TOUR INTO THE PICTURE REVISITED

N. Li and Z. Huang

School of Computing
National University of Singapore
Singapore 117543
{linan, huangzy}@comp.nus.edu.sg

ABSTRACT

TIP (Tour Into the Picture) was introduced by Horry et al. [HAA97]. Given only one picture, a viewer can tour into the scene as painted on the picture. Based on our implementation of TIP, we noticed a problem: the visual quality drops drastically when the viewpoint tours into the scene. It contradicted the real world experience. We addressed this problem by introducing the use of multiresolution representation of the picture. We have achieved the goal that the visual quality keeps nearly unchanged in the touring. Moreover, we have integrated the 3D models into TIP. By estimating the light sources, we generated shadows into the scene.

Keywords: Image based rendering/modeling, tour into picture, VR walk through, multiresolution image, shadowing.

1 INTRODUCTION

Image based rendering, complementary to geometry based rendering in VR walk through, makes use of a set of pre-acquired imagery to generate new views. It is an interesting approach for three major advantages. First, its rendering speed is independent of the scene complexity. Second, the rendering is a process of image resampling so it only relies on the normal CPU not any special rendering hardware. Third, its rendering results are naturally photo realistic.

TIP (Tour into the picture), an image based rendering technique, is introduced by Horry et al. [HAA97]. Given only one picture, it allows a viewer to tour into the scene as painted on the picture. TIP provides a simple but powerful GUI (graphical user interface) which allows users to easily tour into the scene, i.e., to provide its different views.

In general, providing different views of a scene requires more than one picture. In order to have different views from only one picture of the scene, TIP depends largely on the subjective interpretation of the picture. TIP employs a spidery mesh over the picture to obtain a simple pseudo-3D scene model: the background scene is modeled with, at most, five 3D polygons; each foreground object appearing in the picture is represented as a texture-mapped image on a billboard. Note that, in TIP, a user specifies the foreground and background objects. The specification depends on the user's sense and aim. It does not matter what the scene actually is.

The result of TIP is impressive. A user can feel a plain 2D picture becomes a 3D scene. It can be toured into and viewed from different viewpoints. TIP can find applications where only one image of a scene is available such as in an art museum.

Based on our implementation of TIP, we noticed a problem: the visual quality drops drastically when the viewpoint tours into the scene. It contradicted the real world experience. We addressed this problem by introducing the use of multiresolution representation of the picture. We have achieved the goal that the visual quality keeps nearly unchanged in the touring. Moreover, we have integrated the 3D models into TIP. By estimating the light sources, we generated shadows into the scene.

2 OTHER RELEVANT WORK

The progress in imaged based rendering can be traced through three different research areas: photogrammetry, computer vision, and computer graphics. In photogrammetry, the problems of camera calibration, distortion correction, image registration, and photometrics have progressed toward the synthesis of images through the composition of reference images. In computer vision, similarly, problems like robot navigation, image discrimination, and image understanding have naturally led to the same direction. In computer graphics, the progress towards image based rendering was first motivated by the desire to increase the visual realism of the approximate geometric descriptions by mapping images onto their surfaces, i.e., texture mapping [Heck89].

Besides TIP, other image based rendering methods in computer graphics include viewpoint interpolation [CW93, MB95], view morphing [SD96], and interpolation from dense samples [LH96, Gort96]. In viewpoint interpolation, new views were synthesized from two cylindrical panoramic views created by mosaic. The rendering was characterized as reconstruction of a continuous representation of the plenoptic function from a set of discrete samples. The disparity maps were computed between adjacent cylindrical panoramas using a cylindrical variant of the epipolar constraint. New views were synthesized by warping the existing panoramas based on the disparities. View morphing combines image interpolation and image morphing. New views can be synthesized correctly, proved by mathematics, using interpolation if the reference images are first rectified. It consists of three steps: rectification, linear disparity interpolation, and de-rectification. The dense samples based approach constructs an explicit 4D data structure containing a subset of the plenoptic function that captures the complete flow of light in a bounded region of space, i.e., Light Field or Lumigraph. The data structure contains the color intensity for all viewing rays intersecting in a closed volume around objects. New views of the objects can be synthesized by reconstructing light rays passing through the new camera center from a set of discrete samples.

The recent work includes the design of hardware for image based rendering [Rega99, MDK99]; a hierarchical representation for image based rendering with LDI tree [Chan99]; rendering with concentric mosaic [SH99]; application of image based rendering together with geometry based rendering for a guaranteed frame rate [AL99]; and image based modeling of skin aging and wrinkles on the human faces [BKT00].

3 TIMP: TOUR INTO MULTIRESOLUTION PICTURE

As our work is an extension of TIP, we first brief it based on our implementation. The major steps are shown in Figure 1. They are:

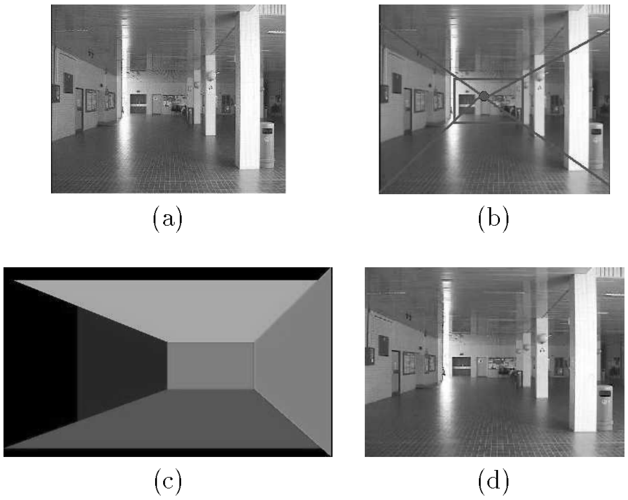


Figure 1: Major steps of the TIP diagram.

1. Construct spidery mesh (Figure 1 (b)): In this step, the user interactively specifies the spidery mesh consisting of one vanishing point, four radial lines, and two rectangles over the original picture (Figure 1 (a)). The two rectangles are the inner rectangle, which intuitively can be regarded as a window out of which we look at the infinity, and the outer rectangle, which corresponds to the outer frame of the input image. The four radial lines radiate from the vanishing point. Each edge of the inner rectangle is parallel to one edge of the outer rectangle. The inner rectangle is also used to specify the rear window in the 3D space that the viewpoint cannot go through.

2. Model the 3D background: In this step, we first make a 2D decomposition of the outer rectangle into five smaller regions each of which is a 2D polygon in the outer rectangle. As illustrated in Figure 1 (c) and Figure 2, they represent the floor, right wall, left wall, rear wall, and ceiling respectively (the rear wall is also the inner rectangle). Then, we can derive the textures of these 2D rectangles directly from the image. Assume (1) every adjacent 3D rectangle is orthogonal to the others, (2) the 3D rear wall

is parallel to the view plane, and (3) the 3D floor is orthogonal to the view up vector. The texture map of each rectangle is exactly the part of image it covers.

3. Render the scene using texture mapping (Figure 1 (d)): In this step, the 3D reconstructed scene consisting of five rectangles and their texture maps can be rendered directly. By changing the viewpoint during the rendering, the touring visual effect is resulted for the image. One example is shown in Figure 3 with three snapshots.

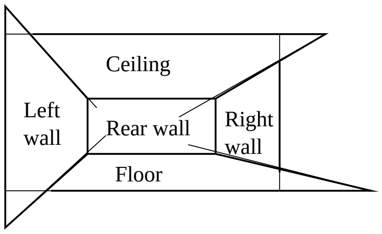


Figure 2: The five reconstructed rectangles (walls).

(a)

(b)

(c)

Figure 3: Tour into the picture of LT27 hall, NUS.

We have found a problem of TIP from our implementation. The image of each frame will get blurred seriously when the viewpoint moves close. This problem is due to the use of a single resolution of the picture. The rendering is sharp at the beginning. When the viewpoint moves in, the synthesized image becomes more and more blur because the size of the part of the picture contributed to the display using texture mapping is going down continuously.

The use of multiresolution representation is a natural solution. Using the multiresolution. When the viewpoint is far, a low resolution can be used. The resolution will increase when the viewpoint tours in. Ideally, the size of the part of the picture contributed to the display, i.e., the sampling rate, should keep unchanged.

First, we derive the multiresolution representation of a picture. Deriving a picture of very high resolution is easy today with the advent of more affordable digital cameras and image scanners. From the highest resolution, different levels of lower resolution can be produced using a common image editor. The image pyramid (Figure 4), a similar data structure used in the mip-mapping [Will83], is used for our purpose.

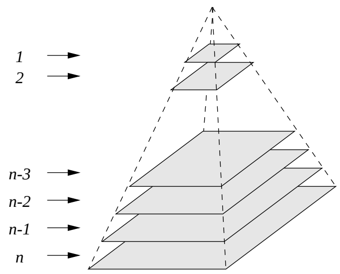


Figure 4: Image pyramid: a data structure representing multiresolution images.

Second, we divide each multiresolution representation into five parts for the five reconstructed rectangles. For each frame, the image to be used may take different resolution for different parts respective to different rectangles (walls). For example, when the viewpoint is approaching the back wall of the scene, the distance from the viewpoint to the other four walls (ceiling, left and right walls) may still remain the same. So only the back wall need to use higher resolution. We divide each image into five parts and store them accordingly. Thus, each wall has its own pyramid of multiresolution images. When touring in, a proper resolution image is selected from the current viewpoint-wall distance for each wall.

Third, we select the most suitable resolution representation for each rectangle in the touring. We try to keep the product of the image resolution and distance of the viewpoint towards the 3D rectangle in a right range. The moment to use a new resolution is determined by the function of distance. In order to maintain the best of display quality, we use a simple way described by the formula: $T_{low} \leq L_i \times D_i \leq T_{high}$, where L_i is the level number of the resolution for the rectangle (wall) i. The L_i equals value 1 for the lowest resolution which is one fourth resolution of the next level, i.e., level 2; D_i is the distance between the current viewpoint and the rectangle (wall) i; T_{low} and T_{high} are constant thresholds. Intuitively, once the product $L_i \times D_i$ smaller (or greater) than the lower (or higher) threshold bounds T_{low} (or T_{high}), a higher (or lower) resolution image is selected to use.

Finally, we improve the rendering by interpolating of the multiresolution images. There is a noticeable abrupt change when a new resolution of image is selected to use. We address the problem by using interpolation so that the display can smoothly transit to the next resolution. We describe it using one wall without the loss of generality. Using the formula in the previous paragraph, at the distance D_0, L_i is adopted for the wall i; at the distance D_1, L'_i is adopted. If the distance D falls into $[D_0, D_1]$, the image used is an interpolation between L_i and L'_i. For simplicity, we linear interpolate the distance: $IMG(D) = (IMG(D_1)(D-D_0)+IMG(D_0)(D_1-D))/(D_1-D_0)$, where $IMG(D)$ represent the image at a distance D. Thus, $IMG(D_0)$ and $IMG(D_1)$ are from the sampling image level L_i and L'_i. In our implementation, the size of a lower resolution image is one fourth of the higher one in the image pyramid. So before the interpolation, the lower resolution one must be scaled four times to the same size.

We show the comparison in Figure 5 and 6 by four snapshots respectively. In Figure 5, the result of TIP has shown that the image quality is decreased when the viewpoint tours in. In Figure 6, the result of TIMP has shown that the image quality is nearly unchanged. For the first two snapshots, the image quality of TIP is better because of the use of higher resolution. However, the decreasing of image quality when getting closer contradicts to the real world experience. Moreover, when the viewpoint is far, the use of higher resolution in rendering lowers down the frame rate of the display.

4 INTEGRATION OF 3D MODELS

Integration of other 3D models can add more contents into the scene that are not painted in the original picture. It is a very useful way to enhance a virtual environment. The difficulty is to seamlessly integrate the 3D models and achieve a consistent visual effect. The factors of consistency include the model size, material, location, lighting condition, and shadows. As the 3D scene is constructed consisting of five rectangles, the consistency problem of size and location is less serious though the object cannot put arbitrary as that in the scene modeled in real 3D, e.g., behind the pillar in Figure 7.

We compute the shadows because they are essential for a 3D scene. The first task for shadowing is to derive light sources of the scene. Though there is work on automatic lighting recovery from images, it is yet to use especially for the recovery from only one image. So we need the viewer to interactively specify the light. It is not difficult in most cases, for example, in the case shown in Figure 7, the light source is an area light from the top left side of the scene.

We apply the real time shadow algorithm described in [MB96]. Briefly, to compute and draw shadows on a plane, we need to derive the shadow matrix and use it to multiply the model matrix of the 3D display pipeline. The shadow matrix is computed from the ground plane matrix and light source position. The default ground plane is xz plane. Given three vectors v_1, v_2, and v_3 inside the ground plane, we have $vp_1 = v_1 - v_2$, $vp_2 = v_1 - v_3$, and the normal vector of ground $v = vp_1 \times vp_2$. Finally the shadow matrix can be computed as $M[i][j] = -vp_1[i] \times v[i]$ for $i = j$ and $M[i][j] = v \cdot vp_1 - vp_1[i] \times v[i]$ for $i \neq j$.

After the shadow matrix is computed, it is used to multiply to the model matrix. And we need to disable depth testing while drawing the shadow, so that a projection of a dark model with depth will be shown on the floor and walls. A shadow of that 3D object can then be displayed in the proper position (Figure 8 (a)). More details and explanations can be found in [MB96].

In order to get better effect, we blend the shadow with the textured mapped ground. We use the alpha value α in the RGB color representation. The α is used to specify how the shadow color blends with the floor color, i.e., the colors of the pixels in the frame buffer remain the same as those without the shadowing but the intensity values are multiplied by the α and added to the frame buffer. One example of the result is shown in Figure 8 (b).

5 IMPLEMENTATION AND MORE RESULTS

We implemented our work on Pentium III 450MHz PC (128MB RAM and 4GB hard disk) using Microsoft Visual C++. The touring is in real time with the frame rate around 20fps. More results are shown in the APPENDIX A (Figure 9, 10, and 11). The original pictures were taken using a 35mm SLR camera and digitized using a scanner.

In our implementation, we also added the constraint for the viewpoint so that it will not be too close to the five rectangles (walls) and out of the scene.

6 CONCLUSION AND FUTURE WORK

We have proposed and implemented a Tour Into Multiresolution Picture, the TIMP as an extension of the TIP. Our results have shown the touring result is more natural with the quality of rendered images nearly unchanged when the viewer touring into the scene. By applying different resolution images based on the distance of the viewpoint to the scene, we can guarantee the rendering to have the pre-defined and acceptable sampling rate. We have also improved the frame rate for the rendering. Furthermore, we have integrated 3D models into the scene of the picture. The shadows were generated for each object from the estimation of light sources. Finally, the shadows were smoothly blended with the image contents to enhance the visual quality.

In future work, we will generalize the assumptions of TIP. For example, we will consider the picture with the rear wall not parallel to the view plane, the 3D floor not orthogonal to the view up vector, or more general, the vanishing point outside the picture.

7 ACKNOWLEDGMENTS

We thank Woon Tong Wing, Wu Yinghui, Wang Yapeng for the help of the implementation and Guo Jugui for the continuous support. The work is partly supported by the NUS scholarship and the NUS Academic Research Fund R-252-000-051-112.

REFERENCES

[AL99] D. G. Aliaga and A. Lastra. *Automatic Image Placement to Provide a Guaranteed Frame Rate.* SIGGRAPH'99, 307-316.

[BKT00] L. Boissieux, G. Kiss, and N. Magnenat Thalmann. *Simulation of Skin Aging and Wrinkles with Cosmetics Insight.* N. Magnenat Thalmann, D. Thalmann, and B. Arnaldi (eds.), Eurographics Workshop on Computer Animation and Simulation 2000, 15-28.

[Chan99] C. F. Chang, G. Bishop, and A. Lastra. *LDT Tree: A Hierarchical Representation for Image-Based Rendering.* SIGGRAPH'99, 291-298.

[CW93] S. E. Chen and L. Williams. *View Interpolation for Image Synthesis.* SIGGRAPH'93, 279-288.

[MB96] T. McReynolds and D. Blythe. *Programming with OpenGL: Advanced Rendering.* SIGGRAPH '96 Course Notes.

[Gort96] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. *The Lumigraph.* SIGGRAPH'96, 43-54.

[HAA97] Y. Horry, K. Anjyo, K. Arai. *Tour Into the Picture: Using a Spidery Mesh Interface to Make Animation from a Single Image.* SIGGRAPH'97, 225-232.

[Heck89] P. Heckbert. *Fundamentals of Texture Mapping and Image Warping.* Master's thesis, UCB/CSD 89/516, CS Division, U.C. Berkeley, June 1989, 86 pages.

[LH96] M. Levoy and P. Hanrahan. *Light Field Rendering.* SIGGRAPH'96, 31-42.

[MB95] L. McMillan and G. Bishop. *Plenoptic Modeling: An Image-Based Rendering System.* SIGGRAPH'95, 39-46.

[MDK99] N. Max, O. Deussen, and B. Keating. *Hierarchical Image-Based Rendering using Texture Mapping Hardware.* Eurographics Workshop on Rendering, 1999, 57-62.

[Rega99] M. J. P. Regan, G. S. P. Miller, S. M. Rubin, and C. Kogelnik. *A Real Time Low-latency Hardware Light-Field Renderer.* SIGGRAPH'99, 287-290.

[SD96] S. M. Seitz and C. R. Dyer. *View Morphing.* SIGGRAPH'96, 21-30.

[SH99] H. Y. Shum and L. W. He. *Rendering with Concentric Mosaic.* SIGGRAPH'99, 299-306.

[Will83] L. Williams. *Pyramidal Parametrics.* SIGGRAPH'83, 1-11.

APPENDIX A: MORE TOURING EXAMPLES

(a)

(b)

(c)

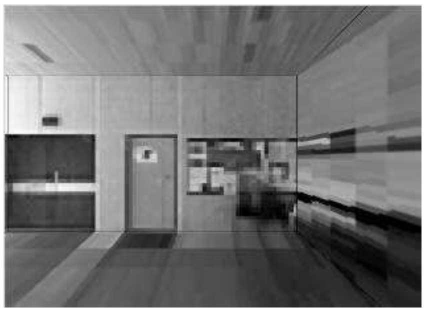
(d)

Figure 5: Four snapshots of TIP.

(a)

(b)

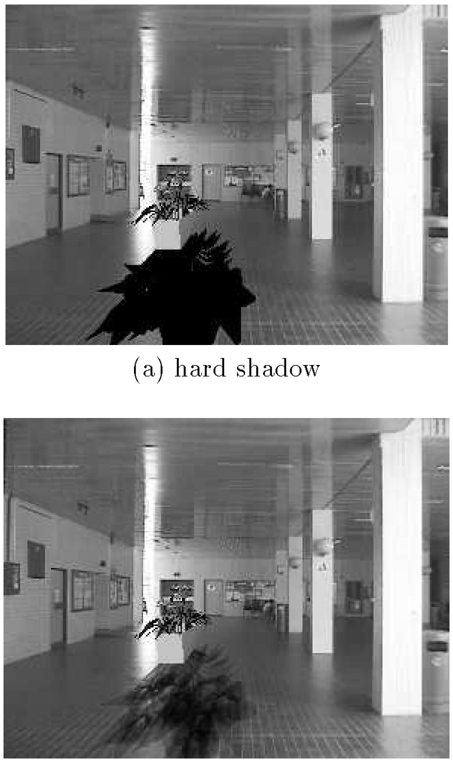
(c)



(d)

Figure 6: Four snapshots of TIMP with the same viewpoints as in Figure 5.

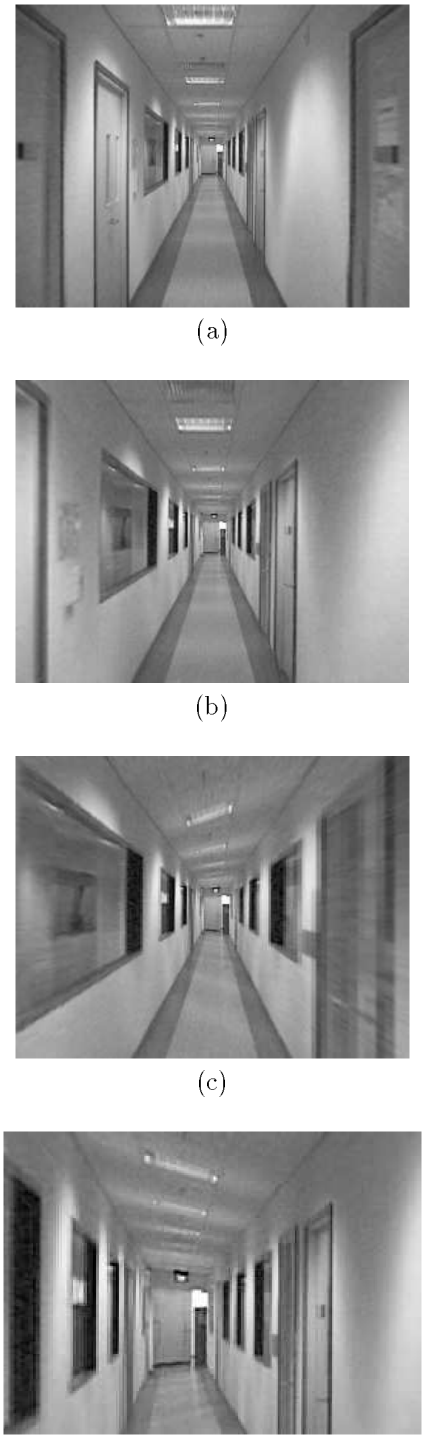
Figure 7: Integrate 3D models in the scene.



(a) hard shadow

(b) soft shadow

Figure 8: Integrate 3D models in the scene with shadow.



(a)

(b)

(c)

(d)

Figure 9: Tour into a corridor of an NUS building.

(a)

(b)

(c)

(d)

Figure 10: Tour into a corner of EPFL campus.

(a)

(b)

(c)

(d)

Figure 11: Tour over a river at Interlarken, Switzerland.