MVA2007 IAPR Conference on Machine Vision Applications, May 16–18, 2007, Tokyo, JAPAN, pp. 178–181.

Robust and Efficient 3-D Reconstruction by Self-Calibration

Hanno Ackermann and Kenichi Kanatani

Department of Computer Science, Okayama University, Okayama 700-8530 Japan
{hanno,kanatani}@suri.it.okayama-u.ac.jp

Abstract

We present a very robust and efficient scheme for 3-D reconstruction from point correspondences over multiple images taken by uncalibrated cameras. Our method is based on iterative projective reconstruction followed by Euclidean upgrading. We accelerate the time-consuming iterations for projective reconstruction by using the power method for eigenvalue computation, accelerating the power method itself, and introducing SOR for further acceleration. Using simulated and real video images, we demonstrate that our techniques dramatically speed up the computation, in some cases about 8,000 times faster.

1. Introduction

Various techniques have been proposed in the past for self-calibration: 3-D reconstruction from point correspondences over multiple images taken by uncalibrated cameras [1]. Here, we adopt the two-stage method consisting of *projective reconstruction* for computing a 3-D shape up to projectivity and *Euclidean upgrading* that transforms it to a correct shape.

For projective reconstruction, we adopt the method of Mahamud and Hebert [6] and the method of Heyden et al. [4]. For the Euclidean upgrading, we modify the method of Seo and Heyden [8], which is based on the *dual absolute quadratic constraint* [10].

It turns out that the iterations of projective reconstruction far exceed the Euclidean upgrading in computation time. In order to reduce the time, we introduce the power method for eigenvalue computation. We also accelerate the power method itself. In addition, we introduce SOR for further acceleration. Using simulated and real video images, we demonstrate that our techniques dramatically speed up the computation, in some cases about 8,000 times faster.

2. Projective Reconstruction

2.1 Notations and Terminologies

We observe N points over M image frames, and let $(x_{\kappa\alpha}, y_{\kappa\alpha})$ be the position of the αth point in the κth frame. Let its 3-D position be $(X_\alpha, Y_\alpha, Z_\alpha)$. We represent these by[1]

$$\boldsymbol{x}_{\kappa\alpha} \simeq \begin{pmatrix} x_{\kappa\alpha} \\ y_{\kappa\alpha} \\ f_0 \end{pmatrix}, \qquad \boldsymbol{X}_\alpha \simeq \begin{pmatrix} X_\alpha \\ Y_\alpha \\ Z_\alpha \\ 1 \end{pmatrix}, \tag{1}$$

where $\simeq$ means that the definition is up to scale. The camera imaging can be modeled by

$$z_{\kappa\alpha}\boldsymbol{x}_{\kappa\alpha} = \boldsymbol{\Pi}_\kappa \boldsymbol{X}_\alpha, \tag{2}$$

where $\boldsymbol{\Pi}_\kappa$ is a 3×4 camera matrix (unknown), and $z_{\kappa\alpha}$ is a constant called the projective depth (unknown). Projective reconstruction is to compute matrices $\boldsymbol{\Pi}_\kappa$ and vectors $\boldsymbol{X}_\alpha$ that satisfy Eq. (2) [1].

[1]The constant f_0 is a scale factor for stabilizing numerical computation [2]. In our experiments, we used the value $f_0 = 600$ (pixels).

2.2 Primal Method

We reformulate the method of Mahamud and Hebert [6], which we call the *primal method*, as follows:

Input: $\boldsymbol{x}_{\kappa\alpha}$, $\kappa = 1, ..., M$, $\alpha = 1, ..., N$, admissible reprojection error $E_{\min}$ (pixels).

Output: $\boldsymbol{\Pi}_\kappa$, $\kappa = 1, ..., M$, $\boldsymbol{X}_\alpha$, $\alpha = 1, ..., N$.

Computation:

1. Initialize the projective depths to $z_{\kappa\alpha} = 1$.
2. For $\alpha = 1, ..., N$, compute the vector $\boldsymbol{p}_\alpha$ that vertically aligns $z_{1\alpha}\boldsymbol{x}_{1\alpha}, ..., z_{M\alpha}\boldsymbol{x}_{M\alpha}$ as its components and normalize it into unit norm.
3. Compute the unit eigenvectors $\boldsymbol{u}_1, ..., \boldsymbol{u}_4$ of
$$\boldsymbol{M} = \sum_{\alpha=1}^{N} \boldsymbol{p}_\alpha \boldsymbol{p}_\alpha^\top, \tag{3}$$
for the largest four eigenvalues.
4. Compute the matrix $\boldsymbol{\Pi}_\kappa$ by
$$\boldsymbol{\Pi}_\kappa = \begin{pmatrix} \boldsymbol{u}_{1\kappa} & \boldsymbol{u}_{2\kappa} & \boldsymbol{u}_{3\kappa} & \boldsymbol{u}_{4\kappa} \end{pmatrix}, \tag{4}$$
where $\boldsymbol{u}_{k\kappa}$ is the 3-D vector consisting of the $3(\kappa-1)+1$th, $3(\kappa-1)+2$th, and $3(\kappa-1)+3$th components of $\boldsymbol{u}_k$.
5. Do the following computations for $\alpha = 1, ..., N$.
   (a) Compute the unit eigenvector $\boldsymbol{\xi}_\alpha$ of the matrix $\boldsymbol{A}^\alpha = (A^\alpha_{\kappa\lambda})$ defined by
   $$A^\alpha_{\kappa\lambda} = \frac{\sum_{k=1}^{4} (\boldsymbol{x}_{\kappa\alpha}, \boldsymbol{u}_{k\kappa})(\boldsymbol{x}_{\lambda\alpha}, \boldsymbol{u}_{k\lambda})}{\|\boldsymbol{x}_{\kappa\alpha}\| \cdot \|\boldsymbol{x}_{\lambda\alpha}\|}, \tag{5}$$
   for the largest eigenvalue, and choose the sign so that $\sum_{\kappa=1}^{M} \xi_{\kappa\alpha} \geq 0$.
   (b) Determine the projective depths $z_{\kappa\alpha}$ by
   $$z_{\kappa\alpha} = \frac{\xi_{\kappa\alpha}}{\|\boldsymbol{x}_{\kappa\alpha}\|}. \tag{6}$$
   (c) Recompute the vector $\boldsymbol{p}_\alpha$.
   (d) Compute the 3-D positions $\boldsymbol{X}_\alpha = (X^k_\alpha)$ by
   $$X^k_\alpha = (\boldsymbol{p}_\alpha, \boldsymbol{u}_k), \qquad k = 1 \sim 4. \tag{7}$$
6. Compute the following reprojection error E:
$$E = f_0 \sqrt{\frac{1}{MN} \sum_{\kappa=1}^{M} \sum_{\alpha=1}^{N} \|\boldsymbol{x}_{\kappa\alpha} - Z[\boldsymbol{\Pi}_\kappa \boldsymbol{X}_\alpha]\|^2} \tag{8}$$
7. If $E < E_{\min}$, stop. Else, go back to Step 3.

The above iterations can be viewed as a special EM algorithm, so the convergence is guaranteed.

2.3 Dual Method

Heyden et al. [4] proposed a different type of iterative projective reconstruction, which we call the *dual method*. We rewrite their method as follows:

Input: $\boldsymbol{x}_{\kappa\alpha}$, $\kappa = 1, ..., M$, $\alpha = 1, ..., N$, admissible reprojection error $E_{\min}$ (pixels).

Output: $\boldsymbol{\Pi}_\kappa$, $\kappa = 1, ..., M$, $\boldsymbol{X}_\alpha$, $\alpha = 1, ..., N$.

Computation:

1. Initialize the projective depths to $z_{\kappa\alpha} = 1$.
2. Compute the vectors

$$\begin{aligned}\boldsymbol{q}_\kappa^1 &= (z_{\kappa1}x_{\kappa1}/f_0,\ z_{\kappa2}x_{\kappa2}/f_0,\ ...,\ z_{\kappa1}x_{\kappa1}/f_0)^\top,\\ \boldsymbol{q}_\kappa^2 &= (z_{\kappa1}y_{\kappa1}/f_0,\ z_{\kappa2}y_{\kappa2}/f_0,\ ...,\ z_{\kappa1}y_{\kappa1}/f_0)^\top,\\ \boldsymbol{q}_\kappa^3 &= (z_{\kappa1},\ z_{\kappa2},\ ...,\ z_{\kappa1})^\top,\end{aligned} \tag{9}$$

   and normalize them so that

$$\sum_{i=1}^{3}\|\boldsymbol{q}_\kappa^i\|^2 = \sum_{\alpha=1}^{N} z_{\kappa\alpha}^2\|\boldsymbol{x}_\kappa\|^2 = 1. \tag{10}$$

3. Compute the unit eigenvectors $\boldsymbol{v}_1$, ..., $\boldsymbol{v}_4$ of

$$\boldsymbol{N} = \sum_{\kappa=1}^{M}\sum_{i=1}^{3}\boldsymbol{q}_\kappa^i\boldsymbol{q}_\kappa^{i\top}, \tag{11}$$

   for the largest four eigenvalues.

4. Compute the 3-D positions $\boldsymbol{X}_\alpha = (X_\alpha^k)$ by

$$X_\alpha^k = (\text{the } \alpha\text{th component of } v_k),\ k = 1 \sim 4. \tag{12}$$

5. Do the following computations for $\kappa = 1, ..., M$.

   (a) Compute the unit eigenvector $\boldsymbol{\xi}_\kappa$ of the matrix $\boldsymbol{B}^\kappa = (B_{\alpha\beta}^\kappa)$ defined by

$$B_{\alpha\beta}^\kappa = \frac{(\boldsymbol{v}_\alpha, \boldsymbol{v}_\beta)(\boldsymbol{x}_{\kappa\alpha}, \boldsymbol{x}_{\kappa\beta})}{\|\boldsymbol{x}_{\kappa\alpha}\| \cdot \|\boldsymbol{x}_{\kappa\beta}\|}, \tag{13}$$

   where $\boldsymbol{v}_\alpha$ is the 4-D vector consisting of the αth components of the basis vectors $\boldsymbol{v}_1$, ..., $\boldsymbol{v}_4$. Choose the sign of $\xi_{\kappa\alpha}$ so that $\sum_{\alpha=1}^{N}\xi_{\kappa\alpha} \geq 0$.

   (b) Determine the projective depths $z_{\kappa\alpha}$ according to Eq. (6).

   (c) Recompute the vectors $\boldsymbol{q}_\kappa^i$, $i = 1, 2, 3$.

   (d) Compute the matrix $\boldsymbol{\Pi}_\kappa$ by

$$\Pi_{\kappa(ij)} = (\boldsymbol{q}_\kappa^i, \boldsymbol{v}_j). \tag{14}$$

6. Compute the reprojection error E in Eq. (8).
7. If $E < E_{\min}$, stop. Else, go back to Step 3.

3. Euclidean Upgrading

3.1 Dual Absolute Quadratic Constraint

The solution of Eq. (2) is not unique. In fact, if we transform one solution $\{\boldsymbol{\Pi}_\kappa, \boldsymbol{X}_\alpha\}$ by an arbitrary nonsingular 4×4 matrix $\boldsymbol{H}$ in the form

$$\tilde{\boldsymbol{\Pi}}_\kappa = \boldsymbol{\Pi}_\kappa\boldsymbol{H}^{-1}, \qquad \tilde{\boldsymbol{X}}_\alpha = \boldsymbol{H}\boldsymbol{X}_\alpha, \tag{15}$$

Eq. (2) is also satisfied. The second of Eqs. (15) implies that the 3-D position $\boldsymbol{X}_\alpha$ is determined only up to an arbitrary *homography* (or *projective transformation*) defined by the matrix $\boldsymbol{H}$.

The true camera matrix $\bar{\boldsymbol{\Pi}}_\kappa$ has the form

$$\bar{\boldsymbol{\Pi}}_\kappa = \boldsymbol{K}_\kappa\begin{pmatrix}\boldsymbol{R}_\kappa & \boldsymbol{t}_\kappa\end{pmatrix}, \tag{16}$$

where $\boldsymbol{R}_\kappa$ and $\boldsymbol{t}_\kappa$ are, respectively, the orientation and origin of the world coordinate system relative to the κth camera, and $\boldsymbol{K}_\kappa$ is the matrix of intrinsic parameters, for which we assume the form

$$\boldsymbol{K}_\kappa = \begin{pmatrix} f_\kappa & 0 & u_{\kappa0} \\ 0 & f_\kappa & v_{\kappa0} \\ 0 & 0 & 1 \end{pmatrix}, \tag{17}$$

meaning that the skew is 0 and the aspect ratio is 1 with the principal point at $(u_{\kappa0}, v_{\kappa0})$ and the focal length f_κ. If the true camera matrix $\bar{\boldsymbol{\Pi}}_\kappa$ is obtained by multiplying the current estimate $\boldsymbol{\Pi}_\kappa$ by a homography $\boldsymbol{H}$ in the form $\bar{\boldsymbol{\Pi}}_\kappa \equiv \boldsymbol{\Pi}_\kappa\boldsymbol{H}$, we have the *dual absolute quadratic constraint* [10]

$$\boldsymbol{\Pi}_\kappa\boldsymbol{\Omega}\boldsymbol{\Pi}_\kappa^\top \simeq \boldsymbol{K}_\kappa\boldsymbol{K}_\kappa^\top, \tag{18}$$

where we define

$$\boldsymbol{\Omega} \equiv \boldsymbol{H}\mathrm{diag}(1, 1, 1, 0)\boldsymbol{H}^\top. \tag{19}$$

3.2 Computational Procedure

Computing $\boldsymbol{\Omega}$ from Eq. (19), we can obtain the rectifying homography $\boldsymbol{H}$. For this, we modify the method of Seo and Heyden [8].

They iteratively updated only the principal point $(u_{\kappa0}, v_{\kappa0})$ so that $\boldsymbol{K}_\kappa^{-1}\boldsymbol{\Pi}_\kappa\boldsymbol{\Omega}\boldsymbol{\Pi}_\kappa^\top\boldsymbol{K}_\kappa^{-1\top}$ approaches a diagonal matrix; the focal length is given from its diagonal elements. We take a more systematic approach: we update the focal length f_κ as well so that it approaches a scalar multiple of the unit matrix $\boldsymbol{I}$ (we omit the details). As a result, we do not explicitly compute $\boldsymbol{\Omega}$ but directly compute the rectifying homography $\boldsymbol{H}$ and the intrinsic parameter matrix $\boldsymbol{K}_\kappa$.

Seo and Heyden [8] stop the iterations when the magnitude of the increment $(\delta u_{\kappa0}, \delta v_{\kappa0})$ is smaller than a specified threshold (e.g., 0.2 pixels). According to our experiments, however, the specified threshold is sometimes not reached in the presence of large noise, and it is difficult to predict a reachable threshold in advance. So, we introduced the weight W_κ that reflects the consistency of the κth frame and computed the median J_{med} over all the frames so that inconsistent frames are ignored (we omit the details). As a result, our scheme is confirmed to always converge in the presence of however large noise.

4. Acceleration Techniques

We ran the above algorithm and found that the iterations of projective reconstruction far exceed the Euclidean upgrading in computation time. In order to

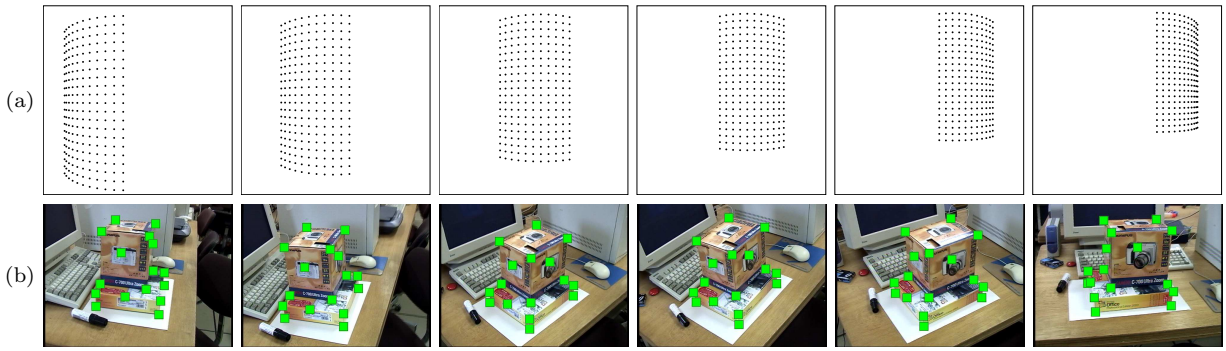

Figure 1: (a) Simulated image sequence (6 frames decimated), tracking 231 points through 11 frames. (b) Real video sequence (6 frames decimated), tracking 16 points through 200 frames.

reduce the computation time, we introduce the following techniques.

To prevent confusion as to which computation we are referring to, let us call, in both methods, Steps 3 $\sim$ 7 the *cycle* of subspace fitting and the substeps (a)$\sim$(d) in Step 5 the *adjustment* of the projective depths[2].

4.1 Power Method

In each cycle, we need to compute the eigenvectors of the $3M \times 3M$ $\boldsymbol{A}^\alpha$ N times for all α (primal) and of the $N \times N$ matrix $\boldsymbol{B}^\kappa$ M times for all κ (dual). However, the computed eigenvectors should not be very different from their values in the previous cycle, so we adopt the power method, which is an iterative algorithm. The computation becomes efficient if we start the iterations from the value in the previous cycle. Such a technique was used for efficiently computing the Tomasi-Kanade factorization [3, 7, 11].

Moreover, our algorithms have double loops of iterations, so as long as the iterations of the outer cycle have not converged, the iterations of the inner adjustment need not strictly converge. Hence, the overall computation time can reduce by relaxing the convergence of the power method in each cycle.

In our experiment, we stopped the power method iterations when the difference between the current and the previous eigenvectors is less than 10^{-5} in norm.

4.2 Power Method Acceleration

The power method is an iterative algorithm. So, it can also be accelerated. Suppose we compute the unit eigenvector $\boldsymbol{w}_1$ of a positive semidefinite symmetric matrix $\boldsymbol{T}$ for the largest eigenvalue λ_1. If we start from $\boldsymbol{\xi}^0$ and write $\boldsymbol{\xi}^k = N[\boldsymbol{T}^k\boldsymbol{\xi}^0]$, where $N[\,\cdot\,]$ denotes normalization to unit norm, we have for $\boldsymbol{\xi}^0$ close to $\boldsymbol{w}_1$ or for a large k

$$\boldsymbol{\xi}^k \approx \boldsymbol{w}_1 + C\gamma^k\boldsymbol{w}_2, \qquad \gamma = \frac{\lambda_2}{\lambda_1}, \tag{20}$$

where $\boldsymbol{w}_2$ is the eigenvector for the second largest eigenvalue λ_2, and C is a constant. Eliminating $\boldsymbol{w}_2$ from this and the same relation for $k+1$, we obtain

$$\boldsymbol{w}_1 \approx \frac{\boldsymbol{\xi}^{k+1} - \gamma\boldsymbol{\xi}^k}{1-\gamma}. \tag{21}$$

If γ is given, this allows us to predict $\boldsymbol{w}_1$ from $\boldsymbol{\xi}^{k+1}$ and $\boldsymbol{\xi}^k$. From Eq. (20), we can estimate γ by

$$\gamma \approx \frac{\|\boldsymbol{\xi}^{k+1} - \boldsymbol{\xi}^k\|}{\|\boldsymbol{\xi}^k - \boldsymbol{\xi}^{k-1}\|}. \tag{22}$$

[2]Our "cycle" and "adjustment" correspond, respectively, to the "E-" and the "M-steps" of the EM algorithm.

Using this, we estimate γ from $\boldsymbol{\xi}^{k-1}$, $\boldsymbol{\xi}^k$, and $\boldsymbol{\xi}^{k+1}$ and replace $\boldsymbol{\xi}^{k+1}$ by $N[(\boldsymbol{\xi}^{k+1} - \gamma\boldsymbol{\xi}^k)/(1-\gamma)]$, accelerating the iterations every other step (from $\boldsymbol{\xi}^0$ and $\boldsymbol{\xi}^1$ to $\boldsymbol{\xi}^2$, from $\boldsymbol{\xi}^2$ and $\boldsymbol{\xi}^3$ to $\boldsymbol{\xi}^4$, ...).

We applied this to the power method computation of the projective depth vectors $\boldsymbol{\xi}_\alpha$ and $\boldsymbol{\xi}_\kappa$ and stopped when the difference between the current and the previous values is less than 0.1 in norm.

4.3 SOR

A well known method of convergence acceleration is the SOR (successive overrelaxation): for a sequence ξ^1, ξ^2, ..., we accelerate ξ^k in the form

$$\xi^k \leftarrow \xi^{k-1} + \omega(\xi^k - \xi^{k-1}), \tag{23}$$

where ω (> 1) is called the *acceleration constant*. It has empirically been known that this scheme works well in many iterative problems, but an appropriate value of ω is very difficult to find except for special types of linear computation; it is usually set by experience.

We apply this scheme to accelerate the projective depth vector $\boldsymbol{\xi}$ ($= \boldsymbol{\xi}_\alpha$ or $\boldsymbol{\xi}_\kappa$) in each cycle using the value $\boldsymbol{\xi}'$ in the previous cycle, replacing $\boldsymbol{\xi}$ by $N[\boldsymbol{\xi}' + \omega(\boldsymbol{\xi} - \boldsymbol{\xi}')]$. I our experiments, we set $\omega = 1.9$.

5. Examples

Figure 1(a) shows six frames from a 11 image sequence of size 600 pixels that simulates perspective projection of a cylindrical surface with focal length 600 pixels. We used the 231 grid points for 3-D reconstruction.

Figure 1(b) shows six frames decimated from a 200 frame video sequence, tracking 16 points as indicated there. They are specified by hand in the first frame and tracked through the rest of the frames by the KLT (Kanade-Lucus-Tomasi) algorithm [9]; we manually intervened whenever the tracking failed.

Table 1 lists the computation time and the number of cycles for the sequence in Fig. 1(a) for different

Table 1: The computation time (sec) and the number of cycles for the sequence in Fig. 1(a) until the reprojection error is 0.1 pixels.

	primal		dual	
	time	cycles	time	cycles
prototype	3.84	89	7.15	3
power	2.24	90	1.25	3
accelerated power	2.37	90	**0.51**	3
SOR	1.12	47	10.76	21
both	1.27	47	5.73	31

efficiency index = 14

Table 2: The computation time (sec) and the number of cycles for the sequence in Fig. 1(b) until the reprojection error is 2.01 pixels.

	primal		dual	
	time	cycles	time	cycles
prototype	2,153.3	300	0.26	5
power	82.8	315	0.87	8
accelerated power	81.3	314	**0.26**	5
SOR	43.4	165	1.85	15
both	42.6	165	1.48	31

efficiency index = 8,282

Figure 2: The 3-D shape reconstructed from the video sequence of Fig. 1(b).

methods. We stopped the iterations of the outer cycle when the reprojection error is 0.1 pixels. We used Pentium 4 3.4GHz for the CPU with 2GB main memory and Linux for the OS. The "efficiency index" means the ratio of the longer computation time of the prototypes to the shortest of all.

Since the number of points is large ($M = 231$) and the number of frames is small ($M = 11$), the primal method is more efficient than the dual method if the prototype is used. Although the number of iterations is very small for the dual method, the computational burden of one iteration is so heavy that the overall convergence lags behind. However, the power method relieves this burden, making the dual method more efficient than the primal. Acceleration of the power method further reduces the computation time to about 0.7% of the prototype (14 times faster).

Table 2 is the corresponding results for Fig. 1(b). We stopped when the reprojection error is 2.01 pixels; due to the tracking accuracy limitation of the KLT, it did not reduce in further iterations.

Since the number of frame is large ($M = 200$), the primal method takes a vast amount of time if the prototype is used. The power method curtailed it to about 4% (26 times faster).

Nevertheless, all these cannot compare with the dual method. Already, the prototype converges so quickly that the use of the power method has an adverse effect. However, acceleration of the power method recovers the original efficiency, and the computation is about 8,000 times faster than the prototype primal method.

For both sequences in Fig. 1(a),(b), the power method converges very quickly for the primal method, its acceleration has little effect, while SOR can further speed up the computation. However, SOR has an adverse effect, and its combination with power method acceleration gains only a little.

Figure 2 shows the reconstructed 3-D shape as a texture mapped polyhedron having the tracked points as vertices; Euclidean upgrading took 0.58 sec. We used the method of Nakatsuji et al. [5] for optimizing the edges between the vertices so that they are compatible with the true shape.

6. Conclusions

We presented a robust and efficient scheme for 3-D reconstruction from point correspondences over multiple images taken by uncalibrated cameras. The main emphasis is on the acceleration of projective reconstruction. We observed the following:

1. The use of the power method dramatically reduce the computation time. This is especially conspicuous for the primal method.
2. Acceleration of the power method has little effect on the primal method but has a significant impact on the dual method.
3. SOR is effective for both the primal and the dual methods, but the effect is relatively small for the dual method.
4. The primal method is favorable for a very large number of points over a small number of frames; the dual method is preferable for a small number of points over a very large number of frames.
5. In realistic situations, the best choice is the dual method with power method acceleration.

However, the performance also depends on the termination condition of inner loop iterations and the choice of the acceleration constant. Further investigation is necessary for their optimal choice.

Acknowledgments: This work was supported in part by the Ministry of Education, Culture, Sports, Science and Technology, Japan, under the Grant in Aid for Scientific Research C(2) (No. 15500113). The authors thanks Akinobu Mori of Canon, Inc., Japan, for helping experiments. They also thank Isao Miyagawa of NTT, Japan, for helpful comments.

References

[1] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, U.K., 2000.

[2] R. I. Hartley, In defense of the eight-point algorithm, *IEEE Trans. Patt. Anal. Mach. Intell.*, **19**-6 (1997-7), 580–593.

[3] R. Hartley and F. Schaffalitzky, PowerFactorization: 3D reconstruction with missing or uncertain data, *Proc. the Australia-Japan Advanced Workshop on Computer Vision*, September 2003, Adelaide, Australia, pp. 1–9.

[4] A. Heyden, R. Berthilsson, and G. Sparr, An iterative factorization method for projective structure and motion from image sequences, *Image Vis. Comput.*, **17**-13 (1999-11), 981–991.

[5] A. Nakatsuji, Y. Sugaya and K. Kanatani, Optimizing a triangular mesh for shape reconstruction from images, *IEICE Trans. Inf & Syst.*, **E88-D**-10 (2005-10), 2269–2276

[6] S. Mahamud and M. Hebert, Iterative projective reconstruction from multiple views, *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, June 2000, Hilton Head Island, SC, U.S.A., Vol. 2, pp. 430–437.

[7] T. Morita and K. Kanade, A sequential factorization method for recovering shape and motion from image streams, *IEEE Trans. Patt. Anal. Mach. Intell.*, **19**-8 (1997-8), 858-867.

[8] Y. Seo and H. Heyden, Auto-calibration by linear iteration using the DAC equation, *Image Vis. Comput.*, **22**-11 (2004-9), 919–926.

[9] C. Tomasi and T. Kanade, *Detection and Tracking of Point Features*, CMU Tech. Rep., CMU-CS-91-132, Apr. 1991; `http://vision.stanford.edu/~birch/klt/`.

[10] B. Triggs, Autocalibration and the absolute quadric, *Proc. IEEE Conf. Comput. Vis. Patt. Recog.*, June 1997, San Juan, Puerto Rico, pp. 609–614.

[11] R. Vidal and R. Hartley, Motion segmentation with missing data using PowerFactorization and GPCA, *Proc. IEEE Comput. Vision Patt. Recog.*, June-July 2004, Washington, D.C., Vol. 2, pp. 310–316.