Pose Estimation from Multiple Cameras Based on Sylvester's Equation

Chong Chen, Dan Schonfeld

*Department of Electrical and Computer Engineering
University of Illinois at Chicago*

Abstract

In this paper, we introduce a method to estimate the object's pose from multiple cameras. We focus on direct estimation of the 3D object pose from 2D image sequences. Scale-Invariant Feature Transform (SIFT) is used to extract corresponding feature points from adjacent images in the video sequence. We first demonstrate that centralized pose estimation from the collection of corresponding feature points in the 2D images from all cameras can be obtained as a solution to a generalized Sylvester's equation. We subsequently derive a distributed solution to pose estimation from multiple cameras and show that it is equivalent to the solution of the centralized pose estimation based on Sylvester's equation. Specifically, we rely on collaboration among the multiple cameras to provide an iterative refinement of the independent solution to pose estimation obtained for each camera based on Sylvester's equation. The proposed approach to pose estimation from multiple cameras relies on all of the information available from all cameras to obtain an estimate at each camera even when the image features are not visible to some of the cameras. The resulting pose estimation technique is therefore robust to occlusion and sensor errors from specific camera views. Moreover, the proposed approach does not require matching feature points among images from

different camera views nor does it demand reconstruction of 3D points. Furthermore, the computational complexity of the proposed solution grows linearly with the number of cameras. Finally, computer simulation experiments demonstrate the accuracy and speed of our approach to pose estimation from multiple cameras.

Key words:

Pose Estimation, Distributed Estimation, Sylvester's Equation, Multiple Cameras, Multiple Views.

1. Introduction

Object pose estimation from monocular or multiple views has been one of the most active research topics over the past few decades. In many applications such as human-computer interaction, interactive-conferencing, and virtual reality, it is essential to monitor the rotation of the object out-of-image-plane (namely, pan and tilt).

The 3D rotation angles can be obtained by using various methods for pose estimation. A detailed discussion of many of the existing techniques for pose estimation from single and multiple cameras is provided in the following section. Despite the enormous advances in pose estimation, the problem of efficient and robust pose estimation from 2D image sequences from multiple cameras remains a difficult challenge. In this paper, we provide an extremely efficient and robust solution to pose estimation from 2D image sequences from multiple cameras based on Sylvester's equation.

We first present a solution to pose estimation from a single camera from 2D image sequences based on Sylvester's equation. In particular, we rely on the feature-based approach to directly estimate the 3D pose from 2D image sequences.

Scale-Invariant Feature Transform (SIFT) (1) is used to extract corresponding feature points from adjacent images in the video sequence. We introduce a unified method for pose estimation based on Sylvester’s equation (2). This approach can be shown to be equivalent to the classical approach to pose estimation based on Singular Value Decomposition (SVD) for 3D models (3). However, unlike pose estimation based on SVD methods, our approach is not limited to 3D models and provides an elegant solution to pose estimation directly from 2D image sequences that does not require any training.

We subsequently extend our approach to pose estimation from 2D image sequences from multiple cameras. We extend our approach to pose estimation by forming the problem of centralized pose estimation from multiple cameras as a solution to a generalized Sylvester’s equation capturing all of the feature points from all cameras. We then derive a distributed solution to the generalized Sylvester’s equation by collaboration among the cameras that relies on iterative refinement of the independent solution to pose estimation based on Sylvester’s equation for each camera. We demonstrate that both the centralized and distributed pose estimation from multiple cameras generate superior estimates than the results of pose estimation from any specific camera view. The proposed approach to pose estimation can consequently be used to improve the estimate for any single camera or provide robust pose estimation from a virtual camera view.

The remainder of the paper is organized as follows: In Section 2, we provide a brief overview of various methods developed for pose estimation from single and multiple cameras. In Section 3, we introduce the method of pose estimation from 2D image sequences from a single camera based on Sylvester’s equation. We extend this approach to pose estimation from 2D image sequences from multiple

cameras in Section 4. We first present a centralized approach to pose estimation from 2D image sequences from multiple cameras as a solution to a generalized Sylvester’s equation. We then provide a distributed solution to pose estimation from multiple cameras by iterative refinement of the independent solution to pose estimation from each camera. In Section 5, we conduct computer simulation experiments and demonstrate the robustness and efficiency of the proposed approach to pose estimation from multiple cameras. Finally, we present a brief summary and discussion of our results in Section 6.

2. Related Work

As our pose estimation approach for multiple camera views is derived from the solution with only one camera, in the following, we will firstly introduce the method of pose estimation from the video sequences of one camera, and then extend to the multi-camera case.

2.1. Pose Estimation from Monocular Camera

In the monocular view case, many approaches have been proposed during the past years, and most can be classified into two major categories: (i) feature-based methods and (ii) appearance-based methods. Appearance-based methods often rely on some training to obtain templates for object pose estimation. Feature-based methods rely on corresponding features from different images. Solutions with redundant data can be classified into two classes: (i) linear methods and (ii) nonlinear methods (4). Nonlinear solutions are generally more robust to noise; however, they suffer from heavy computation, usually require a good initialization, and most cannot guarantee convergence. While linear methods require less

computation, the results are often inferior due to lack of accuracy and corruption by noise.

Ji et al. (4) develop a linear least-squares framework for multiple geometric features including points, lines, and ellipse-circle pairs. Orthonormality constraints due to rotation are approximately imposed within the linear framework. In (5), the transform between feature matches is computed with a hierarchical RANSAC approach. The object pose estimation from corresponding points based on SVD techniques have been well established (3)(6)(7)(8). Horn's algorithm computes the eigensystem of a derived matrix and is similar to the SVD approach (9). Derivation of a closed form solution to this problem can be simplified by using unit quaternions to represent rotation as shown in (10). Olsson et al. (11) generalize the method in (9) by incorporating point, line and plane features in a common framework for finding the globally optimal solution to the problem of pose estimation, but it can only deal with a known object and also parameterize rotations with quaternions. Moreover, a reformulation of the same problem can be represented using the essential matrix (3)(12).

The solutions discussed above can be shown to provide the optimal estimate for pose estimation from 3D points while satisfying the orthogonality of the rotation matrix, e.g. SVD-based pose estimation. Unfortunately, these methods cannot be used for solution from 2D projected feature points since in this case the orthogonality constraints are not satisfied. Instead, in the 2D case, one must incorporate pseudo-orthogonality constraints which capture the projection of the 3D rotation on the image plane. In the existing solutions to these problem, the pseudo-orthogonality constraints are weakly enforced and often ignored. Many methods rely on an unconstrained solution to the pose estimation problem and sub-

sequently adjust the solution to satisfy the pseudo-orthogonality constraints for the 2D model.

We employ the feature-base approach and rely on corresponding 2D points from images to directly estimate the 3D pose while incorporating the pseudo-orthogonality constraints. We demonstrate that pose estimation can be obtained as a solution of Sylvester’s equation, which can be solved with many methods such as Kronecker Product (13) and Bartels-Stewart approach (14). This solution is proved to be equivalent to the SVD-based methods for 3D-3D pose estimation, yet it can also be used for the 2D cases.

2.2. Pose Estimation from Multiple Cameras

The reconstruction from multi-view stereo has received a large amount of attention over the past few decades. In (15), the authors provide a quantitative comparison of several multi-view stereo reconstruction algorithms on their datasets with ground truth. Further research that could be used to recover the 3D structure, including 3D shape and 3D motion, from 2D motion in the image plane is generally referred to as structure-from-motion (SFM) (16). A framework for various camera models is introduced in (17), which provide non-linear methods based on point correspondence across views. SFM methods firstly determine the 2D motion in the image plane and then estimate the 3D shape and 3D motion. 3D reconstruction and shape analysis are beyond the scope of this paper.

In (18), a method is proposed for arbitrary view synthesis from an uncalibrated multiple camera system. In (19), the multiple view matching is achieved by a combination of image invariants, covariants, and multiple view relations. Sturm (20) models cameras as possibly unconstrained sets of projection rays and introduces a hierarchy of general camera models. He also establishes the foundations for

a multi-view geometry of general (non-central) cameras, analogously to the perspective case. Yu and McMillan (21) present a General Linear Camera model to describe many camera models. In (22), structured light patterns are used to extract the raxel parameters of an imaging system, to present a general imaging model. In this framework, Press (23) derives the discrete SFM equations for generalized cameras, and illustrates the connections to the epipolar geometry.

Chang and Chen (24) conclude that pose estimation for a multiple camera system is usually solved by perspective-n-point (PnP) methods or Least Square approaches. Kahl (25) presents a framework to solve geometric structure and motion parameters based on L_∞ -norm, which can be applied for efficient computation of global estimates. In (26), the fractional programming and the theory of convex underestimators are relied to unify the framework for minimizing the standard L_2 -norm of reprojection errors. In this paper, we develop two numerical algorithms with the observations from all cameras, and also rely on the information from other cameras to update the first camera, to make its pose estimate more robust.

Rother and Carlsson (27), based on a reference plane, develop a linear algorithm for computation of 3D points and camera positions from multiple perspective views by finding the null-space of a matrix built from image data using SVD. In (28), a system, consisting of six cameras, is built to remove the inherent ambiguities of confusion between translation and rotation. However, this system does not use one pose estimation for all information from all cameras simultaneously. Frahm et al. (29) combine all information of all cameras to estimate the pose of a multi-camera system. In contrast, we are estimating the object's pose, and by using all information from all cameras to give one pose estimate, we also research

the relationship between this estimate with those from each camera separately.

Object pose estimation from multiple views can also be achieved with appearance or learning based methods. Wang et al. (30) integrate both asymmetric and symmetric rectangle features, AdaBoost learning algorithm and pyramid like architecture for the multi-view face detection and pose estimation in video stream. In (31), face detectors are applied in each camera view, and then the head pose with respect to the room is estimated based on supervised learning. However, there are only five face detectors for frontal, half profile and profile faces, and the system can just output eight possible directions. Similarly, the system based on neural networks (32) can also recognize eight orientations. A learning based procedure with probabilistic boosting network (33) is introduced for joint object detection and pose estimation.

In our system, we are assuming the cameras are fixed, and the orientations and translations between each other are known, similarly to (29). Initially, we suppose that all of the cameras could capture the same feature points in the object, but our conclusion is independent of this assumption. Therefore, in our method, there is no feature matching between images from different cameras, which is a difficult classical problem. We rely on the pose estimation from other cameras to improve the estimate of the first one, and vice versa, and we can estimate pose from a virtual view as well.

This paper is an extension of our previous work on pose estimation from a single camera (34) to multiple-cameras. In (35), we relied on Best Linear Unbiased Estimation (BLUE) to provide an estimate of the pose results from multiple cameras. However, the pose in each camera was estimated independently and the BLUE estimate is only used to provide a weighted average among the different

estimates, thus providing an optimal balance among the pose estimates from each camera. BLUE estimation was used to reduce large pose estimation errors from specific camera views. However, the BLUE estimate inherently results an inferior pose than the estimate provided by pose estimation from the best camera view. By contrast, in this paper, we propose centralized and distributed methods to simultaneously use all of the observations in all cameras-not only the results of pose estimation from independent cameras-to solve a generalized Sylvester’s equation problem. The proposed approach directly estimates the pose from the observations in all of the cameras. Pose estimation using the method proposed in this paper has thus been shown to provide much superior results than pose estimation from any of the individual cameras.

3. Pose Estimation from Monocular View

The proposed approach for pose estimation is based on corresponding points. In our system, we only consider the feature points within the object region in each image. Therefore, a tracking algorithm must firstly be applied to extract the region of interest. The tracking system used in our approach is based on motion-based particle filtering (MBPF) (36). In MBPF, Adaptive Block Matching (ABM) is first used for motion estimation, which characterizes the mean of a Gaussian density function for importance sampling (37).

Given the mean estimate at time $k - 1$, i.e., $\hat{\mathbf{x}}_{k-1}$, to estimate the position of the target at frame k , frame $k - 1$ is divided into 16×16 (pixels) rectangular blocks. The blocks lying entirely inside $\hat{\mathbf{x}}_{k-1}$ are labelled seed blocks, which can be completely defined by one motion vector. The blocks that lie in the boundary are labelled uncertain. Multiple motions are possible for these blocks and hence

they are further divided into seed and uncertain blocks. This procedure is continued until a fixed size is reached. In our implementation, we stop at 8×8 (pixels) blocks. Only the motion of the seed blocks is estimated as follows: Within a certain window of size $C \times C$ of frame k , each seed block in frame $k - 1$ is searched. In our implementation, $C = 33$, corresponding to a horizontal and vertical maximum motion vector of value 16 pixels. The position difference between the block at frame $k - 1$ and the block that turns out to be the most similar in the frame k represents the motion vector of that particular block. One can consider various criteria as a measure of the match between two blocks. In this chapter, we use the absolute value of the sum of intensity differences as the similarity criterion between two blocks.

Initially, the ABM output is a mask (i.e. union of rectangular blocks), covering the estimated position of the target. In our system, an ellipse is used to optimally fit the mask using the Least Squares method. The center of the ellipse is used to represent the 2D translation of the object from the last position. The SIFT features are extracted from the entire image, and only those within the area of the ellipse are kept and used for the solution of Sylvester's equation which provide the 3D rotation estimate. Both the 2D translation from ABM as well as the 3D rotation parameters provided by Sylvester's equation are used to define the mean of the importance sampling density function in particle filtering.

After SIFT is used to extract corresponding feature points from image sequences, RANSAC (38) is applied to remove the false matched pairs. We begin with a known pose, estimate the pose change between two successive frames in a video sequence, and obtain the current state by cumulating all of the past results. Tracking is realized with edge and color based particle filtering.

3.1. Projection from 3D to 2D

Given a point in the 3D real object, its positions before and after motion have the following relationship:

$$\begin{pmatrix} x_{t+1}^j \\ y_{t+1}^j \\ z_{t+1}^j \end{pmatrix} = R_{3 \times 3} \begin{pmatrix} x_t^j \\ y_t^j \\ z_t^j \end{pmatrix} + T_{3 \times 1}, \quad (1)$$

where $R_{3 \times 3}$ is a rotation matrix, and $T_{3 \times 1}$ is a translation vector. The subscript 3×3 and 3×1 point out the matrix dimensions. j , ($j = 1, 2, \dots$) is the index of matched feature pairs.

The problem is to determine the rotation angles and translation vector from the matched points between two successive images. Here, we adopt the image projection model based on the camera pinhole model presented in (3)(6)(39). This model assumes that the relative depth within the object is small compared to the distance of the object to the camera. Then we obtain

$$\frac{x_k^j}{z_k^j} = \frac{u_k^j}{f}, \quad \frac{y_k^j}{z_k^j} = \frac{v_k^j}{f}, \quad k = t, t+1, \quad (2)$$

where f is the focal length, $(u_t^j \ v_t^j)^T$ and $(u_{t+1}^j \ v_{t+1}^j)^T$ are the corresponding projected points in the images respectively. By integrating Eq. (2) with Eq. (1), we obtain

$$\begin{pmatrix} u_{t+1}^j \\ v_{t+1}^j \\ f \end{pmatrix} = \frac{z_t^j}{z_{t+1}^j} R_{3 \times 3} \begin{pmatrix} u_t^j \\ v_t^j \\ f \end{pmatrix} + T'_{3 \times 1}, \quad (3)$$

where

$$T'_{3 \times 1} = \begin{pmatrix} t'_x \\ t'_y \\ f \end{pmatrix} = \frac{f}{z_{t+1}^j} T_{3 \times 1} \quad (4)$$

is the 3×1 translation vector within the image plane. However, the ratio z_t^j/z_{t+1}^j is unknown. Here, we regard $z_t^j/z_{t+1}^j = 1$. A similar model, the weak projective camera model (40), is used with known 3D points, but here we only have 2D feature points and never recover 3D points. This assumption is valid when the object is comparably far from the camera and the depth change is small between successive frames in the video sequence (usually less than 1/24 second). For clarity, in the remainder of this presentation, we assume that the ratio of the depth associated with corresponding features in adjacent frames is unity. Furthermore, if only the first two rows of Eq. (3) are considered, we have

$$\begin{pmatrix} u_{t+1}^j \\ v_{t+1}^j \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \end{pmatrix} \begin{pmatrix} u_t^j \\ v_t^j \\ f \end{pmatrix} + \begin{pmatrix} t'_x \\ t'_y \end{pmatrix}. \quad (5)$$

By rearranging the matrix equation, we obtain

$$\begin{pmatrix} u_{t+1}^j \\ v_{t+1}^j \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix} \begin{pmatrix} u_t^j \\ v_t^j \end{pmatrix} + \begin{pmatrix} l_x \\ l_y \end{pmatrix}, \quad (6)$$

where $l_x = r_{13}f + t'_x$ and $l_y = r_{23}f + t'_y$ are the new translation parameters.

3.2. Pose Estimation Based on Sylvester's Equation

The discussion above is presented with one pair of points. Considering all of the matched points obtained from SIFT, we can compute the Least-Squares estimate with respect to the rotation and translation variables. Thus, the problem of pose estimation from image sequences is given by

$$\begin{aligned} \min_{R_1, R_2, L_x, L_y} & (U'_{t+1} - R_1 P'_t - L_x)(U'_{t+1} - R_1 P'_t - L_x)^T \\ & + (V'_{t+1} - R_2 P'_t - L_y)(V'_{t+1} - R_2 P'_t - L_y)^T, \end{aligned} \quad (7)$$

subject to

$$\begin{cases} R_1 R_1^T + r_{13}^2 = 1 \\ R_2 R_2^T + r_{23}^2 = 1 \\ R_1 R_2^T + r_{13} r_{23} = 0 \end{cases},$$

where $R_{2 \times 2} = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix}$, the top-left four elements of $R_{3 \times 3}$,

$$P'_{t+1} = \begin{pmatrix} U'_{t+1} \\ V'_{t+1} \end{pmatrix} = \begin{pmatrix} u_{t+1}^1 & u_{t+1}^2 & \cdots \\ v_{t+1}^1 & v_{t+1}^2 & \cdots \end{pmatrix}, P'_t = \begin{pmatrix} u_t^1 & u_t^2 & \cdots \\ v_t^1 & v_t^2 & \cdots \end{pmatrix} \text{ and } L = \begin{pmatrix} L_x \\ L_y \end{pmatrix} = \begin{pmatrix} l_x & l_x & \cdots \\ l_y & l_y & \cdots \end{pmatrix}.$$

We apply the Lagrange multiplier method to solve the constrained optimization problem, and obtain

$$\begin{aligned} F = & (U'_{t+1} - R_1 P'_t - L_x)(U'_{t+1} - R_1 P'_t - L_x)^T \\ & + (V'_{t+1} - R_2 P'_t - L_y)(V'_{t+1} - R_2 P'_t - L_y)^T \\ & + \lambda_1(R_1 R_1^T + r_{13}^2 - 1) + \lambda_2(R_2 R_2^T + r_{23}^2 - 1) + 2\lambda_3(R_1 R_2^T + r_{13} r_{23}), \end{aligned} \quad (8)$$

where λ_1 , λ_2 and λ_3 are Lagrange multipliers. Then, the partial derivatives of F are taken with respect to the rotation and translation coefficients to yield

$$\begin{cases} \frac{\partial F}{\partial R_1} = -U'_{t+1} P_t'^T + R_1 P'_t P_t'^T + \lambda_1 R_1 + \lambda_3 R_2 + L_x P_t'^T = 0 \\ \frac{\partial F}{\partial R_2} = -V'_{t+1} P_t'^T + R_2 P'_t P_t'^T + \lambda_2 R_2 + \lambda_3 R_1 + L_y P_t'^T = 0 \end{cases}, \quad (9)$$

$$\begin{cases} \frac{\partial F}{\partial L_x} = -U'_{t+1} + R_1 P'_t + L_x = 0 \\ \frac{\partial F}{\partial L_y} = -V'_{t+1} + R_2 P'_t + L_y = 0 \end{cases}. \quad (10)$$

From Eq. (10), we have

$$L = \begin{pmatrix} L_x \\ L_y \end{pmatrix} = \begin{pmatrix} U'_{t+1} - R_1 P'_t \\ V'_{t+1} - R_2 P'_t \end{pmatrix}. \quad (11)$$

We notice that there is only one translation vector for all of the feature points within a frame. Thus, the translation vector can be determined from the following equation once the rotation parameters are known (3):

$$\begin{pmatrix} l_x \\ l_y \end{pmatrix} = \begin{pmatrix} \bar{u}_{t+1} \\ \bar{v}_{t+1} \end{pmatrix} - \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix} \begin{pmatrix} \bar{u}_t \\ \bar{v}_t \end{pmatrix}, \quad (12)$$

where $\begin{pmatrix} \bar{u}_t \\ \bar{v}_t \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^M u_t^j / M \\ \sum_{j=1}^M v_t^j / M \end{pmatrix}$ and $\begin{pmatrix} \bar{u}_{t+1} \\ \bar{v}_{t+1} \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^M u_{t+1}^j / M \\ \sum_{j=1}^M v_{t+1}^j / M \end{pmatrix}$, the mean of the feature points from two sequential images respectively. M is the number of feature points.

Define $\bar{P}'_{t+1} = \begin{pmatrix} \bar{U}'_{t+1} \\ \bar{V}'_{t+1} \end{pmatrix} = \begin{pmatrix} \bar{u}_{t+1} & \bar{u}_{t+1} & \cdots \\ \bar{v}_{t+1} & \bar{v}_{t+1} & \cdots \end{pmatrix}$ and $\bar{P}'_t = \begin{pmatrix} \bar{U}'_t \\ \bar{V}'_t \end{pmatrix} = \begin{pmatrix} \bar{u}_t & \bar{u}_t & \cdots \\ \bar{v}_t & \bar{v}_t & \cdots \end{pmatrix}$. Then, Eq. (11) is rewritten as

$$L = \begin{pmatrix} L_x \\ L_y \end{pmatrix} = \begin{pmatrix} \bar{U}'_{t+1} - R_1 \bar{P}'_t \\ \bar{V}'_{t+1} - R_2 \bar{P}'_t \end{pmatrix}. \quad (13)$$

By substituting L in Eq. (9), we obtain

$$\begin{cases} -(U'_{t+1} - \bar{U}'_{t+1})P_t'^T + R_1(P'_t - \bar{P}'_t)P_t'^T + \lambda_1 R_1 + \lambda_3 R_2 = 0 \\ -(V'_{t+1} - \bar{V}'_{t+1})P_t'^T + R_2(P'_t - \bar{P}'_t)P_t'^T + \lambda_2 R_2 + \lambda_3 R_1 = 0 \end{cases}. \quad (14)$$

Define $P_{t+1} = P'_{t+1} - \bar{P}'_{t+1}$, $P_t = P'_t - \bar{P}'_t$, $A = P_t P_t'^T$ and $B = P_{t+1} P_t'^T$.

Then, Eq. (14) can be simplified as

$$\begin{cases} -U_{t+1} P_t'^T + R_1 A + \lambda_1 R_1 + \lambda_3 R_2 = 0 \\ -V_{t+1} P_t'^T + R_2 A + \lambda_3 R_1 + \lambda_2 R_2 = 0 \end{cases}, \quad (15)$$

which can be further combined into a matrix equation:

$$-B + R_{2 \times 2} A + \Lambda R_{2 \times 2} = 0, \quad (16)$$

where $\Lambda = \begin{pmatrix} \lambda_1 & \lambda_3 \\ \lambda_3 & \lambda_2 \end{pmatrix}$ is the Lagrange multiplier matrix, and its elements are introduced in Eq. (8). Eq. (16) is known as Sylvester's equation. It is also called Lyapunov's equation in the special case where $\Lambda = A^T$.

Once Sylvester's equation has been solved, the rotation angles between successive frames are obtained. If we assume that we begin with a known pose (for example, the frontal view of the face), then the estimate of the current pose is acquired by accumulating all of the past pose results. The initial pose can be determined by using various methods including template-matching techniques (41).

3.3. Solution of Sylvester's Equation

For the general Sylvester's equation problem (13)

$$S_1 R W_1^T + \cdots + S_n R W_n^T = B, \quad (17)$$

an example solvent with the *vec* operator and Kronecker Product (13) is given by

$$\text{vec}(R) = (W_1 \otimes S_1 + \cdots + W_n \otimes S_n)^{-1} \text{vec}(B). \quad (18)$$

Especially, the Sylvester's equation is solved with

$$\text{vec}(R_{2 \times 2}) = (I \otimes \Lambda + A \otimes I)^{-1} \text{vec}(B). \quad (19)$$

where I is the identity matrix. Sylvester's equation can also be solved with the Bartels-Stewart method (2), the Hessenberg-Schur approach (42), the subspace techniques, and the numerical calculations.

3.4. Lagrange Multipliers

In our case, the challenge for solving Sylvester's equation is that the matrix Λ is unknown. In our experiments, we first determine an approximation of the matrix

$R_{2 \times 2}$. For example, we can obtain an initial estimate of Λ with the assumption that $R_{2 \times 2}$ is an identity matrix (i.e. all three rotation angles are zero):

$$\Lambda_0 = B - A = P_{t+1}P_t'^T - P_tP_t'^T . \quad (20)$$

This initial estimate is reasonable as we only focus on the rotation between two successive images at one time, and these angles are generally close to 0. Then, we can obtain a symmetric Λ_1 by replacing $\Lambda_0(1, 2)$ and $\Lambda_0(2, 1)$ with $(\Lambda_0(1, 2) + \Lambda_0(2, 1))/2$, getting the first evaluation. We must also iteratively compute Λ_i to ensure that the following equation is satisfied:

$$R_1R_2^T \pm \sqrt{(1 - R_1R_1^T)(1 - R_2R_2^T)} = 0 . \quad (21)$$

To choose whether $+$ or $-$ can be decided by the direction of the rotation. This computation is usually very fast and converges in only a few iterations. If the iteration fails to converge after 10 iterations, we discard the computation and maintain the pose estimate from the previous frame.

The pose estimation method proposed based on the solution to Sylvester's equation updates the change in the pose estimate between successive images. Like all such methods, our approach could potentially suffer from error propagation in the video sequence, and we rely on particle filtering (34) to minimize the potential for error accumulation in the pose estimate.

4. Pose Estimation from Multiple Views

4.1. Geometry in Multiple Camera Systems

We have shown object pose estimation from an uncalibrated camera, and in the following, we will use calibrated cameras for multiple views, and actually each

camera can be calibrated separately. For clarity, we begin our discussion with only two camera views. However, the ideas presented here can be easily extended to multiple views; i.e. set a fixed camera view, and project pose estimate from every other camera to the fixed view with the corresponding rotation between these two cameras.

To make a bridge between two camera views, we firstly suppose that all cameras could capture the same points in the object, but our conclusion is independent of this assumption. Moreover, we begin with 3D object points to construct various transform, but only 2D image features are known and finally used. Accordingly, we have the relations:

$$Q_{t+1} = R_{3 \times 3} Q_t + T_{3 \times 1} , \quad (22)$$

$$Q^r = R_{3 \times 3}^o Q^l + T_{3 \times 1}^o . \quad (23)$$

where the subscript t and $t + 1$ represent the coordinates before and after movement, and the superscript l or r stands for *left* camera or *right* camera coordinate system. Consequently, Q_t^l are the 3D points in the *left* camera coordinate system, and Q_t^r are the same points but in the *right* camera coordinate system. $R_{3 \times 3}^o$ and $T_{3 \times 1}^o$ are rotation and translation transform between two cameras, and both are known. The above two equations can be specified as:

$$\begin{cases} Q_{t+1}^l = R_{3 \times 3}^l Q_t^l + T_{3 \times 1}^l , \\ Q_{t+1}^r = R_{3 \times 3}^r Q_t^r + T_{3 \times 1}^r \end{cases} , \quad (24)$$

$$\begin{cases} Q_{t+1}^r = R_{3 \times 3}^o Q_{t+1}^l + T_{3 \times 1}^o , \\ Q_t^r = R_{3 \times 3}^o Q_t^l + T_{3 \times 1}^o \end{cases} . \quad (25)$$

From Eq. (24) and Eq. (25), we obtain

$$Q_{t+1}^r = R_{3 \times 3}^r Q_t^r + T_{3 \times 1}^r$$

$$\begin{aligned}
&= R_{3 \times 3}^r R_{3 \times 3}^o Q_t^l + R_{3 \times 3}^r T_{3 \times 1}^o + T_{3 \times 1}^r \\
&= R_{3 \times 3}^o Q_{t+1}^l + T_{3 \times 1}^o \\
&= R_{3 \times 3}^o R_{3 \times 3}^l Q_t^l + R_{3 \times 3}^o T_{3 \times 1}^l + T_{3 \times 1}^o .
\end{aligned} \tag{26}$$

As Q_t^l is arbitrary, for the rotation matrix, we have

$$R_{3 \times 3}^r = R_{3 \times 3}^o R_{3 \times 3}^l (R_{3 \times 3}^o)^T , \tag{27}$$

$$R_{3 \times 3}^l = (R_{3 \times 3}^o)^T R_{3 \times 3}^r R_{3 \times 3}^o . \tag{28}$$

And for the translation vector, we have

$$T_{3 \times 1}^r = R_{3 \times 3}^o T_{3 \times 1}^l + T_{3 \times 1}^o - R_{3 \times 3}^r T_{3 \times 1}^o . \tag{29}$$

As can be seen, the relationship between rotation, Eqs. (27) and (28), and translation, Eq. (29), is independent of the particular points used as long as they lie on the same object. Therefore, we can use a different set of feature pairs for pose estimation as long as the feature pairs are from the same object and on the same image. Furthermore, beginning with the definition of the pose estimation problem in Eq. (31), our only concern is the relationship between the camera coordinate systems; that is, the different points observed in each camera may be distinct. Accordingly, there is no assumption that all cameras could capture the same points in the object in the proposed method for pose estimation starting in Section 4.2.

4.2. Estimation of the Translation

As explained, $(l_x \ l_y)^T$ can be calculated from Eq. (13). Then with Eq. (6), we can obtain $(t'_x \ t'_y)^T$ from $(l_x \ l_y)^T$ once the focal length f^l and f^r are known. Furthermore, from Eq. (4), Eq. (29) can be rewritten as

$$\frac{z_{t+1}^r}{f^r} T_{3 \times 1}^{r'} = \frac{z_{t+1}^l}{f^l} R_{3 \times 3}^o T_{3 \times 1}^{l'} + T_{3 \times 1}^o - R_{3 \times 3}^r T_{3 \times 1}^o . \tag{30}$$

We can obtain z_{t+1}^l and z_{t+1}^r from Eq. (30), and then the 3D translation $T_{3 \times 1}$ can be computed with Eq. (4).

4.3. Estimation of the Rotation

Although $(l_x \ l_y)^T$ can only be computed after estimating rotation matrix, it can be removed from Eq. (6) by Eq. (13), such that only the rotation matrix remains in Eq. (6). We will focus on the rotation estimation in the following. Then, given two views, the pose estimation problem is concluded as

$$\min_{R^l, R^r} \text{Trace} \begin{bmatrix} (P_{t+1}^l - R^l P_t^l)(P_{t+1}^l - R^l P_t^l)^T \\ + (P_{t+1}^r - R^r P_t^r)(P_{t+1}^r - R^r P_t^r)^T \end{bmatrix}, \quad (31)$$

subject to

$$R^l (R^l)^T = \begin{pmatrix} 1 - r_{13}^{l2} & -r_{13}^l r_{23}^l \\ -r_{13}^l r_{23}^l & 1 - r_{23}^{l2} \end{pmatrix}, \quad (32)$$

$$R^r (R^r)^T = \begin{pmatrix} 1 - r_{13}^{r2} & -r_{13}^r r_{23}^r \\ -r_{13}^r r_{23}^r & 1 - r_{23}^{r2} \end{pmatrix}, \quad (33)$$

where P^l and P^r are the feature points in the images minus their corresponding mean; R^l and R^r are the top-left four elements of $R_{3 \times 3}^l$ and $R_{3 \times 3}^r$ respectively. Eqs. (32) and (33) are constraints for *left* and *right* views because of the orthogonality of rotation matrix, and they are also the matrix representation of the constraints in Eq. (7). Eq. (31) is similar but not the same as Eq. (7) because we have subtracted the mean in the definition of the problem of pose estimation from multiple cameras. Here we also apply the Lagrange Multiplier Method to solve the constrained optimization problem:

$$F = \text{Trace} \begin{bmatrix} (P_{t+1}^l - R^l P_t^l)(P_{t+1}^l - R^l P_t^l)^T \\ + (P_{t+1}^r - R^r P_t^r)(P_{t+1}^r - R^r P_t^r)^T \end{bmatrix}$$

$$+\Lambda^l(R^l R^{lT} - G^l) + \Lambda^r(R^r R^{rT} - G^r) \big], \quad (34)$$

where $\Lambda^l = \begin{pmatrix} \lambda_1 & \lambda_3 \\ \lambda_3 & \lambda_2 \end{pmatrix}$ and $\Lambda^r = \begin{pmatrix} \lambda_4 & \lambda_6 \\ \lambda_6 & \lambda_5 \end{pmatrix}$ are the Lagrange multiplier matrices, and $G^l = \begin{pmatrix} 1 - r_{13}^{l2} & -r_{13}^l r_{23}^l \\ -r_{13}^l r_{23}^l & 1 - r_{23}^{l2} \end{pmatrix}$ and $G^r = \begin{pmatrix} 1 - r_{13}^{r2} & -r_{13}^r r_{23}^r \\ -r_{13}^r r_{23}^r & 1 - r_{23}^{r2} \end{pmatrix}$ (see also Eqs. (32) and (33)).

4.4. Centralized Solution Based on Sylvester's Equation

With the centralized solution, we obtain pose estimation in one camera coordinate system with the observations from all of cameras. From Eq. (27), we obtain

$$R^r = R_{2 \times 3}^o R_{3 \times 3}^l (R_{2 \times 3}^o)^T = R^o R^l (R^o)^T + H, \quad (35)$$

where $R_{2 \times 3}^o$ is the top two lines of $R_{3 \times 3}^o$, R^o contains the top-left four elements of $R_{3 \times 3}^o$. and $H = \begin{pmatrix} r_{13}^o \\ r_{23}^o \end{pmatrix} \begin{pmatrix} r_{31}^l & r_{32}^l \end{pmatrix} (R^o)^T + \begin{pmatrix} r_{11}^o r_{13}^l + r_{12}^o r_{23}^l + r_{13}^o r_{33}^l \\ r_{21}^o r_{13}^l + r_{22}^o r_{23}^l + r_{23}^o r_{33}^l \end{pmatrix} \begin{pmatrix} r_{13}^o & r_{23}^o \end{pmatrix}$. Replacing R^r in Eq. (34) with Eq. (35), we have

$$\begin{aligned} F = \text{Trace} \{ & (P_{t+1}^l - R^l P_t^l)(P_{t+1}^l - R^l P_t^l)^T + \Lambda^l(R^l R^{lT} - G^l) \\ & + [P_{t+1}^r - (R^o R^l R^{oT} + H)P_t^r][P_{t+1}^r - (R^o R^l R^{oT} + H)P_t^r]^T \\ & + \Lambda^r[(R^o R^l R^{oT} + H)(R^o R^l R^{oT} + H) - G^r] \}. \end{aligned} \quad (36)$$

Define $A^l = P_t^l P_t^{lT}$, $B^l = P_{t+1}^l P_t^{lT}$, $A^r = P_t^r P_t^{rT}$ and $B^r = P_{t+1}^r P_t^{rT}$. Then, the partial derivatives of F are taken with respect to R^l to yield

$$\frac{\partial F}{\partial R^l} = -B^l + R^l A^l + \Lambda^l R^l - R^{oT} B^r R^o + R^{oT} R^o R^l R^{oT} A^r R^o + R^{oT} H A^r R^o$$

$$\begin{aligned}
& +R^{oT}\Lambda^r R^o R^l R^{oT} R^o + R^{oT}\Lambda^r H R^o \\
& = -B^l + R^l A^l + \Lambda^l R^l \\
& \quad + R^{oT}[-B^r + (R^o R^l R^{oT} + H)A^r + \Lambda^r (R^o R^l R^{oT} + H)]R^o \\
& = 0 .
\end{aligned} \tag{37}$$

Once Eq. (37) is solved, the object's pose in the *left* camera coordinate system is obtained, with the information from both *left* and *right* cameras.

Eq. (37) is the general Sylvester's equation. Since R^o is the known transform between two cameras, and Λ^r and Λ^l can be evaluated separately with the method developed for the single camera case, consequently, once H is known, we can directly solve R^l from Eq. (37) (see Section 3.3)

$$\begin{aligned}
vec(R^l) = & [A^{lT} \otimes I + I \otimes \Lambda^l + (R^{oT} A^r R^o)^T \otimes (R^{oT} R^o) \\
& + (R^{oT} R^o)^T \otimes (R^{oT} \Lambda^r R^o)]^{-1} \\
& vec(B^l + R^{oT} B^r R^o - R^{oT} H A^r R^o - R^{oT} \Lambda^r H R^o) , \tag{38}
\end{aligned}$$

where I is the identity matrix. On the other hand, if the rotation matrix R^l could be computed, H can be calculated with Eq. (35). Consequently, we develop the following numerical algorithm. Note that Eq. (38) mainly involves the inverse of a 4×4 matrix, which does not change during iterations of computation. Therefore it is just computed once and stored in the memory, and Algorithm 1 will be very fast. Moreover there is only one H term in the two-camera case, and for the N -camera case, there are $(N - 1)$ H terms which are all treated similarly.

Algorithm 1 *Centralized Solution for N Cameras*

compute R from Eq. (38) with the assumption that all H are zeros.

repeat

1. Calculate H for each other camera view with the computed R and Eq. (35).
2. Put all H back into Eq. (38) and compute R .

until R does not change much or the preset iteration number is achieved.

4.5. Distributed Solution Based on Sylvester's Equation

In this subsection, we will separate Eq. (37) in the form of individual views, and obtain the solution accordingly.

Calculating the partial derivatives of Eq. (34), we have

$$\frac{\partial F}{\partial R^l} = -B^l + R^l A^l + \Lambda^l R^l = 0, \quad (39)$$

$$\frac{\partial F}{\partial R^r} = -B^r + R^r A^r + \Lambda^r R^r = 0. \quad (40)$$

Obviously, Eq. (39) and Eq. (40) are Sylvester's equation and are pose estimation from *left* and *right* cameras separately.

We can prove that (see Appendix B), if $\hat{R}^l$ is a solution for Eq. (39), the constructed $\tilde{R}^r = R^o \hat{R}^l R^{oT} + H$ is a solution for Eq. (40). On the other hand, if $\hat{R}^r$ is a solution for Eq. (40), then $\hat{R}_{3 \times 3}^r$ can be obtained from $\hat{R}^r$. We can also prove that $(R_{3 \times 2}^o)^T \hat{R}_{3 \times 3}^r R_{3 \times 2}^o$ is a solution for Eq. (39), where $R_{3 \times 2}^o$ is the first two columns of $R_{3 \times 3}^o$. Hence both $\hat{R}^l$ and $(R_{3 \times 2}^o)^T \hat{R}_{3 \times 3}^r R_{3 \times 2}^o$ are solutions for Eq. (37). It explains that pose estimation from multiple independent views (i.e.

the centralized solution) can be acquired with separate pose estimation from each individual camera (i.e. the Sylvester's equation).

As a result, we can use the estimates of other views to compute the pose with regard to the first camera, even to a virtual camera. The problem of pose estimation from multiple views can be solved with the independent pose estimate from each individual view. The equation obtained for one camera is analogous to that of other cameras. Furthermore, our algorithm can be directly extended to the approaches with N cameras; i.e. set a reference camera view α , and project every other camera n , ($n = 1, \dots, N, n \neq \alpha$) to the selected view α with the corresponding rotation $R^{o(n\alpha)}$.

Note that we can select any view (corresponding to a real or virtual camera) as a reference camera. The concept of a reference camera imposes no limitations and is simply introduced in order to select a camera coordinate system for pose estimation. Thus, the notion of a reference camera is simply used to provide a canonical camera coordinate system for the solution of the pose estimation problem. Furthermore, the pose estimate obtained in the reference camera can then be transformed into the pose in any other view (corresponding to a real or virtual camera).

According to the method of projections onto convex sets (POCS) (43), we develop a distributed solution for Eq. (37) as well.

Algorithm 2 *Distributed Solution for N Cameras*

given a constant $C \in (1, 2)$,
compute R^α from the Sylvester's equation of the reference camera view α .
repeat
 1. Calculate R^n for each other camera view n with the computed R^α
 and Eq. (27).
 2. Put R^n into the Sylvester's equation of camera view n , and compute
 the residual.
 3. Average the residuals for all $n, (n \neq \alpha)$ in Step. 2, and divide by C .
 4. Put the scaled mean residual in Step. 3 back into the Sylvester's
 equation of the reference camera view α , and compute R^α .
until R^α does not change much or the preset iteration number is
 achieved.

The distributed solution is actually using independent pose estimation from each camera. For the two-camera case, if there is no noise, one exact solution is found for both Eq. (39) and Eq. (40); if there is noise, such a solution might not exist. Accordingly, the main procedure in Algorithm 2 recursively updates Eq. (39) with the residual from Eq. (40) (with the previous iteration results) to limit residuals in both equations. For the N -camera case, it repeats the process by using the reference camera to update all of the other cameras, obtain the average residual from them (i.e. the mean residual could be regarded as from one camera), and then update the reference camera view based on this mean residual. In fact, we can also use the weighted average residuals in Step. 3, and the weights can

be set according to the evaluation of the estimates from each camera, i.e. Fisher's Information Matrix (44). The constant is used to adjust the convergence speed, and $C = 1.3$ in this paper.

What is more, it is unnecessary for all cameras to capture the same feature points. In the centralized and distributed solutions, all of the observations (with or without noise, same or different feature points) from each camera view are combined together, which extends the field of view (FOV) of specific cameras, to generate a better estimate by using all these information at the same time. Note that all of our discussion is about the pose update between sequential frames, however, the final pose state can be similarly acquired.

4.6. Pose Estimation with Three Cameras

To better explain our proposed centralized and distributed algorithms, a three-camera case is used as an example. For simplicity, we assume that all of the matrix Λ have been respectively evaluated with the method in Section 3.4, just like the single camera case, and regard that they are known in the following.

4.6.1. Centralized Pose Estimation

In comparison to Eq. (37), the general Sylvester's equation with three cameras becomes

$$\begin{aligned}
& -B^1 + R^1 A^1 + \Lambda^1 R^1 \\
& + R^{12T} [-B^2 + (R^{12} R^1 R^{12T} + H^{12}) A^2 + \Lambda^2 (R^{12} R^1 R^{12T} + H^{12})] R^{12} \\
& + R^{13T} [-B^3 + (R^{13} R^1 R^{13T} + H^{13}) A^3 + \Lambda^3 (R^{13} R^1 R^{13T} + H^{13})] R^{13} \\
& = 0 .
\end{aligned} \tag{41}$$

And the solution is

$$\begin{aligned}
vec(R^1) = & [A^{1T} \otimes I + I \otimes \Lambda^1 + (R^{12T} A^2 R^{12})^T \otimes (R^{12T} R^{12}) \\
& + (R^{12T} R^{12})^T \otimes (R^{12T} \Lambda^2 R^{12}) + (R^{13T} A^3 R^{13})^T \otimes (R^{13T} R^{13}) \\
& + (R^{13T} R^{13})^T \otimes (R^{13T} \Lambda^3 R^{13})]^{-1} \\
& vec(B^1 + R^{12T} B^2 R^{12} - R^{12T} H^{12} A^2 R^{12} - R^{12T} \Lambda^2 H^{12} R^{12} \\
& + R^{13T} B^3 R^{13} - R^{13T} H^{13} A^3 R^{13} - R^{13T} \Lambda^3 H^{13} R^{13}) , \quad (42)
\end{aligned}$$

where H^{12} and H^{13} can be computed respectively with Eq. (35) according to Algorithm 1.

4.6.2. Distributed Pose Estimation

With the distributed pose estimation, we obtain

$$\frac{\partial F}{\partial R^1} = -B^1 + R^1 A^1 + \Lambda^1 R^1 = 0 , \quad (43)$$

$$\frac{\partial F}{\partial R^2} = -B^2 + R^2 A^2 + \Lambda^2 R^2 = 0 , \quad (44)$$

$$\frac{\partial F}{\partial R^3} = -B^3 + R^3 A^3 + \Lambda^3 R^3 = 0 . \quad (45)$$

With Algorithm 2, if $\hat{R}^1$ a solution to Eq. (43), $\hat{R}^2 = R^{12} \hat{R}^1 R^{12T} + H^{12}$ and $\hat{R}^3 = R^{13} \hat{R}^1 R^{13T} + H^{13}$ are fed into Eq. (44) and Eq. (45) to obtain

$$-B^2 + \hat{R}^2 A^2 + \Lambda^2 \hat{R}^2 = \hat{E}^2 , \quad (46)$$

$$-B^3 + \hat{R}^3 A^3 + \Lambda^3 \hat{R}^3 = \hat{E}^3 . \quad (47)$$

Then an iterative $\hat{R}_i^1$ is calculated as a solution to the following equation:

$$-B^1 + \hat{R}_i^1 A^1 + \Lambda^1 \hat{R}_i^1 = (\hat{E}^2 + \hat{E}^3)/(2C) , \quad (48)$$

where C is a constant, and here is $C = 1.3$.

5. Experimental Results

With two or more cameras, we have more data about the object pose; consequently the estimation results should be more noise robust. Even when one camera is highly noisy, we can still use the observations from other cameras to improve this camera’s results. However, more cameras generally mean more computation load. As shown in Section 4, we can decouple the computation from multiple cameras into that from the individual cameras; accordingly the increase in computation load is linear in our system.

5.1. Synthetic Points with Ground Truth

We firstly demonstrate the noise robustness of our algorithm with synthetic points, and the results are shown in Fig. 1. Here we use 20 points and run 5000 times.

In this test, some synthetic points are generated in a 3D space, and a random rotation and translation is applied to the collection of 3D points. Subsequently, the original as well as the rotated and translated 3D points are projected into virtual camera views. The proposed system relies on the projected 2D points for pose estimation. Robustness of the system is assessed by directly applying additive noise to the projected 2D points.

Since the evaluation of the Lagrange Multiplier Matrix Λ (both Λ^l and Λ^r) is crucial, we firstly evaluate Λ by perturbing the matrix with noisy data. We use SNR to measure the error introduced in the matrix Λ , and Λ is finally evaluated by calculating the error of the rotation angles computed from the perturbed matrix. We fix the SNR for the input points as 10dB for this experiment. As shown in Fig. 1(a), the result is promising. For instance, the error in rotation angles is about

(a) under different Lagrange Multiplier evaluations (b) from two views under different input SNR

(c) in the reference view under different input SNR (d) with increasing number of camera views

(e) for different iteration times (f) between centralized and distributed solutions

Figure 1: Pose estimation results of the synthetic data.

5 degrees when SNR= 5dB for Λ . In Fig. 1(b), pose estimates from two views, $R^o = -20$ degrees and $R^o = 30$ degrees, are projected to the reference coordinate system, and no obvious difference is found. Specifically, no optimal viewpoints

are found to give better results in this experiment. With real videos, our system can output better estimates when feature points are more widely distributed within the tracked object area in images.

The results, under different noise levels for the input feature points, are shown in Fig. 1(c). The Least-Squares Method is a method without considering the orthogonality constraints of the rotation matrix, and it deviates greatly as a high noise level presents. However, it is known that the robustness of pose estimation algorithms can be improved by adding these constraints. The centralized solution with various number of cameras is tested as shown in Fig. 1(d), and the distributed solution produces a very similar curve, which has been omitted. We fix $\text{SNR} = 1\text{dB}$ for the feature points in all possible views, and observe the decreasing errors with increasing camera numbers.

The convergence of Algorithm 1 is evaluated in Fig. 1(e), which shows that the error is small even for one iteration, and the results improve less after about 9 iterations. Accordingly, our preset iteration number is 15. The centralized solution and distributed solution are compared in Fig. 1(f). The distributed solution performs better than the centralized solution in the condition of $\text{SNR} > 11\text{dB}$, but it diverges when high noise levels presented. It can be explained as the solution sets for each Sylvester's equation are too far from each other in the heavily noisy cases, and it is not easy to find a balance point between them.

We also test the centralized solution by adding various noise levels to different cameras. In Table. (1), the first three columns are root mean square errors (RMSE) from three cameras separately, and the fourth column is the centralized solution for Camera 3 with the information from all three views. Here the centralized solution is always of less error than any specific camera, because it applies all

observations from all cameras at the same time.

Table 1: Different View with Different Errors

Errors	Cam 1	Cam 2	Cam 3	CS for Cam 3
(degree)	6.8535	7.0722	6.9862	1.3137
(degree)	2.0309	1.8263	7.0935	1.0849
(degree)	11.4608	12.1393	7.2131	3.9289
(degree)	11.9144	1.8148	6.9502	1.5320

5.2. Video Sequences from One Camera

5.2.1. Large Angle Rotation

We also test our algorithm with real video sequences. The *Girl* sequence has 318 frames with a resolution of 128×96 pixels and a frame rate of 25 frames per second (fps). In this video sequence, a girl turns to the right, left, up, and down, respectively, and almost 90 degrees in each direction. This video has a clear blue background and is uniformly illuminated, except for a few frames. This video sequence is used to demonstrate the robustness of Sylvester’s equation algorithm (SEA) for pose estimation of large angles. We attempt to track the entire head of the girl including her hair, since the hair features can be used to estimate the pose. This is particularly helpful when the girl lowers her head, and the facial region in the image becomes too small to contain any distinctive features. Most of the features used in this situation are extracted from the hair.

For comparison, we implemented a method for pose estimation based on Template Matching (TM) (45). We have also used the Pointing Database (46) to construct the head pose model. However, our implementation of the TM method is different from the original algorithm presented in (45). Specifically, in order

to make the head model robust to background clutter, the original algorithm has learned a face skin model from the training data, and we have not provided this model in our implementation of the TM method. For simplicity, we have implemented and tested the TM method on real videos corresponding to tilt and pan independently. We also completed an approach to pose estimation using the essential matrix (EM) method (47).

The ground truth of the pose for this sequence was determined by an average of pose coordinates from experimental and measurement data. Specifically, an experiment in which three people independently circled the object according to the templates provided in Pointing04 (48) and PIE (49) was conducted. Additionally, measurements of the length change in images were performed. The final ground truth was ascertained by computing the mean of the four pose coordinates corresponding to the three independent experiments and measurement results.

In Fig. 2, the girl moves her head up and down in the vertical direction, and the curve in the middle is the false estimate in the horizontal direction with the proposed SEA. Also as shown, this false estimation has been mostly depressed. It is important to note that in our implementation of the TM approach (45), only the templates in the vertical direction are employed. Had we used templates in both vertical and horizontal directions, the pose estimate would have degraded, while the computation of the TM algorithm would have increased substantially because of more templates to be compared.

As shown in Fig. 3(b), the results from the TM method (45) are not stable and always flipping, while SEA results are smoother. In addition, TM algorithm (45) is sensitive to the tracking steps, because the template matching is applied based on tracking. In comparison, the output of SEA is stable, and not so sensitive to

the tracking, because some incorrectly matched feature pairs have been discarded before calculating the pose.

Also as can be seen from Fig. 3(a), the pose in Frame 156 is approximately 30 degrees. However, the EM method provides an estimate of over 50 degrees. In the remainder of the sequence, the girl’s head moves increasingly higher. Yet, the estimated angles provided by EM are increasingly smaller. The EM algorithm assumes that the camera has been calibrated and all of the necessary parameters are assumed to be known. Therefore, the focal length f is replaced with 1 in Eq. (2). In practice, these assumptions violate and consequently the EM estimate is inaccurate.

Figure 2: Pose estimation for the video of *Girl Nodding*: the red curve is the true pose; the green curve is the result from the Template Matching (TM) method (45); and the blue curve is obtained based on our proposed Sylvester’s Equation (SE) algorithm.

Index	156	157	158	159
TRUE (degree)	(-28, 0)	(-33, 0)	(-38, 0)	(-42, 0)
EM (degree)	(-52, -5)	(-50, -5)	(-45, -6)	(-43, -7)
TM (degree)	(-32, 0)	(-20, 0)	(-30, 0)	(-43, 0)
SEA (degree)	(-28, 7)	(-34, 6)	(-40, 5)	(-43, 4)

Figure 3: Pose estimation results of the video *Girl Nodding*: (a) the Essential Matrix (EM) method (47); (b) the Template Matching (TM) method (45); (c) the proposed Sylvester's Equation Algorithm (SEA). The horizontal and vertical bars at the edge of the images are used to represent the estimates of the pan and tilt, and these estimates are also provided in the table.

5.2.2. Translation and Depth Change

SEA is then tested when there are translation and depth change. The *Car* sequence has 230 frames with a resolution of 320×240 pixels and a frame rate

Figure 4: Pose estimation results of a moving car.

of 30 fps. The size of the car in the video is quite small, and environments are a little dark. It is even hard for human eyes to recognize the car and its pose at first sight, but it has no effects on SEA for it is based on SIFT features, which are robust to illumination conditions. In the video, a car slows down at the crossing, waiting for the signals, and then turns its direction. This test sequence observes complicated position and pose change, including pure translation, translation with turning, depth change and occlusion.

In Fig. 4, the first two pictures indicate that the translation does not influence SEA, as before calculating the pose, all of features are shifted by subtracting their mean. Moreover, there are other cars behind our target, but the tracker is not attracted by them due to the use of the color module and gradient module in particle filtering. In the median frame, the car is partly occluded by a pole, but the results after removing the incorrectly matched features in our system are robust to the occlusion. Furthermore, the car is moving and turning, and SEA can also distinguish translation and rotation properly. There is depth change with the last two images, and depth change could cause problems to our system. We regard $z_t^j / z_{t+1}^j = 1$ in Section 3.1 on the assumption that the depth change is small and motion in this direction is slow. In this experiment, this assumption is valid because the speed

of vehicles is usually slow at crossings. However, if the car moves fast to or away from the camera, z_t^j / z_{t+1}^j should be adjusted according to the size change of the object. As the minor depth changes are reflected by very slight scale modification in the projected video sequence, and horizontal and vertical movements, on the other hand, translate into well-defined spatial changes in the captured image sequence, the object in one of two successive frames (t^{th} or $t + 1^{th}$) should be properly enlarged to eliminate the impact of depth change.

5.3. Video Sequences from Multiple Cameras

5.3.1. Two Cameras

The *Head* sequence from each view has 2570 frames with a resolution of 320×240 pixels and a frame rate of 30 fps. A man randomly rotates his head, including some large angles and combined angles, with changing speed. Two cameras are placed at the same distance from the man's head, but 45 degrees apart.

We firstly estimate pose from the *left* and *right* camera separately, and the results are shown in Fig. 5. Then we transform the pose from the *left* camera into the *right* camera coordinate system, and the results are given in the third line of the table of Fig. 5. It can be seen that the Sylvester's equation performs consistently with different video sequences during long time computation, and the projected results are close to the estimates directly from the *right* camera. On the other hand, it indicates the pose estimates are the same after proper transform because the object just acts one pose at one time.

5.3.2. Three Cameras

In addition, the system is tested with three cameras. The *Bottle* sequence from each view has 1011 frames with a resolution of 320×240 pixels and a frame

Figure 5: Pose estimation results of the video *Head*: The first row is from the *left* camera and the second row is from the *right* camera.

rate of 30 fps. This sequence includes pure translation, translation with rotation and depth change. As shown in Fig. 6, it is hard to notice the rotations of the bottle without the bottle label. Throughout this experiment, Camera 3 does not directly face the label, and this camera also has less pixel numbers than the other two. However, the centralized solution (CS) can provide much better results for Camera 3 with the support from other cameras. Also as can be seen from the first two lines in the table of Fig. 6, the Sylvester's equation solutions are very close to the ground truth after the corresponding transform. Thus, if the feature points can be well extracted from images, the Sylvester's equation will provide accurate pose estimate.

It is also tested that cameras are not in the same plane. The *Chair* sequence from each view has 1450 frames and a frame rate of 30 fps. The transform is $R^o = R^z \times R^y = 5 \times 45$ degrees between Camera 2 and Camera 1 and $R^o =$

Index	255	497	918
Cam 1 (degree)	(-8, 0)	(22, -1)	(31, -3)
Cam 2 (degree)	(-51, 0)	(-25, 0)	(-13, 1)
Cam 3 (degree)	(-109, 2)	(-93, 5)	(-92, 8)
CS for Cam 3 (degree)	(-143, 0)	(-113, 0)	(-102, 1)
TRUE of Cam 3 (degree)	(-142, 0)	(-112, 0)	(-101, 0)

Figure 6: Pose estimation results of the video *Bottle*: The three rows are from three cameras separately. The transform is $R^o = 45$ degrees between Camera 1 and Camera 2, and $R^o = 90$ degrees between Camera 2 and Camera 3.

$R^x \times R^y = 10 \times 30$ degrees between Camera 2 and Camera 3. The resolution for Camera 1 and Camera 3 is 160×120 pixels, and it is 640×480 pixels for Camera 2.

As shown in Fig. 7, for the first and third row, they are directly from Sylvester's equation solutions of Camera 1 and Camera 3 respectively, but for the second row, we do not actually process images for Camera 2, but firstly calculate the distributed solution with Camera 1 and Camera 3, and then transform it into the

coordinate system of Camera 2. Since the information in Camera 2 is never used, it acts as a virtual camera. This test reveals that pose estimation with any view can be obtained from available observations of other views.

It is worthy noting that there is more background in our tracked area (the white ellipse), and the Sylvester’s equation algorithm is also robust to this kind of outliers. The computation time is compared in Table. (2), where the numbers indicate the size of images. After we have all features from each video sequence, the centralized and distributed solution are obtained with Algorithm 1 and 2. As it just involves the numerical computation, both algorithms are very fast.

Figure 7: Pose estimation results of the video *Chair*: The first row is from Sylvester’s equation results of Cam 1, the third row is from Sylvester’s equation results of Cam 3, and the second row is the distributed solution of Cam 1 and Cam 3 for Cam 2.

5.4. Quantitative Performance Comparison

There are three different likelihood densities which must be estimated in SEA with the particle filtering method: p_{color} , $p_{gradient}$ and $p_{feature}$, where p_{color} and p_{edge} are especially for tracking, and $p_{feature}$ is for pose estimation. We compare

Table 2: Average Computation Time Comparison for Each Frame from Different Camera Views (with Matlab)

Sylvester's Equation from Camera 1 (160×120)	SIFT for two frames from Camera 1 (160×120)	Total Time for Camera 1 (160×120)	Total Time for Camera 2 (640×480)
0.0142 s	0.1353 s	2.4414 s	0.1791 s

the average computation time of the different likelihood parameters in Table. (3). As we can see, compared with the most time-consuming components which relate to the tracking results, the computation required for pose estimation is small. It is because the feature likelihood model for pose estimation only involves some efficient numerical calculations. Moreover, in our simulations we impose a pre-set limit on the number of iterations (usually less 15 iterations), which results in negligible overhead for the computation time of the iterative algorithms.

Table 3: Average Computation Time Comparison of Different Parts in the system (30 particles)

Total time for each frame	Particle Filtering for each frame	p_{color} for each particle	$p_{gradient}$ for each particle	$p_{feature}$ for each particle
2.4414 s	1.898 s	0.0236 s	0.0335 s	0.0142 s

The proposed SEA greatly depends on the SIFT feature points, so large sized objects in the images with full illumination are preferred. But if the size is large, the number of particles should usually be increased accordingly. Generally speaking, it would be enough that SIFT can extract more than 10 feature pairs within the object region in the images. On the other hand, SIFT algorithm is insensitive to illumination changing in contrast with the template matching method (45). The

initialization of Lagrange multipliers Λ in our system is important as well. In fact, the efficient and accurate estimation of the Lagrange multipliers is still a topic of our future research.

6. Conclusions

In this paper, we present a new approach to pose estimation from 2D image sequences from multiple cameras. Our approach provides a direct estimate of the 3D rotation parameters from 2D image sequences without constructing a 3D model or requiring system training and learning prior to pose estimation. Scale-Invariant Feature Transform (SIFT) points of corresponding features in adjacent images in the video sequence for each camera view are extracted. A centralized solution to pose estimation from multiple cameras is presented based on a generalized Sylvester's equation. We subsequently derive a distributed solution to pose estimation from multiple cameras that relies on independent estimates of the pose obtained for each camera based on a solution of Sylvester's equation. Specifically, collaboration among the multiple cameras is used to generate a linear combination of the independent estimates from each camera. We demonstrate that the proposed approach provides an efficient and robust method for pose estimation from multiple cameras. Moreover, we show that the centralized and distributed solution to pose estimation from multiple cameras are superior to the estimate obtained from any specific camera view. This approach can therefore be used to improve pose estimation from any single view or estimate the pose from a virtual view.

Our system does not 'know' the pose state, but estimate the pose by detecting the state change between frames. There will be accumulation errors with this mechanism. Particle filtering can avoid some large shift in this process, but cannot

recover the system from a false state. On the other hand, our system must begin with a known pose, although it can be arbitrary. In the future, we will incorporate the learning-based template matching approach into our system, so that it can recognize the current pose. For example, we can apply templates in every 50 frames. It does not increase the computation load particularly, but keep particles from deviating the true state. Moreover, our algorithm will not need manual initialization if we have learned templates.

A. Appendix 1: Error Analysis

A limitation in the existing solutions for pose estimation problems is that the error analysis is seldom conducted, and here we will provide the error analysis by applying the perturbation theory about Sylvester's equation into our system.

Assume that the translation vector has been compensated by Eq. (13), and the error of the translation vector has been cumulated into the noises of points. Then, Eq. (7) becomes

$$\min_{R_1, R_2} [(U_{t+1} + N_{U_{t+1}}) - R_1(P_t + N_{P_t})][(U_{t+1} + N_{U_{t+1}}) - R_1(P_t + N_{P_t})]^T + [(V_{t+1} + N_{V_{t+1}}) - R_2(P_t + N_{P_t})][(V_{t+1} + N_{V_{t+1}}) - R_2(P_t + N_{P_t})]^T, \quad (49)$$

where N_{P_t} , $N_{U_{t+1}}$ and $N_{V_{t+1}}$ represent the noise associated with P_t , U_{t+1} and V_{t+1} , respectively.

We follow the same derivation as above to obtain

$$\begin{aligned} A' + \Delta A &= (P_t + N_{P_t})(P_t + N_{P_t})^T \\ &= P_t P_t^T + (N_{P_t} P_t^T + P_t N_{P_t}^T + N_{P_t} N_{P_t}^T), \end{aligned} \quad (50)$$

$$\begin{aligned} B' + \Delta B &= (P_{t+1} + N_{P_{t+1}})(P_t + N_{P_t})^T \\ &= P_{t+1} P_t^T + (N_{P_{t+1}} P_t^T + P_{t+1} N_{P_t}^T + N_{P_{t+1}} N_{P_t}^T), \end{aligned} \quad (51)$$

where $A' = P_t P_t^T$ and $B' = P_{t+1} P_t^T$. P_t and P_{t+1} are consistent with our earlier definition. Moreover, an offset $\Delta\Lambda$ is assumed to represent the error in calculating the Lagrange multipliers. We consider the perturbed Sylvester's equation (50) given by

$$\frac{\|\Delta R_{2 \times 2}\|_F}{\|R_{2 \times 2}\|_F} \leq \sqrt{3}\epsilon\Phi, \quad (52)$$

where

$$\begin{aligned} \Phi &= \|P^{-1}\|_2 \frac{(\|\Lambda\|_F + \|A'\|_F)\|R_{2 \times 2}\|_F + \|B'\|_F}{\|R_{2 \times 2}\|_F}, \\ \epsilon &= \max\left\{\frac{\|\Delta\Lambda\|_F}{\|\Lambda\|_F}, \frac{\|\Delta A\|_F}{\|A'\|_F}, \frac{\|\Delta B\|_F}{\|B'\|_F}\right\} \end{aligned} \quad (53)$$

and $P = I \otimes \Lambda + A' \otimes I$. We use $\|X\|_F$ and $\|Y\|_2$ to denote the Frobenius and Euclidean norms given by $\|X\|_F = (\sum_{ij} |x_{ij}|^2)^{1/2}$ and $\|Y\|_2 = (Y^T Y)^{1/2}$, respectively.

B. Appendix 2: Proof of the Solution for Distributed Pose Estimation

B.1. Distributed 3D Pose Estimation

In the following proof, we assume that all of the translation parameters have been properly compensated.

We can get the same conclusion as Eq. (16) in 3D space (3). Denote $A_{3 \times 3}^l = Q_t^l Q_t^{lT}$, $B_{3 \times 3}^l = Q_{t+1}^l Q_t^{lT}$, $A_{3 \times 3}^r = Q_t^r Q_t^{rT}$ and $B_{3 \times 3}^r = Q_{t+1}^r Q_t^{rT}$, where Q represent the 3D points. Similar to Eq. (39) and Eq. (40), we have

$$-B_{3 \times 3}^l + R_{3 \times 3}^l A_{3 \times 3}^l + \Lambda_{3 \times 3}^l R_{3 \times 3}^l = 0, \quad (54)$$

$$-B_{3 \times 3}^r + R_{3 \times 3}^r A_{3 \times 3}^r + \Lambda_{3 \times 3}^r R_{3 \times 3}^r = 0, \quad (55)$$

where $\Lambda_{3 \times 3}^l = \begin{pmatrix} \lambda_{11} & \lambda_{14} & \lambda_{15} \\ \lambda_{14} & \lambda_{12} & \lambda_{16} \\ \lambda_{15} & \lambda_{16} & \lambda_{13} \end{pmatrix}$ and $\Lambda_{3 \times 3}^r = \begin{pmatrix} \lambda_{21} & \lambda_{24} & \lambda_{25} \\ \lambda_{24} & \lambda_{22} & \lambda_{26} \\ \lambda_{25} & \lambda_{26} & \lambda_{23} \end{pmatrix}$.

According to Eq. (25), we can get

$$B_{3 \times 3}^r = Q_{t+1}^r Q_t^{rT} = (R_{3 \times 3}^o Q_{t+1}^l)(R_{3 \times 3}^o Q_t^l)^T = R_{3 \times 3}^o B_{3 \times 3}^l R_{3 \times 3}^{oT}. \quad (56)$$

Similarly, we also have $A_{3 \times 3}^r = R_{3 \times 3}^o A_{3 \times 3}^l R_{3 \times 3}^{oT}$ and $\Lambda_{3 \times 3}^r = R_{3 \times 3}^o \Lambda_{3 \times 3}^l R_{3 \times 3}^{oT}$, and all of them result from the transform of coordinate system. Then with Eq. (27), if $\hat{R}_{3 \times 3}^l$ is a solution for Eq. (54), then the constructed $\tilde{R}_{3 \times 3}^r = R_{3 \times 3}^o \hat{R}_{3 \times 3}^l (R_{3 \times 3}^o)^T$ is a solution for Eq. (55), as the left side of Eq. (55) becomes

$$-B_{3 \times 3}^r + \tilde{R}_{3 \times 3}^r A_{3 \times 3}^r + \Lambda_{3 \times 3}^r \tilde{R}_{3 \times 3}^r = R_{3 \times 3}^o [-B_{3 \times 3}^l + \hat{R}_{3 \times 3}^l A_{3 \times 3}^l + \Lambda_{3 \times 3}^l \hat{R}_{3 \times 3}^l] R_{3 \times 3}^{oT} = 0. \quad (57)$$

Obviously, with Eq. (28), if $\hat{R}_{3 \times 3}^r$ is a solution for Eq. (55), the constructed $\tilde{R}_{3 \times 3}^l = (R_{3 \times 3}^o)^T \hat{R}_{3 \times 3}^r R_{3 \times 3}^o$ is also a solution for Eq. (54).

B.2. Distributed 2D Pose Estimation

We will prove the 2D case with the conclusion from the 3D case by showing that Eq. (54) and Eq. (55) can be written into Eq. (39) and Eq. (40).

According to our assumption in Eq. (2) and Eq. (3), the 2D points are from scaled 3D points, and here we will have

$$Q^l = \begin{pmatrix} P^l \\ f^l \end{pmatrix}, \quad Q^r = \begin{pmatrix} P^r \\ f^r \end{pmatrix},$$

$$\begin{pmatrix} P^r \\ f^r \end{pmatrix} = R_{3 \times 3}^o \begin{pmatrix} P^l \\ f^l \end{pmatrix}, \quad \begin{pmatrix} P'_{t+1} \\ f'_{t+1} \end{pmatrix} = R_{3 \times 3} \begin{pmatrix} P'_t \\ f'_t \end{pmatrix}, \quad (58)$$

where Q is the 3D points and P' is the 2D points. According to Eq. (54) and Eq. (55), we obtain

$$\begin{aligned}
& - \begin{pmatrix} P_{t+1}^l P_t^{lT} & M(f_t^l \overline{P}_{t+1}^l) \\ M(f_{t+1}^l \overline{P}_t^{lT}) & M(f_{t+1}^l f_t^l) \end{pmatrix} \\
& + \begin{pmatrix} R^l & r_{13}^l \\ r_{31}^l & r_{32}^l & r_{33}^l \end{pmatrix} \begin{pmatrix} P_t^l P_t^{lT} & M(f_t^l \overline{P}_t^l) \\ M(f_t^l \overline{P}_t^{lT}) & M(f_t^l f_t^l) \end{pmatrix} \\
& + \begin{pmatrix} \Lambda^l & \lambda_{13}^l \\ \lambda_{31}^l & \lambda_{32}^l & \lambda_{33}^l \end{pmatrix} \begin{pmatrix} R^l & r_{13}^l \\ r_{31}^l & r_{32}^l & r_{33}^l \end{pmatrix} = 0, \quad (59)
\end{aligned}$$

$$\begin{aligned}
& - \begin{pmatrix} P_{t+1}^r P_t^{rT} & M(f_t^r \overline{P}_{t+1}^r) \\ M(f_{t+1}^r \overline{P}_t^{rT}) & M(f_{t+1}^r f_t^r) \end{pmatrix} \\
& + \begin{pmatrix} R^r & r_{13}^r \\ r_{31}^r & r_{32}^r & r_{33}^r \end{pmatrix} \begin{pmatrix} P_t^r P_t^{rT} & M(f_t^r \overline{P}_t^r) \\ M(f_t^r \overline{P}_t^{rT}) & M(f_t^r f_t^r) \end{pmatrix} \\
& + \begin{pmatrix} \Lambda^r & \lambda_{13}^r \\ \lambda_{31}^r & \lambda_{32}^r & \lambda_{33}^r \end{pmatrix} \begin{pmatrix} R^r & r_{13}^r \\ r_{31}^r & r_{32}^r & r_{33}^r \end{pmatrix} = 0, \quad (60)
\end{aligned}$$

where $\overline{P}_{t+1}^r$ and $\overline{P}_t^r$ are 2×1 vectors, similar to Eq. (12), and M is the number of feature points. Since Eq. (59) and Eq. (60) are identical, we will omit the superscript in the following.

By only considering the top 2×2 elements of the above equations, we have

$$-P'_{t+1}P_t'^T + \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} P_t'P_t'^T \\ M(f_t\overline{P}_t'^T) \end{pmatrix} + \begin{pmatrix} \Lambda & \lambda_{13} \\ & \lambda_{23} \end{pmatrix} \begin{pmatrix} R \\ r_{31} & r_{32} \end{pmatrix} = 0, \quad (61)$$

where

$$\begin{aligned} & -P'_{t+1}P_t'^T + \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} P_t'P_t'^T \\ M(f_t\overline{P}_t'^T) \end{pmatrix} \\ &= -(P_{t+1} + \overline{P}'_{t+1})(P_t + \overline{P}'_t)^T + \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} (P_t + \overline{P}'_t)(P_t + \overline{P}'_t)^T \\ M(f_t\overline{P}'_t^T) \end{pmatrix} \\ &= -[P_{t+1}P_t^T + M(\overline{P}'_{t+1}\overline{P}'_t^T)] + \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} P_tP_t^T + M(\overline{P}'_t\overline{P}'_t^T) \\ M(f_t\overline{P}'_t^T) \end{pmatrix}, \quad (62) \end{aligned}$$

where P is defined in Eq. (15), and also note that $\overline{P} = 0$. Furthermore, from Eq. (58), we directly acquire

$$\overline{P}'_{t+1}\overline{P}'_t^T = \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} \overline{P}'_t \\ f_t \end{pmatrix} \overline{P}'_t^T = \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} \overline{P}'_t\overline{P}'_t^T \\ f_t\overline{P}'_t^T \end{pmatrix}. \quad (63)$$

Consequently, we have

$$\begin{aligned} & -P'_{t+1}P_t'^T + \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} P_t'P_t'^T \\ M(f_t\overline{P}_t'^T) \end{pmatrix} \\ &= -P_{t+1}P_t^T + \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} P_tP_t^T \\ 0 \end{pmatrix}. \quad (64) \end{aligned}$$

Also in Eq. (61), we set the Lagrange multipliers λ_{13} and λ_{23} to 0 to obtain

$$\begin{aligned} & -P_{t+1}P_t^T + \begin{pmatrix} R & r_{13} \\ & r_{23} \end{pmatrix} \begin{pmatrix} P_tP_t^T \\ 0 \end{pmatrix} + \begin{pmatrix} \Lambda & 0 \\ & 0 \end{pmatrix} \begin{pmatrix} R \\ r_{31} & r_{32} \end{pmatrix} \\ &= -B + RA + \Lambda R = 0, \quad (65) \end{aligned}$$

which are exactly Eq. (39) and Eq. (40).

References

- [1] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *Proc. International Journal of Computer Vision* 60 (2004) 91–110.
- [2] R. H. Bartels, G. W. Stewart, Solution of the matrix equation $ax + xb = c$, *Communication of the ACM* 15 (9) (1972) 820–826.
- [3] R. M. Haralick, H. Joo, C.-N. Lee, X. Zhuang, V. G. Vaidya, M. B. Kim, Pose estimation from corresponding point data, *IEEE Transactions on Systems, Man and Cybernetics* 19 (6) (1989) 698–700.
- [4] Q. Ji, M. S. Costa, R. M. Haralick, L. G. Shapiro, A robust linear least-squares estimation of camera exterior orientation using multiple geometric features, *Isprs Journal of Photogrammetry and Remote Sensing* 52 (2) (2000) 75–93.
- [5] J. M. Frahm, M. Pollefeys, Ransac for (quasi-)degenerate data (qdegsac), in: *Proceedings of Computer Vision and Pattern Recognition*, Vol. 1, 2006, pp. 453–460.
- [6] K. S. Arun, T. S. Huang, S. D. Rlostein, Least-squares fitting of two 3-d point sets, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 9 (5) (1987) 698–700.
- [7] S. Umeyama, Least-squares estimation of transformation parameters between two point patterns, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13 (4) (1991) 376–380.

- [8] D. Goryn, S. Hein, On the estimation of rigid body rotation from noisy data, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17 (12) (1995) 1119–1120.
- [9] B. K. P. Horn, H. M. Hilden, S. Negahdaripour, Closed-form solution of absolute orientation using orthonormal matrices, *Journal of the Optical Society of America A* 5 (1988) 1127–1135.
- [10] B. K. P. Horn, Closed-form solution of absolute orientation using unit quaternions, *Journal of the Optical Society of America A* 4 (1987) 629–642.
- [11] C. Olsson, F. Kahl, M. Oskarsson, The registration problem revisited: Optimal solutions from points, lines and planes, in: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006, pp. 1206–1213.
- [12] O. D. Faugeras, S. J. Maybank, Motion from point matches: multiplicity of solutions, in: *Proc. Workshop on Geometry and Robotics*, Vol. 391, 1988, pp. 172–193.
- [13] C. F. V. Loan, The ubiquitous kronecker product, *Journal of Computational and Applied Mathematics* 123 (2000) 85–100.
- [14] G. H. Golub, S. Nash, C. V. Loan, A hessenberg-schur method for the problem $ax+xb=c$, *IEEE Transactions on Automatic Control* 24 (6) (1979) 909–913.
- [15] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, R. Szeliski, A comparison and evaluation of multi-view stereo reconstruction algorithms, in: *Proceedings of Computer Vision and Pattern Recognition*, Vol. 1, 2006, pp. 519–526.

- [16] P. M. Q. Aguiar, J. M. F. Moura, Three-dimensional modeling from two-dimensional video, *IEEE Transactions on Image Processing* 10 (10) (2001) 1541–1551.
- [17] S. Ramalingam, S. K. Lodha, P. Sturm, A generic structure-from-motion framework, *Computer Vision and Image Understanding* 103 (3) (2006) 218–228.
- [18] S. Yaguchi, H. Saito, Arbitrary viewpoint video synthesis from multiple uncalibrated cameras, *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS* 34 (1) (2004) 430–439.
- [19] F. Schaffalitzky, A. Zisserman, Multi-view matching for unordered image sets, in: *European Conference on Computer Vision*, Vol. 2, 2002, pp. 414–431.
- [20] P. Sturm, Multi-view geometry for general camera models, in: *IEEE Computer Conference on Computer Vision and Pattern Recognition*, 2005.
- [21] J. Y. Yu, L. McMillan, General linear cameras, in: *European Conference on Computer Vision*, Vol. 2, 2004, pp. 14–27.
- [22] M. Grossberg, S. Nayar, A general imaging model and a method for finding its parameters, in: *International Conference on Computer Vision*, Vol. 2, 2001, pp. 108–115.
- [23] R. Pless, Using many cameras as one, in: *IEEE Computer Conference on Computer Vision and Pattern Recognition*, Vol. 2, 2003, pp. 587–593.

- [24] W. Y. Chang, C. S. Chen, Pose estimation for multiple camera systems, in: Proceedings of the 17th International Conference on Pattern Recognition, Vol. 3, 2004, pp. 262–265.
- [25] F. Kahl, Multiple view geometry and the l-infinity norm, in: Proc. of International Conference on Computer Vision, 2005.
- [26] S. Agarwal, M. Chandrakar, F. Kahl, D. Kriegman, S. Belongie, Practical global optimization for multiview geometry, in: European Conference on Computer Vision, 2006, pp. 592–605.
- [27] C. Rother, S. Carlsson, Linear multi view reconstruction and camera recovery using a reference plane, International journal of computer vision 49 (2-3) (2002) 117–141.
- [28] P. Baker, C. Fermuller, Y. Aloimonos, R. Pless, A spherical eye from multiple cameras (makes better models of the world), in: IEEE Computer Conference on Computer Vision and Pattern Recognition, Vol. 1, 2001, pp. 576–583.
- [29] J. M. Frahm, K. Koser, R. Koch, Pose estimation for multi-camera systems, in: DAGM, 2004.
- [30] Y. Wang, Y. H. Liu, L. M. Tao, G. Y. Xu, Real-time multi-view face detection and pose estimation in video stream, in: Proceedings of the 18th International Conference on Pattern Recognition, Vol. 4, 2006, pp. 354–357.
- [31] Z. Q. Zhang, Y. X. Hu, M. Liu, T. S. Huang, Head pose estimation in seminar room using multi view face detectors, in: First International Evaluation

Workshop on Classification of Events, Activities and Relationships, CLEAR 2006, 2006, pp. 299–304.

- [32] M. Voit, K. Nickel, R. Stiefelhagen, Estimating the lecturer's head pose in seminar scenarios - a multi-view approach, in: Machine Learning for Multimodal Interaction, Second International Workshop, MLMI 2005, 2005, pp. 230–240.
- [33] J. Zhang, S. K. Zhou, L. McMillan, D. Comaniciu, Joint real-time object detection and pose estimation using probabilistic boosting network, in: IEEE Computer Conference on Computer Vision and Pattern Recognition, 2007.
- [34] C. Chen, D. Schonfeld, J. Yang, M. Mohamed, Pose estimation from video sequences based on sylvester's equation, in: SPIE Proceedings of Electronic Imaging: Science and Technology. Conference on Visual Communications and Image Processing (VCIP), 2007.
- [35] C. Chen, D. Schonfeld, M. Mohamed, Robust pose estimation based on sylvester's equation: Single and multiple collaborative cameras, in: IEEE International Conference on Speech and Signal Processing (ICASSP), 2008.
- [36] N. Bouaynaya, D. Schonfeld, Complete system for head tracking using motion-based particle filter and randomly perturbed active contour, in: Proc. SPIE, Image and Video Communications and Processing, Vol. 5685, 2005, pp. 864–873.
- [37] K. Hariharakrishnan, D. Schonfeld, Fast object tracking using adaptive block matching, IEEE Transactions on Multimedia 7 (5) (2005) 853–859.

- [38] M. A. Fischler, R. C. Bolles, Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Communication of the ACM* 24 (6) (1981) 381–395.
- [39] T. Jebara, A. Azarbeyejani, A. Pentland, 3d structure from 2d motion, *IEEE Signal Processing Magazine* 16 (3) (1999) 66–84.
- [40] R. Horaud, F. Dornaika, B. Lamiroy, S. Christy, Object pose: The link between weak perspective, paraperspective, and full perspective, *International Journal of Computer Vision* 22 (2) (1997) 173–189.
- [41] S. O. Ba, J. M. Odobez, A probabilistic framework for joint head tracking and pose estimation, in: *Proc. 17th International Conference Pattern Recognition*, Vol. 4, 2004, pp. 264–267.
- [42] D. C. Sorensen, Y. Zhou, Direct methods for matrix sylvester and lyapunov equations, *Journal of Applied Mathematics* (6) (2003) 277–303.
- [43] D.C.Youla, H. Webb, Image restoration by the method of convex projections: Part 1 theory, *IEEE Transactions on Medical Imaging* 1 (2) (1982) 81–94.
- [44] A. Chiusos, R. Brockett, S. Soatto, Optimal structure from motion: Local ambiguities and global estimates, *International journal of computer vision* 3 (39) (2000) 195–228.
- [45] S. O. Ba, J. M. Odobez, Evaluation of multiple cues head pose tracking algorithm in indoor environments, in: *Proc. International Conference on Multimedia and Expo*, Amsterdam, 2005.

- [46] N. Gournier, D. Hall, J. L. Crowley, Estimating face orientation from robust detection of salient facial features, in: Proc. Pointing 2004, ICPR, International Workshop on Visual Observation of Deictic Gestures, 2004.
- [47] X. Zhuang, T. S. Huang, Two-view motion analysis: a unified algorithm, Journal of the Optical Society of America A: Optics, Image Science, and Vision 3 (9) (1986) 1492–1500.
- [48] Pointing’04 icpr workshop: Head pose image database (2004).
URL <http://www-prima.inrialpes.fr/Pointing04/data-face.html>
- [49] T. Sim, S. Baker, M. Bsat, The cmu pose, illumination, and expression database, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (12) (2003) 1615–1618.
- [50] N. J. Higham, Perturbation theory and backward error for $ax+xb=c$, in: Numerical Analysis Report, no. 211, University of Manchester, England, 1992.