Texture Painting from Video

Jinhui Hu
University of Southern California
3737 Watt Way, PHE 404
USA(90089), Los Angeles, California
jinhuihu@graphics.usc.edu

Suya You
University of Southern California
3737 Watt Way, PHE 404
USA(90089), Los Angeles, California
suyay@graphics.usc.edu

Ulrich Neumann
University of Southern California
3737 Watt Way, PHE 404
USA(90089), Los Angeles, California
uneumann@graphics.usc.edu

ABSTRACT

Texture mapping is an important research topic in computer graphics. Traditional static texture-maps are limiting for capturing a dynamic and up-to-date picture of the environment. This paper presents a new technique called texture painting from video. By employing live video as the texture resource, we are not only able to create an accurate and photo-realistic rendering of the scene, but also can support dynamic spatio-temporal update in the structure of texture model, database, and rendering system. We present our approaches towards the system requirements and experimental results for both simulation and real datasets.

Keywords

Texture Painting, Image Warping, Image Registration

1. INTRODUCTION

Texture is a crucial element in today's graphics oriented applications. In many cases, the value of the applications is increased if both the geometric information and the appearance of the generated images are accurate and realistic analogues of the real world. Texture mapping is a relatively efficient means to create the appearance of realism without the tedium of modeling and rendering every 3-D detail of a surface.

Effective generation of textures has been becoming an important research issue in computer graphics and image processing. There are many ways to acquire data for creating scene textures, such as texture synthesis and direct texture mapping from imagery. Much research has been conducted on the texture synthesis, a technology inspired by research in texture analysis and statistics. While the results from the stochastic textures are useful, this class of approach is unable to deal well with more complicated textures and hard to achieve photo-realistic effects.

To create an accurate and realistic appearance of rendering scene, real world images are often captured and used for texture creation. While most graphics systems support high-quality texture mapping from real imagery, they are limited to static texture resources that must be created prior to use. Static textures are usually derived from fixed cameras at known or computed transformations relative to the modeled objects. The creation and management of such texture databases is also time consuming since it includes image capture and the creation of mapping functions for each segmented image and model patch Once their relationships are established, the texture images are mapped to the geometric models during scene rendering. Therefore, such static texture-maps are limited for applications requiring a dynamic and up-to-date picture of the environment.

This paper presents a novel technique, called *texture painting from video*, to cope with the aforementioned limitations of the static texture mapping and visualizations. Live video is used as texture resource and mapped dynamically onto the 3D models of scenes to reflect the most recent changes of the environments. The video streams can be acquired from stationary or moving cameras such as handheld camcorder, and their projections onto the 3D model are achieved in real-time. Unlike the traditional texture mapping in which each texture image is a

The Journal of WSCG, Vol.13, ISSN 1213-6964
WSCG'2005, January 31-February 4, 2005
Plzen, Czech Republic.
Copyright UNION Agency – Science Press

priori associated with, and mapped onto, patches of the geometric models, our approach dynamically creates the associations between the model and image as a result of image projection during the rendering process. In this case, we can automate the texture mapping process. As new images arrive, or as the scene changes, we simply update the camera pose and image information, rather than repeat the time consuming process of finding mapping transformations, hence make it possible to handle live video streams in real-time.

Related Work

Texture synthesis is a popular technique for texture creation. Such an approach is able to take a sample texture and generate an amount of images, while not exactly like the original, will be perceived by human beings to be the same texture. The parametric model based approach uses a number of parameters to describe a variety of textures ([Hee95, Por00]). The non-parameterized texture, or example based methods generate textures by directly copying pixels from input textures [Efr99]. Recently, [Ash01] suggested an approach to synthesize textures using whole patches of input images. While the texture synthesis technique has been demonstrated successfully in certain applications to be a useful tool for texture generation, the results of the synthesized textures are not photo-realistic and lack of texture details.

Texture painting is an alternative way to create textures. A number of interactive texture painting systems have been suggested. [Iga01] presented a 3D painting system that allows users to directly paint texture images on a 3D model without predefined UV-mapping. [Ber94] employed the Haar wavelet decomposition of image for multi-resolution texture painting. [Per95] painted multi-scale procedural textures on 3D models. Most of current texture painting systems are interactive, allowing users to easily design and edit textures to achieve desired effects. The results can be aesthetically pleasing, but it is hard to make them photo-realistic.

A straightforward way to produce realistic textures is to use real world images as texture resources. To create a complete texture map covering the entire scene being textured, multiple images are used. [Ber01] used high resolution images captured from multiple viewpoints to create high quality textures. [Roc99] stitched and blended multiple textures for creation of textures. [Ofe97] suggested a quadtree approach to represent multi-resolution textures extracted from image sequences. This method requires user to mark the texture area to be extracted, and then manually track the area for a short image sequence (less than 16 images).

Recently several works suggested using video clips as texture resources. [Sch00] proposed the idea of "video textures". From the input clip of limited length, they can generate an infinitely long image sequence by rearranging and blending the original sequence. [Soa01] presented the idea of dynamic textures that are sequences of images with a certain stationary property in time. By learning a model from the input sequence, they synthesize new dynamic textures. Both above methods use the textures in a non-traditional way to achieve desired effects and goals of applications. Their systems deal only with the special textures of repeated patterns, such as sea waves, smoke plumes, etc. In our approach, however, we are dealing with general image textures, i.e. a multidimensional image that is mapped to a multidimensional space. Rather than synthesize new textures, we directly use the original video captured from any real world scene to produce an accurate and realistic appearance of the environment.

2. TEXTURE PAINTING FROM VIDEO

Using live video as the texture resource can offer us many benefits to reproduce real scene. However, we also encounter several technical barriers needed to be overcome. First, the video streams need to be acquired continually and updated, which may lead to infinite texture storage. A straightforward approach of using certain amount of texture memory is unable to keep old texture data from where the projection was a few moments ago. Such texture retention requires approaches being able to persist in the projection for each new video frame onto a surface area.

Second, since we want to paint the dynamic videos onto the surface of 3D model. Only if the camera positions and orientations are known, these data can be projected onto the scene model correctly, thereby highly precise tracking of camera pose and alignments between image frames are required.

Third, due to lack of the models for dynamic objects, those foreground moving objects need to be segmented from the input video to persist only the background textures being projected onto the surfaces of scenes.

Proposed Approach

We present our approaches towards the system requirements essential to the texture painting from video. We propose novel methodologies for rapid creation of dynamic textures from live video streams and their data retention, storage management, and texture refinement. We also implement a prototype of real-time 3D video painting system based on the methodologies we proposed.

Figure 1 illustrates the main structure of our texture painting from video system. As stated above, the main challenge of using live video as texture-maps is how to effectively handle the infinite video streams within a certain amount of texture buffers so that the rendering algorithm can persist in the texture projection for each new frame onto a surface area. Given the fact that only limited scene geometry can be visible from a viewpoint, we propose the idea of “base texture buffer”, which is a texture buffer associated with a model patch or a group of neighboring patches being visible from the viewpoint (Figure 2). The base texture buffer is first initialized as white texture, and then dynamically updated with the new coming frames.

To update the base texture buffer for each group of visible patches, we transform each new frame to the base texture buffer. Let the projection matrix associated with the base texture buffer is P and the projection matrix of the new coming frame at a viewpoint is P_v. For every new frame, the transformation P_t between the new frame and the base texture buffer is

$$P_t = P * P_v^{-1} \qquad (1)$$

By using equation (1), we are about to dynamically warp every new frame from different viewpoint to the common base texture buffer, and project the updated content onto the visible surface of scene. In this case, we overcome the problems of infinite texture storage and also the time consuming process of polygon clipping for every video frame.

To achieve accurate image alignment, the 3D model is generated and refined based on LiDAR data [You03], and we recover the camera pose using a robust tracking approach proposed in [Neu03]. Then we refine the recovered alignments in 2D image domain to achieve seamless texture images. Several other core steps, including selective texture painting, base buffer selection, and occlusion detection, are also suggested and will be detailed in following sections.

3. APPROACH DETAILS

3.1 Model Based Image Warping

A key part of texture painting from video is to dynamically update the base texture buffer, which is based on a process of model based image warping. First, we select a base texture buffer. The pose of the base buffer relative to the 3D model of the scene is computed. The correspondence of each pixel in the

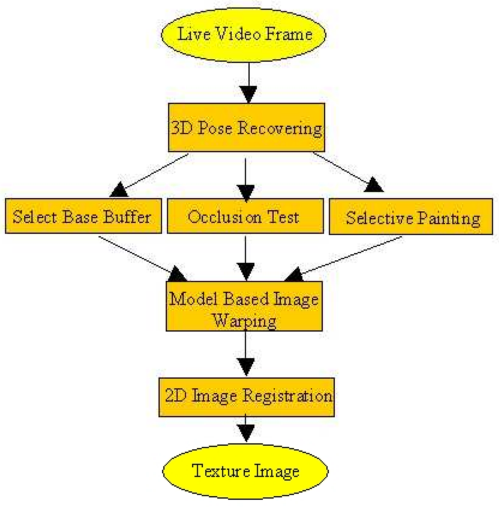

Figure 1 – Overview of the texture painting from video system.

base buffer to the 3D model is computed using equation (2), where I_b denotes the pixel in the base buffer, P_b is the projection matrix of the base buffer, and M is the corresponding point on 3D model. Next, when a new video frame comes, if the model correspondence to the base buffer is visible from current camera position, we update the base buffer with current new frame. This is done by a model based image warping operation.

$$M = I_b P_b^{-1} \qquad (2)$$

$$I_v = P_v M \qquad (3)$$

$$I_b = I_v \qquad (4)$$

As indicated in Figure 2, for each pixel I_b in the base texture, we can find its correspondence M on the 3D model. Given the tracked camera pose and the projection matrix denoted P_v, we project the 3D point M back to the image plane to find its corresponding pixel I_v using the Equation 3. We

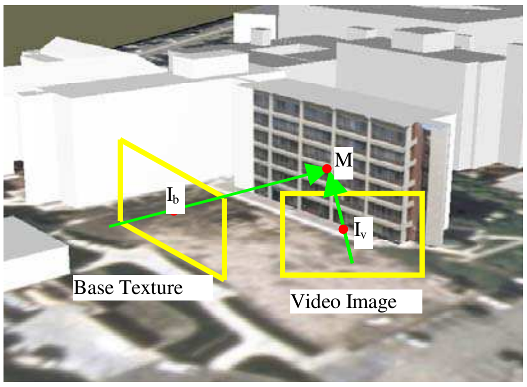

Figure 2 - Model based image warping.

then update its color information in the base texture buffer using Equation 4. This warping process is repeated for every pixel contained in the base texture buffer.

The 3D model based approach is flexible, allowing the camera moving freely in any 3D environment. It requires, however, highly precise camera tracking, which is usually hard to achieve, especially in an outdoor environment. We compensate the tracking errors by employing a 2D image registration approach in Section 3.5. Figure 3 illustrates the result of the model based warping approach.

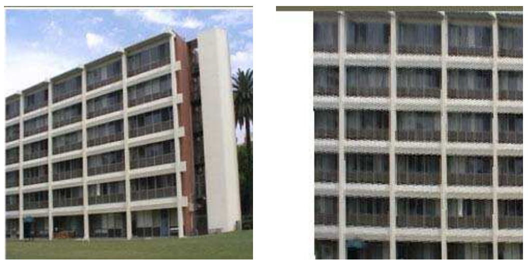

Figure 3 - Result of the model based image warping. Left: original image, right: image warped to base buffer.

3.2 Improve Warped Image Quality

Direct back-warping of the image to the base buffer may result in aliasing. To improve the final texture-maps quality, we use bilinear interpolation for anti-aliasing. As indicated in Figure 4, each pixel coordinates of *I(xb,yb)* in the base texture buffer is treated as a real number. Using the Equations (2-4), we can find its corresponding pixel in video frame, the coordinates of which, *I(xv,yv)*, are also real numbers. Usually, *I(xv, yv)* will not fall into an integer grid in the image. We then interpolate the color information using a four-neighboring bilinear interpolation. Figure 4 shows the result of applying the approach to Figure 3,which apparently improved the image quality.

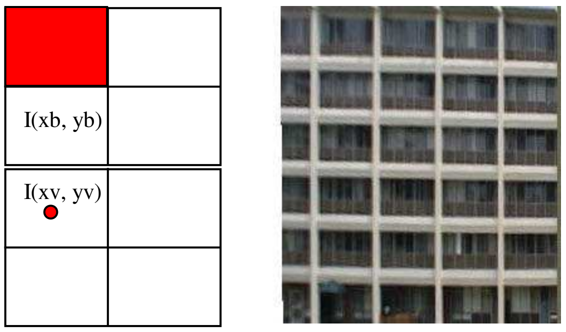

Figure 4 - Using bilinear interpolation to improve warped image quality.

3.3 Occlusion Detection

Under some circumstances, although the 3D model is visible from the base buffer, it may be occluded from current camera viewpoint. The model based warping and texturing will project part of the frame onto the occluded model areas (Figure 5 left). This occlusion problem is solved using depth maps. From each camera viewpoint we render the 3D model to obtain an estimated depth map. When doing the image warping, we first find the corresponding 3D point for an image point being warped. We then project it back to the image plane, and compare its depth value with the estimated depth map. Finally, we keep the pixel projection only if its depth value is less than the corresponding depth value in the depth map. The result is shown in Figure 5.

3.4 Selective Texture Painting

Texture painting from video has the advantage of dynamic texture updating to capture the most recent environment changes. However, if we don't select the content to be textured, it will project everything in the video sequence onto the 3D environment, and consequently give an undesired result. Figure 6 (left) shows a case of moving objects are painted onto the 3D model as part of background textures.

We can view that there are three types of texture information in a video sequence: background textures, foreground textures such as trees, and dynamic objects such as moving people or vehicles. The background textures are the only part we are interested in and intending to process, since they have corresponding 3D model to be textured.

By employing a background learning approach, we segment and remove the undesired objects from the input video. Given a number of training frames, we first learn the images to estimate a background model based on the linear average model. General average method will only work for static cameras. To deal with moving cameras, we first warp each new frame

$$I_{back} = \frac{1}{N}\sum_{i=1}^{N} I_{v,i} \tag{5}$$

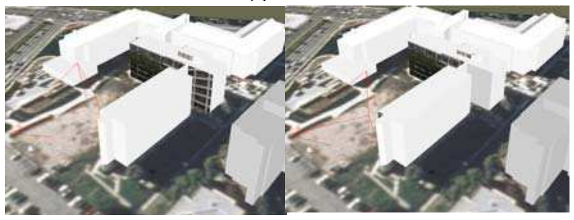

Figure 5 - Occlusion processing. Left: texture painting before depth test. Right: after depth test.

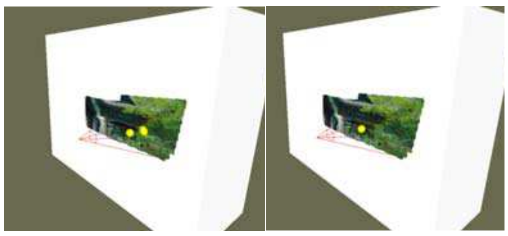

Figure 6 - Selective texture painting. Left: texture painting without background learning. Notice that the moving yellow sphere is painted as part of background texture. Right: after background learning. Only the static scene is painted as background texture.

onto the base texture buffer (which is static relative to the 3D model), then we average each warped frame over the base texture buffer. After that, we segment the objects from the background using image subtraction approach. Figure 6 (right) shows the result after the background learning.

3.5 Refine Texture Alignment

As mentioned above, inaccurate camera pose tracking will result in misalignments between the textured images as shown in Figure 7 (left). The alignment needs to be refined to improve the visualization value. Our approach to this problem is to perform the refinement in 2D image domain, i.e. we first register the video frames based on the camera tracking data and 3D model, and then re-align the registered image with the base buffer using a 2D image registration approach.

- *Motion Model*

Affine model is used for estimating the transformation parameters between the warped video frame and the base texture buffer.

$$\begin{bmatrix} I_{xb} \\ I_{yb} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} I_{xv} \\ I_{yv} \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \quad (6)$$

- *Feature Matching*

Corner feature of scenes is used as matching primitive to find the correspondences between the video frame and the base texture buffer. The Harris algorithm is employed to detect image corners, and the SSD (Sum of Squared Difference) approach is used for feature matching. Since the images have been aligned based on the camera tracking data, the SSD matching space is greatly reduced.

- *Parameter Optimization*

While three pairs of correspondences are sufficient to compute a unique solution for affine transformation, we use all matching corners to guarantee accuracy. Least square approach is used to estimate the optimal parameters.

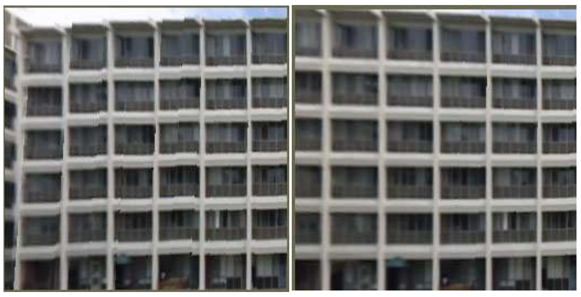

Figure 7 – Refining texture alignment. Left: texture painting using only 3D model based warping, which results in significant misalignment due to inaccurate camera pose. Right: texture painting after refining the texture alignment.

Once the affine parameters are estimated, the 3D-warped image is warped again using the affine transformation to the base texture buffer. Figure 7 (right) shows the refined result of using the approach.

3.6 Select Optimal Base Buffer

Given a 3D scene model, the selection of the base texture buffer is an optimization problem. A straightforward way is to assign one base buffer for each polygon surface. The advantage of this simple method is that it is not required to clip polygon. However, when the number of polygons becomes large, this method becomes unfeasible. So we need to minimize the number of required base texture buffers. One possible approach is to combine as many nearby polygons as possible into one base buffer. However, complex models typically have very varying surface normal, even nearby polygons. Having those polygons with very different normal shared one base buffer will lead to very poor texture reproduction. Another issue arising from the different polygons sharing one base buffer is that we need to clip the 3D model during texture mapping process. The number of clipped polygons will affect the rendering speed, so we would like also to minimize the number of clipped polygons.

The problem of selecting optimal base buffers can be proved to be a NP optimization problem. [Mat99] used a heuristic algorithm to solve the problem. In this paper, we approach to combine neighboring polygons with similar normal into the same base buffer. Our ongoing work is deeply emphasizing this problem.

4. EXPERIMENTAL RESULTS

We have tested the proposed video painting approach on both simulation and real datasets. Figure 8 shows the results from our simulation experiment. A 3D cube model covered by a real image is used to mimic a 3D environment (Figure 8a), and a synthetic camera (shown with a red frustum in Figure 8a) moving freely within the environment is used to "capture" the scene. The image captured from a viewpoint by the camera is shown in the Figure 8b, which is used to simulate input video. We then applied the proposed approach to this scenario. Figure 8e is the input image warping to the base buffer, and Figure 8d shows the images painting onto the 3D model. Since the synthetic camera pose is perfect in this simulation experiment, no further 2D image registration is needed. The whole system is in real time, achieving ~30fps on a 1.1GHZ DELL workstation. It is worth to mention that the camera motion is completely arbitrary, and the captured images are painted persistently onto the correct surface areas, as shown in Figure 8c, and Figure 8f.

We also tested the approach on real data captured by a user walking around USC campus. A portable tracking and video system was used to collect camera tracking data and video streams. The entire campus model was reconstructed using a LiDAR modeling system [You03]. We then applied the approach to the collected dataset. The video frames were first aligned to a temporary buffer based on the tracked camera pose; then the 2D affine transformation between the warped image and base frame was computed to refine the alignment; and finally the warped image is re-warped back to the base buffer using the computed 2D transformation. The whole processing for this scenario achieved ~10fps, in which the most time-consuming part is the 2D image registration. We notice, however, the fact that most part of two successive image frames is overlapped, and there is no need to update the painting for every frame. So, it is sufficient to only update the painting buffer in every certain frame (e.g. every 20th frame) to speedup the system performance. Figure 9 shows the results of the real data experiment.

5. CONCLUSION

Texture is a crucial element in today's graphics oriented applications. Traditional static texture-maps are limiting for capturing a dynamic and up-to-date picture of the environment. This paper presents a new technique of texture painting form video. By employing live video as texture resource, we are not only able to create an accurate and photo-realistic appearance of the rendering scene, but also can support dynamic spatio-temporal update in the structure of texture model, database, and rendering system. We present our approach towards the system requirements and experimental results for both simulation and real datasets.

While the proposed approach is novel, there are still several technical barriers we are addressing in our ongoing work, including

- Selection of the optimal base texture buffer for complex scene models. Currently we simply approach to combine the neighboring polygons with similar normal into the same base buffer, which works fine for most of the man-made scenes' models. We would also like to deeply explore the solution for more complex scenes.
- Real time implementation to support multiple video streams. Today's computing and graphics hardware has reached a stage where many complex real-time computations could be performed with the high-end graphics processors (GPU). How to effectively utilize the programmable GPU features to speedup the video processing is also our emphasis.

ACKNOWLEDGEMENT

This work was supported by the National Geospatial Intelligence Agency (NGA) under a NGA University Research Initiative (NURI) program, and in part by a Multidisciplinary University Research Initiative (MRUI). We thank the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, for their support and facilities. The first author would also like to thank Pamela Fox for her proof reading and comments.

REFERENCES

[Ash01] Ashikhmin, M. Synthesizing natural textures. ACM Symposium on Interactive 3D Graphics, pp. 217–226, 2001.

[Ber94] Berman, D.F. etc. Multiresolution painting and compositing. Proceedings of SIGGRPAH 94, 1994.

[Ber01] Bernardini, F., Martin, I. M. and Rushmeier, H. Hight-quality texture reconstruction from multiple scans. IEEE Visualization and Computer Graphics, Volume: 7, Issue: 4 pp. 318-332,2001.

[Efr99] Efros, A.,and Leung, T. Texture synthesis by non-parametric sampling. ICCV, pp. 1033-1038, 1999.

[Efr01] Efros, A., and Freeman W. T. Image quilting for texture synthesis and transfer. Proceedings of SIGGRAPH 2001, pp. 341–346,2001.

[Hee95] Heeger, D.J. and Bergen, J.R. Pyramid based texture analysis/synthesis. Proceedings of SIGGRPAH 95, pp.229-238, 1995.

[Iga01] Igarashi, T., and Cosgrove, D. adaptive unwrapping for interactive texture painting. ACM Symposium on Interactive 3D Graphics. 2001.

[Jia01] Jiang B., Neumann U., Extendible Tracking by Line Auto-Calibration. International Symposium on Augmented Reality, pp.97-103, New York, October 2001.

[Mat99] Matsushita K., and Kaneko, T. Efficient and handy texture mapping on 3d surfaces. In Proc. of Eurographics, pp.349-358, 1999.

[Neu03]Neumann, U., You, S., Hu, J., Jiang, B. and Lee, J. W. Augmented virtual environments (AVE): dynamic fusion of imagery and 3D models, IEEE Virtual Reality, pp. 61-67, Los Angeles California, 2003.

[Ofe97] Ofek, E. etc. Multiresolution textures from image sequences. IEEE Computer Graphics and Applications, Volume:17, Issue:2 pp.18-29, 1997.

[Per95] Perlin, K. Live paint: painting with procedural multiscale textues. Proceedings of SIGGRAPH, pp.153–160, 1995.

[Por00] Portilla, J., and simoncelli, E.P. A parametric texture model based on joint statistics of complex wavelet coefficients. IJCV 40, 1(Oct.) pp. 49-70, 2000.

[Roc99] Rocchini, C., Cignoni, P. and Montani, C. Multiple textures stitching and blending on 3D objects. In Eurographics Rendering Workshop, 1999.

[Sch00]Schodl, A., Szeliski, R., Salesin, D. H., and Essa, I. Video textures. Proceedings of SIGGRAPH, pp. 489–498, 2000.

[Soa01] Soatto, S., Doretto, G., and Wu, Y. Dynamic textures. In Proceeding of IEEE International Conference on Computer Vision, II, pp. 439–446, 2001.

[Wei00] Wei, L. Y., and Levoy, M. Fast texture synthesis using tree structured vector quantization. Proceedings of SIGGRAPH, pp.479–488, 2000.

[You03]You, S., Hu, J., Neumann, U. and Fox P. Urban Site Modeling From LiDAR, Second International Workshop on Computer Graphics and Geometric Modeling, Montreal, CANADA, 2003.

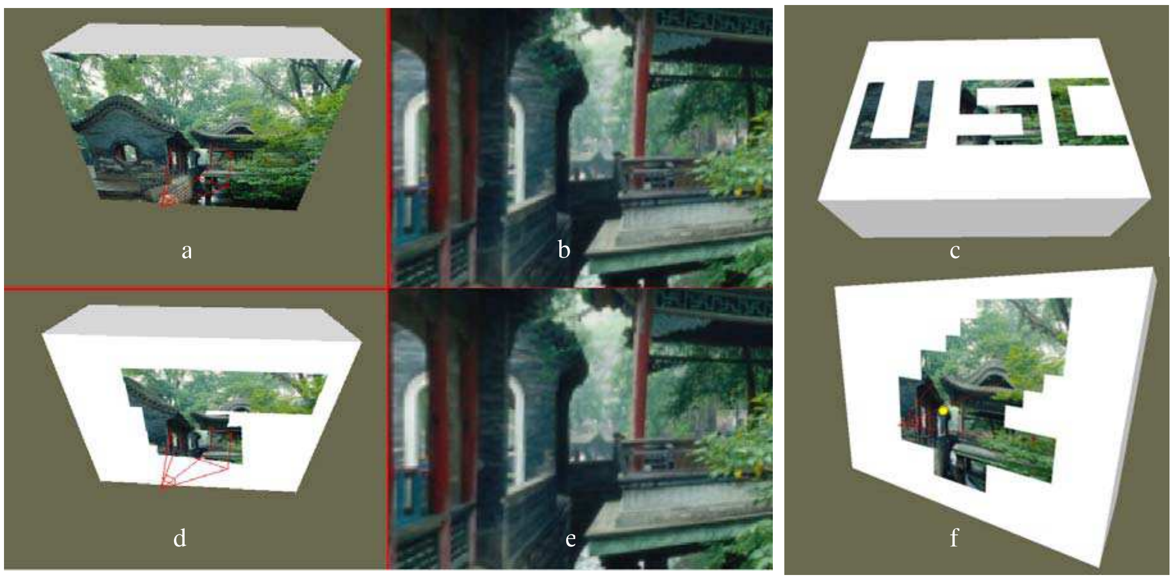

Figure 8 - Texture painting with simulation dataset. A cube model covered by a real image is used as a simulated environment (a). A synthetic camera moving inside the environment is shown in red frustum. The image viewed from the camera's viewpoint, which is used as simulation of input video (b). The video frame is warped based on 3D model onto the base buffer (e), then painted as texture onto the same 3D model (d). The (c), (f) show the painting results of simulated data.

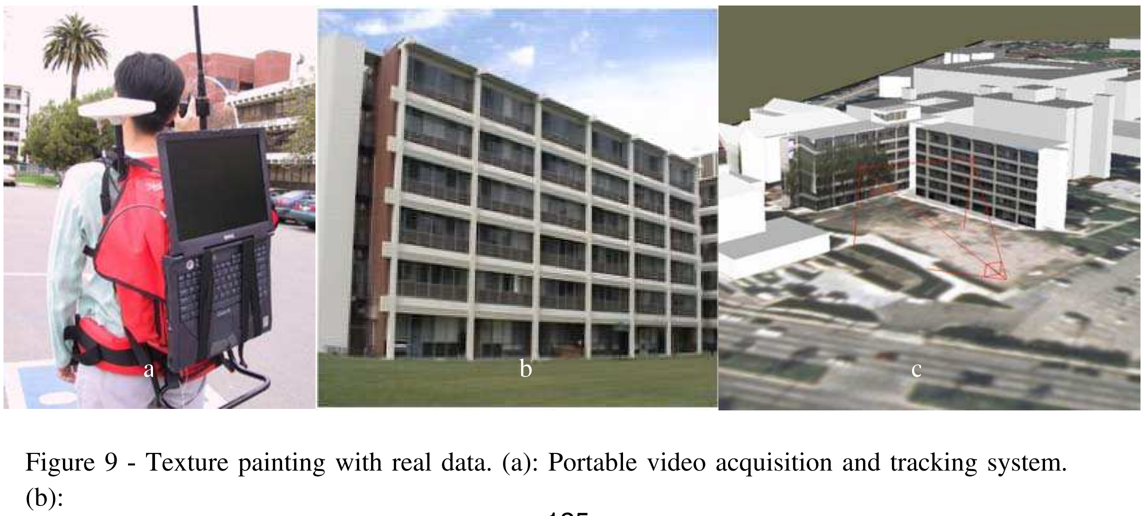

Figure 9 - Texture painting with real data. (a): Portable video acquisition and tracking system. (b):