Multi-view Geometry Estimation for Light Field Compression

Ismail Oner Sebe, Prashant Ramanathan and Bernd Girod

Stanford University, Department of Electrical Engineering
Information System Laboratory
Stanford, CA 94305 USA
Email: {iosebe, pramanat, bgirod}@stanford.edu

Abstract

Geometry-aided light field compression requires accurate geometry for the efficient representation of the light field image data. In this work, we propose a method to directly estimate a geometry model from the light field images, such that it maximizes the compression efficiency. Previous work on this problem uses least-squares to solve a set of optical-flow based equations. This approach suffers from stability problems and gives sub-optimal results, due to outliers included in the set of equations. We propose an extension to this approach that addresses some of its short-comings. Specifically, we use weighted least-squares to identify and suppress the effects of outliers and a multi-resolution estimation algorithm for the geometry model. The experiments performed on real and synthetic data show that our technique stabilizes the algorithm, decreases the total running time by a factor of 10 and decreases bit-rate by up to 10% over the previous work. For the synthetic light field sequences, we show that this achieves the optimal achievable compression efficiency, under our constrained arrangement.

1 Introduction

Image-based rendering is an alternative to traditional image synthesis techniques that enables us to generate novel views from a finite set of images. Light field rendering is one such image-based rendering method [1, 2].

A light field is a 4-D data set which can be parameterized as a 2-D array of 2-D light field images. Unfortunately, light fields are extremely large data sets, due to the number of high-resolution images needed to render a photorealistic image. Typical light field data sets vary from a few megabytes to tens of gigabytes [3]. The efficient compression of light field data is essential. Direct vector quantization, followed by entropy coding, is suggested [1], but only low compression ratios are attained, around 120:1.

Disparity compensation, which is the analogue of motion compensation in video processing, is an efficient technique for light field compression. Light fields are composed of images of the same object from different views, which gives rise to a strong correlation between these images. Disparity compensation exploits this redundancy by predicting the current image from neighboring reference images. Disparity or depth values are either specified for a block of pixels, or inferred from a geometry [4, 5, 6].

In this paper, we consider disparity-compensation using an explicit geometry model of the object. Unfortunately, limited-accuracy computer vision techniques and representation of the data by a finite number of bits lead to deviations from the actual geometry. Even small inaccuracies in the geometry model may result in serious loss of compression efficiency [7].

In our previous work, we proposed to automatically refine the geometry using the acquired images [8]. This geometry refinement method is based on an optical-flow formulation to minimize the disparity-compensated prediction error. We observed a 10% bit-rate reduction which represented only half the possible bit-rate reduction, given the degrees of freedom of the geometry model. Moreover, the proposed algorithm had stability problems and long computation time.

In this paper, we refine the geometry estimation method described in [8] in order to improve compression efficiency and overcome some of its limitations. We base our technique on weighted least squares and multi-resolution processing of the light field images. Our new method has several advantages such as achieving optimum compression ratios, stabilizing the overall algorithm, and reducing the execution time by a factor of 10.

2 Geometry-Based Disparity Compensation

State-of-the-art light field compression algorithms [4, 5, 6, 7, 8] use disparity compensation. The underlying idea of disparity-compensated prediction is that a pixel in a light field image can be predicted from corresponding pixels in the reference images. The light field images are taken from a set of calibrated cameras, which enables us to establish the correspondence between the images if the depth value of the pixel is known. We assume the object surfaces are *Lambertian*, i.e. looks similar from two different views if the pixel is not occluded.

Prediction from other images is done using the hierarchical coding structure described by Magnor and Girod [4]. Depending on the viewing angle, each image is predicted from two to four neighboring reference images. The prediction order of the images is pre-defined such that the reference images are coded before the predicted images. Multiple reference images are used to minimize the occlusions. In addition, rendering quality depends on the number of scene images available.

3 Proposed Algorithm for Geometry Refinement

The technique we propose estimates a geometry model for best compression performance. A silhouette-based model is used as an initial estimate. An optical-flow based shape refinement formulation is used to derive linear equations for the geometry model. These equations are solved using weighted least squares. The geometry model is refined using a multi-resolution hierarchical pyramid approach.

3.1 Silhouette-based Geometry Model

For each data set, we initialize our program with the silhouette-based geometry. We assume that silhouette information is available for the light field images. The silhouette-based geometry is obtained by shrinking a 642-vertex subdivided icosahedron model, that is larger than the actual object, to fit the silhouette in each view. This shrinking is achieved by moving a vertex radially towards the center of the model if it lies outside the silhouette in a particular view. Applying this procedure for each image, we thereby obtain a 642-vertex silhouette-based geometry of the object.

3.2 Optical-flow-based equations

Each light field image may be predicted from multiple reference images that have been previously encoded, according to the hierarchical structure discussed in the previous section. Throughout the paper, we refer to the image to be predicted and a pixel in that image as the *target* image and target pixel, respectively, and the images and pixels used for prediction as *reference* images and *reference* pixels. Any given target pixel corresponds to a 3-D point along the viewing ray $\mathbf{l}$ in 3-D space. We parameterize this line $\mathbf{l}$ in world coordinates by the depth value t,

$$\mathbf{l}(t) = \begin{bmatrix} l_x(t) \\ l_y(t) \\ l_z(t) \end{bmatrix} \tag{1}$$

This line is a function of the intrinsic and extrinsic camera parameters of the target view, as well as the target pixel position. Projection of this line onto the reference image gives us the *epipolar line*,

$$\mathbf{e}(t) = \begin{bmatrix} e_X(t) \\ e_Y(t) \end{bmatrix}. \tag{2}$$

The epipolar line is also parameterized by the depth value t, and is a function of pixel position in the target image, camera parameters of the target image and the reference images.

Specifying the depth value t fixes both the 3-D point in 3-D space and a specific point in the reference image. Thus, we obtain a corresponding reference pixel for the target pixel. For a Lambertian surface and the true depth value t, we observe the same image intensity at the target and reference pixels. An error in depth value will result in a prediction error, which will lead to an intensity difference denoted by ΔI. The intensity difference is a function of intensity value at the actual depth value and the current depth value,

$$\Delta I = I_t - I_r(e(t_0)) = I_r(e(t)) - I_r(e(t_0)), \tag{3}$$

where t_0 is the current (inaccurate) depth value, and t is the correct depth value that results in zero prediction error. If we assume the intensity gradient

$$\mathbf{g} = \begin{bmatrix} g_x \\ g_y \end{bmatrix} \tag{4}$$

to be locally constant over this region in the reference image, we can write the intensity difference as an optical flow equation:

$$\Delta I = \mathbf{g}^T(\mathbf{e}(t) - \mathbf{e}(t_0)). \tag{5}$$

We need to express the depth value in terms of the parameters of the geometry model,

$$t = h(\mathbf{p}), \tag{6}$$

where $\mathbf{p}$ is the vector of geometry parameters and h is a nonlinear multivariate function that maps the geometry parameters to the depth of a given pixel.

We use a triangle mesh geometry model in order to represent our objects. With this model, our geometry parameters become the vertices of this triangle mesh. Furthermore, we restrict the movement of the vertices to one degree of freedom, radially from the center of the model. Although this parameterization makes it possible for us to solve the estimation problem, it limits our compression efficiency. We will discuss this further in Section 4.

We now describe our mapping function h. We assume that the viewing ray $\mathbf{l}$ intersects with only one of the triangle faces in the triangle mesh model. Therefore, we can determine the depth value t using only the three vertices that define that triangle face, instead of the whole geometry. By characterizing this triangle face as an infinite plane, we obtain a differentiable function h that describes the depth parameter t in terms of geometry parameters $\mathbf{p}$. We assume that other vertices will not affect this pixel through occlusions, and that the pixel will not move off this triangle. Both of these assumptions are valid when we restrict the changes on the geometry parameters to be small. We enforce this restriction by adding regularization terms into our set of equations.

Substituting (6) into (5), we obtain

$$\Delta I = \mathbf{g}^T(\mathbf{e}(h(\mathbf{p})) - \mathbf{e}(h(\mathbf{p}_0))) \tag{7}$$

where $\mathbf{p}_0$ denotes the current geometry configuration. In the above equation, both $\mathbf{e}$ and h are nonlinear functions of geometry parameters $\mathbf{p}$. In order to solve these equations efficiently, we need to linearize them. We linearize the epipolar line equation around $\mathbf{p}_0$, giving us

$$\mathbf{e}(h(\mathbf{p})) \approx C_1\mathbf{p} + C_2. \tag{8}$$

We can then write

$$\mathbf{e}(h(\mathbf{p})) - \mathbf{e}(h(\mathbf{p}_0)) \approx C_1\Delta\mathbf{p} \tag{9}$$

where $\Delta\mathbf{p} = \mathbf{p} - \mathbf{p}_0$ and C_1 is a matrix of size $2 \times N$, with N as the number of geometry parameters.

Combining (9) and (7), we obtain the following equation

$$\mathbf{g}^T(C_1\Delta\mathbf{p}) \approx \Delta I \tag{10}$$

which we use to solve for the geometry parameters.

3.3 Weighted Least Squares Solution

We extract one equation per target-reference pixel pair from the light field images using the geometry model, and arrange them to form the following matrix equations

$$A\Delta\mathbf{p} \approx \mathbf{b} \tag{11}$$

where

$$A = \begin{bmatrix} (\mathbf{g}^T C_1)^{(1)} \\ (\mathbf{g}^T C_1)^{(2)} \\ \ldots \\ (\mathbf{g}^T C_1)^{(M)} \end{bmatrix} \tag{12}$$

and

$$\mathbf{b} = \begin{bmatrix} (\Delta I)^{(1)} \\ (\Delta I)^{(2)} \\ \ldots \\ (\Delta I)^{(M)} \end{bmatrix}. \tag{13}$$

As we mentioned in the previous section, linearization is only valid for small $\boldsymbol{\Delta}p$ values, thus we add additional equations in order to regularize the solution. This gives us the equation

$$\begin{bmatrix} A \\ \lambda I \end{bmatrix} \Delta\mathbf{p} \approx \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix} \tag{14}$$

where λ is the regularization constant and I is the identity matrix. A larger λ value results in a smaller $\boldsymbol{\Delta}p$, and makes the problem more stable. On the other hand, a very large λ value slows the convergence of the solution. The λ value is chosen empirically and takes into account the number of optical-flow equations used.

The least squares method is used in our previous work [8] to solve (14). When applied iteratively in an analysis-synthesis loop, the algorithm sometimes does not converge. Moreover, the solution found is sub-optimal. Our method in [8] uses all target-reference pixel pairs in its matrix equation, many of which may be outlier data. In this work, we investigate methods for removing these outliers and improve the results.

Least squares is capable of finding the optimum solution in the presence of a non-biased noise, but is sensitive to outlier data. Robust estimators are employed in order to remove the outliers from the data without any assumptions of a distribution for the outliers or the data [9, 10]. Robust estimators try to minimize

$$\sum_i \rho(r_i), \tag{15}$$

where $\rho(r_i)$ is a symmetric, positive definite function of the errors r_i with a unique minimum at zero. We choose $\rho(r_i)$ to be less than r_i^2, used by least-squares, so that it is less sensitive to outliers. We solve the estimation problem first with least squares in order to get an initial estimate at every iteration. Weighted least squares is an example of a robust estimator, where each equation is weighted according to a confidence measure. We use the initial estimate to guide our choice of weights. In our technique, we solve the following iterated weighted least-squares problem

$$min \sum_i w(r_i) r_i^2, \tag{16}$$

where the r_i is the residual error from our least squares solution. We have tried different weighting strategies, including L1, L2, Fair, Huber, Cauchy [10], and a windowing function. Windowing, which classifies the data as either outliers or inliers, gives the best results. The windowing weight function has a threshold constant that needs to be estimated. The windowing function we have used is

$$w(r_i) = \begin{cases} 1, & \text{if } |r_i| < k\sigma \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

where σ is the standard deviation of the residuals and k is a constant. The value of k is selected empirically. The specific values used for k are described in Section 4.2. We use k multiple times in different contexts. We subscript k to denote which variable it is used for (e.g. k_r for residuals and k_d for displacements).

The windowing function needs an estimate of the standard deviation to eliminate the outliers from the data. Unfortunately, the value of σ is not known and needs to be estimated from the data. In the case of no outliers, σ can be found directly from its definition, but in the presence of outliers a robust method to estimate σ is required. A robust estimate of σ can be found as follows [10]:

$$\hat{\sigma} = 1.4826(1 + \frac{5}{n-p})\mathbf{median}_i |r_i| \tag{18}$$

where n is the number of equations and the p is the number of parameters. We use this estimate as our σ value in (17).

The matrix A and the vector b in (14) are weighted by (17) and a least squares solution of this new matrix is found. For stability of the algorithm, the regularization rows are not weighted. Although weighted least squares is an iterative method, we apply the weighted least squares method only once per iteration, since multiple iterations of weighted least squares from the same initial point led to spurious changes in the geometry parameters during our experiments.

3.4 Displacement Vector Estimation and Boundary Removal

Our assumption of a locally constant gradient over a region is valid only for small displacements. The displacement vector magnitude can be estimated as it is suggested in [11]

$$||\hat{d}|| = \sqrt{\frac{(I_r(t) - I_r(t_0))^2}{(\frac{\delta \bar{I}}{\delta X})^2 + (\frac{\delta \bar{I}}{\delta Y})^2}}. \tag{19}$$

Equations whose displacement vector exceeds a certain threshold should be treated as outliers. The threshold value for the displacement vector is found again adaptively by using (17). Here, we replace the r_i with $||\hat{d}||_i$ to find the standard deviation σ_d of the displacement vector. These weights are multiplied with the weights from the previous section.

Estimation of the gradient values is error-prone on the boundary of the objects. These points generate spurious data, which may not be detected by the above method. Thus, these points are also marked as outliers. Using all three methods for our data set results in a rejection of $20-30\%$ of the equations. In Section 4, we demonstrate the improvement we gain by removing these outliers.

3.5 Multi-Resolution Processing of the Light Field Images

We have mentioned that the optical-flow equations are valid only for a small region around where these parameters are calculated. Using a lower resolution version of the light field images enables us to consider a wider range of pixel positions, resulting in a more stable algorithm than the one for the higher resolution version. In addition, due to subsampling, the number of equations is smaller, which leads to much faster computation of the solution. But, the solution of these equations are only a rough estimate for the geometry model, since all the high frequency information is lost in the filtering process. Therefore, we use a multi-resolution approach for geometry estimation.

The refined geometry from lower resolution images provides the initial estimate for the higher resolution. Multi-resolution processing is more stable than applying geometry refinement directly to the silhouette in full-resolution. The number of iterations needed for convergence of the higher resolution estimation varies from data set to data set when multi-resolution estimation is used. Furthermore, although multi-resolution is more stable, it does not further improve our compression efficiency.

4 Results

We test our proposed algorithm on both real and synthetic light field data sets. An initial approximation of the geometry model is created using silhouette information both for full-resolution and multi-resolution schemes. We estimate the geometry using the technique proposed in Section 3. In our algorithm, we are constrained to only geometry refinement in the radial direction. We, therefore, can construct an optimal-constrained geometry, which represents the best achievable geometry with our radial constraints. For the synthetic test sequences, we have both the true geometry and the optimal-constrained geometry. These geometries give additional performance comparisons for the synthetic sequences. We compare the compression efficiency of various geometries by measuring its rate-PSNR curves. All the light fields are coded with a geometry-aided light field coder, which is described next.

4.1 Light Field Coder

The light field coder in our work uses block-based disparity compensation both with and without an explicit geometry model [4, 5]. Images are divided into 8×8 blocks, which may be coded with several modes. These modes are the INTRA mode, where DCT-based image compression is used for the block; the GEO mode, where an explicit geometry model is used to predict the block from reference images; the STD (standard) mode, where a depth value is specified to predict the block from the reference images; and the COPY mode, where a block from the same image location is simply copied from the reference images. In the GEO and STD modes, a DCT-based residual encoder is used. Mode-selection is performed by minimizing a rate-distortion Lagrangian cost function

$$J = D + \lambda_M R \tag{20}$$

where D is the sum-squared-error distortion of the block image, and R is the rate in bits for the block. The rate-distortion curve is obtained by varying quantizer parameter Q. The Lagrangian multiplier λ_M is used to trade-off rate versus distortion and is found by the following equation, which is commonly used in video compression, [12]

$$\lambda_M = 0.85Q^2. \tag{21}$$

4.2 Experiments

We have used three different light field data sets in our experiments. Two of these data sets, *Star* and *Cube*, are synthetic light fields composed of 26 images of resolution 256×256. The third data set, *Garfield*, is a real-world light field composed of 29 images of resolution 384×288. The *Garfield* data set has only frontal views of the object, which results in some instability in the geometry at the boundaries of our visible region.

We refine the silhouette-based initial geometry for each object using our proposed technique. We have used a regularization constant $\lambda = 10^{10}$ a weighted least squares threshold constant k_r=3 for residuals both in lower and full-resolution, and a displacement estimation threshold k_d=3 in full-resolution and k_d=5 for the lower resolution. We use 3×3 subsampling for lower-resolution images. Resolutions lower than this level do not seem to improve the geometry at all because the images already small and the objects occupy only quarter of the area of the images. All experiments include boundary removal. Typically, we need around 150 to 200 iterations to obtain our final geometry.

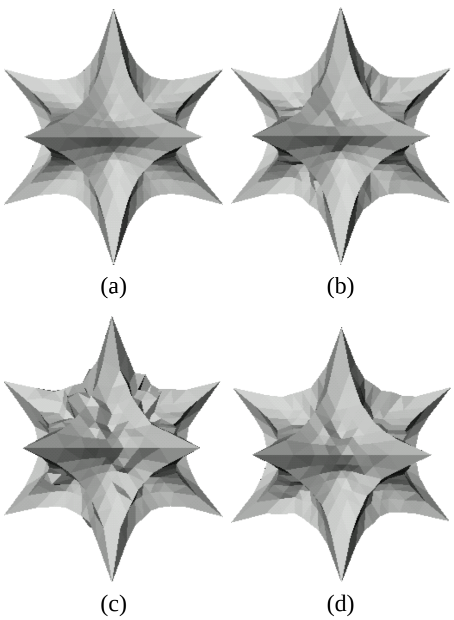

Figure 1: Geometry models for the *Star* light field: (a) the optimal-constrained geometry, (b) the estimated geometry at full-resolution, (c) the geometry from our previous work, (d) the estimated geometry at low resolution.

Figure 1 shows results of our algorithm for the *Star* light field. Figure 1(a) shows the optimum geometry we can achieve with the degrees of freedom of the triangle mesh model. Figure 1(b) shows the final estimated geometry using full-resolution images. Figure 1(c) shows the output of our previous work [8], and Figure 1(d) shows the refined geometry after lower resolution but before full-resolution. Figure 2 illustrates the results for the *Garfield* light field. The output of the previous work [8] is given as reference in Figure 2(a); Figure 2(b) and 2(c) show the outputs of the full and lower resolutions, respectively. We note here that our proposed algorithm is more stable and does not have the obvious geometry inaccuracies of the previous work.

Figure 3 and 4 show the rate-PSNR curves using various geometry models for our synthetic light fields *Star* and *Cube*, respectively. In the synthetic sequences, the gap between the optimal-constrained geometry and the reconstructed geometry vanishes. This shows that our scheme almost perfectly reconstructs the best possible geometry using only image data, given our radial constraints. Even the geometry model reconstructed from lower resolution images gives close compression ratios to the previous work. The proposed technique realizes approximately 95% of the gain possible under the constrained arrangement.

Figure 5 shows the rate-PSNR curve for the real-world light field data set, *Garfield*. This data set

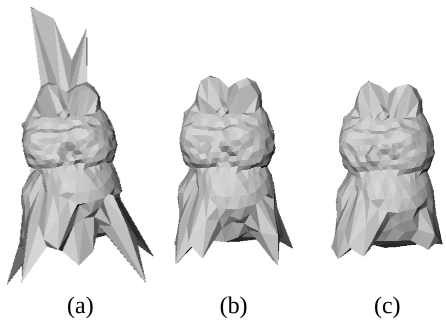

Figure 2: Geometry models for the *Garfield* light field: (a) the geometry from our previous work, (b) the estimated geometry at full-resolution, (c) the estimated geometry at low resolution.

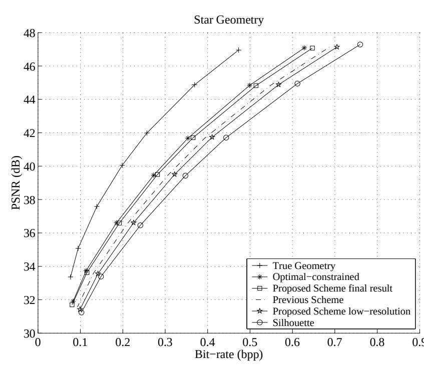

Figure 3: Rate-distortion curves for *Star*. We see a 20% bit-rate reduction over the silhouette-based geometry. Also, we almost reach the optimal-constrained geometry compression efficiency.

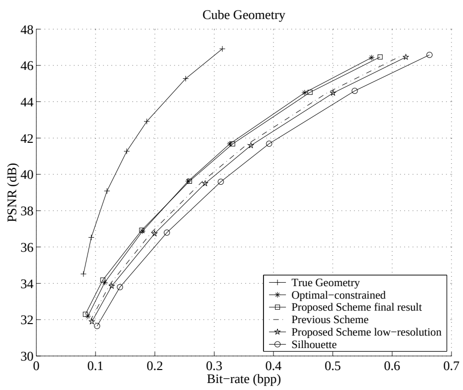

Figure 4: Rate-distortion curves for *Cube*. We see a 18% bit-rate reduction over the silhouette reconstructed geometry. Also, we reach the optimal-constrained geometry compression efficiency.

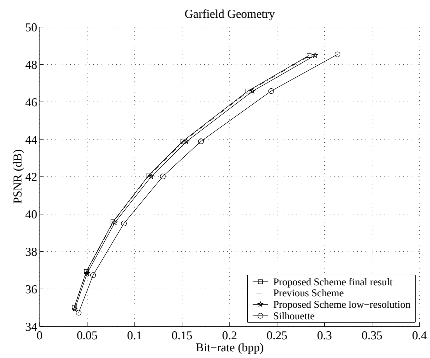

Figure 5: Rate-distortion curves for *Garfield*. We see a bit-rate reduction of 12% over the silhouette-based geometry.

has different characteristics than the synthetic light fields. Both the improvement from silhouette to lower resolution and from lower resolution to full-resolution is small compared to the synthetic light fields. We suspect that this performance degradation is due to two reasons. First, the camera parameters may not be accurate, which results in effects that can not be accounted for given our model. Second, real-world images have camera noise, whereas we assume noise-free images. The effects of the parameter inaccuracies and camera noise become less important at lower resolutions due to filtering. Despite the limitations, our proposed technique gives slightly better results compared to our previous work, and it is more stable as seen in Figure 2.

Table 1 shows the average rate and PSNR improvements of various geometry models for the light field data sets we have used in our experiments. In all the cases, our technique gives better results than any other constructed geometry. The best compression efficiency, given our constrained arrangement, is almost achieved. In our experiments, we had a $12-20\%$ bit-rate reduction compared to the silhouette-based geometry. This corresponds to $0.9-1.8$dB improvement.

PSNR in [dB]	Star	Cube	Garfield
Previous Work	1.08	0.86	0.82
Full resolution	1.80	1.54	0.87
Lower resolution	0.72	0.63	0.65
Optimum-constrained	2.09	1.67	NA
Bit-rate in %	**Star**	**Cube**	**Garfield**
Previous Work	12.5	9.60	11.14
Full resolution	19.9	16.4	11.66
Lower resolution	8.53	7.10	8.99
Optimum-constrained	22.5	17.3	NA

Table 1: Average bit-rate and distortion improvements. All numbers are relative to the silhouette of the corresponding objects.

5 Conclusion

We have presented an algorithm to automatically estimate a geometry model for disparity-compensated light field compression. Our proposed algorithm improves upon our previous work [8] by increasing the compression efficiency, stabilizing the overall algorithm, and reducing the computation time by up to a factor of 10. Our experiments show bit-rate savings of 16-20% for the synthetic light fields and 12% for a real-world light field, over the silhouette-reconstructed geometry model. Furthermore, the results from the synthetic data sets indicate that the proposed algorithm achieves optimal compression efficiency under the constrained arrangement. Future work includes relaxing these constraints and incorporating joint camera parameter estimation into geometry estimation for real-world sequences.

References

[1] Marc Levoy, and Pat Hanrahan, "Light field rendering", in *Computer Graphics, SIGGRAPH96*, pp. 31-44, August 1996

[2] S.J. Gortler, R. Grzeszczuk, R. Szeliski and M. Cohen, "The Lumigragh", in *Computer Graphics,SIGGRAPH96*, August 1996

[3] Marc Levoy, and Kari Pulli, et al., "The Digital Michelangelo project: 3D scanning of large statues", in *Computer Graphics, SIGGRAPH00*, pp.131-144, August 2000

[4] Marcus Magnor, and Bernd Girod, "Data compression for light field rendering", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 10, No 3, April 2000.

[5] Marcus Magnor, Peter Eisert, and Bernd Girod, "Model-aided coding of multiviewpoint images for immersive viewing", in *Proceedings of the IEEE International Conference on Image Processing ICIP-2000*, Vancouver, Canada, vol. 2, pp. 919-922, September 2000

[6] Xin Tong, and Robert M. Gray, "Coding of multi-view images for immersive viewing", in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing ICASSP 2000*, Istanbul, Turkey, vol. 4, pp. 1879-1882, June 2000

[7] Marcus Magnor, "Geometry-Adaptive multiview coding techniques for image-based rendering", Ph.D. thesis, University Erlangen-Nuremberg, Germany, 2001

[8] Prashant Ramanathan, Eckehard Steinbach, Peter Eisert, and Bernd Girod, "Geometry refinement for light field compression", *International Conference on Image Processing ICIP-2002*, Rochester, NY, September 2002

[9] P.H.S. Torr, and D.W. Murray, "The Development and comparison of robust methods for estimating the fundamental matrix", *International Journal Computer Vision*, pp. 11-33, July 1996

[10] Zhengyou Zhang, "Parameter estimation techniques: A tutorial with application to conic filtering", *Image and Vision Computing Journal*, Vol.15, No.1, pp. 59-76, 1997

[11] Peter Eisert, "Very low bit-rate video coding using 3-D computer models", Ph.D. Thesis, University Erlangen-Nuremberg, Germany, 2000

[12] Gary J. Sullivan, and Thomas Wiegand, "Rate-distortion optimization for video compression", *IEEE Signal Processing Magazine*, vol. 15, pp. 74-90, November 1998