LIGHTFIELD COMPLETION

By

Liron Yatziv

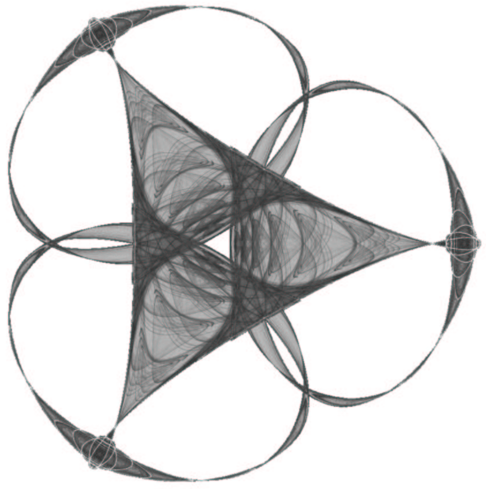
Guillermo Sapiro

and

Marc Levoy

IMA Preprint Series # 1957

(February 2004)



LIGHTFIELD COMPLETION[1]

Liron Yatziv, Guillermo Sapiro
Electrical and Computer Engineering
University of Minnesota

Marc Levoy
Computer Science
Stanford University

ABSTRACT

A light field is a 4D function representing radiance as a function of ray position and direction in 3D space. In this paper we describe a method for recovering gaps in light fields of scenes that contain significant occluders. In these situations, although a large fraction of the scene may be blocked in any one view, most scene points are visible in at least some views. As a consequence, although too much information is missing to employ 2D completion methods that operate within a single view, it may be possible to recover the lost information by completion in 4D - the full dimensionality of the light field.

The proposed light field completion method has three main steps: Registration, initial estimation, and high dimensional texture synthesis and/or inpainting. At the registration stage, the set of images are shifted and re-projected so that the corresponding pixels from different images are aligned in the reconstructed light field. Following this, the estimation step uses a naive technique to fill-in parts of gaps using the available information from the multiple images. This serves as the initial condition for the next and last step, where the missing information is recovered via high dimensional texture synthesis and/or inpainting. These two steps of initial condition and completion are iterated. The algorithm is illustrated with real examples.

1. INTRODUCTION

Scenes are frequently observed with significant occlusions between the camera and the objects of interests. When multiple views of the scene are available, from different camera positions, digitally removing the occluding objects becomes a feasible task following the appropriate integration of information from the whole data set. In this paper we propose an algorithm to accomplish this goal.

Our objective is then to remove occluders from a 3D/4D light field. The light field is made out of an array of images where the angle difference between two adjacent ones is relatively small [1] (see also [2]). The occluder can be of complex shape, such as foliage, and may have a large perspective size, but must be at a different depth than the occluded object. Our method confronts difficult cases where more is occluded than is revealed, and regular 2D texture synthesis [3-5] and image inpainting [6] methods perform poorly. Such methods fail mainly because the known data is scattered in the light field and there are no available large areas of information as in ordinary applications. Moreover, the whole occluded scene is not completely revealed by the multiple views, and simple integration operations across images, such as the ones proposed in [7, 8], fail as well. There is then a need to work with the complete 3D/4D space, as proposed in this work.

2. OUR APPROACH

We represent the light field as a 4D array (s,t,u,v), as suggested in [1]. The single view images are in the (s,t) plane and the (u,v) coordinates represent the camera translations in a plane, where the camera is always looking ahead toward the object, see Figure 1. The completion process is done by first registering the light field, then creating a median image as initial guess, and finally filling-in the gaps using a guided, edge oriented, texture synthesis technique. In the concluding remarks we explain the possibilities to add 4D inpainting to this approach.

2.1. Registration

Registering the light field is needed when the occluded object shows changes in angle and/or translation within the light field. The registration process receives as input the raw light field (Figure 1a) and transforms it so all corresponding pixels of the images ((s,t) plane slices) are aligned as much as possible in the s and t axes (Figure 1b). In general, the registration involves re-projection as explained in [1] and the classical computer vision literature (see Figure 2 for an example). In our examples, the focal plane is selected to be the plane of the center image. Note that for this registration step, we can use fully automatic registration algorithms particularly developed to handle significant occlusions, [9], and the large vision literature in the subject.

2.2. Median image

Once the light field is registered, a median computation is preformed on the pixels color in the u and v axes directions, creating a new single 2D image we call the *median image*. The median image is likely to have gaps in it, since parts of the scene might have never been covered by the light field, see Figure 1c,d. We also give a confidence value $C_M \in [0,1]$ to each pixel in the median image, where gaps have C_M=0 (no confidence) and the rest of the data gets values in the (0,1] range. This confidence value is based on the amount of non occluded pixels in the u and v axes and on their similarity to the computed median color. We chose the calculation of the confidence as follows, though other choices may be used as well (see concluding remarks section):

[1] Work supported by NSF, ONR, NSF CAREER Award, and PECASE. We thank M. Bertalmio and K. Patwardhan for conversations on image inpainting and completion.

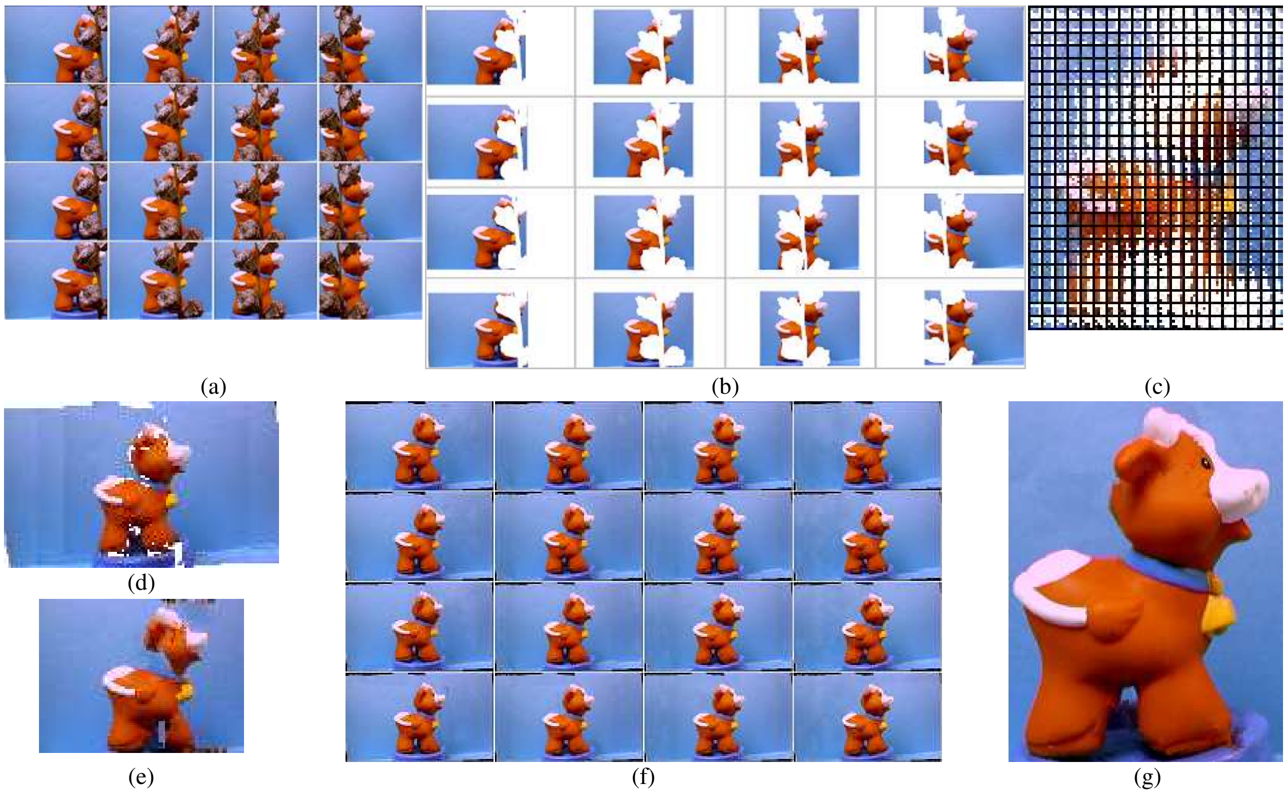


Figure 1: *Examples of the completion process for data with small camera motions. (a) Raw light field. A 3D eucalyptus branch occludes a 3D toy cow. (b) Branch removed and light field registered. Each image is extended in size and the original image is placed so that the cow stays in the same location in all images. (c) Each small square represents the rays leaving one point on the (s,t) plane bound for all points on the (u,v) plane (each square then gives the different views values for a given pixel). Only 1% of the rays are shown, and they visualize the data available per pixel, white representing lack of data in that particular view (due to occlusion). The median image is created by taking the median in each such square. (d) Median image. (e) A sample image from the light field when running the texture synthesis without any initial estimation. (f) Completed light field using the same parameters as in (e) but with the initial estimation. (g) Zoom in on one of the final images in the completed light field.*

$$C_M = \Pr_M \cdot D_M$$

$$D_M = \frac{|UV|}{U + V - 1} \;;\; \Pr_M = 1 - \frac{\sum_{p_i \in ST} \left(I(p_i) - I(p_M)\right)^2}{|UV| \cdot \frac{I_{\max}^2}{2}}$$

Here, *Pr* represents the pixel similarity, D_M is the data factor ($D_{M,}\ \Pr_M \in [0,1]$), *UV* is the set of not occluded pixel colors along the *u* and *v* axes, *I*(*Pm*) is the median color, and *U* and *V* are the number of pixels in the light field along the *u* and *v* directions respectively. As can be seen in Figure 1e, using this median image as initial condition is crucial when significant chunks of data are missing in the 4D light field, see next.

2.3. Texture Synthesis

Finally, we fill-in the gaps in the light field using a texture synthesis technique, extending the priority ideas proposed in [10] (which is in itself an extension of [3] that incorporates ideas from inpainting [6]). In [10], a temporary priority value was introduced, which determines the order in which gaps are filled. The priority is based upon the confidence of the neighborhood and the data term, which is given by the strength of isophotes hitting the gap's border (inspired by [6]). Extending this idea to higher dimensions is straightforward and therefore not detailed here. We should only note that for simplicity, we kept the computation of the data term in the 2D (*s,t*) plane.

2.3.1. Initial estimation

When the gaps are fairly large, the Markovian filling-in process loses reliability and performs poorly as it advances away from the original data. To address this problem, and inspired in part by [11], we fill the gaps with the median image as an initial estimation. Thus, colors are copied from the same (*s,t*) coordinate in the median image to the corresponding gaps in the light field and the appropriate confidence C_M is attributed to those pixels as described above. Pixels having confidence smaller then 1 are considered not final. Since the gaps maybe fairly large, this ini-

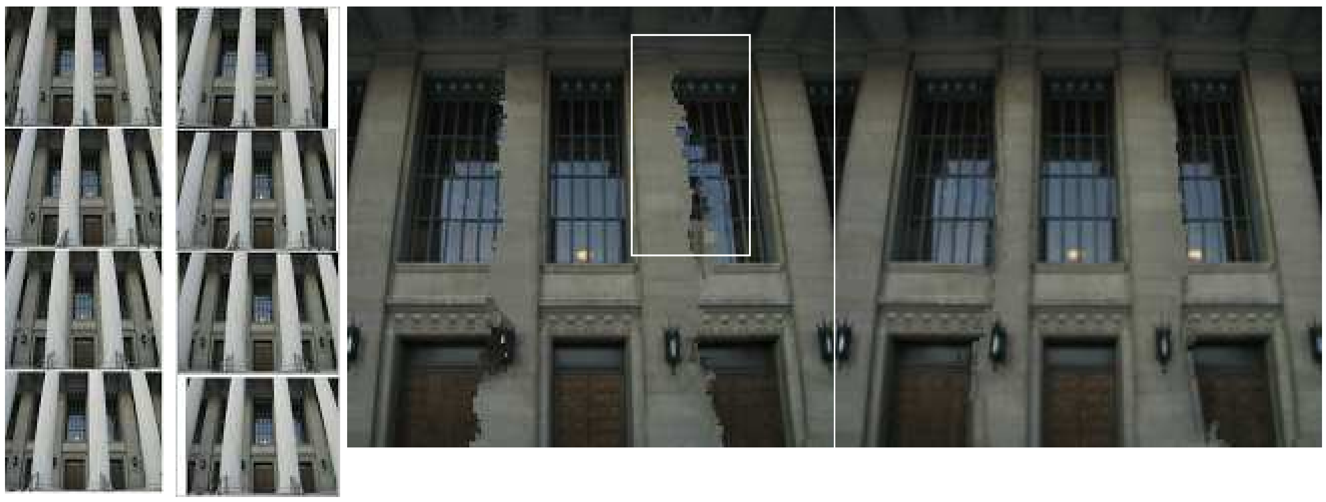


Figure 2: *Sample images from a 3D light field (i.e,. v axis is 1 image wide) of an horizontal 10 images set. The goal is to remove the building columns. (a) raw images (b) registered by re-projecting (c) center image when running the texture synthesis without any initial estimation. (d) center image with our approach, where the columns have been removed and the building is now visible.*

tial estimation is crucial to keep the texture synthesis process from getting overruled by noise, Figure 1e.

2.3.2. Nearest neighborhood

After computing the initial condition, the texture synthesis process is done similarly to [10], but in the higher full light field dimension, and with a different distance function. Instead of using the simple sum squared differences (SSD) as in [10], we take into account non final pixels and partial neighborhoods. We use a weighted sum squared differences where the product of corresponding pixel's confidence is used as weight. Since for the challenging data we address, the gap shape is large and complex, the data is scattered and it is very unlikely that many full 4D neighborhoods will be available for comparison (being this the main reason why ordinary texture synthesis fails). We are forced then to look into partial neighborhoods (neighborhoods that contain gaps in them). The distance is then weighted by the squared total weight in order to give advantage to more complete neighborhoods:

$$d(\psi_p,\psi_q)=\frac{\sum\limits_{p_i'\in\psi_p,q_i'\in\psi_q} C(p_i')C(q_i')\big(I(p_i')-I(q_i')\big)^2}{\Big(\sum\limits_{p_i'\in\psi_p,q_i'\in\psi_q} C(p_i')C(q_i')\Big)^2}$$

Here, p_i' and q_i' are the ith corresponding pixels, $I(p_i)$ is the color intensity of p_i, $C(p_i)$ is the confidence of p_i, and ψ_p is the neighborhood of p.

2.3.3. Placement of new patch

The closest neighborhood found is considered the most reliable, but there are cases where other neighborhoods are nearly as reliable as this one. Thus, pixel values from such neighborhoods are averaged with the pixel confidence as weight and placed in the gap, overriding any initial estimation. After this, a new confidence value is assigned to changed pixels as described in [10]. This process gradually achieves the replacement of any initial estimation or gap by information that fits best the known data. The averaging contributes in two ways: The degree of reliability increases, and several partial neighborhoods can complete one another so the amount of filled pixels increases.

2.3.4. Iterations

To further improve the quality of the results we iterate the process, using the texture synthesis results as the initial estimation for the next iteration. Each iteration increases the confidence of the pixels in the gap, since more information is available to start with. In theory, the process can be stopped when all the pixel confidences are high enough. In practice, due to computational time constrains, few iterations are sufficient for good results.

3. EXAMPLES AND COMPUTATIONAL CONSIDERATIONS

The complete process of completion can be seen in Figure 1, which uses a 3D occluder. The restored light field contains completed images that are similar though not identical, since there is a slight change in angle. Figure 2, is a real world scene example where the occluding columns have been removed. Figure 3 shows an additional example, with a real 2D occluder. This time the light field is larger but the occluder is scattered and covers most of the image. We should note that all the examples used exactly the same parameters (e.g., neighborhood size).

The texture synthesis process as presented in [10] performs an exhaustive search before each patch/pixel is placed, and has the additional overhead of priority calculations, which makes it slower than [3], who's run-time is proportional to the sizes of the image and the gap. A 4D light field can easily reach enormous size. For example, the light field in Figure 1 is considerably small and contains only 16 images, making a total of

(a)

(b)

(c)

(d)

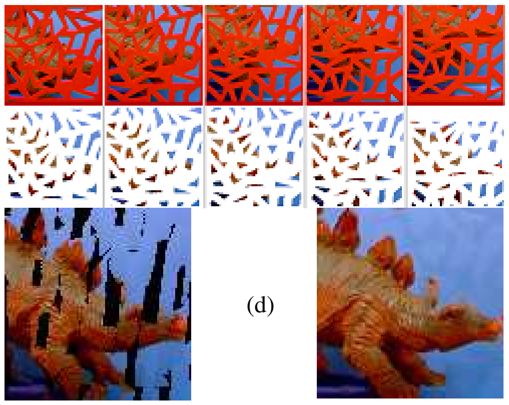


Figure 3: *Sample images from a 3D light field (i.e,. u axis is 1 image wide) of a vertical 37 images set, where a red cut piece of paper is in front of the object. Each image was taken when the camera had a different vertical position. The Stegosaurus (the occluded object) head is closer to the camera than the rest of the body. (a) The raw light field. Both the screen and the Stegosaurus shift between images. (b) Images registered only by translation and the occluder paper removed. (c) Median image. Seems like a good estimation but contains faults due to the changes in angle (e.g., middle plate, nostrils). (e) Sample result image. Most faults were fixed. The minor faults could have been avoided with a more accurate re-projection.*

1,024,000 pixels, out of which 260,538 are occluded (before extending). When dealing with such large amounts of data and a large amount of pixels to fill, the computational time becomes a major issue. A straight forward implementation of the texture synthesis algorithm results in an unbearable computational cost (days of computation). Unfortunately, the need to deal with partial neighborhoods restricts us from using techniques developed in part to reduce this computational cost, such as image quilting [4] and tree-structured vector quantization [5].

We addressed this problem partially by taking advantage of the unique properties of a registered light field, where corresponding pixels would remain locally in the (s,t) plane. Therefore, instead of an exhaustive search over the whole light field, we limited the search in the (s,t) plane to be local, and do search the complete u and v directions. If no satisfactory match is found, the search continues to the rest of the light field. This approach reduces the computational cost by 2-3 order of magnitudes, from days to hours.

4. CONCLUDING REMARKS

In this paper we have introduced a method for completing a light field. The method addresses cases where the occluded regions are large, and successfully completes the light field.

In cases where the gaps in the light field are smaller than those here considered, other known space-time video completion methods can be used effectively. Often, using the immediate boundaries of the gaps is sufficient. For this, the inpainting technique proposed in [6], which is restricted to 2D, can be generalized to higher dimensions in the following way:

$$\frac{\partial I}{\partial t} = -\left|\nabla(\Delta I)_{\nabla^{\perp} I}\right| \cdot \left|\nabla I\right| \cdot \operatorname{sgn}\left(\nabla\left|\nabla I\right| \cdot \nabla(\Delta I)_{\nabla^{\perp} I}\right)$$

Here, $\nabla(\Delta I)_{\nabla^{\perp} I}$ is the projection of the gradient of the Laplacian to the plane perpendicular to the gradient. The idea is to select the perpendicular to the gradient that is the projection of $\nabla(\Delta I)$ and change its direction to where the image is less sharp. We plan to combine this high dimensional inpainting with the texture synthesis approach here described, following [12], to further improve the results. We also want to investigate replacing the initial guess given by the median filter and the binary masks per image with matting techniques from computer graphics [13]. We also plan to extend the work here presented to moving objects. Results on this will be reported elsewhere.

11. REFERENCES

[1] M. Levoy and Pat Hanrahan. “Light field rendering,” In Proc. SIGGRAPH 1996, pages 31-42, August 1996.

[2] S. Gortler, R. Grzeszczuk, R. Szeliski, and M. Cohen, “The Lumigraph,” ,” In Proc. SIGGRAPH 1996, August 1996.

[3] A. Efros and T. Leung. “Texture synthesis by non-parametric sampling.” In Proc. ICCV, pp. 1033–1038, Kerkyra, Greece, Sep 1999.

[4] A. Efros and W. Freeman. “Image quilting for texture synthesis and transfer,” In Proc. SIGGRAPH ’01, Los Angeles California, August, 2001.

[5] L. Wei and M. Levoy. “Fast texture synthesis using tree-structured vector quantization,” In Proc. SIGGRAPH 2000, pp. 479-488, July 2000.

[6] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. “Image inpainting,” In Proc. SIGGRAPH 2000, New Orleans, July 2000.

[7] M. Irani and S. Peleg, “Motion analysis for image enhancement: Resolution, occlusion, and transparency ,” Journal of Visual Communication and Image Representation, Vol. 4, No. 4, pp. 324-335, December 1993.

[8] A. Duci, A. Yezzi, S. Mitter, and S. Soatto, “Region matching with missing parts,” Proc. of the ECCV, LNCS, Springer Verlag, May 2002.

[9] K. Habuka and Y. Shinagawa, “4D light field interpolation using a mechanism to control optical flow, preprint, 2004.

[10] A. Criminisi, P. Perez, and K. Toyama. “Object removal by exemplar-based inpainting,” IEEE CVPR 2003, Madison, Wisconsin, US, June 2003.

[11] I. Drori, D. Cohen-Or, and H. Yeshurun. “Fragment-based image completion,” SIGGRAPH 2003, San Diego, July 2003.

[12] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, “Simultaneous structure and texture image inpainting,” IEEE Trans. Image Processing 12, pp. 882-889, 2003.

[13] T. Porter and T. Duff, “Compositing digital images,” Proc. SIGGRAPH 1984, pp. 253-259, 1984.