Camera Field Rendering

Minglun Gong and Yee-Hong Yang
Department of Computer Science, University of Alberta, Edmonton, Alberta, Canada

May 23, 2002

Abstract

Previous image-based rendering approaches rely on either accurate three-dimensional geometric information or dense image samples. It is, therefore, hard to render a complex real scene due to the difficulties of the following two problems: (1) how to model and represent a complex real scene using three-dimensional geometry; and (2) how to sample a real scene densely and to compress the samples. To avoid these problems, we propose two novel interpolation techniques, which can produce reasonable rendering results for sparsely sampled real scene. The first one, which is a color-matching based interpolation, searches for a possible physical point along the testing ray using color information in nearby reference images. The second technique, which is a disparity-matching based interpolation, tries to find the closest intersection between the testing ray and the disparity surfaces defined by the nearby reference images. Both approaches are designed as backward rendering techniques and they can be combined to produce robust results. In addition, a new sampling scheme called camera field is used. A camera field can be directly acquired using a two-dimensional array of cameras mounted on the corresponding support surface. Since no resampling process is required, we can fully utilize the resolution of the cameras. More importantly, new parameterization scheme allows us to find the corresponding point among reference images directly. This makes it possible to implement the proposed two interpolation approaches in the image space, rather than in the three-dimensional world space.

1 Introduction

Image-based rendering approaches, which have been advocated as an alternative to geometry-based approaches, have gained in popularity in the last several years. Many algorithms have been proposed in this area. Some of them rely on accurate geometric information on the scene but use a few images only[4, 12, 17, 15, 16]. Others require densely sampled images but no prior knowledge of geometrical information of the scene[11, 7, 25, 8].

1.1 Motivations

Even with advanced techniques, accurate geometric information of a real scene is often hard to acquire. This makes approaches in the latter category attractive. However, these approaches normally require very dense sampling of the scene; otherwise, the generated image will be blurry. Although various compression techniques could be applied to make the data more manageable, to obtain dense samples itself is a difficult problem for a real scene.

It is well known that fewer samples are needed to generate acceptable results if we know the geometry, even an approximate geometry, of the scene. Chai et al. analyze the minimum sampling rate required using the light field parameterization scheme[2]. Their results show that the more accurate geometrical information we have, the fewer samples are needed to generate result of a given effective resolution.

Therefore, our first motivation is to propose a rendering technique to render sparsely sampled scene with or without accurate geometrical information. The first technique, which is a color-matching based interpolation, uses depth information implicitly by searching for a possible physical point along the epipolar lines in nearby reference images. The second approach, which is a disparity-matching based interpolation, explicitly use the pre-computed disparity maps and search for the intersection between the testing ray and the disparity surfaces defined by nearby reference images.

In addition, many image-based rendering techniques use specially designed ray parameterization scheme, such as the light slab[11, 7], the concentric mosaics[25, 5], and the image-based object[15]. These parameterization schemes do not fit with the images captured using multiple cameras. A resampling process is needed to project the captured images into the varying sampling schemes. This reprojection process will, of course, reduce the effective resolution of the captured image.

The solution to the above problem forms our second motivation of the proposed approach. The parameterization scheme used in this paper is analogous to a two-dimensional array of pinhole cameras attached on a support surface. Therefore it naturally fits with images acquired by multiple cameras. Different kinds of support surfaces, such as planes or parametric surfaces, can be used with the same parameterization scheme. Which kind of surface can sample the scene more efficiently depends on the applications.

1.2 Related work

Our camera field rendering approach has some similarities with work reported in ref[8, 1] in that we all try to use images captured by cameras directly without resampling. In dynamically reparameterized light field[8], the scenes are sampled using a two-dimensional array of pinhole cameras attached to a planar surface. Hence, it is equivalent to the camera field that is defined on a planar surface. Our parameterization scheme is a more general scheme since we can also define camera field on other surfaces such as a sphere and a cylinder. In addition, whether dynamically reparameterized light field can produce sharp image depends on if the focal surface defined by the user is close enough to the geometry of the scene. This feature gives the user flexibilities to adjust the focus of the rendering result. On the other hand, it makes it difficult to use if the user just wants sharp rendering results of the scene. To solve this problem, our approach tries to automatically adjust the focus for different parts of the scene using either color consistency information or pre-calculated disparity information.

The rendering approach of unstructured lumigraph[1] is similar to that of dynamically reparameterized light field. As improvements, it uses three-dimensional geometric proxies provided by human interactions, which can more accurately represent the scene than focal surface does; and it also allows images to be taken at arbitrary camera positions. As consequences, the rendering process has to be performed in the three-dimensional world space. However, it is still difficult to use this approach to render sparsely sampled complex scenes since the three-dimensional proxies need to be accurate enough to avoid blurry rendering result. Our approach uses a more restricted parameterization scheme than that is used in unstructured lumigraph. The restrictions we put on cameras' positions make it possible to find the corresponding points among different views without going back to the third dimension. Therefore, no matrix computations are involved and the rendering can be implemented efficiently in the image space. Since our approach can make use of inaccurate disparity maps generated using stereovision techniques, we can produce reasonable results for sparsely sampled complex scene, without human interaction.

Our color-matching based interpolation approach searches for color consistent points. Therefore, it has similarities with existing image-based modeling techniques[24, 9]. However, we use color constancy as the sole means of rendering. Since we make no attempts to reconstruct the three-dimensional geometry of the scene, we can avoid the problems of representing and rendering the scenes.

Several techniques have been proposed to render sparsely sampled scenes with the help of depth information[22, 23]. Hence, they are similar to our disparity-matching based interpolation in that we all try to render multiple depth/disparity images. However, as forward rendering techniques, these approaches need to warp multiple images and composite the results together using Z-buffer. The rendering results in ref[22] show that holes exist under certain camera position, which is a common problem in forward mapping approaches. In contrast, our interpolation technique is a backward rendering approach, and therefore, we need not use Z-buffer and to face the problem of filling in holes. In addition, these approaches seem to rely on accurate depth information since all depth information are used no matter it is locally smooth or noisy. No results for real scene is given in ref[22], and the preliminary results in ref[23] show that the faulty depth information reduces the rendering quality considerably. Furthermore, the backward rendering process we used also has the merit that it can be easily parallelized and distributed on several computers since the rendering processes for different pixels are totally independent.

Our disparity-matching based interpolation also defers from existing backward rendering techniques[10, 21] in that it searches within multiple disparity images simultaneously to find the closest intersection. In addition, comparing with the backwards mapping approach proposed by Laveau and Faugeras[10], we define the disparity map as an independent property of an image, rather than as the mapping relation between two reference images. Therefore, instead of mapping the first epipolar line into the second image and calculating its intersections with the second epipolar line, we propose a much simpler searching algorithm to find intersections independently for every nearby reference image. Furthermore, the backward displacement map rendering technique[21] can only be applied to a restricted family of images, in which "depth differences in adjacent pixels are always meant to represent a surface slope, and therefore, must be treated as being connected"[21]. As shown in their results, rubber sheet effects will appear when this restriction is violated. In contrast, our approach can handle self-occluded surface or multiple surfaces without the undesired rubber sheet effects. Hence, it can be applied to render much more complex scenes. However, in order to search within multiple disparity images simultaneously, the price we paid is that we cannot make use of the coherence as suggested by Schaufler and Priglinger[21].

The organization of this paper is as follows. In the next section, the proposed camera field rendering approach is discussed. Section 3 presents the experimental results. The paper concludes in Section 4 with discussions on future work.

2 Camera Field Rendering

We name our approach as camera field rendering because it uses a two-dimensional array of cameras to sample the scene. The parameter scheme for the camera field is discussed in section 2.1. Two new interpolation techniques are introduced in section 2.2 and 2.3.

2.1 Parameterization Scheme

The light field approach uses two planes to sample the four-dimensional ray space. Under the rayset framework[5], the sampling scheme used by light field can be defined by the following support function:

$$\begin{aligned}
\mathbf{S}_x(u,v,s,t) &= \mathbf{F}_x(u,v) \\
\mathbf{S}_y(u,v,s,t) &= \mathbf{F}_y(u,v) \\
\mathbf{S}_z(u,v,s,t) &= \mathbf{F}_z(u,v) \\
\mathbf{S}_\theta(u,v,s,t) &= \Theta(\mathbf{G}(s,t)-\mathbf{F}(u,v)) \\
\mathbf{S}_\phi(u,v,s,t) &= \Phi(\mathbf{G}(s,t)-\mathbf{F}(u,v))
\end{aligned} \tag{1}$$

where, $\mathbf{F}(x,y)$ and $\mathbf{G}(x,y)$ are the parametric equations of the UV and the ST plane, respectively. Θ and Φ are functions that calculate the θ and ϕ angles of a vector, which are defined as:

$$\Theta(\mathbf{v}) = \tan^{-1}(\frac{\mathbf{v}_y}{\mathbf{v}_x}) \quad \Phi(\mathbf{v}) = \sin^{-1}(\frac{\mathbf{v}_z}{\|\mathbf{v}\|})$$

In contrast, the camera field we propose is defined on a surface, which we call the support surface. Under the rayset framework, the parameterization scheme can be defined by the following support function:

$$\begin{aligned}
&\mathbf{S}_x(u,v,s,t) = \mathbf{F}_x(u,v) \\
&\mathbf{S}_y(u,v,s,t) = \mathbf{F}_y(u,v) \\
&\mathbf{S}_z(u,v,s,t) = \mathbf{F}_z(u,v) \\
&\mathbf{S}_\theta(u,v,s,t) = \Theta(f\mathbf{n}(u,v) + s\frac{\partial \mathbf{F}(p,q)}{\partial p}|_{u,v} + t\frac{\partial \mathbf{F}(p,q)}{\partial q}|_{u,v}) \\
&\mathbf{S}_\phi(u,v,s,t) = \Phi(f\mathbf{n}(u,v) + s\frac{\partial \mathbf{F}(p,q)}{\partial p}|_{u,v} + t\frac{\partial \mathbf{F}(p,q)}{\partial q}|_{u,v})
\end{aligned} \tag{2}$$

where, $\mathbf{F}(x,y)$ is the parametric equation of the support surface, $\mathbf{n}(x,y)$ the normal of the support surface at (x,y), and f the focal length of the camera.

Obviously, to sample such a rayset, we can mount a two-dimensional array of pinhole cameras on the support surface, and adjust the cameras so that the viewing directions are along the normals of the support surface. When the cameras are aligned accurately enough, the images captured under such a setting will form a camera field directly. Otherwise, a rectification process may be required.

Depending on the scene or object to be sampled, different kinds of support surfaces can be selected, e.g. planes, parametric surfaces, and free-form surfaces. In particular, if a plane is used, then we have a planar camera field. The above support function degenerates into:

$$\begin{aligned}
\mathbf{S}_x(u,v,s,t) &= \mathbf{F}_x(u,v) \\
\mathbf{S}_y(u,v,s,t) &= \mathbf{F}_y(u,v) \\
\mathbf{S}_z(u,v,s,t) &= \mathbf{F}_z(u,v) \\
\mathbf{S}_\theta(u,v,s,t) &= \Theta(f \cdot \mathbf{n} + \mathbf{F}(s,t) - \mathbf{F}(0,0)) \\
\mathbf{S}_\phi(u,v,s,t) &= \Phi(f \cdot \mathbf{n} + \mathbf{F}(s,t) - \mathbf{F}(0,0))
\end{aligned} \tag{3}$$

where, $\mathbf{F}(x,y)$ is the parametric equation of the plane, $\mathbf{n}$ the normal of the plane.

Different parametric surfaces, such as cylinder, sphere, and cone, can also be used. In this paper, a cylinder is used as an illustration. We also believe that the proposed method can be applied to other parametric surfaces as well. For a cylindrical camera field, the support function 3 can be simplified to:

$$\begin{aligned}
\mathbf{S}_x(u,v,s,t) &= \mathbf{C}_x + R \cdot \sin(u) \\
\mathbf{S}_y(u,v,s,t) &= \mathbf{C}_y + R \cdot \cos(u) \\
\mathbf{S}_z(u,v,s,t) &= \mathbf{C}_z + v \\
\mathbf{S}_\theta(u,v,s,t) &= u + \tan^{-1}(\frac{s}{f}) + k\pi \\
\mathbf{S}_\phi(u,v,s,t) &= \tan^{-1}(\frac{t}{f})
\end{aligned} \tag{4}$$

where $\mathbf{C}$ is the center of the cylindrical camera field, R the radius of the cylinder. When k is equal to 0, the cameras face outward and sample the environment. When k is equal to 1, the cameras face the inside of the cylinder and sample the object and the environment behind the object.

It is noteworthy that the cylindrical camera field is different from the cylindrical panorama[3, 13] that has been used for a long time in the image-based rendering area. The cylindrical panorama defines the image plane on a cylinder and is a two-dimensional rayset, while the cylindrical camera field defines the centers of the projections on a cylinder and is a four-dimensional rayset.

2.2 Color-Matching Based Interpolation

In the first step, we try to interpolate the sparse views without explicit knowledge of depth information. The corresponding approach, which searches for a physical point in the scene along the testing ray, is discussed in the rest of this section.

First we illustrate the sampling problem using a planar camera field. Figure 1 shows a cross-section of a planar camera field. Assume that two cameras are set up at location C_u and C_{u+1} with their viewing directions perpendicular to the support plane (UV). A novel ray, Cm, intersects the

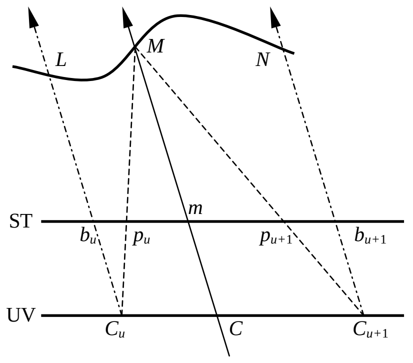


Figure 1: Interpolation between reference images for planar camera field.

support plane at C and the image plane (ST) at m. Using linear interpolation, the illumination at m is given by:

$$I = \frac{C_uC}{C_uC_{u+1}} I_u(b_u) + \frac{CC_{u+1}}{C_uC_{u+1}} I_{u+1}(b_{u+1})$$

where $I_u(x)$ denotes the intensity of pixel x in image u.

Obviously, if the support plane is not sampled densely enough, the interpolation result will be blurry because it is obtained by interpolating between different points, in particular, point L and N, in the scene. As shown in Figure 1, the intersection between ray Cm and the object is point M, whose projections on image u and image $u+1$ are p_u and p_{u+1}, respectively. Hence, a better result will be obtained if we can interpolate between these two pixels, i.e.:

$$I = \frac{C_uC}{C_uC_{u+1}} I_u(p_u) + \frac{CC_{u+1}}{C_uC_{u+1}} I_{u+1}(p_{u+1})$$

Now the question is how to determine the locations of pixel p_u and p_{u+1}, which are the projections of the same point, M, in the scene. In the color-matching based approach, this is done by searching physical points in the scene. First of all, we need to make the following two assumptions:

- Any point in the scene that is visible from the novel viewpoint is also visible in the nearby four cameras.
- The projections of the same physical point in the scene should have a higher level of color consistency than the projections from different physical points.

If the above assumptions hold, we know that along the testing ray, the projection of the intersection should have the highest level of consistency in color, i.e. have the smallest dissimilarity[1]. Hence, what we need to do is to project the points on the testing ray to nearby reference images. The point, whose projections give the smallest dissimilarity, will be the intersection that we look for.

[1]In this paper, the dissimilarity between two pixels is defined as the Euclidean distance between the intensities of the pixels in the RGB color space.

The above assumptions are not always hold in the real world. The first assumption is invalid around the occluding boundaries in the scene. The second assumption will fail for regions without textures. However, the result, shown in Figure 11(b), indicates that our approach is not sensitive to the violation of the first assumption since it is very likely that the overall dissimilarity for the physical point is still the smallest even if the point is occluded in one or two nearby cameras. The result, shown in Figure 12(b), indicates that our approach is not sensitive to textureless area either, such as the walls of the building, since it can produce the correct color even though it may wrongfully predict the location of the physical point.

Projecting three-dimensional points to an image involves a lot of computations. However, based on the properties of the camera field parameterization scheme and the epipolar constraint, we can use an efficient search algorithm, which works in two-dimensional image space. The general searching algorithm is the same even though the equations used depends on the format of the support surface. In the following, we discuss the case for planar and cylindrical camera field first, and then give the pseudocode for the general searching algorithm.

As shown in Figure 1, we know that the vanishing point for ray Cm on reference image C_u is b_u. Without loss of generality, we can assume that the intersection is in front of the focal plane. This is definitely true when we are using a real camera since all the real objects are in front of the camera. For synthetic scenes, however, we need to place the image plane carefully to make sure that no virtual object is in between the image plane and the center of projection.

Under the above assumption, we can limit the search within the interval $b_u m$ for reference image u. In addition, based on similar triangles shown in Figure 1, we know that:

$$\begin{aligned}\frac{Cm}{CM} &= 1 - \frac{mM}{CM} = 1 - \frac{p_u m}{C_u C} = \frac{b_u p_u}{C_u C} \\ \frac{Cm}{CM} &= 1 - \frac{mM}{CM} = 1 - \frac{p_{u+1} m}{C_{u+1} C} = \frac{b_{u+1} p_{u+1}}{CC_{u+1}}\end{aligned}$$

Now we define function $E_u(p_u)$ as:

$$E_u(p_u) = \frac{Cm}{CM} = \frac{b_u p_u}{C_u C} \tag{5}$$

Therefore, if different pixels p_u from different reference images u are the projections of the same point on the testing ray Cm, the function $E_u(p_u)$ must have the same value for different u.

Similar relations can also be deduced for the cylindrical camera field. Figure 2 shows the projection of a cylindrical camera field on the X-Y plane. Using triangles C_uCM and $CC_{u+1}M$ we can find the following relations:

$$\begin{aligned}\frac{CM}{\sin(\frac{\pi-\alpha_u}{2} - \theta_u)} &= \frac{C_u C}{\sin(\alpha_u - \theta + \theta_u)} \\ \frac{CM}{\sin(\frac{\pi-\alpha_{u+1}}{2} + \theta_{u+1})} &= \frac{CC_{u+1}}{\sin(\alpha_{u+1} + \theta - \theta_{u+1})}\end{aligned}$$

Therefore, we can define $E_u(p_u)$ as:

$$E_u(p_u) = \frac{Cm}{CM} = (-1)^k \frac{Cm \sin((-1)^k \alpha_u + \theta_u - \theta)}{C_u C \cos((-1)^k \frac{\alpha_u}{2} + \theta_u)} \tag{6}$$

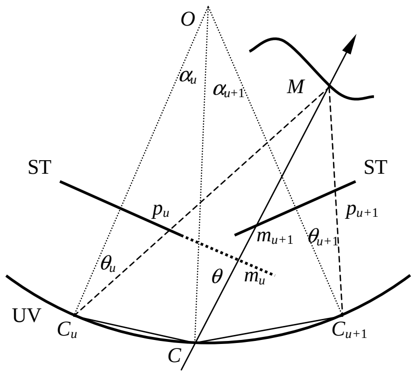


Figure 2: Interpolation between reference images for the cylindrical camera field.

where k is equal to 0 when C is on the right hand side of C_u, and k is equal to 1 otherwise.

As a result, what we need to do is to simultaneously move the current pixel, p_u, along the epipolar line and keep $E_u(p_u)$ the same for all nearby reference images u. The pixels that have the smallest dissimilarity are the projections of the intersection that we search for. The pseudocode for this algorithm is shown in Figure 3.

Figure 4 shows the experimental results for a planar camera field, which is captured by the camera matrix at the University of Tsukuba. Four reference images are used to interpolate the novel view, whose viewpoint is at the center of the square that is formed by the viewpoints of the four reference images. Figure 4(a) is generated using linear interpolation. Since all reference images have equal weights, the generated result is the same as cross-blending the four reference images together. Figure 4(b), which is much sharper, is generated using color-matching based interpolation.

As shown in Figure 4(b), artifacts still exist, mainly because the two assumptions we used are not satisfied. For example, many details are lost in the rendered image, such as texts on the poster and highlights on the lamp. This is because of the violation of the second assumption, i.e., the projections of a physical point do not have the smallest dissimilar value.

Figure 5 uses a planar camera field to illustrate the cause of this problem. As shown in the figure, ray Cm intersects the surface at location M, and the projections of M on the two images are p_u and p_{u+1} respectively. The ray $C_u n_u$ intersects the surface at location N, the ray $C_{u+1} l_{u+1}$ intersects the surface at location L, and their intersection, W, lies on the ray Cm. Assume the dissimilarity between n_u and l_{u+1} is smaller than that of p_u and p_{u+1}. Then, based on the second assumption, we will conclude that there exists a point in the scene at location W, which generates the observed projections n_u and l_{u+1}. Actually, in the above example, even though the two dissimilarities are the same, we will still pick the false target since location W as compared with M is closer to the viewpoint.

2.3 Disparity-Matching Based Interpolation

In order to solve the above problem, we need to acquire depth information from the image. Previous approaches try to calculate the three-dimensional model of the object in the scene using the visual

```
For each reference image u
  Set p[u] = $m_u$, E[u] = $E_u(p_u)$,
    Color[u] = $I_u(p_u)$;
While ( true ) {
  Set mean = weighted-average of Color[u]
    for different u;
  Set error = weighted-sum of the
    Euclidean distances between Color[u]
    and mean;
  If ( error < min_error )
    Update best_match = mean,
      min_error = error;
  Find the image u that has the highest
    value of E[u];
  Move p[u] one pixel closer to the $b_u$;
  If ( $b_u p_u < 0$ ) Break;
  Update E[u] = $E_u(p_u)$, Color[u] = $I_u(p_u)$;
}
return best_match;
```

Figure 3: Color-matching based interpolation approach.

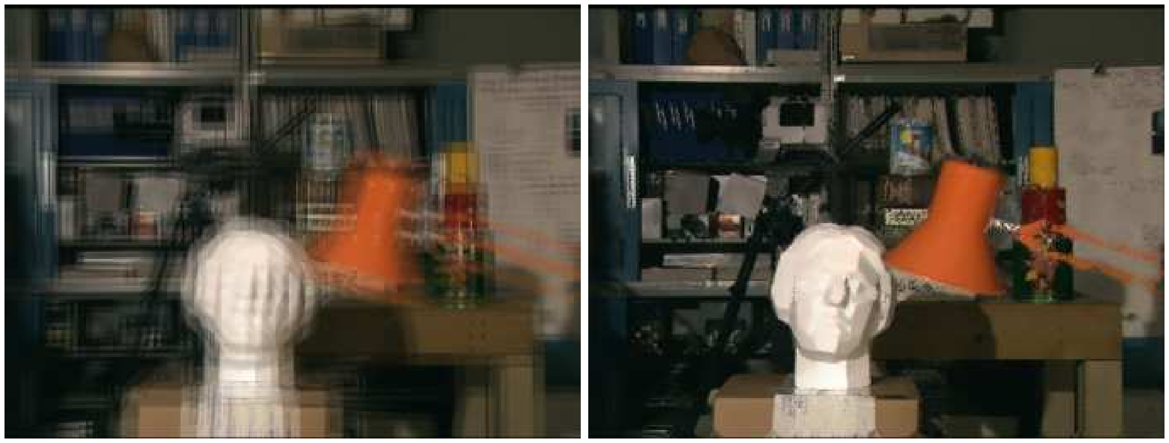


Figure 4: Interpolation result for the “head and lamp” (a) linear (b) color-matching.

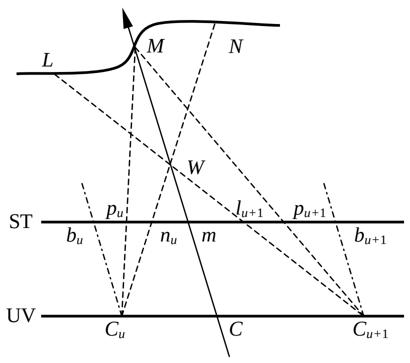


Figure 5: Incorrect interpolations using color-matching based approach.

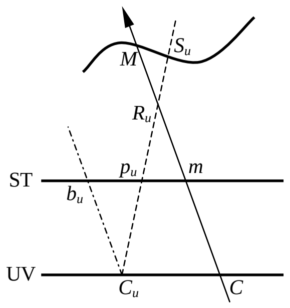


Figure 6: Intersection searching using a single disparity image for planar camera field.

hull[7] or stereovision[18] techniques. However, since the real scenes are rich in details, representing the three-dimension model itself is a difficult problem. Using volume representation[7] requires a large memory, and thus, it is more appropriate for low-resolution representations. Using boundary representation[18] will normally produce too many polygons, and a complex model simplification process is needed.

Here, we try to avoid using the three-dimensional model of the scene. Disparity maps, which give the view-dependent depth information, are used in this paper. There are several different ways to define disparities. The one similar to the inverse distance[14] is used here, which has the following formula:

$$\delta(m) = \frac{Cm}{CM} \tag{7}$$

where C is the center of projection, m the pixel on the image plane, M the intersection between the ray, Cm, and the object in the scene. Since we assume all the intersections are in front of the image plane, the legal value of disparity is between 0 and 1.

For a synthetic scene, accurate disparity maps can be obtained during the rendering process. For a real scene, the disparity maps are generated by our recently proposed stereovision algorithm[6], which can effectively remove mismatches caused by both occlusions and false targets. However, the errors in the disparity maps are inevitable. Our experiments show that for real scenes, most of the existing stereo algorithms can only achieve about 90% absolutely correct disparity values, although about 98% disparity values generated are within the range of ground truth ± 1. Therefore, it is necessary to have a rendering algorithm that is not sensitive to these errors.

Suppose we can obtain relatively accurate disparity information. Now, the question is how to interpolate existing multiple images with disparities. Here, in this paper, we will propose a backward searching approach. In the rest of this section the cases for planar and cylindrical camera fields are discussed first, followed by the general searching algorithm.

Figure 6 shows the cross-section of a planar camera field. Suppose we want to find the intersection of ray Cm with objects in the scene using a known view, whose center of projection is C_u. As was mentioned above, we know that the image of intersection M lies on segment $b_u m$. For any pixel p_u on $b_u m$, we can calculate the length of $C_u R_u$ using the following equation:

$$\frac{C_u p_u}{C_u R_u} = \frac{b_u p_u}{C_u C} \Rightarrow C_u R_u = C_u p_u \times \frac{C_u C}{b_u p_u}$$

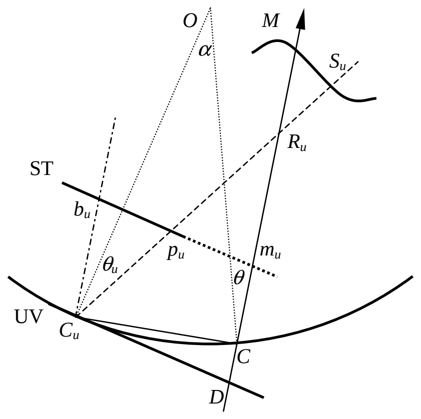


Figure 7: Intersection searching using a single disparity image for a cylindrical camera field.

Also, based on the known disparity map, we can calculate the length of C_uS_u by:

$$\delta(p_u) = \frac{C_up_u}{C_uS_u} \Rightarrow C_uS_u = C_up_u \times \frac{1}{\delta_u(p_u)}$$

Now we can define another function $F_u(p_u)$ as:

$$F_u(p_u) = \frac{1}{C_up_u}(\frac{1}{C_uS_u} - \frac{1}{C_uR_u}) = \delta_u(p_u) - \frac{b_up_u}{C_uC} \tag{8}$$

Obviously, if the test ray intersects the surface at location M and its projection on image u is p_u, then function $F_u(p_u)$ should be zero. Otherwise, $F_u(p_u) > 0$ means the intersection with the surface is in front of the intersection with the testing ray, and $F_u(p_u) < 0$ means the intersection with the surface is behind the intersection with the testing ray. Therefore, the problem of searching the corresponding point is equivalent to the problem of finding the zero-crossing point of $F_u(p_u)$. $F_u(p_u)$ can be efficiently evaluated since it only needs one addition and one division operation.

A similar function can be found for the cylindrical camera field as well. In Figure 7, the length of C_uR_u can be calculated by:

$$\frac{C_up_u}{C_uR_u} = \frac{b_up_u}{C_uD} \Rightarrow C_uR_u = C_up_u \times \frac{C_uD}{b_up_u}$$

where we have:

$$\frac{C_uC}{\sin(\frac{\pi+\alpha}{2} - \theta)} = \frac{C_uD}{\sin(\frac{\pi}{2} - \alpha + \theta)} \Rightarrow C_uD = \frac{C_uC\cos(\frac{\alpha}{2} - \theta)}{\cos(\alpha - \theta)}$$

Hence, the equation for $F_u(p_u)$ becomes:

$$F_u(p_u) = \frac{1}{C_up_u}(\frac{1}{C_uS_u} - \frac{1}{C_uR_u}) = \delta_u(p_u) - \frac{b_up_u\cos(\alpha - \theta)}{C_uC\cos(\frac{\alpha}{2} - \theta)} \tag{9}$$

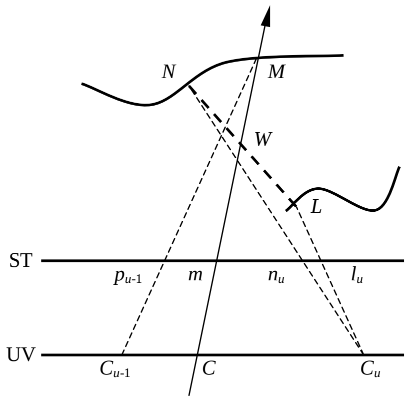


Figure 8: The "rubber sheets" problem caused by linear interpolation.

$F_u(p_u)$ is not continuous since the disparity function, $\delta_u(p_u)$, is defined on discrete samples. In order to find the zero-crossing point, linear interpolation can be used to reconstruct the continuous function. Since linear interpolation will also connect pixels that belong to different objects together, it will generate "rubber sheets" in the resulting images, in which different objects are stretched inappropriately. The following scenario illustrates the cause of the rubber sheets.

As shown in Figure 8, ray Cm intersects the background surface at location M, which is projected at location p_{u-1} in image C_{u-1} but not visible in image C_u. Two pixels n_u and l_u are adjacent to each other in image C_u. Ray $C_u n_u$ intersects the background surface at location N, and ray $C_u l_u$ intersects the foreground surface at location L. Under such a scenario, we have $F_u(l_u) > 0$ and $F_u(n_u) < 0$. Therefore, linear interpolation will give us a zero-crossing point, which indicates that the intersection is at location W. Since location W is closer than location M, the color for ray Cm will be computed by interpolating point L and N, which is not correct.

To remove the rubber sheets effects, we set up a threshold t. Whenever $|F_u(x) - F_u(x+1)| > t$, we assume that pixels x and $x+1$ are the projections of points on two different objects. Therefore, no intersection will be computed any more.

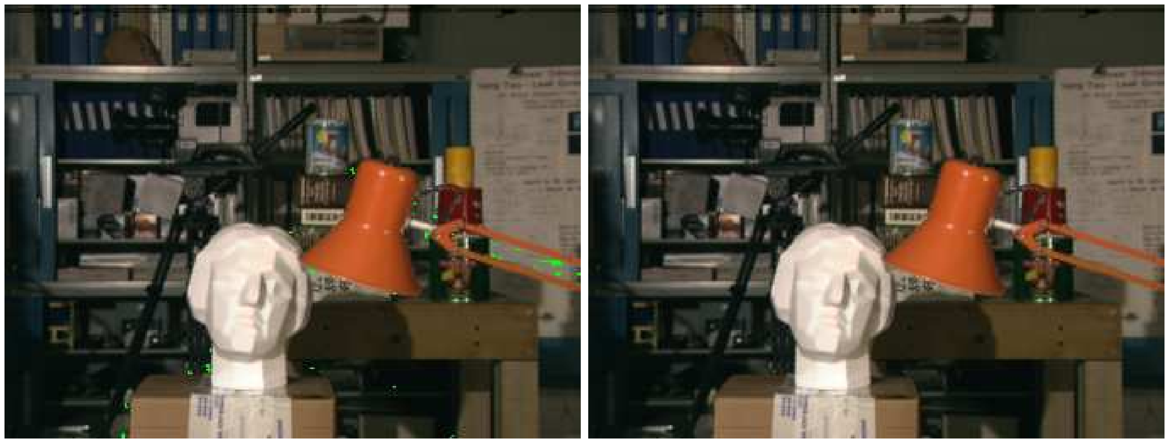
The above discussion is for a single reference image. When multiple reference images are available, we need to find the closest zero-crossing point among different function, $F_u(x)$, for different reference image u. The corresponding pixel will be used to color the novel ray. In the case when two or more zero-crossing points have the same closeness to the center of projection, the final color is produced through weighted-averaging the corresponding pixels according to the distances between the reference views and the virtual view.

It is noteworthy that there is no need to search for all the zero-crossing points and project them back to three dimensions to find which one is the closest. We know that a point on Cm that is closer to the center of projection is projected to a pixel that is closer to m. Therefore, we can simultaneously move the current pixel, p_u, along the epipolar line and keep $E_u(p_u)$ the same for all nearby reference image u. The first zero-crossing point we found will be the closest intersection. The pseudocode of the algorithm is shown in Figure 9.

Figure 10(a) shows the rendering result of disparity-matching based approach for the same

```
For each reference image u
  Set p[u] = $m_u$, E[u] = $E_u(p_u)$,
    NewF[u] = $F_u(p_u)$;
While ( true ) {
  Find the image u that has the highest
    value of E[u];
  Set OldF[u] = NewF[u], NewF[u] = $F_u(p_u)$;
  If ( NewF[u]$\times$OldF[u]<0 &&
    abs(NewF[u]–OldF[u])>$t$ )
    Return $I_u(p_u)$;
  Move p[u] one pixel closer to the $b_u$;
  If ( $b_u p_u$<0 )
    Return “no intersection found”;
}
```

Figure 9: Disparity-matching based interpolation approach.



(a) (b)

Figure 10: Interpolation result for the “head and lamp” (a) disparity-matching (b) combined.

planar camera field. The disparity information is computed using our stereovision algorithm. The result demonstrates that the details of the scene, e.g. highlights on the lamp and texts on the poster, are preserved. However, since a tight threshold is used to remove the "rubber sheet" effect, the disparity-matching based approach fails to find the correct intersections in some areas, which are marked by green pixels in Figure 10(a).

The above problem is caused by the errors in the disparity maps. Our strategy of providing a robust rendering algorithm is to combine the two interpolation techniques together. That is, try to search for an intersection using the disparity-matching based approach first. If no intersection is found, then use the color-matching based approach instead to find the possible physical point using color consistency.

This strategy is justified by the following observations. Normally the intensity-based stereovision algorithms are prone to error in solid color areas. This causes the disparity maps generated to be rather noisy in these areas. Since a high degree of discontinuity exists, the zero-crossing points found by the disparity-matching based approach tend to be larger than the given threshold, and hence, no intersection is found. However, in the solid color areas, the color-matching based approach can do a much better job. The result of the combined approach is shown in Figure 10(b). It shows that a smooth and detailed result can be generated.

3 Experimental Results

The camera matrix uses a two-dimensional array of cameras mounted on a plane to capture real scenes[20]. Therefore, the images obtained naturally fit into the planar camera field. One such dataset, the "Santa Claus", is used to test our algorithm.

Figure 11 shows the rendering results for the "Santa Claus" dataset. The original dataset contains 9×9 images with 636×472 resolution . Four reference images, whose coordinates are (3,3), (3,5), (5,3), and (5,5) in the dataset, are used to interpolate the in-between view, i.e., the reference image, whose coordinates are (4,4) in the dataset. As a result, we can use this reference image, shown in Figure 11(e), as a ground truth to evaluate the rendering results. Since the disparity between neighboring reference image is very large (102 pixels, nearly one sixth of the image width), the linear interpolation method, as shown in Figure 11(a), gives very blurry result. The result generated by the color-matching based approach, shown in Figure 11(b), is sharp and smooth. However, closer inspection shows the artifacts in areas around the mouth and the right eye. In addition, we also lost the details on the hat of the Santa Claus and on the wall. Figure 11(c) shows the result of the disparity-matching based approach, in which the artifacts are removed and the details are preserved. The green pixels in Figure 11(c) show the areas where no intersections are found. In the combined approach, shown in Figure 11(d), these areas are filled in by the color-matching based approach. The absolute difference between the result of the combined approach and the ground truth is shown in Figure 11(f). The darker the area the larger is the error. One can see that the error is quite small.

Since we do not have a cylindrical camera field for real scenes, a synthetic scene is used to test the algorithm. The scene we used is an architectural model, which contains many fine details. 72×10 images are used to sample the cylinder, resulting one reference image per 5 degrees in the horizontal direction. This is a very sparse sampling compared with the light field approach, in

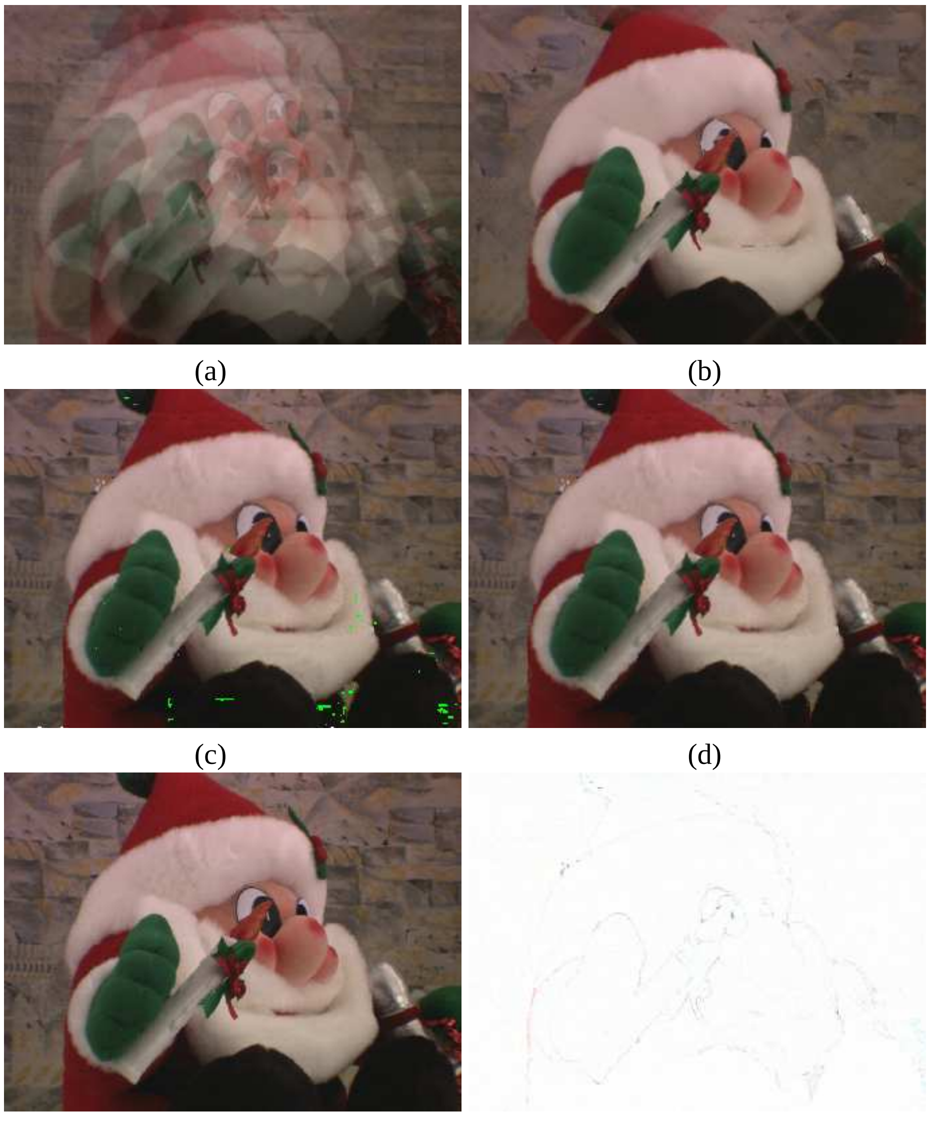

(a) (b) (c) (d) (e) (f)

Figure 11: Rendering results for a planar camera field, “Santa Claus”.

which 32×32 images are used to sample a 45×45 degree area of the "Buddha" model, and with the concentric mosaics, which samples 20 concentric circles in 3000 angular directions, resulting one sample per 0.12 degrees in the horizontal direction.

Figure 12 shows the results of the algorithm using a synthetic scene. The disparity information of the scene is obtained from the rendering process, and therefore, is accurate. Figure 12(a) shows the in-between view interpolated using linear interpolation. The result of the color-matching based approach is shown in Figure 12(b). Artifacts show up near the boundary, and details, such as the frames for the windows and textures on the lawns, are lost. Figure 12(c) shows the "rubber sheet" artifacts of the disparity-matching based approach when no threshold is used. As shown in Figure 12(d), these artifacts are removed after a threshold, $t = 0.1$, is used. Since accurate disparity maps are used, only few pixels around the balcony are left without any intersection found. For comparison, the rendering result for the in-between viewing position is also computed and shown in Figure 12(e). The absolute difference between the rendered image and interpolated result of the combined approach is shown in Figure 12(f). Even though Figure 12(f) shows large errors around the edge of the surfaces, closer inspection indicates that these fine details do show up in our rendering result. The errors are actually caused by the interpolation, which tends to blend the edges.

The animations that are generated using our approach and the linear interpolation approach are also available and are submitted along with the paper.

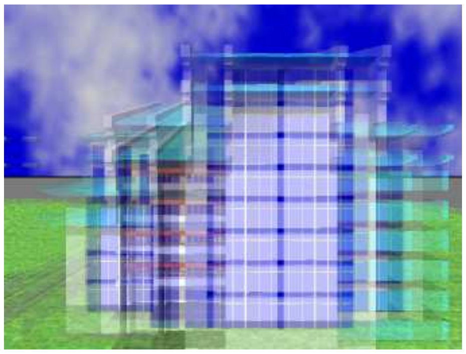
4 Conclusions

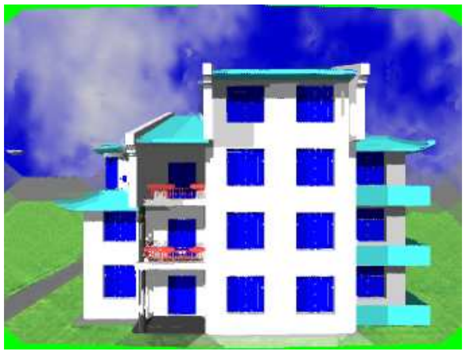
In this paper, a new sampling scheme, camera field, is proposed. A camera field can be acquired using a two-dimensional array of cameras mounted on the corresponding support surface, with the viewing directions of the cameras aligned with the normals of the support surface. Since no resampling process is required, we can fully utilize the resolution of the cameras.

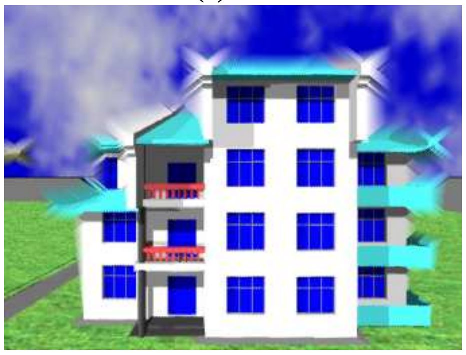
Camera fields can be defined on different kinds of support surfaces, such as planes, parametric surfaces, and free-form surfaces. Which kind of surface can sample the scene more efficiently depends on the applications. A planar camera field is similar to a light slab that has one of the two sampling planes put at infinity. A single planar camera field can be used to sample one side of an environment or one side of an object. Same as in light field rendering, we can place several planar camera fields to fully sample the scenes. Sampling the scene using planar camera field directly (without resampling) is practical. Modern techniques have demonstrated the capability of building multiple cameras on the same board, and therefore, these cameras are aligned sufficiently for this application. Alternatively, we can also use a single camera attached to a vertical precision X-Y table.

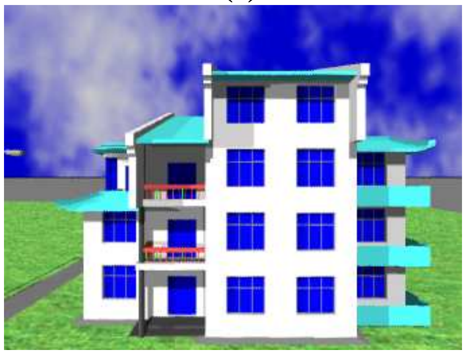
Outward looking cylindrical camera field can fully sample the surrounding environments, except the top and the bottom. It provides more uniform sampling than using four light slabs, and therefore, requires less data. Outward looking cylindrical camera field is also a higher-dimensional version of concentric mosaics. Hence, it gives the user the freedom to move up and down within the cylinder. Since the interpolation technique we used can handle sparse samples, the data required by the cylindrical camera field is not any higher than that required by concentric mosaics[25].

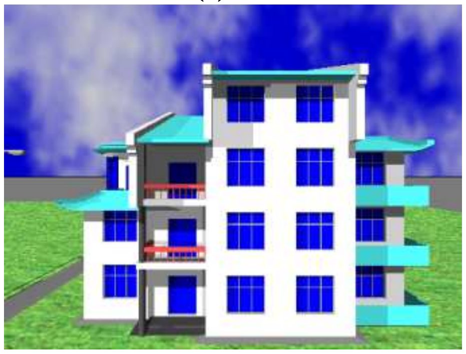
An inward looking cylindrical camera field can fully sample the surrounding of an object and the


(a)


(b)


(c)


(d)


(e)

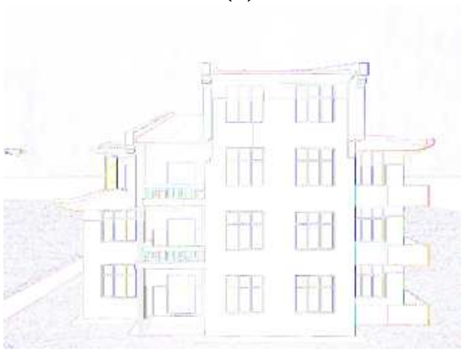

(f)

Figure 12: Rendering result for a cylindrical camera field, the “architecture model”.

environment behind it, except the top and the bottom. Compared with the object-centered concentric mosaics[5], an inward looking cylindrical camera field does not require accurate depth information. In addition, it can be captured using normal pinhole cameras, while object-centered concentric mosaics requires either parallel projected scanners or resampling existing images.

Although not discussed in this paper, we believe that the algorithms given in Figure 3 and Figure 9 can be applied to other types of parametric surfaces or free-form surfaces as well. For example, we can use a spherical camera field to fully sample an object or an environment. An inward looking spherical camera field has the same sampling scheme as that used in object movie[3]. Since the disparity information is calculated and utilized, an inward looking spherical camera field can provide parallax effects, which cannot be generated using object movies.

Since linear interpolation produces blurry results when only a sparse rayset is available, two novel interpolation techniques are proposed. The first one, the color-matching based approach, does not require explicit depth information. For any given testing ray, it searches for a possible physical point along the testing ray using color information of nearby reference images. If more than one point is found, the one closer to the center of projection is used. This approach is robust, but it may give the wrong interpolation result when the two assumptions are not satisfied.

The second technique, the disparity-matching based approach, employs the calculated disparity maps for the reference images. A backward searching process is used to find the closest intersection between the testing ray and the disparity surface defined by the nearby reference images. The rubber sheet effect is removed using a threshold. This approach can generate correct results even when occlusions exist. However, it may give holes when the disparity maps calculated are noisy.

Our strategy of providing a robust rendering algorithm is to combine the two interpolation techniques together. That is, try to search for an intersection using the disparity-matching based approach first. If no intersection is found, the color-matching based approach is used to fill the hole. This strategy works very well since the intensity-based stereovision algorithm we used is prone to error in uniform color areas, where the color-matching based approach can do a very good job.

Both the color-matching based and disparity-matching based approaches are backward rendering techniques. Comparing with the forward rendering techniques, in our approach, there is no need to reproject all the samples and to fill the Z-buffer. In addition, we do not need to face the problem of how to fill the holes as are common in forward mapping approaches. The concepts of these two interpolations can be applied to arbitrary camera setting as long as the positions and orientations of the cameras are known. However, we demonstrate that after restricting the cameras' positions using a planar or cylindrical support surface, we can easily find the corresponding points among different views without going to the third dimension. Therefore, no matrix computations are involved and the interpolations can be implemented efficiently in the image space.

Our disparity-matching based approach for planar camera field is similar to the interpolation algorithm proposed by Satoh, et al.[19]. Compared with their approach, we have suggested methods to remove the "rubber sheet" effect and to accommodate the potential errors in the disparity map. Furthermore, we use a more general parameterization scheme and propose general interpolation algorithms. The formulas for cylindrical camera fields are also derived.

Both interpolation approaches we propose have a time complexity of $O(d \times r \times c)$, where r and c are the width and height of the image to be generated, and d is the difference between the minimum and maximum disparities of neighboring reference images. This means that the performance is independent of the number of reference images and their resolutions. Obviously, the denser the rayset is sampled, the smaller is the value of d. When the dataset sampled by the light field is used, where d is close to 1, the speed of our algorithm is close to that of the light field rendering approach. With the value of d increasing, the cost for computations increases linearly, but the cost for sampling decreases quadratically.

In practice, for the “head and lamp” scene, which contains 25 images with 384×288 pixels and whose maximum disparity is 14, our implementation takes $0.2 \sim 0.4$ second to render a frame (256×256 in size) on our 1.6GHz Pentium 4 PC running Windows 2000. Since our current implementation does not utilize any hardware acceleration and is not optimized, we believe there is still room for improvements in performance.

In summary, the proposed approach can be used in different environments and can produce acceptable rendering results when only sparse samples are available. Future works include deriving equations for other types of support surfaces, and finding a general method to calculate the minimum sampling rate required for different support surfaces.

Acknowledgements

The authors would like to thank Y. Ohta of the University of Tsukuba for supplying images for the “Santa Claus” scene.

References

[1] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured lumigraph rendering. In *Siggraph Annual Conference*, pages 425–432. ACM, August 12-17 2001.

[2] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum. Plenoptic sampling. In *Siggraph Annual Conference*, pages 307–318. ACM, July 23-28 2000.

[3] S. E. Chen. Quicktime vr - an image-based approach to virtual environment navigation. In *Siggraph Annual Conference*, pages 29–38. ACM, August 9-11 1995.

[4] S. E. Chen and L. Williams. View interpolation for image synthesis. In *Siggraph Annual Conference*, pages 279–285. ACM, August 1-6 1993.

[5] M. Gong and Y.-H. Yang. Rayset and its applications. In *Graphics Interface*, pages 141–148. Canadian Human-Computer Communications Society, June 7-9 2001.

[6] M. Gong and Y.-H. Yang. Genetic-based stereo algorithm and disparity map evaluation. *International Journal of Compute Vision*, To Appear 2002.

[7] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The lumigraph. In *Siggraph Annual Conference*, pages 43–54. ACM, August 4-9 1996.

[8] A. Isaksen, L. McMillan, and S. J. Gortler. Dynamically reparameterized light fields. In *Siggraph Annual Conference*, pages 297–306. ACM, July 23-28 2000.

[9] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. In *International Conference on Computer Vision*, pages 307–314, September 20-25 1999.

[10] S. Laveau and O. D. Faugeras. 3-d scene representation as a collection of images. In *International Conference on Pattern Recognition*, volume 1, pages 689–691. IAPR, October 9-13 1994.

[11] M. Levoy and P. Hanrahan. Light field rendering. In *Siggraph Annual Conference*, pages 31–42. ACM, August 4-9 1996.

[12] L. McMillan and G. Bishop. Head-tracked stereoscopic display using image warping. In *International Symposium on Electronic Imaging: Stereoscopic Displays and Virtual Reality Systems II*, volume 2409, pages 21–30. SPIE, February 7-8 1995.

[13] L. McMillan and G. Bishop. Plenoptic modeling: an image-based rendering system. In *Siggraph Annual Conference*, pages 39–46. ACM, August 9-11 1995.

[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] M. M. Oliveira and G. Bishop. Image-based objects. In *Symposium on Interactive 3D Graphics*, pages 191–198. ACM, April 26-28 1999.

[16] M. M. Oliveira, G. Bishop, and D. McAllister. Relief texture mapping. In *Siggraph Annual Conference*, pages 359–368. ACM, July 23-28 2000.

[17] P. Rademacher and G. Bishop. Multiple-center-of-projection images. In *Siggraph Annual Conference*, pages 199–206. ACM, July 19-24 1998.

[18] P. Rander, P. J. Narayanan, and T. Kanade. Virtualized reality: constructing time-varying virtual worlds from real world events. In *IEEE Visualization*, pages 277–283. IEEE, October 1997.

[19] K. Satoh, I. Kitahara, and Y. Ohta. 3d image display with motion parallax by camera matrix stereo. In *International Conference on Multimedia Computing and Systems*, pages 349–357. IEEE, June 17-23 1996.

[20] K. Satoh and Y. Ohta. Occlusion detectable stereo using a camera matrix. In *Asian Conference on Computer Vision*, pages 331–335. IAPR, December 5-8 1995.

[21] G. Schaufler and M. Priglinger. Efficient displacement mapping by image warping. In *Eurographics Workshop on Rendering*, pages 175–186. EG, June 21-23 1999.

[22] H. Schirmacher, W. Heidrich, and H.-P. Seidel. High-quality interactive lumigraph rendering through warping. In *Graphics Interface*, pages 87–94, May 15-17 2000.

[23] H. Schirmacher, M. Li, and H.-P. Seidel. On-the-fly processing of generalized lumigraphs. In *Eurographics Annual Conference*. EG, September 4-7 2001.

[24] S. M. Seitz and C. R. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Conference on Computer Vision and Pattern Recognition*, pages 1067–1073. IEEE, June 17-19 1997.

[25] H.-Y. Shum and L.-W. He. Rendering with concentric mosaics. In *Siggraph Annual Conference*, pages 299–306. ACM, August 8-13 1999.