Light Field Rendering of Dynamic Scene

Wei Li, Qi Ke, Xiaohu Huang, and Nanning Zheng
Institute of Artificial Intelligence and Robotics
Xi'an Jiaotong University

Abstract:

Image based rendering has displayed advantage in speed over traditional geometry based rendering algorithms. With the four dimensional light field descriptions, static scene can be generated with interactive rate on ordinary computer. The limitation of the scheme is that it can only handle static scene and fixed illumination. This paper raises the idea to decompose the light field into sub-light-fields that do not change as scene changes. It expands the advantage of light field rendering to dynamic scenes where the position and orientation of objects, lights and viewpoint can be modified arbitrarily.

The sub-light-fields include: ambient light field and spot light field. The latter is actually an eight-dimensional space. Because diffuse reflection is independent on view direction, this paper presents a four-dimensional representation of spot light field. Considering the linearity of diffuse reflection to different spot lights, the spot light fields of an object can be represented by the reflection light field to a pure-color light with unit intensity, to decrease storage and preprocessing. Owing to the coherency in their data structures, data of the corresponding point in the ambient light field, diffuse light field and depth field are combined into a 5-dimensional vector which can be compressed efficiently with vector quantization. The algorithm given in this paper accurately computes typical characteristics of dynamic scene such as changes in surface color and shadow.

Keywords: Image-based Rendering, Ambient Light Field, Spot Light Field, Diffuse Light Field, and Depth Field.

1. Introduction:

The traditional geometry based graphics rendering algorithm needs to handle description of the borders of divided scenes (polygon rendering) or sampling of space functions (volume rendering) and all kinds of lighting models. When the geometry model is very complicated, it will take quite a long time to render an image if we want to display the scene as real as possible.

In recent years, researchers are more and more interested in image based rendering, which takes rendered or natural images as input and generate new scenes after a series of simple operations (such as memory operation or linear interpolation). Rendering time is reduced by these algorithms compared with traditional methods and generally not related to the complicity of the scene.

Environment Map is a representative of early image based rendering algorithms. An Environment Map records the colors of rays casting onto a point from all directions. QuickTime VR [4] released by Apple Inc. is exactly a commercial system based on an idea in which users can view virtual environments in any direction from a fixed viewpoint.

The most serious problem in the Environment-Map-based algorithm is that the viewpoint is fixed. Thus there are only three degrees of freedom when a viewer interacts with the scene in the space of five degrees of freedom, that is, only the viewing direction can be changed. To overcome this shortcoming, a new image composition method based on image warping and view interpolation is put forth [5] and [3]. In this algorithm, the depth of each pixel is needed or corresponding points between images should be predetermined.

Recently, Marc Levoy and Pat Hanrahan brought about a light field rendering algorithm [1]. At the same time, a similar algorithm based on the so-called Lumigraph is also presented by Steven Gortler *et. al.* [2]. The light field is a function that describes the variation of radiance according to space coordinates and casting directions. It contains information of the scene from any viewpoint in any direction and makes it unnecessary to match pixels between images. Light field rendering can be used to generate static scenes of any complexity at interactive speed on low-cost workstations or PCs.

Only static scenes can be handled is the obvious shortcoming of the light field rendering method mentioned above, that is, the positions and directions of objects and lights are not allowed to change. However in dynamic scenes, the reflected rays from the object surface and the shadow on an object cast by another are variables to the relative positions of objects and lights. In this paper, we present a light field rendering algorithm to render dynamic scenes. That is, the positions and orientations of objects, lights and viewpoint as well as the lights' colors are all changeable by users or application programs. The basic idea of the algorithm is to divide the light field of the whole scene into several sub-light fields that will not change with the scene, but only shift or rotate as a whole. We propose two kinds of sub-light fields: the ambient light field and the spot light field that are respectively independent and dependent on ray direction. Then, we only need to sample and superimpose the sub-light field while rendering. Shadow effect also becomes easy to produce using sub-light field. Because the average amplitude of the sub-light fields is smaller than the corresponding static scene, data can be compressed more efficiently. Furthermore, as the size of the raw data of the reflection light field is only one third of the ambient light field, data volume required by this algorithm will not increase much compared with the light field rendering for static scenes.

The main purpose of this paper is to present an interactive rendering technique for dynamic virtual environment generation on general computers, to replace the traditional geometry based method. The algorithm is restricted to the scenario of producing light fields from synthetic images, hence the rendering system can obtain precise position, rotation and depth information of lights and objects. Thus, it is reasonable for a rendering system to know how the scene has changed, and describes such changes as position and direction parameters of the objects and lights.

In the rest of the paper, we will first briefly review the light field representation of static scene [1] and [2], then describe the definition and the construction of ambient light field, diffuse light field and depth field in detail, as well as the image composition method based on these fields. Several rendered images and slices of light fields of a room containing a girl are presented as experimental results associated with data compression statistics and analysis. Finally, we conclude the paper.

2. The Light Field Representation of a Scene

The light field is a continuous function of radiance depending on the five variables representing the space coordinates and directions (three for transition and two for rotation). McMillan and Bishop describe the 5-dimensional Plenoptic Space, with a series of panorama projected on a cylinder [3]. This may be regarded as an equivalent notation to light field. Assuming the air to be transparent, the radiance along a ray through empty space remains constant, so a four-dimensional representation of light field is enough to replace the five-dimensional Plenoptic space. Marc Levoy *et. al.* and Steven Gortler *et. al.* have successfully described the light field by intersecting arbitrary rays with two parallel planes.

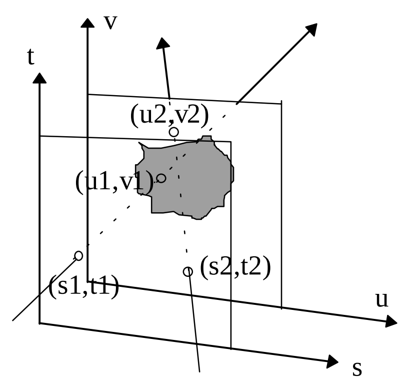


Figure 1 Parameterization of light field [2]

As shown in Figure 1, any ray in the 3-dimensional space can be determined by intersecting with *uv-plane* and st. For example, the two rays in the figure can be represented as *ray(u1,v1,s1,t1)* and *ray(u2,v2,s2,t2)*, where *u, v, s, t* are plane coordinates in *uv-plane* and st respectively. *Uv-plane* parallels *st-plane*. A ray in a 3-dimensional space is uniquely corresponding to a point *(u, v, s, t)* in the light field, whose value is the color emitted by the scene in the opposite direction of *ray(u, v, s, t)*. We define the light field as L, and *L(u, v, s, t)* represents the value of *ray(u, v, s, t)*. With up to 6 pairs of surrounding planes, the surface of an object can be completely recorded. For computation simplicity, these plane pairs are mutually perpendicular.

Obviously, the quality of images will be improved as the resolutions of *uv-plane* and st increase. However, the burden on memory will also grow with it. Therefore, the sampling rate should be a compromise between quality and storage. Suppose that the object stands near to *uv-plane*, then *uv-plane* can be regarded as the final image plane and its resolution is close to the resolution of the image. When the values of s and t are set while *u, v* vary, we get the image of *uv-plane* viewed from *(s, t)*. It is easy to imagine that when *(s, t)* is slightly moved on st-plane, the image will not change greatly, thus the resolution of *st-plane* can be lower than *uv-plane*.

If light fields established, the work for rendering is to sample the light field. Figure 2 shows the relationship between light field and a pixel in an arbitrary image. *(u, v), (s, t)* are coordinates of the intersections of *uv-* and *st-plane* with the ray from the viewpoint to the center of the pixel, so *ray(u,v,s,t)* can be used to represent this ray with *L(u, v, s, t)* to be its value *(color)*. Theoretically, the light field is a continuous space of no limitation. However, it has to be represented by bounded and discrete samples for practical use. Since the intersections seldom coincide with the sampling grids in *uv-* and *st-plane*, interpolation is necessary. When all the values of pixels are obtained by casting rays and sampling light field, an image is finished.

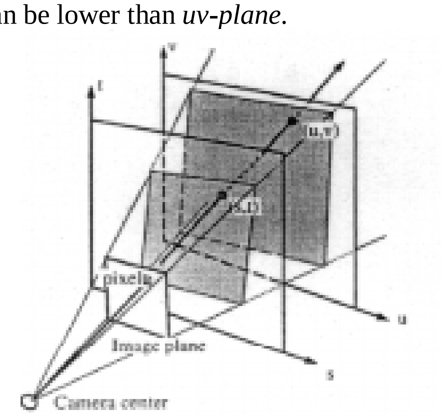


Figure 2 Relationship between light field and a pixel in an arbitrary image [2]

The algorithm based on light field can be regarded as a table-looking method. It completes much time-consuming computation during pre-procession and acquires high speed during rendering at the expense of memory.

3. Dynamic Scene Rendering

In dynamic scenes, the color and shadow of an object change with the variation of the relative position between light and object and between objects. If we decompose the whole light field into a series of sub-light fields that just shift or rotate as the scene changes, without any change in data and their internal structure, dynamic scene can be rendered by superimposing the radiance. The variations mentioned here include:

(1) shifting and rotating of rigid objects;
(2) position and color (radiance) changes of spot lights, and changes of the number of lights;
(3) changes in view-point and view-direction.

Non-rigid objects are not taken into consideration because the sub-light field is going to vary with transformation of objects, although there is no shift or rotation.

3.1 Ambient Light Field and Diffuse Light Field

Suppose ***P*** is a point on the surface and *l* is a spot light with radiance I_l . In fact I_l is composed of three elements: R, G and B. According to the basic lighting model, the radiance of ***P*** can be represented as follows:

$$I_p = I_{ad} + \sum_l I_{ld} + I_{lS} \qquad (1)$$

This value can also be regarded as the color of the ray emitted from ***P*** in the opposite direction of viewing. In the formula, I_{ad} is the diffuse reflection to the ambient light, which is independent on positions of lights. I_{ld} is the diffuse reflection to *l*. When incident angle $\theta = [0, \pi/2]$, $I_{ld} = I_l k_d \cos\theta$, otherwise: 0. k_d is a constant, so I_{ld} is only related to the incident angle. I_{ls} is the specular reflection contributed by l , which varies with the incident angle and the angle between reflecting ray and the view direction.

When there is only ambient light and no spotlight in the scene, we only need to establish a light field for each object according to the method discussed above. Image is generated by sampling from the light field of the object closest to the viewpoint. We call the light field only contributed by ambient light as ambient light field.

If there is a spot light in the scene, where objects or lights shift, not only the reflection from the object surface but also the shape and position of the shadow, are liable to change. Both the diffuse reflection and specular reflection are affected by the direction of the incident light. As shown in Figure 3, a point in the reflection light field is determined by eight parameters: *(u1,v1,s1,t1,u2,v2,s2,t2)*, because of the variation of the incident direction. For the immense data, it is impossible to represent this 8-dimensional light field with the same method describing static light field. To make the scheme feasible, we have to look for another way.

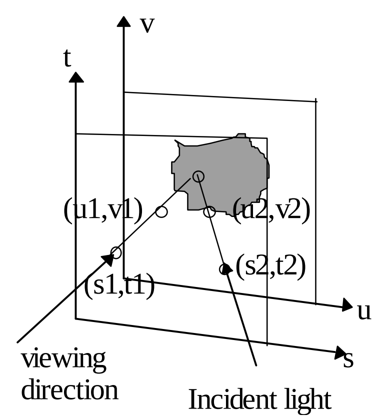


Figure 3 Parameterization of reflective light field.

Actually, diffuse reflection is independent on the viewing direction (unrelated to *u1,v1,s1,t1*). That makes it possible to describe the diffuse light field as a 4-dimensional space. For example, *(u2,v2,s2,t2)* represents a point in the diffuse light field, whose value is determined by the intensity of the diffuse reflection at the intersection where the ray casing in the direction of *(u2,v2,s2,t2)* meets the surface of the object. Unfortunately, it is impossible to decide the incident direction only from view direction and the position of spot light. As in Figure 4, only when the surface point of an object is on *uv-plane*, say B, is it possible to determine the diffuse reflection of the point contributed by the spot light based on just the position of the light and the view-direction. While observing point A, the depth information of A is needed. Thus, for each object, a depth field should be established, whose 4-dimensional coordinate system is similar to the light field. The values in this field are defined as the distance from *uv-plane* to the intersection point of *ray(s, t, u, v)* and the object surface.

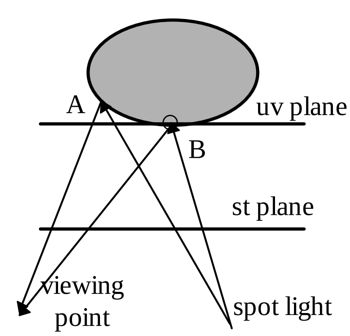


Figure 4 Incident angle should be determined by depth information in addition to the position of viewpoint and spotlight.

Suppose *P* is a point on the surface of an object exposed to two spot lights *l1* and *l2* with radiance I_{l1} and I_{l2} respectively. Then the diffuse reflection from *P* caused by *l1* and *l2* respectively are: $I_{l_1 d} = I_{l_1} k_d \cos\theta$ and $I_{l_2 d} = I_{l_2} k_d \cos\theta$, where θ is the incident angle variable. Obviously, for any θ,

$I_{l_1 d} / I_{l_2 d} = I_{l_1} / I_{l_2}$. In other words, the ratio of corresponding pixels' values in the diffuse light fields generated by different lights is constant. Therefore, if we establish a light field of a spotlight with unit intensity, the data of the diffuse light field with several spotlights may be multiplied by an appropriate factor. For most materials, the ratio of red, green and blue components in the diffuse reflection to white spotlight is the same or nearly the same as the ratio of the three elements in the original color. If ambient light is white, the ratio of the three-color elements of a surface point can be determined by the ambient light field of the object.

Suppose that, lighted by the white unit ambient light, the colors of the rays from point P is: $(R_A(P), G_A(P), B_A(P))$, lighted by a white spot light with intensity (1, 1, 1), the diffuse reflection at P is: $(R_d(P), G_d(P), B_d(P))$. Let:

$$L_d(P) = \begin{cases} \dfrac{R_A(P) + G_A(P) + B_A(P)}{R_d(P) + G_d(P) + B_d(P)}, & \text{if } R_d(P) + G_d(P) + B_d(P) \neq 0 \\ 0, & \text{else} \end{cases} \quad (2)$$

Then when P is exposed to light l of intensity (R_l, G_l, B_l), the diffuse reflection at P contributed by l is:

$$\left(R_{dl}(P), G_{dl}(P), B_{dl}(P)\right) = \left(R_l L_d(P), G_l L_d(P), B_{dl} L_d(P)\right) \quad (3)$$

Because of this, we just need to construct a diffuse light field for a unit-intensity white spot light and store only a ratio for each ray.

Specular reflection relies on not only incident angle, but the angle between reflection and viewing direction as well. Normal of the surface may be saved for computing specular reflection according to lighting model. In fact, specular reflection for most materials is too little to be considered. In this paper, specular light is omitted.

In conclusion, for each object, we will construct:

1) An ambient light field in which each point records R,G,B values;
2) A diffuse light field in which each point records a ratio Ld(P) defined in formula (3);
3) A depth field in which each point records the depth message.

3.2 Construction of Light Field

Sample points in the light field can be obtained directly by using ray-tracing scheme. However, this needs to revise the existent rendering flow and makes it unfeasible for some systems. According to Marc Levoy, the images in the scene can be regarded as two-dimensional slices in a four-dimensional space [1], thus the light field can be established by rearranging the pixels in a series of images.

The ambient light field can be constituted similarly to static light field. But in diffuse field, the viewpoint and the spot light should be overlapped while rendering, so as to ensure that all surface points in the object exposed to the spot light are visible in the images. The construction of depth field is similar. The 2-dimensional slices of depth field are obtained by recording and geometrically transforming the depth information while rendering images slices for ambient light field.

During image rendering, the center of the object should be put at the origin of the coordinate and uv and st planes parallel with coordinate axes. Moreover, the affection by other spotlights and the occlusion by other objects should be obviated.

3.3 Image Reconstruction

After the ambient light field, the diffuse light field and the depth field for all moving objects are constructed; the scene can be rendered according to the position and orientation of objects and spotlights. What is different from static scene rendering is that variation in reflection and shadow should be dealt with.

3.3.1 Sampling and superposing in light field

The procedure of light field rendering is similar to ray casting algorithm. Rays are emitted from the viewpoint and go through the center of each pixel on the image. Compute the coordinates of intersection of each ray with *uv*- and *st-plane*: (u_1, v_1, s_1, t_1), search for the color $L(u_1, v_1, s_1, t_1)$ in the ambient light field and depth information from depth field to find out the precise intersection of the incident ray with the object surface point P. Then calculate the *(u,v,s,t)* of the ray from point light l to P. The diffuse contribution by light l on P can be obtained from the diffuse light field related to the object and the light: $L_{dl}(u_2, v_2, s_2, t_2)$. Finally, the diffuse reflection is added to the ambient light, according to the shadow effect. The color of P is:

$$I_P = L(u_1, v_1, s_1, t_1) + \sum_l (I_{dl}(u_2, v_2, s_2, t_2) Shadow(P, l)) \quad (4)$$

In the above formula, when P is visible to l, *Shadow* (P, l)=1, otherwise, =0.

For efficiency, the coordinates of viewpoint and spot light should be translated and rotated first, to make the surrounding-plane pairs align with coordinate axes and the center of each object overlap the origin.

To reduce the run time computation of intersection, tables should be initiated in which objects are sorted according to the distance from object to view point and to spot lights. Suppose there are N_l spotlights, then $N_l + 1$ tables should be established, with the nearest object at the front. During rendering, compute the intersection of the rays with the objects according to the order in the table. If intersection occurred, there is no need to process more objects. That is, if the ray from the spotlight or the viewpoint is obstructed by some object, the computation of intersection will break, which is similar to "early ray termination".

3.3.2 shadow computation based on light field

The shadow discussed in this paper happens when a part of rays from a spot light are obstructed by an object, which makes other objects or other parts of the same object completely or partly unreachable from the light, hence resulting a darker area. So there are three steps to determine the shadow:

1) Decide whether there is any object intersecting with the line connecting the surface point and the spotlight.
2) Decide whether this point is obstructed by other parts of the same object. This information may be obtained from the depth field. As shown in Figure 5, if *d2<d1*, point A can not be lighted by the spot light. If and only if *d2=d1*, in other words, A overlaps B, does A expose to the point light.
3) Decide whether this point is within the lighting area of the light. To limit the lighting area will produce an effect of a searchlight.

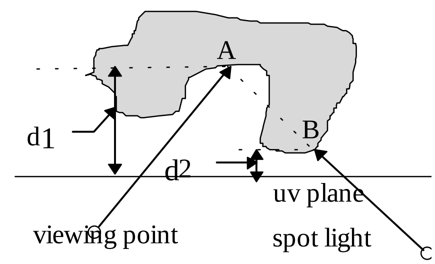


Figure 5

3.3.3 Summary of the rendering algorithm

Figure 6 shows the flow diagram of the rendering algorithm based on ambient light field, diffuse light field and depth field. All the variables are written in italic.

```
listObjectsFromView = buildSortList(viewing_point );
for(each spot light light){
        listObjectsFromLight[ light ] = buildSortList(light );
        for(each pixel ){
            cast ray from viewing_point and passing the center of pixel: ray1;
            for( each object in listObjectsFromView){
                (u1, v1, s1, t1) =the corresponding coordinates of ray1 in the ambient light field of object;
                Ipixel = LightAmbient( object, u1, v1, s1,t1);
                point = the actual intersection point of ray1 and the surface of object from depth field;
                d1 = DepthField(object, u1,v1,s2,t2);
                for(each spot light){
                    if (there is no other object between light and object) {
                    cast ray for light and passing point : ray2;
                    (u2, v2, s2, t2) = the coordinates of ray2 in diffuse light field of (object, light) pair;
                    d2 = DepthField(object, u2,v2, s2,t2);
                    if(d2==d1)
                        Ipixel += LightDiffuse(object, light, u2, v2, s2, t2);
}     }     }       }
```

Figure 6 The rendering algorithm based on ambient light field, diffuse light field and depth field.

In the diagram, *listObjectFromView* and *listObjectFromLight* represent the lists corresponding to the viewpoint and the spotlight respectively. These lists are produced in Function *buildSortList*. The value of the point indexed by *(u, v, s, t)* in ambient and diffuse light field of the object to the light can be obtained with function *LightAmbient(object, u, v, s, t)* and *LightDiffuse(object, light, u, v, s, t)*. The diffuse reflection is computed according to(3). Ipixel stands for the color of the pixel.

3.4 Data Compression

The weakness of light field rendering algorithm is that it requests enormous memory for light field. Fortunately, the data in light field are highly redundant and able to be compressed efficiently. Levoy *et. al*. got the compression rate over 100:1 by taking the advantage of vector quantification and entropy encoding. In the algorithm proposed in the paper, three groups of data similar in structure need to be used: ambient light field, diffuse light field and depth field. All the three types of fields are vectors quantified.

Because the object's reflections to ambient light and spot light are only parts of the total reflection and the values are greater than or equal to zero, the average amplitude of the data in ambient light field or diffuse light field is surely less than that of the static light field of the same scene. Therefore, the data of ambient light field and diffuse light field is easier to be compressed, or involved less distortion when compressed to the same rate as static light field.

Though the ambient light field has to record R,G,B values like the static scene light field, the diffuse light field only needs to record one ratio for each point. Furthermore, owing to the limited resolution of images, the value in depth field only requires definite precision and eight bits is enough.

Taking advantage of the similarity between ambient light field, diffuse light field and depth field, we merge the three into a new structure in which each element is a five dimensional vector composed of R, G, B color elements in ambient light field, a ratio element in diffuse light field and a depth element in depth field. While compressing the data, several points in the field are combined into a larger vector according to the resolutions of *uv*- and *st-plane*. For example, if four contiguous points are selected in *uv-plane*, *st-plane* respectively, 4*4*5= 80 dimensional vectors are constructed

Not only the object's surface color, but also its shape is recorded in static and ambient light field. Whereas diffuse light field only records surface normal and depth field only records depth information. Obviously when they are sampled at the same resolution, there will be more redundancy in diffuse light field and depth field and greater compression rate could be achieved.

According to the scheme in this paper, a set of light fields and a depth field must be established for each object. It seems that the data size would be enormous for the memory to hold. However, when there are several objects in the scene, the user seldom let one object occupies the whole view. Thus, the resolution of the object view could be far lower than that of the whole scene and the sampling density on *uv-plane* of an object's light field could be far lower than that of a similar static scene. If the user insists to view the object closely, the image quality has to be dropped.

4. Experimental Results

Table 1 Compression statistics of the objects in the scene

Object	Head	Upper body	Lower body	Upper arm	Forearm & hand	Thigh
surrounding plane pairs	5	5	4	5	5	5
sampling rate in uv plane	64*64	64*64	64*64	64*32	64*32	64*32
sampling rate in st plane	32*32	32*32	32*32	16*8	32*16	32*16
Raw Data	125.83	104.86	83.89	5.57	21.10	21.10
vector Dimension	80	80	80	20	20	20
number of codewords	16384	16384	16384	8192	16384	16384
size of codebook	1.37	1.31	1.31	0.16	0.33	0.33
index array size	2.29	2.29	1.84	0.49	1.84	1.84
total size	3.66	3.60	3.15	0.65	2.17	2.17
compression rate	34.4	29.1	26.6	8.6	9.7	9.7
Object	crus	Foot	Wall	Lower column	Upper column	Ceiling & Floor
surrounding plane pairs	5	6	4	4	4	2
sampling rate in uv plane	64*32	16*16	1024*512	512*64	32*32	1024*1024
sampling rate in st plane	16*8	16*16	1*1	1*1	16*16	1*1
Raw Data	5.57	1.97	2.10	0.13	6.55	2.10

vector Dimension	20	20	16		20	16
number of codewords	8192	4096	16384		8192	16384
size of codebook	0.16	0.08	0.26		0.16	0.26
index array size	0.49	0.15	0.17		0.49	0.23
total size	0.65	0.23	0.43	0.13	0.65	0.49
compression rate	8.6	8.6	4.9	1	10.1	4.3

This algorithm is implemented on a Pentium 133 with 32M memory. Figure 7 displays several images of the dynamic scene tested. The resolution of image is 320*240. It is easy to see that this algorithm properly expresses variation in reflection and shadow. In the scene, no object generates specular reflection, but the images still have strong sense of reality. In Figure 7(b), as a spotlight varies from white to blue, a blue spot is formed on the floor.

Each part of the body is regarded as an independent object. Figure 8 shows a 2-dimensional slices of the objects' ambient light field and diffuse light field. The ambient light field chiefly shows texture (color) feature of objects and the diffuse reflection light field contains the information of surface normal.

It costs about 0.2 second in average to render the dynamic scene with light field technique discussed in this paper on Pentium 133; whereas it will cost over 3 minutes on the same machine with geometry-based methods.

The total size of the light field data of all the objects in the scene is 17.98 MB.

5. Conclusion

The conceptions of ambient light field, diffuse light field and depth field are defined in this paper. An algorithm based on these light fields is also presented which expands the light field rendering from the static scenes to dynamic scenes. This algorithm achieves interactive speed on PCs for the scene used in the paper, complicated enough for the application of general virtual environments.

The basic idea in this paper is to divide the whole light field of the scene into several sub-light fields, which do not change with the variation of scenes except for coordinate transformations. Thus rendering the scene only needs to sample and superimpose the sub-light fields. The shadow computation is contained in the computation of the diffuse light field.

Although the time-complexity of the algorithm is related to the number of objects and lights in the scene, the complexity of shapes and textures of objects does not affect rendering speed. Compared with static light field rendering, this algorithm not only needs some extra addition and multiplication, but also more times to find out intersections and to access the memory during image reconstruction. Although these operations slow down the rendering to some extent, they make light field rendering applicable for dynamic scenes and the rendering speed is still much faster than geometry based rendering.

If transparency is included in light field, the algorithm needs to be slightly modified to which we will pay attention in the future. The fact that our algorithm needs depth information is a deficiency, however we can apply the depth information to image interpolation so as to devise a hybrid system of light field rendering and image interpolation technique which, on one hand, enjoys the high speed of light field rendering while on the other alleviates the burden of memory.

References

[1] Levoy, M. and Hanrahan, P. Light Field Rendering. In Computer Graphics. Annual Conference Series, 1996, pp. 31-42.

[2] Gortler, S., Grzeszczuk, R., Szeliski, R., Cohen, M. The Lumigraph. In Computer Graphics. Annual Conference Series, 1996, pp. 43-54.

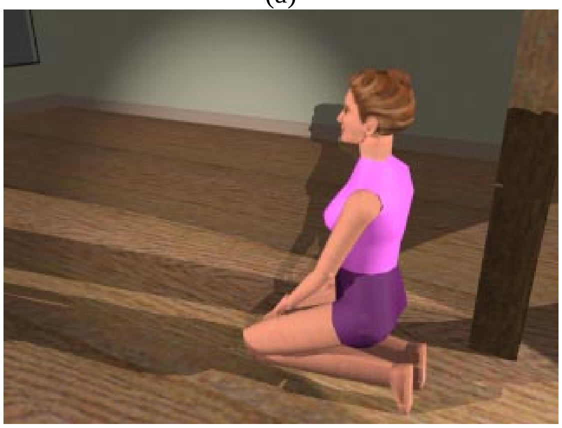
[3] McMillian, L., and Bishop G. Plenoptic Modeling: An Image-Based Rendering System. In Computer Graphics. Annual Conference Series, 1996, pp. 39-46.

[4] Chen, S. E., QuickTime VR-an image-based approach to virtual environment navigation. In Computer Graphics. Annual Conference Series, 1995, pp. 29-38.

[5] Chen, S. E., and Williams, L. View interpolation for image synthesis. In Computer Graphics. Annual Conference Series, 1993, pp. 279-288.

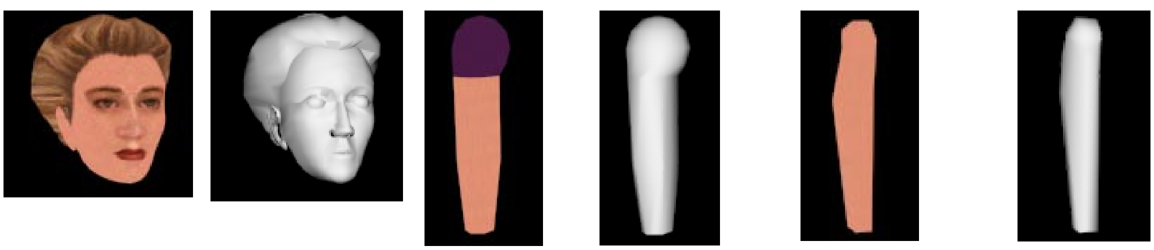

(a)


(b)


(c)


(d)

Figure 7 Dynamic scenes rendered with light field


head thigh crus

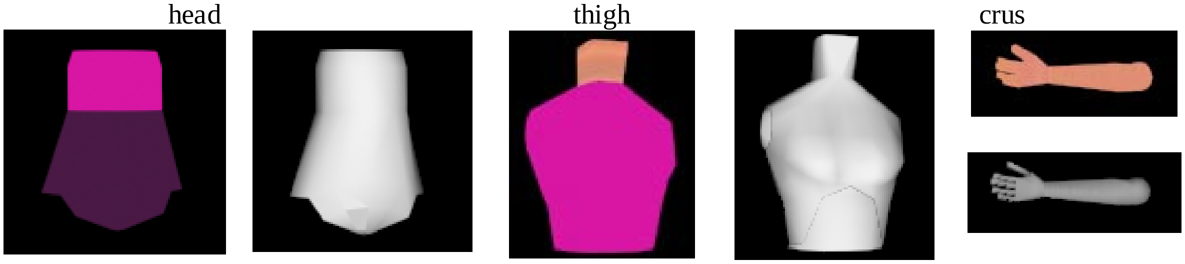

lower body upper body forearm and hand

Figure 8 2-D image slices in ambient light field and diffuse light field of partial objects in the dynamic scene.