Modeling and Rendering of Impossible Figures

TAI-PANG WU
The Chinese University of Hong Kong
CHI-WING FU
Nanyang Technological University
SAI-KIT YEUNG
The Hong Kong University of Science and Technology
JIAYA JIA
The Chinese University of Hong Kong
and
CHI-KEUNG TANG
The Hong Kong University of Science and Technology

This article introduces an optimization approach for modeling and rendering impossible figures. Our solution is inspired by how modeling artists construct physical 3D models to produce a valid 2D view of an impossible figure. Given a set of 3D locally possible parts of the figure, our algorithm automatically optimizes a view-dependent 3D model, subject to the necessary 3D constraints for rendering the impossible figure at the desired novel viewpoint. A linear and constrained least-squares solution to the optimization problem is derived, thereby allowing an efficient computation and rendering new views of impossible figures at interactive rates. Once the optimized model is available, a variety of compelling rendering effects can be applied to the impossible figure.

Categories and Subject Descriptors: I.3.3 [**Computer Graphics**]: Picture/Image Generation; I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis; J.5 [**Computer Applications**]: Arts and Humanities

General Terms: Algorithms, Human Factors

Additional Key Words and Phrases: Modeling and rendering, human perception, impossible figure, nonphotorealistic rendering

ACM Reference Format:

Wu, T.-P., Fu, C.-W., Yeung, S.-K., Jia, J., and Tang, C.-K. 2010. Modeling and rendering of impossible figures. ACM Trans. Graph. 29, 2, Article 13 (March 2010), 15 pages. DOI = 10.1145/1731047.1731051 http://doi.acm.org/10.1145/1731047.1731051

1. INTRODUCTION

Impossible figures (Figures 1 and 2) have long been used in applications such as computer games, nonphotorealistic rendering, image synthesis, and photo-montage [Alexeev 2008]. They are often blended with geometrically possible scenes to create special effects in computer graphics applications.

Modeling and rendering impossible figures has received much less attention in computer graphics, however, possibly due to the following difficulties. First, typically only a single view of an impossible figure (or its image) is provided. Second, it is impossible to build a 3D model that corresponds to the impossible figure without severe structure disconnection or deformation. One example is shown in Figure 3. Even when such a 3D model is meticulously constructed, it can provide one view only [Elber 2002; Lipson 2002]. This is because an impossible figure is a special kind of 2D drawing that captures *locally possible* 3D parts, but the overall geometry is *globally inconsistent* without structure deformation. See in particular Figure 2(b) for the nine-cube arrangement that was first drawn by Reutersvard in 1934. When we view the entire

The research was supported in part by the Hong Kong Research Grant Council under grant numbers 412307 and 620309 and MOE AcRF Tier1 Grant with project number RG 13/08 M5202000.

Authors' addresses: T.-P. Wu, Computer Science and Engineering Department, the Chinese University of Hong Kong, Shatin, N. T., Hong Kong; email: tpwu@cse.cuhk.edu.hk; C.-W. Fu, School of Computer Engineering, Nanyang Technological University, Singapore, 639798; email: cwfu@ntu.edu.sg; S.-K. Yeung, Computer Science and Engineering Department, the Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong; email: saikit@cse.ust.hk. J. Jia, Computer Science and Engineering Department, the Chinese University of Hong Kong, Shatin, N. T., Hong Kong; email: leojia@cse.cuhk.edu.hk; C.-K. Tang, Computer Science and Engineering Department, the Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong; email: cktang@cse.ust.hk.

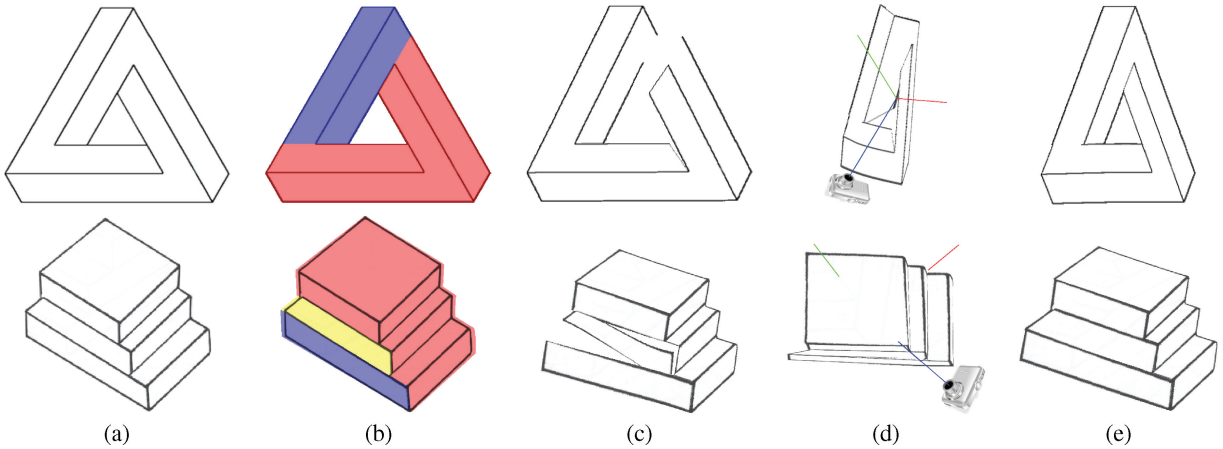


Fig. 1. (a) The impossible figure is segmented into parts (as shown in (b)), which are possible in 3D. (c) When the camera viewpoint is changed, the figure starts to collapse. Our system automatically optimizes at interactive rate a *view-dependent* 3D model for rendering the impossible figure at the novel view. Such a model looks distorted, as shown in (d), except at the camera viewpoint specified by the user, as shown in (e). Shown in the top is penrose triangle and the bottom is impossible staircase.

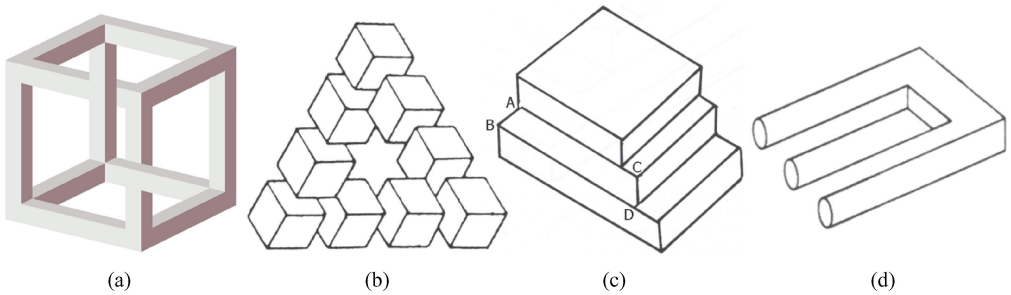


Fig. 2. Representative examples of the four basic classes of impossible figures: (a) the impossible cuboid, (b) the nine-cube arrangement, (c) the impossible staircase, and (d) the impossible bar.

drawing altogether, inconsistency in the global structure confuses our visual perception on the model geometry.

In this article, we present an interactive system for modeling and rendering impossible figures. Refer to Figure 1: the user first segments a 2D view of the impossible figure and models a set of locally possible 3D parts. The segmentation makes it easy to model each part, which is often rectilinear, using existing modeling tools. On the same 2D view, a few connection and collinearity constraints are then specified. This segmentation and constraint specification step is done only once. As the user changes the camera viewpoint, the system automatically optimizes a *view-dependent* 3D model that connects the parts and produces the impossible figure at the desired novel viewpoint.

Our main technical novelty lies in the automatic optimization algorithm (Section 4.2) which connects individual 3D parts to generate the 2D impossible figure. To guarantee the output to be an impossible figure, all the computations performed on the 3D parts are subject to the constraints which maintain straight connections on the 2D projected view. For example, similar to Figure 3, although the 3D geometry may be severely deformed in the 3D space after optimization, the rendered impossible figure still looks plausible specifically at the novel camera viewpoint.

Interestingly, the automatic optimization turns out to have a straightforward and *linear* least-squares solution, thereby allowing efficient computation to support interactive modeling and rendering (see the accompanying video accessible in the ACM Digital Library). In this article, we show examples on how to model and render the following classes of impossible figures [Ernst 1987].

—*Depth Interposition.* See the impossible cuboid in Figure 2(a), where the optical illusion is caused by structural inconsistency due to the problematic depth ordering.

—*Depth Contradiction.* Refer to the nine-cube arrangement in Figure 2(b), and also the Penrose triangle by the Penroses [1958] in Figure 1, where propagation of local 3D information gives rise to global structural inconsistency.

—*Disappearing Normals.* See Figure 2(c) for an example of the impossible staircase, where the plane *ABCD* depicted in the figure could appear to be horizontal at one side while vertical at the other, making it impossible to assign a consistent normal across the whole plane.

—*Disappearing Space.* See Figure 2(d) where the silhouette of the impossible trident is not closed.

M.C. Escher [M.C. Escher Foundation ; Schattschneider and Emmer 2003] was a renowned art master who popularized impossible figures by skillfully embedding models of impossible figures in architectural drawings. Figure 4 presents more impossible figures used in this article. *Ascending and Descending*, *Double Penrose Triangles*, *Waterfall* and *Construction* were created based on depth

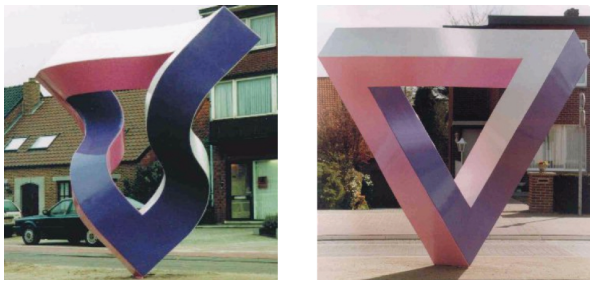


Fig. 3. Only one view of the impossible figure can be produced from the corresponding 3D model with heavily twisted structure.

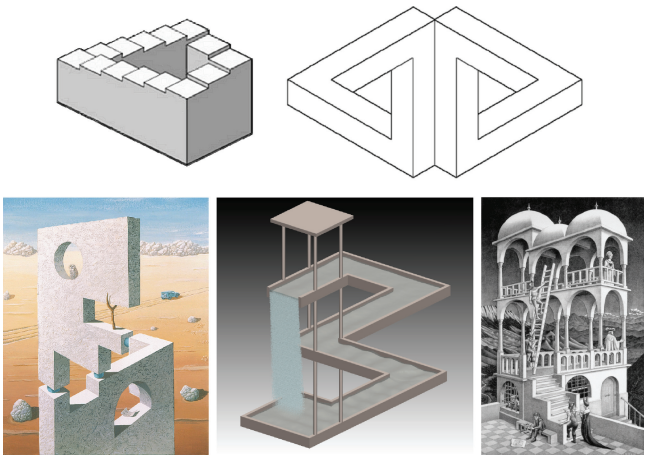


Fig. 4. Top: *Ascending and Descending* (left) and *Double Penrose Triangles* (right). Bottom: *Construction* (left), *Waterfall* (middle), and M. C. Escher's original *Belvedere* (right).

contradiction. *Belvedere* was created based on the depth interposition technique demonstrated in the impossible cuboid.

2. LITERATURE REVIEW

A plausible approach to rendering an impossible figure is to skip the 3D modeling step and generate a novel view by 2D warping of the input view, as typically only one view of the impossible figure is given. In fact, 2D artists have experimented with simple tricks such as 2D scaling and stretching. However, when the novel view is far from the reference view, 2D warping fails in making the transformed figure look like a 3D solid. Another advantage of a 3D approach is that temporal coherence in animating an impossible figure is automatically and smoothly maintained, as a 3D model is available for animation.

Recognizing the importance of operating in the 3D space, 3D geometric approaches have been proposed. Elber [2002] created physical models for impossible figures, including those by M.C. Escher. Lipson [2002] made use of LEGO bricks to build physical models of various impossible figures rendered by Escher. Manual construction of such a 3D model can be a tedious process. Moreover, the resulting model only allows rendering the impossible figure at restrictive viewpoints. In contrast, our general approach takes both 2D and 3D into consideration, which can be used to render impossible figures in any one of the four classes. Also, our efficient optimization algorithm models and renders impossible figures at interactive speed.

2.1 Approaches

Technically, our work is closely related to Rademacher [1999] where view-dependent geometry was introduced. The work was originated from a cel animation application where a cartoon character drawn by an artist exhibits some extent of distortion in successive keyframes. Since there are no consistent 3D models capable of representing exactly the object at the corresponding viewpoints, deformations are needed to match the input geometry (base model) and the drawings. In addition to the base model, however, their method requires the user to input a set of key deformations and a set of key viewpoints for each keyframe. Using this system, even if it is possible to produce novel views for an impossible figure, specifying the two sets of information is tedious. If it is indeed impossible, then the artist needs to imagine and draw up the novel views (keyframes) of the impossible figures. This is difficult, as we have no idea how many keyframes are required for different types of impossible figures.

One may also relate our work to multi-perspective rendering [Yu and McMillan 2004]. In fact, we share somewhat the same spirit, but we work on a different problem domain and operate with different techniques: multi-perspective rendering blends different views of images into a single one so that the user can see the whole object/scene in a single picture. Our method blends possible 3D parts by nonrigid transformation with the consideration of local structure consistency (against the constraints) to produce the optical illusion subject to viewpoint changes.

For artwork of impossible figures, Simanek [1996] presented a proposal to create false perspectives on impossible figures; simple tricks such as prolonging the horizontal dimension were used to create stereo pairs of impossible figures. Savransky [1999] described impossible 3D scenes by encoding linear transformations between neighboring 3D parts in the impossible 3D world, and then by solving for a correct viewpoint (a modelview transformation with respect to the camera) that optimally projects the impossible scene. Uribe [2001] proposed to use a set of triangular tiles to design and create 2D impossible figures. This method was applied in rendering one of the levels in the computer game Diablo II by Blizzard Entertainment.

2.2 Related Work

Khoh and Kovesi [1999] generated novel views of impossible figures by using two complementary halves, which are 3D models related by an inversion transform in the image plane. The thickness of the two complementary halves needs to be adjusted after inversion transform. Their approach works for a particular subset of impossible figures. A similar approach was adopted by Tsuruno [1997], where a 3D model was constructed to create different views of *Belvedere*. Sugihara [1986; 1997] provided a foundation for interpreting line drawing for possible objects. Unfolded surfaces that are geometrically possible can be generated for allowing one view of an impossible figure to be projected. The main goal of this article is to produce physical toys from impossible figures. Owada and Fujiki [2008] proposed a system for modeling impossible figures. They also implemented a constraint solver to seamlessly combine multiple 3D parts in a projected 2D domain, by considering 3D line orientations to render an impossible figure at novel viewpoints. Their working prototype, however, used nonlinear optimization and only allowed a narrow range of viewpoints. Our optimization, on the other hand, has a linear least-squares solution that can

Table I. A Comparison between Various Rendering and Modeling Approaches

Method	Classes	Geometric Input	User Input	Optimization	Camera Path	Rendering
Tsuruno [1997]	1	3D meshes	unknown	unknown	designated	PR
Savransky [1999]	1,2	3D meshes	transformation	linear	estimated and fixed	NPR, PR
Khoh and Kovesi [1999]	1	3D lines	unknown	linear	unrestricted	LA
Owada and Fujiki [2008]	1,2,3,4	3D meshes	stroke/dragging/dialog	non-linear	restricted	LA
Our Method	1,2,3,4	3D meshes	point/line of correspondence	linear	restricted	LA, NPR, PR

Classes indicate the type of impossible figures that the methods could handle—1: depth interposition, 2: depth contradiction, 3: disappearing normals and 4: disappearing space; LA, NPR and PR stand for Line-Art, NonPhotorealistic Rendering, and Photorealistic Rendering, respectively.

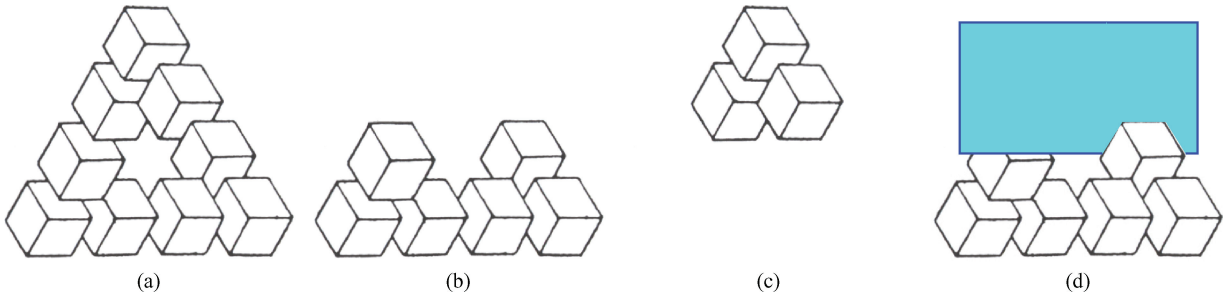


Fig. 5. (a) Nine-cube arrangement; (b) and (c) are the two possible parts of (a); (d) a tilted plane is placed as shown in the 3D model. This plane serves as the image plane where (c) is projected to produce the impossible figure in (a).

be efficiently computed. Moreover, as we will explain in the sequel, our approach can quantitatively identify the subset of viewpoints where the impossible figure ceases to exist.

An ideal system should allow the user to model different classes of impossible figure within a unified framework and using minimal effort. Because impossible figures still look plausible under specific viewpoints, we should still categorize impossible figures in ways similar to conventional graphics rendering such as Line-Art (LA), NonPhotorealistic Rendering (NPR), and PhotoRealistic Rendering (PR). Table I summarizes the characteristics of the approaches designed to model and render impossible figures.

It can be noticed that Tsuruno [1997][1] and Khoh and Kovesi [1999] are capable of handling one class of impossible figures only; specifically, they showed only one instance in their respective publication.

On the other hand, our method outperforms previous methods in terms of input (the type of impossible figures that can be handled) and output (the type of rendering techniques that can be applied). While Owada and Fujiki [2008] proposed a competitive method, we show that our stable and linear optimization method (using constrained thin-plate spline warping) can provide a much higher frame rate (several versus ∼40 frames per second) and support interactive rendering, which are instrumental to game and movie applications.

Regarding user input, Savransky [1999] requires the user to manually specify local and rigid transformations (rotation and translation), for every pair of relating 3D parts. Owada and Fujiki [2008] require the user to perform edge deletion (via brush stroke), 3D part stitching (via mouse dragging), and depth and orientation control (via dialog). Different from these two approaches, we employ point and line correspondences to specify the necessary constraints in a static view in the modeling stage. Such a specification need only be done once and no further editing is required after that. We will discuss the advantages of our user inputs as compared with the two approaches mentioned earlier in Section 5.2.

One drawback of our system is that, similar to Owada and Fujiki [2008], the set of viewpoints is restricted. This is possibly due to the fact that impossible figures are only feasible at a finite set of viewpoints. Although the method proposed by Khoh and Kovesi [1999] does not have viewpoint restriction, the rendered object suffers from severe distortion, especially at some degenerate viewpoints.

3. ANALYSIS

The human has a remarkable ability to make instant connection to 3D on seeing 2D drawings. This connection, however, can lead to interesting problems. When presented an impossible figure before our eyes, we can perceive local 3D structure of individual parts in the drawing. When we view the entire drawing as a whole, structural inconsistency arises and we become aware that the figure is invalid as we attempt to mentally build a globally consistent 3D structure from the impossible figure.

3.1 Dissecting an Impossible Figure

We believe that the visual confusion caused by viewing an impossible figure lies in the presence of multiple 3D structures projected by using *different* transformations; some of them violate the laws of rigid camera transform, whereas the resulting projected 2D structures still retain 3D semantics.

For example, if we separate the upper three cubes from the rest in Figure 5(a), as shown in 5(b) and 5(c), the two subfigures correspond to 3D rectilinear structures and can be respectively obtained using an orthographic camera. Now, suppose we put a tilted plane between the two cubes in the 3D model of Figure 5(b), thereby producing 5(d). Then, we project orthographically the 3D model of Figure 5(c) onto the plane. Combining the resulting

[1]Tsuruno [1997] is published in the form of a video clip, and the technical detail for producing this video is not published anywhere else. Moreover, no follow-up from the same group has appeared afterward. So, we have no information on the technique they used.

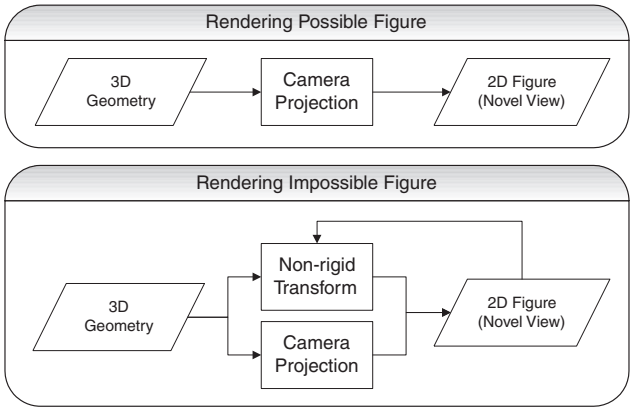
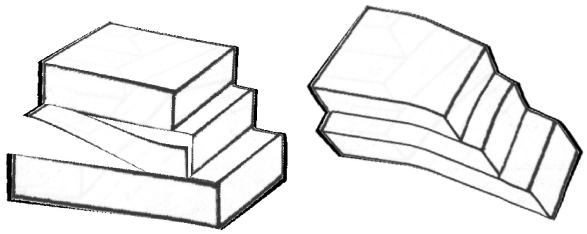


Fig. 6. Rendering possible and impossible figures.



(a) (b)

Fig. 7. Two renderings respectively dominated by (a) rigid transformation and (b) nonrigid transformation.

image with Figure 5(b), we now obtain Figure 5(a), the nine-cube arrangement.

In addition to using a rigid standard camera, this example demonstrates that nonstandard or unconventional projective transformation is needed to create the impossible figure. While the orthographic camera is one form of rigid transformation, the preceding tailor-made procedure (that is, the usage of a tilted plane and combination of images) can be characterized by a more general, nonrigid transformation.

To our knowledge, there is no camera model available to generate an impossible figure by appropriately combining rigid and nonrigid transformation.

3.2 Rigid and Nonrigid Transformation

The preceding analysis suggests that a reconciliation process is required of rigid and nonrigid transformation in producing an impossible figure. Specifically, while a novel view should be specified with respect to a rigid camera, it is necessary to incorporate nonrigid transformation to connect multiple 3D structures so as to produce global structural inconsistency. The nonrigid transformation, on the other hand, must be constrained so that straight line connections are maintained at the novel view. We term such constrained nonrigid transform *view-dependent modeling*.

Figure 6 illustrates the differences between the possible and impossible figure in terms of modeling and rendering. Note that the former is a standard image formation process, while the latter (view-dependent modeling) involves the projected 2D view with constraints on how the 3D model should be transformed to achieve global structure inconsistency.

Enforcing appropriate constraints in nonrigid transformation is essential in constructing impossible figures. Figure 7 illustrates the situations where either rigid (projective camera) transformation or nonrigid transformation dominates the rendering process. Figure 7(a) shows the impossible staircase rendered at a novel viewpoint that maximizes the size of the rigid parts. The 3D perception is strong but the figure is no longer impossible at the novel view. On the other hand, Figure 7(b) shows a novel view generated by free-form warping of the input figure. Though preserving the topology, the figure is curved and does not plausibly resemble a novel view derived from the same object that produces the original drawing.

4. VIEW-DEPENDENT MODELING

Typically, we are only given a single view of an impossible figure as input. In this case, after segmentation, the user constructs a 3D model for each possible part (Section 4.1). Then, the user marks up 3D constraints in the form of points and lines on the 2D figure. A view-dependent model will be automatically optimized for rendering the impossible figure at the novel view (Section 4.2). Please also refer to the real-time capture in the accompanying video.

4.1 Segmentation and Modeling of 3D Possible Parts

We provide one simple segmentation strategy for each of the four classes of impossible figures. The goal is to produce a set of locally possible parts that can be modeled using existing modeling tools such as Maya and 3D Studio Max, or systems like Ju et al. [2007], Nealen et al. [2007], and Wu et al. [2007]. In some demonstrated examples, a height field rather than a full 3D model was used. The tools used in segmentation and modeling are not the focus of our work.[2]

Refer to Figure 1. The Penrose triangle is decomposed into two possible polyhedral parts. The problematic plane is segmented from the impossible staircase, making both the resulting polyhedron and the plane feasible in 3D. To make the 3D model consistent with the original view, for the Penrose triangle, we need curved surfaces but flat normals, while for the impossible staircase, we need a flat surface but curved normals.

Next, refer to Figure 8. The problematic corner of the impossible cuboid (the inverted Y-shaped component) is disconnected from the cuboid, making the two independent parts simple 3D models. This corner will be treated as the foreground layer, and the rest of the cuboid as background layer. For the impossible trident, the problematic bars are disconnected from the object body, making all segmented parts individually possible in 3D.

One may notice that multiple segmentation strategies exist for a given impossible figure. Ideally, we seek one which contains at least one region whose size is the maximum. This largest region will be considered as the rigid part where rigid transformation will be applied. The rationale for this strategy is that nonrigid transformation of other parts, which may reduce the effect of 3D solid perception, will thus be minimized as much as possible. The tradeoff between rigid and nonrigid transformation was illustrated in Figure 7. While more than one segmentations are feasible, our efficient and automatic optimization system makes it easy for the user to experiment with different strategies.

[2]In fact, automatic detection of global impossibility and performing "proper" segmentation remains elusive and limited to line drawings. Interested readers may refer to Huffman [1971], Sugihara [1982], and Heyden [1996] for more detail.

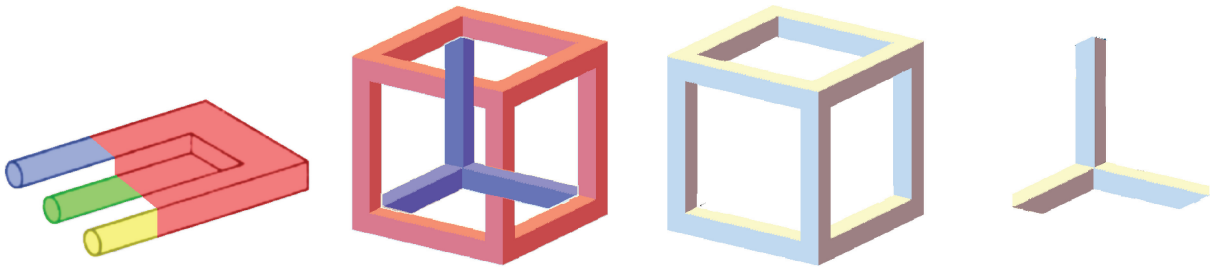


Fig. 8. Parts segmentation for impossible trident and impossible cuboid. See Figure 1 for Penrose triangle and impossible staircase.

4.2 Automatic Optimization

Refer to Figure 1. The user selects one part as the *reference part* (shown in red), and moves the rigid camera to the desired viewpoint. The selected reference part is rendered as a rigid body in the novel view and no nonrigid deformation is applied to it.

The other parts will undergo nonrigid transformation to optimize a connected 3D model that produces the impossible figure at the desired novel viewpoint. Because a view-dependent model can provide only one view, to render a sequence of novel views, the computation needs to be automatic and efficient.

In this section, we describe our efficient algorithm which connects the 3D segmented parts so as to create the impossible figure at the novel view. We propose to implement such parts connection by applying Thin-Plate Spline (TPS) warping [Bookstein 1989] in the 3D domain, subject to the criteria necessary for a 2D impossible figure. We choose TPS because it minimizes the Laplacian (or curvature) of the warping energy, where a natural and smooth deformation can be obtained. This smoothness property also allows graceful degradation when the solution does not exist at certain viewpoints (more detail in the discussion section). Besides, it is well known that the deformation of TPS is smooth and stable with respect to changes in input even without explicitly enforcing temporal coherence, so we have less degree of freedom to consider. Moreover, the TPS model is computationally efficient. Also, note that we warp the 3D parts in the 3D space instead of warping in the projected 2D domain because, as we will see, operating in 3D allows us to readily handle depth ordering.

In the following, we first provide a concise review of TPS, and then define the constraints to the TPS solution to achieve view-dependent modeling.

4.2.1 *Review of Thin-Plate Spline Warping.* Let $\mathbf{p} = (x, y, z)^T$ and $f(\mathbf{p}) : \mathbb{R}^3 \longmapsto \mathbb{R}^3$ be a mapping function, TPS warping in 3D is defined by

$$f(\mathbf{p}) = \mathbf{a}_1 + x\mathbf{a}_2 + y\mathbf{a}_3 + z\mathbf{a}_4 + \sum_i^s \mathbf{w}_i U(||\mathbf{p}_i - \mathbf{p}||), \quad (1)$$

where $U(r) = -|r|$ in 3D [Wahba 1990], s is the number of input sites which is equal to the number of matching point pairs.[3] $\{\mathbf{a}_j \in \mathbb{R}^3 | j = 1, 2, 3, 4\}$ and $\{\mathbf{w}_i \in \mathbb{R}^3 | i = 1, \ldots, s\}$ are the model parameters to be estimated.

Denote $\mathbf{T} = (\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4, \mathbf{w}_1, \ldots, \mathbf{w}_s)$, which is a $3 \times (s+4)$ matrix, to be the unknown parameters. Denote $\mathbf{v} = (1, \mathbf{p}^T, U(||\mathbf{p}_1 - \mathbf{p}||), \ldots, U(||\mathbf{p}_s - \mathbf{p}||))^T$, which is a $(s+4)$ column vector, to be the known input. Eq. (1) can be written as the following matrix form.

$$f(\mathbf{p}) = \mathbf{T}\mathbf{v} \quad (2)$$

Suppose that we have a discrete set of matching samples $\{(\mathbf{p}_i, \mathbf{m}_i)\}$ such that $\mathbf{p}_i \in \mathbb{R}^3 \longrightarrow \mathbf{m}_i \in \mathbb{R}^3$, where $\{\mathbf{p}_i\}$ is the set of input sites and $\{\mathbf{m}_i\}$ is the set of mapping targets, and $\mathbf{v}_i = (1, \mathbf{p}_i^T, U(||\mathbf{p}_1 - \mathbf{p}_i||), \ldots, U(||\mathbf{p}_s - \mathbf{p}_i||))^T$, we can estimate the model parameter $\mathbf{T}$ by solving the following set of linear equations.

$$\mathbf{T}\,(\mathbf{v}_1 \ \cdots \ \mathbf{v}_s) \ = \ (\mathbf{m}_1 \ \cdots \ \mathbf{m}_s) \quad (3)$$

Standard TPS considers the null space of the input sites [Bookstein 1989]; that is, Eq. (3) has to be solved subject to the following condition:

$$\mathbf{T}\,(\mathbf{O} \ \mathbf{p}'_1 \ \cdots \ \mathbf{p}'_s)^T = \mathbf{0}, \quad (4)$$

where $\mathbf{O}$ is a 4×4 zero matrix and $\mathbf{p}'_i = \begin{pmatrix} 1 \\ \mathbf{p}_i \end{pmatrix}$. Given the matrix forms shown in Eqs. (3) and (4), we are ready to derive the conditions for our view-dependent modeling.

4.2.2 *Connection Constraint.* The resulting impossible figure must be connected in the rendered 2D view. Without loss of generality, suppose we have a set of n matching point pairs $\{(\mathbf{p}_{ik}, \mathbf{p}_{ig}) | i = 1, \ldots, n\}$ that corresponds to parts k and g. After TPS transformation, $\mathbf{p}_{ik}$ and $\mathbf{p}_{ig}$ have to be connected in 3D, which automatically enforces 2D connection in the projected camera view.

Let $\mathbf{T}_k$ and $\mathbf{T}_g$ be the model parameters corresponding to the mapping functions for parts k and g. We have two cases to consider as shown in Figure 10: (1) the connection between the reference part and a deformable part, and (2) the connection between two deformable parts.

Case 1. Suppose that part g is the reference part, according to Eq. (3), we have.

$$\mathbf{T}_k\,(\mathbf{v}_{1k} \ \cdots \ \mathbf{v}_{nk}) \ = \ (\mathbf{p}_{1g} \ \cdots \ \mathbf{p}_{ng}) \quad (5)$$

Case 2. When no reference part is involved in the connection, we cannot assume either $\mathbf{p}_{ik}$ or $\mathbf{p}_{ig}$ to be the mapping target, as in Case 1. This is because parts k and g will be deformed simultaneously. Instead, we minimize the 3D distance between these two points.

$$\mathbf{T}_k\,(\mathbf{v}_{1k} \ \cdots \ \mathbf{v}_{nk}) \ - \mathbf{T}_g\,(\mathbf{v}_{1g} \ \cdots \ \mathbf{v}_{ng}) \ = \mathbf{0} \quad (6)$$

Similar to the derivation of Eq. (4), we need to consider the null space of these input points:

$$\mathbf{T}_\Lambda\,(\mathbf{O} \ \mathbf{p}'_{1\Lambda} \ \cdots \ \mathbf{p}'_{s\Lambda})^T = \mathbf{0}, \quad (7)$$

[3] From a user's perspective, input sites are the user-clicked points; see Section 5.2.

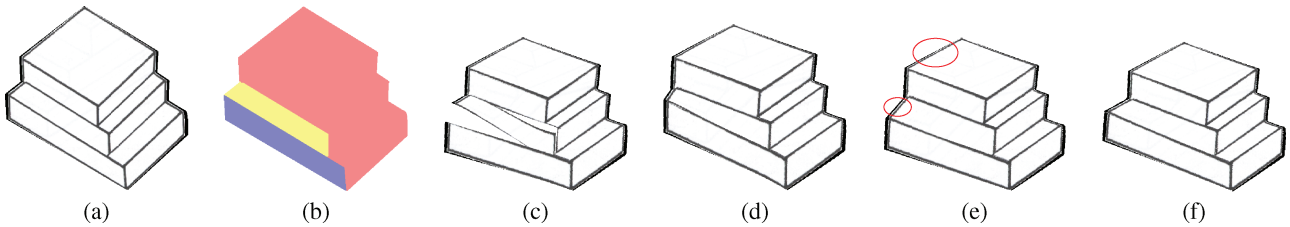


Fig. 9. (a) Impossible staircase; (b) segmentation; (c) a novel view before optimization; (d) a novel view optimized subject to the connection constraint, where the yellow region deforms severely while the red region does not; (e) a novel view optimized subject to both the connection and collinearity constraints. Compared with (a), the two highlighted structures should be parallel; (f) a novel view optimized subject to the connection, collinearity, and parallel constraints.

where $\Lambda \in \{k, g\}$. Eqs. (5), (6), and (7) together constitute a set of linear equations that enforces the basic connectivity in the optimized 3D model. Figure 9(c) and 9(d), respectively, show the results before and after applying the connection constraint at a novel view.

4.2.3 *Collinearity Constraint*. Since each part may be deformed by different mapping functions, the resulting deformation can be biased to a certain part, resulting in uneven deformation and unwanted distorted appearance. Specifically, there is no guarantee that after TPS warping the transformed points in the contact area will have the same gradient. For example, in Figure 9(d), the corresponding yellow region shown in Figure 9(b) undergoes large distortion while the red region undergoes small deformation.

To eliminate such artifacts on the resulting 2D figure, we need to minimize the changes of the mapping functions at the matching points. Let $\mathbf{C} = (\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)^T$ be the known 3×4 camera projection matrix corresponding to the novel viewpoint. Then, for each matching point pair $(\mathbf{p}_{ik}, \mathbf{p}_{ig})$, we have

$$\begin{aligned} \frac{\partial}{\partial x}\mathbf{c}_1^T \begin{pmatrix} f_k(\mathbf{p}_{ik}) - f_g(\mathbf{p}_{ig}) \\ 0 \end{pmatrix} = 0, \\ \frac{\partial}{\partial y}\mathbf{c}_2^T \begin{pmatrix} f_k(\mathbf{p}_{ik}) - f_g(\mathbf{p}_{ig}) \\ 0 \end{pmatrix} = 0, \end{aligned} \tag{8}$$

where $f_k(\cdot)$ is the mapping function for part k. This equation explicitly minimizes the difference in the 2D gradient across the contact area of the mapping functions on the screen space. Figure 9(e) shows the result after applying the collinearity constraint, where the deformation is not biased to any of the two nonreference parts (colored yellow and blue in 9(b)).

4.2.4 *Parallel Constraint*. Since TPS warping is used, the underlying structure of the parts can be deformed severely. While this is allowed in 3D, we need to constrain the projected 2D structures to protect the shape from apparent distortion while achieving the global structure inconsistency. To this end, we impose the parallel constraint during TPS warping. For example, there is apparent distortion in the figure shown in Figure 9(e) when compared with the input: the circled structures in 9(e) should be parallel to each other, similar to 9(a).

Recall that the selected reference part remains rigid throughout the optimization; otherwise, enforcing parallelism will be elusive in TPS warping when all parts are allowed to deform simultaneously. To specify the parallel line constraint, the user marks up on the 2D view a pair of corresponding lines between the reference part (rigid) and the deformable parts (nonrigid). An example is shown in Figure 11. Along the pair of corresponding lines, similarity in 2D gradients will be maximized. Mathematically, we set up the following set of linear equations for achieving this goal:

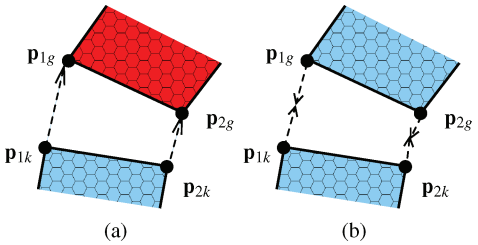


Fig. 10. Connection constraint. (a) case 1: the points in the blue region approach to the points in the red region (the reference part); (b) case 2: the corresponding points approach each other.

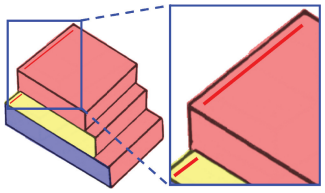


Fig. 11. Parallel (line) constraint. The pair of red lines are corresponding.

$$\mathbf{c}_s^T \begin{pmatrix} f_k(\mathbf{l}_a) - f_k(\mathbf{l}_b) \\ 1 \end{pmatrix} = \mathbf{c}_s^T \begin{pmatrix} \mathbf{l}'_a - \mathbf{l}'_b \\ 1 \end{pmatrix}, \tag{9}$$

where $\mathbf{C} = (\mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3)^T$ is the 3×4 camera projection matrix and $s \in \{1, 2\}$, $\mathbf{l}_a$ and $\mathbf{l}_b$ are the two endpoints of a line in part k, $\mathbf{l}'_a$ and $\mathbf{l}'_b$ are the two endpoints of the line in the reference region. Figure 9(f) shows the final result after applying the parallel constraint.

4.2.5 *Optimization*. Mathematically, the set of linear equations in Eqs. (5)–(9) can be combined into the following form:

$$\mathbf{A}\mathbf{h} = \mathbf{b}, \tag{10}$$

where a *linear* least-squares solution exists. $\mathbf{A}$ is the matrix containing the set of coefficients for calculating the Gram matrix[4] (i.e., $\mathbf{A}^T\mathbf{A}$). For the dimensions, $\mathbf{A}$ is an $N \times 3(s+4)$ matrix while $\mathbf{h}$ and $\mathbf{b}$ are $3(s+4)$ column vectors, where N is the number of constraint equations formed using Eqs. (5)–(9) and s is the number of input sites.

[4]In general, given a set of vectors $\mathcal{G}$, the Gram matrix $\mathbf{G}$ that corresponds to $\mathcal{G}$ is the matrix of all possible inner products of $\mathcal{G}$, that is, $[\mathbf{G}]_{ij} = \mathbf{g}_i^T\mathbf{g}_j$, where $\mathbf{g}_i$ and $\mathbf{g}_j \in \mathcal{G}$.

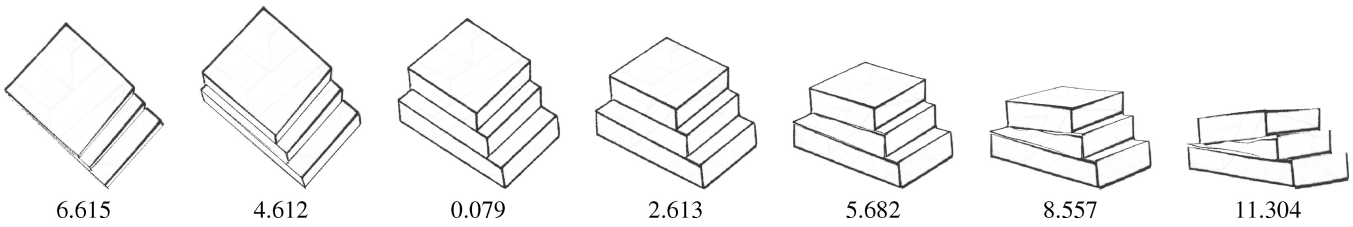


Fig. 12. Error measurement of sample novel views of the impossible staircase. The errors shown here are calculated by applying Eq. (11), followed by multiplying the result by 1000.

Directly expanding the linear system expressed by Eq. (10), on the other hand, is complicated and unnecessary: in our implementation, we derive the Gauss-Seidel solutions directly from Eqs. (5)–(9) and combine them, which is a common strategy adopted for a simpler and faster implementation requiring less memory storage. To make the solution more stable, all the 3D inputs are normalized in the range of $[-1, +1]$ before the computation.

The Gauss-Seidel method provides two advantages over the direct method: (1) it can be used to handle a very large linear system when the direct method is not practical (e.g., the Gram matrix can be a $1000^2 \times 1000^2$ matrix, which is very typical in imaging problems such as Poisson matting [Sun et al. 2004]); and (2) it is capable of constraining the solution space by imposing inequality, which is useful for handling depth ordering.

As mentioned in Section 4.2, we may need to handle objects like the impossible cuboid which has two depth layers. To restrict the depth of a region within some range, for example, the center portion of the inverted Y-shaped component has to be displayed in front of the background layer, we can impose *range constraints* on the z-coordinate (e.g., $z > q$ where q is a constant) in the solver when solving Eq. (10). This can easily be done when a Gauss-Seidel solver (with/without successive overrelaxation) is used. In doing so, the center of the inverted Y-shaped component will always be located in front of the background layer, while the three extreme ends will be connected to the background layer.

4.2.6 *Error Measurement.* Different from standard TPS, our solution is a least-squares solution with an overconstrained system. As a result, for some viewpoints, the optimized solution $\mathbf{h}$ to Eq. (10) may not maintain the strict equality. This means that some of the constraints may not be adequately satisfied, which causes the impossible figure to collapse.

In practice, we can quantify the validity of the optimized result by computing the Root-Mean-Square (RMS) error between the optimized 3D model and the constraints encoded in Eq. (10), that is,

$$\sqrt{\frac{||\mathbf{Ah} - \mathbf{b}||^2}{N}}, \tag{11}$$

where N is number of rows in $\mathbf{A}$, which is equal to the number of constraint equations formed using Eqs. (5)–(9). As mentioned, it is not necessary to expand the matrix $\mathbf{A}$. To calculate the numerator of Eq. (11), we sum up the squared errors of each equation formed using Eqs. (5)–(9). The RMS errors for some sample novel views of the impossible staircase are shown below the subfigures in Figure 12. At extreme viewpoints, we may not be able to maintain the straight line structure in the result where the impossible figure ceases to exist (indicated by Eq. (11)). The viewpoints corresponding to large errors can therefore be pruned away and labeled as invalid.

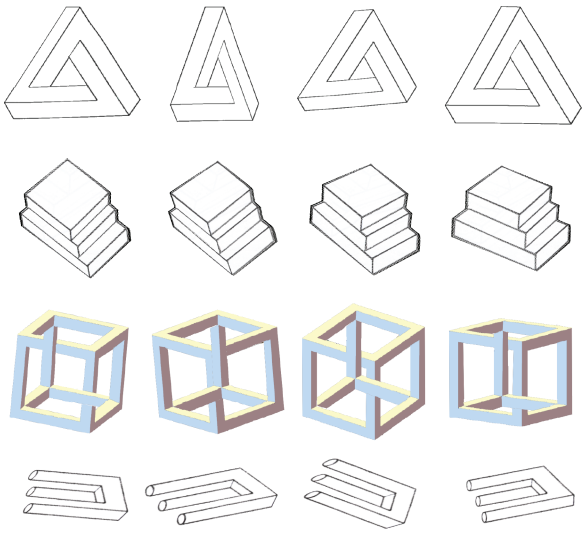


Fig. 13. Novel views of basic impossible figures.

Table II. Average Modeling and Rendering Computing Time (in seconds)

	Modeling (FPS)	Rendering (FPS)	Total
Cuboid	0.05 (18.60)	0.02 (44.61)	0.08 (13.13)
Penrose	0.07 (15.01)	0.02 (41.08)	0.09 (10.99)
Staircase	0.04 (24.76)	0.03 (39.39)	0.07 (15.20)
Trident	0.14 (7.32)	0.01 (87.56)	0.15 (6.76)

All times are measured on a PC (Intel Core 2 Quad CPU Q9400 running at 2.66 GHz), with 4GB RAM and Geforce 9800 graphics board. The corresponding Frame Rates per Second (FPS) are also shown.

4.3 Basic Impossible Figures

Figure 13 shows four novel views for the basic impossible figures. Table II tabulates the running time of our system on modeling and rendering basic impossible figures. Note again that the user marks up constraints only once. The view-dependent models are then automatically generated at the novel views.

For the Penrose triangle, the collinearity constraint is instrumental in preserving the structure linearity in the projected figure, because the parts segmentation cuts the two silhouette edges as shown in Figure 1. We have demonstrated how the three constraints operate using the impossible staircase as the running example.

For the impossible cuboid, the foreground layer is optimized with the range constraint described in Section 4.2.5. Similar to the Penrose triangle, the collinearity constraint is essential to the impossible trident in preserving the linear structure during TPS

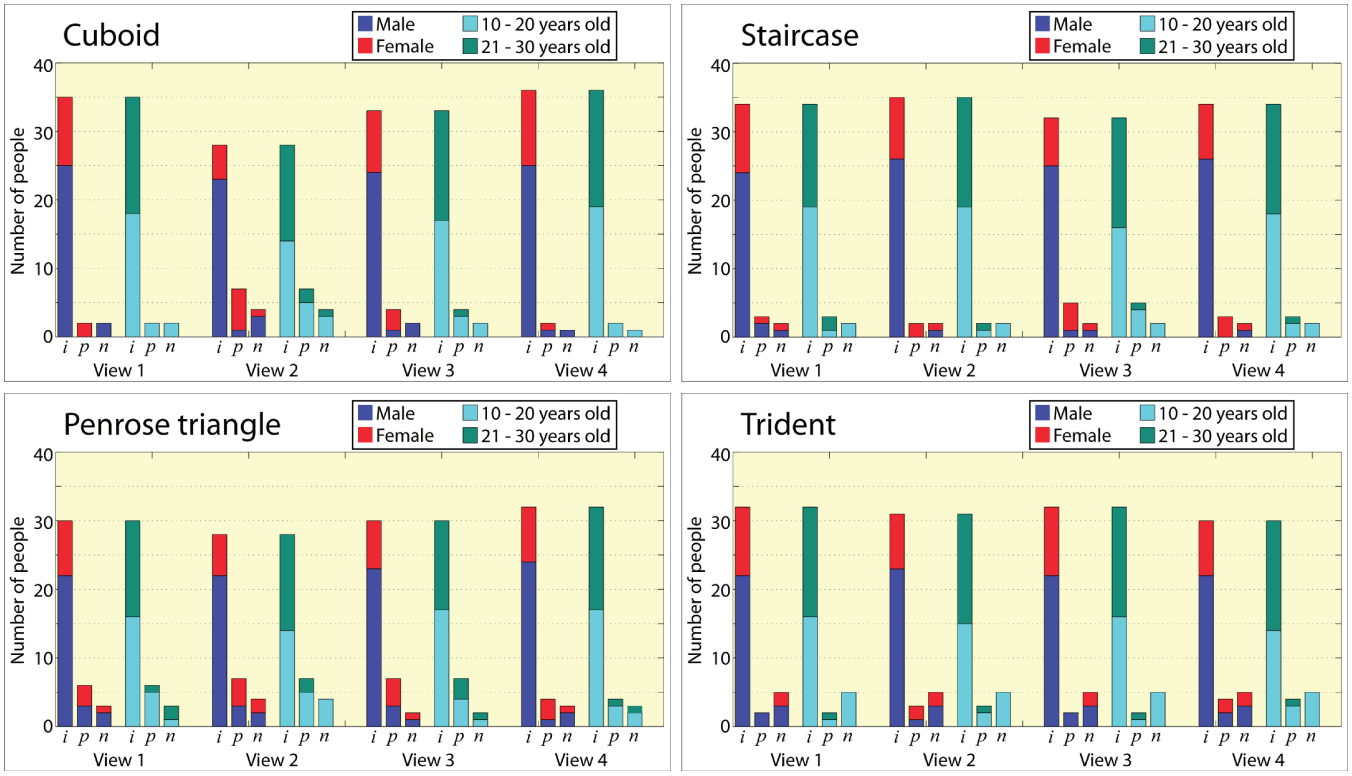


Fig. 14. Survey results. For each basic impossible figure, five views were presented: the original view and four novel views derived from the original view using our system. Users were asked to label each image as either impossible (i), possible (p), or not sure (n).

warping, because its parts segmentation cuts across the three bars (Figure 8) to produce the corresponding locally possible parts. In addition, the parallel constraint preserves the parallelism of the three bars in the resulting novel views.

4.4 Survey

Human visual perception on impossible figures can be subjective and varies widely. We conducted a user survey to investigate the quality of novel views of impossible figures generated by our system.

After an explanation of impossible figures is given, the subjects were first presented the original input of the four basic impossible figures (Penrose triangle, impossible cuboid, impossible staircase, and impossible trident). The order of showing was randomized across different subjects to avoid bias. If they can articulate why the input figure is impossible in 3D, the experiment would proceed to the next phase, where they were presented four novel views of the respective four basic impossible figures generated by our system (some of them are shown Figure 13). Again, the order of showing was randomized. The users were asked to label them either as "impossible," "possible," or "not sure" within a time limit of 20 seconds.

A total of 47 persons in different age groups and genders participated in our survey; 39 persons proceeded to the next phase. Figure 14 shows the result: we found that over 76% of users label our figures as "impossible." A vote of "not sure" is cast when the user cannot decide within the time limit. In general, we found that if the users were able to articulate why the original figure is impossible in 3D, they would be able to label the novel views as impossible as well.

Among the four cases, it requires the least effort for participants to label the impossible cuboid, which also scores the highest on being "impossible" among all the figures. On the other hand, a participant requires on average the most time in labeling the Penrose triangle, which also scores the lowest on being "impossible." Some users commented after the survey that the figures are "fantastic" and "interesting to view"; some are intrigued by the confused 3D perception (because they were actually seeing an image of a 3D model, albeit a view-dependent one optimized by our system), while others even proposed how to use overlay photography to create an impossible cuboid figure.

5. DISCUSSION

5.1 Viewing Range of Impossible Figures

We have already demonstrated in Figure 12 that we can use our efficient system to stretch the rendering limit of the impossible staircase, by modeling and rendering the figure at viewpoints far away from the input view, which was quite difficult previously without a tedious modeling step.

When the figure starts to collapse, the degradation is observed to be quite graceful. This is due to the use of thin-plate spline in our constrained optimization. We have the same observation for other examples as well. The RMS error gives a quantitative measure on the distance between the result and the model, or in other words, how bad the failure cases are.

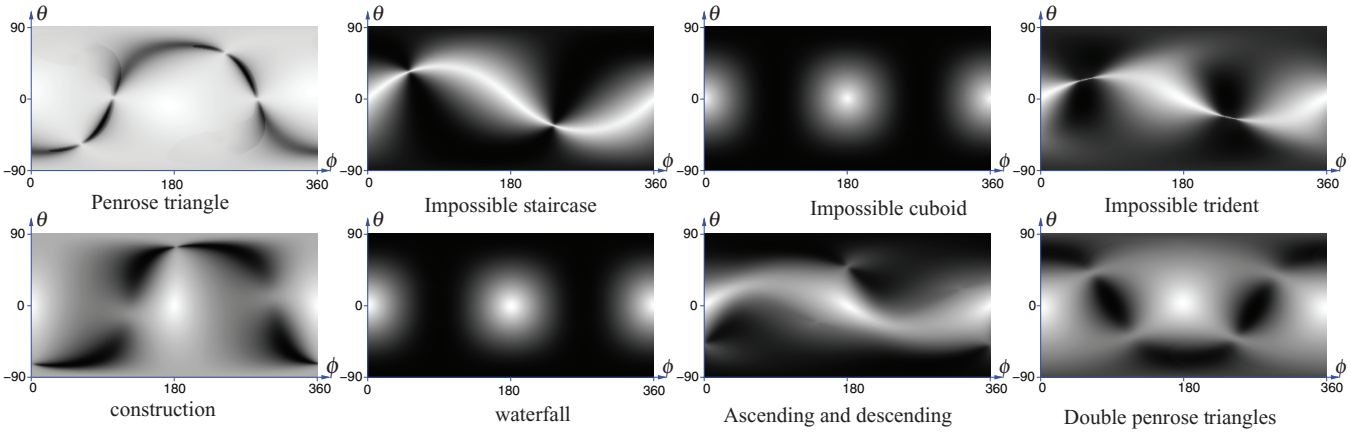


Fig. 15. The RMS error at all possible viewpoints. θ and ϕ denote, respectively, the altitude and azimuth (in degree) of the camera orientation (we assume that the image center is the principal point). $(\theta, \phi) = (0^\circ, 0^\circ)$ is where the input impossible figure is observed. The color at each pixel ranges from black to white (i.e., grey-scale), which is inversely proportional to the RMS error. That is, smaller error is mapped to higher value (i.e., the color is closer to white), and vice versa. Note that the RMS error is normalized to [0, 255] by considering the maximum and minimum errors in each figure, where all RMS errors are computed using Eq. (11) at the respective camera viewpoint.

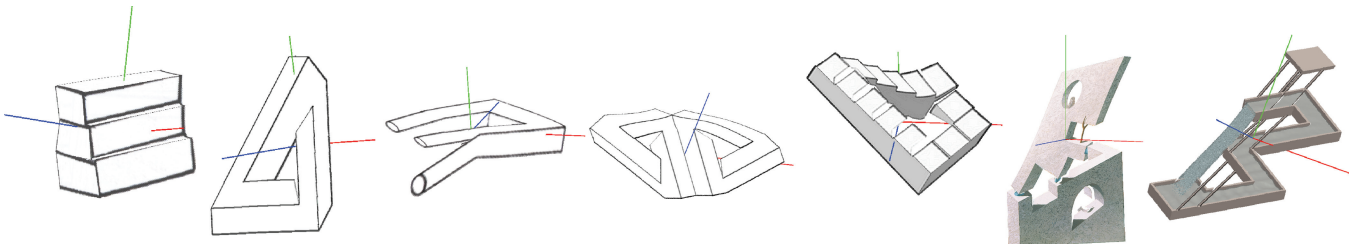


Fig. 16. View-dependent models. While they look distorted when viewed at *other* camera viewpoints, the generated 2D impossible figures consist of straight lines when such models are viewed at the specified viewpoint.

Figure 15 plots the RMS errors (inversely proportional to black/white (i.e., grey-scale)) for all examples shown in this article at all possible viewpoints. We may use these error plots to restrict the viewing positions of our system (e.g., by simple thresholding) where the impossible figure remains feasible without disconnection or severe deformation.

While the extent of feasible camera viewpoints is partially dependent on the choice of method (e.g., Savransky [1999] and Owada and Fujiki [2008]), we believe that the nature and behavior of an impossible figure also play a significant role. This explains why Khoh and Kovesi [1999] produced severe distortion at some of the views in the respective examples. In other words, Figure 15 also reflects the difficulty of each example. Future work should expand the extent of feasible viewpoints and thus the coverage of the bright region of these plots.

Similar to how 3D modeling artists build a scene to render an impossible figure, the optimized view-dependent model can in fact be severely deformed, although at the user-specified novel camera view where the model is optimized, we observe an impossible figure with straight connections. Figure 16 shows the optimized 3D models viewed at other camera viewpoints.

5.2 User Inputs Comparison

As our system only requires user-supplied point and line correspondences on the input figure, the form of user input in our system is more user-friendly than Savransky [1999] and Owada and Fujiki [2008] (see Table I).

Savransky [1999] requires the user to input the so-called *relation*, which is actually the rotation and translation between each pair of 3D parts, therefore, the user needs to have some basic knowledge on affine transformation and the proposed algorithm (e.g., the structure of the *scene graph*) in order to specify the relations. Thus, creating an impossible figure using this system is a more difficult and less intuitive task for general users.

On the other hand, Owada and Fujiki [2008] proposed an improved interface that is more user-friendly than Savransky [1999]: by limiting the target output to line-art, their interface allows the user to experiment with an impossible scene by deleting 3D lines with brush strokes, stitching 3D parts with mouse dragging, and controlling depth and normal orientation of the surfaces with dialogs. This system provides a large degree of freedom on editing. On the downside, the user should understand the notion of constructing an impossible figure before s/he can effectively utilize the provided tools for creating novel impossible figures.

Different from Owada and Fujiki [2008], our system avoids any possibility that the user edits the input 3D geometry in a haphazard manner. Rather, we showed that by only marking up 2D points and lines correspondence, an impossible figure at a novel viewpoint can be readily produced. The user only needs to be concerned about connectivity and parallelism, which are easier to understand than the principles underlying the construction of impossible figures.

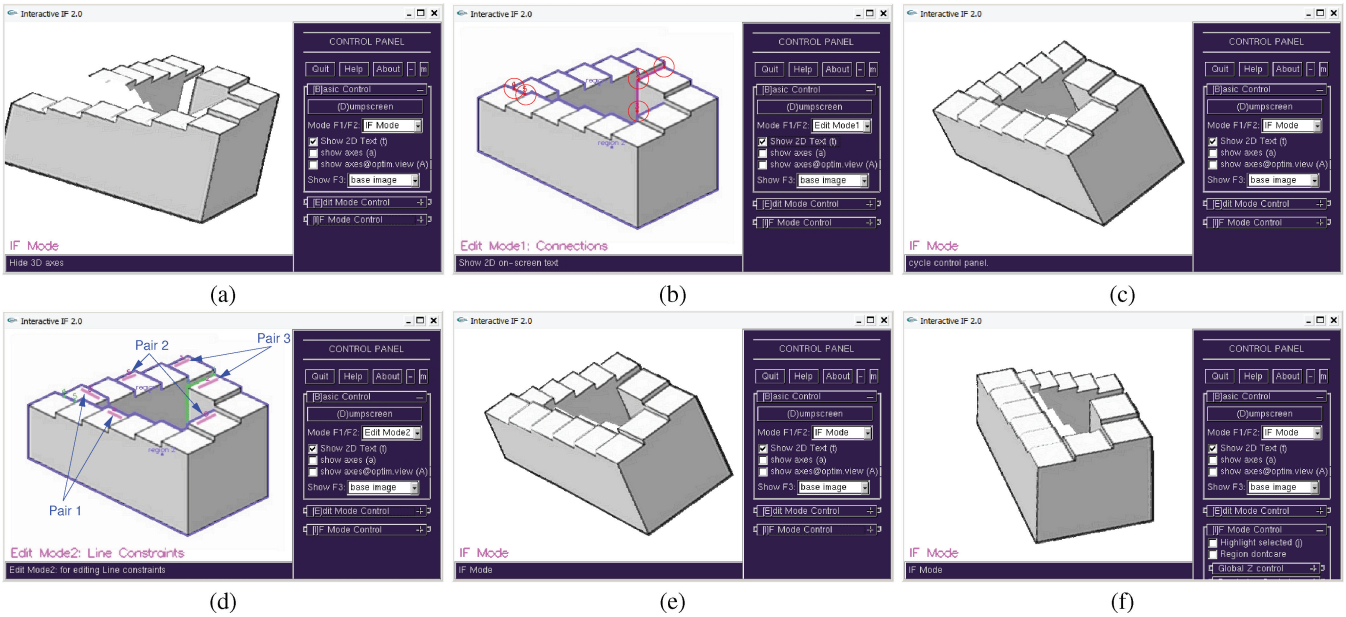


Fig. 17. A rundown for demonstrating typical user interaction: (a) User changes camera viewpoint and observes discontinuity. (b) User specifies points correspondence to connect 3D parts (indicated by the red circles). (c) User observes that parallelism is not preserved. (d) User specifies lines correspondence to enforce parallelism (indicated by the blue arrows). (e)–(f) Novel views are generated.

Table III. Number of User-Supplied Points/Lines Correspondence for the Examples Shown in this Article

Example	# of Point Pairs	# of Line Pairs
Penrose Triangle	6	5
Impossible Staircase	10	1
Impossible Cuboid	9	3
Impossible Trident	15	6
Construction	17	3
Waterfall	45	2
Ascending and Descending	5	3
Double Penrose Triangles	30	6

Here, we show a complete rundown on how a user operates using our user interface.

(1) The user examines the input 3D geometry by changing camera viewpoint and observes that some 3D parts are disconnected in a novel view (Figure 17(a)).

(2) The user resets back to the original viewpoint and specifies points correspondence to connect the 3D part(s) (Figure 17(b)). Note that a single mouse click is sufficient for the system to produce a pair of points, because the projected 3D parts are connected to each other at this viewpoint.

(3) The user changes the viewpoint again and finds that parallelism is not preserved (Figure 17(c)).

(4) The user resets to the original viewpoint and specifies line correspondence to enforce parallelism (Figure 17(d)).

(5) Now, the user can enjoy the novel views generated by changing camera viewpoints (Figure 17(e)–17(f)).

This typical rundown demonstrates that our system provides instant feedback for the user to evaluate the results, and that editing the constraints is an easy task. No knowledge is needed on the notion of how impossible figure is constructed. Table III tabulates the number of points/lines of correspondence that was marked up for the examples shown in this article.

Finally, we believe that our approach is not at odds with the two aforementioned approaches (Savransky [1999] and Owada and Fujiki [2008]). By integrating relevant techniques a more versatile system might be achieved.

6. RESULTS

Existing approaches for constructing 3D models for rendering impossible figures are expensive and provide limited views of the figure. Using our interactive modeling and rendering system, for the first time we are able to produce complex illumination effects on different classes of impossible figures in animation.

6.1 Shading an Impossible Figure

Normals are a core component in shading. However, the surfaces optimized by our system are deformed. If we shade the normals obtained directly from the deformed surface produced after the TPS-optimization, we will perceive an unnatural surface. To tackle this problem, the normals used for shading are sampled from the original 3D model. These normals are rotated by the same amount that is applied to the rigid reference part. In doing so, the shaded surface will still look natural even when the model has been deformed severely. Figure 18 shows a series of renderings depicting a directional (and point) light source being moved from left to right across the Penrose triangle and impossible cuboid.

6.2 BRDF

Using the optimized model, isotropic BRDF data [Matusik et al. 2003] can be arranged on impossible figures. More specifically, we

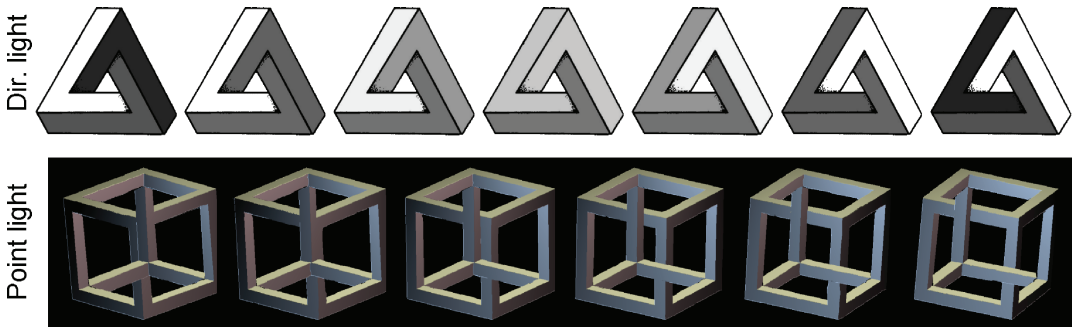


Fig. 18. Top: directional lighting; bottom: point source lighting with variable viewpoint changes.

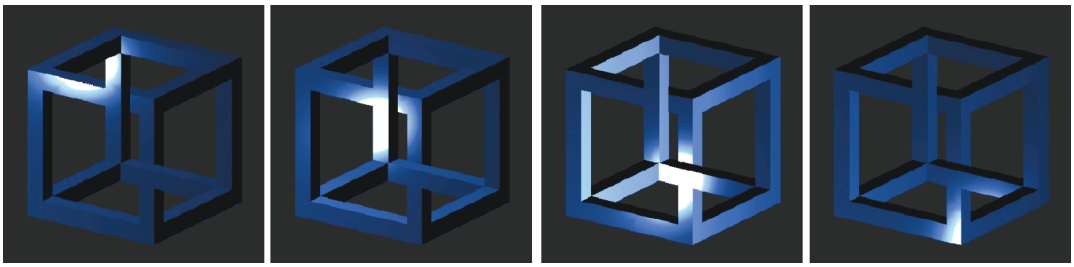


Fig. 19. Dressing the impossible cuboid with an isotropic BRDF.

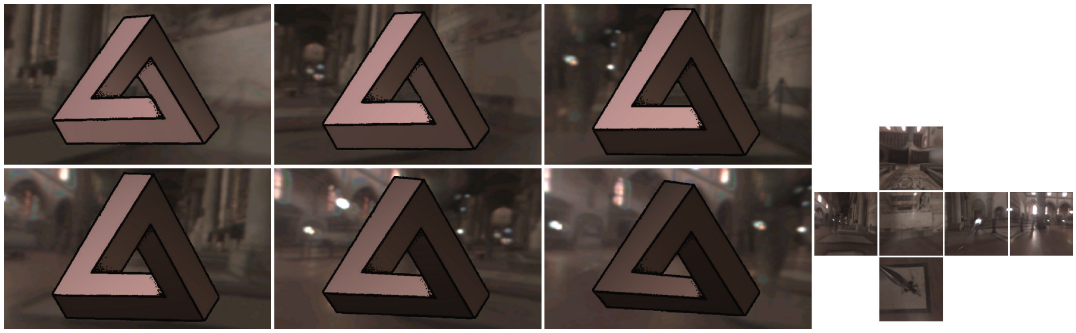


Fig. 20. Viewing the impossible figure under a rotating distant environment.

first compute the light and view vectors for each pixel fragment on the impossible figure, and then compute the angles between: (1) the light vector and the pixel's normal; (2) the view vector and the pixel's normal; and (3) the projected light and view vectors (after projection onto the tangent plane of the pixel). Thus, we can look up the reflectance value in the isotropic BRDF data and shade each pixel accordingly. See Figure 19 for a rendering example with the specular blue phenolic BRDF covering the impossible cuboid. In the image sequence shown, we move a search light around the impossible cuboid.

6.3 Environment Lighting

In addition, we can also put an impossible figure and rerender it under a distant environment. Specifically, we employ the importance sampling approach [Agarwal et al. 2003] by approximating the illumination of a distant environment using a limited number of directional lights. Efficient sampling algorithms have been proposed, and in our implementation, around 200 lights are extracted. Then, for each sample (directional light), we render the impossible figure by local illumination, and produce the final result by summing up the rendering results from multiple passes of such local illumination. Figure 20 presents the composed rendering results of the Penrose triangle under the GRACE environment. Here, we gradually change the viewpoint on the Penrose triangle while simultaneously rotating the distant environment.

6.4 Rendering Artworks of Impossible Figures

Refer to Figures 21–25 of the article. The input figures were shown in Figure 4. High-resolution images are available in the supplemental material in the ACM Digital Library. The live captures shown in the accompanying video depict the modeling and rendering results of the following artworks of impossible figures. The video also demonstrates the ease of specifying the constraints (connection and collinearity constraints are shown as green points while parallel constraints are shown as pink lines in the article and the video).

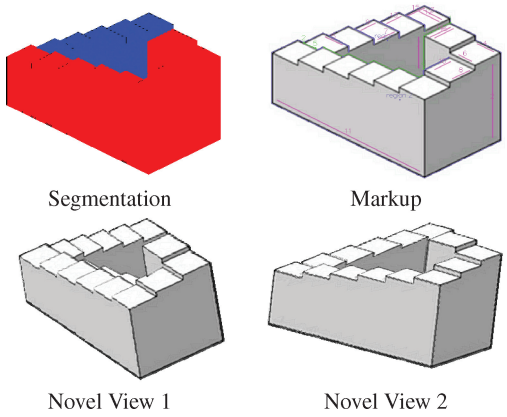


Fig. 21. *Ascending and Descending*.

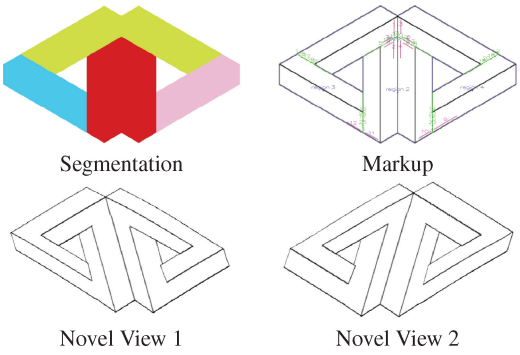


Fig. 22. *Double Penrose Triangles*.

Fig. 23. *Construction*.

6.4.1 *Ascending and Descending*. Like the Penrose triangle, this figure belongs to the class of *depth contradiction*. Refer to Figure 21 which shows the parts segmentation, constraint markups, and novel views. The red part is chosen as the reference, which undergoes rigid transform under normal camera projection (which can be perspective or orthographic). The connection constraints maintain the connectivity of the two parts at the novel view. The parallel constraints reduce the distortion effect on the blue part, which undergoes structure deformation for connecting to the transformed red part.

6.4.2 *Double Penrose Triangles*. Using our system we can compose and render new impossible figures by blending existing impossible figure parts, where the view-dependent geometry will be optimized in exactly the same fashion, that is, by connecting the input parts in 3D to produce an impossible figure at the chosen view. In Figure 22, we juxtaposed two Penrose triangles. In the optimization process, the red part is the reference part. Constraints are specified to bind the two triangles together while all straight and parallel connections are maintained among the pertinent parts. This example is difficult because of the severe structural conflict within and between the two Penrose triangles.

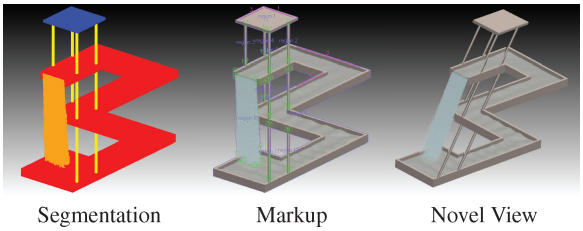


Fig. 24. *Waterfall*.

6.4.3 *Construction*. Similar to the Penrose triangle, *Construction* was constructed using two locally possible parts, and each part was modeled as a height-field. Figure 23 shows the constraint markup, parts segmentation, and novel views. The lower part (the red part) is chosen as the reference part which undergoes rigid body transformation. Nonrigid transformation is applied to the upper part in the constrained-TPS optimization. The collinearity constraints protect the overall shape from severe distortion, while the parallel constraints enforce the left and right sides of both parts to be straight and parallel in the rendered novel views. Note that the figure only starts to collapse at novel views far from the input view. At these viewpoints, the rate of collapse is accelerated by the use of height-field, which is not a full 3D representation.

6.4.4 *Waterfall*. Similar to *Ascending and Descending*, this figure also belongs to the class of depth contradiction. This is one of the most difficult examples where the two "Penrose-triangle"-structures are shackled together with multiple pillars. Figure 24 shows the parts segmentation, constraints, and novel views. The red part is chosen as the reference. The connection constraints are used to enforce the necessary connectivity to hold the figure as one fully connected component at the novel view. The parallel constraints are used to preserve the overall shape of the top roof. Note that while the novel views still preserve all straight connections, the waterfall model looks skewed at the novel view due to the severe structural inconsistency inherent in this impossible figure.

6.4.5 *Belvedere*. This impossible figure belongs to the class of *depth interposition*. The height-fields of the possible parts, namely, the upper level and the lower level, are available. Connection constraints are used to enforce the pillars to be connected to both the upper and lower levels. Similar to *Construction*, parallel and collinearity constraints are specified to maintain the overall shape of the entire architecture. Figure 25 shows some novel views generated. Notice that we inpainted the background layer to further enhance the rendering effect of this masterpiece by M. C. Escher.

6.4.6 *Possible and Impossible*. Finally, we blend our impossible object (that is, the optimized view-dependent model) into a geometrically possible 3D scene to create special effects. Figure 26 (top and middle row) shows several snapshots of the animation sequence of such scenes. The novel viewpoints are specified using a rigid camera, where standard perspective transformation is applied to the possible objects and also to the rigid part of the view-dependent model. Constrained-TPS optimization is applied to the nonrigid parts of the model as described in the article. Using

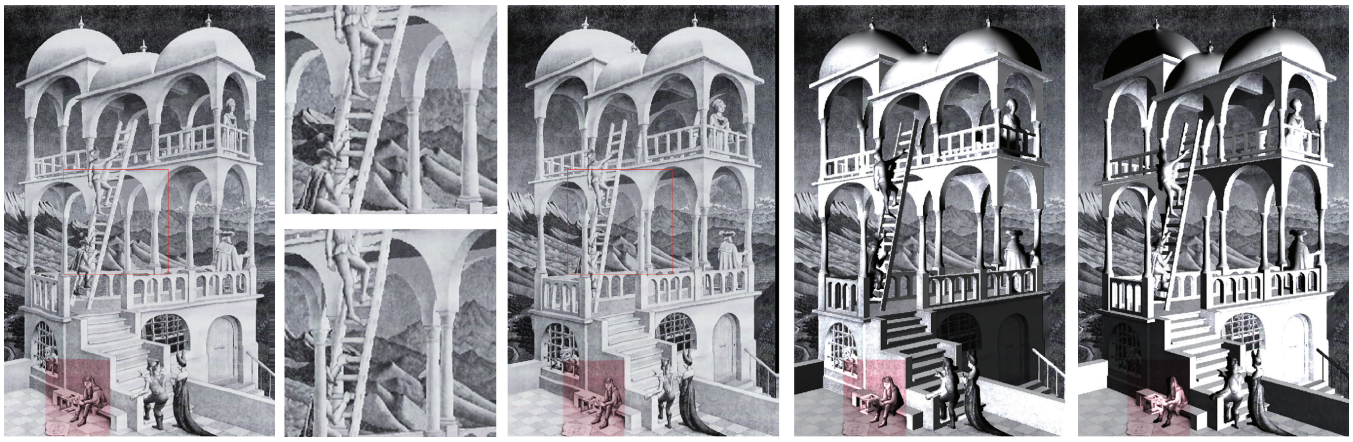


Fig. 25. Novel views of *Belvedere*. Using the optimized view-dependent model, we can relight *Belvedere* under different lighting configurations.

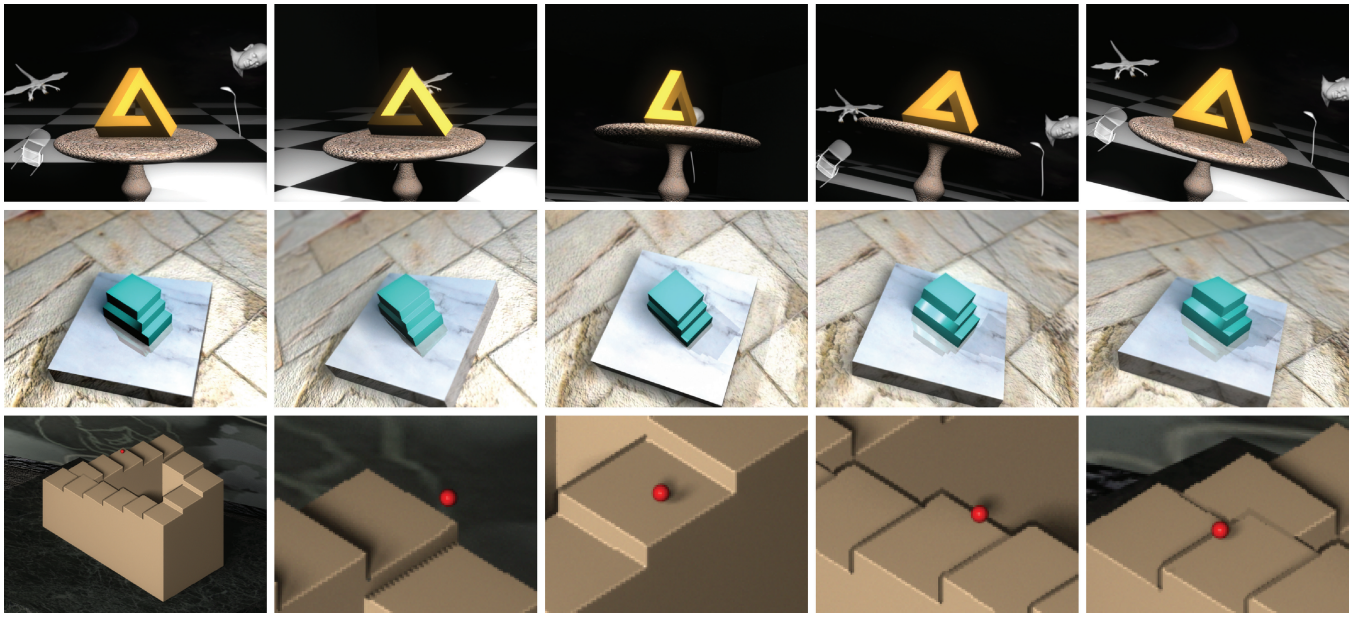


Fig. 26. Modeling and animating 3D scenes with possible and impossible objects. See the submission video for the complete animation.

previous 3D approaches it is difficult to produce these animations, because constructing a per-frame 3D model was done by hand or else required expensive computation.

Note in particular the bottom row of Figure 26 where we show the zoom-in views of a possible object (a ball) bouncing on an impossible object (*Ascending and Descending*). This involves collision detection and response handling. The problem is nontrivial when impossible objects are involved: a possible and an impossible object cannot interact directly in the same 3D space because the latter is highly deformed. While this is a future work to pursue, here we adopt a simple approach to produce this visual effect, by rendering each frame in two layers: the bouncing ball and the remainder of the scene.

7. CONCLUSION

Impossible figures have long been used in applications such as computer games, nonphotorealistic rendering, and image synthesis. This article investigates a practical approach for modeling and rendering impossible figures. Our approach is motivated by how a 3D modeling artist builds view-dependent models for rendering impossible figures. Modeling and rendering of impossible figures are coupled. This led to our *view-dependent modeling* approach which connects possible 3D parts for rendering novel views of an impossible figure. Our mathematical formulation shows that a linear least-squares solution exists for view-dependent modeling, thus allowing us to implement an efficient system to model and render novel views of impossible figures at interactive speed. This formulation also provides a numerical mean for pruning away invalid viewpoints where the impossible figure ceases to exist. Once optimized, the 3D model can be used to create compelling visual effects previously restricted to possible 3D graphics models.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their useful comments and suggestions. The *Construction* painting used in this

article, "Fantoomachtig bouwsel in Grote Woestijn" (1997) was created by Jos de May.

REFERENCES

AGARWAL, S., RAMAMOORTHI, R., BELONGIE, S., AND JENSEN, H. W. 2003. Structured importance sampling of environment maps. *ACM Trans. Graph. 22*, 3, 605–612.

ALEXEEV, V. 2008. Impossible world. http://im-possible.info/english/.

BOOKSTEIN, F. 1989. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Trans. Patt. Analy. Mach. Intell. 11*, 6, 567–585.

ELBER, G. 2002. Escher for real. http://www.cs.technion.ac.il/∼gershon/EscherForReal.

ERNST, B. 1987. *Adventures with Impossible Figures*. Tarquin, Stradbroke, England.

HEYDEN, A. 1996. On the consistency of line-drawings, obtained by projections of piecewise planar objects. *J. Math. Imaging Vis. 6*, 4, 393–412.

HUFFMAN, D. A. 1971. Impossible objects as nonsense sentences. *Mach. Intell. 6*, 295–323.

JU, T., ZHOU, Q.-Y., AND HU, S.-M. 2007. Editing the topology of 3d models by sketching. *ACM Trans. Graph. 26*, 3, 42.

KHOH, C. W. AND KOVESI, P. 1999. Animating impossible objects. www.csse.uwa.edu.au/∼pk/Impossible/impossible.html.

LIPSON, A. 2002. Andrew lipson's LEGO page. http://www.andrewlipson.com/lego.htm.

MATUSIK, W., PFISTER, H., BRAND, M., AND MCMILLAN, L. 2003. A data-driven reflectance model. *ACM Trans. Graph. 22*, 3, 759–769.

M.C. ESCHER FOUNDATION. The official M.C.Escher website. http://www.mcescher.com.

NEALEN, A., IGARASHI, T., SORKINE, O., AND ALEXA, M. 2007. Fibermesh: Designing freeform surfaces with 3D curves. *ACM Trans. Graph. 26*, 3, 41.

OWADA, S. AND FUJIKI, J. 2008. Dynafusion: A modeling system for interactive impossible objects. In *Proceedings of the Conference on Non-Photorealistic Animation and Rendering (NPAR)*. 65–68.

PENROSE, L. S. AND PENROSE, R. 1958. Impossible objects: A special type of illusion. *Brit. J. Psychol. 49*, 31–33.

RADEMACHER, P. 1999. View-dependent geometry. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'99)*. ACM Press/Addison-Wesley, New York, 439–446.

SAVRANSKY, G., DIMERMAN, D., AND GOTSMAN, C. 1999. Modeling and rendering Escher-like impossible scenes. *Comput. Graph. Forum 18*, 2, 173–179.

SCHATTSCHNEIDER, D. AND EMMER, M., Eds. 2003. *M.C.Escher's Legacy: A Centennial Celebration*. Springer.

SIMANEK, D. E. 1996. The principles of artistic illusions—Adding depth to illusions. http://www.lhup.edu/∼dsimanek/3d/illus2.htm.

SUGIHARA, K. 1982. Classification of impossible objects. *Percept. 11*, 65–74.

SUGIHARA, K. 1986. *Machine Interpretation of Line Drawings*. The MIT Press.

SUGIHARA, K. 1997. Three-Dimensional realization of anomalous pictures–An application of picture interpretation theory to toy design. *Patt. Recogn. 30*, 7, 1061 – 1067.

SUN, J., JIA, J., TANG, C.-K., AND SHUM, H.-Y. 2004. Poisson matting. *ACM Trans. Graph. 23*, 3, 315–321.

TSURUNO, S. 1997. The animation of M.C. Escher's "Belvedere." In *Proceedings of the ACM SIGGRAPH Visual Conference*. 237.

URIBE, D. 2001. A set of impossible tiles. http://im-possible.info/english/articles/tiles/tiles.html.

WAHBA, G. 1990. *Spline Models for Observational Data*. SIAM.

WU, T.-P., TANG, C.-K., BROWN, M. S., AND SHUM, H.-Y. 2007. Shapepalettes: Interactive normal transfer via sketching. *ACM Trans. Graph. 26*, 3, 44.

YU, J. AND MCMILLAN, L. 2004. A framework for multiperspective rendering. In *Proceedings of the 15th Eurographics Workshop on Rendering Techniques (EGSR)*. 61–68.

Received October 2009; accepted December 2009