Human Video Textures

Matthew Flagg* Atsushi Nakazawa†* Qiushuang Zhang‡*
Sing Bing Kang§ Young Kee Ryu** Irfan Essa* James M. Rehg*

*Georgia Institute of Technology †Osaka University ‡Google, Inc. **Sun Moon University §Microsoft Research

Figure 1: *An example of a human video texture generated from 6 separate sets of motion capture data and video. The transition frames are shown clearly while the others are faded out for clarity. Colors correspond to specific clips which are interleaved between transitions.*

Abstract

This paper describes a data-driven approach for generating photo-realistic animations of human motion. Each animation sequence follows a user-choreographed path and plays continuously by seamlessly transitioning between different segments of the captured data. To produce these animations, we capitalize on the complementary characteristics of motion capture data and video. We customize our capture system to record motion capture data that are synchronized with our video source. Candidate transition points in video clips are identified using a new similarity metric based on 3-D marker trajectories and their 2-D projections into video. Once the transitions have been identified, a video-based motion graph is constructed. We further exploit hybrid motion and video data to ensure that the transitions are seamless when generating animations. Motion capture marker projections serve as control points for segmentation of layers and nonrigid transformation of regions. This allows warping and blending to generate seamless in-between frames for animation. We show a series of choreographed animations of walks and martial arts scenes as validation of our approach.

Keywords: image-based rendering, motion capture, layered motion

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

*This research was done while Nakazawa and Ryu were Visiting Professors and Zhang was a Masters student in the Computational Perception Lab at the Georgia Institute of Technology.

1 Introduction

The ability to produce photo-realistic animations of humans in a controllable manner is highly desirable for interactive games. For example, in the genre of side-scrolling fighter videogames, the game Mortal Kombat was one of the first to employ photographs of actors rather than the animated synthetic characters employed by its leading predecessor, Street Fighter. The heightened realism provided by Mortal Kombat's early form of image-based rendering likely contributed to its vast popularity. Other applications include the creation of interactive celebrity animations which capture the realism of a well-known actor or actress but provide for interactive control. Photo-realistic avatars are also potentially well-suited for training systems specializing in negotiation and other strategic person-to-person interaction scenarios. Current model-based 3-D training systems lack the subtle nuances of body language which require photorealism to be most effective. Crowd synthesis for games is another domain in which photo-realistic controllable animations could prove useful.

While both motion capture and video data can capture aspects of realistic human movement, current techniques for manipulating this data fall short of the goal of creating photorealistic controllable humans. Motion capture data encodes the realistic dynamics of human movement and can be used to synthesize realistic animations, but the task of endowing the resulting 3-D characters with a photo-realistic appearance is still quite challenging. Likewise, video data implicitly captures the complex relationships between movement, lighting, and appearance, but existing techniques for synthesizing novel video sequences from captured video have not been successfully applied to complex human movement. Techniques based on stereo or multi-view geometry have been used to provide interactive camera control during the replay of captured movement sequences [Zitnick et al. 2004; de Aguiar et al. 2008], but these techniques do not address the problem of synthesizing novel movement sequences from a corpus of examples.

A promising approach to generating photo-realistic video output is

to rearrange the frames in a captured video sequence, thereby providing a limited ability to generate controlled animations [Schödl et al. 2000; Schödl and Essa 2002; Agarwala et al. 2005; Celly and Zordan 2004]. Frame rearrangement depends upon the ability to select good transition points and interpolate video frames across these transitions without introducing visual artifacts. Unfortunately, standard image-based similarity metrics, such as squared pixel differences, fail to choose good transition points when applied to video clips of human motion. These metrics do not capture the complex changes in appearance that are characteristic of humans in motion, especially with related articulations and non-rigidity. Furthermore, secondary motions like hair tousling and the crumpling of clothing add to the complexity of measurement of similarity.

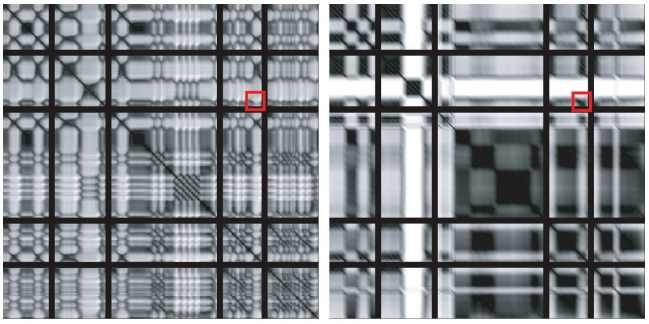


Figure 2: ***(A)*** *Transition cost matrix computed using figure silhouette alone.* ***(B)*** *Cost matrix computed from a traditional motion graph distance metric. Black bars indicate clip boundaries in time. Note how many local minima are evident in (A) in comparison to (B). Also, note that there are low cost regions in matrix (B) that correspond to relatively high costs in matrix (A) (e.g. red highlighted region).*

A plausible human video texture should only contain transition images generated from similar pose sequences. However, robust tracking of limbs and loose clothing while accounting for self-occlusions remains an open problem. As an improvement on the SSD metric used in video textures, we computed a similarity matrix based on the figure silhouette (Fig. 2A). Each matrix entry is the sum of matte pixels following rigid registration (using silhouette centroid and height) and XOR intersection, aggregated over 1.5 seconds of video. Figure 2 illustrates the differences between a silhouette-based and motion capture-based cost matrix (which represents ground truth). Our silhouette-based similarity matrix is clearly inadequate for identifying plausible transitions for human video textures. There are many more local minima in Figure 2A than 2B and a local minima in one matrix does not always correspond to a local minima in the other.

To further complicate issues, the highly structured nature of human movement and our sensitivity in observing them makes small misalignments during transitions immediately visible (e.g. the ghosting artifacts at silhouette boundaries if the limbs are misregistered) as illustrated in Figure 3.

In this paper, we address the challenge of synthesizing photorealistic human motion by leveraging the complementary characteristics of motion capture data and video. We use motion capture data that is synchronized to our video source to identify candidate transition points in video clips. By leveraging accurate positional information from markers, these metrics succeed where standard video-based ones fail. Once the transitions have been identified, a video-based motion graph (*video graph*) is then constructed by registering, segmenting and blending source video clips to compute transition clips. By utilizing the image-plane projections of motion capture markers as ground control points, we can accurately separate occluding body parts into separate layers and composite layers across transitions in a seamless manner. Finally, animation frames are sequentially generated from a traversal of the video graph by concatenating source and pre-computed transition clips.

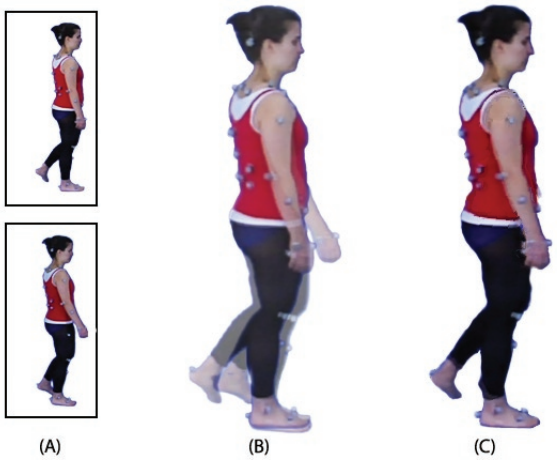


Figure 3: ***(A)*** *One of the challenging goals of transition synthesis is to interpolate two frames from incoming and outgoing video clips.* ***(B)*** *Transitions face ghosting artifacts on both the exterior and interior silhouette regions when cross-fading following rigid registration of the figures.* ***(C)*** *Our method addresses ghosting outside the silhouette using iterative silhouette deformation and interior ghosting using a novel approach to layered segmentation.*

The primary contributions of this paper are:

- A method for simultaneously capturing marker and video data of human movement and constructing a video graph, which enables the creation of *human video textures.*
- A *technique for synthesizing seamless transitions for human video textures.* This technique uses marker reprojections to control a moving least-squares image warp, and
- A *novel graph cut algorithm for the segmentation of human motion into layers* which exploits marker flow and a superpixel representation of the image data.

To demonstrate the viability of our approach at synthesizing choreographed human movement videos, we present several sets of results (Section 5) including: (1) gait motions, to test our system in its detection of transitions and rendering motion on the most basic of human movements, and one for which we are most sensitive to irregularities and (2) an extreme martial arts expert with significant self-occlusion, motion in the depth direction and secondary motions of his clothes.

2 Related Work

There are three categories of previous work that are related to our goal of video-based synthesis of human movement: (a) novel view synthesis for captured motions, (b) video texture-based human motion synthesis and (c) video synthesis from mocap data.

A number of works capture high-quality representations of human movement, which enable the replay of captured content from novel 3-D viewpoints. Zitnick et al. [2004] use a segmentation-based stereo algorithm to generate a two-layer representation of video frames and compute mattes near depth discontinuities. They

demonstrate the ability to synthesize high quality in-between views. Another common approach to the capture and reconstruction of human motion in 3-D is based on the multi-viewpoint construction of the visual hull [Carranza et al. 2003; Sand et al. 2003]. While all of these works support the free viewpoint replay of captured human motion in 3-D, they do not address the problem of manipulating captured content to resynthesize completely new sequences.

Our approach is an extension of video textures [Schödl et al. 2000]. In particular, we relate to the method of video sprites [Schödl and Essa 2002], an extension of video textures which supports flexible control of transitions by the user, resulting in controllable animations. The frame-level similarity measures used in standard video textures limits them to relatively simple nonrigid or stochastic motions such as waving flags or walking hamsters. As we have illustrated in Figure 2, these methods cannot be applied directly to human video content.

In [Celly and Zordan 2004], Zordan et. al. describe an extension of video textures for a carefully-chosen subset of human movements. Their approach employs the ratio of width to height of human silhouette bounding boxes as a feature for identifying transitions. This approach can easily identify motions that generate similar ratios, such as a kick from the right leg versus left leg. For a sufficiently large database of movements, however, this metric will not be sufficiently selective and will require extensive manual intervention. Furthermore, their synthesis technique is based on morphed transitions [Beier and Neely 1992] using the bounding box information. In contrast, our method exploits motion capture data and performs layered motion segmentation to more robustly identify transitions and overcome the ghosting artifacts that result from morphing.

Our use of motion capture data draws from a substantial body of work on data-driven synthesis of 3-D animations of human characters [Kovar et al. 2002]. In this paper, we propose a representation of captured video and mocap content which we call the *video graph*. This data representation allows us to jointly leverage mocap and video data to synthesize novel human motions with articulations, and nonrigid variations due to the secondary motion of clothes and hair.

There are two additional works on image-based animation of human movement which are related to our approach. A technique described in [Cobzas et al. 2002] addresses the problem of capturing and animating the fine-scale clothing deformations of a moving arm. In contrast, we address the re-synthesis of the entire figure. The method of [Hornung et al. 2007] animates a single image of an articulated creature from motion capture data using manually specified correspondences between image features and 3-D motion features. Like our approach, their method handles limb occlusions using a layered representation of the image and hole-filling using in-painting techniques. However, the crucial step of fitting an initial layered mesh to the image is done with manual intervention. Furthermore, their method does not handle video content.

The related work which is perhaps the most similar to ours is the method of Starck et al. [2007]. This approach combines reconstruction using a visual hull method with the re-synthesis of human motions. They employ a spherical matching algorithm based on the 3-D mesh of the reconstructed figure to identify good transitions. This approach enables them to construct a type of motion graph using video information alone. However, their technique is limited to genus zero surfaces and relies on accurate 3-D reconstruction. As a consequence, it tends to smooth geometric features (such nose, ears, hair, and clothing) due to reconstruction error and limited geometric resolution. In contrast, our goal is to manipulate source pixels directly with minimal loss of fidelity. This is necessary to preserve crucial details such as loose hair and the folds in clothing. By supplementing video with motion capture data, we obtain the ability to easily identify transitions and synthesize non-genus zero poses.

3 Identifying Video Graph Transitions

The original motion graphs paper [Kovar et al. 2002] computed transition candidates as a function of clouds of points over a window of frames in time. Likewise, our method computes motion similarity from the markers directly, instead of from a fitted skeleton, by computing the L2 distance of their positions and trajectories from frame to frame over a fixed window of 0.25 seconds worth of 3-D marker positions. We use a fixed threshold on this distance metric and hand-select this metric to balance the quality of the transitions with the number of candidates.

Since we are using a single video camera in our capture setup, there is only one viewing direction into the scene for which we have video data. As a consequence, the construction of the video graph must be constrained to preserve the continuity of motion with respect to this viewing direction. This has two consequences for the detection of transitions in building the video graph. First, in computing the cost of a potential transition, we align the two candidate clips by pure translation of the marker points, rather than the rigid body rotation and translation which is used in a standard motion graph. This is because we cannot rotate the captured figure in 3-D without extrapolating the effect of that rotation to our video, a process which is unlikely to produce the realism that we desire.

The second difference is that we prune the set of initial 3-D-based transitions by examining only the 2-D projections of the marker data into the video camera plane. Following camera calibration (see Appendix for details), we match clips when they are compatible with respect to the projected positions and velocities (marker flow) of their marker sets in 2-D, not in 3-D. More specifically, we prune transition candidates if any corresponding marker flows form an angle greater than a threshold (which we set to 90 degrees).

In our Kung Fu dataset, we picked an initial motion graph threshold that generated approximately 100 transition candidates. The following transition pruning step left about 20 transitions in the Kung Fu video graph and we use these in the paper and video results.

4 Generating Transitions for Video Graphs

A human video texture is generated by traversing the video graph, replaying the stored clips and previously synthesized transition clips in an analogous manner to a motion graph[1]. Synthesis of natural-looking transitions is the key technical challenge. Any misregistration of the limbs or torso in transitioning between clips will result in highly visible artifacts such as ghosting and popping. In order to generate believable transitions, we must compute the correspondence between the figure in a pair of clips, warp the video frames so as to bring the images of the figures into alignment, and blend the two clips together in the transition region. We will show that these video-manipulation tasks can be simplified considerably through the use of marker data.

Transitions are generated over a 15 frame window, as illustrated in Figure 4. The set of synthesized frames are organized into three groups of five frames each, called pre-transition, transition, and post-transition. In the transition region, pixels from both clips are combined to produce the final output. We found that the addition of pre- and post-transition warping resulted in smoother transitions. In these regions there is no blending, and the final output pixels are

[1]This approach is well-suited for real-time rendering because all frames are computed and stored before run-time.

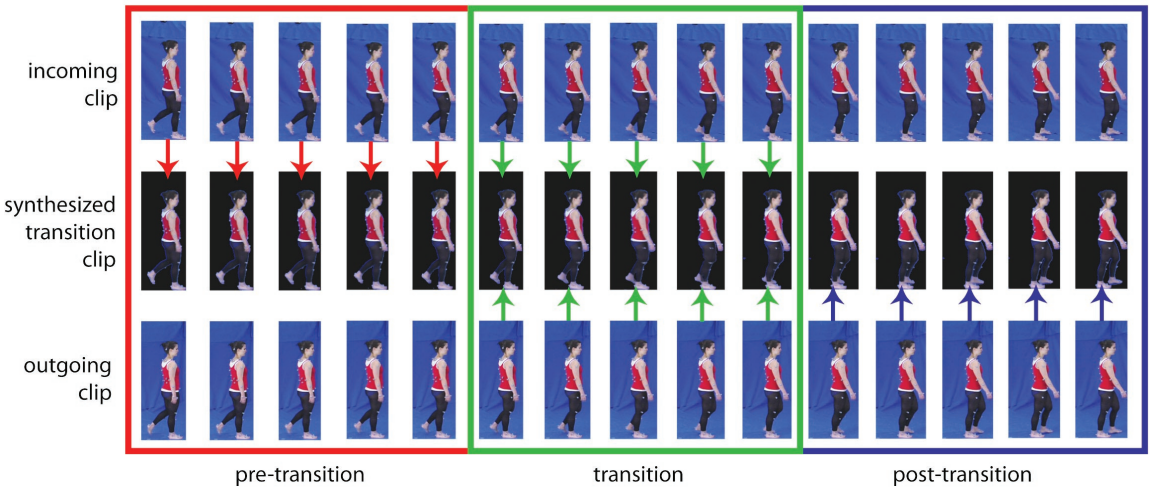


Figure 4: ***Transition Synthesis:*** *Transition synthesis involves interpolating an incoming clip (top row) with a corresponding outgoing clip captured at a different point in time (bottom row). By linearly interpolating 2-D marker projections from an incoming (or outgoing) frame, a moving least squares warp may be computed to bring marker projections into alignment in the synthesized result (middle row). At the halfway point (8th column), the interpolant value (ranging from $\frac{1}{16}$ to $\frac{15}{16}$) is 0.5 and the warped frames are 50-50 blended. Note that the arm exhibits self-occluding motion – we segment limbs into layers for separate warping and blending from the background layer. Frames in the pre- and post-transition regions (red and blue boxes) are warped and added to the output without blending. This pre- and post-warping reduces warping distortion in the transition region (green box). Arrows denote frames that contribute pixels to the synthesized transition clip.*

taken from a single clip (the incoming clip for the pre-transition region and the outgoing clip for post-transition). However, by warping the pre- and post-transition frames towards their corresponding frames we reduce the amount of warp that is needed in the transition region, which in turn reduces visible artifacts. The total warp that aligns a pair of frames can be broken down into two component warps, one for each frame. These component warps are depicted by arrows in Figure 4.

The key step in the synthesis of a transition is the computation of a pair of warps from a pair of frames. This process can be divided into two conceptual steps: *registration* and *layer segmentation*. The registration step identifies correspondences between the figure regions in both frames and computes the pair of warps, one from each direction. For frames in the transition region, where the magnitude of the warp is the largest and pixels from both clips are blended together, it is important to take into account the fact that different parts of the body (e.g. the arms and torso) may be undergoing self-occluding motions. For example, in the case where the right arm moves across the torso, a single smooth warp for the entire frame will not suffice (the warp would need to "tear" at the occlusion boundary). Column 8 in Figure 4 illustrates such a case: the right hand is outside the leg in the incoming clip and partly inside the leg in the outgoing clip. We address this challenge by automatically segmenting the body parts into layers. In the given example, we segment the right arm pixels from the rest of the body. The arm then comprises a foreground layer while the remaining figure pixels constitute a background layer. We can then compute separate smooth warps for the corresponding segmented layers in a pair of frames and composite the warped layers together in generating the final output.

The complete image synthesis process for a pair of transition frames consists of the steps illustrated in Figure 5. In the first step, rigid translation and scaling of the two video clips produces a coarse alignment between pairs of corresponding frames. Then for each frame, we compute a nonrigid warp using moving least squares (MLS) [Schaefer et al. 2006]. The MLS warp uses a set of control points which consist of reprojected markers and a set of point samples from the silhouette boundary. Thus the registration step aligns the silhouette boundaries between the frames. The third step performs a layer-based decomposition of the input video so that moving limbs can be segmented out and warped separately from the background layer. In the final step, the output clips are blended and composited, and any remaining holes are filled by inpainting, to produce the output clip. We now describe each of these steps in more detail.

4.1 Iterative Silhouette Deformation

In order to produce an accurate alignment of figure regions from two frames, we proceed in two stages. In the first stage, we begin by identifying the point correspondences corresponding to the projected positions of background layer markers which are visible in both frames. We do an initial MLS warp with these marker projections as control points. Because we know which body part each marker is attached to, we can trivially assign each visible marker to its corresponding layer.

The initial warp brings the background layer body shapes into approximate alignment, but it is insufficient for full body warping. This is because such control points do not provide explicit information about the boundary. If the boundary is not taken into account, ghosting invariably occurs in the interpolated frames. Therefore, in the second stage we perform additional warping of the two body shapes in order to bring their silhouette boundary curves into alignment.

Our alignment method is essentially an application of the Iterated Closest Point algorithm [Besl and McKay 1992]. It has the advantages of being simple to implement and very effective for curves which are already in close proximity to each other. This will always be the case in our application due to the reliability of the initial warp using the projected markers. We apply additional silhouette alignment constraints to the MLS warp by sampling control points from the silhouette curves and incrementally updating the MLS solution. We can measure the degree of overlap between body shapes by computing the Sum of Absolute Differences (SAD) between the warped mattes in the two frames. This measure counts the number

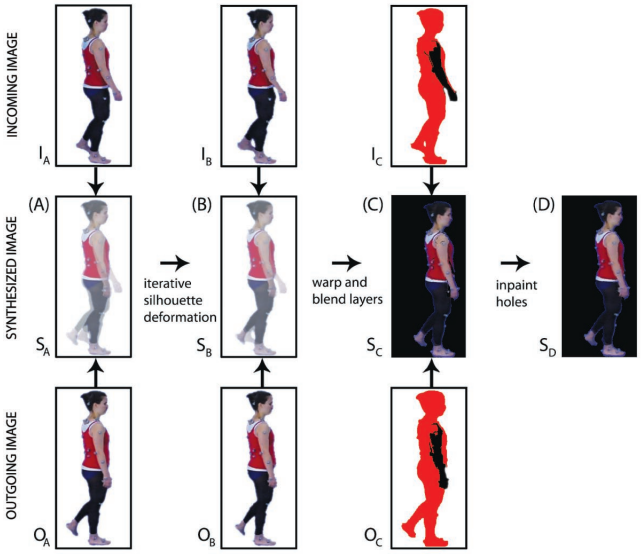


Figure 5: ***Transition Image Synthesis Pipeline:*** *The following steps generate output S_D from incoming image I_A and outgoing image O_A: (A) I_A and O_A are rigidly registered to align root marker projections and silhouette height, producing S_A. (B) Iterative silhouette deformation is applied to I_A and O_A, producing I_B and O_B - note the reduced ghosting behind the legs in S_B from S_A. (C) I_A and O_A are segmented into motion (limb) layers which are warped, blended and composited onto the background layer in back to front order, producing S_C. (D) Finally, image in-painting is applied to S_C to fill holes between composited layers, resulting in S_D.*

of pixels that lie outside the overlap region. We identify point correspondences between contours that minimize the SAD error measure and add them to the control point set for the MLS estimate. After reaching convergence, the intersection of the registered mattes is computed and used for final composition to eliminate remaining exterior ghosting artifacts (Fig. 5 shows the registered result before matte intersection to illustrate how ghosting outside the silhouette is reduced before it is eliminated with intersection).

4.2 Layered Motion Segmentation

A key step in the accurate treatment of self-occluding motion (e.g., when an arm moves in front of the torso) is to decompose the figure region into layers that can be warped separately. In general, the problem of automatically decomposing an arbitrary video sequence into an appropriate set of layers is quite challenging and has received considerable attention. In our application, we can leverage the context and motion estimates obtained from our reprojected marker set to solve for the layer segmentations using a novel application of MRF labeling via graph cuts.

The first step in our process is to segment the figure pixels in each frame, resulting in a set of pixel regions known as superpixels. We use the method of [Felzenszwalb and Huttenlocher 2004] to generate the superpixels. We then generate a Markov Random Field (MRF) [Li 1995] model for the segmentation problem, where the observation nodes are superpixels and the hidden labels are either foreground or background. After performing MRF inference via graph cuts (we use the method of [Boykov and Kolmogorov 2004]), we assign the label for each superpixel to all of the image pixels that it contains. The superpixel approach reduces the number of nodes that are required in the MRF, thereby reducing the memory and computation requirements.

Our assignment of edge costs in the graph is illustrated in Figure 6 and specified in Equation 1 below. We leverage the initial warps obtained from the limb-labeled marker projections to separate the foreground and background layers effectively. In this example there is one foreground layer corresponding to the arm and one background layer. Assume we are warping from outgoing frame B to incoming frame A. We first compute two separate warps of image B based upon the foreground and background visible marker sets. The results of applying these warps to the image B are illustrated in the middle column of Figure 6. The cost of associating a superpixel in image A with the background (torso) label is computed by comparing the superpixel color with the corresponding region in the background-warped output (Fig. 6(3)). The cost for a foreground layer assignment is similarly computed using the foreground warped output (Fig. 6(2)).

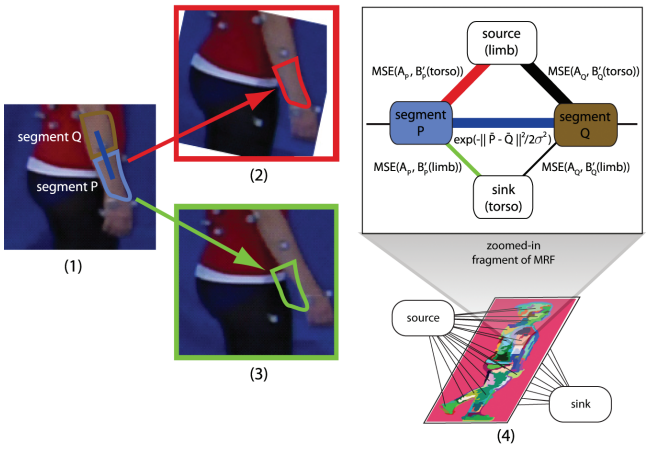


Figure 6: ***Graph cut based motion layer segmentation:*** *In this example, image A represents the image to segment and image B represents the corresponding image in the transition pair.* ***(1)*** *Image A with cyan superpixel P to be labeled as foreground (limb) or background (torso).* ***(2)*** *Image B is warped towards image A using foreground (limb) markers, producing $B^{'}(limb)$,* ***(3)*** *Image B is warped toward image A using background (torso) markers, producing $B^{'}(torso)$.* ***(4)*** *Fragment of MRF model showing the organization of the cost terms, where thick lines denote high capacity. Following graph cut, superpixels connected to the source (sink) are labeled as foreground (background).*

More specifically, we minimize the following energy function over all candidate binary labelings L of image A:

$$E(L) = \sum_{p \in A} \left(\frac{1}{|\,p\,|} \sum_{i \in p} \|A_i - B_i^{'}(L_p)\|^2 \right) + \beta \sum_{pq \in N} T\left(L_p \neq L_q\right) \cdot \exp\left(-\|\overline{A_p} - \overline{A_q}\|^2 / 2\sigma^2\right) \quad (1)$$

where p is a superpixel, A_i is the RGB color of pixel i in image A and $B_i^{'}(L_p)$ is the color of the corresponding pixel in image B under the MLS warp specified by label L_p. In the third summation term, which penalizes discontinuities between neighboring superpixels $pq \in N$ with similar color, function $T\left(L_p \neq L_q\right)$ is 1 if the condition inside parenthesis is true and 0 otherwise. Also in the third summation, β is a design parameter (which we set to 30 in our experiments), $\overline{A_p}$ and $\overline{A_q}$ are the mean colors of superpixels p and q respectively and σ is a noise parameter.

The intuition behind this formulation is that superpixels that belong to a particular layer in one frame of a pair will be well-matched to

the pixels in the corresponding frame, once the layer-specific warping has been performed. Note that if the relative motion between foreground and background is very small, or if the foreground and background colors are very similar, it may not be possible to segment the layers accurately. Fortunately, in that case an accurate segmentation is not needed since ghosting artifacts will only be visible if there is a color mismatch. Otherwise, when the wrong warp is applied it is quite unlikely that the superpixel will find significant support in the image pixel values. In general we deal with multiple possible foreground layers by doing a series of binary segmentations, one for each possible foreground. In each case we used the limb-labeled marker sets to compute the appropriate warps. Following graph cut segmentation for each layer, we enforce segmentation consistency between image A (B) and B (A) by eliminating foreground-labeled superpixels in image A (B) that do not overlap by a threshold amount (0.3) in area with corresponding foreground-labeled superpixels in image B' (A').

4.3 Rendering the Transition Frames

Given a set of segmented and warped layers for each frame in the transition region from each video clip, we perform compositing and hole-filling to render the final video sequence. First we add the pre- and post-transition frames to the output sequence by applying a global warp to each frame. Then we composite the segmented layers from each pair of frames in the transition region. We use the painters algorithm and composite the layers from back to front (i.e. background layer followed by foreground). There will be a separate foreground layer for each MRF segmentation. We use the average distance from the camera of the mocap markers in each layer to determine their order.

A significant problem in compositing the foreground layers is the presence of holes due to missing background pixels. For example, in order to align the right arm across a pair of images, we may need to shift the arm up slightly in one frame and down in the other. These shifts will create holes at the trailing edge of the warp because we are uncovering background layer pixels which are not present in the source imagery. We address this problem in two ways. First, we differentiate between overlapping and nonoverlapping pixels in compositing the layers. Overlapping pixels belong to the intersection of the corresponding foreground layers from both frames (incoming and outgoing), while nonoverlapping pixels belong to one layer and not the other. We crossfade overlapping pixels in order to generate a smooth transition in appearance. We then composite the nonoverlapping pixels directly into the output buffer. This allows us to use all of our foreground layer pixels and minimizes the number of background holes that must be filled.[2] Finally, any remaining holes in the background layer are filled automatically using in-painting [Efros and Leung 1999].

5 Results

In this section, we present results from applying our method to three different datasets resulting from three capture sessions with two different subjects. For each dataset, we constructed the video graph and generated synthetic video sequences. The full sequences are available in the video accompanying this paper. In this section we highlight specific transitions from these sequences in order to illustrate some of the results of our method.

The first result consists of backflips performed by an acrobatic martial artist. The second result features the same martial artist performing a range of Kung Fu fighting moves. The third result exhibits female gait motion.

[2]This strategy does have the property of making the foreground layers "fatter," but it does not seem to be a significant source of artifacts in practice.

5.1 Martial Arts Demonstration

Figure 7 shows three example transition composites from a martial arts expert captured wearing two separate costumes: (a) flowing black shorts and (b) baggy red pants. Our system identified a transition during punches toward the camera from a set of captured backflips. This is an example of a non-trivial transition that would be difficult to identify by hand. Despite significant self-occluding motion from his left arm, the transition result displayed Figure 7(C) shows his hand in focus. One side effect of our method leaves neighboring pixels to a moving layer blurry, which is caused by a combination of hole removal and non-overlapping regions of the layer.

5.2 Gait Motion

We also captured a woman performing simple walks which serve as tough examples since people are used to observing gait. Normal gait also provides clear sources of self-occlusion as people swing their arms. As evidenced in Fig. 7(B) and 7(E), our layered segmentation method significantly reduces texture mismatch caused by the arms moving beside the body.

We have chosen to leave the visible markers in all but one of the results we present in this paper. This facilitates the comparison between frames and ensures that any visible artifacts are the result of our core algorithm. In practice, marker removal would be performed on each frame to generate the final output and we show an example of this manual correction in Fight Sequence 2 in the included video. This is a standard operation in many production scenarios [Borshukov et al. 2007] and commercial software is available to facilitate it.

6 Limitations and Discussion

The key challenge in synthesizing photo-realistic human motion from video is the need to establish correspondences between regions of pixels across the video sequence. These correspondences are needed to (a) estimate the extent of motion between frames (to identify potential transitions) and (b) to segment, warp, and align images between frames to create seamless transitions. Standard computer vision methods for video analysis are unable to reliably identify the correct correspondences with sufficient accuracy to support our application.

Our solution to the correspondence problem is to leverage motion capture technology to obtain accurate and reliable correspondence information in the form of marker data. While we believe this approach has enabled us to obtain synthesis results which would be difficult to achieve through any other method, it is worthwhile to discuss the practical limitations of our current solution. One set of issues center around the capture environment, which must be tailored to the conflicting needs of video- and motion-capture. Costume design and marker placement must be carefully thought-out. Clothing was tightened and adjusted appropriately to ensure marker visibility and stability during the sessions. The motion of the actors themselves was not choreographed in a precise manner, but they were certainly aware of the need to conform to the motion style of a side-scrolling video game.

Because we currently employ a single video camera, we are sensitive to parallax effects (parallax can be observed, for example, in the relative motion of the shoulders with respect to the center of the chest). Fortunately there has been significant progress in multi-camera technology for 3-D capture of human motion, and we believe that we can leverage these results to extend our system's capabilities to 3-D. This would also enable 3-D visualizations of newly synthesized content and open up additional application domains.

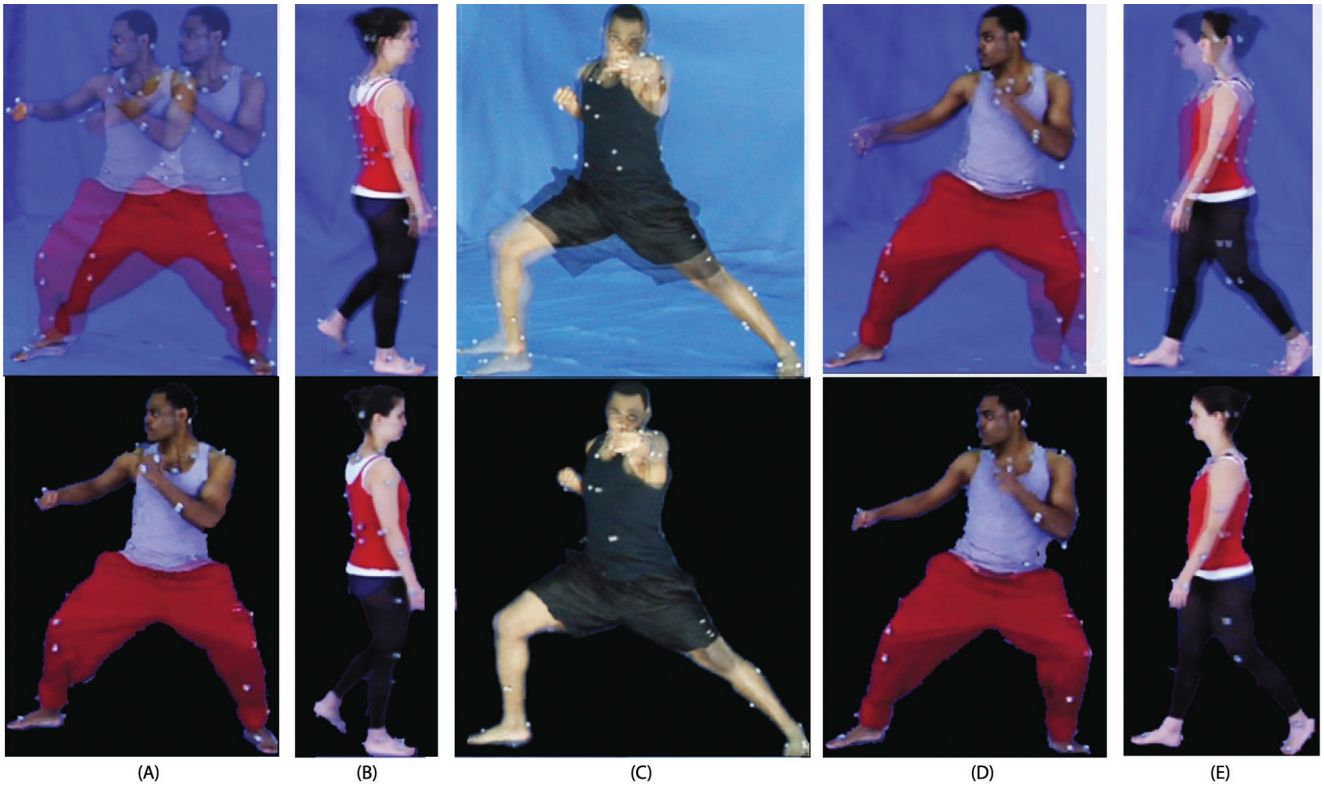


Figure 7: ***Transition Composites:*** *Images in the top row show transition frames cross-faded after rigid registration. Note the ghosting artifacts inside and outside the figures' overlapping silhouettes. The bottom row shows the result of applying our transition synthesis method.*

Another issue that we plan to address in future work is relighting the synthesized content to reduce changes of intensity across transitions, which is evident in the results, and animation in new environments.

7 Conclusion and Future Work

In conclusion, we have presented a method for creating controllable photorealistic animations of human movement. By capturing video and motion capture data in tandem, we have demonstrated that video clips of similar pose sequences from different points in time may be identified from 3D and 2D projected marker trajectories. We have shown how to render synthetic video clips for transitioning video across gaps in time via novel registration and segmentation algorithms.

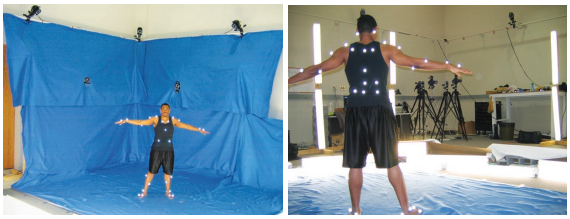
We found that transition video clips may be easily noticed in the presence of ghosting artifacts and discontinuities in motion. Our system's exploitation of marker data in addition to captured pixels was vital for the elimination of these artifacts. While our method is capable of generating transition clips which are difficult to detect during video playback, careful examination of still transition frames reveals subtle artifacts such as blurriness and non-smooth layer boundaries that expose their synthetic nature.

Also, as in the case of motion graphs, we identified some transitions that were surprising in the sense that they could not have been easily predicted by a capture session director. For example, the two transitions composed using transition frames shown in Fig. 7(A) and 7(C) would have been difficult to identify by hand. Therefore, the transition discovery capabilities of motion graphs carry over to the video domain in our computation of video graphs.

A number of interesting extensions to our work are possible. First, our layered motion segmentation could be improved by adding a second video camera and incorporating additional stereo cost terms in the MRF. This would introduce depth discontinuity information in finding limb boundaries in addition to the information provided by oversegmentation (in a similar fashion to [Zitnick et al. 2004]). Second, our method faces the challenges of trading transition quality for motion responsiveness (graph connectivity) which is common to all motion graph-like interactive animation systems. By expanding motion and video data via interpolation of subsequences between similar foot placement events [Zhao and Safonova 2008], more transitions may be introduced to improve character responsiveness to interaction. Finally, we are excited about the possibilities for re-animating videos of humans using separate motion capture of alternate characters. The technique of [Hornung et al. 2007] for animating a still picture using motion capture data could be extended to video using the methods presented in this paper.

8 Appendix: Data Capture and Calibration

The hardware basis of our setup is a commodity motion capture system and video camera. Our motion capture setup is a 12 camera Vicon system capturing 3-D marker trajectories at 120Hz. Video was recorded using a single Panasonic HVX200 camera capturing 1280 x 720 images at 60Hz. In order to synchronize motion capture video signals in software following capture, actors were asked to clap twice in a characteristic manner before and after each performance. These events were used to solve for a simple scale and translation transformation. Because the motion capture frames were captured at twice the sampling rate of the video, motion capture data was accurately resampled in time for each frame of video.



(a) (b)

Figure 8: *(a) Capture volume with blue screen and mocap cameras. (b) Fluorescent lighting and co-located mocap and video cameras. Note the lack of a standard mocap suit on the subject, and the use of mocap markers attached to clothing as well as skin.*

For calibration purposes, a custom calibration target was designed to strobe infrared and visible green light in lockstep with the motion capture cameras. The calibration target is a lit ping pong ball mounted at the tip of a wand. In the video camera images, a green LED inside the ball causes it to appear as a bright green spot against a black background. It can be automatically segmented from the video frames using a simple thresholding operation in HSV color space. The ball also contains an IR LED and appears as a single large marker which can be localized in 3-D using the standard Vicon video processing pipeline. The corresponding 2-D projections of each 3-D ball position are obtained through color space analysis of the video frames. Using this set of normalized 2-D to 3-D correspondences, the camera projection matrix is resectioned using the Gold Standard Algorithm 6.1 described in [Hartley and Zisserman 2004].

Another fundamental issue is to determine, in each frame captured by the video camera, which markers are actually visible. The standard Vicon software can compute the projections of each marker along the camera viewing axis, but since it sees the markers from multiple views it cannot guess the visibility of a particular marker with respect to an arbitrarily-chosen viewing direction. We implement this visibility test in hardware by positioning an extra Vicon camera next to our video camera, in attempt to colocate their viewing axes.

Acknowledgements

This material is based upon work supported in part by the National Science Foundation under NSF Grant IIS-0205507 and also in part by the Strategic Information and Communications R&D Promotion Programme (SCOPE), Japan.

References

AGARWALA, A., ZHENG, K. C., PAL, C., AGRAWALA, M., COHEN, M., CURLESS, B., SALESIN, D., AND SZELISKI, R. 2005. Panoramic video textures. *ACM Transactions on Graphics 24*, 3, 821–827. 2

BEIER, T., AND NEELY, S. 1992. Feature-based image metamorphosis. *In Computer Graphics (Proceedings of ACM SIGGRAPH 92) 26*, 2, 35–42. 3

BESL, P. J., AND MCKAY, N. D. 1992. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 2, 239–256. 4

BORSHUKOV, G., HABLE, J., AND MONTGOMERY, J. 2007. *Playable Universal Capture chapter in GPU Gems 3*. Addison Wesley. 6

BOYKOV, Y., AND KOLMOGOROV, V. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on Pattern Analysis and Machine Intelligence 26*, 9, 1124–1137. 5

CARRANZA, J., THEOBALT, C., MAGNOR, M. A., AND SEIDEL, H.-P. 2003. Free-viewpoint video of human actors. *ACM Transactions on Graphics 22*, 3, 569–577. 3

CELLY, B., AND ZORDAN, V. 2004. Animated people textures. In *17th International Conference on Computer Animation and Social Agents*. 2, 3

COBZAS, D., YEREX, K., AND JGERSAND, M. 2002. Dynamic textures for image-based rendering of fine-scale 3d structure and animation of non-rigid motion. In *Eurographics*, 1067–7055. 3

DE AGUIAR, E., STOLL1, C., THEOBALT, C., AHMED1, N., SEIDEL, H., AND THRUN, S. 2008. Performance capture from sparse multi-view video. *ACM Transactions on Graphics 27*, 3, 4:1–4:10. 1

EFROS, A. A., AND LEUNG, T. K. 1999. Texture synthesis by non-parametric sampling. In *ICCV (2)*, 1033–1038. 6

FELZENSZWALB, P. F., AND HUTTENLOCHER, D. P. 2004. Efficient graph-based image segmentation. *Int. J. Comput. Vision 59*, 2, 167–181. 5

HARTLEY, R. I., AND ZISSERMAN, A. 2004. *Multiple View Geometry in Computer Vision*, second ed. Cambridge University Press. 8

HORNUNG, A., DEKKERS, E., AND KOBBELT, L. 2007. Character animation from 2d pictures and 3d motion data. *ACM Transactions on Graphics 26*, 1. 3, 7

KOVAR, L., GLEICHER, M., AND PIGHIN, F. 2002. Motion graphs. *ACM Transactions on Graphics 21*, 3, 473–482. 3

LI, S. Z. 1995. *Markov random field modeling in computer vision*. Springer-Verlag, London, UK. 5

SAND, P., MCMILLAN, L., AND POPOVIC, J. 2003. Continuous capture of skin deformation. In *ACM Transactions on Graphics*, vol. 22, 578–586. 3

SCHAEFER, S., MCPHAIL, T., AND WARREN, J. 2006. Image deformation using moving least squares. *ACM Transactions on Graphics 25*, 3, 533–540. 4

SCHÖDL, A., AND ESSA, I. A. 2002. Controlled animation of video sprites. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA)*, 121–127. 2, 3

SCHÖDL, A., SZELISKI, R., SALESIN, D. H., AND ESSA, I. 2000. Video textures. In *Proceedings of ACM SIGGRAPH 2000*, ACM Press / ACM SIGGRAPH, 489–498. 2, 3

STARCK, J., AND HILTON, A. 2007. Surface capture for performance-based animation. *IEEE Comput. Graph. Appl. 27*, 3, 21–31. 3

ZHAO, L., AND SAFONOVA, A. 2008. Achieving good connectivity in motion graphs. In *Proceedings of the 2008 ACM/Eurographics Symposium on Computer Animation*. 7

ZITNICK, C. L., KANG, S. B., UYTTENDAELE, M., WINDER, S., AND SZELISKI, R. 2004. High-quality video view interpolation using a layered representation. *ACM Transactions on Graphics 23*, 3, 600–608. 1, 2, 7