Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2007)
D. Metaxas and J. Popovic (Editors)

Animation Collage

Christian Theobalt [1], Christian Rössl [2], Edilson de Aguiar [3], and Hans-Peter Seidel [3]

[1]Stanford University
[2]INRIA Sophia-Antipolis, France
[3]MPI Informatik, Germany

Abstract

We propose a method to automatically transform mesh animations into animation collages, i.e. moving assemblies of shape primitives from a database given by an artist. An animation collage is a complete reassembly of the original animation in a new abstract visual style that imitates the spatio-temporal shape and deformation of the input. Our algorithm automatically decomposes input animations into plausible approximately rigid segments and fits to each segment one shape from the database by means of a spatio-temporal matching procedure. The collage is then animated in compliance with the original's shape and motion. Apart from proposing solutions to a number of spatio-temporal alignment problems, this work is an interesting add-on to the graphics artist's toolbox with many applications in arts, non-photorealistic rendering, and animated movie productions. We exemplify the beauty of animation collages by showing results created with our software prototype.

Categories and Subject Descriptors (according to ACM CCS): J.5 [Arts and Humanities]: Fine arts I.3.7 [3D Graphics and Realism]: Animation I.3.5 [Computational Geometry and Object Modeling]: Geometric algorithms

1. Introduction

The collage is an abstract and expressive visual style that has found its way into many forms of visual arts, including painting, photography and sculpturing. When creating a collage (from the French word coller=glueing), the artist intends to build a new whole by assembling different primitive forms in a special way.

Many researchers in computer graphics have been inspired by the idea to develop algorithms that enable the computer and even unexperienced users to reproduce the look of certain styles of visual arts. This is also true for collages. Kim et al. [KP02] have developed a system that can automatically turn arbitrary photographs into collage mosaics that comprise of an arrangement of elementary image tiles. Rother et al. [RBHB06] automatically arrange and blend photographs from a database into a perceptually pleasing way. Gal et al. [GSP*07] present a recent method to approximate static 3D shapes with other meshes, but they do not handle the case of mesh animations.

In this paper, we introduce a new method that brings together the traditional art form of a collage with the most prominent art form in computer graphics, namely 3D animation. Our algorithm allows the computer artist to automatically convert her favorite mesh animation into a moving assembly of 3D shape primitives in a database. This *animation collage* is glued together in such a way that it approximates the sequence of shapes of the original mesh animation, while deforming in the same spatio-temporally consistent way as the original. While the method can fully-automatically build moving collages, our algorithm purposefully gives the artist the possibility to postprocess and fine-tune the results according to her imagination. To our knowledge, the proposed method is the first of its kind in the literature.

Since to most people the term collage is synonymical with a static piece of art, we had to firstly define what ought to be the visual style of an animation collage. This particular style eventually determines the new algorithmic contributions of our approach which are detailed in the remainder of this paper. First, we decided to process mesh animations since this makes our algorithm independent of the specific way that was used to create the input in the first place. Second, our method ought to not only approximate the shape of meshes at single time steps individually. It rather decomposes and reassembles the mesh animations in such a way that the time-varying shape and the spatio-temporal deformation of the assembly is plausible. We concluded that the most appropriate way to achieve this goal is to automatically decompose the mesh animations into moving approx-

SCA 2007, San Diego, California, August 04 - 05, 2007

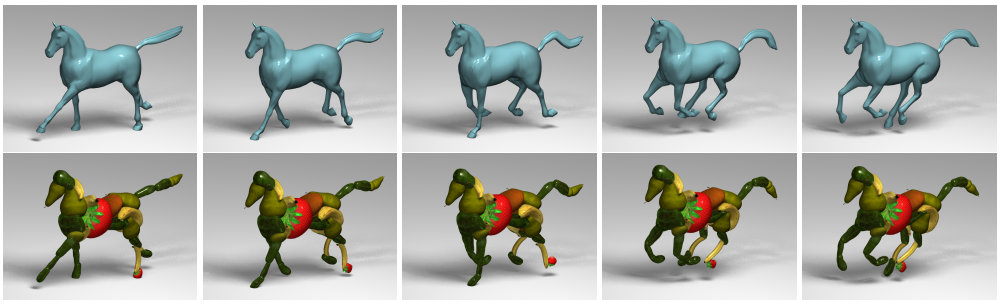


Figure 1: *Our method automatically generates moving 3D collages out of mesh animations by rebuilding them as moving assemblies of shape primitives. In the example above the galloping horse has been transformed into two galloping sets of fruits.*

imately rigid volume segments, henceforth called *animation cells*. This decomposition is learned from the moving input meshes by means of a spectral clustering approach. Our algorithm employs a spatio-temporal matching criterion that analyzes the motion and deformation of each animation cell and finds a shape primitive in the database that best approximates its time-varying geometry. Shape primitives and cells are spatio-temporally aligned and the fitted shapes are moved and deformed according to the deformation of the cells. The deformations are driven via spatio-temporally consistent offset shapes that are automatically built from the animation cells and whose tightness controls the deformation's stiffness. Since it was our aim to develop new algorithmic recipes for a novel artistic tool, we took care that the animator can influence the final result at all stages of the processing pipeline.

2. Related Work

While our idea was inspired by the work of visual artists, our algorithmic approach was inspired by previous research on mesh and animation decomposition, image mosaicing, as well as shape matching and approximation.

Recently, researchers proposed to exploit temporal coherence in mesh animation for compression [Len99]. Sattler et al. [SSK05] propose to use Lloyd relaxation and clustered principal component analysis to group vertices based on their trajectories. Günther et al. [GFW*06] also apply a Lloyd relaxation to motion-decompose mesh animations for fast ray-tracing. However, in contrast to our segmentation approach, their method does not consider spatial coherence of the clustered surface areas, their optimization may run into local minima, and automatic determination of the number of clusters is difficult. James and Twigg [JT05] proposed an algorithm to cluster triangles in mesh animations based on their rotations using a mean shift approach. While their segmentation yields statistically meaningful near rigid areas, they neither correspond to underlying kinematic hierarchies nor are they guaranteed to be spatially connected.

In contrast to prior work, our approach generates more meaningful spatially and temporally consistent segmentations of mesh animations. In other words, they closely correspond to the true kinematic skeleton hierarchy if the input can be assumed to have one, as in the case of our moving animals. Our decomposition approach is based on spectral clustering, it is robust and produces plausible results even for challenging motions. As an additional benefit, it can produce segmentations at different user-controlled levels of detail, and it can also automatically detect the optimal number of clusters [NJW02].

Our segmentation approach is related to recent methods from optical motion capture that automatically reconstruct kinematic skeletons from marker trajectories [KOF05, HSDK05, YP06, dATS06]. They are based on the idea that the segmentation of these marker-trajectories into rigid bodies corresponds to finding an optimal clustering into mutually intersecting motion spaces, each of them corresponding to one rigid body. To perform this segmentation, they typically cluster marker trajectories based on distance invariants between points on the same segments. We capitalize on these ideas, and exploit similar distance invariants, a spatial coherence criterion, and a volumetric decomposition approach to subdivide mesh animations into plausible approximately rigid segments using a robust spectral clustering approach.

Animation collages has also been inspired by the work on automatic image mosaicing [KP02, Hau01]. In a similar line of thinking, we developed a way to decompose mesh animations into spatio-temporally meaningful volume cells. By this means, we can efficiently find approximating shapes for each cell individually. Furthermore, the explicit cell decomposition enables us to rebuild animations with a minimum of overlap between shapes while only having to solve local matching problems.

The approximation aspect of our pipeline is also related to previous work on fitting of static shapes with ellipsoids or general quadrics. Such fitting methods have been used in geometry processing, e.g., for efficient data transmission [BK02], medical visualization [BJMR01], or segmentation and piecewise shape approximation [WK05] in reverse engineering. In our application, we don't have the freedom to

construct best approximating shapes, but are bound to work with given primitives from a database.

Therefore, we can also capitalize on previous work on 3D shape matching. Global shape matching approaches compare different shapes based on numerical descriptors, such as shape distributions [OFCD01], Fourier descriptors [VSR01], moment invariants [ETA02, NK03], or spherical harmonics [KFR03]. Local shape matchers can also identify correspondences between subparts of shapes using artificial surface features [GCO06] or topology graph-based alignment [HSKK01]. Since we are attacking a time-dependent matching problem, we have developed a novel spatio-temporal matching approach based on spherical harmonic descriptors [KFR03].

Also related to our approach is the idea presented by Gal et al. [GSP*07]. They intend to rebuild static triangle meshes from individual shapes in a database. Static shape approximation is a very challenging research problem in itself. In building collages of animated scenes, we have to take on challenges very different from the static setting.

We have combined a variety of algorithmic recipes to rebuild mesh animations as sequences of approximating shapes. To our best knowledge, this is the first paper in the literature that takes on this problem. While we propose a solution to an interesting spatio-temporal matching problem, we also produced a tool that is fun to use. It enables easy creation of an artistic visual style that has many applications in non-photorealistic rendering, cartoon production and the visual arts in general.

3. Overview

The input to our algorithm comprises of two elements. The first is a triangle mesh animation of N time-steps, i.e. a mesh structure $\mathscr{M} = (\mathscr{V}, \mathscr{T})$ with constant graph structure — here denoted by vertices $\mathscr{V}$ and their triangulation $\mathscr{T}$ — and time-dependent data: associated to every vertex $v_j \in \mathscr{V}$ is a sequence of 3D positions $p_t(v_j) = (x_j, y_j, z_j)_t$, $1 \leq j \leq \#\mathscr{V}$, $0 \leq t < N$. Then coordinate sets $P_t = \{p_t(v_j)\}$ together with $\mathscr{M}$ describe a time-varying surface. The second input element is a database of K static shapes, each being represented as a textured triangle mesh.

The first step in our pipeline is the motion-decomposition of the mesh. To this end, we employ a segmentation based on spectral clustering that analyzes the motion of the mesh and delivers contingent triangle patches of $\mathscr{M}$'s surface that represent approximately rigid elements, Sect. 4 and Fig. 2(a). To enable the fitting of shape primitives, we transform the rigid surface elements into approximately rigid volume cells, so-called animation cells. To this end, a sequence of medial axis meshes is computed from the animation which, in conjunction with the previously identified rigid surface segments, is used to create these closed volume cells, Sect. 5 and Figs. 2(b),(c). Once the animation cells have been identified, we automatically fit to each of them a shape primitive from the database. For each animation cell, an optimal shape for fitting is selected based on spherical harmonic shape descriptors and a spatio-temporal matching procedure, Sect. 6. The final moving collages are generated by deforming the fitted shapes according to the transformation of their respective animation cells, Fig. 2(d). To achieve this, we generate spatio-temporally consistent offset meshes from the animation cells that drive the shape primitives' deformations, Sect. 7.

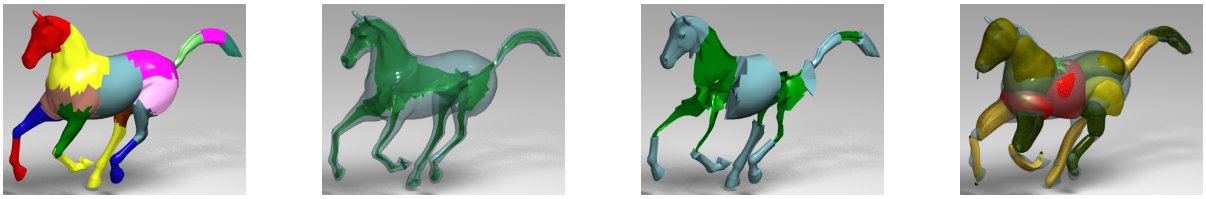
4. Rigid Body Segmentation

The first step in our pipeline segments the input animation given by $\mathscr{M}$ and P_t into spatially coherent patches of triangles that undergo approximately the same rigid transformations over time-steps $0 \leq t < N$. Our surface segmentation examines the motion trajectories of the mesh's vertices and their mutual distance variation over time. Our motivation for decomposing the mesh into approximately rigid patches is that this seems intuitive and plausible to the viewer. Many characters in cartoons and animation films, for instance the main actors in 20-th century Fox's "Robots" [Rob05], were rendered in this particular style.

The deformations of general mesh animations can not be described by rigid transformations alone. Animators often purposefully combine rigid transformations with non-rigid ones in order to create a lifelike look. In contrast to motion segmentation approaches that generate merely a statistically plausible segmentation, we intend to isolate the underlying rigid deformations from the non-rigid ones in a kinematically plausible way. By this we mean that in case there exists a true kinematic segment hierarchy, our method approximates it as good as possible. For us it is also important that regions on the surface are spatially connected, a requirement that is also not fulfilled by many related methods such as [JT05, GFW*06]. Only if this is assured, a faithful decomposition into volume cells becomes feasible, Sect. 5.

We apply the spectral clustering method of Ng et al. [NJW02] to a subset of sample vertices. Similar to [dATS06] in the context of optical motion capture, we use an affinity criterion to obtain clusters of similarly moving vertices, which we then transform into rigidly moving coherent triangle patches, Fig. 3.

Given is a mesh $\mathscr{M} = (\mathscr{V}, \mathscr{T})$. To initialize our segmentation approach, we select a subset $\mathscr{V}_s \subset \mathscr{V}$ of ℓ sample vertices that are required to be distributed evenly on the mesh. For sample selection we only consider a reference pose P_{t_r} (typically the first time step $t_r = 0$ of the animation), and employ the curvature-based segmentation method of Yamauchi et al. [YGZS05] to decompose $\mathscr{M}$ into ℓ surface patches. The sample vertices are chosen as the vertices closest to the patch centers. This subsampling enables reasonably fast de-



(a) (b) (c) (d)

Figure 2: *Important steps in our pipeline: the mesh is decomposed into rigidly moving surface patches (a), skeletons are extracted (b), and animation cells assembled (c), here only some cells are shown. Shapes are spatio-temporally fitted to the cells and deformed over time to build the animation collage (d).*

composition of even large meshes. We typically choose ℓ to be in the range of $1000-2000$.

The motion trajectories of the sample vertices throughout the whole animation form the input to our spectral clustering approach which groups them into k approximately rigidly moving groups. It exploits the invariant that mutual distances between points on the same rigid patch should only exhibit a small variance while the mesh is moving.

Spectral clustering commences by building a spatial affinity matrix $\mathbf{A}$. For our purpose, we define $\mathbf{A}$ with entries

$$a_{i,j} = \exp\left(-\frac{\sigma_{i,j}}{\rho^2}\right), \qquad (1)$$

where $\sigma_{i,j}$ is the standard deviation of the Euclidean distance $\delta(i,j,t)$ between sample vertex i and sample vertex j over time and $\rho = \frac{1}{N^2}\sum_t \delta(i,j,t)$ is a scaling term controlling convergence behavior (where N is again the number of frames in the sequence). The scaling term reduces the affinity values of vertex pairs with large average mutual distance. Thereby, we try to force spectral clustering to put spatially far apart groups of vertices with similar motion into separate clusters. Intuitively, the symmetric matrix $\mathbf{A}$ encodes the likelihood that two samples lie on the same rigid body.

The clustering approach proceeds by rearranging the ℓ input samples in k well-separated clusters on the surface of a k-dimensional hypersphere. It achieves this by building a diagonal matrix $\mathbf{D}$ whose (i,i)-element is the sum of $\mathbf{A}$'s i-th row. Now the Laplacian matrix $\mathbf{L} = \mathbf{D}^{-1/2}\mathbf{A}\,\mathbf{D}^{-1/2}$ is built, its k largest eigenvalues $e_1,\ldots,e_k$ are computed and stacked into columns to form the matrix $\mathbf{X} \in \mathbb{R}^{\ell\times k}$. The rows of $\mathbf{X}$ are normalized and considered as points in $\mathbb{R}^k$. Then the ℓ row vectors are split into k clusters using standard k-means clustering. Every original sample vertex $v_i \in \mathscr{V}_S$ is assigned to a cluster j if and only if row i of $\mathbf{X}$ was assigned to cluster j. We remind that k-means clustering is effective because in the transformed data $\mathbf{X}$ clusters are well-separated.

The above procedure separates the sample vertices into k distinct approximately rigidly moving clusters of vertices. From this, we associate triangle clusters $T_0 \cup \ldots \cup T_{k-1} = \mathscr{T}$ by assigning each triangle $\Delta = (w_0, w_1, w_2) \in \mathscr{T}$ to the marker vertex v_j whose average geodesic distance to w_0, w_1, and w_2 is minimal. Δ is added to that triangle cluster for which the sample v_j is part of the associated vertex cluster. The resulting clusters partition the mesh. Note that (a discrete approximation to) geodesic distance is used here to provide contiguous surface patches.

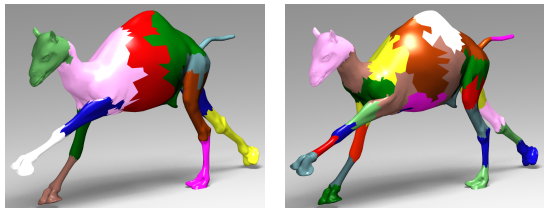


Figure 3: *Segmentation into approximately rigid surface patches – While with $k = 13$ patches (left) merely the larger rigid segments are identified, but the feet are merged with the lower legs, at $k = 31$ the full kinematic detail has been discovered (each color used several times, adjacent segments colored differently).*

Spectral clustering has a number of intriguing advantages over other clustering approaches. Conveniently, the optimal number of clusters may be inferred from the eigen-gap of $\mathbf{X}$ [NJW02]. However, the optimal segmentation is not always the one favored by the artist. Thus she also has the opportunity to specify the number of rigid surface patches that she would like to find. In Fig. 3, the segmentation of the camel's mesh is shown for two different values of k. At a smaller k-value, the lower legs and the hoofs form one cluster since the relative motion between these two was less significant than the relative motion between other segments. For a larger k-value, they have been split in two, which shows that with increasing level k of detail, our segmentation intuitively produces plausible and more detailed segmentations. Furthermore, spectral clustering is robust against outliers, performs well on segmentation problems with a challenging cluster structure on which other approaches, like standard k-means, perform significantly worse [NJW02].

Generally, our affinity metric (1) in combination with the triangle clustering procedure generates coherent surface patches in all our test cases which is important for later steps of our overall pipeline. In the rare case that we obtain disconnected patches nonetheless, we split them.

5. Building Animation Cells

Approximately rigid surface patches are not yet the appropriate shape representation for building animation collages. Although each patch is the outer boundary of a volumetric subsegment that moves approximately rigidly, it does not describe the spatial extent of this subsegment in the interior of the original mesh. To approximate these volumetric subsegments, we extend each surface patch into a so-called *animation cell*, i.e. a closed watertight triangle mesh that bounds an approximately rigidly moving slice of the original mesh's volume. Fig. 2(c) shows some animation cells computed from the horse animation. This volumetric decomposition of input animations has a couple of advantages. First, volumetric animation cells define 3D placeholders to which approximating shapes are to be fitted in order to generate visually pleasing collages with decent shape and deformation approximation. Thus, volumetric decomposition is a clever way to break down the fitting problem for the whole mesh into a set of fitting problems for individual cells. If we make sure that a fitting shape approximates the outline of an animation cell at each time step of the video, we implicitly take care that mutual shape intersections remain in the range that is anyway needed to prevent big holes in the collage. Furthermore, by deforming approximating shapes like their encompassing rigid cells, the deformation of the shapes remains in visually pleasing bounds.

The input for building animation cells is the set of rigid surface patches $T_0, \ldots, T_{k-1}$ that was computed in the previous section. It is our goal to extend each surface patch into a closed and watertight animation cell mesh. Looking only at the graph structure of each patch, this is easily achieved by inserting a new vertex for every boundary loop, triangulating the arising fan, and thereby removing the boundary loop. Although the principal idea is fairly easy, a proper way to insert the additional vertex for the boundary loop is crucial. In the remainder of this section, we discuss how we compute closing vertices based on skeleton meshes and describe how we generate spatio-temporally consistent cells.

We firstly compute a sequence of medial axis meshes $\mathscr{S}_0, \ldots, \mathscr{S}_{N-1}$ from the input animation, Fig. 2(b). We take care that the number of vertices and the connectivity of each $\mathscr{S}_i$ match the properties of the respective $\mathscr{M}_i$ that it was computed from: for computing the skeletal mesh $\mathscr{S}_i$, we employ the Voronoi-based two-sided approximation of the medial axis that has originally been proposed in [HBK02]. Every vertex of $\mathscr{M}_i$ is associated with a Voronoi cell. One-to-one correspondences between the vertices of $\mathscr{M}_i$ and the vertices of $\mathscr{S}_i$ are established by using the Voronoi poles as skeletal mesh vertices [ABK98]. The connectivity of $\mathscr{S}_i$ is copied from $\mathscr{M}_i$. In consequence, we have established one-to-one correspondences between both vertices and triangles of the outer and the skeleton mesh. In order to remove undesired spikes in the skeletal mesh, we employ tangential Laplacian smoothing.

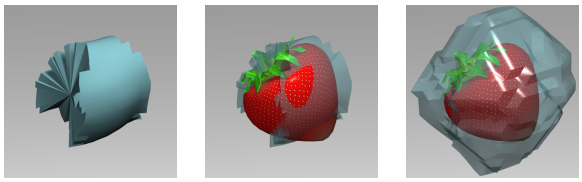


Figure 4: *An animation cell at a certain time-step for the center segment of the horse (left), the strawberry fitted to it (center), and the offset cell for the center segment (right).*

As the skeletons share the graph structure of the animation meshes they can be partitioned into the same patches. In the following, we describe how to build animation cells for all patches at a single time step t. By applying the same procedure to all time steps, we generate the appropriately deformed versions of each cell. Consider a patch T_k and its associated vertex positions taken from P_t. For all boundary loops of T_k at time-step t, we compute the center of gravity of associated vertex positions in the skeleton $\mathscr{S}_t$. The new vertex for fan-triangulation of the boundary loop is positioned at this center. Fig. 4 (left) illustrates this using a center segment of the horse animation as an example.

Note that a simple strategy like choosing the center of gravity of the boundary loop's vertices for fan-triangulation would not have fulfilled our requirements. Our experiments showed that in this case very flat animation cells may occur for extreme geometric configurations which would lead to inappropriate volumes for the subsequent steps of our method. Depending on the geometric setting, similar problems may occur if one tries to directly triangulate a hole without inserting a vertex. Although our cell decomposition does not strictly partition the volume, it generates volume slices that are tailored to our purpose.

6. Assembling the Collage

As already outlined in the previous section, the sequence of moving animation cells can be regarded as a sequence of volumetric placeholders to which approximating shapes from the database are fitted. In our opinion, it is the most reasonable and visually most appealing way to approximate each cell with a single shape. While by this means, it may not be possible to exactly reproduce the true shape of each cell, in particular its outer surface, the overall appearance of the mesh animation is still very faithfully approximated. We already argued that the decomposition of the animation into cells bears many further algorithmic advantages. The overall fitting problem simplifies to a fitting problem between individual shapes and segments. Furthermore, it becomes easier to assure that the shapes which we fit not only match the outlines of their respective animation cells at a single time step. Since the animation cells undergo mainly rigid deformations, the slight non-rigid deformations of the approximating shapes that may be necessary to assure good approximation over the whole sequence can be kept in reasonable bounds, Sect. 7.

To put this into practice, a shape similarity measure is needed, Sect. 6.1, that we apply in our spatio-temporal matching and alignment approach to fit a database shape to a deforming animation cell, Sect. 6.2.

6.1. Shape Similarity Measure

Due to the diversity of database shapes, and potential differences in orientation and uniform scaling, we require a similarity measure that is rotation-, pose- and scale-independent. Spherical harmonic descriptors fulfill all these requirements and have proven to be superior to many other global descriptors [KFR03]. Given a mesh $\mathscr{K}$ we compute its spherical harmonic descriptor as follows: First, the spatial occupancy function $O(x,y,z)$ is sampled on a regular grid within the mesh's bounding box by rasterizing the mesh into a voxel volume of dimension $\ell_1 \times \ell_2 \times \ell_3$ [NT03, Min03]. The voxel volume is intersected with q equidistant spheres $W_0,\ldots,W_{q-1}$ that are concentrically arranged around the center of gravity of the voxel set. The discretized occupancy function is resampled on the surface of each spherical shell, yielding q spherical functions $o_0(\theta,\phi),\ldots,o_{q-1}(\theta,\phi)$. Each of these q spherical functions is decomposed into its harmonic frequency components

$$f_\ell(\theta,\phi) = \sum_{m=-\ell}^{m=\ell} a_{\ell m} Y_\ell^m(\theta,\phi),$$

where ℓ is the frequency band, $a_{\ell m}$ is the m-th coefficient on band ℓ, and Y_ℓ^m is the m-th spherical harmonic basis function for band ℓ [Gre03]. For each spherical function, the norms of its frequency components are computed as

$$SH(o_h) = \{\|f_0(\theta,\phi)\|, \|f_1(\theta,\phi)\|, \ldots, \|f_{\ell-1}(\theta,\phi)\|\}.$$

The complete shape descriptor $D(\mathscr{K})$ is the two-dimensional $q \times h$-array that is indexed by the sphere radius and the frequency band. Four our purpose, we found that descriptors with $h = 20$ and $\ell = 10$ are sufficient. The difference $d(D_1,D_2)$ between two descriptors D_1 and D_2 is obtained by interpreting each of them as $(q \cdot h)$-dimensional vectors and computing the angle that they span.

6.2. Spatio-temporal Shape Fitting

Using the above shape descriptor and the associated distance measure, we find a shape from the database that best matches the time-varying shape of each animation cell throughout the whole sequence. To keep processing times and memory consumption in reasonable bounds while sampling the range of deformations sufficiently densely, we propose the following approach to fit a shape to one animation cell $\mathscr{Z}_j$:

At first, a set of representative time steps $0 \le t_1,\ldots,t_r < N$ is chosen in which the whole mesh undergoes a characteristic range of deformations. Let the vertex positions of $\mathscr{Z}_j$ at the r time steps be $P_{t_1}(\mathscr{Z}_j),\ldots,P_{t_r}(\mathscr{Z}_j)$. For each of the r cell poses, a descriptor is computed, yielding a set $U = \{D_{t_1}(\mathscr{Z}_j),\ldots,D_{t_r}(\mathscr{Z}_j)\}$. For each of the shapes in the database and their associated descriptors D_i, a global distance to all descriptors in U is computed as

$$d_{\text{glob}}(D_i,U) = \sum_{t\in t_1,\ldots,t_r} d(D_i,D_t(\mathscr{Z}_j)) \quad \text{for } 0 \le i < K\,.$$

The above distance assesses the spatio-temporal goodness-of-fit between a database shape and a cell over time. Accordingly, the index c of the database shape that matches the shape of $\mathscr{Z}_j$ at all representative time steps best is found as

$$c = \operatorname*{argmin}_i d_{\text{glob}}(D_i,U)$$

Database shape c is fitted to the single pose $P_{t_m}(\mathscr{Z}_j)$ of $\mathscr{Z}_j$ out of all the representative poses which best matches the shape of c, thus

$$t_m = \operatorname*{argmin}_{t\in\{t_1,\ldots,t_r\}} d_{\text{glob}}(D_c,P_t(\mathscr{Z}_j))$$

This way we guarantee that the shape is fitted to the optimally matching configuration of the cell, and thus the required transformation of c during the fitting itself is minimal. The effect of spatio-temporal fitting is compared against single-time step fitting in Fig. 5.

Figure 5: *Effect of spatio-temporal fitting for the tail of the camel (left): One frame of the result animation is shown with a single (center) or three representative time steps (right) used during spatio-temporal fitting.*

The fitting itself first coarsely registers shape c and $\mathscr{Z}_j$ in pose $P_{t_m}(\mathscr{Z}_j)$ by aligning their centers of gravities and principal components in orientation and scaling. Thereafter, the initial fitting is refined by running an ICP-like alignment [BM92].

Although shape matching and fitting are fully automatic processes, user interaction is possible to meet artistic preferences. On the one hand, the user can restrict matching to a subset of the database or even manually choose a collage shape which will be fitted into the cell. On the other hand, she can manually adjust the fitted shape's position, scale and orientation.

Finally, we note that while there are many applications for *partial* shape matching, the use of a *global* approach is essential to our method: our goal is to faithfully fill in the whole cell. This way, self-intersections and artifacts during the animation are minimized since the cell's have a near-rigid structure. Furthermore, bigger holes in the collage are prevented. Consequently, we decided to fit only one shape per cell to provide the most plausible results for the animation. Putting it another way, we first simplify the global filling problem to a local one using cell decomposition and cell matching, and only then individual collage shapes are matched globally to cells.

Figure 6: *With larger offsets, the fruits in the horse's leg deform more stiffly (right) than with tighter ones (left).*

7. Animating the 3D collage

In the final stage of our method we compute the animation collage from the set of animation cells and associated collage shapes. Remember that each of the collage shapes has potentially been fitted to a different reference time step t_m by the procedure described in the previous section.

Our approach proceeds again cell-wise: per-frame transformation of the animation cells is propagated to the fitted shapes. We consider this transformation as a general deformation of the cell, which is expected to be nearly rigid. Among the many surface deformation methods which are available in the literature we chose free-form deformation based on 3D mean-value coordinates [Flo03, FKR05, JSW05], because it can directly process our input data, produces good results and is moreover robust and simple in implementation. Using the standard approach, the triangulation of the animation cell would serve directly as control mesh for the deformation: the collage shape is rigidly fitted into the geometry of the animation cell $P_{t_m}(\mathscr{Z})$ at a reference time step t_m, and then mean value coordinates are used to reconstruct shapes which deform like the geometry of cell $P_t(\mathscr{Z})$ at any other time step $0 \leq t < N$. However, there are two requirements which render this immediate deformation impractical for our special animation collage setting: first, we want to provide the user some additional global control over the deformation, second we have to ensure numerical robustness. Both requirements are related, and we apply a two-step deformation with spatio-temporal offsets which is described in the following.

The key idea of our deformation is to use certain morphological offsets of the animation cells as control meshes, i.e., roughly speaking, the cells' geometry is extruded in direction of the surface normal (but avoiding self-intersections). This way, we can control the stiffness of the deformation by taking advantage the well-known behavior of the pseudo-harmonic fields induced by mean value coordinates for the new, enlarged boundary polyhedra, Fig. 6. One such offset cell for the torso segment of the horse is shown in Fig. 4(right) At the same time we avoid numerical instabilities resulting from evaluating the field in the vicinity of the control mesh, a situation which cannot be generally avoided when using the animation cells' geometry directly. Now the challenge is to efficiently compute *offset cells* which are not only geometrically consistent but provide also consistent control meshes for different time-steps similar to the original animation cells.

Our offsets can be easily constructed in a volumetric representation using level-sets. As accuracy is not crucial for us here, we use an even simpler approach and compute a discrete voxel model for each animation cell mesh (from [Min03]) which is then dilated and converted back into a triangle mesh using marching cubes. Here the voxel resolution and dilation radius define the offset radius. This way we compute an offset cell for each animation cell w.r.t. the reference time step. The computation is efficient and guarantees watertightness and no self-intersections, however, we lose control over the combinatorial structure of the offset mesh, i.e., it does not share the animation cell's connectivity. Therefore we require an alternative mapping to the animation cell which is provided by a second deformation step.

The final animation of a single collage shape proceeds as follows: first, the time-dependent geometry of the animation cell is used as control mesh to compute a deformed offset cell. Second, the offset cell is used as control mesh and its deformation is propagated to the collage shape. Note that by volumetric construction the genus of the offset cell might differ from that of the respective animation cell. However, using the free-form deformation approach influencing the the cell's volume this does not imply any problem.

The offset radius provides an additional parameter to control the effect of the deformation: the more distant the offset, the stiffer the deformation. Fig. 6 shows an example. While this parameter has no physically plausible meaning we find it intuitive to use without asking too much of the user.

8. Results and Discussion

We have generated a variety of animation collages from mesh sequences of a *galloping* and a *collapsing horse*, a galloping *camel*, as well as two scanned humans performing a motion captured *jump* and a *cartwheel* (video only). For none of the sequences, ground truth kinematics, e.g. in the form of animation skeletons, are available. Our shape database comprises of 20 collage shapes, ranging from fruits, over industrial shapes like screws, to barrels and bottles (see additional material). Even the authors as unexperienced artists were able to create a variety of animation collages with different appealing visual styles by restricting the database to subsets of shapes, Fig. 7 shows a selection. The collages do not only have very aesthetic looks at single-time steps, but nicely approximate the spatio-temporal appearance of the input animation, see Fig. 1 and, in particular, the accompanying video. On all our test sequences, our motion-based segmentation approach created plausible animation cells that closely correspond to the true kinematics of the moving subjects, a prerequisite for appealing animation collages, see Fig. 3.

The top row in Fig. 7 shows freeze-frames from the front and the back of an animation collage showing a *jumping human*. For the original sequence motion capture data was used to animate a laser-scan. With both fruits, as well as barrels,

Seq.	N	$\#\mathcal{T}$	k	Seg	Cell	Fit	Anim
jumping	216	26 312	25	416	401	46	171
gal. horse	48	16 843	35	297	190	64	134
camel	47	43 758	13	914	139	59	131
cartwheel	120	7 318	11	45	188	22	68

Table 1: *Typical animation data: Shown are time-steps N, number of triangles in mesh $\#\mathcal{T}$, and number of animation cells k. Furthermore, measured computation times (in seconds) are given for four parts of the pipeline (see Sect. 8).*

bottles and screws, the 25 computed animation cells are both faithfully and sometimes funnily approximated. We particularly liked the head being automatically approximated by a strawberry or a pear. Please refer to the video to also see that due to our offset-based deformation, the moving collages maintain the subtle motion details of the input which lends them a very human-like motion style.

The *galloping horse* is shown for fine and coarse segmentations with 35 and 17 animation cells, respectively. In comparison one sees clearly the non-rigid transformation for the coarse segmentations, for instance in the lower legs where the hoofs are not covered by additional cells. However, even these deformations appear rather natural due to stiffness provided by the offset cells. Of course the individual impression depends on the viewers particular expectations and preferences, and spending more animation cells usually leads to better approximations of rigid parts. Notably, many viewers to who we presented our work particularly liked the slight non-rigid deformations. Moreover, our choice of using a single collage shape per cell is justified by the fact that the animation looks plausible and there are hardly any holes and only few self-intersections.

The *galloping camel* is shown for fine (left block) and coarse segmentations (right block) with 31 and 13 animation cells, respectively. We would like to point out the nice approximation of the hunch, e.g., by pears (coarse) and by bananas (fine). Especially the arrangement of bananas in is non-trivial and the impression of a sophisticated design is given. Such arrangements are enabled by the partition in animation cells, and the example thus nicely confirms our algorithmic approach to break the fitting problem into cell-wise problems.

The majority of results in Fig. 7 was generated fully-automatically. Only in some cases, marked with a black dot in the upper right, we changed the position or the kind of at most two shapes to match personal preferences. Table 1 summarizes information on input animation complexity and run-times measured on a Pentium IV 3.0 GHz. Our approach is very efficient: run-times are dominated by rigid body segmentation (**Seg**) and cell generation (**Cell**), both of which need to be run just once per input sequence. In particular for large meshes most of the processing time for segmentation is due to the geodesic distance computation when assembling triangle patches. The most time-consuming part during cell computation is the mean-value-coordinate-based generation of offset segments. Shape fitting and alignment (**Fit**), as well as the animation of the collage (**Anim**) are performed once for each collage style created from an input sequence. Even for sequences with more than 200 frames, both steps can be finished in less than four minutes. Run-times could be further sped up by working with decimated input or cell meshes, however we always worked on the originals. In comparison to the rendering times of around 4 minutes per frame (with POVRAY at 1600x1200 resolution), the overall processing times are negligible. The same is true for any of the manual corrections which were performed in only a couple of seconds.

Our approach is subject to a couple of limitations. While our *clustering* algorithm provides excellent results for the shown input animations, its output generally depends on these inputs: a meaningful segmentation can only be expected if there is actual relative motion of all limbs in an animation. We conclude that this approach works best for animations with kinematic structure. For the *melting horse* animation shown in the video this is not given. Therefore, the galloping horse segmentation was used to generate it which yields good results.

For our *fitting* we remark that the following limitations apply: inherently, fitting is limited by the shapes available in the database. If no proper shape is available, unnatural deformations may be necessary. As the database also defines the visual style, we leave this as an artistic problem that has to be considered by the user. From a more technical point of view, our approach requires that segments span a volume. The approach is thus not able to fit collage shapes to, e.g., nearly planar segments. Note that the spatio-temporal segmentation (which provides the segments) is not at all affected. In addition, for fitting we assume that the animation cells are in motion and hence being transformed. This is different to considering a static fitting problem, where alternative approaches such as Gal et al. [GSP*07] may show better results, and where it may be advantageous to allow for more shapes per cell.

Despite these limitations we presented a compendium of spatio-temporal matching and segmentation techniques that enable us to build beautiful animation collages from mesh animations.

9. Conclusion

We defined the animation collage as a visual style which expresses a time-variant input shape in terms of an arrangement of shape primitives in space-time. While this is motivated from static scenes, the processing of animations turns out to be a setting of its own which reveals new and different problems. With our novel approach to fully-automatic creation of animation collages from animations we particularly provide new solutions to a couple of geometric problems: automatic and kinematically plausible segmentation of the animated mesh into near-rigid surface patches, spatio-temporal matching and fitting of shapes from a database, and

spatio-temporally consistent transfer of rigid transformation and non-rigid deformation from the time-varying surface to collage shapes. A key ingredient of our method is the motion-based decomposition of the overall fitting problem into smaller local ones. Our software prototype is fun to use and allows also untrained people to produce high-quality results. For future work we see a variety of applications of our algorithmic components in animation compression, spatio-temporal matching and fitting. Furthermore, color and texture harmony could be incorporated as matching criteria.

Acknowledgements. This work is supported by EC within FP6 under Grant 511568 with the acronym 3DTV, the German Academic Exchange Service (DAAD), and the Max-Planck Center for Visual Computing and Communication. We would like to thank Torsten Langer, Hitoshi Yamauchi, and Shin Yoshizawa for letting us use their code.

References

[ABK98] AMENTA N., BERN M., KAMVYSSELIS M.: A new voronoi-based surface reconstruction algorithm. In *Proc. ACM SIGGRAPH* (1998), pp. 415–421.

[BJMR01] BANÉGAS F., JAEGER M., MICHELUCCI D., ROELENS M.: The ellipsoidal skeleton in medical applications. In *ACM Solid Modeling and Applications* (2001), pp. 30–38.

[BK02] BISCHOFF S., KOBBELT L.: Ellipsoid decomposition of 3D-model. In *IEEE 3DPVT* (2002), pp. 480–489.

[BM92] BESL P., MCKAY N.: A method for registration of 3-d shapes. *IEEE Trans. PAMI 14*, 2 (1992), 239–256.

[dATS06] DE AGUIAR E., THEOBALT C., SEIDEL H.-P.: Automatic learning of articulated skeletons from 3d marker trajectories. In *Proc. ISVC* (2006), pp. 485–494.

[ETA02] ELAD M., TAL A., AR S.: Content based retrieval of vrml objects: an iterative and interactive approach. In *Proc. Eurographics workshop on Multimedia 2001* (2002), pp. 107–118.

[FKR05] FLOATER M. S., KÓS G., REIMERS M.: Mean value coordinates in 3D. *CAGD 22*, 7 (2005), 623–631.

[Flo03] FLOATER M. S.: Mean value coordinates. *CAGD 20*, 1 (2003), 19–27.

[GCO06] GAL R., COHEN-OR D.: Salient geometric features for partial shape matching and similarity. *ACM Transaction on Graphics 25*, 1 (2006), 130–150.

[GFW*06] GÜNTHER J., FRIEDRICH H., WALD I., SEIDEL H.-P., SLUSALLEK P.: Ray tracing animated scenes using motion decomposition. *Computer Graphics Forum (Proc. Eurographics) 25*, 3 (2006), 517–525.

[Gre03] GREEN R.: Spherical harmonic lighting: The gritty details. In *Tutorial: Game Develpers Conference* (2003).

[GSP*07] GAL R., SORKINE O., POPA T., SHEFFER A., COHEN-OR D.: Non-realistic expressive modeling. p. to appear.

[Hau01] HAUSNER A.: Simulating decorative mosaics. In *Proc. ACM SIGGRAPH* (2001), pp. 573–580.

[HBK02] HISADA M., BELYAEV A., KUNII T. L.: A skeleton-based approach for detection of perceptually salient features on polygonal surfaces. *Comp. Graphics Forum 21*, 4 (2002), 1–12.

[HSDK05] HORNUNG A., SAR-DESSAI S., KOBBELT L.: Self-calibrating optical motion tracking for articulated bodies. In *IEEE Virtual Reality 2005* (2005), pp. 75–82.

[HSKK01] HILAGA M., SHINAGAWA Y., KOHMURA T., KUNII T.: Topology matching for fully automatic similarity estimation of 3d shapes. In *Proc. ACM SIGGRAPH* (2001), pp. 203–212.

[JSW05] JU T., SCHAEFER S., WARREN J. D.: Mean value coordinates for closed triangular meshes. *ACM Transaction on Graphics (Proc. SIGGRAPH) 24*, 3 (2005), 561–566.

[JT05] JAMES D., TWIGG C.: Skinning mesh animations. *ACM Transactions on Graphics (Proc. SIGGRAPH) 24*, 3 (2005).

[KFR03] KAZHDAN M., FUNKHOUSER T., RUSINKIEWICZ S.: Rotation invariant spherical harmonic representation of 3d shape descriptors. In *Proc. Eurographics/ACM Symp. on Geometry Processing* (2003), pp. 156–164.

[KOF05] KIRK A., O'BRIEN J., FORSYTH D.: Skeletal parameter estimation from optical motion capture data. In *CVPR* (June 2005), pp. 782–788.

[KP02] KIM J., PELLACINI F.: Jigsaw image mosaics. In *Proc. ACM SIGGRAPH* (2002), pp. 657–664.

[Len99] LENGYEL J.: Compression of time-dependent geometry. In *Proc. I3D* (1999), pp. 89–95.

[Min03] MIN P.: binvox. http://www.google.com/search?q=binvox, 2003.

[NJW02] NG A. Y., JORDAN M., WEISS Y.: On spectral clustering: Analysis and an algorithm. In *Proc. NIPS 2002* (2002).

[NK03] NOVOTNI M., KLEIN R.: 3d zernike descriptors for content based shape retrieval. In *Proc. ACM Symp. on Solid Modeling* (2003), pp. 216–225.

[NT03] NOORUDDIN F. S., TURK G.: Simplification and repair of polygonal models using volumetric techniques. *IEEE TVCG 9*, 2 (2003), 191–205.

[OFCD01] OSADA R., FUNKHOUSER T., CHAZELLE B., DOBKIN D.: Matching 3d models with shape distributions. In *Proc. Shape Modeling International* (2001), pp. 154–166.

[RBHB06] ROTHER C., BORDEAUX L., HAMADI Y., BLAKE A.: Autocollage. *ACM Transactions on Graphics (Proc. SIGGRAPH) 25*, 3 (2006), 847–852.

[Rob05] ROBOTS:. www.robotsmovie.com, 2005.

[SSK05] SATTLER M., SARLETTE R., KLEIN R.: Simple and efficient compression of animation sequences. In *Proc. Symp. on Computer Animations* (2005), pp. 209–217.

[VSR01] VRANIC D., SAUPE D., RICHTER J.: Tools for 3d-object retrieval: Karhunen-loeve transform and spherical harmonics. In *IEEE MMSP* (2001), pp. 293–298.

[WK05] WU J., KOBBELT L.: Structure recovery via hybrid variational surface approximation. In *Comp. Graphics Forum (Proc. Eurographics)* (2005), vol. 24, pp. 277–284.

[YGZS05] YAMAUCHI H., GUMHOLD S., ZAYER R., SEIDEL H.-P.: Mesh segmentation driven by gaussian curvature. *The Visual Computer 21*, 8-10 (2005), 649–658.

[YP06] YAN J., POLLEFEYS M.: Automatic kinematic chain building from feature trajectories of articulated objects. In *Proc. CVPR* (2006), pp. 712–719.

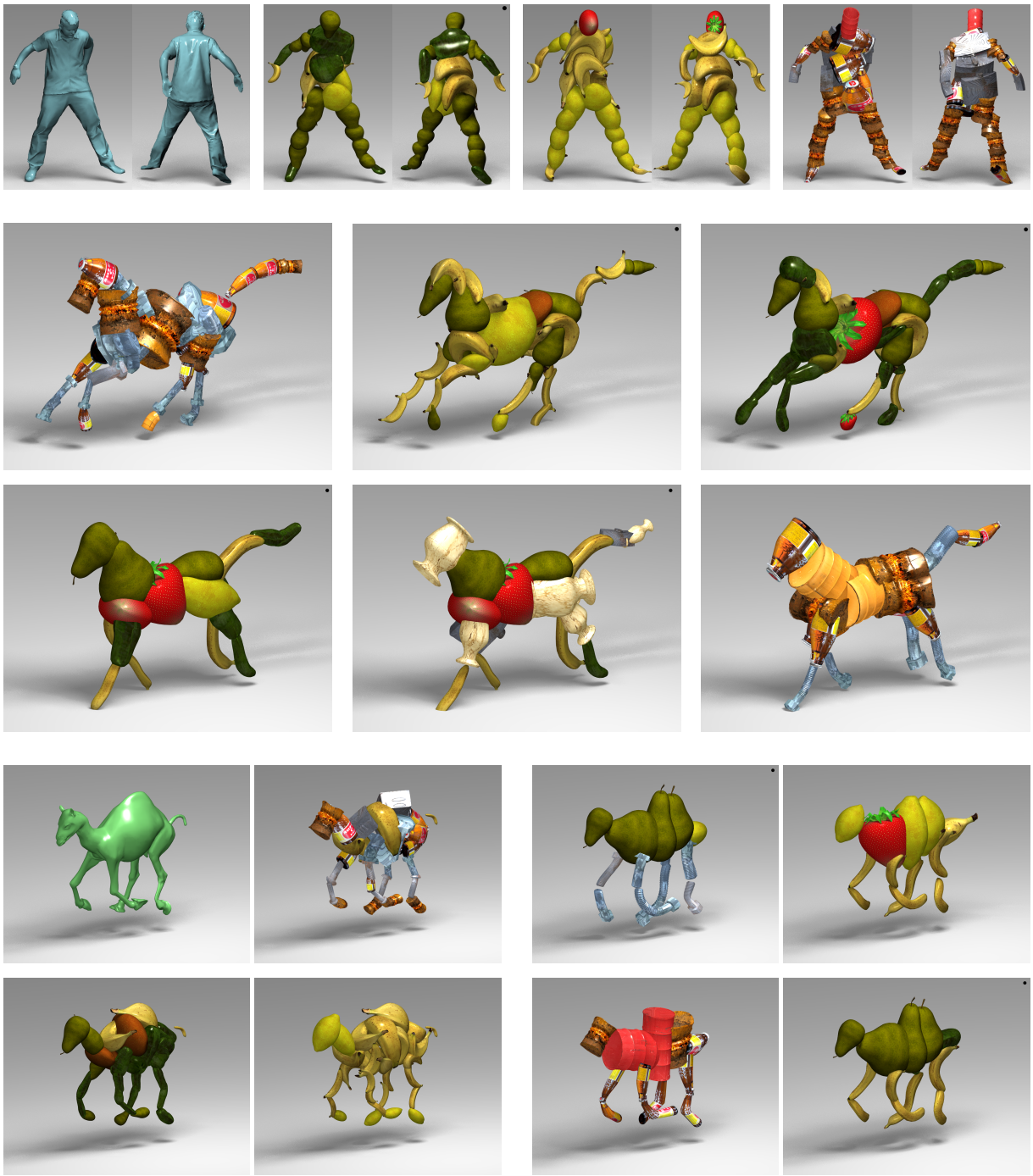


Figure 7: *Different visual styles for distinct frames of jumping human, galloping horse, and galloping camel sequences. Black dots in the upper right mark animations including manual changes of position or kind of at most two shape primitives.*