Multi-layered Image-Based Rendering

Sing Bing Kang
Compaq Computer Corporation
Cambridge Research Lab
One Kendall Square, Bldg. 700
Cambridge, MA 02139
sbk@crl.dec.com

Huong Quynh Dinh
College of Computing
Georgia Institute of Technology
Atlanta, GA 30332-0280
quynh@cc.gatech.edu

Abstract

In this article, we describe a multi-layered image-based rendering system that can use different types of input to produce and render new environments. Each separate input that can be manipulated independently is called a *layer*. In our implementation, the types of layers that can be manipulated are the *image-based* and *3-D-based* layers. The computation required for rendering the newly-crafted environment is reduced by using cached composite snapshots of that environment at reference poses. These cached snapshots are used to directly generate novel views, and the original layers are used only when necessary. Another key concept is the identification of types of holes generated as a result of pixel transfer from the composite snapshots to the generated view. For optimal rendering quality, the algorithm used in filling these holes is specific to the hole type (either *intralayer* or *interlayer*). The ideas embodied in our multi-layered IBR system are useful in augmenting the capabilities of applications that require fast and geometrically consistent rendering of 3-D scenes such as video editing.

Keywords: Image-based rendering, layered representation, trilinear/trifocal tensor

Introduction

3-D scene rendering is accomplished either through the conventional graphics pipeline of manipulating and projecting a 3-D model or using techniques that operate on a collection of input images. The latter, also known as image-based rendering (IBR), generally has the positive characteristics of having rendering speeds independent of scene complexity, non-reliance on 3-D graphics accelerators, moderate CPU costs, and excellent potential for producing realistic-looking output. Classifications of image-based rendering techniques are described in detail in [7].

The work described in this article falls into a category of image-based rendering techniques called *geometrically-valid pixel reprojection*, or transfer methods. This kind of method uses a small number of images to generate novel views based on feature correspondences across the images and applies computer vision techniques such as stereo, structure from motion, or projective recovery for the pixel transfer computation. Our proposed technique extends existing methods to allow the creation of new and geometrically-consistent environments from disparate sources. This is accomplished using a *layered* representation. In our representation, each layer is deemed to be different if it comes from a different source, or its pose can be independently controlled relative to all other layers.

Prior Work

The fundamental limitation of prior IBR methods is their assumption that images used in reprojection represent a static scene taken from multiple camera viewpoints. This is a reasonable assumption in visualizing models of a single rigid object or scene, but it is inadequate for applications such as video editing and synthesis, where there are likely to be multiple moving objects of interest. We overcome this limitation by introducing the notion of multiple layers of video into image-based rendering. We handle multiple rigid motions by assigning each one to a separate video layer. In our multi-layer IBR algorithm, each layer is reprojected separately, and the reprojected layers are combined to form the total image.

Layered video representations including correspondence information are described in [19] for video coding. However, this work does not describe a geometrically correct rendering algorithm and a means for editing the models or combining models from multiple sources to generate novel video sequences. Rather, it simply focuses on coding an existing sequence efficiently. Conventional video editing systems, such as the Avid system described in U.S. Patents 5,644,364 [8] and 5,654,737 [3], also employ a layered representation but do not include geometric information and cannot support geometrically-valid 3-D rendering.

In Baker and Szeliski's work [2], they refer to layers as disconnected image regions, each of which having discontinuous depths relative to all others. They describe a technique that, given the segmented regions in multiple

images, iteratively estimate the depth at each layer. In our work, we would consider the result of this "multi-layered" stereo recovery technique as being one layer or multiple layers, depending on whether they are treated as a whole or independently (respectively) in generating new environments.

In another related work, Shade *et al.* [16] described a rendering technique that uses in part a *layered-depth image*, or LDI, in which multiple depths may be encoded within a single pixel. The depth is *constant* within each layer. This is not necessarily so for our notion of a layer. Their technique extends McMillan and Bishop's painter algorithm [11] to transfer all depth levels within each pixel. In comparison, we cache a pair of views; new views are computed using these cached views first before filling in possible holes.

Organization

In this article, we describe the architectural concepts of our multi-layered IBR system in Section , which allows the inclusion of different types of inputs (i.e., layers) such as stills, video, and 3-D models. To set the stage for multi-layered IBR, we describe pixel transfer mechanism for a single layer in Section . Subsequently, Section details how the composite from the different inputs are generated and used to produce novel views. Experimental results using images of real and synthetic objects are shown in Section . We provide a discussion of the merits and disadvantages of our approach in Section , followed by concluding remarks in Section .

We now describe the concepts for the multi-layered IBR system. Note that not all the ideas have been implemented; parts that have been implemented are clearly delineated in Section .

The multi-layered IBR system

The general architecture for our proposed multi-layered image-based rendering system is depicted in the block diagram in Figure 1. It has four main parts: input, view selection, viewpoint synchronization, and novel view generation. The input to the system is a set of models which provide the sources for novel view generation. We have identified three types of models: a collection of still images of a static scene, a matted video sequence corresponding to a single coherent rigid body motion, and a conventional 3-D graphics model. There are two basic control inputs to the system: a reference index that specifies which part of each model should be rendered, and a virtual view that specifies the virtual camera which will be used to render and composite all of the models.

The first processing step in the block diagram is an intra-layer indexing stage, in which the reference indices and virtual views are used to select the correct part of

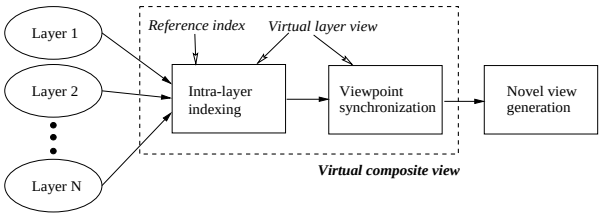


Figure 1: Architecture for multi-layer image-based rendering system.

each model. For example, for a collection of images, one can choose a region within the images or the image as the reference frame. For a 3-D model, one can choose a part or reference viewpoint. In rendering a video clip, the reference index would specify a desired frame while the virtual view specifies a change in the virtual camera position relative to the true camera position. Once the desired part of each model has been identified, each model is rendered into a set of pixel values in an output buffer.

The novel view is usually a perturbation of the reference viewpoints of the "background" layer, which we define as the layer that dominates the output by virtue of its relative size. These reference viewpoints are used as global references; hence, after all the reference indices have been identified, each layer is then transformed to be compatible with the "background" reference viewpoints and merged to create composite reference images. These cached images are then used to generate novel views using any pixel transfer technique.

A point of clarification is in order. For the case where only one image is used *and* the depth is known, then there is only one "reference frame." For the case where no ground truth is known, at least two images are used to create a novel view, assuming that full point correspondence across these images is known. In this case, these images are the "reference frames," and their respective viewpoints, "reference viewpoints."

Multi-layered model representation

In this section, we describe in more detail the three types of models corresponding to stills, video, and 3-D models. The expanded block diagram in Figure 2 shows each type along with a more detailed view of the processing steps. The first model type consists of a set of still images, not necessarily in any particular order, describing a static scene. This model is often used to describe a static background, which forms the backdrop for foreground objects, such as actors and actresses on a movie set. Along with the images are a set of correspondence maps, one for each image, that match pixels in that image to corresponding pixels in the other stills. These correspondences

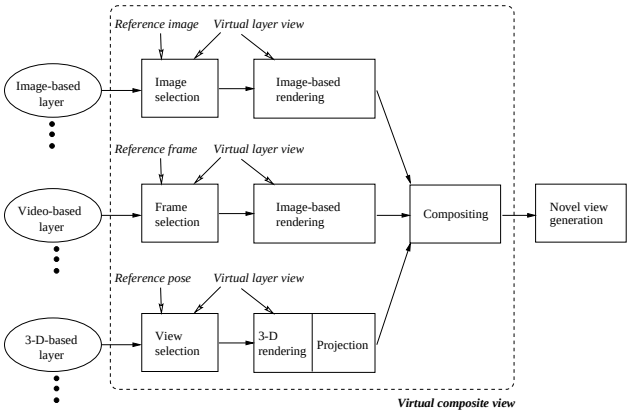


Figure 2: Detailed view of architecture for multi-layer image-based rendering system.

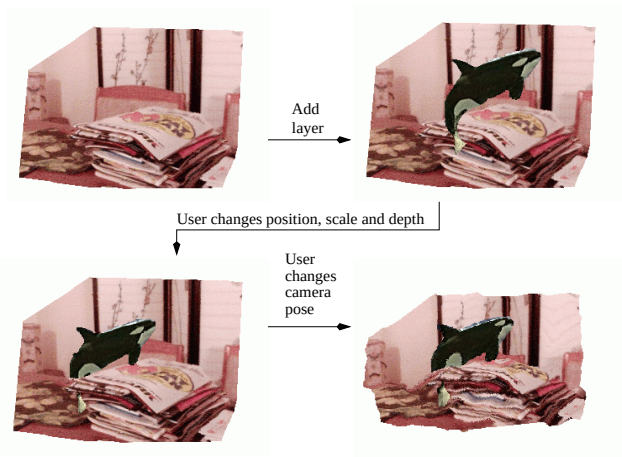


Figure 3: Example of background with one independent foreground object (whale figure). At the bottom right, the virtual camera has been tilted.

indirectly specify the relative geometry in the image data.

The second type of input data is a matted video sequence, or video layer. The matte is a mask which specifies the pixels in each frame of the video that are associated to the model. Each video layer describes a single coherent rigid body motion. A video sequence containing multiple moving objects would produce multiple models, each one containing a different matte sequence which selects a single object. As with the still image model, each video frame has an associated correspondence map which brings its pixels into correspondence with pixels in the previous video frame.

In addition to the pixel and correspondence data, each of the still and video models also contain a description of the pose, position, and intrinsic camera parameters for the camera for each image in the model. Intrinsic camera parameters include the focal length, aspect ratio, and image skew.

Figure 3 shows an example output image synthesized from two input models, a still model describing the background image of a stack of papers on a desk, and a whale model. Note that more complicated models for objects such as the human figure can be constructed from a collection of lower-level image models with associated kinematic constraints (e.g., [13]).

The third and final type of input layer is a conventional computer graphics model, consisting of a set of explicit 3-D surfaces (whose representation may be polygons, Non-Uniform Rational B-Splines (NURBS), etc.) with texture-mapped or shaded surfaces. The 3-D model may be a volumetric model as well. There is no explicitly stored correspondence information with this model, since correspondences can be generated automatically given two viewpoints of the 3-D model.

Intra-layer indexing

Each input layer is represented either by a set of image stills, a collection of video frames, or a 3-D model. In order to synthesize images from these layers, the user must first specify which part of each of the models to use. An image layer, for example, consists of a number of frames, any one of which could be used as reference for synthesis. The reference index, along with the desired virtual camera view, determine the specification.

In the case of a video input layer, the index specifies a particular reference frame from the sequence, around which new viewpoints can be synthesized (see Figure 2). Similarly, for a collection of image stills, the user can choose a reference image. In each of these cases the virtual camera input specifies a camera motion relative to the camera configuration for the reference image. In the 3-D model case, the index is the reference pose at which the 3-D model will be rendered using conventional 3-D graphics. The rendered 3-D model can then be processed in exactly the same manner as the other still-image-based and video-based layers.

Once the intra-layer indexing is done, the system then makes all the layers compatible by transforming all of them to a global reference frame (for our case, corresponding to the "background" layer). This allows the creation of composite reference images, from which new views can be generated using a pixel transfer technique.

Implemented version of multi-layer IBR system

The implemented version of our multi-layered IBR system is a subset of the general architecture described above. We have not implemented the analysis and manipulation of the video layer with indexable frames. Instead, we use

only image-based and 3-D-based layers. In addition, with regards to camera intrinsic parameters, we recover only the camera focal length from input images.

In our implemented system, each image-based layer is either an image with a predefined depth distribution, or is generated using two input images using a stereo technique. We create a new environment by interactively adding new layers to the first existing layer, which is considered the "background." Direct and indirect shape encoded in each layer is used to determine the appearance of the new environment at the same reference viewpoints as the background layer. These cached composite images are used to generate new viewpoints, and holes that may result are filled appropriately.

In summary, our approach comprises:

1. Generation of single layer from two images:
   (a) Register images (using spline-based registration [18])
   (b) Recover epipolar geometry and camera focal length
   (c) Compute trilinear tensor [1] and use it to transfer pixels for the creation of novel views
   (d) Fill holes (intralayer holes) through direct pixel interpolation [6]
2. Generation of composite from multiple layers:
   (a) Create a cached composite image pair from the input layers
   (b) Treat the cached composite image pair as a single layer in generating novel viewpoints
   (c) Identify types of holes: intralayer holes and interlayer holes
      - Fill intralayer holes through direct pixel interpolation [6]
      - Fill interlayer holes through limited search (forward or inverse mapping relative to screen space)

Image-based rendering for a single layer

There is a variety of ways that one can adopt to generate novel views from corresponded images. If the camera parameters are known, then depth images can be computed and used directly. Two other geometrically-valid reprojection methods use the epipolar constraints (in the form of the 3×3 *fundamental matrix*) between features across two images [9], and the trilinearities linking features across three images [1]. For comparisons between these last two methods, please see [7]. In our work, we use the trilinear tensor approach.

Image synthesis using the trilinear tensor

Our approach makes use of a standard approach to image-based rendering using what is termed trifocal tensors, or trilinear tensors. The trilinear tensor is a global $3 \times 3 \times 3$ entity that links correspondences across three images [17]. The trilinear tensor is of the form α_i^{jk}, with $i, j, k = 1, 2, 3$. Point correspondences across three images ($\mathbf{p}$, $\mathbf{p}'$, and $\mathbf{p}''$) are linked by the trilinear tensor in the following manner (using the Einstein summation convention):

$$p^i s_j^\mu r_k^\rho \alpha_i^{jk} = 0 \tag{1}$$

with $\mu, \rho = 1, 2$, s_j^1 and s_j^2 representing two lines intersecting at $\mathbf{p}'$, and r_k^1 and r_k^2 representing two lines intersecting at $\mathbf{p}''$. There are four separate constraints associated with (1), each constraint corresponding to a different combination of s_j^1 or s_j^2 and r_k^1 or r_k^2.

If a trilinear tensor is known for a set of three images, then given a pair of point correspondences in two of these images, a third corresponding point can be directly computed in the third image without resorting to any projection computation. This idea has been used to generate novel views from either two or three reference images [1].

The idea of generating novel views from two or three reference images is rather straightforward. First, the "reference" trilinear tensor is computed from the point correspondences between the reference images. In the case of only two reference images, one of the images is replicated and regarded as the "third" image. If the camera intrinsic parameters are known, then a new trilinear tensor can be computed from the known pose change with respect to the third camera location. The new view can subsequently be generated using the point correspondences from the first two images and the new trilinear tensor.

In our work, we recover the elements of the initial trilinear tensor and camera focal lengths using the 8-point algorithm [10] and an image-based metric that minimizes projection errors.

Resampling issues

A version of the painter's algorithm [4] is used to ensure that pixels are transferred in the correct order so that more distant parts of the scene are occluded by nearer parts of the same scene. This algorithm first computes the projection of the virtual camera center on the reference image. If the virtual camera center is in front of the camera center corresponding to the reference image, then scanning proceeds from the outer parts of the reference image towards the projected virtual camera center. Otherwise, scanning proceeds from the projected virtual camera center away from it. This variant of the painter's algorithm was originally described in [12]. As an alternative, a standard Z buffer approach could also be used.

In the single layer case, the final step of image-based rendering is pixel interpolation to remove any holes that may have been left after image transfer. Holes arise when a small preimage patch from the reference image is mapped onto a larger patch on the target image. Many interpolation techniques exist; a simple but effective method is Elliptical Weighted Averaging [6] which employs interpolant kernels with adaptive sizes. Note that there is no separate compositing step in the single layer case.

Compositing for multi-layered IBR

The primary difference between single and multiple layer IBR is the need for a compositing stage which combines the image transfer outputs from a set of layers into a single image. The image transfer methods described in the previous section can be applied independently to each input layer, since each layer describes a rigid body motion. There are two possible approaches to compositing. First, if the depth were available at each output pixel from each layer, standard Z buffering could be employed. Second, the painter's algorithm for the single layer described above can be extended to the multiple layer case.

The painter's algorithm has a potential advantage over the Z buffer approach in that it does not require a depth comparison at each pixel transfer and may not require as much storage. The painter's algorithm is complicated by two factors in the multi-layer case: layers must be drawn in depth order and missing pixels must be filled in. In the case where there are multiple overlapping layers, care must be taken to paint the "farthest" layers first, so pixels from the closer layers will be painted last. This presumes the existence of a drawing order for the layers; it may be necessary to split some layers into multiple parts so that they can be ordered (see [15, 4] for details). In particular, it is possible to use a modified version of the Binary Space Partition (BSP) tree representation described in [5] to produce ordered layers from arbitrary virtual camera views. We did not implement the BSP algorithm, since we do not wish to limit our representations to be piecewise planar, as assumed by the algorithm.

The multi-layered IBR algorithm hinges on the generation of reference, or cached, images from which new viewpoints can be generated. In the case of multiple layers, we first create cached reference composite images using depth ordering. The user can interactively modify the size, orientation, and position of each layer independently. Once this has been done, novel viewpoints of the new environment can be generated from these cached reference composite images. In other words, the composite reference images are treated as a *single new layer*. The difference is that holes caused by object disocclusion can be filled by referring back to the corresponding layers.

The alternative is to transfer each layer separately in generating new views. However, it may be inefficient in cases where most of the layer regions are already exposed in the composite reference (i.e., cached) image, and there are only a few and relatively small regions in which occlusion occur. Here, a complete rendering of all of the other layers is undesirable, since most of their pixels will be overwritten. We can use an alternative algorithm in which the pixel transfer of the cached composite image is followed by a hole filling stage (if required) in which any remaining unspecified pixel values are rendered.

As in the single layer case, we must also employ interpolation to fill holes due to sampling that result from mapping smaller patches to larger screen areas. We term this kind of holes *intralayer holes*. In the multiple layer case, however, holes (termed *interlayer holes*) may occur as a result of layer disocclusion or exposure. Interlayer holes should not be filled in the exact same manner as intralayer holes. Doing so would cause textures from different layers to be blended by interpolation, resulting in an undesirable mixing of textures. In applying the approach of cached composite transfer followed by gap filling, we must distinguish between the intralayer and interlayer holes. Intralayer holes are identified by those that are surrounded by pixels from the same layer. Interlayer holes are surrounded by pixels from different layers.

Filling intralayer holes

As mentioned before, intralayer holes occur as a result of mapping a smaller patch in a reference view to a larger patch in the virtual (screen) view. Such holes can be removed through any standard interpolation technique, but our choice is to use the Elliptical Weighted Average (EWA) filter [6].

Filling interlayer holes

We have considered two methods for the interlayer hole filling step, namely, forward mapping from the layers to screen space, and inverse mapping from screen space to the layers.

Forward mapping

In the forward mapping process, only regions of each layer that are not part of the cached composite image are involved. The general idea is to compute, for each unexposed pixel in each layer, its new location corresponding to the new camera viewpoint. If a pixel is mapped to an interlayer hole, its depth is computed and stored. Once this is done for all the unexposed pixels, depth ordering is then used to determine the right pixel to expose. Note that depth computation and comparison are necessary *only* at the interlayer hole locations.

Inverse mapping

This alternative method is the inverse of the forward image transfer, as it is a kind of backprojection from a desired pixel in the output image to a layer in the model. Out of the possibly many pixel candidates in all the layers, epipolar search that is based on camera geometry is used to narrow the search and select the correct one. The search process goes as follows: Suppose the relative camera poses for two images are known or computed. Then for a given pixel in one image, the corresponding pixel in the other image is constrained to lie on a line called the *epipolar line*. Using this fact, we can then find the missing pixel by search along the epipolar lines and checking if the pixel in question is also a corresponding point. In the event of multiple candidates across different layers, depth sorting is performed to choose the frontmost pixel to transfer.

Comparison between the two hole-filling approaches

Unsurprisingly, the inverse mapping approach turned out to be highly prone to errors in both camera geometry and point correspondences, since it relies on search along epipolar lines and the correspondence, or disparity, map. In cases where real images are used and no correct ground truth is known, the challenge of recovering both accurate point correspondences and camera geometry is very difficult, as known in the computer vision community.

As a result, there is a tendency for this technique to either blur edges or produce an offset of texture, as exemplified by Figure 4(c)). In addition, it is also slower in comparison to the forward mapping approach. This is due to computation associated with the epipolar search. In the example shown in Figure 4, the rendering rate for the inverse mapping approach for hole filling is 0.9 frames/sec, compared to 2.5 frames/sec for the forward mapping alternative. The size of the image used is 240×320 and the platform on which the program was run is an Alpha PC with an operating frequency of 533 MHz.

Other features of implementation

In addition to being able to input fully-corresponded image pairs, the image-based layer can also be specified as a single masked image with user-specified depths. In our implemented version, its depth can be specified as flat, the same depth variation as the background layer, or having a global "bump." By global "bump", we mean a depth distribution whose relative depth is inversely proportional to the distance of the pixel to the boundary.

Once added into the environment, the input layer can be interactively scaled, rotated, and translated relative to the other layers.

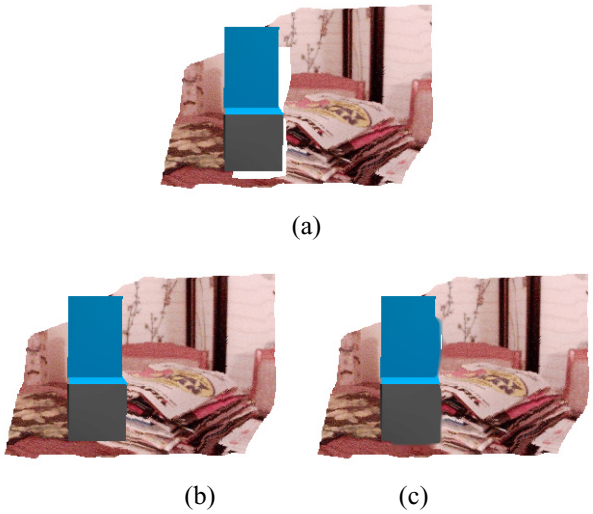


Figure 4: Example of interlayer hole-filling: (a) Result of forward pixel transfer of cached composite image corresponding to a new view with no hole filling, (b) Result of forward mapping, (c) Result of inverse mapping.

Results

We have run a series of experiments involving various image-based and 3-D-based layers, and in this section, show a couple of sets of results. The first set uses an image pair of an arch in a museum as the background image-based layer; the other two layers, the buddha and molecule, are 3-D-based layers. Snapshots of these layers, with both changing camera viewpoints and layer poses, are shown in Figure 5. The original image size of the image-based layer is 320×240.

Figure 6 shows another set of results, this time using only artificially created cutouts and user-defined depths. As expected, the image quality is higher using layers with exactly known depths and predefined camera motion as in Figure 5.

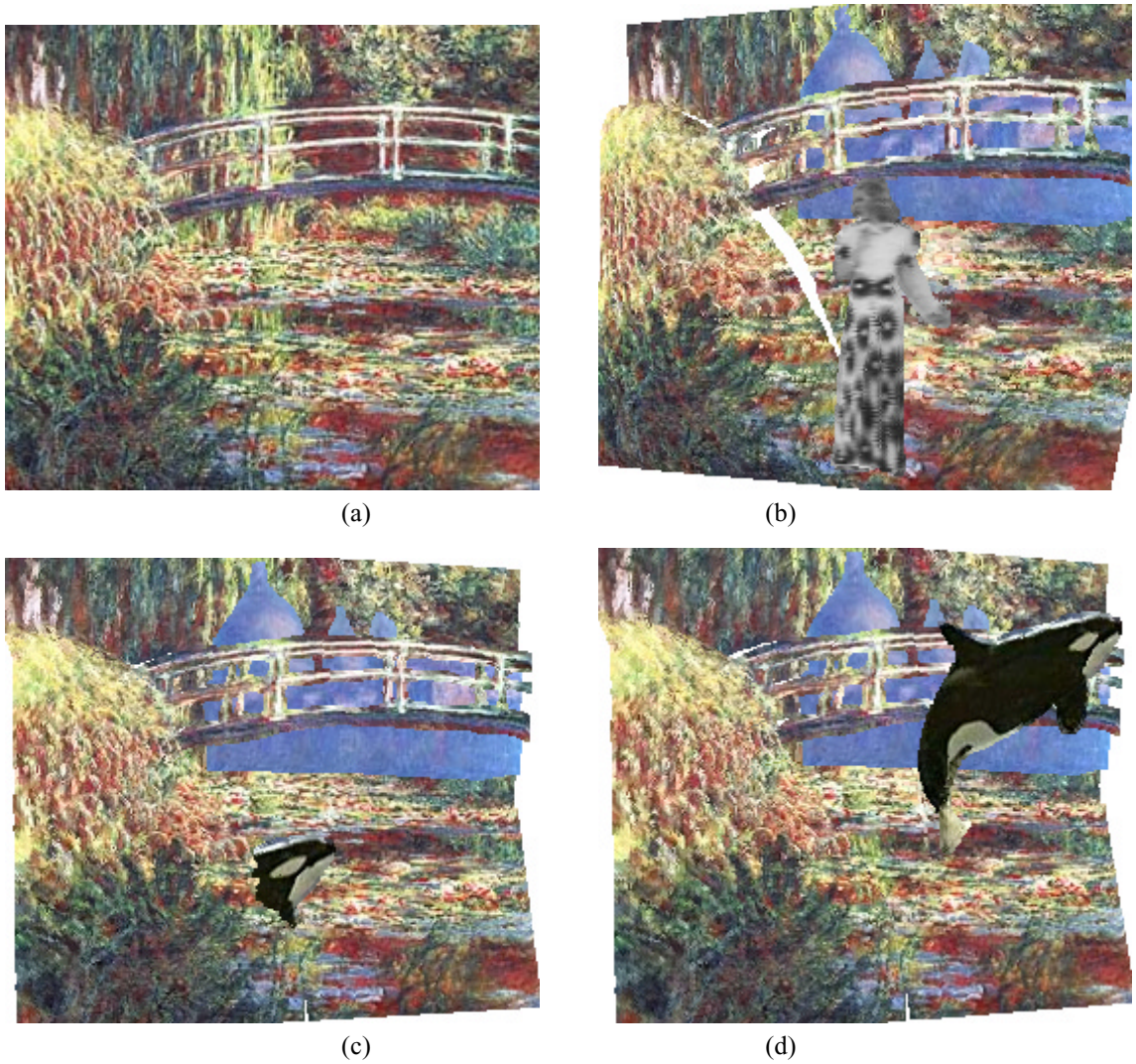
In another experiment, we used a total of 12 layers (see Figure 7). With the proposed cached composite method, we get a rendering speed of 0.30 secs per frame, while with the direct layer by layer approach, we get 0.35 secs per frame. These numbers are based on 100 runs. The current implementation of the cached composite method is not well optimized, which in part explains the relatively small improvement in performance.

Discussion

The true challenge in IBR is using real images with no ground truth, and it is no different in our multi-layered IBR system. Accurate correspondence is the key to providing realism in generating new viewpoints. It is known

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

Figure 5: Example of multi-layered IBR: (a) Background image-based layer only, (b) Background layer with buddha layer, (c) Same as (b), but a slight rotation of the buddha layer, (d) Change in viewpoint (slight change in camera tilt) with buddha layer moved and a molecule layer added (and moved to behind the arch). The background in (c) "wriggles" because of inexact correspondences between the real images of the arch. This in turn is caused by lack of texture in certain places. The inexact correspondence has the same effect as misestimating depth.



(a) (b)

(c) (d)

Figure 6: Another example of multi-layered IBR: (a) Background image-based layer only (from a Monet painting), (b) Addition of "Ginger Rogers" layer (at a different viewpoint), (c,d) Willy the whale jumping out of Monet's bush (at another slightly different viewpoint). Note that the image-based layer of the Monet painting is just a single layer with depth, and the gaps observed are due to depth discontinuities within that layer.

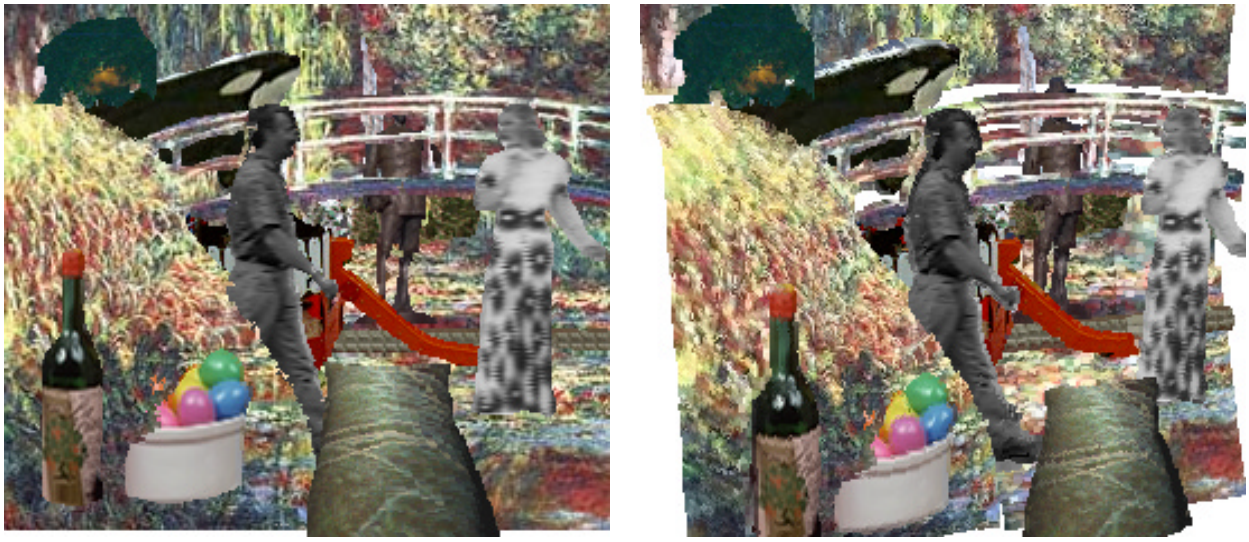


Figure 7: Another experiment with 12 different layers: (Left) Original view, (Right) A side view.

within the computer vision community that extracting correspondence between images is difficult. The problems of object occlusion, lack of texture, photometric variation, and camera inperfections, which occur in practice, pose formidable barriers to correct image registration and recovery of camera geometry.

There is also the known problem of choosing representative viewpoints as reference in the image-based layer. If we choose images corresponding to significant camera motion (i.e., large baseline), image registration is made much harder due to significant appearance changes. However, assuming that correspondence is accurate, camera geometry and scene structure can be extracted more reliably. On the other hand, images taken at small baselines result in much easier image registration, but less robust recovery of camera geometry and scene structure. Obviously, a compromise is to use many more images taken at successively reasonable baselines (but at a proportionally higher computational expense). This technique is also known as multibaseline stereo [14].

One of the limitations of our current multi-layered IBR approach is related to caching the composite reference images. For layers other than the "background" layer, their transferred pixels at the composite reference images is a result of resampling of the original sources. This resampling step is necessary in order to convert their current frames to a common frame, namely that of the "background" layer. Generating new viewpoints using these cached composite reference images involves another resampling step for these pixels, resulting in further possible compromise in visual quality. This is the price paid for facilitating rendering of multiple layers.

As future work, we would extend our present implementation to be able to accept and manipulate the frame-indexable video-based layer.

Conclusions

We have described our vision for a multi-layered IBR system that allows disparate image and 3-D sources (or layers) to be merged and manipulated to produce new environments. In this system, the novel viewpoints of these new environments can be produced in a geometrically consistent manner through the use of both computer vision and graphics techiques. We have identified three types of layers: image-based, 3-D-based, and video-based. Our implementation to illustrate our ideas currently can input and manipulate image-based and 3-D-based layers.

Central to our multi-layered IBR system is the notion of using cached composite reference images to generate novel views of the new environment. This reduces the computational cost of rendering by reusing the composite reference images and reverting to the original layers only when necessary. Critical to the quality of the output is the identification of the types of holes created during pixel transfer from the composite reference images, namely *intralayer* and *interlayer* holes. A different type of hole dictates the use of a different hole-filling algorithm for optimal view synthesis.

Such a system can be used in any applications that require visualization of 3-D environments, such as video editing and synthesis for the entertainment industry.

We have not yet considered lighting effects (such as shadows) subsequent to the addition of multiple layers. It is not immediately clear if such effects can be incor-

porated without extensive modification of our proposed framework. This would be an interesting topic for future work.

*

Acknowledgment

We appreciate the initial fruitful discussions with Jim Rehg on the multi-layer image-based rendering architecture that helped in the development of some of the ideas implemented in this work.

References

[1] S. Avidan and A. Shashua. Novel view synthesis in tensor space. In *Conference on Computer Vision and Pattern Recognition*, pages 1034–1040, San Juan, Puerto Rico, June 1997.

[2] S. Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *Conference on Computer Vision and Pattern Recognition*, pages 434–441, Santa Barbara, CA, June 1998.

[3] Harry Der, Barry Horne, and Jeffrey Kurtze. Media pipeline with mechanism for real-time addition of digital video e ffects. U.S. Patent 5,654,737, August 1997.

[4] J. Foley, J. van Dam, S. Feiner, and J. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, 1990.

[5] H. Fuchs, Z. M. Kedem, and B. F. Naylor. On visible surface generation by a priori tree structures. *Computer Graphics (SIGGRAPH'80)*, pages 124–133, 1980.

[6] N. Greene and P. Heckbert. Creating raster Omnimax images from multiple perspective views using the Elliptical Weighted Average filter. *IEEE Computer Graphics and Applications*, pages 21–27, June 1986.

[7] S. B. Kang. A survey of image-based rendering techniques. Technical Report CRL 97/4, Cambridge Research Lab., Digital Equipment Corp., August 1997.

[8] Jeffrey Kurtze, Ray Cacciatore, Peter Zawojski, Eric C. Peter s, and John Walsh Jr. Media pipeline with multichannel video processing and playback. U.S. Patent 5,644,364, July 1997.

[9] S. Laveau and O. Faugeras. 3-D scene representation as a collection of images and fundamental matrices. Technical Report 2205, INRIA-Sophia Antipolis, February 1994.

[10] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[11] L. McMillan and G. Bishop. Head-tracked stereoscopic display using image warping. In *SPIE Symposium on Electronic Imaging Science*, San Jose, CA, February 1995.

[12] L. McMillan and G. Bishop. Plenoptic modeling: An image-based rendering system. *Computer Graphics (SIGGRAPH'95)*, pages 39–46, August 1995.

[13] D. D. Morris and J. M. Rehg. Singularity analysis for articulated object tracking. In *Conference on Computer Vision and Pattern Recognition*, pages 289–296, Santa Barbara, CA, June 1998.

[14] M. Okutomi and T. Kanade. A multiple baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363, April 1993.

[15] J. M. Rehg and T. Kanade. Model-based tracking of self-occluding articulated objects. In *Proc. of Fifth Intl. Conf. on Computer Vision*, pages 612–617, Boston, MA, 1995.

[16] J. Shade, S. Gortler, L.-W. He, and R. Szeliski. Layered depth images. *Computer Graphics (SIGGRAPH'98)*, pages 231–242, July 1998.

[17] A. Shashua. Algebraic functions for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):779–789, 1995.

[18] R. Szeliski and J. Coughlan. Hierarchical spline-based image registration. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 194–201, Seattle, Washington, June 1994. IEEE Computer Society.

[19] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(5):625–638, September 1994.