Computers & Graphics 51 (2015) 146–156

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

SMI 2015

3D shape and texture morphing using 2D projection and reconstruction

Michael Ludwig[a], Seth Berrier[b], Michael Tetzlaff[a], Gary Meyer[a]

[a] University of Minnesota, Department of Computer Science and Engineering, United States
[b] University of Wisconsin Stout, Department of Math, Stats and Computer Science, United States

ARTICLE INFO

Article history:
Received 12 March 2015
Received in revised form
11 May 2015
Accepted 15 May 2015
Available online 29 May 2015

Keywords:
3D morphing
2D image morphing
Surface light fields
Lumigraphs

ABSTRACT

We present an algorithm for morphing shape and view-dependent texture that utilizes an unstructured lumigraph representation to (1) gracefully handle topological changes, (2) create plausible appearance transformations, and (3) provide user control independent from the underlying mesh structure. It works by reducing the otherwise complex problem of a 3D shape and material morph into multiple simpler 2D morphs. The shape morph operates on 2D depth maps and then reconstructs a conventional mesh, making it insensitive to the input geometry quality. Our morphing algorithm robustly handles topology changes and outputs complete meshes and lumigraphs for later use. We show how 2D image morphs can be computed from 3D correspondences, while eliminating ambiguities that might result in the projected 2D morphs.

1. Introduction

Designers and artists have always worked from examples, imitating and expanding what has come before just enough to advance to the next level. Photogrammetry and unstructured lumigraph rendering techniques can now be used to preserve these existing works and samples while removing their physical constraints. As photography becomes more and more ubiquitous, tools that operate on easily captured representations from imagery will help cement their use by consumers, designers, and non-technical users. In this paper we present a novel technique for morphing texture and geometry that can be used within the fields of design, special effects, and cultural heritage. The endpoints of the morph represent existing examples with desirable qualities. Our technique allows morphing of the shape of the targets while also appropriately blending color properties. These two targets could be samples of existing materials or artifacts that form the inspiration of a new product in the design discipline. They might be scans of actors faces where the desired effect is to blend between expressions to create a novel performance or to combine the qualities of entirely different characters [1,2]. It is a tool for use both in early conceptual design and in final stages of effect rendering. It provides a framework for bringing the substantial success of image morphing [3–5] to 3D shape and texture morphing. It does this by reducing the complex nature of unstructured lumigraph data to a set of robust and well aligned 2D image morphs.

The 2D images utilized in our morphing algorithm represent a surface light field, specifically, an unstructured lumigraph. A light field is a simplification of the 5D plenoptic function that encodes the transport of light in a 3D space as a field [6,7], reducing it to a 4D function. Modern representations of light fields combine approximations of the object's surface with the fact that the space around the object is un-occluded to reduce further to a 3D surface approximation and a set of 2D samples of the object [8–10]. In other words, the surface light field is represented with a set of color images of the space from various views. This is exactly the same data acquired using commercial photogrammetric applications like Agisoft PhotoScan and 123D Catch: many photographs of an un-occluded object intelligently analyzed to reconstruct the object's surface. Unstructured lumigraph rendering and photogrammetry can be considered as different sides of the same coin. Unstructured lumigraph rendering focuses on reproducing the appearance of an object from photographs; any geometry used is often a crude proxy. Photogrammetry emphasizes accurately reconstructing the geometry of the object, but reduces the appearance to simple diffuse textures. In the majority of the examples present in this work, we have used Agisoft PhotoScan to create a high-fidelity geometry for morph targets from real world objects and then render them using a modified unstructured lumigraph algorithm [9].

We believe that our approach offers several advantages over prior 3D morphing algorithms. First, our approach computes a 3D geometric morph without the need for re-meshing or merging of the targets. This eases topological constraints in previous mesh-based morphing techniques. Second, we do not lose parametric information on the surface (i.e. appearance attributes) as is typical of volumetric and distance field approaches. Our method can gracefully handle topology changes while simultaneously morphing surface color and

http://dx.doi.org/10.1016/j.cag.2015.05.005
0097-8493/© 2015 Elsevier Ltd. All rights reserved.

texture. Third, our approach can be considered an unstructured lumigraph morphing algorithm producing complete intermediate lumigraphs; prior techniques worked in the absence of geometry or in an implicit space making it difficult to extract the morphed geometry. Finally, our approach demonstrates a masking operation that simply and effectively improves the 2D image morph results while allowing the morph to be specified in 3D. This allows our algorithm to be modular and trivially take advantage of further advances in the field of 2D image morphing.

We achieve this by representing surface properties as an unstructured lumigraph, and morphing views of the lumigraph in 2D space, then projecting the morphed views onto morphed geometry. The geometry morph is similarly handled by morphing 2D depth maps of the geometry, then reassembling these depth maps into a mesh by creating a point cloud and applying Poisson surface reconstruction. We achieve compelling results using both synthetic data sets to emphasize certain topological changes, and real-world data sets captured from photogrammetry. We demonstrate our approach's capacity to control the morph based on texture features not present within the geometry.

In the following section we present an overview of our technique and outline the algorithm. We then discuss related work in Section 3. A detailed description of our algorithm follows in Section 4. Finally, we present several examples and a discussion of the overall results and possible future work in Section 5, before a brief conclusion in Section 6.

2. Overview

Instead of directly morphing the 3D field of light as in previous techniques [11,12], we engage the 2D samples of the unstructured lumigraph directly and construct many 2D morphs. These morphs are executed in parallel and use a common set of correspondences to ensure that the motion of each morph is consistent between 2D samples. We do this to the original color images of the unstructured lumigraph to achieve a morph of the appearance. Simultaneously, we compute a similar 2D morph using a ‘depth image’ representing the distance to the surface at each pixel for each view of the object. This achieves a morph of the surface geometry between the targets.

At any point in the linear progression of the morph a new surface can be extracted from the warped depth images. A set of 2D appearance samples can be extracted from the warped color images. Together, these can be used to render a new unstructured lumigraph that possesses qualities of both targets and is a full 3D representation with appropriate appearance properties.

2.1. Our algorithm

We now introduce our algorithm using the barrel to traffic cone example shown in Fig. 2. A more detailed description is presented in Section 4:

Pre-processing of the unstructured lumigraphs: Prior to any processing we must generate a depth image for each view of the lumigraph (if not already present). This is a straightforward process using the proxy geometry present in an unstructured lumigraph. For the example shown in Fig. 2 the color images were rendered from mesh data. Depth images were simply extracted from the z-buffer used in this rendering.

Creation of a sparse 3D correspondence: Pairs of points are placed on the two targets to identify locations that should be in correspondence. We manually place these correspondences directly on the surface of each lumigraph (the green dots in Fig. 2a). These correspondences exist in 3D on the proxy geometry of the lumigraph.

The shapes of the barrel and traffic cone are similar to primitive shapes (a cylinder and cone). The rotational symmetry of these shapes was used to place the correspondences algorithmically and establish a dense sampling across both surfaces. Many correspondences on the lid of the barrel map to a very small area at the top of the traffic cone. The stretching of the cone to fill the larger area of the lid causes artifacts in some views of the object. However, when rendered as a lumigraph, these artifacts disappear with careful filtering.

Creation of dense 2D correspondences: For each view of the unstructured lumigraphs we wish to establish a dense correspondence across both the color image and depth image. We project the 3D correspondences into each view (Fig. 2b) and use them to construct a triangle mesh (Fig. 2c). This mesh represents a bilinear parameterization across the image.

Not all correspondences will be visible on both targets. Some are occluded by the object's surface. These are identified by comparing their projected distance to the values of the distance image and removing those that do not match. There will also be some correspondences that are visible in only one of the two targets. These still prove important in the overall morph and are used in the mesh triangulation despite not being visible on the other target. Some of these features can be seen in Fig. 2b near the base of the barrel and the edges of the traffic cone.

Creation and execution of the 2D morphs: To create the final 2D morph, a mesh is created for each target (as described in the previous step). The images (both color and depth images) are textured onto these 2D meshes. The positions of the mesh vertices are then linearly interpolated between the various targets while the values of the images are blended to generate a simple 2D morph. There are many of these 2D morphs, one for each view in the original unstructured lumigraph representation.

In Fig. 2c we show the color images from each target and the warping mesh calculated for the barrel correspondences. The in-between positions are simple linear interpolations of the vertex positions of this mesh.

Reconstruction of in-between stages: The individual depth images are processed to produce a point-cloud representation of the surface. The point-cloud is then converted into a triangle mesh using Poisson surface reconstruction [13]. This surface is combined with the warped and blended color images to form the basic components of an unstructured lumigraph (Fig. 2d). As in a typical unstructured lumigraph approach the warped and blended color

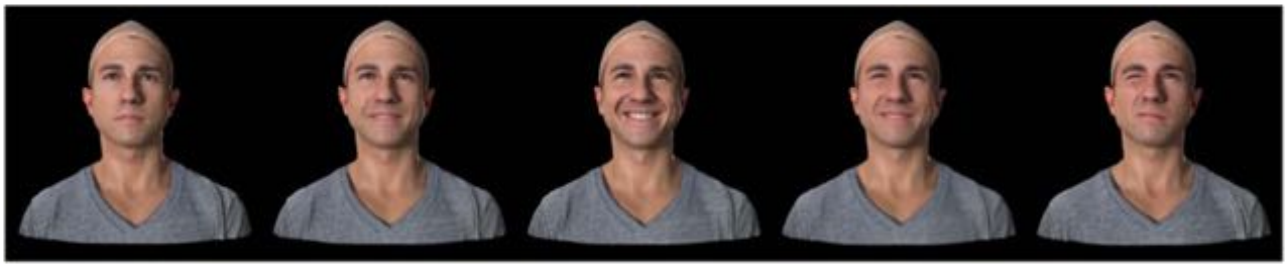

Fig. 1. A full 3D shape and texture morph through three different poses of a facial animation.

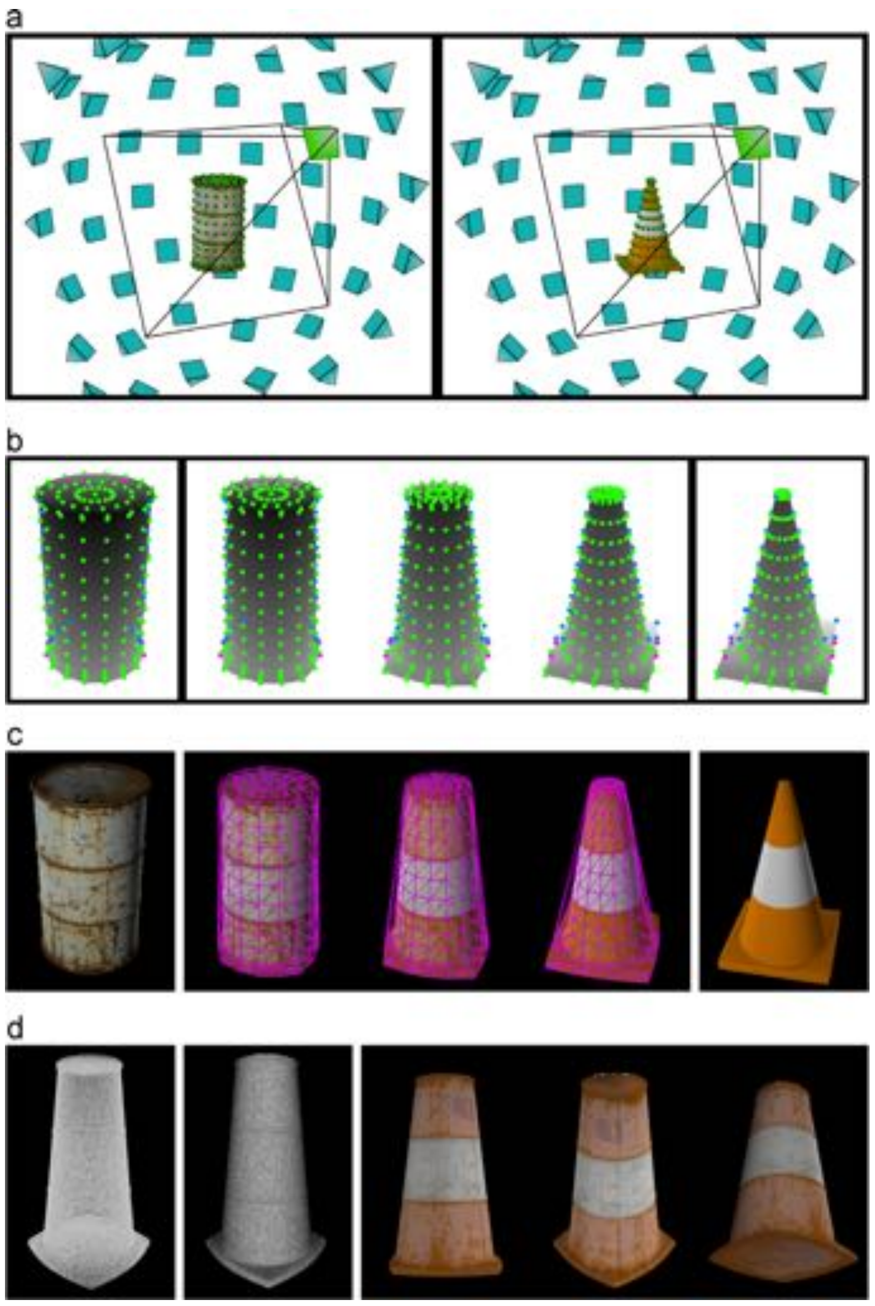

Fig. 2. An example morph used to demonstrate the algorithm in Section 2.1. The top images show the setup of the two targets with manual 3D correspondences. Other images show the intermediates as the left and right most images and the warped and blended results. (a) An example morph between synthetic unstructured lumigraphs of a barrel and a traffic cone. The green dots are correspondences manually placed on their surfaces. The pyramids around the objects are some of the many views present in the lumigraph data. (b) The correspondences from Fig. 2a projected onto the 2D depth images. Correspondence visibility is determined by comparing the projected depth with the value in the depth image (green = visible on both targets, magenta = barrel only, cyan = cone only). (c) Visible correspondences from Fig. 2b are triangulated in image space to form a mesh. Color and depth images are simultaneously warped by texturing them onto the meshes. By interpolating mesh vertex positions the color and depth images are warped and blended to become the input of a new unstructured lumigraph. (d) A point cloud is computed from the warped depth images (leftmost). This structure is converted to a mesh via Poisson surface reconstruction (second from left). The mesh is used as the proxy geometry while the warped color images are weighted and reprojected as an unstructured lumigraph. The three images to the right are renderings of this 50/50 interpolated unstructured lumigraph from several different views. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

images are intelligently combined and re-projected onto the proxy geometry to allow for novel views of the object from any angle.

3. Related work

3D metamorphosis techniques (as surveyed by [14]) fall into two broad categories: volumetric techniques and boundary/polygonal techniques. In volume based approaches, the targets are defined implicitly by a level-set function [15]. Recent work in volumetric morphing has produced very high quality geometric results as demonstrated by [16]. In general these techniques avoid all of the problems associated with topological constraints on the morph targets. However, properties associated with the object surface (such as appearance properties) are lost in the level-set representation. Our approach builds on the successes of these techniques by also working in a sampled space (image space) where the surface is implicit. In addition, we support morphing of texture and offer more control over the movements in the morph through the placement of features.

Polygonal surface approaches to 3D morphing are much more varied and primarily struggle to bring the morph targets into correspondence (as noted by [14]). This usually requires mapping both objects to common geometry such as a cylinder or sphere. Refs. [17,18] map the targets to common geometry and merge them into a single mesh while [19,20] use remeshing techniques to achieve their correspondence. The work of [21] establishes a control mesh over the surface of the morph targets and uses the correspondence of this mesh rather than the entire targets to compute the morph. This approach can allow for more complex topologies and even changes in genus. In general, no single technique in this area solves the correspondence problem for all possible topologies of objects and most require very complex algorithms from computational geometry to achieve good results. Our technique side-steps this problem by reducing the inherently 3D problem into a set of 2D problems where morphing has been far more successful. We have no topological constraints and support changes in genus. In fact, the use of 2D morphing techniques to achieve a robust 3D morph is a fundamentally different approach from any prior 3D shape morphing technique.

Additionally, with implicit morphs, texture and appearance transfer has been a problem rarely addressed. Dinh [22] provides a means to transfer texture across topologies using an implicit function. Unfortunately this function provides limited control over the movement of specific features. Our method provides explicit control over both the texture and shape, while still supporting topological changes.

While image based rendering is a large and varied research area, there are three main representations of surface light fields that have proven effectiveness: the *lumisphere* representation developed by [8], the *unstructured lumigraph* representation developed by [9] and the *light field mapping* technique developed by [10]. We have chosen to use the *unstructured lumigraph* representation for our work due to its ease of implementation on modern hardware and its direct use of the imagery that samples the object. This makes it particularly convenient for rendering photogrammetric data. It has also seen several enhancements in recent years [23,24] improving the efficiency and quality of the representation.

Surface light field morphing has already been demonstrated by [25] based on the *light field mapping* technique of [10]. Their technique is restricted because they rely on a geometric morph for their surface proxy, thus suffering from the same limitations of mesh morphing described above. Other approaches such as [11,12] eschew any detailed surface representation and use the dual planar or cylindrical light field representations. Although they avoid topological issues from geometry, the lack of geometry hinders their application to design and synthesis and precludes them from being considered a full-fledged shape morph.

Our approach is most closely related to that of Wang et al. [12], which reduces the light field morph to a set of 2D image morphs. While Wang et al. warp individual views as we do, there are significant differences between their approach and ours. They establish correspondences between morph targets in 2D instead of 3D. This means that the effort required to specify correspondences scales with the number of views used, whereas our method can use the same 3D correspondences across many views, substantially reducing this effort. To alleviate this issue, Wang et al. do not compute full intermediate lumigraphs as we do but only a series of images along a camera path. This path is utilized to reduce the number of 2D views considered, thus reducing the required number of correspondences to be specified. However, having a full lumigraph at each stage of the morph is desirable because it allows for real-time interaction with the intermediate lumigraphs. It is not clear that their algorithm would scale to warping all views simultaneously for complete intermediate lumigraphs. Even if it did, it would still require features to be placed in all views and significantly increase user interaction. Finally, Wang et al. require users to manually resolve feature occlusion where we do not. Aside from correspondence placement, our approach requires no additional user interaction.

4. Implementation

Within this section we expand upon the algorithm first presented in Section 2 to describe critical details of the approach and its limitations. The discussion is arranged similar to the overview. First, morph preparation is explained, including discussion of how real-world objects are prepared and the details of the user interface for morph specification. Second, the 2D morph process is described, as well as our modifications that allow the many 2D morphs to be recombined. Third, the details of 3D shape and texture reconstruction are presented. Lastly, we discuss limitations of this approach and how they may be mitigated.

4.1. Morph preparation

Before any morph can be computed, two target unstructured lumigraphs must be built. A lumigraph, $\mathcal{L}$, can be constructed in a number of ways, so long as it has a geometry proxy, G, a set of intrinsic and extrinsic camera properties, $\{K^1, K^2, \dots\}$, and a set of color images, $\{C^1, C^2, \dots\}$. A set of depth images $\{D^1, D^2, \dots\}$ (needed later in our algorithm) can be automatically generated from G and the corresponding K^i, or provided explicitly from a depth camera when available. The sets of camera poses, color images, and depth images are of the same size, equal to the cardinality of the lumigraph, $|\mathcal{L}|$. For convenience we assume that these are consistently ordered so that the index $i \in \{1, \dots, |\mathcal{L}|\}$ refers to the same view sample in all three sets.

When a target is a synthetic object described as a textured triangle mesh, the lumigraph color images are produced by rendering the target from a set of known camera poses. Poses should be chosen to adequately sample the range of incoming viewing angles of the object. There are a number of well studied solutions to evenly sampling a sphere that can be leveraged [26]. Thus, G, each C^i, and K^i are well-defined. When a real world object is desired, the lumigraph data must be constructed from photographs of the object. In Section 5, the morph targets were digitized from a set of photographs using the photogrammetry tool, Agisoft PhotoScan. $\{C^1, C^2, \dots\}$ is simply the photographs taken and G is the high resolution triangle mesh PhotoScan produces[1]. Alternate methods of lumigraph capture, such as a Kinect or other time-of-flight camera combined with the KinectFusion algorithm, can record the lumigraph imagery and build a geometry proxy quickly. Both photogrammetry and the Kinect compute the necessary camera pose, K^i for each view. With real-world objects, sometimes the views are predetermined by using cameras in fixed positions arranged on a gantry system. When the camera is handheld (as is the case with our examples in Section 5), we take photos spaced approximately 10° apart around a circle at up to 5 levels (one at eye level, two below and two above). The top and bottom level images do not need to be as frequent and we typically switch to 20° angular steps around these circles. This process yields 90–120 views for each object depending on how precise the spacing is and how many levels are used.

[1] As PhotoScan is proprietary, the details of its approach are unknown but it clearly utilizes computer vision algorithms typical of photogrammetry.

As will be made in clear in Section 4.2, the morph algorithm requires the views of the two target lumigraphs to be aligned. Two lumigraphs, $\mathcal{L}_1$ and $\mathcal{L}_2$, are aligned if $|\mathcal{L}_1| = |\mathcal{L}_2|$ and $K_1^i = K_2^i$ for each $i \in \{1, \ldots, |\mathcal{L}_1|\}$. This property can be easily satisfied by construction when a morph target is synthetic or photographed using a gantry system. If the views are not aligned, the second lumigraph must be re-sampled to produce views that are aligned. Unlike re-meshing (which is frequently required in mesh-based morphing algorithms) this re-sampling is a simple, forward rendering problem achieved by aligning the origin and basis of the two coordinate systems and rendering the lumigraph using the camera poses from the other target. Similarly, if each lumigraph target has a different view set cardinality, any missing views can be rendered. This re-sampling introduces only minor artifacts that are indiscernable once the views are blended and rendered as an unstructured lumigraph. The real-world morphing results shown in Section 5 have all been aligned by this re-sampling process.

The lumigraph re-sampling process can also be used to inflate the cardinality of the lumigraph so that there are more 2D morphs computed. This could be useful if the structure of the objects being morphed required better sampling than what was used during the lumigraph construction. We have not found this necessary in practice. Indeed the number of photographs necessary to adequately perform photogrammetry from all directions (90–120) is sufficiently large that no issues presented themselves when computing the 2D morphs in Section 5.

Once the two target lumigraphs are captured and aligned, the user must provide a morph specification. The morph specification is a sparse set of correspondences between the coordinate systems of each target. Our approach represents a correspondence as a pair of positions in 3D space, one for each target. It is not necessary that these points lie on the surface of each target but that is a restriction imposed by our user interface and is a natural approach for a surface light field representation. In our user interface, the user clicks on the surface of the proxy geometry for one target lumigraph and a new 3D feature will appear at that position on the surface. Simultaneously, a corresponding feature point will be created on the other target lumigraph (see Fig. 3a). The position of this feature is estimated by projecting the feature from the first target onto a bounding sphere centered at the centroid of the first proxy geometry, and then projecting from the same position on a bounding sphere centered at the centroid of the second proxy geometry back onto the other lumigraph's proxy geometry. The user will usually have to tweak the position of this automatically generated feature to place it in the desired location. The user can also specify 3D cubic interpolating curves by connecting four 3D feature points. While our implementation only supports points and curves, in theory, the software could be extended to support any 3D geometric shape which can be projected into 2D and used as input to the 2D morph algorithm. For instance, lines and polygons could be specified in 3D and then projected into 2D to mimic the interface used by Wang et al.

Specifying the correspondence points for a morph is a user intense process that involves iteratively adding and refining the correspondence positions in each target space. Often a significant number of correspondences need to be specified (in the hundreds) to preserve the edge quality present in the textures of each morph target. This is a shortcoming of the simple linear 2D morph algorithm described in the next section. However, due to the modular nature of our approach the 2D morph algorithm can be upgraded independently, and the successes in matching patterns and edges across images can be leveraged to reduce the user effort. Even though a large quantity of correspondences must be provided, by specifying them in 3D space the morph specification is independent of the lumigraph size of each target. Specifying 2D correspondences in each view would entail significantly more work for the size of the photogrammetrically acquired lumigraphs in our results (90+ views).

4.2. 2D morphing

To construct a morph between the target lumigraphs $\mathcal{L}_1$ and $\mathcal{L}_2$ we construct a set of 2D image morphs from the sets of color images and depth images within the two lumigraphs. Each 2D morph occurs between a pair of color images, (C_1^i, C_2^i), or depth images, (D_1^i, D_2^i), where $i \in \{1, \ldots, |\mathcal{L}_1|\}$. For this reason, the two lumigraphs must be aligned as described in Section 4.1. Thus, $K_1^i = K_2^i$ and will be referred to as K^i. There will be as many 2D morphs as there are views in the lumigraphs. All of these 2D morphs are constructed and executed simultaneously and independent of one another unaffected by order (the process is both associative and commutative). Each 2D morph M^i is constructed once for each view in K^i and then applied to both (C_1^i, C_2^i) and (D_1^i, D_2^i) in the same manner.

When constructing the 2D morph M^i, the 3D feature pairs from the morph specification are projected into the view using K^i to produce projected 2D feature pairs. The depth of the projected point (z-value) is compared to the corresponding pixel value in D_1^i or D_2^i and points that do not match are assumed to be occluded. If the feature is occluded in only one target, it is retained but flagged as 'partially occluded'. If it is occluded in both targets, it is discarded from the 2D morph entirely. The 2D feature pairs are triangulated twice into triangle meshes T_1^i and T_2^i, where T_1^i is formed by the projected points of the first target, and T_2^i is similarly formed from the feature locations in the second target. Partially occluded features are only included in the target for which they were visible. We use simple Delaunay triangulation in our implementation. A border of edges that frame the entire image may optionally be added to T_1^i and T_2^i to help preserve the silhouette of the object. We found in our examples this border was not needed.

The pair of triangulations generated for the two targets represents a dense correspondence when treated as a linear mapping across the images. To execute a 2D morph M^i, we texture either C_1^i or D_1^i onto T_1^i (treated as a polygonal mesh in the xy-plane) and texture the matching C_2^i or D_2^i onto T_2^i. The morph is executed at a particular interpolation value, α between 0 and 1, which produces a color image C_α^i and a depth image D_α^i. To determine these images, the vertices of the meshes formed by T_1^i and T_2^i are linearly interpolated between the paired feature position in the other target (even if it had been partially occluded). Simultaneously to interpolating position, the two triangle meshes are layered on top of one another and visually combined using alpha blending. If a pixel in the resulting C_α^i or D_α^i has a contribution from only one target it is masked or discarded. To properly resolve depth ambiguities when regions of the warped images self-overlap, a simple depth test is performed using the warped depth images. These two additional blend operators allow corresponding regions of the 3D objects that happen to be occluded in a particular view to be properly reconstructed when all 2D morphs are recombined.

All 2D morphs are executed independent of one another in no particular order to produce a new set of $\{C_\alpha^1, C_\alpha^2, \ldots\}$ and $\{D_\alpha^1, D_\alpha^2, \ldots\}$ images. All that remains is the computation of a geometry proxy, G_α, from the newly morphed depth images to constitute a new $\mathcal{L}_\alpha$.

4.3. Shape and texture reconstruction

The final step of the algorithm is to recombine the $|\mathcal{L}_1|$ 2D morphed color and depth images back into a 3D representation. In particular the geometry proxy G_α must be computed. For every i, each depth value in the pixels of D_α^i is unprojected by K^i and added

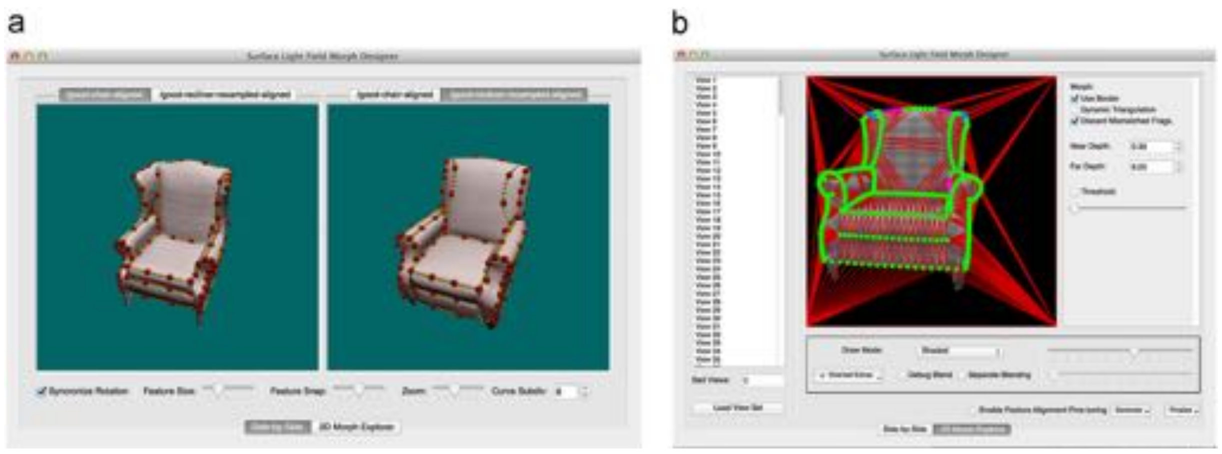

Fig. 3. Screenshots of our morph specification tool. (a) Feature-specification UI. Feature points are in red. Interpolated curve samples are in yellow. The texture of the chairs is hidden to make feature points more visible. (b) Interactive display of 2D morphs for each view with auxiliary information enabled. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

to a point cloud representing the morphed geometry. Note that the order in which the morphed depth images are added to the point cloud does not matter; the final set of points is the same. The point cloud will contain potentially noisy data, as well as duplicate points for regions that are visible in multiple images. The point cloud is filtered and smoothed in a manner similar to [27] to eliminate these issues. Finally Poisson surface reconstruction is applied to the point cloud to form the geometry G_α for the morph stage.

We leverage the freely available implementation of screened Poisson surface reconstruction from [13]. In the majority of cases, the default parameters suffice, but, depending on the scale of the objects, it is necessary to adjust the octree depth and related parameters. The default, or selected parameters, are constant for all steps of the morph; it has not proven necessary to select Poisson reconstruction parameters for individual stages. Poisson surface reconstruction requires oriented point clouds, but normals are not provided in the point cloud constructed above. Instead, these are generated algorithmically using the 'compute normals for points sets' filter in MeshLab [28]. This filter builds a neighborhood around each individual point to fit a plane, assigns the normal from that plane to the point, and smooths the results to help eliminate noise.

The computed geometry, G_α, morphed color images, C^i_α, morphed depth images, D^i_α, and original camera poses, K^i, form a new unstructured lumigraph, $\mathcal{L}_\alpha$. This can be rendered as a light field without any further processing or data generation. We opt for this approach because many of the real-world objects morphed in our results section include appearances with view-dependent attributes. Normally, the photogrammetry techniques used to acquire the targets discard such material information, but our view-based multi-2D morph algorithm allows that information to be preserved. For synthetic morphs, there is no reason spatially varying BRDF parameters could not be baked into textures and morphed in addition to the color. Surface normals could also be morphed in this manner to produce oriented point clouds, but MeshLab performed adequately. The primary benefit of our approach is that regardless of the data stored within the 2D data images, they are warped consistently across all channels allowing the results to be integrated back into three dimensions.

It is not required that the color or material information be morphed and reconstructed after the creation of G_α. If a shape morph is all that is desired, the sequence of computed geometries represents a valid geometric morph with similar behavior to implicit or volume based approaches. As shown in Section 5, in certain applications texture morphing is necessary to correctly specify the morph correspondences. In these scenarios, our approach affords more control over both structure and texture correspondences compared to implicit approaches.

4.4. Discussion and limitations

We have opted to use perhaps the simplest 2D morphing algorithm as the foundation for our 3D morph. The main benefit to using this triangle-based algorithm is we can guarantee that the exact same dense mapping is applied to both the color and depth images. 2D morph algorithms based on optical flow or energy minimization are inherently more difficult to apply to images with additional depth channels.

The linear interpolation used in our 2D morphing does impact the number of feature points placed, as well as the types of structural changes the objects may undergo, such as rigid body rotations. Not surprisingly these problems have been well-studied in the 2D morphing literature. For example, as-rigid-as-possible (ARAP) morphing [29] improves the interpolated paths used in the morph and automated image morphing [30] reduces the number of user-specified correspondences. So long as (1) the depth images can be transformed identically as the color images, (2) any depth ambiguities are eliminated using a depth test to display the 'nearest' color, and (3) only pixels visible in both targets are blended together, then any 2D morphing algorithm should be usable. In this way our 3D morphing algorithm is highly modular, allowing novel 2D morphing contributions to be used directly. With the recent use of ARAP in distance field morphing [16], the applicability of more complex 2D morph techniques such as those described here seem feasible despite the potential complications of maintaining consistency in the depth channel. We save this exportation for future work.

Lastly, a note on performance. In practice we do not execute the morph in real time due to the computational demands of constructing a triangle mesh, G_α, and the I/O operations for saving the images to disk. Instead the various $\mathcal{L}_\alpha$'s are precomputed for a number of in-between positions equally spaced along the liner interpolation paths. There are three factors that affect the theoretic runtime of our algorithm: the number of views $|\mathcal{L}|$, the number of correspondences n, and the number of interpolated positions precomputed m. All operations that are affected by n have a computational cost close to zero compared with rendering or surface construction. $|\mathcal{L}| \cdot m$ 2D morphs are generated, and m point clouds and triangle meshes are constructed from those morphed images. The mesh generation is the most significant bottleneck, although transferring the imagery from the GPU comes second.

As a concrete example, the gnome morph shown in Section 5 has $n = 390$, $|\mathcal{L}| = 110$, and α is sampled 15 times. The images in each view of the gnome morph are 1170×1762. The generation of 2D morphed images took 9.49 min, but this process creates 3300 images, which is approximately 173 ms per image. The time per image includes the cost of transferring the image from the GPU and saving it to disk. When parallelized across morph stages, point cloud generation took 5.25 min and Poisson surface construction took 29.15 min using a depth of 10. These measurements were made on a 2013 MacBook Pro laptop with an Intel i7.

5. Results

To demonstrate the effectiveness of our approach we have created 3D morphs of shape and texture for several real-world objects. Most of these examples were created using photogrammetry (as implemented in the program PhotoScan by Agisoft) and represent scans of real objects. In this section we break down the results to first demonstrate morphing of shape, second illustrate morphing of texture detail, and finally show complete examples where shape and texture are morphed simultaneously.

5.1. Shape metamorphosis

The top of Fig. 4 shows a standard change of genus computed with our algorithm. Here the canonical genus 0 shape is transformed into the canonical genus 1 shape with a clean transition as the hole opens in the middle. Many of the shapes that were acquired with PhotoScan represent complex topologies of arbitrary genus. Our approach is insensitive to these complexities. At the bottom of Fig. 5 we show a genus change where a protruding loop that is present on one of the targets was absent on the other. The change is similar to the previous canonical example but is part of a much more complex object that was scanned photographically. This is a close up of a full morph depicted at the bottom of Fig. 7 showing how the handle must close up and shrink into the smooth surface to achieve an accurate morph. In the complete morph the topology is changing from genus 1 on the bell to genus 5 on the wine vessel. A similar demonstration of a change of genus (from genus 0 to genus 1) is shown in the middle of Fig. 4. Here we have recreated an example morph from [15] where a dinosaur transforms into an iron with a closed handle. The necessary handle opens up cleanly as the shapes change.

In the bottom of Fig. 4, we show a morph between surfaces with different numbers of boundaries. The first chair is not watertight (the bottom is open with a single boundary edge). The second chair is watertight with no boundary edges. Our approach fills in the non-solid interior of the chair with the boundary eventually spanning the chair bottom to have a full watertight surface. Note that there is also a change in genus from 2 to 0 (there is a slight gap between the back of the chair and the bottom on the non-watertight target). This particular example was generated synthetically using a virtual photogrammetry system rather than photographs of real chairs, but the missing data on the bottom of the first chair is representative of the holes that may be present in photographically acquired shapes. The top of Fig. 5 shows some more difficult geometry where a thin feature from one target (the dried stem of a pumpkin) transforms into a small handle on the top of a wine vessel (see middle of Fig. 7 for full morph). Both of these shapes were photographically acquired and had significant noise present especially for the thin pumpkin stem. Our approach handles this situation without issues.

5.2. Texture metamorphosis

In Fig. 6 we show a morph where the correspondences that needed to be established were not represented in the geometry. The portraits of Picasso and Van Gogh have feature correspondences placed on key facial features like the eyes and mouths. For this particular example, however, the portrait was simply textured onto a flat plane with no geometry to identify those features. Existing mesh-based methods would require the plane to be tesselated into a control mesh to accomplish this morph. Existing volumetric methods, on the other hand, such as [16] would lose all of the detail in the paintings and would only morph the geometry of the frame. The paintings would have to be morphed using a separate 2D morphing algorithm and then merged with the 3D frame morph manually. The ability of our technique to simultaneously handle a color texture morph makes this type of transformation much simpler. In fact, many of the morphs we created for this work benefited from the presence and accuracy of the texture morph. The first example shown in Fig. 1 required similar features to the portrait morph that were more prominent in the color data than in the geometry. It was also vital that these features stay well registered as the morph executes.

The example at the top of Fig. 8 also benefited from the presence of texture. Here the garden gnome statues had many correspondences that when placed at the edges or points of geometric features did not properly align with the edge of a material or paint color. Using the color imagery as a guide made for a much better registered correspondence. We demonstrate with these examples how correspondence features can be placed on the surface regardless of the presence of an edge or vertex in the proxy geometry. In our approach, the morph is computed by projecting these features into the 2D views along with the depth and color information. The features guide the morph without the need to tessellate or re-mesh the underlying geometry.

5.3. Full morph examples

We designed several example morphs using our technique most of which used targets that were photographically acquired. The objects shown in Fig. 7 were constructed from 80–100 photographs taken at multiple angles under fixed lighting. These photographs were processed with Agisoft PhotoScan to determine the extrinsic and intrinsic properties of the camera and to reconstruct an accurate geometric proxy. All morphs were designed using the interface shown in Fig. 3. Establishing the correspondence required one to two hours of interaction but note that this work also affords control over the motion and style of the final morph. The images used for these objects were taken in an unstructured and hand-held manner and were not identical for the two targets in the morph. This required resampling one of the targets so that the camera angles and projected views matched. We found that this required only a trivial change to our existing lumigraph rendering system and did not introduce significant artifacts.

The top of Fig. 7 shows two real pumpkins with varying appearance and a shape morph that includes accurate registration of the rather thin stem from the green pumpkin. The middle of Fig. 7 shows the same green pumpkin transforming into an ancient Chinese wine vessel with a somewhat similar cylindrical shape but very different surface texture. This ritual bronze vessel (Léi)[2] is part of the collection at the (withheld for review) Minneapolis Institute of Arts and was photographed under very different conditions than the pumpkin. The resampling involved did not

[2] https://collections.artsmia.org/index.php?page=detail&id=972

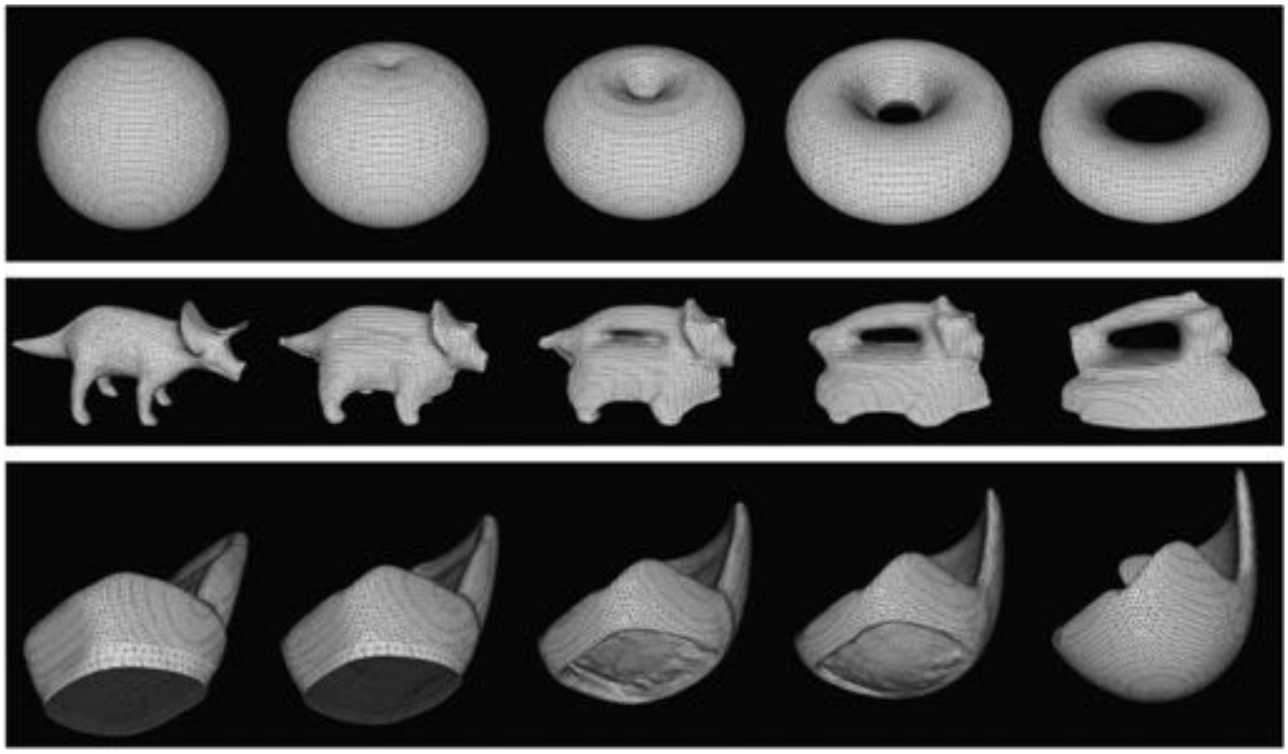
Fig. 4. Examples of our algorithm handling difficult geometry. Top: a basic change of genus computed with our approach; middle: recreation of an example in [15] showing a change of genus from 0 to 1; bottom: morphing between a non-watertight surface (chair with an open bottom) and a watertight one (chair enclosed on all sides).

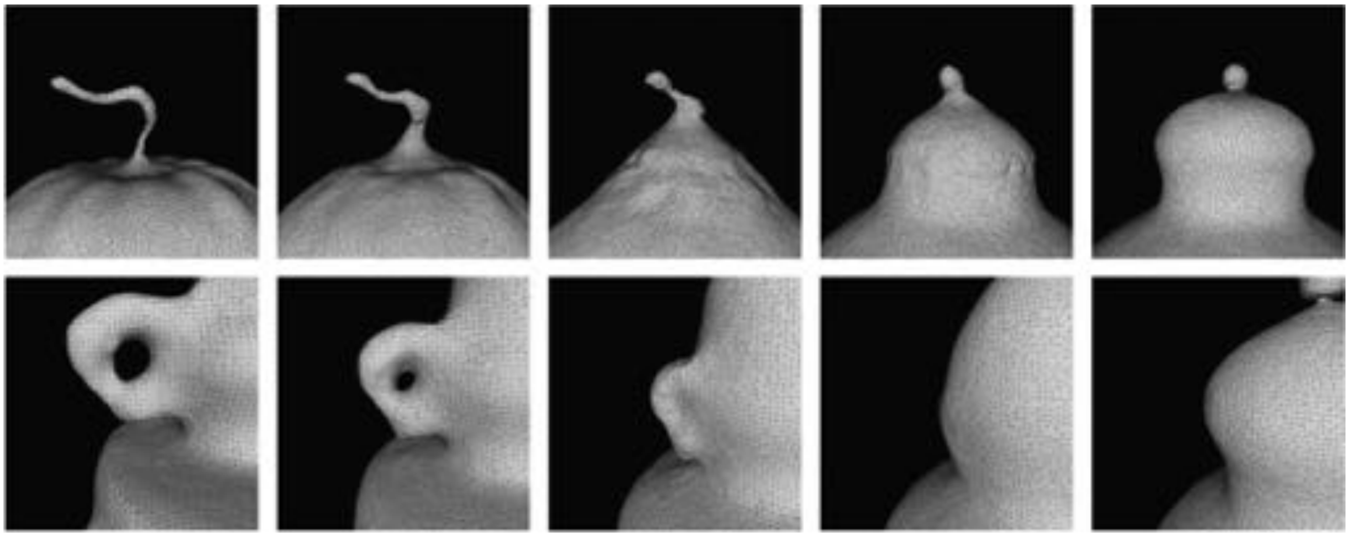
Fig. 5. Difficult geometry from photographically acquired objects processed with our algorithm. Top: a thin feature (a dried pumpkin stem) morphing into a handle on top of a wine vessel; bottom: a close up of the full morph shown in the bottom of Fig. 7 showing a change in genus as the bell morphs into the wine vessel.

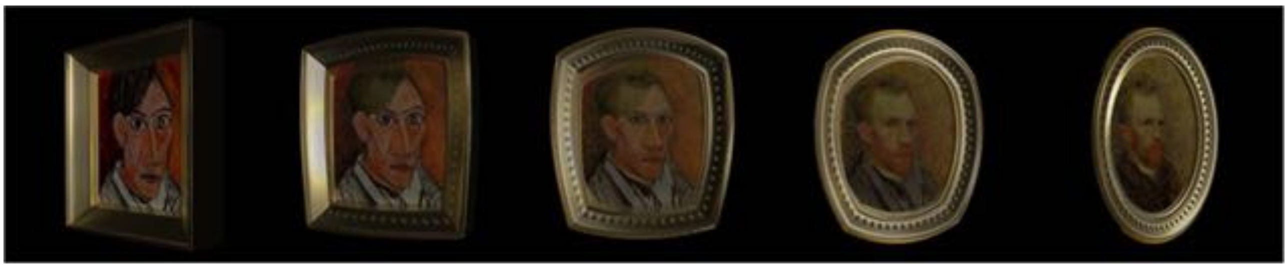
Fig. 6. The important features in these two portraits are not present in the geometry so the texture is vital to establishing a good correspondence.

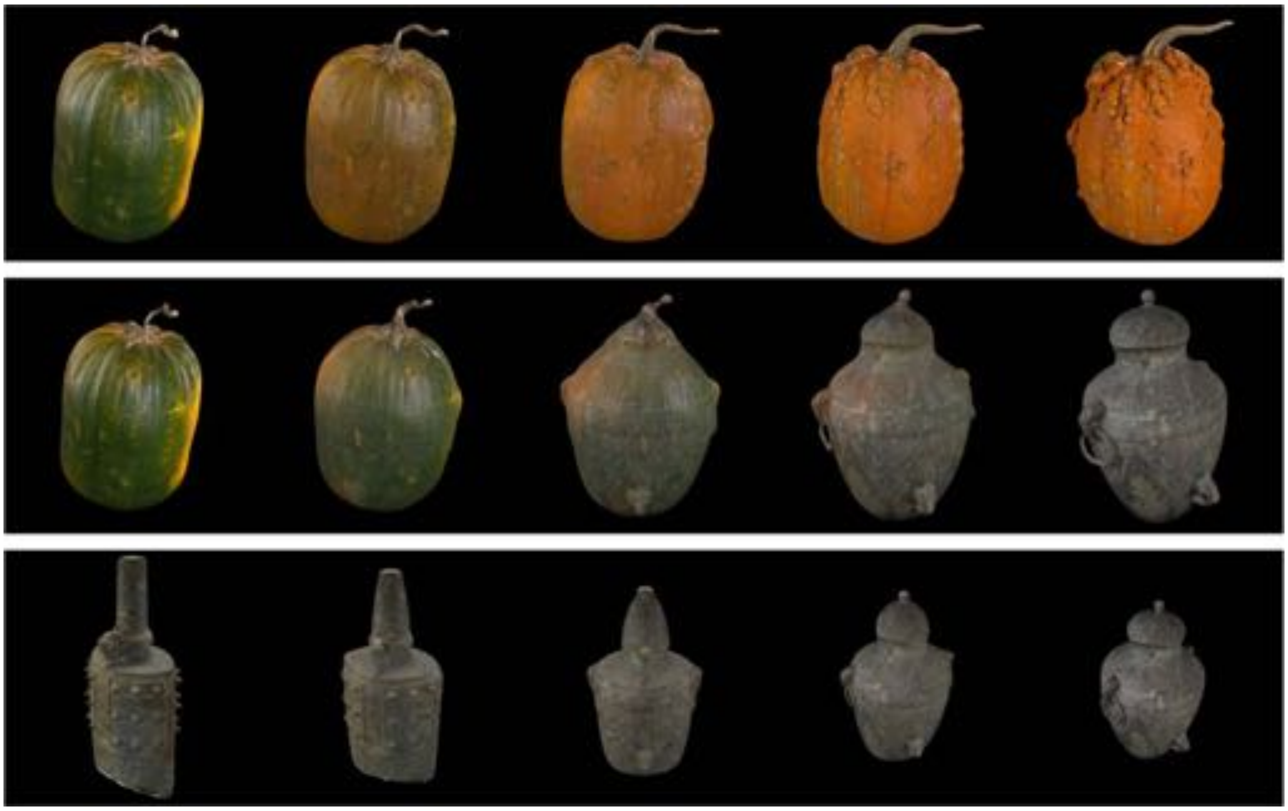

Fig. 7. Morphing of several photographically acquired objects.

degrade the quality of the resulting lumigraph used for the morph. The bottom of Fig. 7 shows a morph that includes the same wine vessel as well as another Chinese bronze artifact from the same museum, an ancient bell[3]. This example represents a potential application of this work allowing comparison of these artifacts without disturbing them or even having the artifacts be physically present. While the color properties are somewhat similar between these two objects their shapes are quite different and constructing a metamorphosis requires handling a change in genus from 1 to 5. Our approach naturally handles all of these challenges with no special treatment.

The top of Fig. 8 was also constructed from real objects scanned using photogrammetry techniques. Here, two small garden gnomes with similar poses and colors are morphed. The transition between these two objects could provide useful information for character animation and design in a video game. Game design has increasingly engaged photogrammetry as a tool in the character modeling process and our algorithm can help establish correspondences between these objects for generation or retargeting of animation between multiple characters or skins.

The middle and bottom of Fig. 8 are meant to demonstrate the usefulness of our approach for common design tasks. The middle sequence shows a morph between doors from a stock and custom Ford Mustang with different shape and color smoothly transitioning from one to the other. An automotive designer could use this visualization to try and find the ideal positioning of key elements like side-view mirrors or door handles and key holes. Contour lines adjust subtly between these two objects, and at any point in-between the designer might find the ideal aesthetic of their automobile.

The bottom of Fig. 8 features a fashion design task where two stylistically different shoes with tall heels are compared. There is a significant change in the size of the opening from one to the other as well as a dramatic shift from the textured red leather to the leopard print. In-between we see many alternatives that include shape and appearance qualities from both targets. As with the automotive design task, here any one of these in-between points could serve to communicate the intent of the design in a precise, digital manner. The designer will have a fully interactive, 3D representation of the morph with reasonable and realistic color properties.

[3] https://collections.artsmia.org/index.php?page=detail&id=821

5.4. Morph accuracy

The accuracy of the morph depends on a number of different factors. First we will discuss the accuracy of the shape morph before considering the resulting color texture produced by our approach. Warping depth images instead of a conventional triangle mesh allows us to morph between objects that would otherwise have very different tessellations. Interpolating the depth values in linear space accurately samples the morphed shape. However, because we discard regions of the images that do not have contributions from one of the targets, it is theoretically possible for every view to discard the depth information for the same surface region. In this case the point cloud would have no information and a hole would form. A direct view on a surface provides the most pixel samples to convert into points, but even an angled surface relative to the camera plane can provide enough information to reconstruct the morphed shape. This is the case with the shoe morph shown at the bottom of Fig. 8, which undergoes rather drastic shape changes around the opening. In many views the shoe opening is occluded for one target and visible in the other. In all the morphs we have prepared, we have yet to see a situation where there was not enough depth information to reconstruct an accurate point cloud. If this does occur, it is possible to increase the resolution or number of views captured so the surface is better recorded.

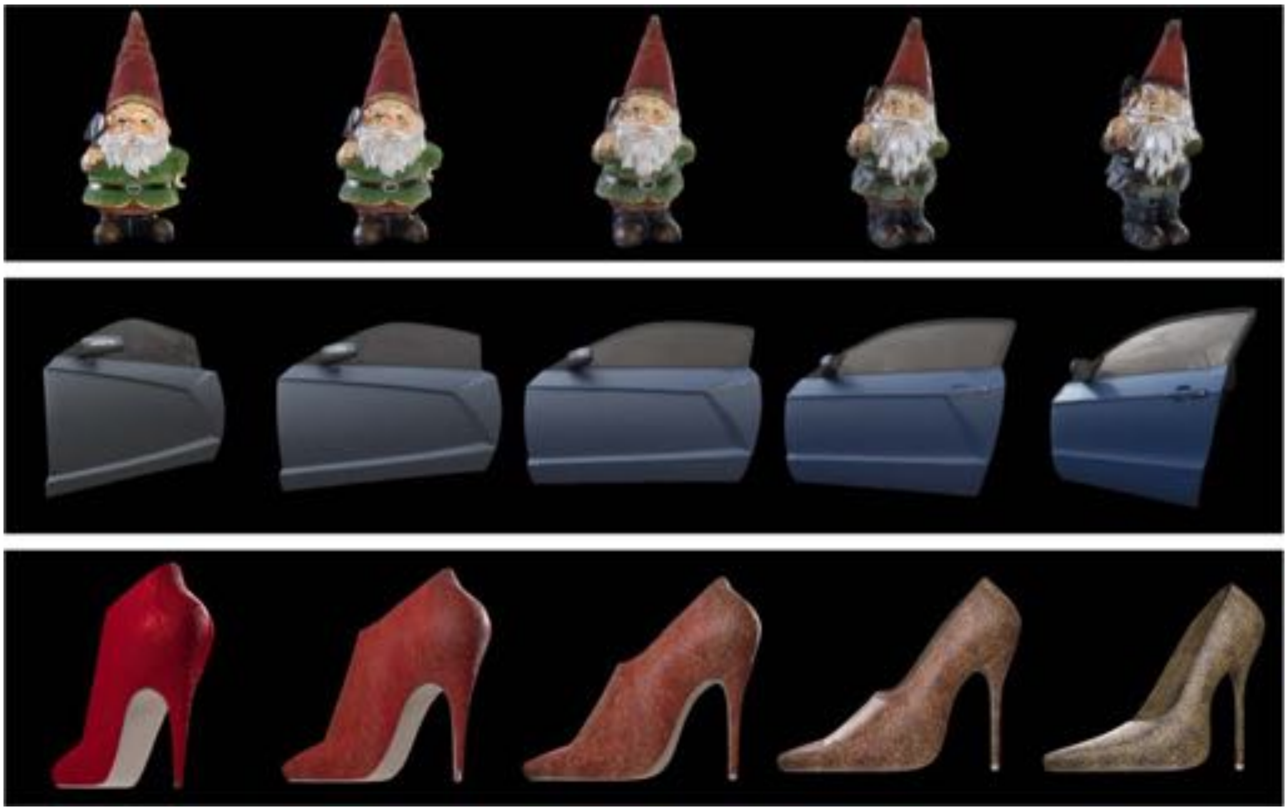

Fig. 8. Full 3D shape and texture morphs of various design artifacts.

Masking can also affect the view-dependent texturing in the final object, even when the shape reproduction is of high quality. If a pixel would have captured specular reflection but was masked out, the camera blending function may select samples that have the expected diffuse reflection but are from a view that did not receive the specular reflection. This has the tendency to dim specular highlights that are near the silhouette edges of an object. Specular reflections in unstructured lumigraphs are already difficult and is generally considered a sampling problem. The same applies here and given how small the artifacts are, we do not think it detracts from the quality of our morphs. Indeed, it highlights that our morphing algorithm is robust to noise and aliasing produced by 2D morphing algorithms.

The interpolation that is done as part of our morph to determine the color of the intermediate object is conceptually similar to the Gouraud shading that is done to compute the color of a pixel on an image scan line, and it is subject to the same limitations. Because the lighting and viewing conditions are the same for each pair of corresponding images used to determine each object's lumigraph representation, the color interpolation will be correct for objects that have the same shape and surface reflectance properties (limited by the ability of the unstructured lumigraph rendering technique to correctly represent specular reflection). If the shape, or surface reflectance, or both are different between the two objects then the color interpolation will be approximate. Interpolation of surface normals (as in Phong shading) and other reflectance parameters would be necessary to improve the morphed color in this case.

5.5. *Future work*

The identification of the 3D correspondences in our current implementation is a manual step. The user clicks on the surfaces to place and adjust features in correspondence on the two targets. There is ample opportunity to introduce either partial or full automation here by employing techniques from computer vision and the larger body of morphing.

In addition, our current implementation only supports two targets. There is nothing about our algorithm that precludes supporting an arbitrary number of targets but the implementation complexity would naturally increase.

Lastly, as mentioned in Section 4.4, there is potential to employ a more advanced 2D morphing algorithm. As long as care is undertaken to preserve the assumptions mentioned, this has the potential to improve the 3D metamorphosis as a whole by computing more rigid interpolation paths or eliminating misalignment due to our simple bi-linear approach.

These are all promising directions for future work.

6. Conclusion

In this paper we have presented a new technique for creating a morph between two three dimensional objects. Given the images and depth maps that are the starting point for an unstructured lumigraph representation for two separate objects, we establish correspondence points between the two objects and project the location of these points onto the images and the depth maps. Two dimensional morphing techniques are then used to create a new set of images and depth maps that can be used to form a lumigraph representation for an intermediate version of the two objects. The interpolated depth maps are employed to produce a point cloud representation that is polygonized to generate a geometric proxy for the new object and the warped images are processed and re-projected onto that proxy to create a final lumigraph representation.

For a problem that typically requires complex algorithms from computational geometry, our approach achieves plausible results by reducing the inherently three dimensional problem into a set of two dimensional problems where morphing has been far more successful. We obtain consistently good results for photographically scanned objects where genus, manifoldness and Euler characteristic can vary dramatically. While we only employ simple color interpolation in our current method, it gives reasonable results for diffuse surfaces commonly used to produce unstructured lumigraph representations.

We believe that our results have practical applications in several different areas. For movie special effects, the morphed object can be photographed using continuous camera movements instead of making pictures from a single fixed point of view as must be done today for existing two dimensional morphing techniques. In design, exemplars that capture desired aspects of the final product can be used as endpoints in our morphing method to experiment with blended shapes and colors that may lead in new design directions. In game creation, character models can be retargeted through the correspondence we compute and animation can be designed using real figures and natural poses that may better suit the skills of the designer.

Acknowledgments

The data used to create the example shown in Fig. 1 was provided by Lee Perry-Smith of Infinite Realities LTD. Photographs of the Chinese vase and bell artifacts shown in Figs. 7 and 5 were provided by Dan Dennehy and Charles Walbridge of the Minneapolis Institute of Arts. The authors would like to express their gratitude to these individuals and everyone that has helped contribute to this work. This research was funded by the Ford Motor Company and by a University of Minnesota GIA award.

Appendix A. Supplementary material

Supplementary data associated with this paper can be found in the online version at http://dx.doi.org/10.1016/j.cag.2015.05.005.

References

[1] Alexander O, Rogers M, Lambeth W, Chiang JY, Ma WC, Wang CC. The digital emily project: achieving a photorealistic digital actor. IEEE Comput Graph Appl 2010;30(4):20–31.
[2] Debevec P. The light stages at UC Berkeley and USC ICT. ⟨http://gl.ict.usc.edu/LightStages⟩; 2008.
[3] Beier T, Neely S. Feature-based image metamorphosis. Comput Graph 1992;26(2):35–42.
[4] Lee SY, Chwa KY, Hahn J. Image morphing using deformable surfaces. In: Proceedings of computer animation'94., Geneva, Switzerland Los Alamitos, CA: IEEE Computer Society Press; 1994. p. 31–9.
[5] Ruprecht D, Müller H. Image warping with scattered data interpolation. IEEE Comput Graph Appl 1995;15(2):37–43.
[6] Adelson EH, Bergen JR. The plenoptic function and the elements of early vision. Comput Models Vis Process 1991:3–20.
[7] Levoy M, Hanrahan P. Light field rendering. In: Proceedings of SIGGRAPH '96. New York, NY, USA: ACM; 1996. p. 31–42. ISBN 0-89791-746-4.
[8] Wood DN, Azuma DI, Aldinger K, Curless B, Duchamp T, Salesin DH, et al. Surface light fields for 3d photography. In: Proceedings of SIGGRAPH '00. New York, NY, USA: ACM; 2000. p. 287–96. ISBN 1-58113-208-5.
[9] Buehler C, Bosse M, McMillan L, Gortler S, Cohen M. Unstructured lumigraph rendering. In: Proceedings of SIGGRAPH '01. New York, NY, USA: ACM; 2001. p. 425–32. ISBN 1-58113-374-X.
[10] Chen WC, Bouguet JY, Chu MH, Grzeszczuk R. Light field mapping: efficient representation and hardware rendering of surface light fields. ACM Trans Graph 2002;21(3):447–56.
[11] Zhang Z, Wang L, Guo B, Shum HY. Feature-based light field morphing. ACM Trans Graph 2002;21(3):457–64.
[12] Wang L, Lin S, Lee S, Guo B, Shum HY. Light field morphing using 2d features. IEEE Trans Vis Comput Graph 2005;11(1):25–34.
[13] Kazhdan M, Hoppe H. Screened poisson surface reconstruction. ACM Trans Graph 2013;32(3):29:1–13.
[14] Lazarus F, Verroust A. 3d metamorphosis: a survey. Vis Comput 1998;14:8–9.
[15] Cohen-Or D, Solomovic A, Levin D. Three-dimensional distance field metamorphosis. ACM Trans Graph 1998;17(2):116–41.
[16] Weng Y, Chai M, Xu W, Tong Y, Zhou K. As-rigid-as possible distance field metamorphosis. Comput Graph Forum 2013;32(7):381–9 URL⟨http://dx.doi.org/10.1111/cgf.12246⟩..
[17] Kent JR, Carlson WE, Parent RE. Shape transformation for polyhedral objects. In: Proceedings of the 19th annual conference on computer graphics and interactive techniques. SIGGRAPH'92. New York, NY, USA: ACM; 1992. p. 47–54. ISBN 0-89791-479-1.
[18] Praun E, Hoppe H. Spherical parametrization and remeshing. In: ACM SIGGRAPH 2003 papers. SIGGRAPH'03. New York, NY, USA: ACM; 2003. p. 340–49. ISBN 1-58113-709-5.
[19] Lee AWF, Sweldens W, Schröder P, Cowsar L, Dobkin D. Maps: multiresolution adaptive parameterization of surfaces. In: Proceedings of the 25th annual conference on computer graphics and interactive techniques. SIGGRAPH'98. New York, NY, USA: ACM; 1998. p. 95–104. ISBN 0-89791-999-8.
[20] Lee AWF, Dobkin D, Sweldens W, Schröder P. Multiresolution mesh morphing. In: Proceedings of SIGGRAPH'99. New York, NY, USA: ACM; 1999. p. 343–50. ISBN 0-201-48560-5.
[21] DeCarlo D, Gallier J. Topological evolution of surfaces. In: Proceedings of the conference on graphics interface'96. GI'96. Toronto, Ontaria, Canada: Canadian Information Processing Society; 1996. p. 194–203. ISBN 0-9695338-5-3.
[22] Dinh HQ, Yezzi A, Turk G. Texture transfer during shape transformation. ACM Trans Graph 2005;24(2):289–310.
[23] Eisemann M, Decker BD, Magnor M, Bekaert P, Aguiar ED, Ahmed N. Floating textures. In: EuroGraphics 2008, vol. 27, no. 2; 2008.
[24] Davis A, Levoy M, Durand F. Unstructured light fields. Comp Graph Forum 2012;31(2pt1):305–14.
[25] Jeong E, Yoon M, Lee Y, Ahn M, Lee S, Guo B. Feature-based surface light field morphing. In: Proceedings of the 11th Pacific conference on computer graphics and applications. Washington, DC, USA: IEEE Computer Society; 2003. p. 215–23. ISBN 0-7695-2028-6.
[26] Weisstein EW. Sphere point picking from mathworld—a wolfram web resource. URL ⟨http://mathworld.wolfram.com/SpherePointPicking.html⟩; 2015.
[27] Hornung A, Kobbelt L. Interactive pixel-accurate free viewpoint rendering from images with silhouette aware sampling. Comput Graph Forum 2009;28(8):2090–103.
[28] ISTI CNR. Meshlab. URL ⟨http://meshlab.sourceforge.net⟩; 2015.
[29] Alexa M, Cohen-Or D, Levin D. As-rigid-as-possible shape interpolation. In: Proceedings of SIGGRAPH'00. New York, NY, USA: ACM; 2000. p. 157–64. ISBN 1-58113-208-5.
[30] Liao J, Lima RS, Nehab D, Hoppe H, Sander PV, Yu J. Automating image morphing using structural similarity on a halfway domain. ACM Trans Graph 2014;33(5):168:1–12. http://dx.doi.org/10.1145/2629494 URL ⟨http://doi.acm.org/10.1145/2629494⟩.