Head-Tracked Light Field Movies: A New Multimedia Data Type

Gavin S. P. Miller, Steven M. Rubin, Philip M. Hubbard,
Jenny Dana, and John Woodfill

Interval Research Corporation, 1801 Page Mill Road, Building C,
Palo Alto, California 94304
{Miller, Strubin, Hubbard, Jdana, Woodfill}@Interval.com

Abstract. This paper presents a new multimedia data type based on time-varying light fields. Such a light field movie may be precomputed using computer graphics, or photographically captured from real objects as they change over time. Such movies allow a user to experience motion parallax in response to head motion in addition to motion cues within the scene as well as stereo vision cues. Important elements to make such media practical include light field movie capture devices, effective compression schemes with real-time playback, and accurate head tracking. This paper addresses these areas and presents results from the current system as well as directions for future work.

1 Introduction

As the area of multimedia matures, desktop video has become a well-supported data type with both hardware support (such as MPEG2) and efficient software-based algorithms such as vector quantization [1]. However, even the best video systems (analogue or digital) do not create a flawless illusion of looking through a window at a real object. This is due to the absence of certain visual cues, two of the most important of which are stereo disparity (derived from a different image at each eye) and motion parallax in response to head motion. As the power of personal computers continues to advance it is appropriate to consider multimedia data types that accommodate these cues. We start by examining prior work.

"Fish tank" virtual reality [2] (and its horizontal counterpart a "visual workbench" [3]) has been presented as a way to create the illusion of a 3-D world using a conventional flat display. In such a system, a user's head is tracked, and 3-D graphics is used to render the appropriate image to be presented to each eye, using stereo glasses. While such systems can create a convincing effect they have two severe limitations: the images must be rendered with small enough latency for the perceived world to remain spatially stable as the user moves his or her head, and the scenes need be represented using 3-D models. The rendering capability of the hardware places a constraint on the maximum visible scene complexity and it is very hard to create convincing models of real-world scenes and people.

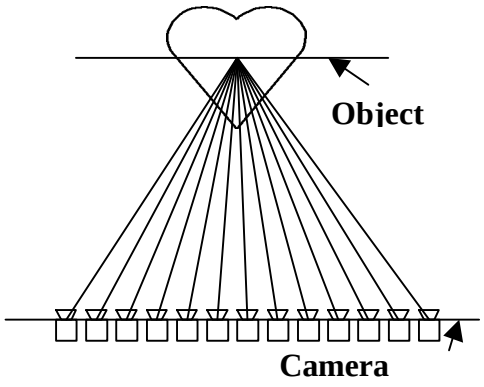
These constraints may be overcome using image-based rendering such as light fields [4] or lumigraphs [5]. (These, in turn, can be seen as generalizations of QuickTimeVR object movies [6], with sufficiently dense sampling of orientation

space to allow interpolation of views by blending.) By combining light field rendering with head-tracking and stereo display we can achieve a form of virtual holography, in which a user's head position determines the view presented to each eye. This paper describes such a system for light fields. By extending light fields to be time varying, we create "light field movies".

An ideal light-field movie system would consist of a camera that can observe an object or a scene, and then create the illusion of viewing that scene remotely, either live or from a recording medium. (Such scenes could also be rendered synthetically.) The scene, when viewed, would provide a convincing illusion of depth. To see behind an object, a user would move to one side to gain a different point of view. Specular materials would glint and sparkle in response to head motion and both eyes would fuse their views to allow the perception of a flawless 3-D world.

1.1 Capturing Light Field Movies

A capture device for such a system is illustrated in Fig. 1. A linear array of cameras is directed towards a target object (the heart) centered on the object plane. The cameras would be separate physical devices, for a live video system. However, when the scene is static or only changes slowly, a single camera may be used in combination with a positioning table. Such an arrangement is shown in Fig. 2 and Video 1 [17]. The camera is translated by a linear table and rotated, to track the center of the object.



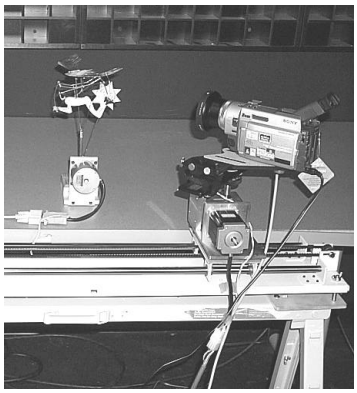


Fig. 1. Light field capture geometry

Fig. 2. Linear light field gantry filming "Icarus"

During image capture, the camera translates laterally. Since the camera also tracks the object, a re-mapping must be used to compute the image as if projected back onto the object plane.

1.2 Displaying Light Field Movies

To view light field movies requires a display that shows an image to each eye that has been reconstructed from the appropriate slices of camera data. This can be done in one of two ways. The display can project light in a view-dependent fashion so that the current eye position selects the appropriate slices (like a conventional hologram), or we can track the user's eye positions and display to each eye an image dependent on that eye's location. The optical modulation scheme has the advantage that the user does not need to wear special glasses, and there can be more than one viewer. However, special optical elements are required [7][8] and the entire light field video stream must be decompressed to modulate the display. This leads to prohibitive bandwidth requirements, equivalent to hundreds of channels of uncompressed video. The tracking-based approach only requires the rendering of two images per time step. To implement such a system requires an appropriate display, a head-tracking device and a graphics system fast enough to refresh the images with imperceptible lag. We will consider each piece in turn.

Stereo display can be achieved using a conventional cathode ray tube in combination with stereo glasses. If the use of glasses is considered undesirable, then an autostereoscopic display [9] may be employed, although effective systems are not currently available commercially. The rendering subsystem would compute the appropriate image for each eye with minimal latency and high frame rate. A software-based motion-light-field renderer is on the edge of being practical for such a task. An algorithm for this is described in Sections 2 and 3. This opens up the prospects of supporting light field stills and movies as first class multimedia data types, once an acceptable tracking system is perfected. Recent research suggests that the perceptible lag for such a system may need to be as low as 10 to 15 ms [10], which implies the need for hardware rendering support to achieve imperceptible latency. A high accuracy, low latency head-tracking system is described in Section 4.

2 Light Field Geometry and Sampling

This section describes the geometry and generation of the light fields for this paper.

2.1 Light Field Geometry

Levoy and Hanrahan [4], and Gortler et al. [5] described a way to render light fields using rays indexed by their intersection with two planes. We place one of the two planes of the light field to be coincident with the screen pixels. (This is identical to the geometry required to generate a slit hologram [11].) Illustrated in Fig. 3.

Given a new viewing location L, we project a ray from a screen pixel through L back onto the capture plane. This gives us the P, Q parameters for the captured image for which the corresponding X, Y pixel holds the correct view of the scene. By similar triangles P and Q can be seen to vary linearly with screen location X and Y.

A light field with only horizontal motion parallax may be captured by moving the capture camera along a linear trajectory parallel to the imaging plane. Such images may be rendered or captured using a gantry such as that in Fig. 2. Moving off this

trajectory in depth (towards and away from the screen) will only lead to mild projection errors. Moving up and down leads to more pronounced artifacts, but this happens less readily, when the user is sitting in front of a monitor on a chair.

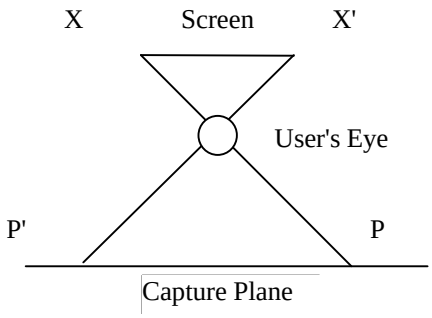


Fig. 3. Plan view of capture camera versus view location

2.2 Motion Light Field Generation

Motion light fields may be rendered or captured photographically. Plates 1 and 2 show stills from computer generated light field movies (Videos 2 and 3 [17]). We also wanted to capture motion-light fields of real objects. Our approach was to move the objects slowly enough to be captured by the gantry. Stop frame animation was achieved using a mechanical automaton (Icarus) under stepper motor control. A full sweep was made of the gantry, for each stepper motor position. Each sweep involved 100 images (in P) with 6 sub-steps averaged for depth of field filtering. The animation can be seen in Video 4 [17]. Plate 3a illustrates the center view, and the view constructed from the far right-hand camera position, after a corrective projective mapping. Plate 3b and Video 5[17] illustrates melting ice to generate motion. A longer-term goal is the construction of a light-field movie camera.

3 A Software Renderer for Motion Light Fields

We are interested in implementing light field movie playback on current personal computers. To achieve this requires both efficient resampling of the light field data (based on the current viewing location) and efficient decompression schemes.

3.1 Constant Background Optimization

To take advantage of regions of unchanging background, we constructed a “foreground map” which represented all pixels that were not background for at least one of the orientations. Plate 1 shows views of a time-varying ornament along with its foreground map. During display, any pixels not tagged by this map will not change

from the background color. Such regions do not require light-field samples, nor do they require resampling during rendering. Using this optimization, rendering performance for the decorative ornament was doubled compared to light fields that updated all of the screen pixels. Plate 2a represents a motion light field for which the foreground optimization is ineffective. The shark, during its motion, touches almost all of the foreground pixels.

3.2 Coherence for Efficient Resampling

Given the screen-aligned parameterization of Section 2.1, the question arises of the most efficient way to resample the light field. For a one-degree-of-freedom light field (i.e. those with only horizontal motion parallax) a single interpolation is required rather than the bilinear interpolation required for 2 degrees of freedom. It is worth noting for efficiency, that computational gains are possible by computing a column of pixels at a time. Since P changes linearly with X, P is constant for a given column of pixels. By arranging X, Y images to be contiguous in Y, it is possible to achieve good memory caching performance along a column. Additional savings may be achieved by using cheaper forms of blending such as 1-bit interpolation, where two values are averaged, if the blend value is more than one half, otherwise the first value is copied. This leads, on average, to a reduction in rendering time of one third. Plate 4 shows a comparison of blending with 1-bit interpolation.

The current implementation generates an off-screen buffer and then copies that buffer to the screen. The maximum frame rate for the pixel copy was approximately 78 frames per second, which significantly degraded the performance of our algorithms. All timing for light fields in this paper correspond to gray-scale stereo images (since we used red-green stereo for display) and are for a Xeon-based personal computer running at 400 MHz.

3.3 Light Field Movie Compression

For a light field movie of some duration, the data sizes can be very large. Our Icarus sequence is 10,000 images (3 GB uncompressed) for 3 seconds of motion. The shark sequence was 256 images in orientation by 230 time steps or 17.25 GB of uncompressed data for 7.5 seconds of motion. A naïve light field movie channel might require up to hundreds of video channels of a conventional cable system.

These requirements can be alleviated somewhat by the same tricks used with video clips on CDROMS. Short sequences can be interspersed with less expensive media, movie loops can show a cyclical or brief event, and compression may be used to reduce the cost of data transmission and storage provided that decompression may be done in real time. Just as vector quantization was used to enable software-based video codecs, VQ has been applied successfully to light fields [4]. However, in the scheme presented, a fixed compression ratio of 24 to 1 was all that was possible on the fly. To gain lossless compression of the VQ indices required an off-line decompression step of the entire set of indices for a given time step. A preferable scheme would allow decompression of just the set of indices to be viewed.

3.4 Vector Quantization Compression

As a starting point for our compression schemes, we employed the vector quantization scheme of Hanrahan and Levoy, although our scheme used 16 pixel blocks to represent the two spatial dimensions, one orientation, and time. Table 1 shows the compression ratios obtained using VQ coding with run-length elimination of constant backgrounds. Frames per second (FPS) is shown for one and two processor machines. As expected, the shark and ornament compress well. Since these scenes are grayscale, we would expect a ratio of about eight-to-one without run length encoding,

Table 1. Compression ratios for run length VQ

Sequence	Ratio	FPS 1 Proc.	FPS 2 Proc.
Ornament	43.4	46.7	60.2
Shark	43.05	23.8	35.0
Snake	7.83	23.4	35.8
Icarus	7.88	22.0	32.6

3.5 Codebook Minimization

We generated VQ codebooks with 16384 sample blocks. To gain further compression, and to speed encoding, we combined codebook entries together by iteratively merging the two closest codebook entries. This was continued until the mean-squared difference of the closest pair was above a given threshold. (For the examples in this paper, this threshold was set to 2.0, where intensity values range from 0 to 255. The mean squared error of our compressed image sequences was approximately 1.5 levels out of 256.) The entries in the codebook were, in turn, used to compress the images leading to an array of block indices. Further compression was then possible using various lossless schemes for these indices.

In the special case of light fields with 1 DOF, we can use coherence in a column of pixels (or indices) to use various forms of lossless encoding on the fly. Before each column of pixels is resampled, we decode its corresponding column of indices.

3.6 Optimum Word-Size Encoding

Fixed table Huffman coding may be used to compress the indices. Table 2 shows the playback and compression rates for the different movie sequences. Huffman decoding is slow, but it acts as a good method to compare with the performance of other, less computationally expensive algorithms. Given the large number of similar images to be compressed, we were able to store large conditional probability tables to aid with lossless compression. In one scheme we used the conditional frequency distribution for a given index, we determined the optimal number of bits to represent the most frequent 2^N entries of that distribution.

In the bit stream, each index was encoded with a leading bit, which determined whether the subsequent bits represented a short code, or a full sized code. The meaning of the short code was determined by the previously decoded index. This

scheme outperformed Huffman coding in both compression ratio and speed. However, it more than halved the performance of the renderer.

Table 2. Huffman Coding performance

Sequence	Rel. Ratio	FPS 1 Proc.	FPS 2 Proc.
Ornament	1.22	12.9	21.2
Shark	1.66	12.6	17.8
Snake	2.36	4.4	8.1
Icarus	1.71	3.5	6.2

Table 3. Conditional Word-Size performance

Sequence	Rel. Ratio	FPS 1 Proc.	FPS 2 Proc.
Ornament	1.43	25.7	33.0
Shark	2.10	18.8	26.5
Snake	3.74	10.7	12.1
Icarus	2.56	9.1	10.9

Table 4. Index Pair-Quad Coding performance

Sequence	Rel. Ratio	FPS 1 Proc.	FPS 2 Proc.
Ornament	1.08	35.7	50.8
Shark	1.81	22.7	33.8
Snake	2.80	20.7	31.5
Icarus	1.89	18.9	29.8

3.7 Word Aligned Lossless Encoding

Given the basic cost of unpacking bit streams, we decided to explore using encoding schemes that were word aligned. Since our codebook sizes were restricted to a maximum of 14 bits, we could employ the additional two bits in a two byte word for additional compression information. A conditional frequency table was used to come up with a list of the N most frequent index pairs. N was limited to a maximum of 2^{14}. The table was not grown beyond the point at which the frequency of a given pair was less than two. In practice all 14 bits worth of table were always used. A second table was generated for successive pairs of indices into the first. This index-pair conditional frequency table was used to generate a quad index table that stored the M most common pairs of index pairs. M was limited to a maximum of 2^{15}, or less if the next most common pair of index pairs had a frequency of less than 4.

The list of indices for an image was encoded as follows. If the four indices matched an entry in the quad index table, then the most significant bit was set and the address in that table was stored in the other 15 bits. If not, then each pair of indices would be tested to see if they were in the index pair table. If so, the second most significant bit was set, and the address in that table was entered into the remaining 14 bits of the code. Finally, if neither was true, the original code index was stored in the code word. This allowed for very efficient decoding as is evident from Table 4. The compression

ratios obtained were somewhat lower than the conditional word size scheme. However, rendering performance nearly matched the light field renderer with only run length encoding, and is the preferred method for our system. This compression algorithm can be seen as a form of variable block size VQ coding.

3.8 Temporal Coherence

To exploit temporal coherence, our implementation kept the uncompressed indices of light field 0. The run length encoding scheme kept the indices for subsequent light fields where they were not identical to those for light field 0. Two blocks were considered "identical" if their mean squared difference was less than a threshold (in this case 2.0). As the indices were decompressed, the zero runs were filled in with the corresponding indices from light field 0. Rendering performance was unaffected relative to the non-frame-differenced scheme. Table 5 shows the results of using index differencing. The relative ratios are with respect to the size of the VQ compressed image.

Table 5. Compression ratio with index differencing

Scheme	Snake	Icarus
None	5.77	1.93
Huffman	9.02	2.81
Cond. Word	11.49	3.9
Pair-Quad	8.66	2.75

4 Tracking

As described in Section 1.2, one approach to the display of light field movies requires tracking of the user's eyes. This section summarizes the requirements of tracking and some techniques for meeting these requirements.

4.1 Requirements and Existing Techniques

The tracking system must be able to determine the eyes' positions in three dimensions, including depth. The ideal system would track the eyes directly, but in practice, the system may track the head and infer the eye positions. The tracker must provide real-time performance, the most important aspect of which is low latency. Ideally, latency should be below 10 to 15 ms [10]. The tracker must provide high accuracy, with freedom from jitter and drift being most important. Finally, the tracker must be convenient for the user. Convenience is reduced if the user must wear special equipment, or if the user must manually initialize the tracker.

Tracking can be accomplished by a variety of means: mechanical arms like the Immersion MicroScribe, electromagnetic systems like the Polhemus FASTRAK, active vision-based systems like the LED-based Northern Digital OptoTrak and the

infrared technique of Morimoto et al. [12], and passive vision-based techniques like color-histogram algorithms of Bradski [13] and Birchfield [14].

4.2 A Multi-Modal Vision-Based Algorithm

For our tracking system, we wanted the convenience of a passive vision-based technique, with the further convenience of no user initialization. Our algorithm was inspired by the success of previous multi-modal algorithms [15].

The first technique in our algorithm uses traditional ideas for matching templates in an intensity image by minimizing the sum of squared differences (SSD). The other technique uses a depth image computed by real-time stereo hardware [16], searching this image for a *face-blob*, a region with a depth profile similar to a human face. The algorithm uses the face-blob's position to initialize the intensity template and to correct for drift in the template matching process.

The face-blob finder searches the depth image for the shallowest pixel (closest to the stereo cameras). In regions around this pixel, it performs a sequence of tests to count the pixels within a specified depth interval from the shallowest pixel. If a face-blob was found at the current frame but not the previous one, then a face has entered the scene and the tracking algorithm grabs a new intensity template at the face-blob's position. The algorithm also uses a vector from the template match position to the face-blob's position to slightly adjust the overall tracked position, thus reducing drift.

Plate 5a shows a user in front of the two cameras used by our stereo hardware. We have added a modest texture to the user's red-blue glasses, as texture helps both algorithm techniques. Some results of our algorithm appear in Plate 5b and Video 7 [17].

The algorithm reliably finds users' heads and begins tracking without user initialization. It comes close to meeting our requirements for real-time performance. Given the depth image, it takes 15-16 ms to update the tracked position using a 300 MHz Pentium II. Our current version of the stereo hardware introduces two frames (66 ms) of latency to produce the depth image, but a future version should eliminate much of this latency. The algorithm's accuracy also comes close to meeting our requirements, but there is still some jitter.

5 Conclusions and Future Work

We presented a novel software-based light field renderer that can run at from 30 to 60 frames per second with two to four times higher compression ratios than previous techniques, both for precomputed animations and captured images.

The system originally used a mechanically-based tracker. By combining real-time stereo with template-based correlation, we developed a passive vision-based tracker, although work remains to be done in terms of robustness, jitter and latency.

Our work to date has shown that light field movies can provide a vivid illusion of viewing a moving 3-D scene. However, a number of problems remain. The most pressing of which is the large data size of the files even after compression. It seems likely that static light fields could be downloaded over the web given next generation modems. However, downloading entire light field movies would be tedious.

Similarly, broadcasting light field movies would require large amounts of bandwidth, probably on the order of tens of video channels given better compression techniques.

A likely near-term application of light fields would be e-commerce, allowing users to preview photographically captured static objects. Short light field movie clips may also be added to DVDROM titles, such as encyclopaedias, where they would be well suited to showing cyclical mechanisms, plants growing etc. They could also be used for any 3-D scientific visualization too slow to render in real time, such as time-dependent volumetric scanned medical data or weather simulations.

References

1. Gersho, A., Gray, R.M.: Vector Quantization and Signal Compression. Kluwer Academic Publishers, Dordrecht (1992)
2. Deering, M.F: High Resolution Virtual Reality. Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New York (1992) 195—202
3. Cutler, L.D., Frolich, B., Hanrahan, P.: Two-Handed Direct Manipulation on the Responsive Workbench. 1997 Symposium on Interactive 3D Graphics, ACM (1997) 107—114
4. Levoy, M., Hanrahan, P.: Light Field Rendering. Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New York (1996) 31—42
5. Gortler, S.J., Grzeszczuk, R., Szeliski, R., Cohen, M.F.: The Lumigraph. Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New York (1996) 43—54
6. Chen, S.E.: QuickTime VR–An Image-Based Approach to Virtual Environment Navigation. Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New York (1995) 29—38.
7. Lucente, M., St-Hilaire, P., Benton, S.A., Arias, D.L., Watlington, J.A.: New Approaches to Holographic Video. SPIE Proceedings #1732, Holographics International (1992) 377—386
8. Kajiki, Y., Yoshikawa, H., Honda, T.: Three-dimensional Display with Focused Light Array. SPIE Proceedings #2652, Practical Holography X (1996), 106—116
9. Shiwa, S., Tersutani, N., Akiyama, K., Ichinose, S., Komatsu, T.: Development of Direct-View 3D Display for Video-phones Using 15 inch LCD and Lenticular Sheet, IEICE Trans. Inf. and Syst. Vol E77-D, No. 9, Sept. 1994.
10. Regan, M., Miller, G.S.P., Rubin, S.M., Kogelnik, C.: A Real-Time Low-Latency Hardware Light-Field Renderer. To appear Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New York (1999)
11. Halle, M.: Holographic Stereograms as Discrete Imaging Systems. SPIE Proceedings #2176, Practical Holography VIII (1994) 73—84
12. Morimoto, C., Koons, D., Amir, A., Flickner, M.: Pupil Detection and Tracking Using Multiple Light Sources. IBM Research Report, 1998, available from the WWW site: domino.watson.ibm.com/library/cyberdig.nsf/Home.
13. Bradski, G.R.: Computer Vision Face Tracking for Use in a Perceptual User Interface. Intel Technology Journal, Q2 1998, http://developer.intel.com/technology/itj/q21998.
14. Birchfield, S., Elliptical Head Tracking Using Intensity Gradients and Color Histograms. Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition (1998) 232—237
15. Darrell, T., Gordon, G., Harville, M., Woodfill, J.: Integrated Person Tracking Using Stereo, Color, and Pattern Detection. Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition (1998) 601—609
16. Woodfill, J., Von Herzen, B.: Real-Time Stereo Vision on the PARTS Reconfigurable Computer. Proceedings IEEE Symposium on Field-Programmable Custom Computing Machines (1997) 242—250
17. Video Clips 1-7 http://web.interval.com/papers/1999-076

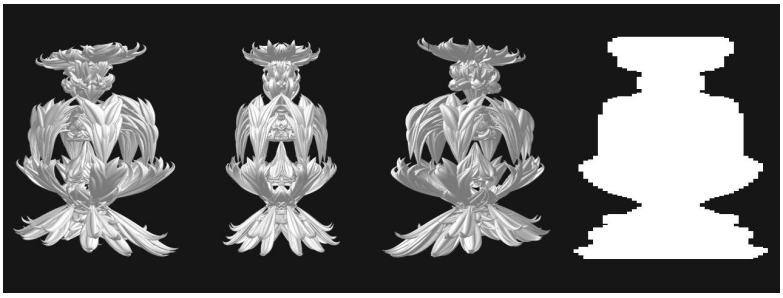


Plate 1. Three views of an animated ornament with its foreground pixel map

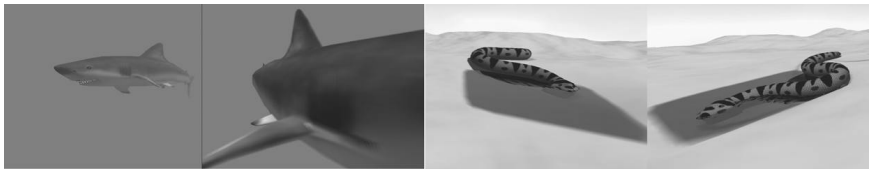


Plate 2. Two images from a) shark light field movie and b) snake light field movie

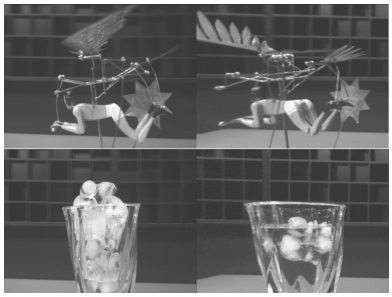


Plate 3. Images from a) the Icarus automaton light field and b) the melting ice time lapse light field

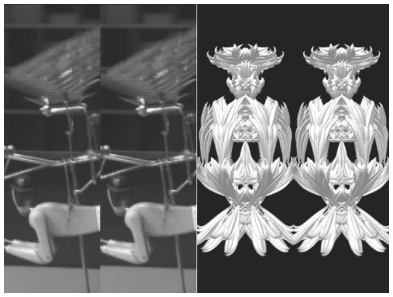


Plate 4. Two comparisons of 1-bit interpolation (on left) versus linear blending (on the right)

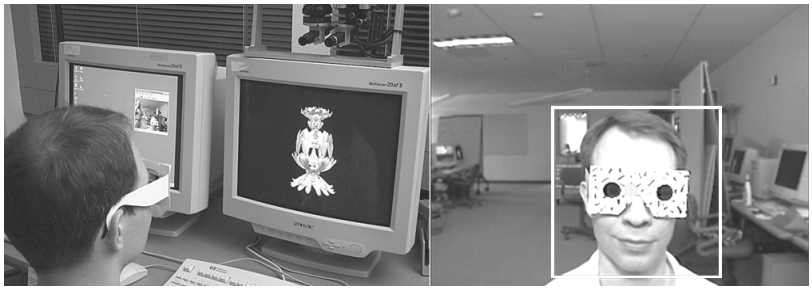


Plate 5. The vision based tracker a) seen in use and b) with the tracking result as seen by the cameras.