computing Machinery, New York, NY, USA, 251–254. https://doi.org/10.1145/2512349.2512794

[52] Daniel Vogel and Ravin Balakrishnan. 2010. Occlusion-Aware Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '10)*. Association for Computing Machinery, New York, NY, USA, 263–272. https://doi.org/10.1145/1753326.1753365

[53] Manuela Waldner, Raphael Grasset, Markus Steinberger, and Dieter Schmalstieg. 2011. Display-Adaptive Window Management for Irregular Surfaces. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces* (Kobe, Japan) *(ITS '11)*. Association for Computing Machinery, New York, NY, USA, 222–231. https://doi.org/10.1145/2076354.2076394

[54] Andrew D. Wilson, Shahram Izadi, Otmar Hilliges, Armando Garcia-Mendoza, and David Kirk. 2008. Bringing Physics to the Surface. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology* (Monterey, CA, USA) *(UIST '08)*. Association for Computing Machinery, New York, NY, USA, 67–76. https://doi.org/10.1145/1449715.1449728

[55] Raphael Wimmer, Fabian Hennecke, Florian Schulz, Sebastian Boring, Andreas Butz, and Heinrich Hußmann. 2010. Curve: Revisiting the Digital Desk. In *Proceedings of the 6th Nordic Conference on Human-Computer Interaction: Extending Boundaries* (Reykjavik, Iceland) *(NordiCHI '10)*. Association for Computing Machinery, New York, NY, USA, 561–570. https://doi.org/10.1145/1868914.1868977

[56] Geoff Wyvill, Craig McPheeters, and Brian Wyvill. 1986. Data Structure for Soft Objects. *The Visual Computer* 2, 4 (February 1986), 227–234.

11 APPENDICES

A TECHNICAL BACKGROUND: IMPLICIT DECALS

A decal position is set by its center $\mathbf{p}_i$, where $f_i(\mathbf{p}_i) = 1$ and its orientation is given by a local frame $(\mathbf{s_i}, \mathbf{t_i})$. In the implicit decal formulation, the field function f_i is defined by composing two components. A distance field $d_i : \mathbb{R}^2 \rightarrow [0,1]$, providing a distance, eventually anisotropic (i.e. varying differently in different directions), from any point $\mathbf{p}$ in space and the decal center $\mathbf{p}_i$, and a compactly supported fall off function $g : [0,1] \rightarrow [0,1]$ such that $f_i(\mathbf{p}) = g(d_i(\mathbf{p}))$ exhibits the aforementioned properties. In de Groot et al. [10] implicit decals, they are defined as follows:

$$d_i(\mathbf{p}) = \frac{\|\mathbf{p} - \mathbf{p}_i\|}{b_i(\mathbf{p})}, \quad \text{and} \quad g(d) = \begin{cases} (1-d^2)^3 & \text{if } d \leq 1 \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where the function $b_i(\mathbf{p})$ defines the decal shape by computing the Euclidean distance from $\mathbf{p}_i$ and the decal influence limit in the radial direction $\mathbf{p} - \mathbf{p}_i$ in the case of an anysotropic function b_i defining a square decal. For a circular decal of radius R_i, we would just set $b_i(\mathbf{p}) = R_i = C^{te}\ \forall \mathbf{p}$.

A fundamental advantage of the field function formulation is to enable the automatic generation of contact deformations between n decals by combining their field functions f_i with a composition operator $l : [0,1]^n \rightarrow [0,1]$ and define a unique field function $F : \mathbb{R}^2 \rightarrow [0,1]$ as $F(\mathbf{p}) = l(f_1(\mathbf{p}), \ldots, f_n(\mathbf{p}))$, which includes all decals deformed. De Groot's contact formulation is defined as:

$$F(\mathbf{p}) = \frac{1}{2} + \left(f_k(\mathbf{p}) - \frac{1}{2}\right) \prod_{j \neq k} h(f_j(\mathbf{p}), f_k(\mathbf{p})) \quad \text{where} \quad k = \underset{i}{\text{argmax}}\,(f_i(\mathbf{p})) \tag{2}$$

and

$$h(x,y) = \begin{cases} 1 - \left(\frac{x+y-1}{2y-1}\right)^{\frac{1}{1-y}} & \text{if } x + y > 1 \\ 1 & \text{otherwise.} \end{cases} \tag{3}$$

To summarize, a decal D_i is controlled by its center position $\mathbf{p}_i$, its local frame $(\mathbf{s_i}, \mathbf{t_i})$ and its shape function $b_i(\mathbf{p})$.

Finally, the key idea of the implicit decal representation is to define its parameterization $(u_i(\mathbf{p}), v_i(\mathbf{p}))$ using the inverse function $g^{-1}(F(\mathbf{p}))$. It returns the distance $d_i(\mathbf{p})$ in an isolated decal D_i, and automatically transfers the decal deformations on the parameterization when they collide. De Groot et al. introduced the parameterization in polar coordinates. In our case, the cartesian (u, v) coordinates defined in $[0,1]^2$ are more convenient and we derive them as:

$$\begin{cases} u_i(\mathbf{p}) = \frac{1}{2}\left(\frac{\tilde{b_i}(\mathbf{p}) \cdot s_i}{b_i(\mathbf{p_i} + \mathbf{s_i}) \cdot g^{-1}\left(\frac{1}{2}\right)} + 1\right) \\ v_i(\mathbf{p}) = \frac{1}{2}\left(\frac{\tilde{b_i}(\mathbf{p}) \cdot t_i}{b_i(\mathbf{p_i} + \mathbf{t_i}) \cdot g^{-1}\left(\frac{1}{2}\right)} + 1\right) \end{cases} \quad \text{with} \quad \tilde{b_i}(\mathbf{p}) = g^{-1}(F(\mathbf{p})) \cdot b_i(\mathbf{p}) \cdot \frac{\mathbf{p} - \mathbf{p}_i}{\|\mathbf{p} - \mathbf{p}_i\|}. \tag{4}$$

B DECAL SHAPE FUNCTION EQUATIONS

Following this definition of implicit decals, we first present functions b_i producing our decal shapes. We then introduce a new technical contribution: the use of field function deformers with their formulation. With deformers, each decal is defined by its own field function and thus it also

includes its own deformed parameterization with which its own content is mapped. This is both more efficient and not subject to any indeterminacy when complex deformations are used. This is implemented by replacing $g^{-1}(F(\mathbf{p}))$ by $g^{-1}(f_i^k(\mathbf{p}))$ in Equation 4, which yields the following formulation for $\tilde{b_i}(\mathbf{p})$:

$$\tilde{b_i}(\mathbf{p}) = g^{-1}(f_i^k(\mathbf{p})) \cdot b_i(\mathbf{p}) \cdot \frac{\mathbf{p} - \mathbf{p}_i}{\|\mathbf{p} - \mathbf{p}_i\|} , \tag{5}$$

A square decal is defined as a square at the decal boundary and at its influence limit. Squares are interpolated with a circle to smooth the in-between field (see Figure 2-a) and thus the contact deformations. A square of size R_i (i.e. an edge length of $2R_i$), is defined by the shape function b_i^s as follows:

$$b_i^s(\mathbf{p}) = t^2 \bar{b_i^s}(\mathbf{p}) + (1 - t^2) R_i , \quad \text{with} \quad t = \begin{cases} \frac{s}{r_i} & \text{if } s \leq r_i \\ \frac{R_i - s}{R_i - r_i} & \text{if } r_i < s \leq \frac{r_i + R_i}{2} \\ \frac{r_i - s}{r_i - R_i} & \text{if } \frac{r_i + R_i}{2} < s \leq R_i \end{cases} \tag{6}$$

where

$$r_i = g^{-1}\left(\tfrac{1}{2}\right) , s = \max(|\mathbf{s_i} \cdot (\mathbf{p} - \mathbf{p}_i)|, |\mathbf{t_i} \cdot (\mathbf{p} - \mathbf{p}_i)|) , \quad \text{and} \quad \bar{b_i^s}(\mathbf{p}) = \frac{R_i \|\mathbf{p} - \mathbf{p}_i\|}{s} . \tag{7}$$

A square with rounded corners decal is represented by squares whose corners are defined by a quarter of a circle. For a square of size R_i rounded at an angle θ_i (see Figure 2-b), the shape function b_i^{rs} is defined as follows. We first define:

$$s_\mathbf{p} = |\mathbf{s_i} \cdot (\mathbf{p} - \mathbf{p}_i)| \text{ and } t_\mathbf{p} = |\mathbf{t_i} \cdot (\mathbf{p} - \mathbf{p}_i)| . \tag{8}$$

Then, if $(t_\mathbf{p} \leq s_\mathbf{p} \tan\theta_i)$ or $(s_\mathbf{p} \leq t_\mathbf{p} \tan\theta_i)$, the decal is a square and:

$$b_i^{rs}(\mathbf{p}) = \frac{R_i \|\mathbf{p} - \mathbf{p}_i\|}{\max(s_\mathbf{p}, t_\mathbf{p})} , \tag{9}$$

Otherwise, we compute the intersection of a ray starting from $\mathbf{p}_i$ in the direction of $\mathbf{p}$ and the arc of circle defining the rounded corner. To do so, we first compute:

$$\mathbf{r} = \frac{1}{\|\mathbf{p} - \mathbf{p}_i\|} \begin{pmatrix} s_\mathbf{p} \\ t_\mathbf{p} \end{pmatrix} \text{ and } \mathbf{o} \begin{pmatrix} R_i \tan\theta_i \\ R_i \tan\theta_i \end{pmatrix} , \tag{10}$$

then:

$$a = \mathbf{r}^2 , \; b = -2\mathbf{o}.\mathbf{r} , \; c = \mathbf{o}^2 - (R_i(1 - \tan\theta_i))^2 , \tag{11}$$

and finally:

$$b_i^{rs}(\mathbf{p}) = \frac{-b + \sqrt{b^2 - 4ac}}{2a} . \tag{12}$$

A circular decal is easily defined with a constant shape function set with its size: $b_i^c(\mathbf{p}) = R_i$ $\forall$ $\mathbf{p}$, as depicted in Figure 2-c.

C CONSTRAINTS

Following similar notations than Mellado et al. [33], we denote $\mathbf{X} = \{\mathbf{x}_i \in \mathcal{S}, i = 1..k_d\}$ the set of decal positions, $\mathcal{S}$ the current display area, and $\Pi(\mathbf{X})$ the *conformity*, which measures how much the decal positions $\mathbf{X}$ respect the desired properties. The optimized positions of the decals $\mathbf{X}^*$ are computed as follows:

$$\mathbf{X}^* = \underset{\mathbf{X}}{\operatorname{argmin}}\, \Pi(\mathbf{X}), \qquad \text{with} \qquad \Pi^X = \Pi_\mathcal{S}^X(\mathbf{X}) + \Pi_{\text{constr}}(\mathbf{X}). \tag{13}$$

The goal of the first term is to penalize decals lying outside of the display area, and the goal of the second term is to penalize decals that do not respect prescribed constraints. They are defined as follows:

$$\Pi_{\mathcal{S}}(\mathbf{X}) = \sum_{i=1}^{k_d} c_{\text{gam}}(\mathbf{x}_i, r_i)^2, \quad \text{with} \quad \Pi_{\text{constr}}(\mathbf{X}) = \sum_{j=1}^{k_c} \left(c_j(\mathbf{X}_j)\right)^2. \tag{14}$$

C.1 Gamut constraint

In order to ensure fast evaluation with dynamic display area and decals with different radii, we compute a signed distance field $d_{\mathcal{S}}(\mathbf{x})$ to the display area boundary (negative inside, positive outside), and define the gamut constraint $c_{\text{gam}}(\mathbf{x}, r)$ as follows:

$$c_{\text{gam}}(\mathbf{x}, r) = \begin{cases} e_{\text{step}} + d_{\mathcal{S}}(\mathbf{x}) - r & \text{if } d_{\mathcal{S}}(\mathbf{x}) - r > 0 \\ 0 & \text{otherwise,} \end{cases} \quad \text{and} \quad d_{\mathcal{S}}(\mathbf{x}) = \begin{cases} \left|\mathbf{x} - \text{proj}_{\mathcal{S}}(\mathbf{x})\right|_2 & \text{if } \mathbf{x} \notin \mathcal{S} \\ - \left|\mathbf{x} - \text{proj}_{\mathcal{S}}(\mathbf{x})\right|_2 & \text{otherwise.} \end{cases} \tag{15}$$

The operator $\text{proj}_{\mathcal{S}}(\mathbf{x})$ projects $\mathbf{x}$ on the boundary of the display area $\mathcal{S}$. e_{step} is a constant error term added to out-of-gamut positions, we used $e_{\text{step}} = 10$ for all our experiments.

C.2 Distance constraints

The **minimum distance** constraint penalizes overlapping decals. In order to ensure fast evaluation of this constraint, we assume without loss of generality that the decal's shape can be approximated by axis aligned square boxes of size r_i. Thus, overlap detection and cost can be efficiently computed using the L_1 distance between the decal's center:

$$c_{\text{mind}}(\mathbf{x}_a, \mathbf{x}_b) = min\left(0, |\mathbf{x}_a - \mathbf{x}_b|_1 - (r_a + r_b)\right). \tag{16}$$

The **maximum distance** constraint ensures that decals belonging to the same group remain in close proximity to each other. In contrast to c_{mind}, we look for proximity and do not want to favor distances in x and y directions (see Figure 3-D). Thus, we penalize decals when their L2 distance goes higher than a given threshold d_{max}:

$$c_{\text{maxd}}(\mathbf{x}_a, \mathbf{x}_b) = max\left(0, |\mathbf{x}_a - \mathbf{x}_b|_2 - d_{\text{max}}\right). \tag{17}$$

C.3 Alignment constraint

The cost is computed w.r.t. the distance between the anchor line l and each decal $\mathbf{x}_k$ belonging to a group denoted $\mathbf{X}$, as:

$$c_{\text{anchor}}(\mathbf{X}) = \sum_k \left|\text{proj}_l(\mathbf{X}_k) - \mathbf{X}_k\right|_2. \tag{18}$$

D DECAL DEFORMER EQUATIONS

For all deformer equations, we set k as:

$$k = \underset{l}{\text{argmax}}\left(f_l\right), \; l = 1..n. \tag{19}$$

The Overlapping-Union deformer is built to keep unchanged the field function f_i outside all other decal boundaries, and where $f_i \geq f_j$ ($j = 1..n$ and $j \neq i$). Where $f_i < f_j$, f_i is set to a value in $[0, \frac{1}{2}]$ to adequately define the outer area of the deformed decal. We thus propose the following formulation:

$$k = \underset{l}{\text{argmax}}\,(f_l)\,,\ l = 1..n\,,\ \text{ and } z^{ou}(f_i,\ldots,f_j,\ldots),\ j \neq i, = \begin{cases} f_i & \text{if } i = k \text{ or } (f_i \leq \frac{1}{2} \text{ and } f_k \leq \frac{1}{2}) \\ \text{otherwise:} & \\ f_i - f_k + \frac{1}{2} & \text{if } f_i - f_k + \frac{1}{2} > 0 \\ 0 & \text{otherwise .} \end{cases} \tag{20}$$

The Squashing-Union deformer is very similar to the overlapping-union (Equation 20). The difference is in the inner decal part where we want to squash the decal content. This is done using an adaptation of de Groot contact composition operator [10] as follows:

$$z^{su}(f_i,\ldots,f_j,\ldots),\ j \neq i, = \begin{cases} H(f_i,f_k) & \text{if } i = k \\ f_i & \text{if } f_i \leq \frac{1}{2} \text{ and } f_k \leq \frac{1}{2} \\ \text{otherwise:} & \\ f_i - f_k + \frac{1}{2} & \text{if } f_i - f_k + \frac{1}{2} > 0 \\ 0 & \text{otherwise ,} \end{cases} \tag{21}$$

where

$$H(x,y) = \tfrac{1}{2} + (x - \tfrac{1}{2})h(x,y)\,, \tag{22}$$

and $h(x,y)$ is defined in Equation 3.

The Overlapping-Blending deformer has the same inner behavior as the overlapping-union (Equation 20) except for a small inner band of width e close to the decal boundary that is mapped on the blending boundary. This enables the generation of the black contouring of the blended decals, as can be seen in Figure 4-right. The blending is a well know composition operator for combining field functions and we propose to model it using its simplest formulation, i.e. by summing the combined field functions [4, 56]. In our case, the blending is only located in the small inner band of width e and in the outside decal areas. We thus end up with the following formulation of our deformer:

$$z^{ob}(f_i,\ldots,f_j,\ldots),\ j \neq i, = \begin{cases} f_i & \text{if } i = k \text{ and } f_i > \frac{1}{2} + e \\ \text{otherwise:} & \\ \sum_{l=1}^{n} f_l & \text{if } \sum_{l=1}^{n} f_l \leq \frac{1}{2} + e \text{ and } i = k \\ \text{otherwise:} & \\ f_i - f_k + \frac{1}{2} & \text{if } 0 < f_i - f_k + \frac{1}{2} < \frac{1}{2} \\ 0 & \text{if } f_i - f_k + \frac{1}{2} \leq 0 \\ \frac{1}{2} + e & \text{otherwise .} \end{cases} \tag{23}$$

The Squashing-Blending deformer, follows the formulation of the overlapping-blending deformer (Equation 23). We just have to add the squashing deformation inside the decal. It is thus

defined as follows:

$$z^{sb}(f_i,\ldots,f_j,\ldots),\ j\neq i,\ =\begin{cases} H(f_i,f_k) & \text{if } i=k \text{ and } f_i>\frac{1}{2}+e \\ \text{otherwise:} & \\ \sum_{l=1}^{n} f_l & \text{if } \sum_{l=1}^{n} f_l \leq \frac{1}{2}+e \text{ and } i=k \\ \text{otherwise:} & \\ f_i-f_k+\frac{1}{2} & \text{if } 0<f_i-f_k+\frac{1}{2}<\frac{1}{2} \\ 0 & \text{if } f_i-f_k+\frac{1}{2}\leq 0 \\ \frac{1}{2}+e & \text{otherwise .} \end{cases} \tag{24}$$

where $H(x,y)$ is defined in Equation 22.

The Squashing-Rigid deformer is used to squash a decal field f_1 against a rigid display field f_2 boundary. It is thus a binary deformer $z^{sr}(f_1,f_2)$. It is built such that where the decal and the display interpenetrate and outside the display boundary, the outer display is removed from the decal. Inside, an inside squashed field is built along the display boundary. This is implemented as follows:

$$z^{sr}(f_1,f_2)=\begin{cases} f_1 & \text{if } f_1<\frac{1}{2} \text{ and } f_2\leq\frac{1}{2} \\ 1-f_2 & \text{if } f_1\geq\frac{1}{2} \text{ and } f_2\geq\frac{1}{2} \\ \max\left(0,\frac{1}{2}-\sqrt{\left(\frac{1}{2}-f_1\right)^2+\left(\frac{1}{2}-f_2\right)^2}\right) & \text{if } f_1<\frac{1}{2} \text{ and } f_2>\frac{1}{2} \\ (1-t(f_2))f_1+\frac{t(f_2)}{2} & \text{otherwise ,} \end{cases} \tag{25}$$

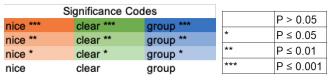
where

$$t(x)=(2x)^4 \tag{26}$$

The Overlapping-Rigid deformer is very similar to the Squashing-Rigid deformer (Equation 25). The only difference is that the inner decal content is hided by the gamut rather than being squashed. Its equation is thus:

$$z^{or}(f_1,f_2)=\begin{cases} f_1 & \text{if } f_1<\frac{1}{2} \text{ and } f_2\leq\frac{1}{2} \\ 1-f_2 & \text{if } f_1\geq\frac{1}{2} \text{ and } f_2\geq\frac{1}{2} \\ \max\left(0,\frac{1}{2}-\sqrt{\left(\frac{1}{2}-f_1\right)^2+\left(\frac{1}{2}-f_2\right)^2}\right) & \text{if } f_1<\frac{1}{2} \text{ and } f_2>\frac{1}{2} \\ f_1 & \text{otherwise .} \end{cases} \tag{27}$$

E ESTIMATES AND P-VALUES FOR THE SURVEY PAIRWISE COMPARISONS



(a) (b)

Fig. 12. (a) Color encoding of the p-values in the estimate tables for the three metrics: nice, clear and grouping. (b) Numerical ranges of p-values

Received July 2021; revised September 2021; accepted September 2021

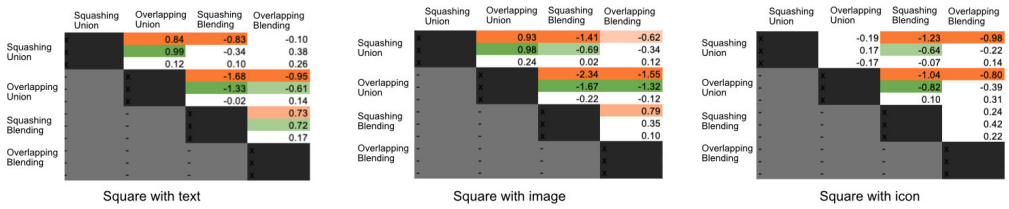


Fig. 13. Tables with the estimate values for the Square decal shape. Significance is color encoded.

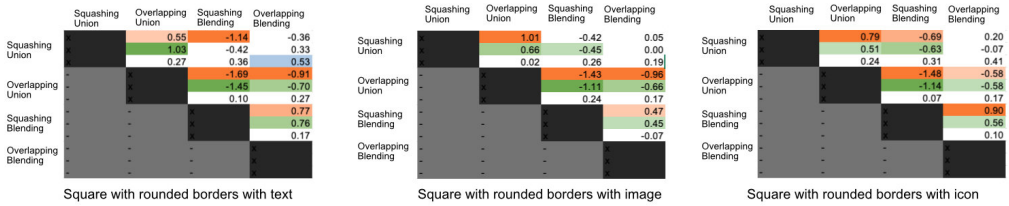


Fig. 14. Tables with the estimate values for the Square with rounded borders decal shape. Significance is color encoded.

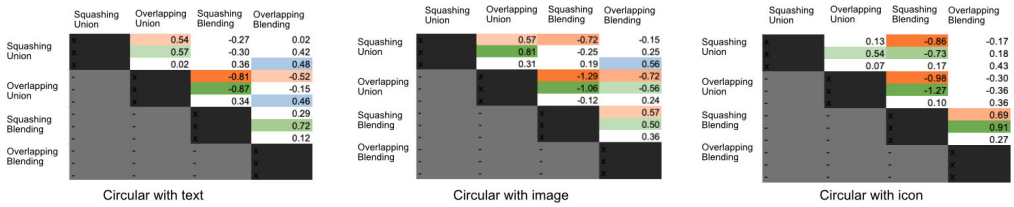


Fig. 15. Tables with the estimate values for the Circular decal shape. Significance is color encoded.