EUROGRAPHICS '98 / N. Ferreira and M. Göbel
(Guest Editors)

Volume 17, (*1998*), *Number 3*

Optical Flow Rendering

Tae-Joon Park[†], Seungyong Lee[‡], and Sung Yong Shin[†]

[†]Department of Computer Science, Korea Advanced Institute of Science and Technology, Taejon, Korea
[‡]Department of Computer Science and Engineering, Pohang University of Science and Technology, Pohang, Korea

Abstract

This paper proposes a new approach to image-based rendering that generates an image viewed from an arbitrary camera position and orientation by rendering optical flows extracted from reference images. To derive valid optical flows, we develop an analysis technique that improves the quality of stereo matching. Without using any special equipments such as range cameras, this technique constructs reliable optical flows from a sequence of matching results between reference images. We also derive validity conditions of optical flows and show that the obtained flows satisfy those conditions. Since environment geometry is inferred from the optical flows, we are able to generate more accurate images with this additional geometric information. Our approach makes it possible to combine an image rendered from optical flows with an image generated by a conventional rendering technique through a simple Z-buffer algorithm.

Keywords: image-based rendering, optical flow, stereo matching, depth information, warp function, statistical analysis, image composition

1. Introduction

Recently, demands for real-time high quality image generation have ever increased in various applications of computer graphics. Although advanced rendering techniques such as ray-tracing and radiosity make it possible to generate photo-realistic images, it is still hard, if not impossible, to generate such images in real-time with those techniques. Furthermore, they require a lot of human effort to build complex environment geometry.

This has motivated a new approach called *image-based rendering*. In this approach, an image viewed from an arbitrary position and orientation is interpolated using scene information extracted from a set of reference images, and thus image generation time only depends on image size. If real photographs are used as reference images, photo-realistic images can be obtained. Hence, image-based rendering is appropriate to generate realistic images of a real environment without any geometric data.

McMillan[13] insisted that image-based rendering is a reconstruction procedure of an unknown plenoptic function, that defines the radiant energy incident to an eye position through an angle of every incident direction in an environment. In this point of view, reference images are samples of the plenoptic function. From those samples, the plenoptic function is interpolated by an image-based rendering technique.

One approach to reconstruct a plenoptic function is to generate a warp function that describes the relative movement of each pixel with respect to camera movement[3, 4, 10, 12, 13]. Stereo matching techniques are employed to derive warp functions between image pairs[13]. Since the amount of a warp for an object is inversely proportional to the object depth, we can obtain a warp function from depth information captured through special hardware such as Z-buffers[3, 4, 12] or range cameras[10]. The warp function approach requires a relatively small amount of storage. However, the performance of this approach depends on the quality of warp functions. For a real environment whose exact geometry can not be captured easily, it is hard to compute reliable warp functions.

Another approach is to construct a light field that approximates the plenoptic function of an environment[6, 7, 11, 16, 17]. A light field consists of samples that describe intensities of lights incident through predefined directions. Maintaining a sufficiently large number of samples, these approaches can

interpolate images viewed from an arbitrary camera position and orientation. However, the size of samples tends to be too large to handle. Recently, researches to reduce sample size have been reported. Pulli et al.[16] proposed a method that reduces the size of samples using rough geometric information obtained by range cameras. Sloan et al.[17] improved the lumigraph system[6] for time-critical applications. Gu et al.[7] characterized an object point in relation to ray samples in a light field. This characterization seems very promising to extract geometric information from the light field and to eliminate duplicated data.

An environment map is a sample of a plenoptic function at a fixed viewpoint. In the QuickTime VR system[2], Chen used a set of environment maps to construct an image-based navigation system. In the system, cylindrical images taken at regular grid points are used as plenoptic samples. Szeliski[18, 19] proposed a method to construct an environment map by image mosaics. Although these approaches can generate very realistic images, positions and orientations of viewpoints are restricted.

In this paper, we propose a new approach to image-based rendering, called *optical flow rendering*, that generates images using optical flows extracted from reference images. We also derive conditions for valid optical flows and present an analysis technique to construct a set of valid optical flows. This technique constructs reliable optical flows from a sequence of matching results between reference images. Those optical flows are rendered to generate an image viewed from an arbitrary camera position and orientation. Since an optical flow gives the depth of an object point, the set of all optical flows for a scene approximates environment geometry. Hence, optical flow rendering can generate realistic in-between images by interpolating reference images with this additional geometric information. Our approach makes it possible to combine an image obtained by optical flow rendering with that generated by a conventional rendering technique through a simple Z-buffer algorithm.

The remainder of this paper is organized as follows. Section 2 describes the representation of an optical flow set and its validity conditions. Construction and rendering of a valid optical flow set are explained in Section 3 and Section 4, respectively. Section 5 presents experimental results. In Section 6, we conclude this paper.

2. Representation of an Optical Flow Set

2.1. Reference Image Acquisition

Consider a uniform 2D grid embedded in a plane that has $n \times n$ grid points in it (Fig. 1). We assume that the horizontal and vertical distances between adjacent grid points are all the same. This distance is called the *baseline* of the grid. Reference images are captured at the grid points with the same camera focal length and orientation. $I_{i,j}$ denotes the reference image taken at a grid point $(i, j), 1 \leq i, j \leq n$.

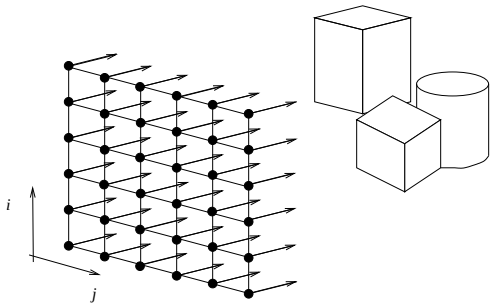


Figure 1: *Reference image acquisition system*

The pixel at (x,y) in $I_{i,j}$ whose intensity is c is denoted by $I_{i,j}(x,y,c) = (i, j, x, y, c)$.

We also assume that the x-axis of the image plane is parallel with the horizontal direction of the grid. Therefore, an object point appears at the pixels that have the same y-coordinate in a horizontal sequence of reference images. Symmetrically, this point also appears at the pixels that have the same x-coordinate in a vertical sequence of reference images.

2.2. Optical Flows

2.2.1. Optical flow representation

Let an object point p be mapped onto a pixel (x,y) in a reference image $I_{i,j}$. Suppose that the point p appears at a horizontally displaced position (x',y) in an adjacent reference image $I_{i,j+1}$. Since reference images are acquired at the points on a uniform grid with the same orientation, the displacement $d = (x' - x)$ is invariant regardless of j. Similarly, the vertical displacement $d' = (y' - y)$ of two pixels (x,y) and (x,y') in a pair of adjacent images, $I_{i,j}$ and $I_{i+1,j}$, is also invariant for any i. Moreover, the horizontal and vertical displacements d and d' are the same because the image acquisition grid has the same horizontal and vertical spacing. We call this displacement the *disparity* of the point p.

The disparity d can be computed from the depth information of p by[5]

$$d = \frac{F \times B}{Z}, \tag{1}$$

where F is the focal length of the camera, B is the baseline of the image acquisition grid, and Z is the depth of the point p from the plane containing the grid. Note that the disparity d is inversely proportional to the depth Z of p.

An *optical flow* describes a sequence of pixels in reference images, that correspond to the same object point in the environment. In particular, a horizontal optical flow f corresponding to an object point p gives a sequence of pixels, at most one per reference image, in the same row of the image acquisition grid. Let the pixel sequence given by f consists

of pixels from reference images $I_{i,j}$ for $j_0 \le j \le j_1$. Then, we represent f as follows:

$$f = (d, I_{i,j}(x_0, y_0, c_0); [j_0, j_1]), \tag{2}$$

where d is the disparity of p, and $I_{i,j}(x_0, y_0, c_0)$ is the pixel of intensity c_0 at (x_0, y_0) in $I_{i,j}$. $I_{i,j}(x_0, y_0, c_0)$ is the reference pixel of f from which the other pixels in the sequence are computed. With a proper interpretation, f can also be used to represent a vertical optical flow. In this paper, however, f refers to a horizontal flow unless explicitly stated otherwise.

2.2.2. Pixel sequence

The optical flow $f = (d, I_{i,j}(x_0, y_0, c_0); [j_0, j_1])$ of an object point p determines the pixel at which p appears in each of reference images that belong to the ith row of the grid between the j_0th and the j_1th reference images, inclusively. The horizontal *pixel sequence* $PS(f)$ of the optical flow f is defined as follows:

$$\begin{aligned} PS(f) \quad = \quad & \{I_{i,j'}(x, y_0, c) | x = x_0 + \lfloor (j' - j) \times d + 0.5 \rfloor \\ & \text{and} \quad j_0 \le j' \le j_1\}, \end{aligned} \tag{3}$$

where $\lfloor\ \rfloor$ denotes the floor function.

If the intensity of p is varied because the specular reflection or p is occluded by other objects, pixels at which p appears may not be covered with a pixel sequence of a single optical flow. In this case, p is represented by more than one optical flows that may have different pixel sequences and intensities.

2.3. An Optical Flow Set

2.3.1. Visibility relationship

We can establish an occlusion relationship between two optical flows. Let $f = (d, I_{i,j}(x_0, y_0, c_0); [j_0, j_1])$ and $f' = (d', I_{i',j'}(x'_0, y'_0, c'_0); [j'_0, j'_1])$ be two given optical flows. If the following two conditions

$$d < d' \tag{4}$$

and

$$PS(f) \cap PS(f') \neq \phi \tag{5}$$

hold, we say that f is *occluded* by f'. Eq. (4) means that f' is closer to the camera than f. With the optical flow f', the pixel sequence $PS(f)$ is classified into two sets:

$$PS(f) = PS^{f'}_{vis}(f) \cup PS^{f'}_{inv}(f), \tag{6}$$

where

$$PS^{f'}_{inv}(f) = PS(f) \cap PS(f') \tag{7}$$

and

$$PS^{f'}_{vis}(f) = PS(f) - PS^{f'}_{inv}(f). \tag{8}$$

Given a set S of optical flows, we can classify $PS(f)$ into two sets with respect to S:

$$PS(f) = PS^{S}_{vis}(f) \cup PS^{S}_{inv}(f), \tag{9}$$

where

$$PS^{S}_{inv}(f) = \bigcup_{f' \in S} PS^{f'}_{inv}(f) \tag{10}$$

and

$$PS^{S}_{vis}(f) = PS(f) - PS^{S}_{inv}(f). \tag{11}$$

2.3.2. Validity conditions

In image-based rendering, unknown scene information is extracted from reference images and used to generate an image viewed from a new camera position and orientation. Since complete scene information is not available, there are no concrete measures to check the validity of interpolated images. However, we can easily give a constraint that every image-based rendering technique must satisfy. That is, the interpolated image at a grid point should be identical with the reference image at the same point.

Suppose that we have a set S of optical flows generated from reference images. We can reconstruct the reference images from the set S by converting each optical flow f in S to pixels in $PS(f)$ by Eq. (3). We can apply painter's algorithm for hidden surface removal to resolve the overlapping pixels from different optical flows. If the optical flows in S are scanned in the increasing order of disparity, pixels from far-away flows are overwritten by pixels from nearby flows if they overlap.

A set of optical flow is said to be *valid* if it can exactly reconstruct all reference images. Let RP be the set of all pixels in the reference images. To reconstruct whole pixels in reference images, S must satisfy

$$RP \subseteq \bigcup_{f \in S} PS(f). \tag{12}$$

We classify the pixel sequence $PS(f)$ of a flow $f = (d, I_{i,j}(x_0, y_0, c_0); [j_0, j_1])$ into two subsets:

$$PS(f) = PS_{true}(f) \cup PS_{false}(f), \tag{13}$$

where

$$\begin{aligned} PS_{true}(f) \quad = \quad & \{I_{i,j'}(x, y, c) | I_{i,j'}(x, y, c) \in PS(f), c = c_0 \\ & \text{and} \quad j_0 \le j' \le j_1\} \end{aligned} \tag{14}$$

and

$$PS_{false}(f) = PS(f) - PS_{true}(f). \tag{15}$$

The pixels in $PS^{S}_{inv}(f)$ are occluded by other flows, and thus only the pixels in $PS^{S}_{vis}(f)$ contribute to reference image reconstruction. This gives an additional condition for a valid optical flow set S as follows:

$$PS^{S}_{vis}(f) \subseteq PS_{true}(f) \text{ for all } f \in S. \tag{16}$$

Eq.'s (12) and (16) together give our validity conditions for the optical flows. Eq. (12) guarantees that no pixels in any reference images are missing. Eq. (16) forces the reconstructed images at the grid points to be identical with the reference images themselves.

3. Construction of an Optical Flow Set

3.1. Stereo Matching

In order to find disparities of object points, we perform stereo matching[5, 8, 9, 14] between each pair of adjacent reference images $I_{i,j}$ and $I_{i,j+1}$. The stereo matching between $I_{i,j}$ and $I_{i,j+1}$ can be described as a mapping $M_{i,j}$:

$$M_{i,j} : I \times I \rightarrow D. \tag{17}$$

$M_{i,j}(x,y)$ estimates the disparity d of a flow $f = (d, I_{i,j}(x,y,c); [j_0, j_1])$ at the pixel $I_{i,j}(x,y,c)$. Suppose that the pixel $I_{i,j}(x,y,c)$ is matched with $I_{i,j+1}(x',y,c)$. Then,

$$M_{i,j}(x,y) = x' - x. \tag{18}$$

From the Eq. (3), the true values x'^* and x^* of x' and x are in intervals $[x'-0.5, x'+0.5)$ and $[x-0.5, x+0.5)$, respectively, i.e.,

$$\begin{aligned} x'^* &\in [x'-0.5, x'+0.5), \text{ and} \\ x^* &\in [x-0.5, x+0.5), \end{aligned} \tag{19}$$

and thus

$$\begin{aligned} x'^* - x^* - 1 < M_{i,j}(x,y) &< x'^* - x^* + 1, \\ d - 1 < M_{i,j}(x,y) &< d + 1, \text{ or} \\ M_{i,j}(x,y) - 1 < d &< M_{i,j}(x,y) + 1. \end{aligned} \tag{20}$$

Consider a 2D plane spanned by two orthogonal axes: one for the column index j of a reference image $I_{i,j}$ and the other for the x-coordinate of a pixel. For a fixed i, a point (j,x) in this plane gives the x-coordinates of pixels $I_{i,j}(x,y,c)$ for all $j_0 \le j \le j_1$. Therefore, a line l, that is given by $x = d \times j + x_0$, enumerates the (j,x) pairs for all pixels in $PS(f)$ for $f = (d, I_{i,j}(x_0,y_0,c_0); [j_0, j_1])$. However, the exact value of d is hard to obtain. Let $I_{i,j}(x_0,y_0,c_0)$ be matched with $I_{i,j+1}(x,y_0,c_0)$. Then, the line segment ls joining two points (j,x_0) and $(j+1,x)$ should be near l. From Eq. (19),

$$x_0^* \in [x_0 - 0.5, x_0 + 0.5) \text{ and } x^* \in [x-0.5, x+0.5). \tag{21}$$

Therefore, x_0^* and x^* are represented as vertical intervals of unit length at the endpoints (see Fig. 2). In order to satisfy Eq.'s (19) and (21), l must intersect both of the vertical intervals.

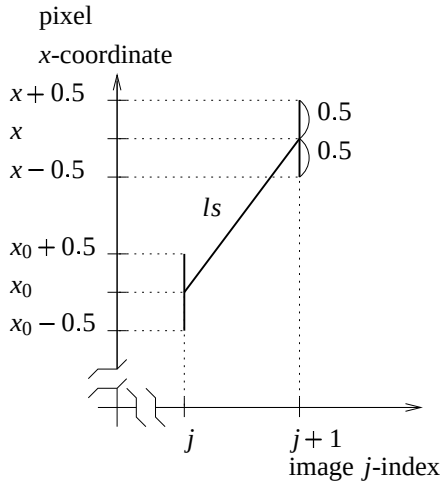


Figure 2: *Vertical intervals derived by a stereo matching,* $x = x_0 + M_{i,j}(x_0, y)$

3.2. Robust Generation of an Optical Flow

3.2.1. Matching chains

Let $M = (I_{i,j}(x_0,y_0,c_0), I_{i,j+1}(x_1,y_0,c_0), I_{i,j+2}(x_2,y_0,c_0), \cdots)$ be a sequence of pixels such that each pair of adjacent pixels are matched with each other, i.e.,

$$x_{k+1} = x_k + M_{i,j+k}(x_k, y_0) \text{ for all } k = 0, 1, 2, \cdots. \tag{22}$$

This sequence of pixels is said to be a *matching sequence.* We represent this sequence in the (j,x) plane as a sequence of line segments together with the unit vertical intervals at their endpoints (see Fig. 3). Each segment connects two point $(j+k, x_k)$ and $(j+k+1, x_{k+1})$ for $k = 0, 1, 2, \cdots$. In the ideal case, there exists a line intersecting all vertical intervals (see Fig. 3(a)). However, this is not the case, in general, due to specular reflection and occlusion (see Fig. 3(b)). A matching sequence may be divided into two or more subsequences, called *matching chains.* A matching chain C is a subsequence of a matching sequence that satisfies the following properties:

- There exists a line l that intersects all vertical intervals derived by stereo matching of adjacent pixels in C.
- C is not contained in another matching chain.

3.2.2. Feasible regions

To check if a matching sequence M is a matching chain, we need to find a line l that intersects all vertical intervals derived from stereo matching of adjacent pixels in M. To find such a line efficiently, we employ the notion of point-line duality that transforms a line given by $ax + by + 1 = 0$ in the plane into a point (a,b) in the dual plane and vise versa. For a simple polygon, it was proven that rays intersecting two edges of the polygon are transformed to a convex polygon in the dual space[1]. In the same way, all lines simultaneously intersecting two vertical intervals derived from a stereo matching $M_{i,j}(x,y)$ also form a convex region in the dual space. This convex region is said to be the *feasible region* of $M_{i,j}(x,y)$. That is, any line in the original plane represented

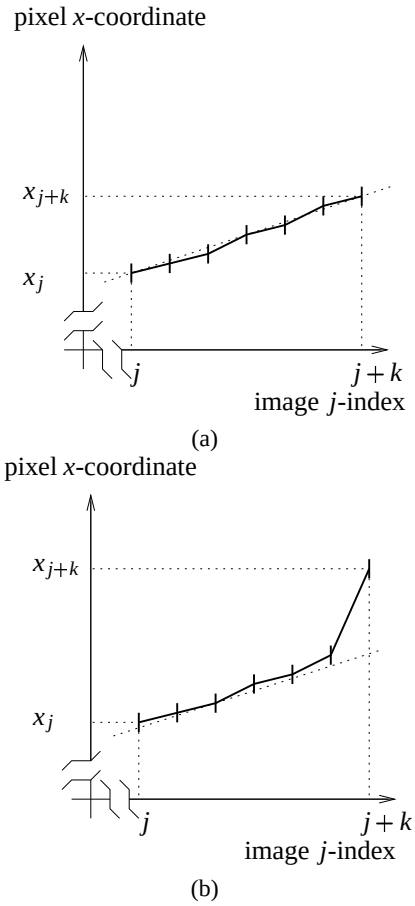


Figure 3: *Two matching sequences*

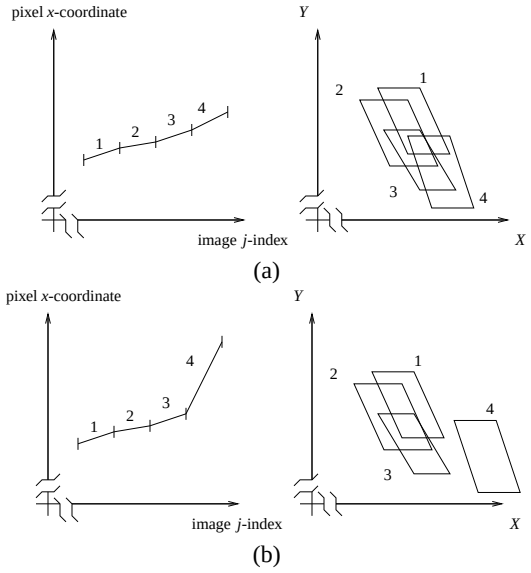


Figure 4: *Intersection test after dual transform*

by a point in the feasible region of a stereo matching map $M_{i,j}(x,y)$ can possibly be a true line l that gives $M_{i,j}(x,y)$.

Given a matching sequence M, we can find the joint feasible region for M by computing the intersection of the individual feasible regions of stereo matching maps derived from all adjacent pairs of pixels in M. As shown in Fig. 4(a), if the joint feasible region is not empty, then there exists a line l intersecting all vertical intervals simultaneously. Otherwise, such a line l does not exist (see Fig. 4(b)). We employ an efficient 2D convex polygon intersection algorithm to find the joint feasible region[15].

3.3. Generation of an Optical Flow Set

Given reference images, we construct matching chains as follows: First, every pixel in all reference images is marked *unused*. Then, all pixels in image $I_{1,1}$ are traversed row by row. The pixels in each row are scanned from left to right. For each pixel in $I_{1,1}$, we construct a matching sequence M starting from the pixel by adding pixels in the adjacent reference images one by one. By definition, a matching sequence of unit length is a matching chain. At each time when a pixel is added to M, M is tested if it is still a matching chain. If M is not a matching chain any more, we generate a new matching chain with the pixels in M except the last one and add it to the set of matching chains. The pixels in the new matching chain are marked *used*. Removing the pixels in the chain from M, we resume the construction of M.

After finishing the traversal of pixels in $I_{1,1}$, we have a set of all matching chains starting from all pixels in $I_{1,1}$. Then, the pixels in a reference image adjacent to $I_{1,1}$ are traversed in the same sequence to generate new matching chains. We generate matching sequences starting from only the pixels marked *unused* because the pixels marked *used* already belong to matching chains obtained previously. This process continues until the pixels in all reference images are marked *used*. Fig. 5 summarizes this procedure as a pseudo code.

Now, we show how to obtain an optical flow from a matching chain. Suppose that we have a matching chain $C = (I_{i,j}(x_0,y,c), I_{i,j+1}(x_1,y,c), I_{i,j+2}(x_2,y,c), \cdots, I_{i,j+e}(x_e,y,c))$. It is natural to choose, as the line l that intersects all vertical intervals for C, the line given by the center point (x_c,y_c) of the joint feasible region. That is, the equation of l is $x = -x_c/y_c \times j - 1/y_c$. Since this line intersects the pixel $I_{i,j}(x_0,y,c)$, an optical flow generated from C becomes

$$f = (-\frac{x_c}{y_c}, I_{i,j}(x_0,y,c); [j, j+e]). \tag{23}$$

Let SC be the set of matching chains generated by the procedure in Fig. 5. The pixels in the matching chains in SC covers all pixels in the reference images. Let S be the set of optical flows obtained by converting every matching chain

```
procedure make_matching_chain;
input: a set of reference images, {I_{i,j}};
output: a set of matching chains, SC;
begin
  mark all pixels in I_{i,j} unused;
  SC = empty_set;
  for (i, j) = (1,1) to (m,n) begin
    for all unused pixel (x,y) in I_{i,j} begin
      sequence M = empty_sequence;
      append I_{i,j}(x,y,c) into M;
      int check = 0; jj = j; xx = x;
      while check = 0 and jj ≠ n begin
        xx = xx + M_{i,jj}(xx,y);
        jj = jj + 1;
        if (xx,y) is inside of image begin
          append I_{i,jj}(xx,y,c) into M;
          joint feasible region test for M;
          if M ≠ a matching chain begin
            remove (xx,y) from M;
            check = 1;
          end
        else
          check = 1;
      end
      chain C = M;
      mark all pixels in C used;
      insert C to the set SC;
    end
  end
end
```

Figure 5: *A procedure for generating a set of matching chains*

C in SC. S satisfies Eq. (12) because $PS(f)$ of a flow f in S contains all pixels in the corresponding chain C. Also, every flow f in S trivially satisfies Eq. (16) from the definition of a matching chain. Hence, S is a valid optical flow set.

4. Rendering of an Optical Flow Set

4.1. Reference Image Reconstruction

Let S be the set of optical flows constructed by the technique presented in Section 3. Then, we can easily reconstruct all reference images from S by scanning the flows in S in the increasing order of disparities. For each flow f in S, we compute the pixel sequence $PS(f)$ by Eq. (3). Pixels in $PS^S_{inv}(f)$ are overwritten by other flows. Since S satisfies Eq.'s (12) and (16), we can reconstruct reference images without any holes.

To increase the rendering speed, we use a set of buckets $B_{i,j}$, one per reference image, that contains the optical flows to reconstruct the reference image. An optical flow $f = (d, I_{i,j}(x,y,c); [j_0, j_1])$ is pointed by buckets $B_{i,j'}$ for $j_0 \leq j' \leq j_1$. When a reference image $I_{i,j}$ is reconstructed, it is sufficient to consider only the flows referenced by the bucket $B_{i,j}$.

4.2. In-between Image Generation

Let (i_f, j_f) for $i_f, i_f \in R$, be an arbitrary viewpoint on the plane that contains the image acquisition grid. To generate an in-between image I_{i_f,j_f} without holes, we use the post-warping algorithm proposed by Mark et al.[12]. To apply this algorithm, we first reconstruct four reference images, $I_{i,j}, I_{i+1,j}, I_{i,j+1}$, and $I_{i+1,j+1}$, such that $i \leq i_f \leq i+1$ and $j \leq j_f \leq j+1$. The disparities of all pixels in these four reference images are derived from the optical flows in S.

We then generate four intermediate images, $I^1_{i_f,j_f}, I^2_{i_f,j_f}, I^3_{i_f,j_f}$ and $I^4_{i_f,j_f}$, by applying the post-warping algorithm to $I_{i,j}, I_{i+1,j}, I_{i,j+1}$, and $I_{i+1,j+1}$, respectively. Overlapping pixels in the intermediate images are resolved by scanning optical flows in the increasing order of disparities. Finally, the in-between image I_{i_f,j_f} is obtained by taking the weighted average of those four intermediate images through an α-blending hardware in a graphics workstation. The weights for intermediate images are calculated from the distances from (i_f, j_f) to $(i,j), (i+1,j), (i,j+1)$, and $(i+1, j+1)$, respectively. The disparity at each pixel in I_{i_f,j_f} is also determined by taking the weighted average of the disparities of the corresponding pixels in the intermediate images.

4.3. Rendering with an Arbitrary Camera Position and Orientation

If we know the intensity of every ray incident to a given camera position, it is possible to generate an image viewed from the position with an arbitrary camera orientation. We show how to construct all such rays and their intensities. Since we know the disparity of an optical flow, we can compute the coordinates of the object point that corresponds to the flow. Thus, given an optical flow set S, we can produce a set of all rays incident to a camera position. The intensity of a ray can also be given by its corresponding optical flow.

We now describe how to generate an image with the incident rays. If only one ray hits a pixel, the intensity of the pixel is easily defined. If there are no rays incident to a pixel, its intensity is given by its adjacent pixels. Let $f = (d, I_{i_0,j}(x,y,c); [j_0, j_1])$ be an optical flow. For a given camera position, suppose that a ray r produced by f intersects the plane containing the image acquisition grid at (i_f, j_f) for $i_f, j_f \in R$. Since the flow f is defined for a sequence of reference images $I_{i_0,j}$ for $j_0 \leq j \leq j_1$, the ray r is valid at (i_f, j_f) for $i_f = i_0$ and $j_0 \leq j_f \leq j_1$. Suppose that two or more such rays intersect the plane at the same point (i_f, j_f). Then, one of them contains the others, and thus they hit the same pixel. If there are rays valid at (i_f, j_f), the intensity of the pixel is determined by disparity comparison between optical flows that produce those rays. If there are no

	lion	hall	room
number of flows	642312	628378	636498
average length of chains	3.81	4.22	3.84
number of pixels	2359296	2359296	2359296

Table 1: *Experimental statistics*

valid rays, then the intensities of rays valid at adjacent grid points $(i, j), (i+1, j), (i, j+1)$, and $(i+1, j+1)$ are blended for filling the pixel, where $i \leq i_f \leq i+1, j \leq j_f \leq j+1$.

Since the optical flows are maintained in a set of buckets, we can easily access optical flows that produce rays valid for each grid point. Using the view volume of a given camera, we can choose small number of buckets that may contain all optical flows to generate the image viewed from the camera. Thus, we can produce all rays efficiently without traversing whole optical flows.

5. Experimental Results

We use an array of 6×6 images of size 256×256 as a set of reference images. The lion images are real images captured by a special equipment[11]. The others are synthesized images rendered by conventional rendering techniques. However, we do not use any geometric information of the model in the experiments.

Fig.'s 6, 7, and 8 show examples of images generated from the optical flow sets. Images at four corners are reference images that are reconstructed. The others are in-between images obtained by interpolating four reference images. Since an optical flow set gives depth information of an object point, interpolated images correctly reflect the viewpoint changes and look realistic.

Table 1 shows the number of optical flows and the average length of matching chains for the examples in Fig.'s 6, 7, and 8. The number of flows are much smaller than that of pixels in whole reference images. Thus, optical flows represent reference images more compactly. Since we use an array of 6×6 reference images, the length of a matching chain should be an integer from one to six. The average chain length gives how many pixels in reference images are connected into a matching chain on average.

In Fig. 9, we show an example of image composition. Fig. 9(a) is a background image rendered from an optical flow set. Fig. 9(b) is an image obtained by a conventional rendering technique. The depth information of pixels in Fig. 9(b) is obtained during rendering the scene. Then, the images in Fig.'s 9(a) and (b) are combined by a Z-buffer. Fig. 9(c) shows the resulting image.

6. Conclusion

In this paper, we proposed a new approach to image-based rendering that uses optical flows. An analysis technique is proposed to obtain a set of valid optical flows without any special equipments such as range cameras. The proposed technique distinguishes reliable scene information from noise by analyzing a sequence of matching results between reference images. Thus, it works robustly even with matching errors.

Since a set of optical flows gives geometric information of an environment, optical flow rendering can generate realistic images by interpolating reference images with this geometric information. Furthermore, optical flow rendering has a strong representation power. Representing an object point as one or more optical flows that may have different pixel sequences and intensities, we can handle specular reflection and occlusion that have been hard to handle. Hence, we can apply the optical flow rendering to generate images of a real environment, where a stereo matching technique may not work well.

By connecting several pixels into a matching chain, we combine multiple pixels into a single flow. Thus, optical flows can represent scene information compactly. Since the image generation speed mainly depends on the number of optical flows, we will try to reduce the number of flows while preserving scene information. We have observed that an unexpected matching error results in a wrong matching sequence from which several matching chains are produced. Therefore, we may reduce the number of flows by analyzing such matching errors and connecting the wrongly separated matching chains into a correct one.

Acknowledgments

We thank to the light field project group of Stanford University for opening their experimental results to the public. We used in the experiment the lion images obtained from their web page. We also thank to Mr. Hyung-jin Ha for his kind and helpful comments and suggestions.

References

1. B. Chazelle and L. J. Guibas. Visibility and Intersection Problems in Plane Geometry. *Discrete & Computational Geometry*, 4(6):551–581, 1989.
2. S. E. Chen. QuickTime VR - An Image-Based Approach to Virtual Environment Navigation. In *Computer Graphics(ACM SIGGRAPH '95)*, pages 29–38, 1995.
3. S. E. Chen and L. Williams. View Interpolation for Image Synthesis. In *Computer Graphics(ACM SIGGRAPH '93)*, pages 279–288, 1993.
4. L. Darsa, B. C. Silva, and A. Varshney. Navigating

Static Environments using Image-Space Simplification and Morphing. In *Proceedings 1997 Symposiums on Interactive 3D Graphics*, pages 25–34, 1997.

5. O. Faugeras. *Three-Dimensional Computer Vision : A Geometric Viewpoint.* MIT Press, 1993.
6. S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. The Lumigraph. In *Computer Graphics(ACM SIGGRAPH '96)*, pages 43–54, 1996.
7. X. Gu, S. J. Gortler, and M. F. Cohen. Polyhedral Geometry and the Two-Plane Parameterization. In *Rendering Techniques '97(Proceedings of the Eurographics Workshop)*, pages 1–12, 1997.
8. T. Kanade. Development of a Video-Rate Stereo Machine. In *Proceedings of 94 ARPA Image Understanding Workshop*, pages 549–558, 1994.
9. T. Kanade, H. Kano, S. Kimura, A. Yoshida, and K. Oda. Development of a Video-Rate Stereo Machine. In *Proceedings of International Robotics and System Conference(IROS'95)*, pages 95–100, 1995.
10. T. Kaneko and S. Okamoto. View Interpolation with Range Data for Navigation Applications. In *Proceedings of Computer Graphics International 1996*, pages 90–95, 1996.
11. M. Levoy and P. Hanrahan. Light Field Rendering. In *Computer Graphics(ACM SIGGRAPH '96)*, pages 31–42, 1996.
12. W. R. Mark, L. McMillan, and Gary Bishop. Post-Rendering 3D Warping. In *Proceedings 1997 Symposiums on Interactive 3D Graphics*, pages 7–16, 1997.
13. L. McMillan and G. Bishop. Plenoptic Modeling : An Image-Based Rendering System. In *Computer Graphics(ACM SIGGRAPH '95)*, pages 39–46, 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. F. P. Preparata and M. I. Shamos. *Computational Geometry; An Introduction.* Springer-Verlag, 1985.
16. K. Pulli, M. F. Cohen, T. Duchamp, H. Hoppe, L. Shapiro, and W. Stuetzle. View-based Rendering : Visualizing Real Objects from Scanned Range and Color Data. In *Rendering Techniques '97(Proceedings of the Eurographics Workshop)*, pages 23–34, 1997.
17. P.-P. Sloan, M. F. Cohen, and S. J. Gortler. Time Critical Lumigraph Rendering. In *Proceedings 1997 Symposiums on Interactive 3D Graphics*, pages 17–23, 1997.
18. R. Szeliski. Video Mosaics for Virtual Environments. *IEEE Computer Graphics & Applications*, 16(2):22–30, March 1996.
19. R. Szeliski and H.-Y. Shum. Creating Full View Panoramic Image Mosaics and Environment Maps. In *Computer Graphics(ACM SIGGRAPH '97)*, pages 251–258, 1997.

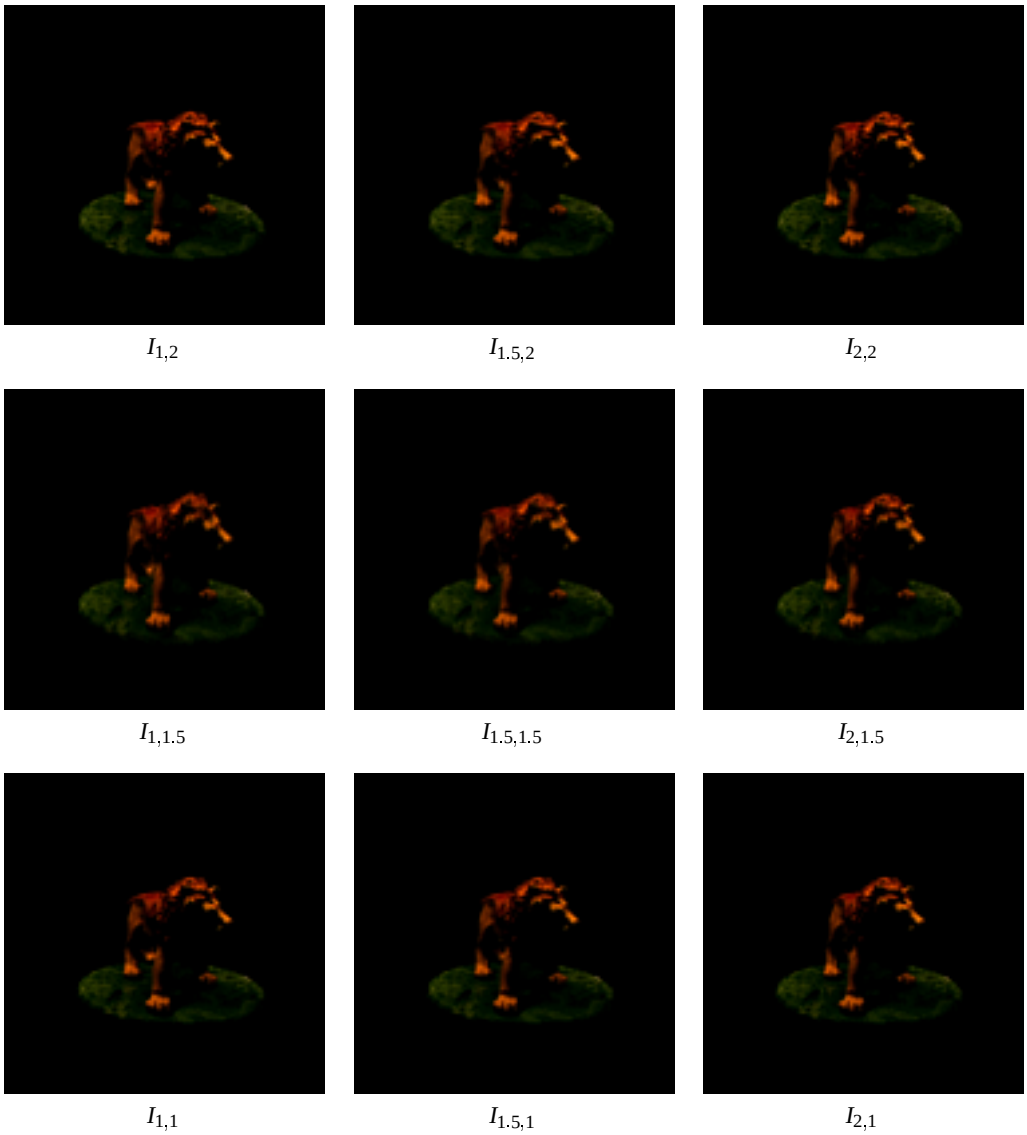


Figure 6: *Image generation example I: lion*

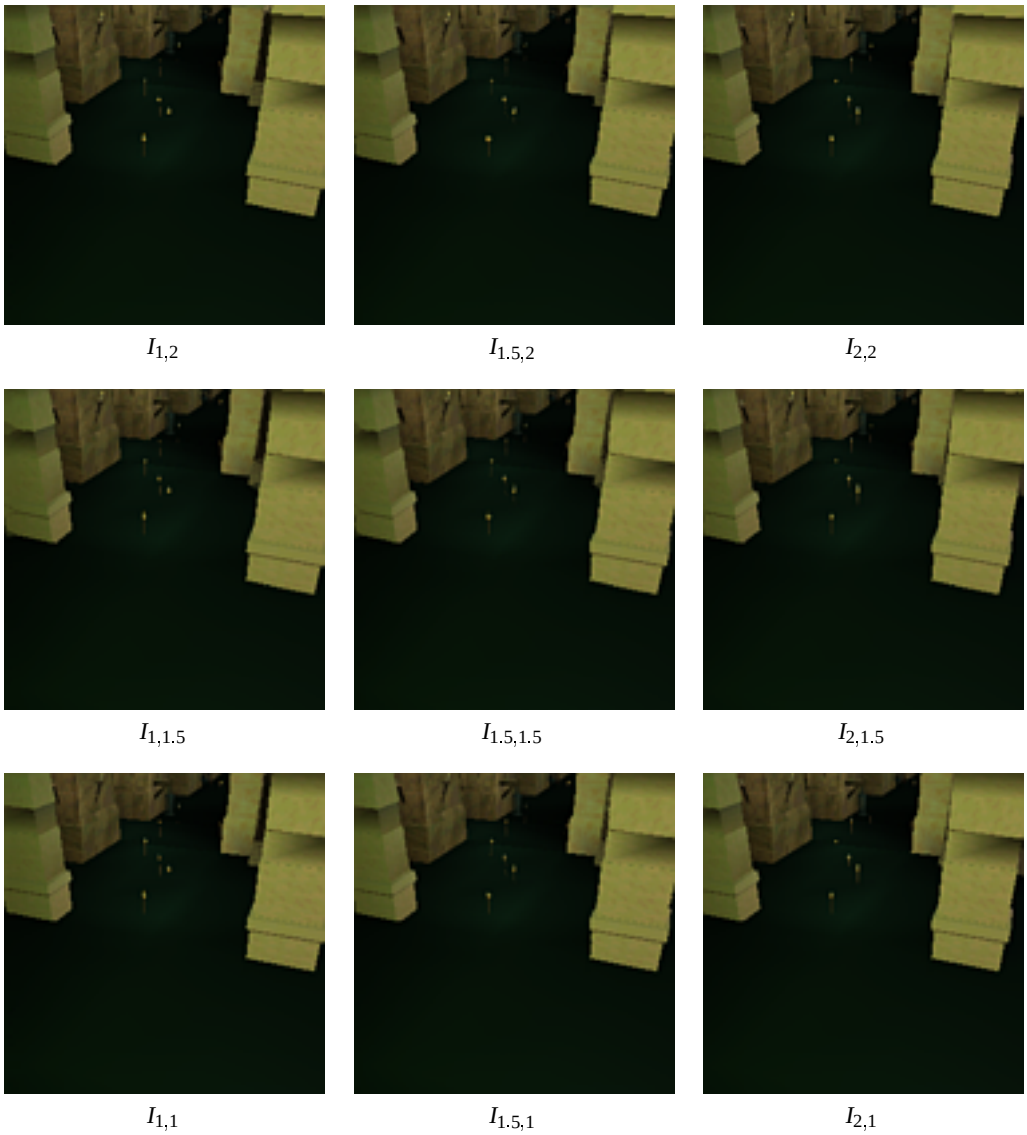
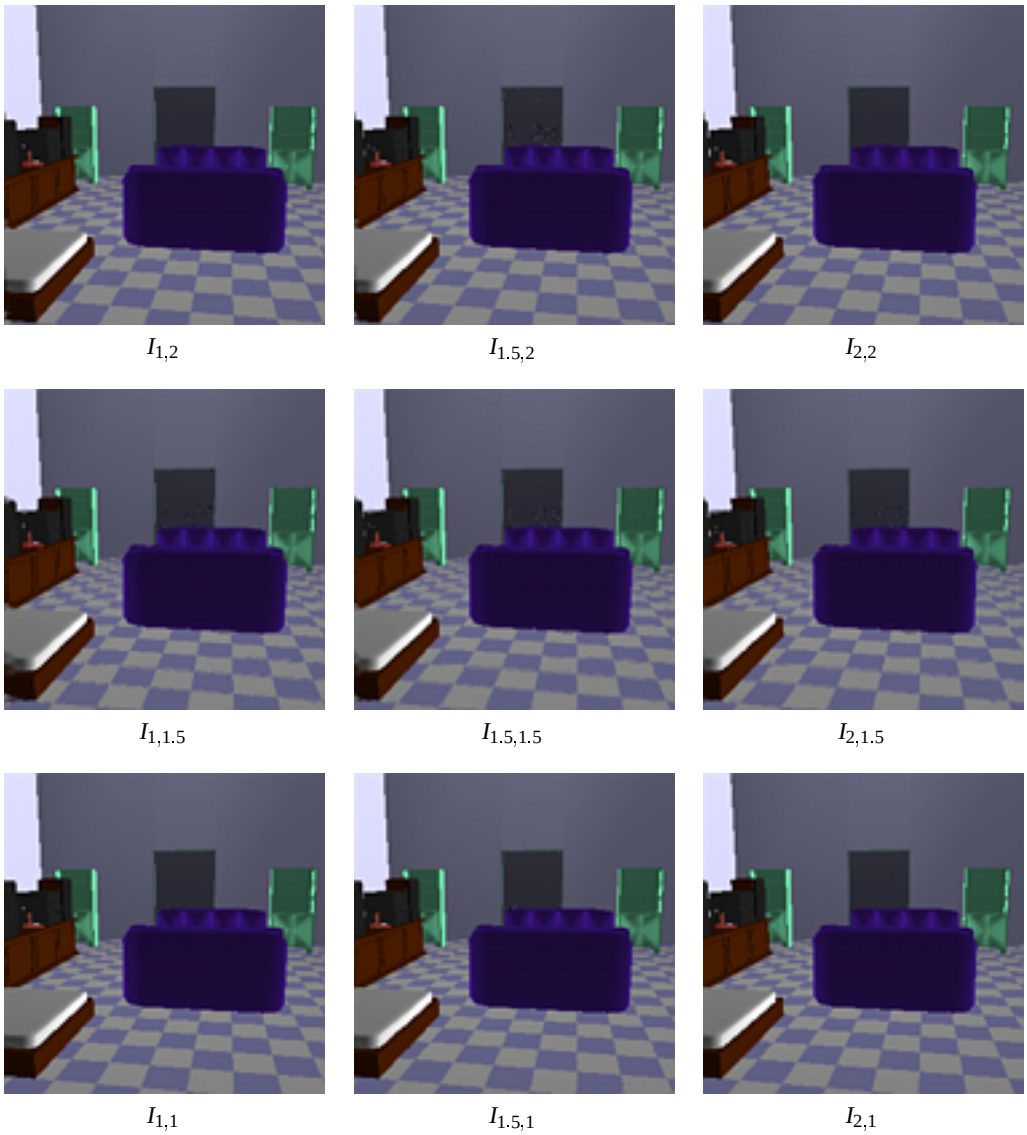


Figure 7: *Image generation example II: hall*



$I_{1,2}$ $I_{1.5,2}$ $I_{2,2}$

$I_{1,1.5}$ $I_{1.5,1.5}$ $I_{2,1.5}$

$I_{1,1}$ $I_{1.5,1}$ $I_{2,1}$

Figure 8: *Image generation example III: room*

(a) (b) (c)

Figure 9: *Image composition example*