Efficient Depth-aware Image Deformation Adaptation for Curved Screen Displays*

Shao-Ping Lu[1,2], Ruxandra-Marina Florea[1,2], Pablo Cesar[3,4], Peter Schelkens[1,2], Adrian Munteanu[1,2]
[1]Vrije Universiteit Brussel (VUB), Department of Electronics and Informatics (ETRO), Pleinlaan 2, Brussels, Belgium
[2]IMEC, Kapeldreef 75, B-3001 Leuven, Belgium
[3]Centrum Wiskunde & Informatica (CWI), Amsterdam, The Netherlands
[4]Delft University of Technology, Delft, The Netherlands
{slu, fruxandr, pschelke, acmuntea}@etrovub.be p.s.cesar@cwi.nl

ABSTRACT

The curved screen has attracted considerable attentions in recent years, since it enables to enlarge the view angle and to enhance the immersive perception for users. However, existing curved surface projections are frequently prone to geometric distortion or loss of content. This paper presents a *content-aware* and *depth-aware* image adaptation solution for curved displays. An efficient optimization approach of image deformation is proposed to preserve local scene content and to minimize scene geometry distortion. To follow the original 3D perception of objects in different depth layers, the depth information is re-mapped for individual content scaling. Objective evaluation results reveal that our approach can effectively preserve foreground objects. We also perform a subjective evaluation of the proposed solution, and compare it to two alternative mapping methods, which are tested on different curvatures on both a traditional screen and an ad-hoc curvature-controllable curved display. Experimental results demonstrate that our approach outperforms other existing mapping methods for immersive display of rectangle images on curved screens.

CCS CONCEPTS

•**Computing methodologies** →**Perception;** Image processing;

KEYWORDS

Curved screens, immersive effect, depth maps, image adaptation.

ACM Reference format:
Shao-Ping Lu[1,2], Ruxandra-Marina Florea[1,2], Pablo Cesar[3,4], Peter Schelkens[1,2], Adrian Munteanu[1,2]. 2016. Efficient Depth-aware Image Deformation Adaptation for Curved Screen Displays. In *Proceedings of ACM Conference, Washington, DC, USA, July 2017 (Conference'17),* 9 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

*This research was supported by the Fonds Wetenschappelijk Onderzoek (FWO), Vlaanderen (projects G025615, G084117).

Conference'17, Washington, DC, USA

DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Providing users with an immersive viewing experience can prove useful in entertainment, medical care, military simulations, or teleconferencing [20]. This has prompted extensive research in the creation of realistic 3D environments. Ultra-high-definition (UHD) displays, as well as 3D screens have been developed with the same purpose in mind [21, 25]. Following this trend, mainstream manufacturers are currently investing in the production of curved displays, both in the TV and in the smartphone sector. The idea of curved screens is by no means new, as curved projection surfaces are commonly used in cinemas across the globe. The curvature is enforced to avoid distortion which would normally occur towards the vertical edges of a flat screen, given the considerable distance between the projector and the screen. Here, the feeling of immersion comes from the considerable size of the projection surface. In contrast, curved screen TVs and curved smartphone displays rely on the curvature to provide the user with an immersive viewing experience.

Displaying a rectangular input image on a curved surface is not straightforward. Specifically, the input image has to be adapted to the curved surface in a way that: 1) the local content in the scene should avoid being roughly cut or implausibly stretched in the output domain; 2) the objects belonging to different depth layers should be scaled individually, and 3) the scene geometry should not be unreasonably deformed.

In this paper, we introduce an efficient *content-aware* and *depth-aware* mesh-based warping from a planar input image to a curved screen display. The input rectangular image is firstly mapped onto an intermediary planar surface with curved top and bottom boundaries. The intermediary planar image then undergoes a perspective projection onto the curved surface. Our method introduces minimal distortion to foreground objects and ensures local similarity preservation even for small depth content elements.

In summary, the main contributions in this paper are as follows:

- We are the first to describe the problem of the content-aware and depth-aware curved screen display adaptation for rectangle images;
- We introduce an efficient global energy-based solution to find the desired image deformation for the curved screen display;
- We perform a subjective evaluation of our proposed method, employing an ad-hoc curvature-controllable curved screen display system.

In the next section a state-of-the-art overview is provided. We detail our proposed method in Sec. 3, and discuss the experimental results in Sec. 4. Finally, we give the conclusion to our work. We further provide a video demo and a supplemental document.

2 RELATED WORK

The proposed work mainly focuses on the mapping from the input image to the curved screen, which technically requires image scaling (or image retargeting). Naive approaches in image retargeting came in the form of cropping or uniform scaling. However, while the first inadvertently implies loss of content, the second will typically introduce visible distortion in important areas. Consequently, numerous methods have been proposed in the area of content-aware image retargeting. The literature usually divides such methods in two general classes: discrete, and continuous methods.

The first class comprises the family of seam carving methods [1, 19, 22, 24, 27, 28]. Such methods progressively eliminate low energy seams until the image reaches its target dimension. Classical seam carving [27] assumed seams that span the entire height or width of the image, and comprise exactly one pixel per row or, correspondingly, column of the image. Mansfield *et al.* [19] employ a user-provided relative depth map in obtaining a layer decomposition of the input image. Shen *et al.* [28] have developed a depth-aware single image seam carving approach. Gradient vector flow and saliency guide the seam removal in [1]. Seam carving has also been investigated in the context of video data [24]. As an extension work, [22] allows for irregular output domains.

As successful as seam carving methods have been in the area of image retargeting, loss of content is still implied. An alternative comes in the form of continuous methods, which involve mesh-based warping models on the image. Particularly, such methods segment the input image into pixels, which are then subjected to a non-uniform deformation. The deformation is the result of an optimization process that accounts for spatial constraints such that the deformation distortion on important pixels/regions is minimized.

Much like seam carving methods, a subclass of continuous methods focus on resizing and, consequently, assumes a planar input and output. The methods in [4, 8, 12] consider planar regions, straight lines or vanishing points when minimizing the deformation distortion. Mapping stitched panoramic images has also been addressed in [15]. Chang *et al.* [6] introduce a Hough space formulation of the deformation energy to preserve salient regions, and line structures. Inspired by the Scene carving work [18], Lee *et al.* [13] introduce a hybrid method for stereoscopic image resizing. Wang *et al.* [29] proposed an image resizing method that assigns spatially varying factors by optimization. In [32], the distortion energy is distributed to relatively non-salient regions of an image. Finally, while most methods perform conformal mappings relying on quad meshes, Guo *et al.* [11] opt for a saliency-based Delaunay triangulation of the input image.

A second class of continuous methods addresses the issue of retargeting between a planar image and a spherical or cylindrical alternative. Early efforts were made by Levy *et al.* [14], which investigated texture mapping via a quasi-conformal parameterization. [5] proposed a semi-automatic mapping from a wide-angle image on the viewing sphere, to a flat image. Chang *et al.* [7] introduced a two-step projection model, wherein a viewing hemisphere is mapped to the image plane. Finally, mapping planar images to curved surfaces has been addressed in [25] and [23]. Specifically, Sajadi *et al.* [25] estimate the camera parameters and projector properties for cylindrical displays. In order to achieve seamless display on a curved screen using overlapping projectors, Raskar *et al.* [23] address geometric calibration and alignment issues. The discrete Ricci flow-based conformal mapping of [21] was investigated to provide a natural immersive effect in specific virtual reality environments (e.g. 5-sided CAVEs).

The literature provides several extensions of continuous video deformation methods, such as video resizing [30], video stabilization [16] and fisheye video correction [31]. Compressed-domain alternatives have been provided in [9] and [33] to meet low memory requirements in mobile phones. It should be noted that in the aforementioned works the output is still a rectangular planar image.

Our depth-based image deformation approach belongs to the second class of retargeting methods (i.e. with continuous deformation), and our target display can be seen as embedded in a cylindrical domain. We note that when displaying content on such a curved screen, the deformed output image has a non-rectangular shape that discrete methods cannot produce.

With respect to other continuous methods, [5, 7, 14] consider a curved input and a planar output. In contrast, our method addresses the inverse problem of projecting a planar image onto a curved surface. Moreover, we target a cylindrical, rather than a spherical output. We note that the literature distinguishes between these two cases; due to their different behavior near the zenith and the nadir, separate projection models need to be investigated. In addition, to our knowledge, there is no content-based warping method in current literature that automatically maps an image to a variable-curvature surface.

Finally, both [21] and our proposed method target local shape-preservation. However, we focus on curved screen displays, whereas [21] targets flat panel displays. Additionally, our image deformation is subjected to several constraints such as straight line preservation and depth-based scaling.

3 PROPOSED METHOD

3.1 Problem formulation

Our target display can be seen as embedded in a cylindrical domain. Fig. 1 shows the mapping relation between the image plane, defined by the flat rectangle *ABCD*, and the curved screen. The center of curvature V is *the viewpoint*. Rectangular images can be displayed on such curved surfaces in one of two ways: *(i)* by positioning a projector in the center V of the circle describing the curvature (i.e. the viewpoint), and subsequently *projecting* the input images, or *(ii)* by stretching the input images to fit the desired output domain, then *directly displaying* them on the screen surface (e.g. with Liquid Crystal Displays (LCDs)).

Reverse perspective mapping (w.r.t. the viewpoint V) of the content on the curved screen will produce an image on the image plane. Thus, in order to investigate different display solutions on the curved screen, one needs to *(i)* derive an optimized mapping relation between the image plane and the curved screen, and *(ii)*

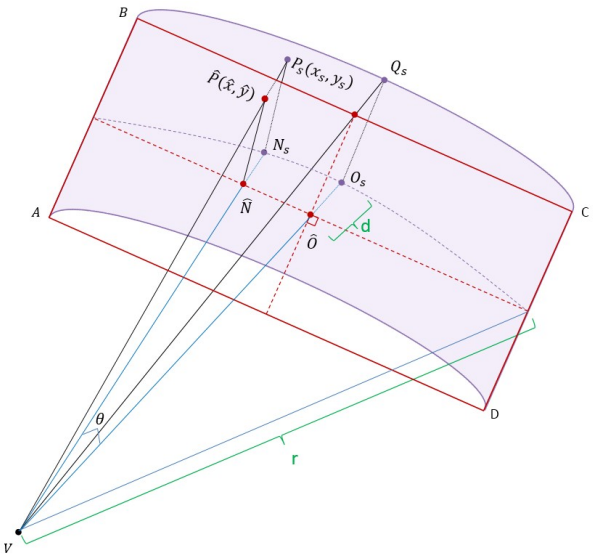


Figure 1: Geometric relation between the viewpoint V, the image plane (red boundary) and the curved output screen (purple outline).

accurately assess any occurring distortion between the input image and its correspondent on the image plane.

Suppose the width and height of the input image are w and h, respectively. Here we refer to pixels in the original image as $P(x, y)$, where x and y denote the column and row index (relative to the top-left corner), respectively. Similarly, pixels in the deformed image on the image plane are denoted by $\hat{P}(\hat{x}, \hat{y})$. Finally, pixels in the output image, i.e. on the curved screen, are denoted by $P_s(x_s, y_s)$.

The literature provides several classical approaches to mapping, such as perspective and uniform mapping. The **perspective mapping** relation between the input image and and its correspondent in the image plane assumes that each pixel coordinates follows the relation:

$$\begin{cases} \hat{x} = \frac{w}{2} + (r - d)\tan\theta \\ \hat{y} = \frac{h}{2} + \frac{1}{r}(y - \frac{h}{2})(\frac{r-d}{\cos\theta}), \end{cases} \tag{1}$$

where r is the radius of the screen curvature, d is the distance between the center of the curved screen and the center of the image plane, and θ represents the deformation angle. In the case of perspective projection, it requires imposing the constraint $0 < \hat{y} < h$ in Eq. 1, on the width of the output image. This implies that the visible content on the image plane will be less than the content in the input image. Consequently, the visible content on the image plane will appear *cropped*.

An alternative solution comes in the form of **uniform mapping**, i.e. uniformly scaling the input image and directly displaying the result on the curved screen. When mapping the output result on the image plane by means of a reversed perspective projection (w.r.t. the viewpoint), the following should hold:

$$\hat{x} = \frac{w}{2} + \frac{w * \theta}{2\arccos(\frac{r-d}{r})}. \tag{2}$$

In contrast to the two previously-mentioned methods, **the proposed mapping** is content-aware, and guided by the depth information of the input image. Our method involves mapping the original rectangular image onto a planar surface whose top and bottom boundaries are represented by symmetrical curves. The geometry of the boundaries is dictated by the curvature of the display surface. The intermediary planar image is then re-mapped onto the curved surface (see the proposed framework in Fig. 2). In particular, we impose the following four constraints on the image deformation: *(i)* the shape of all individual content elements should be locally preserved as much as possible between input and output; *(ii)* foreground objects should incur minimal scaling distortion; *(iii)* straight lines should be well preserved in the deformed image; and *(iv)* the deformed image should be well aligned to the (straight) vertical, and (curved) horizontal boundaries of the curved screen. We will tackle each of these constraints in the following subsections.

3.2 Globally optimized energy based image deformation

Local similarity preservation: Mapping a rectangular image to a curved display is open to multiple valid solutions. Following the modeling in [32], we prefer that the entire output image appears less distorted in terms of shapes and angles of the local content, and that the distortion is smoothly changed in the 2D coordinates. This can be achieved if each point to be mapped undergoes a *conformal mapping* (or similarity transformation).

Let us assume $\acute{P}(\acute{x}, \acute{y})$ to be a pixel in the deformed image on the image plane. Moreover, let $\acute{P}(\acute{x}, \acute{y})$ be the result of the similarity transformation underwent by $P(x, y)$. The new homogeneous coordinates are obtained as:

$$\begin{bmatrix} \acute{x} \\ \acute{y} \\ 1 \end{bmatrix} = \alpha \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) & 0 \\ \sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{3}$$

where φ is the rotation angle, α represents a scaling factor, and t_x and t_y are the translation variables in the horizontal, and vertical direction, respectively.

Next, consider that under our desired (i.e. optimized) deformation, the pixel $P(x, y)$ is mapped to a position $\hat{P}(\hat{x}, \hat{y})$ on the image plane. Compared to the similarity transformation, the energy associated to the desired deformation can be written as:

$$E_S(\boldsymbol{p}) = ||\begin{bmatrix} \acute{x} \\ \acute{y} \end{bmatrix} - \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix}||^2 = ||\begin{bmatrix} x & -y & 1 & 0 \\ y & x & 0 & 1 \end{bmatrix} D_s - \begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix}||^2, \tag{4}$$

where the optimal similarity transformation matrix is denoted as $D_s = [\alpha\cos(\varphi) \;\; \alpha\sin(\varphi) \;\; t_x \;\; t_y]^T$.

In our solution, the image deformation is aided by a quad mesh $M = \{F, V\}$, where $F = \{q_m, 0 \leqslant m \leqslant N_q\}$ is the set of rectangular mesh faces, and $V = \{v_k = (x_k, y_k), 0 \leqslant k \leqslant N_v\}$ is the set of vertices. The quad mesh is superimposed on the 2D image. Without loss of content, we will refer in the following to an arbitrary mesh face q defined by the four vertices v_a, v_b, v_c and v_d, where v_a and v_b, as well as v_c and v_d, are placed diagonally from each other.

As proven in [32], applying the similarity transformation on the vertices of q has an associated energy whose linear least square

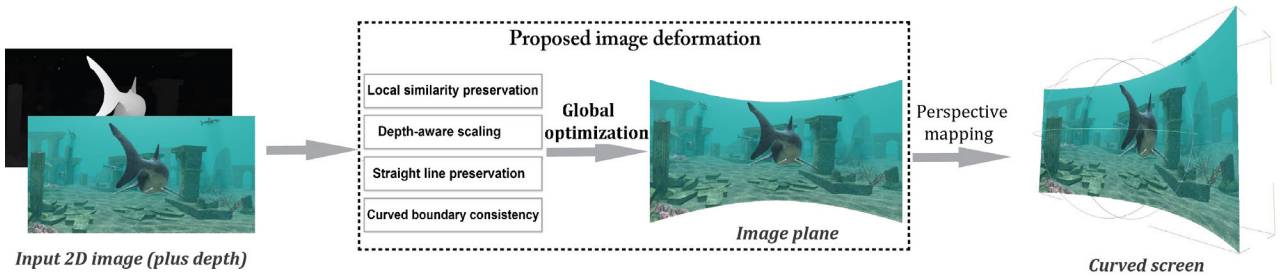


Figure 2: The proposed framework for 2D rectangle image deformation adaptation for curved screen displays.

minimization leads to:

$$E_S = \sum_q ||A_q(A_q^T A_q)^{-1}A_q^T - I)\hat{B}_q||^2, \tag{5}$$

where I is the $8 * 8$ identity matrix and

$$A_q = \begin{bmatrix} x_a & -y_a & 1 & 0 \\ y_a & x_a & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots \\ x_d & -y_d & 1 & 0 \\ y_d & x_d & 0 & 1 \end{bmatrix}, \quad \hat{B}_q = \begin{bmatrix} \hat{x}_a \\ \hat{y}_a \\ \vdots \\ \hat{x}_d \\ \hat{y}_d \end{bmatrix}. \tag{6}$$

In this equation, (x_a, y_a) and $(\hat{x}_a, \hat{y}_a)$ are mapping pairs (before and after the deformation) corresponding to the vertex v_a. Similar statements hold for vertices v_b, v_c and v_d.

Depth-aware scaling: We now impose a constraint on the desired deformation to ensure depth-aware foreground content preservation. Formally, the foreground content preservation energy is denoted as:

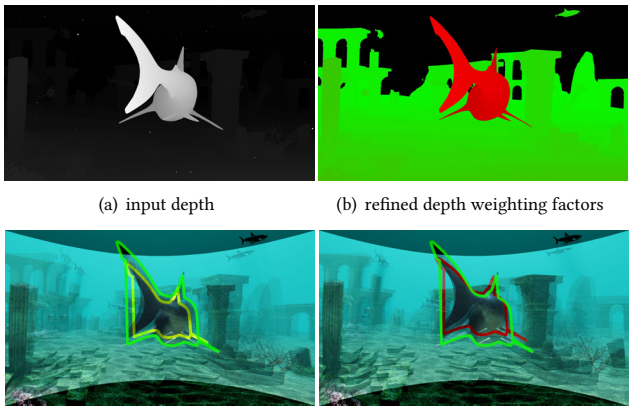
$$\begin{aligned} E_F = \sum_q \gamma(q)((d_x(\hat{v}_a, \hat{v}_b) - d_x(v_a, v_b))^2 \\ + (d_y(\hat{v}_a, \hat{v}_b) - d_y(v_a, v_b))^2 \\ + (d_x(\hat{v}_c, \hat{v}_d) - d_x(v_c, v_d))^2 \\ + (d_y(\hat{v}_c, \hat{v}_d) - d_y(v_c, v_d))^2), \end{aligned} \tag{7}$$

where $d_x(\cdot,\cdot)$ and $d_y(\cdot,\cdot)$ are distances in the horizontal and vertical directions, respectively. As before, v_a, v_b, v_c and v_d are the four vertices describing face q in the employed quad mesh. To impose a smooth distribution of the depth map weighting factors between 0 and 1, we introduce a sigmoid-like function:

$$\gamma(q) = \frac{2}{e^{-\frac{\kappa(q)}{40}} + 1} - 1, \tag{8}$$

where $\kappa(q)$ is the mean depth value of all pixels enclosed by face q. We include in Fig. 3 a visual example of an input depth map and associated weighting factors using the above-mentioned approach. A more detailed evaluation of the foreground area preservation is provided in the experimental results section. It should be noted that integrating more complicated methods with depth-based weighting (e.g. attention models in [2, 10]) would potentially improve the deformation result, though increasing the complexity of the algorithm.

Straight line preservation: Previous work in the area of image deformation [4, 7, 12, 31] has highlighted the importance of preserving straight lines. To ensure this, we impose an additional



(a) input depth (b) refined depth weighting factors

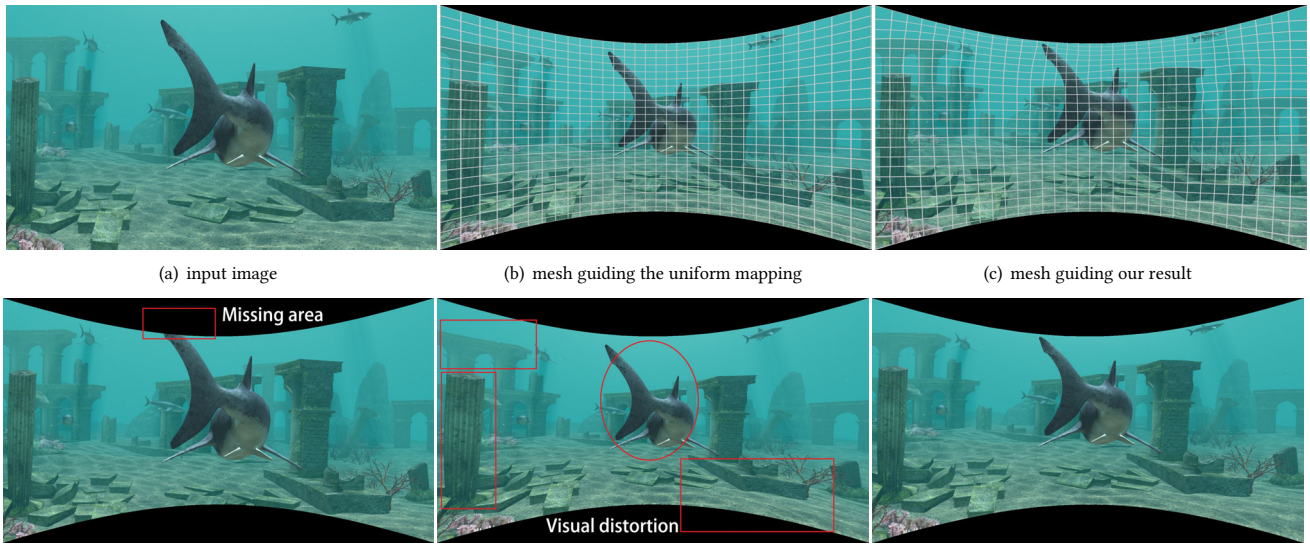
(c) uniform mapping vs. input (d) proposed approach vs. input

Figure 3: Example of depth-aware foreground preservation. The top row depicts (a) the input depth, and (b) the refined depth weighting factors. The bottom row shows the overlay comparison of the deformed foreground objects with (c) uniform mapping and (d) our proposed approach. The green areas are detected input foreground objects, while the yellow region in (c) and the red region in (d) are the deformed results using uniform mapping, and our proposed approach, respectively.

constraint on the image deformation. Consider a straight line with start, and end points (x_0, y_0) and (x_n, y_n) respectively. Assume that the quad mesh aiding the image deformation will cut such a straight line into n segments. Let (x_i, y_i), with $0 < i < n$, be the cutting points. We define the straight line preservation energy as:

$$\begin{aligned} E_L = \sum_l (\sum_{i=1}^{n} ((\hat{y}_i - \hat{y}_{i-1}) - \hat{\lambda}(l)(\hat{x}_i - \hat{x}_{i-1}))^2 \\ + \omega_1(\hat{\lambda}(l) - \lambda(l))^2), \end{aligned} \tag{9}$$

where $\lambda(l)$ and $\hat{\lambda}(l)$ are the original, and desired slopes of the straight line l in the input and output images, respectively, and ω_1 is a weighting factor. The scope of such a definition is two-fold. On the one hand, we attempt to make each resulting line segment straight. On the other hand, we intend to preserve the original slope of the line, from the original to the output image.



(a) input image (b) mesh guiding the uniform mapping (c) mesh guiding our result

(d) perspective mapping (e) uniform mapping (f) our result

Figure 4: Curved screen display adaptation for rectangular images. Our result is better than some traditional methods (i.e., using uniform stretching or perspective mapping). In the first row, (a) is the input color image (the input depth is shown in Fig. 3), (b) and (c) are deformed meshes with uniform mapping and our approach, respectively. The second row shows the output results with different methods.

The $(\hat{x}_i, \hat{y}_i)$ coordinates in Eq. 9 can be obtained by means of linear interpolation between the four surrounding quad mesh vertices, i.e. the vertices defining the mesh face which encloses the pixel $\hat{P}(\hat{x}_i, \hat{y}_i)$. The solution can then be seen as a rearrangement problem, or optimal deformation of the quad mesh, in which new quad mesh vertex positions are derived such that $\hat{\lambda}(l) = \lambda(l)$. Note that some nonlinear approaches, e.g. iteration-based methods [12], can be introduced to find the optimal $\hat{\lambda}(l)$ and $(\hat{x}_i, \hat{y}_i)$. However, we simplify the equation by setting $\hat{\lambda}(l) = \lambda(l)$. Such a global solution will prevent the algorithm from converging to local minima. In addition, this enforces each segment of the straight line to maintain its original direction, while the straight line preservation energy E_L becomes a quadratic function of the mesh vertices. Finally, the proposed simplification comes with the added benefit of time efficiency.

Curved boundary constraints: One needs to impose additional constraints such that the mesh vertices on the boundaries of the original image are mapped to the corresponding boundaries of the curved screen. Let $v_i(x_i, y_i)$ be a boundary vertex. Preserving the two vertical (i.e. left and right) boundaries has an associated energy defined as:

$$E_V = \sum_{v_i \in V_L} \hat{x}_i^2 + \sum_{v_i \in V_R} (\hat{x}_i - w)^2, \tag{10}$$

where V_L and V_R are the sets of mesh vertices on the left and right boundaries, respectively. As before, w is the width of the input image.

We now consider the two horizontal (i.e. top and bottom) boundaries, and define the associated boundary preservation energy as:

$$E_H = \sum_{v_i \in V_T} \chi((\hat{x}_i, \hat{y}_i), C_T) + \sum_{v_i \in V_B} \chi((\hat{x}_i, \hat{y}_i), C_B), \tag{11}$$

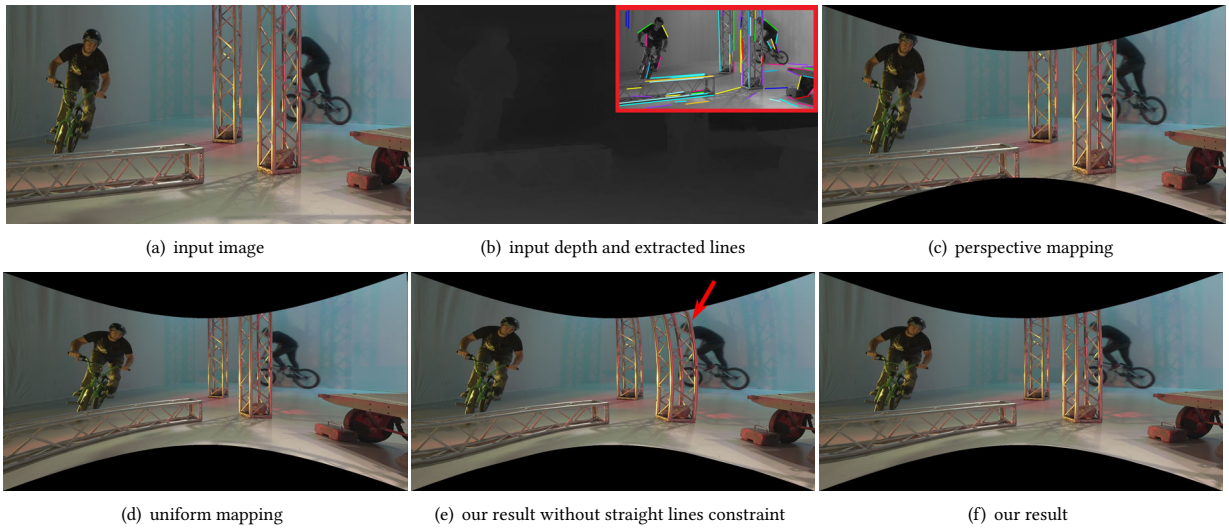
where V_T and V_B are the sets of mesh vertices corresponding to the top and bottom boundaries, respectively. The function $\chi(\cdot, \cdot)$ represents the shortest Euclidean distance between a vertex and a function describing a 2D curve. Finally, C_T and C_B are functions describing the (curved) top and bottom boundaries, respectively. Setting $y = 0$, and $y = h$ into Eq. 1, any pixels that lie on the top, and bottom boundaries of the deformed image respect the conditions:

$$\begin{cases} C_T : (r-d)^2 + (\hat{x} - \frac{w}{2})^2 - r^2(1 - \frac{2\hat{y}}{h})^2 = 0 \\ C_B : (r-d)^2 + (\hat{x} - \frac{w}{2})^2 - r^2(1 + \frac{2\hat{y}}{h})^2 = 0 \, . \end{cases} \tag{12}$$

Total energy: Taking into consideration all previously-defined energy functions and constraints, the overall energy of the optimization function is equal to:

$$E = E_S + \beta_f E_F + \beta_l E_L + \beta_b (E_H + E_V) \, , \tag{13}$$

where $\beta_f = 50$, $\beta_l = 500$ and $\beta_b = 10^6$ are corresponding weighting factors for each aforementioned constraint. As it can be deduced from the magnitude of the weighting factors, we aim for a near-perfect match between the output boundaries and the target boundaries. Moreover, we assign a higher importance to straight line and foreground preservation than to shape preservation. Experiments have shown that these parameters enable our proposed system to reach the desired results.



(a) input image (b) input depth and extracted lines (c) perspective mapping

(d) uniform mapping (e) our result without straight lines constraint (f) our result

Figure 5: Comparison for the *BMX* image, between uniform mapping, perspective mapping, and our proposed approach (with/without the straight line preservation constraint).

3.3 Energy minimization

Intuitively, minimizing the energy function E_H is a non-linear quadratic problem. Alternative solutions may be reached by employing iterative methods or constructing a more complicated framework to guide the quad mesh deformation. However, for the sake of efficiency, we impose that for any vertex $v_i(x_i, y_i)$ situated on the top (or bottom) boundary of the input image, its targeted deformation $\hat{v}_i(\hat{x}_i, \hat{y}_i)$ should lie on C_T (or C_B). Moreover, we impose the constraint $\hat{x}_i = x_i$ during the deformation. Thus, the total energy remains a linear combination of all linear quadratic functions, and it can be directly solved by a linear least-square minimization. Experiments have shown that such a linear formulation generates a satisfactory deformation. In our implementation, we use Pardiso [26] to efficiently solve the described linear system.

4 EXPERIMENTAL RESULTS

4.1 Implementation and Runtime cost

We implemented our approach on the Microsoft Visual C++ 2012 platform on a high-end portable computer with 2.3GHz Quad-Core Intel-i7 CPU and 8GB memory. The input image is uniformly separated as 32x24 mesh grids. After the desired mesh coordinates are solved by the globally optimal deformation, we use the GPU-based standard texture mapping to efficiently render the output image.

The computational cost in our system mainly lies in the straight line detection, GPU-based texture mapping and the global optimization. For a full HD image (e.g. the *Shark* case), the time costs for the three previously-mentioned aspects are about 0.03s, 0.10s and 0.05s, respectively, and the total execution time is less than 0.25s.

4.2 Results

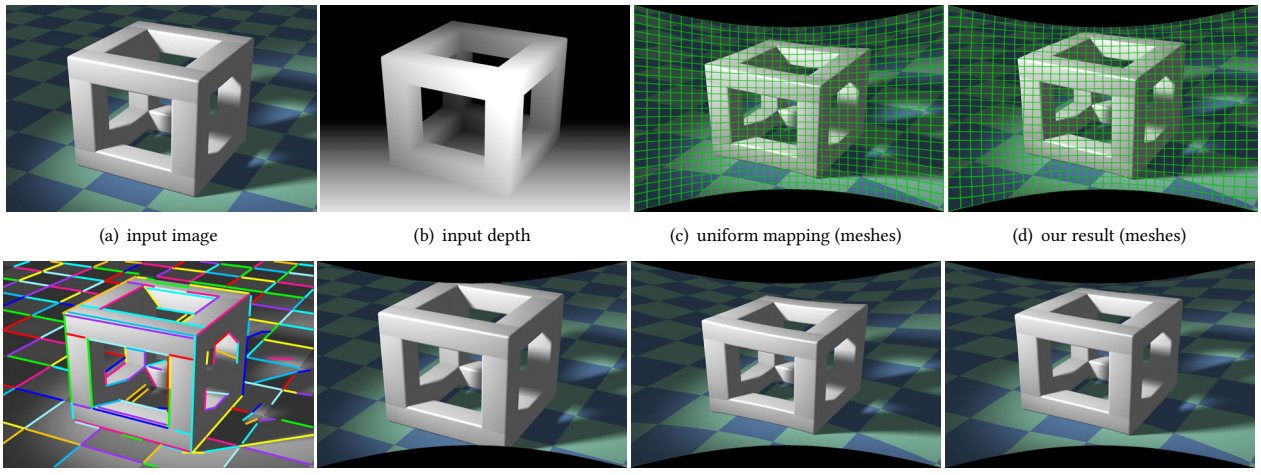
We have compared our algorithm against both uniform, and perspective mapping, on various test images.

Fig. 4 depicts the results for the *Shark* image. The meshes used to guide the uniform mapping, and our proposed approach are also given in this figure. It is important to observe that perspective mapping preserves the scene geometry, though at the cost of losing content (see the ground area and the tip of the shark tail). The image generated through uniform mapping preserves the scene content, but affects the geometry (see the stones on the ground, the pillar and the distant arch). In contrast, our approach efficiently preserves the content of the image, as well as the scene geometry. As an example, the shape of the shark is better preserved through our method than through uniform mapping.

Fig. 5 depicts the results obtained for the *BMX* image. The comments on perspective, and uniform mapping made in the previous paragraph for the *Shark* image hold in this case as well. In addition, we include results generated by our approach with and without straight line preservation (see Fig. 5(e) and Fig. 5(f)). One can easily observe the importance of straight line preservation in the output result, as well as the accuracy of the straight line modeling in our algorithm.

More examples can be seen in Fig. 6 and the supplemented file. In such examples that the geometry of the planar is better preserved with our approach than with uniform mapping; as expected, perspective mapping crops the result. It is also obvious that uniform mapping bends straight lines, while our approach can avoid such a negative effect. In general, one can observe that the uniform mapping introduces geometric distortion for the plane area, while perspective mapping crops content on the top, and bottom areas. In contrast, our approach is not prone to such phenomena.

A previous example of the depth-aware foreground preservation was given in Fig. 3. Comparing to the original size of the shark (green area both in Fig. 3(d) and Fig. 3(c)), one can easily see that the result produced by our approach (red area in the bottom-right) produces a more faithful result than the uniform mapping (yellow



(a) input image (b) input depth (c) uniform mapping (meshes) (d) our result (meshes)

(e) extracted lines (f) perspective mapping (g) uniform mapping (h) our result

Figure 6: Results generated by uniform mapping, perspective mapping, and our proposed approach for the *Cubic* image.

Table 1: The foreground preservation comparison.

	Perspective mapping				***Uniform mapping***				***Our approach (without E_F)***				***Our approach (with E_F)***			
Curvature (%)	3.5	9.5	23.5	29.5	3.5	9.5	23.5	29.5	3.5	9.5	23.5	29.5	3.5	9.5	23.5	29.5
Shark	**0**	**0**	X	X	7.7	23.0	59.0	74.4	7.3	21.7	52.5	66.1	6.7	18.6	**45.6**	**57.4**
Baby1	X	X	X	X	4.7	13.8	36.3	45.9	4.7	13.5	35.6	44.7	**4.6**	**13.0**	**33.1**	**41.6**
BMX	X	X	X	X	10.4	28.6	77.0	97.7	9.7	24.2	65.5	88.9	**8.9**	**21.3**	**61.1**	**76.0**
BookArrival	X	X	X	X	7.6	20.3	52.7	66.8	6.8	18.7	46.4	61.1	**6.3**	**16.9**	**42.0**	**53.1**
Cubic	**0**	**0**	X	X	10.2	28.1	73.9	94.0	9.8	25.8	66.0	81.8	8.2	23.3	**58.7**	**73.5**
Plane	**0**	**0**	**0**	X	3.2	10.1	27.5	35.1	2.8	8.8	23.3	27.9	2.3	6.9	16.0	**21.4**

(a) controllable curvature display (b) extreme curvatures display on TV

Figure 7: The evaluation setups on both the controllable-curvature screen and TV.

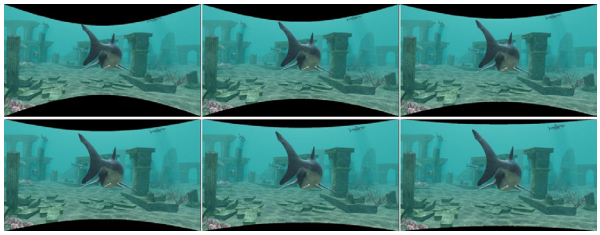
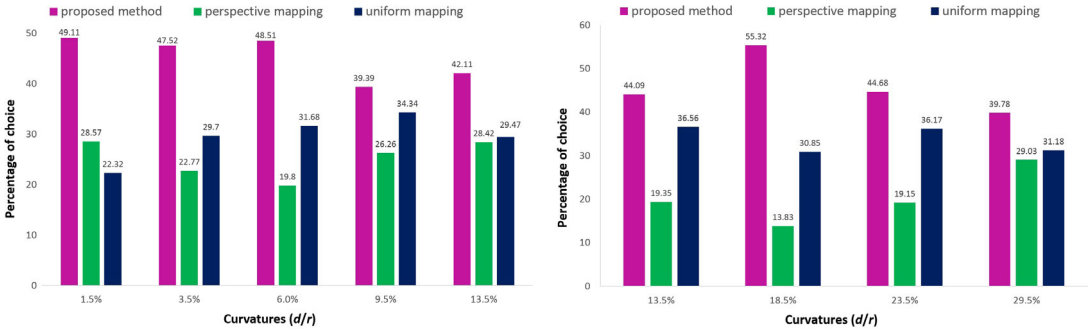


Figure 8: Our mapping results with different curvatures.

area in the bottom-left). In addition, to objectively assess the accuracy of the foreground preservation, we define the deformation distance by:

$$\mathcal{D} = \frac{1}{m}\sum_i \sqrt{d_x^2(v_i, \hat{v}_i) + d_y^2(v_i, \hat{v}_i)}, \tag{14}$$

where m is the number of discrete points on the foreground boundary. Note that we align the centroid of the foreground before computing the deformation distance. We have investigated various images with different curvatures. As it can be seen in Table 1, perspective mapping tends to be eliminated in most cases, due to the fact that it cuts the foreground (see the label X in the table). On the other hand, the deformation distances in our method are all smaller than those using uniform mapping. Moreover, with increasing curvature, the gains in terms of deformation distance become more significant. In other words, our method is more competitive in terms of foreground preservation when the screen curvature is more pronounced.



(a) evaluation on curvature-controllable curved display (b) evaluation for extreme curvatures display

Figure 9: Subjective evaluation with various curvatures for both our curvature-controllable display and extreme curvatures.

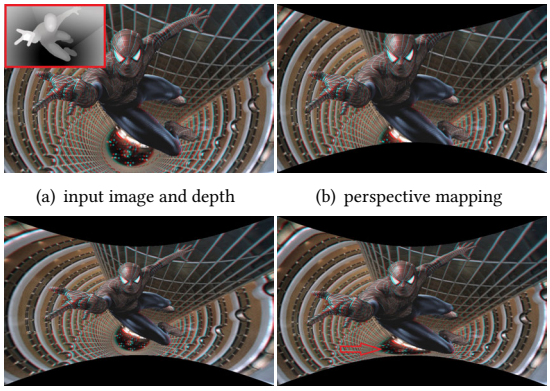
4.3 User study

We performed a subjective evaluation of our proposed approach, of perspective, and of uniform mapping. We note that commercially-available curved screen TVs have a fixed curvature and a small bending degree. For this reason, we opted for constructing an ad-hoc curvature-controllable curved display (see Fig. 7(a)). In this way, we manage to avoid curvature restrictions and to also prove the generality of our method. The dimensions of the test surface are 120x240 cm. A set of 6 test images (see the first column of Table 1) were projected on the test screen with a single projector and subjected to the evaluation. For generality, we included in the experiment both natural and artificial images, with various degrees of foreground complexity. We keep the distance r to the viewpoint fixed, and express the distance d between the center of the curved screen and the center of the image plane as a percentage of r. We randomly selected the ratios of 1.5%, 3.5%, 6.0%, 9.5%, and 13.5% to ensure a good test base of screen curvatures.

The tests have been performed in a standard compliant ITU-R BT.500-1138 [3] visual quality test facility, consistent with classical still image evaluations. The evaluation methodology is in line with user studies presented in [11] and [12], in the sense that the users were asked to mark their preference among the displayed results.

The viewing distance was inside the effective view angle and close to the sweet spot of the curved screen. We opted for a single-stimulus evaluation, in which the result of each projection method was evaluated in pre-recorded, non-interactive 10-second test sequences. In total, each session lasted for 10 minutes, which includes the 10 seconds allocated per method for each image, as well as the scoring time. 15 participants (including one professional expert) have been invited to evaluate the test sequences. For each sequence, different mapping results were shown in a random order, and the participants were asked to vote their preference. Fig. 8 depicts several mapping results with different curvatures.

Extreme curvatures. We also performed the evaluation for extreme curvatures, where the deformed images (i.e. on the image plane) were displayed on a classical TV screen (see Fig. 7(b)). This was still in the same standard compliant ITU-R BT.500-1138 visual quality test facility, with the same characteristics as in the first evaluation. The test conditions were identical to the first evaluation, with the exception that the reference input image was shown



(a) input image and depth (b) perspective mapping

(c) uniform mapping (d) our failed result

Figure 10: A failure case. Although our algorithm can better preserve the foreground, implausible distortions are introduced due to the inaccurate depth map and the lack of free-form curves preservation.

before the three mapping results. In this case, the curvature of the screen corresponded to d being 13.5%, 18.5% 23.5%, and 29.5% of r.

Experimental results can be seen in Fig. 9(a) and Fig. 9(b), where the two aforementioned test setups are calculated for various curvatures, respectively. In all cases, our method was chosen with a majority over both perspective, and uniform mapping. Perspective mapping had the lowest scores. This is likely to be caused by the fact that perspective mapping crops the displayed content.

5 DISCUSSION AND CONCLUSION

In this paper, we present an efficient content-aware and depth-aware mapping method for curved screen displays. Our method faithfully preserves not only the content, but also the geometry of the objects inside each input picture. Moreover, our method manages to preserve both foreground, and straight line preservation.

The proposed approach has several limitations. Firstly, our optimization solution relies on accurately weighting of the depth maps and extraction of straight lines. Any inaccuracies can result in visual distortion. Moreover, our solution does not extract and preserve free-form lines. As a consequence, such lines might be subjected

to undesired deformation. Inaccurate depth maps and the fact that our algorithm does not identify or preserve any existing curves may lead to a sub-optimal result as well. Fig. 10 depicts such an failure example. Although our method faithfully preserves the foreground (see Spiderman figure), the area around the vanishing point is distorted. A better analysis of the 3D scene geometry, as well as of camera parameters, might improve the proposed mapping.

One of the most likely directions of our future work is an extension of the proposed approach to stereoscopic image and 2D video (with associated depth maps). In the latter case, spatio-temporal consistency would be critical. Our approach might also benefit from robust and accurate video content description(e.g. [17]). For this reason, we will investigate the possibility of incorporating motion object modeling and adaptive video mesh representation formats into the proposed framework.

REFERENCES

[1] S. Battiato, G. M. Farinella, G. Puglisi, and D. Ravi. 2014. Saliency-Based Selection of Gradient Vector Flow Paths for Content Aware Image Resizing. *IEEE Trans. Image. Process.* 23, 5 (2014), 2081–2095.

[2] I. Bogdanova, A. Bur, and H. Hugli. 2008. Visual Attention on the Sphere. *IEEE Trans. Image. Process.* 17, 11 (2008), 2000–2014.

[3] ITU-R Rec. BT.500-13. 2012. *Methodology for the subjective assessment of the quality of television pictures.* Technical Report.

[4] Robert Carroll, Aseem Agarwala, and Maneesh Agrawala. 2010. Image warps for artistic perspective manipulation. *ACM Trans. Graph.* 29, 4 (2010), 127:1–9.

[5] Robert Carroll, Maneesh Agrawal, and Aseem Agarwala. 2009. Optimizing content-preserving projections for wide-angle images. *ACM Trans. Graph.* 28 (2009), 43:1–10.

[6] Che-Han Chang and Yung-Yu Chuang. 2012. A line-structure-preserving approach to image resizing. In *Proc. IEEE Int. Conf. CVPR.* 1075–1082.

[7] Che-Han Chang, Min-Chun Hu, Wen-Huang Cheng, and Yung-Yu Chuang. 2012. Rectangling Stereographic Projection for Wide-Angle Image Visualization. In *Proc. IEEE Int. Conf. ICCV.* 2824–2831.

[8] Wang Chen, Irene Cheng, Zihui Xiong, Anup Basu, and Maojun Zhang. 2011. A 2-point algorithm for 3D reconstruction of horizontal lines from a single omni-directional image. *Pattern Recognition Letters* 32, 3 (2011), 524–531.

[9] C. W. Deng, W. S. Lin, and J. F. Cai. 2012. Content-Based Image Compression for Arbitrary-Resolution Display Devices. *IEEE Trans. Multimedia* 14, 4 (2012), 1127–1139.

[10] Y. M. Fang, W. S. Lin, B. S. Lee, C. T. Lau, Z. Z. Chen, and C. W. Lin. 2012. Bottom-Up Saliency Detection Model Based on Human Visual Sensitivity and Amplitude Spectrum. *IEEE Trans. Multimedia* 14, 1 (2012), 187–198.

[11] Yanwen Guo, Feng Liu, Jian Shi, Zhi-Hua Zhou, and Michael Gleicher. 2009. Image retargeting using mesh parametrization. *IEEE Trans. Multimedia* 11, 5 (2009), 856–867.

[12] Kaiming He, Huiwen Chang, and Jian Sun. 2013. Rectangling panoramic images via warping. *ACM Trans. Graph.* 32, 4 (2013), 79:1–9.

[13] Ken-Yi Lee, Cheng-Da Chung, and Yung-Yu Chuang. 2012. Scene warping: Layer-based stereoscopic image resizing. In *Proc. IEEE Int. Conf. CVPR.* 49–56.

[14] Bruno Levy, Sylvain Petitjean, Nicolas Ray, and Jrome Maillot. 2002. Least squares conformal maps for automatic texture atlas generation. *ACM Trans. Graph.* 21 (2002), 362–371.

[15] Dongping Li, Kaiming He, Jian Sun, and Kun Zhou. 2015. A Geodesic-Preserving Method for Image Warping. In *Proc. IEEE Int. Conf. CVPR.* 213–221.

[16] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. 2009. Content-preserving warps for 3D video stabilization. *ACM Trans. Graph.* 28 (2009), 44:1–9. Issue 3.

[17] Shao-Ping Lu, Song-Hai Zhang, Jin Wei, Shi-Min Hu, and Ralph R Martin. 2013. Timeline Editing of Objects in Video. *IEEE Trans. Visual. Comput. Graphics* 19, 7 (2013), 1218–1227.

[18] A. Mansfield, P. Gehler, L. Van Gool, and C. Rother. 2010. Scene carving: Scene consistent image retargeting. In *Proc. Int. Conf. ECCV.* 143–156.

[19] A. Mansfield, P. Gehler, L. Van Gool, and C. Rother. 2010. Visibility maps for improved seam carving. In *Proc. Int. Conf. ECCV. workshop.*

[20] Rufael Mekuria, Michele Sanna, Stefano Asioli, Ebroul Izquierdo, Dick C. A. Bulterman, and Pablo Cesar. 2013. A 3D Tele-immersion System Based on Live Captured Mesh Geometry. In *Proc. 4th ACM Multimedia Systems Conference.* 24–35.

[21] Kaloian Petkov, Charilaos Papadopoulos, Min Zhang, Arie E Kaufman, and Xianfeng Gu. 2012. Interactive visibility retargeting in VR using conformal visualization. *IEEE Trans. Visual. Comput. Graphics* 18, 7 (2012), 1027–1040.

[22] Shaoyu Qi and Jeffrey Ho. 2012. Seam segment carving: retargeting images to irregularly-shaped image domains. In *Proc. IEEE Int. Conf. ECCV.* 314–326.

[23] Ramesh Raskar, Jeroen Baar, Thomas Willwacher, and Srinivas Rao. 2004. Quadric transfer for immersive curved screen displays. *Comput. Graph. Forum* 23, 3 (2004), 451–460.

[24] Michael Rubinstein, Ariel Shamir, and Shai Avidan. 2008. Improved seam carving for video retargeting. *ACM Trans. Graph.* 27 (2008), 16:1–9.

[25] Behzad Sajadi and Aditi Majumder. 2011. Autocalibrating tiled projectors on piecewise smooth vertically extruded surfaces. *IEEE Trans. Visual. Comput. Graphics* 17, 9 (2011), 1209–1222.

[26] Olaf Schenk, Matthias Bollhöfer, and Rudolf A Römer. 2008. On Large-Scale Diagonalization Techniques for the Anderson Model of Localization. *SIAM Rev.* 50, 1 (2008), 91–112.

[27] Shai Avidan Shamir and Ariel. 2007. Seam carving for content-aware image resizing. *ACM Trans. Graph.* 26, 3 (2007), 1–10.

[28] Jianbing Shen, Dapeng Wang, and Xuelong Li. 2013. Depth-Aware Image Seam Carving. *IEEE Trans. Cybernetics* 43, 5 (2013), 1453–1461.

[29] Y. Wang, C. Tai, O. Sorkine, and T. Lee. 2008. Optimized scale-and-stretch for image resizing. *ACM Trans. Graph.* 27, 5 (2008), 8:1–8.

[30] Yu-Shuen Wang, Hongbo Fu, Olga Sorkine, Tong-Yee Lee, and Hans-Peter Seidel. 2009. Motion-aware temporal coherence for video resizing. *ACM Trans. Graph.* 28, 5 (2009), 127:136.

[31] Jin Wei, Chen-Feng Li, Shi-Min Hu, Ralph R Martin, and Chiew-Lan Tai. 2012. Fisheye video correction. *IEEE Trans. Visual. Comput. Graphics* 18, 10 (2012), 1771–1783.

[32] Guo-Xin Zhang, Ming-Ming Cheng, Shi-Min Hu, and Ralph R Martin. 2012. A Shape-Preserving Approach to Image Resizing. *Comput. Graph. Forum* 28 (2012), 1897–1906.

[33] J. Y. Zhang, S. W. Li, and C. C. J. Kuo. 2014. Compressed-Domain Video Retargeting. *IEEE Trans. Image. Process.* 23, 2 (2014), 797–809.