High-Resolution Video from Series of Still Photographs

Ge Jin and James K. Hahn

Department of Computer Science, The George Washington University,
Washington DC, 20052, USA
{jinge, hahn@gwu.edu}

Abstract. In this paper, we explored the problem of creating a high-resolution video from a series of still photographs. Instead of enhancing the resolution from the video stream, we consider the problem of generating a high-resolution video as an image synthesis problem. Using the continuous shot in the digital camera, we can get a series of still photographs at 2 to 3 frames pre second. The main challenge in our approach is to synthesize the in between frames from two consecutive still images. The image synthesis approach varies based on the scene motion and image characteristics. We have applied optical flow, image segmentation, image filtering and skeleton based image warping techniques to generate high-resolution video.

Keywords: Video Synthesis, Optical Flow, Image Segmentation.

1 Introduction

In spite of continuous development in digital video technology, making a movie-quality video is still beyond the limit of consumer-level digital camcorder. Compared with digital camcorder, the consumer-grade digital camera can capture images at much higher quality and resolution. Currently, more and more digital cameras are capable of taking continuous shot images. Continuous shot images can be considered as an intermediate stage between a single image and a video clip. These continuous images have a high-resolution feature of still image, while preserving the continuous motion cues in the video clip. Fast professional digital and sports camera can take continuous shot images at extremely high frame rates and these images can be directly converted to a video. However, for the consumer level digital camera, the frame rate of the continuous shot images is still limited to 2 to 3 frames per second. In this paper, we will concentrate on the problem of using low frame rate continuous shot images to synthesize high resolution video.

Fully automated video synthesis from any type of continuous images would be the ultimate goal. However, to prove the feasibility of making videos from continuous shot images, we have simplified the problem by limiting the types of scenes and allowing user interaction during the image segmentation stage. In this paper, we have limited our scenes to four different types of motions: static

G. Bebis et al. (Eds.): ISCV 2006, LNCS 4291, pp. 901–910, 2006.
© Springer-Verlag Berlin Heidelberg 2006

environment with moving camera, static camera with moving rigid object, static camera with character walking and static camera with moving trees. A general scene may consist of all these motions simultaneously, but we have tackled these problems separately.

In order to make a high resolution video from continuous shot images, we need to synthesize in between frames from two consecutive images. For the static environment with a moving camera, we used optical flow based image warping to generate in between images. For the rigid moving object, we used image segmentation and image morphing to synthesize in between frames. For the human walking scenes, we combined optical flow based and skeleton based image warping to produce human walking images. Finally, for the moving trees, we applied simple alpha blending followed by contrast enhancement filtering to generate the images of moving trees.

Most of algorithms described in this paper are fairly simple modifications of already known techniques. So, the main contribution of this paper is formulating the problem of high resolution video synthesis from still photographs, and proving the feasibility of our approach by performing experiments on four different types of moving scenes.

The rest of this paper is organized as follows: In section 2, we will look into some related works. The section 3 will provide detailed description of synthesizing in between images from different types of continuous shot images. We will show our experiment and result in section 4 and conclusion and future works in section 5.

2 Related Works

Dense optical flow method generates one to one pixel matching between two images. [1] calculated the optical flow using steepest descendent energy minimization method. Another well known optical flow algorithm is local motion optimization using the KLT method [2][3]. Further, the computation speed is increased by hierarchical searching [4] at different image scales. Recently, [5] describes a video matching method to align different video sources. They used robust estimation of flow field to interpolate and extrapolate at missing areas. We choose the KLT method for our optical flow calculation, for its efficiency and excellent result in quickly moving rigid objects. For our purpose, since the viewpoint or the motion is restricted by the two consecutive images, we do not need to recover the camera position and 3D information. In this context, our work is more related with feature based image morphing [6], skeleton based image warping [7] and view morphing works [8]. Commercial product like Apple Shake is using optical flow for re-timing, however, if the motion between two consecutive frames is large, the synthesized result presents motion-blur effect.

Image-based rendering focuses on generating new images and videos from given images. Some works used single image to generate a new image or an animated video. Tour into the picture [9] used a vanishing point to separate the image into spidery mesh regions and warped the image regions based on

the camera movement. QuickTime VR [10] used panoramic image to generate camera panning and zooming in static environment. Stochastic motion texture is used to animate the movement of passive elements driven by the wind [11]. These works have produced convincing results. However, since there is only one image, the changes in view point or the animated motion is still limited to some small scale. We consider our work as a natural extension of video synthesis from a single image. Stop motion animation with image based motion blur is proposed to synthesize in between frames, where the image segmentation and optical flow techniques are used to synthesize motion blur effect [12]. Our proposed method differs from [12]'s work, where we are focused on synthesizing clear trajectory of the motion path. Motion magnification [13] is proposed to magnify the small motions in image sequence, however, for our case the motion in two consecutive images are relatively large and as a result, it is difficult to apply their method for our case.

In the image segmentation, graph cut based methods have produced good results both in still images and videos [14][15][16]. To extract clear alpha matte, Bayesian matting [17] and Poisson matting [18] approaches have been introduced to segment hairy objects. Another simple and efficient segmentation method is based on cellular automata [19]. For the video synthesis, the segmented image usually needs user refinement to remove the flickering artifact. So, for our case, we used the cellular automata based method to perform the initial segmentation and refine the result with user interaction.

3 Image Synthesis Using Continuous Shot Images

In order to synthesize a new image from two given images, we matched two consecutive images by calculating the optical flow and used the flow field to warp the source image to the destination image. For the static environment with moving camera, the optical flow with pseudo inverse image warping produces convincing results. However, for the scene with moving objects, only using optical flow could not generate good results. Some artifacts will occur at the boundaries of moving objects. So, we used the cellular automata based method to segment out the foreground layers and further refined it by user interaction. By using the optical flow at the foreground object layer, we morphed the image of a moving object to the in between frames. For the articulated figure such as walking human, image matching could not establish correct pixel mapping between walking legs. We have separated the human body into upper body and lower legs. For the upper body, we consider it as a rigid object. For the legs, we calculated the skeleton structure at each image, and used the skeleton based image warping to warp the legs. For the moving trees, we have applied simple alpha blending followed by contrast enhancement filtering to generate the in between frames.

3.1 Optical Flow Based Image Warping

Synthesis of in between frames for the camera moving at a static environment is image matching and warping problem. If the camera is panning at a fixed

viewpoint, the problem becomes a panoramic image stitching and projective warping problem. For our case, we experimented with the camera moving along a certain path. Since we are only using one camera and not calculating the camera internal parameters and positions, the image matching between two adjacent frames is purely image based. We are dealing with the images that taken at a short time interval (about 0.5 second), the optical flow between two consecutive images will not be too large. Suppose $I(x, y)$ and $J(x, y)$ are two adjacent images. The optical flow vector $d = [d_x, d_y]$ maps the image point $u = [u_x, u_y]$ in image I to the image point $v = [v_x, v_y]$ in image J.

$$J(v_x, v_y) = I(u_x + d_x, u_y + d_y) \tag{1}$$

The optical flow vector d is calculated by minimizing the residual error at $u(u_x, u_y)$ with a certain window size.

$$\epsilon(d) = \epsilon(d_x, d_y) = \sum_{x=u_x-w_x}^{x=u_x+w_x} \sum_{y=u_y-w_y}^{y=u_y+w_y} (I(x, y) - J(x + d_x, y + d_y))^2 \tag{2}$$

The Pyramid Lucas Kanade optical flow algorithm calculates the flow vector at small image scale and propagates it into the next level as an initial guess. The error minimization is done by a gradient descendent method. In this paper, we did not propose a new optical flow algorithm. Instead, we used the pyramid LK optical flow library in OpenCV [20] to calculate the optical flow between two images.

Once we get the optical flow between two images, we can use the flow vector to warp one image to the other. Mathematically, it is a forward warping from image I to J using optical flow. However, the forward mapping has some problems with aliasing and holes. It can be done with two-pass method, but since the motion flow field is very smooth, we used pseudo inverse warping similar with [11]'s work. We used a linear factor f to smoothly change the flow vector from $[0, 0]$ to $[d_x, d_y]$. Since the frame rate of continuous shot images is about 2-3 frames per second, we need to generate about 10 images between two continuous shot images. So the linear factor increases from 0 to 1 with the interval of 0.1. The pseudo inverse warping function is described in equation (3).

$$I^*(x, y) = I(x - f * d_x, y - f * d_y) \tag{3}$$

The pseudo inverse warping works well on the scenes with a smooth vector field. However, when there is a moving object or the scene topology changes a lot, we could not simply warp the image using optical flow.

3.2 Movement of a Rigid Object

For the static camera with a moving object, we need to segment out the moving object first. To do this, we initially calculated the background image by averaging the full sequence of images. The background image was further refined by using

a large threshold value (Summed RGB Color Difference) to disregard the foreground pixels and only use the remaining pixels to get an averaged background. To get more accurate background, this process is repeated by decreasing the threshold value. For our case, two steps of iteration with large and small threshold value have generated good results.

Fig. 1. Left: Background estimation with averaging, Right: Refined with large and small threshold

In the next step, we used simple RGB color threshold to segment out the foreground object. The result is further refined by eroding and dilating operation to remove background pixels and fill the holes inside foreground object. After this, we set the pixels at segmentation boundary as uncertain pixels.

Fig. 2. Left: Segmentation with uncertain pixels marked in dark gray, Right: Segmentation result with Cellular Automata

Finally, we used the cellular automata based method to segment out the foreground object [19]. Each pixel is assigned with label, power, and its RGB color value. The label has three types: foreground, background and uncertain. The power of foreground and background pixels is set to 1 and the uncertain pixels are set to 0. At each iteration step, the neighbouring pixel with higher power and color similarity will try to change current pixel's label and power.

L_i:Label of Pixel i; P_i:Power of Pixel i(0 to 1)
RGB_i: RGB value of pixel i; $g(RGB_i - RGB_j) = 1 - \frac{\|RGB_i - RGB_j\|}{Max\|RGB\_Diff\|}$

$for\ \forall\ pixel\ j \in Neighbor\ of\ pixel\ i$
$\quad if\ (g(RGB_j - RGB_i) * P_j^t > p_i^t$
$\quad\quad L_i^{t+1} = L_j^t;\qquad\qquad (t:\ iteration\ step)$
$\quad\quad P_i^{t+1} = g(RGB_j - RGB_i) * P_j^t;$
$\quad endif$
$end\ for$

The automatic segmentation result with cellular automata is quite good. In order to remove the flicking effect, we improved the segmentation result by moving the boundary lines as described in [14].After segmentation, we calculated the optical flow of the moving object using the method described in section 1. The segmented images usually do not perfectly match and there will be small variations in camera focus and lighting conditions. Because of this, simply warping the source image to the destination image will cause flickering effect. Thus, we used image morphing to generate the in between frames. First, we warped the destination image to the source image using optical flow and got a warped source image J^*. All the pixel value in image J^* comes from the destination image. A new image is generated by linearly blending source image I and warped source image J^*. The morphing equation is described in equation 4.

$$I^*(x,y) = (1-f) * I(x - f * d_x, y - f * d_y) + f * J^*(x - f * d_x, y - f * d_y) \quad (4)$$

For the reflections and shadows, if we use image warping at the reflection part, it will make some changes to the texture of the background scene. So, we used the difference image and warp the difference image using optical flow. The difference image does not contain good features, and this will cause some abrupt changes in the synthesized reflection image. So we used the low pass filtering to smooth the warped reflection images.

Fig. 3. Left: Reflection Image. Right: Difference Image.

3.3 Human Walking

For the walking scenes, we have separated the human body into upper body and lower legs. For the upper body, we consider it as a rigid moving object and

the in between images are generated using the method described in section 2. For the legs, we calculated the skeleton structure at each image, and used the skeleton based image warping to warp the legs from one image to the other. At

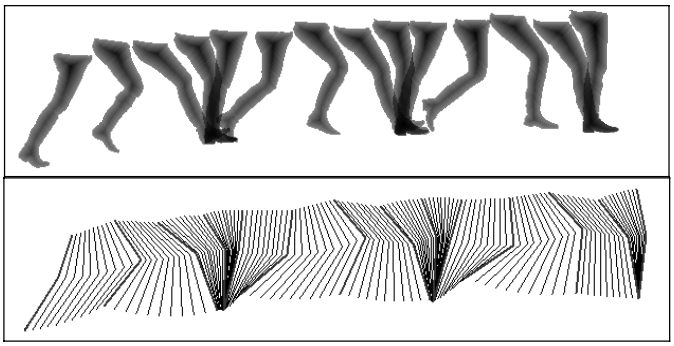

Fig. 4. Upper image: Lower legs in image sequence; Lower image: In between skeleton calculation with forward kinematics

each inner pixel inside the segmented matte, we calculated the shortest pixel distance to the matte boundary. The skeleton structure is easily calculated by searching for the local maxima and the hough transform. The in between skeleton structure is calculated using a simple forward kinematics method. We used the feature based image morphing described in [6] to warp the leg images. At the boundaries of upper body and lower legs, there will be some discontinuity because of different image warping methods. So at these boundaries, we calculated the pixel displacement by linearly blending the displacement from optical flow based warping and skeleton based warping. The left leg is warped in a similar way. One

Fig. 5. The result of combining the upper body and lower leg

problem with left leg is: some of the left legs are heavily occluded by the right legs. For those heavily occluded left legs, we manually painted those left legs using nearby left leg images.

3.4 Trees

The tree movement has stochastic motion feature, so, it is very difficult to use image matching based method to generate the in between frames. We have applied a simple alpha blending followed by debluring operation to generate the in between frames. If we play the alpha blended image sequence, it will look like highly motion blured video. So we used the contrast enhancing filter to decrease the bluring effect. However, the contrast enhancing operation on original images will make the original continuous shot images much sharper than the alpha blended images. So we used simple method to blur the original images. I_0 is the original continuous shot image, I_{-1} and I_1 are the continuous shot images before and after I_0. The new alpha blended image is generated using equation (5).

$$I_0^*(x,y) = 0.1 * I_{-1}(x,y) + 0.8 * I_0(x,y) + 0.1 * I_1(x,y) \tag{5}$$

With this simple approach, we can generate a slightly motion blured video of tree movement.

Fig. 6. The result of tree motion synthesis

4 Experiment and Result

We used Canon 20D digital camera, with remote switch to capture continuous shot images. We can capture about 300 continuous images without interruption. Since the frame rate is 2-3 frames per second, it can be thought as a 3-minute video clip. The camera shutter speed needs to be fast. For indoor scenes, the shutter speed is set to 1/60–1/40, and the ISO is set to 1600. For outdoor scenes, the shutter speed is set to 1/200–1/125, and the ISO is set to 100-200. We used the smallest image size (1728 X 1152) for capturing continuous images. For our experiment, we down sampled the image to 1440 X 960. We used workstation with Intel 3.2Ghz Xeon Processor and 2GB memory for our experiment. For the programming, we used Visual C++ with OpenCV library and Matlab6.5.

In our video synthesis, the main speed bottleneck is the optical flow calculation. Since we are using the pseudo inverse image warping, the window size in optical flow calculation is set to 15–30. For the camera moving at a static environment, the window size is set to 15. For the moving objects and human walking, since the motion is large, we set the window size to 30. The optical flow computation time is about 180 seconds at image size 1440 X 960 with window size 15. For the window size 30, the computation time is about 720 seconds. Since the optical flow computation takes too much time, we have experimented

using the optical flow at a smaller image size, and linearly interpolated to get the enlarged optical flow at a bigger image size. The optical flow computation at 720 X 480 with window size 15 is about 50 seconds. The image synthesis time using inverse warping is less than one second per image. For the camera moving at a static environment, the warped image has some holes at the image boundary. We simply cropped the whole sequence of synthesized images to generate a high resolution video. The cropped image is 1080 X 720, and it is still much bigger than the images from digital camcorder. For our human walking video, since we only used simple forward kinematics based approach, the resultant human walking presents some moon walking effect.

Fig. 7. The result of proposed method

5 Conclusions and Future Works

In this paper, we introduced a new way to synthesize a high resolution video. Instead of enhancing the resolution from video stream, we used continuous shot still images to synthesize the video. In image synthesis using adjacent frames, we have found that the approaches need to be varied based on the motion in the scene. We have simplified the problem by limiting the types of scenes. For the static environment, we used optical flow with pseudo inverse image warping. For the rigid moving object, we used segmentation and optical flow to animate the rigid moving objects. For the reflected image, we used the difference image for image warping and applied low pass filtering to smooth the result. For the human walking, we combined optical flow based image warping and skeleton based image warping to generate the human walking video. Finally, we used simple alpha blending and contrast enhancement filtering to generate the video of tree movement. With our approach, it is possible to make a movie quality video using consumer level digital cameras. Even in the future, as long as the resolution of digital cameras is higher than the digital camcoders, our approach is an alternative way to overcome the resolution limit of digital video devices. For the future work, we want to implement optical flow algorithm on GPU. Many video based rendering approaches are depending on optical flow calculation, but the optical flow computation is too slow for the interactive applications. We will experiment on different types of scenes, for example, driving scenes, moving camera with moving objects, waving water and so on. We also think about using a carefully calibrated video and camera pair. The continuous shot image from the camera is going to be warped based on the optical flow from the video image sequence.

References

1. Horn B., Schunck B.: Determining Optical Flow. In Artificial Intelligence, Vol. 17: (August 1981),185–203.
2. Lucas B., Kanade T.: An iterative image registration technique with an application to stereo vision. In Proc. of International Joint Conference on Artificial Intelligence, 1981. 674–679.
3. Shi J., Tomasi C.: Good features to track. In IEEE CVPR'1994, 593–600.
4. Bouguet, J. Y.: Pyramidal Implementation of the Lucas Kanade Feature Tracker. Intel Corporation, Microprocessor Research Labs, 2000,
5. Sand P., Teller S.: Video matching. In ACM Transactions on Graphics, Vol. 23(3):(2004), 592–599
6. Beier T., Neely S.: Feature-based image metamorphosis. In Proc. ACM SIGGRAPH, (July 1992), 35–42.
7. Wolberg G.: Skeleton Based Image Warping. In Visual Computer, Vol .5(1): (1989) 95–108
8. Seitz S.M., Dyer C. R.: View morphing. In Proc. ACM SIGGRAPH'1996, 21–30.
9. Horry Y., Anjoy K., Arai K.: Tour into the picture: using a spidery mesh interface to make animation from a single image. In Proc. ACM SIGGRAPH, (Aug. 1997), 225–232.
10. Chen S. E.: Quicktime VR - an image-based approach to virtual environment navigation. In Proc. of ACM SIGGRAPH'95 (1995), 29–38.
11. Chuang Y.Y., Goldman D.B., Zheng K.C., Curless B., Salesin D.H., Szeliski R.: Animating pictures with stochastic motion textures. In ACM Transactions on Graphics, Vol. 24(3), 2005. 853–860.
12. Brostow, G. J. and Essa, I.: Image-based motion blur for stop motion animation. In Proc. of ACM SIGGRAPH '01, (2001) 561–566
13. Liu, C., Torralba, A., Freeman, W. T., Durand, F., and Adelson, E. H.: Motion magnification. In ACM Trans. Graphics, Vol. 24(3): (2005), 519–526
14. Li Y., Sun J., Tang C.K., Shum H.Y.: Lazy snapping. In ACM Transactions on Graphics, Vol. 23(3): (2004), 303–308.
15. Rother C., Kolmogorov V., Blake A.: "GrabCut": interactive foreground extraction using iterated graph cuts. In ACM Trans. on Graphics, Vol. 23(3): (2004),309–314.
16. Li Y., Sun J., Shum H.Y.: Video object cut and paste. In ACM Transactions on Graphics, Vol. 24(3): (2005), 595–600
17. Chuang Y.Y., Agrawala M, Curless B., Salesin D.H., Szeliski R: Video matting of complex scenes. In Proceedings of ACM SIGGRAPH, (2002), 243–248
18. Sun J., Jia J., Tang C.K., Shum H.Y.: Poisson matting. In ACM Transactions on Graphics, Vol. 23(3): (2004), 315–321.
19. Vezhnevets V., Konouchine V.: "Grow-Cut" - Interactive Multi-Label N-D Image Segmentation. In Int. conf. on the Computer Graphics and Vision (Graphicon 2005), 150–156.
20. OpenCV: OpenCV: The Open Computer Vision Library. Sourceforge.net/projects/opencvlibrary.