Pose Estimation from Video Sequences Based on Sylvester's Equation

Chong Chen, Dan Schonfeld, Junlan Yang [a] and Magdi Mohamed[b]

[a]Department of Electrical and Computer Engineering, University of Illinois at Chicago;
[b]Physical Realization Research Center of Excellence, Motorola Labs

ABSTRACT

An approach is introduced in this paper to track the object motion and estimate pose jointly within the framework of particle filtering, which can directly estimate the 3D poses from 2D points in the images. The Adaptive Block Matching technique (ABM) is firstly used to improve the computational efficiency of particle filtering. Next, Scale-Invariant Feature Transform (SIFT) is applied to extract feature points. We can show that pose estimation from the corresponding points can be concluded as a Sylvester's Equation, and the solvent of the equation is the pose state. The weight of particle filtering is measured by the color histogram, the edge and the distance between corresponding projected feature points. Finally, we will demonstrate the performance of our algorithm with experimentally results.

Keywords: Pose Estimation, Particle Filtering, Sylvester's Equation, Lagrange Multiplier

1. INTRODUCTION

Pose analysis is currently one of the most active research topics in computer vision, which aims to recover body poses and motion parameters from static images or video sequences. The retrieved data have a wide spectrum of promising applications in many areas such as virtual reality, human-machine interaction, multimedia information coding and transferring.

Many solutions have been proposed by researchers during the past years. Most of them can be classified into two major categories, feature-based methods and appearance-based methods. Braathen et. al. estimate pose state with particle filtering, while firstly a set of images are used to estimate the camera parameters and 3D head geometry. A framework for joint head tracking and pose estimation was realized in,[1] with particle filter as well. They also need training data for modelling with one Gaussian and three Gabor filters. Pose estimation with SVD methods from corresponding points have been well established,[2] but they are utilizing 3D points and the whole theory is based on some orthogonal suppositions, which is not satisfied in the 2D-2D case.

We employ the feature-base approach in this paper, and apply the 2D points from images to directly estimate the 3D poses. It can be shown that pose estimation problem can be induced into a Sylvester's Equation, which can be solved with many methods, such as Kronecker Product and Bartels-Stewart approach. Moreover particle filtering[3] provides a flexible framework, for it can also deal with nonlinear and non-Gaussian situations. These algorithms have been very popular in the video tracking from the CONDENSATION algorithm.[4] We will also apply the particle filtering method to realize object tracking and pose estimation simultaneously, and the color histogram, the edge detection and the distance between projected feature points are measured as the likelihood in this process.

The rest of this paper is organized as follows. Section 2 describes the implement of the particle filtering to realize object tracking, and Section 3 illustrates the method of pose estimation based on Sylvester's Equation. Although these two sections are accomplished simultaneously within the framework of particle filtering in our system, they are bewritten in separate two sections for the emphasis on pose estimation. In Section 4, experimental results on videos show the efficiency of our algorithm. And a conclusion is given in Section 5.

The authors would like to thank the Physical Realization Research Center of Excellence, Motorola Labs, for supporting this work.

2. OBJECT TRACKING WITH PARTICLE FILTERING

2.1. Particle Filtering

Particle filer methods are processes to propagate probability densities for moving object based on Bayesian Rules.

$$p(x_k|Z_{1:k}) = \beta p(z_k|x_k)p(x_k|Z_{1:k-1}) \tag{1}$$

where x is the state, Z is the observed data, β is a normalized constant, and $p(z_k|x_k)$ is the likelihood. In our system, we directly take Nidhal's tracking program,[5] and cite part of her paper in the following. She implements a motion based particle filter with the Importance Sampling Algorithm. The idea is to put more particles in areas where the posterior may have higher density to avoid useless particles that lead to more expensive computation. As a result, Adaptive Block Matching (ABM) algorithm is applied firstly.

For our applications, a head pose estimation system is specified as an example, and a five-parameter super ellipse is used to model the head state, defined as $x = (O_x, O_y, a, b, \theta)$, where (O_x, O_y) are the center, (a, b) are short and long radius, and θ is the rotation angle in the image plane. Particle filtering is to use Monte Carlo method, to recursively estimate the density by a set of random particles.

$$p(x_k|Z_{1:k}) \approx \sum_{i=1}^{N} \omega_k^i \delta(x_k - x_k^i) \tag{2}$$

where $\omega_k^i, i = 1, ..., N$ are associated weights for the particles, and N is their total number.

The state transition density has been modelled as a 1^{st} order random walk dynamic system. The likelihood $w_{total}^{(i)} = p(Z_k|X_k^{(i)}) = w_{color}^{(i)} \times w_{edge}^{(i)} \times w_{feature}^{(i)}$, where $w_{color}^{(i)}$ and $w_{edge}^{(i)}$ denote the likelihood based on color and edge of the i^{th} sample, which are related with the tracking parameters, and $w_{feature}^{(i)}$ is computed with the pose parameters. For each frame, the final output is completed as

$$\hat{x}_k = \sum_{i=1}^{N} \omega_k^i x_k^i \tag{3}$$

2.2. Color Histogram

The color model proposed in[6] is utilized in this paper, which is a simple and well suited system for capturing the multi-modal patterns of object color. Let $q_k = \{q_k^i\}_{i=1,...,n}$ denote the particle histogram and $p_k = \{p_k^i\}_{i=1,...,n}$ the model histogram at time k. A popular measure between two distributions p and q is the Bhattacharyya coefficient

$$\rho[p, q] = \sum_{i=1}^{N} \sqrt{p(u)q(u)}du \tag{4}$$

The larger ρ is the more similar the two distributions are. As a distance between two distributions, we define the measure

$$d = \sqrt{1 - \rho[p, q]} \tag{5}$$

Small distances correspond to large weights

$$w_{color}^{(i)} = \frac{1}{\sqrt{2\pi}\sigma_c} e^{-\frac{d^2}{2\sigma_c^2}} \tag{6}$$

which are specified by a Gaussian with variance σ_c.

The histogram is a simple and well suited system for capturing multi-modal patterns of color of any object. An adaptive color histogram of the model will not loose the object in case of changing lighting conditions and/or appearance of the target.

$$q_k^{(i)} = (1-\alpha)q_{k-1}^{(i)} + \alpha q_E^{(i)} \tag{7}$$

where $q_E^{(i)}$ is the histogram of the estimated state vector and α is the forgetting process factor.

To protect the tracker against updating the model when the object has been partially lost, the model histogram is only updated when the total weight of the estimated state vector is above a specified threshold value.

2.3. Edge Detection

We use the model developed in.[4] The observations are collected on normal lines to the contour. Let $Z_{k,\varphi}$ denote the edge detection observations on line φ at time k; $Z_{k,\varphi} = Z_1, Z_2, \ldots, Z_I$. At most one of the I edges is the true contour point. Let p_0 be the prior probability that none of the I edges is the true contour edge. With the assumption that the clutter is a Poisson process along the line with spatial density γ and the true target measurement is normally distributed with standard deviation σ_z, we obtain the edge likelihood model as follows:

$$p(Z_{k,\varphi}|\lambda_\varphi) \propto \frac{1}{\sqrt{2\pi}\sigma_z p_0 \gamma} exp(-\frac{(min_j(Z_j - \lambda_\varphi))^2}{2\sigma_z^2}) \tag{8}$$

where λ_φ represents the pixel along the normal line φ belonging to the sample ellipse.

By assuming independence between the different L normal lines, we have the following overall likelihood function for sample i

$$w_{edge}^{(i)} = \sum_{\varphi=1}^{L} p(Z_{k,\varphi}|\lambda_\varphi) \tag{9}$$

2.4. Initialization

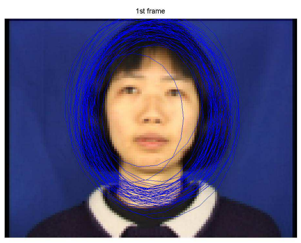
To initialize our program, we must circle the object region in the first image with some points manually, and the system could produce an ellipse with Least-Square method with these points. For example, five to six points are enough to mark the head. We can also indicate the face with higher accuracy carefully by more points, but it is not necessary for our algorithm as shown in our experiments.

After the initialization, our system could produce some particles with the pre-defined pdf, and the picture is shown in Figure 1. To catch up with the fast movement, we need some more separately distributed particles, and the pdf is correspondingly flat, while for the slow moving object, the pdf is of a sharp peak to put more particles around the last position.

3. POSE ESTIMATION BASED ON SYLVESTER'S EQUATION

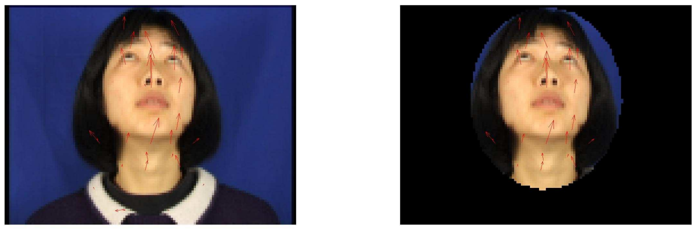
First of all, we should obtain the stable matching pairs between two successive frames. In our system, SIFT is applied, because it transforms the image data into another scale-invariant coordinates relative to locale features, and can provide the location as well as scale and orientation for each feature point, which is consequently highly distinctive and robust across a substantial range of affine distortion, changing in 3D viewpoint, addition of noise and changing in illumination.[7]

Here only the feature points within the object region in each image are considered. But the area circled usually includes the region of the scene background and the neck. What's more, there probably are some errors with the features themselves, which may result from the incorrectly identified features or distortion in the measurement in feature points with SIFT. As a result, we firstly get rid of some false pairs by their slope and length. As can be seen in Figure 2, if an arrow's direction or length is distinctly different from most of the arrows, it indicates an unwanted pair of feature points which should be eliminated. RANSAC[8] could also be introduced here to estimate the transform matrix without taking account of the incorrect points.



(Each particle is represented by a blue ellipse.)

Figure 1. Tracking with Particle Filter



(Feature points extracted from the whole image and from the object region. Arrows indicate the direction and distance of the motion for each feature point from two successive images.)

Figure 2. Feature Points Extracted with SIFT

And after that, the current problem is how to figure out the rotation angles and translation vector from the matching points between two successive images.

For a point in the 3D real object, its positions before and after motion have the relationship as follows.

$$\begin{pmatrix} x_2^j \\ y_2^j \\ z_2^j \end{pmatrix} = R_{3\times 3} \begin{pmatrix} x_1^j \\ y_1^j \\ z_1^j \end{pmatrix} + T \tag{10}$$

where $R_{3\times3}$ is a 3×3 rotation matrix, and T is a 3×1 translation matrix. $j, (j = 1, 2, \ldots)$ represents the j^{th} feature point.

With the perspective projection model, if (u_1^j, v_1^j) and (u_2^j, v_2^j) are the corresponding projected points in the image respectively. We have

$$\frac{x_k^j}{z_k^j} = \frac{u_k^j}{f} \qquad \frac{y_k^j}{z_k^j} = \frac{v_k^j}{f} \qquad k = 1, 2 \tag{11}$$

where f is the focus length.

Replacing the unnecessary parameters of points in the real model from the above two equations, we finally get[9]

$$\begin{pmatrix} u_2^j \\ v_2^j \\ f \end{pmatrix} = \frac{z_1^j}{z_2^j} R_{3\times 3} \begin{pmatrix} u_1^j \\ v_1^j \\ f \end{pmatrix} + T' \tag{12}$$

where $T' = T \cdot f/z_2^j$ is the 3×1 translation matrix within the image plane.

However z_1^j/z_2^j is unknown for each pairs of points in the images. It also indicates that the primary difficulty in completing these systems is that the depth information is lost in the process of projecting 3D object into 2D images. We will take some measures to solve the problem while less affecting the accuracy of our algorithm.

As regards to this problem, we have two methods. The first one is to deem all of the feature points in the same plane; that is all of their Z coordinates are set to the same. It is reasonable to make such an assumption as z_1^j and z_2^j are comparably large and the change in the Z direction is very small between the immediate frames from a video. As a result, $z_1^j/z_2^j = 1$. Adams et. al[10] showed that this simple homography-based method worked fast and accurately when the planar assumption is valid, and accuracy degrades as the excursions from the plane become significant. In our experiments, results under this supposition are generally acceptable.

What's more, similar to,[11] we can also model the 3D face as a ball, and z^j can be estimated as

$$z^j = d + \sqrt{b^2 - (u^j)^2 - (v^j)^2} \tag{13}$$

where b is the long radius of the super-ellipse, and d is estimated distance between the object and the camera, which could be utilized to adjust our final results. The bigger d is, the closer to 1 the factor z_1^j/z_2^j is. This method also has better perform than the former one through our experiments, and d is usually between 100 and 1000 pixel. For a clearer deduction, we assume that this factor has be compensated in the following, so Eq. 12 is simplified as

$$\begin{pmatrix} u_2^j \\ v_2^j \\ f \end{pmatrix} = R_{3\times 3} \begin{pmatrix} u_1^j \\ v_1^j \\ f \end{pmatrix} + T' \tag{14}$$

What's more, if we delete the third line, Eq. 14 becomes

$$\begin{pmatrix} u_2^j \\ v_2^j \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \end{pmatrix} \begin{pmatrix} u_1^j \\ v_1^j \\ f \end{pmatrix} + \begin{pmatrix} t'_x \\ t'_y \end{pmatrix} \tag{15}$$

By rearranging the matrix equation, we get

$$\begin{pmatrix} u_2^j \\ v_2^j \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix} \begin{pmatrix} u_1^j \\ v_1^j \end{pmatrix} + \begin{pmatrix} l_x \\ l_y \end{pmatrix} \tag{16}$$

where $l_x = r_{13} f + t'_x$ and $l_y = r_{23} f + t'_y$.

All of the above is just for one point, and generally more than five matching points within the object region could be obtained with SIFT. So if we take Least-Square Error as the criterion, the 2D-2D pose estimation problem is concluded as

$$\begin{aligned} min \quad & (U_2 - R_1 P_1 - L_x)(U_2 - R_1 P_1 - L_x)^T \\ + \quad & (V_2 - R_2 P_1 - L_y)(V_2 - R_2 P_1 - L_y)^T \end{aligned} \tag{17}$$

subject to

$$\begin{cases} R_1R_1^T + r_{13}^2 = 1 \\ R_2R_2^T + r_{23}^2 = 1 \\ R_1R_2^T + r_{13}r_{23} = 0 \end{cases}$$

where $R_{2\times 2} = \begin{pmatrix} R_1 \\ R_2 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix}$, which is actually the top-left four elements of $R_{3\times 3}$, $P_2 = \begin{pmatrix} U_2 \\ V_2 \end{pmatrix} = \begin{pmatrix} u_2^1 & u_2^2 & \dots \\ v_2^1 & v_2^2 & \dots \end{pmatrix}$, $P_1 = \begin{pmatrix} u_1^1 & u_1^2 & \dots \\ v_1^1 & v_1^2 & \dots \end{pmatrix}$ and the translation vector $L = \begin{pmatrix} L_x \\ L_y \end{pmatrix} = \begin{pmatrix} l_x & l_x & \dots \\ l_y & l_y & \dots \end{pmatrix}$.

We can apply Lagrange method to solve the constrained optimization problem.

$$\begin{aligned} F &= (U_2 - R_1P_1 - L_x)(U_2 - R_1P_1 - L_x)^T \\ &+ (V_2 - R_2P_1 - L_y)(V_2 - R_2P_1 - L_y)^T \\ &+ \lambda_1(R_1R_1^T + r_{13}^2 - 1) + \lambda_2(R_2R_2^T + r_{23}^2 - 1) \\ &+ 2\lambda_3(R_1R_2^T + r_{13}r_{23}) \end{aligned} \tag{18}$$

where λ_1, λ_2 and λ_3 are Lagrange multipliers.

Then the partial derivatives of F are taken, for the rotation coefficients and translation coefficients respectively.

$$\begin{cases} \frac{\partial F}{\partial R_1} = -U_2P_1^T + R_1P_1P_1^T + \lambda_1R_1 + \lambda_3R_2 + L_xP_1^T = 0 \\ \frac{\partial F}{\partial R_2} = -V_2P_1^T + R_2P_1P_1^T + \lambda_2R_2 + \lambda_3R_1 + L_yP_1^T = 0 \end{cases} \tag{19}$$

$$\begin{cases} \frac{\partial F}{\partial L_x} = -U_2 + R_1P_1 + L_x = 0 \\ \frac{\partial F}{\partial L_y} = -V_2 + R_2P_1 + L_y = 0 \end{cases} \tag{20}$$

From Eq. 20, we can obtain

$$L = \begin{pmatrix} L_x \\ L_y \end{pmatrix} = \begin{pmatrix} U_2 - R_1P_1 \\ V_2 - R_2P_1 \end{pmatrix} \tag{21}$$

As can be seen, there is only one translation coefficient for all the feature points in a frame. Therefore, we can take the same method as in,[12] and the translation vector is quickly determined from the following equation once the rotation parameters are known.

$$\begin{pmatrix} l_x \\ l_y \end{pmatrix} = \begin{pmatrix} \overline{u_2} \\ \overline{v_2} \end{pmatrix} - \begin{pmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{pmatrix} \begin{pmatrix} \overline{u_1} \\ \overline{v_1} \end{pmatrix} \tag{22}$$

where $\begin{pmatrix} \overline{u_1} \\ \overline{v_1} \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^M u_1^j/M \\ \sum_{j=1}^M v_1^j/M \end{pmatrix}$ and $\begin{pmatrix} \overline{u_2} \\ \overline{v_2} \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^M u_2^j/M \\ \sum_{j=1}^M v_2^j/M \end{pmatrix}$, the mean of the feature points in the image coordinates from the two sequential images, and M is the number of feature points.

If we define $\overline{P_2} = \begin{pmatrix} \overline{U_2} \\ \overline{V_2} \end{pmatrix} = \begin{pmatrix} \overline{u_2} & \overline{u_2} & \dots \\ \overline{v_2} & \overline{v_2} & \dots \end{pmatrix}$ and $\overline{P_1} = \begin{pmatrix} \overline{U_1} \\ \overline{V_1} \end{pmatrix} = \begin{pmatrix} \overline{u_1} & \overline{u_1} & \dots \\ \overline{v_1} & \overline{v_1} & \dots \end{pmatrix}$, then replace the translation vector in Eq. 23 with these results.

$$\begin{cases} -(U_2 - \overline{U_2})P_1^T + R_1(P_1 - \overline{P_1})P_1^T + \lambda_1R_1 + \lambda_3R_2 = 0 \\ -(V_2 - \overline{V_2})P_1^T + R_2(P_1 - \overline{P_1})P_1^T + \lambda_2R_2 + \lambda_3R_1 = 0 \end{cases} \tag{23}$$

Before further deduction, some vectors are defined here, $P_2' = \begin{pmatrix} U_2' \\ V_2' \end{pmatrix} = \begin{pmatrix} U_2 - \overline{U_2} \\ V_2 - \overline{V_2} \end{pmatrix}$, $P_1' = P_1 - \overline{P_1}$, $A = P_1' P_1^T$ and $B = P_2' P_1^T$.

Eq. 23 can be simplified as

$$\begin{cases} -U_2' P_1^T + R_1 A + \lambda_1 R_1 + \lambda_3 R_2 = 0 \\ -V_2' P_1^T + R_2 A + \lambda_3 R_1 + \lambda_2 R_2 = 0 \end{cases} \tag{24}$$

which can be further combined in one matrix equation.

$$-B + R_{2\times 2} A + \Lambda R_{2\times 2} = 0 \tag{25}$$

where $\Lambda = \begin{pmatrix} \lambda_1 & \lambda_3 \\ \lambda_3 & \lambda_2 \end{pmatrix}$.

Eq. 25 is Sylvester's Equation, which is also called Lyapunov's Equation in the special case with $\Lambda = A^T$.

Once the equation is solved, the angles rotating from the immediate former frame is obtained, and to accumulate all of the past data, the current pose is acquired. There is also the situation that we cannot obtain more than 3 points within the object region by SIFT. Here we just keep and show the results of the last frame.

3.1. Sylvester's Equation and SVD

In this section, we point out that SVD method offers a solution for the Sylvester's Equation in the 3D-3D case; while in the 2D-2D case, the Sylvester's Equation could not be solved by SVD.

As shown in,[12] they also obtain the same equation as Eq.25 in the 3D-3D case.

$$-B + RA + \Lambda R = 0 \tag{26}$$

where $RR^T = I$ and R is a 3×3 matrix, and $\Lambda = \begin{pmatrix} \lambda_1 & \lambda_4 & \lambda_5 \\ \lambda_4 & \lambda_2 & \lambda_6 \\ \lambda_5 & \lambda_6 & \lambda_3 \end{pmatrix}$. A and B are similar as above.

Quickly we can get the value of Λ by right multiply R^T.

$$\Lambda = BR^T - RAR^T \tag{27}$$

Because Λ is symmetric and RAR^T is symmetric too, we have

$$BR^T = (BR^T)^T = RB^T \tag{28}$$

The same as in paper,[12] SVD can be applied in the following.

$$B = USV^T \tag{29}$$

And we have the final solution.

$$R = UV^T \tag{30}$$

On the other hand, in the 2D-2D case, the orthonormality constraints have be changed in the deduction, $R_{2\times 2} R_{2\times 2}^T = \begin{pmatrix} 1 - r_{13}^2 & -r_{13} r_{23} \\ -r_{13} r_{23} & 1 - r_{23}^2 \end{pmatrix}$, which is an arbitrary matrix, and SVD approaches cannot be applied. However it can be computed with Kronecker Product, Bartels-Stewart approaches or some numerical methods.

3.2. Kronecker Produc

Kronecker product has a thriving application in image processing and signal processing, and it provides a very effective way for fast linear transforms. And with the increasing power of computers, this important matrix operation will play a greater role in the future.[13] An example solvent with Kronecker Product and vec operator is

$$vec(R_{2\times 2}) = (I \otimes \Lambda + A \otimes I)^{-1} vec(B) \tag{31}$$

More information about Kronecker Product and vec operator can be found in.[13]

3.3. Error Analysis

Another limitation in the existing solutions for pose estimation problems is that the error analysis is seldom conducted. But benefiting from the thorough investigation about Sylvester's Equation in the past years by researchers from the mathematics, automatic control and other fields, we can directly apply their perturbation theory into our system.

According to Higham,[14] the perturbed Sylvester's Equation is considered.

$$\begin{aligned}(\Lambda + \Delta\Lambda)(R_{2\times 2} + \Delta R_{2\times 2}) &+ (R_{2\times 2} + \Delta R_{2\times 2})(A + \Delta A) \\ &= B + \Delta B\end{aligned} \tag{32}$$

we have the following results

$$\frac{\|\Delta R_{2\times 2}\|_F}{\|R_{2\times 2}\|_F} \le \sqrt{3}\epsilon\Phi \tag{33}$$

where

$$\begin{aligned}\Phi &= \|P^{-1}\|_2 \frac{(\|\Lambda\|_F + \|A\|_F)\|R_{2\times 2}\|_F + \|B\|_F}{\|R_{2\times 2}\|_F} \\ \epsilon &= max\{\frac{\|\Delta\Lambda\|_F}{\|\Lambda\|_F}, \frac{\|\Delta A\|_F}{\|A\|_F}, \frac{\|\Delta B\|_F}{\|B\|_F}\}\end{aligned} \tag{34}$$

And $P = I \otimes \Lambda + A \otimes I$. Here the Frobenius norm is used, $\|X\|_F = (\Sigma_{ij}|x_{ij}|^2)^{1/2}$ and $\|Y\|_2 = (Y^T Y)^{1/2}$.

3.4. Lagrange Multipliers

The difficulty in our method is that the matrix Λ is unknown. In our experiments, we firstly calculating an approximation of $R_{2\times 2}$. For example, if we replace the translation vector in Eq. 16 with Eq. 21, and then we can obtain the following with the least square method.

$$R_{2\times 2} = P_2' P_1'^T (P_1' P_1'^T)^{-1} \tag{35}$$

And then put back to Eq. 25 to calculate the Λ, which is the initial evaluation. We should also compute iteratively to get a Λ such that the following equation satisfies.

$$min \quad \|R_1 R_2^T \pm \sqrt{(1 - R_1 R_1^T)(1 - R_2 R_2^T)}\| \tag{36}$$

where whether $+$ or $-$ can be decided by the direction of the rotation.

3.5. Distance Between Projected Feature Points

With the obtained rotation values $R_{2\times 2E}$ and translation parameters L_E, we can compute $w_{feature}^{(i)}$. Firstly, the feature points from the first frame are projected into the second frame, and the projected points (u_{12}^j, v_{12}^j) are

$$\begin{pmatrix} u_{12}^j \\ v_{12}^j \end{pmatrix} = R_{2\times 2E} \begin{pmatrix} u_1^j \\ v_1^j \end{pmatrix} + \begin{pmatrix} l_{xE} \\ l_{yE} \end{pmatrix} \tag{37}$$

Because there are usually more than five pairs filled, we cannot guarantee $u_{12}^j = u_2^j$ and $v_{12}^j = v_2^j$. Obviously small distances are evaluated with large weight, calculated as

$$w_{feature}^{(i)} = \Sigma_j \frac{1}{\sqrt{(u_2^j - u_{12}^j)^2 + (v_2^j - v_{12}^j)^2}} \tag{38}$$

4. EXPERIMENTAL RESULTS

We have demonstrated our algorithm on different image sequences with preliminary results. The out of plane rotation is the most useful in pose estimation, so these two angles are shown above and left respectively. The scale of the object is identified by the size of the ellipse. Moreover the orientation of the ellipse also indicates the in-plane rotation angles.

We have used 50 particles per frame in all the following simulations. To compare our proposed Sylvester's Equation approach against the state of the art, we also implemented an approach with Essential Matrix (EM)[15] and the method (GM) in.[16] We also employ the Pointing database[17] to build the head pose models, and each pose is trained by nine samples. Our emulation is different with their original one mainly in two aspects. In,[16] they claimed that they chose $K = 2$ clusters when dealing with the feature vectors, but here we just realized $K = 1$ for the simplification. On the other hand, to further make the head models more robust to background clutter, they also learned a face skin model from the training data, but we do not append this part.

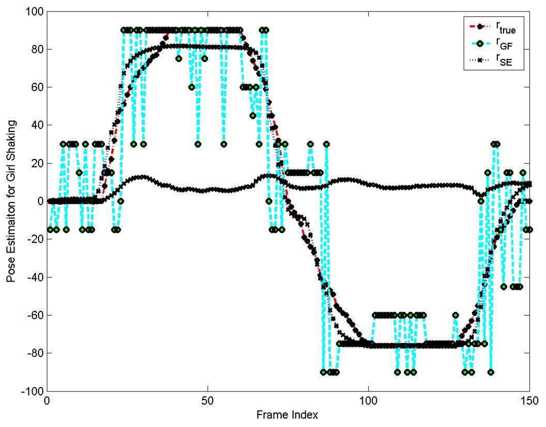
The *Girl* sequences [*] has 318 frames with a resolution of 720×576 pixels and a frame rate of 25 frames per second (fps). But in our experiments, it is firstly resized as 128×96 pixels, and the head region is about 40×40 pixels. A girl with a shock of black hair looked right, left, up and down respectively, and almost fully turned in each direction. This video has clear blue background, and is uniformly lighted excepted some eventual frames. We use this video to demonstrate the robustness of our Sylvester's Equation (SE) algorithm for the large angles.

As can be seen from Figure. 3 (a) and (b), we compare our proposed SE method with GM approach[16] with this test video. Generally our proposed SE method could catch the movement closely, and the false estimation in the other direction is also mostly depressed.

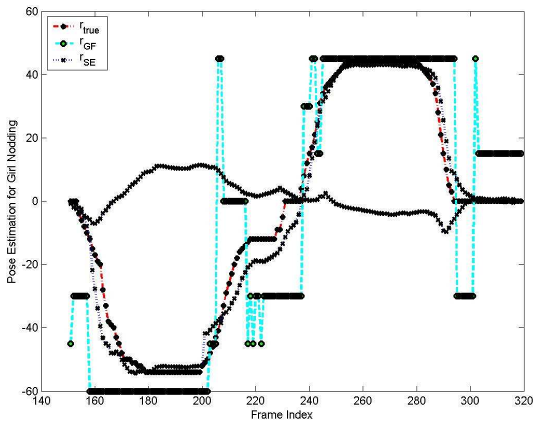
These two figures also indicate that our SE method performs better with the shaking movement than the nodding moving, which can be explained. Firstly we model the head as a ball, so it should be symmetric in all four directions. This assumption is accurate with the horizontal movement, but is not the fact in the vertical direction, because when a person look up and down with large angles, the rotation center is not the ball center any more. And it cannot be rectified with an ellipsoid model either, so we take the measure of shifting the center a little downwards to simulate the actual human head rotation.

On the other hand, the GM[16] method could output just some discrete pose states because of the limited training. What's more, these results are not stable and always flipping. Particularly as can be seen in Figure. 4 (a), these four successive frames have the same pose state, but the GM approach outputs three different estimation results because of the tiny difference in the head region resulting from the tracking outputs. Although it has combined tracking and pose estimation within the framework of particle filtering, the GM algorithm is still sensitive to the tracking steps, because it is a template matching method after all. In comparison, the output with our method is stable, and not so sensitive to tracking results, because we have discarded some falsely matched feature pairs automatically.

[*]These image sequences are downloaded from http://sampl.ece.ohio-state.edu/database.htm



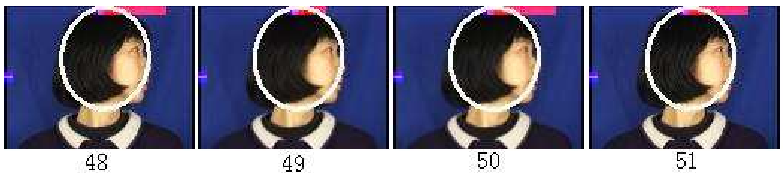
(a)



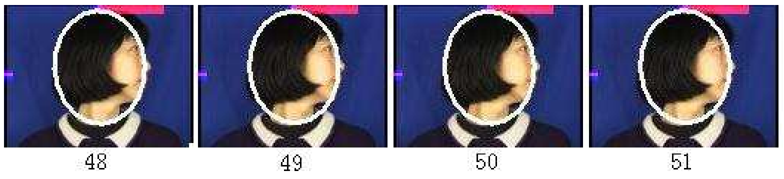
(b)

(The red curve is the true value, the cyan one is the result from GM method[16] and the blue one is with our proposed SE method. The line in the middle is the output noise in the other direction with our SE method.)

Figure 3. Pose Estimation for Girl Shaking and Girl Nodding



(a)



(b)

Figure 4. Pose Estimation results of the video Girl Shaking: (a) GM method[16]; (b) the proposed SE algorithm

The test video *Boy Nodding* † has 217 frames. The resolution is 320×240 and the frame rate is 30 fps. With a complex background, a boy is nodding slightly and frequently. This demo is not very clearly shot, and is used to show the sensitivity of our algorithm.

As shown in Figure. 5 (a), GM method[16] performs much worse comparing the above experiment. Because we trained the system with some clearly identified pictures, and the above test video is also with the good illumination, but this video is dark. As can be seen from Figure. 6, because the video is not so clear, it is harder to track the head movement and extract enough information from these sequences. The GM approach is a template-matching method, and these kinds of methods are generally sensitive to the illumination environments.

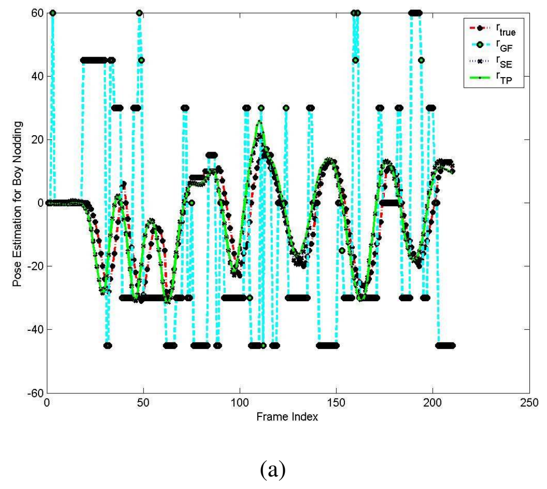
We also compared our proposed SE method with a version of first tracking and then pose estimation approach (TP) with the same coefficients as in the SE algorithm. And the output error is shown in Figure. 5 (b). As can be seen in Figure. 5 (a), the TP method also follows the true values closely, which indicates that our proposed method is not sensitive to the

†These image sequences are downloaded from http://www.cs.bu.edu/groups/ivc/data.php

tracking results. We have explained that some false corresponding feature pairs have been deleted by their length and slope, so our proposed method will be accurate if most of the features are correctly identified.

It is obvious that our proposed SE method is better than the traditionally first tracking and then pose estimation method. The reason is that in Section 3.5, the points are projected back with the estimated parameters to measure the weights of particle filtering, which is some kind of a feedback process to choose the best evaluated particles for the next frame. We applied just 50 particles for these experiments, and less noise in Figure. 5 (b) could be achieved with more particles.

On the other hand, as shown in Figure. 6, although there is some noise in our methods, but generally it does not affect the accurate to catch the true values.



((a) The red curve is the true value, the cyan one is the result from GM method,[16] the blue one is with our proposed SE method, and the green curve results from TP method. (b) These two lines are the output noise in the shaking direction from SE and TP algorithm respectively.)

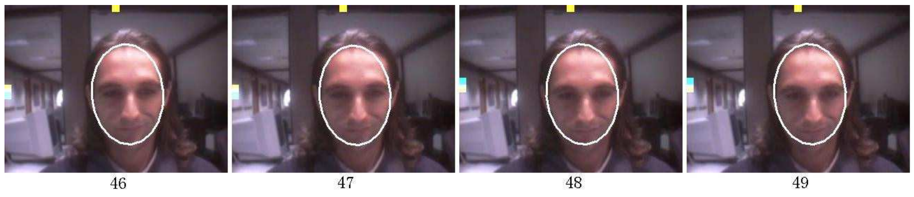
Figure 5. Pose Estimation for Boy Nodding and the Noise in the other direction.

5. CONCLUSIONS

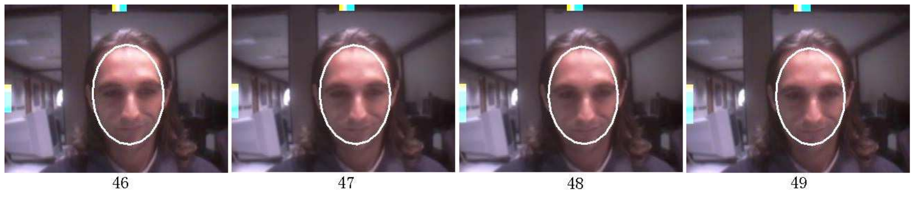
This paper introduces a method to put tracking and pose estimation simultaneously within the framework of particle filtering. Our approach could directly estimate the 3D transform parameters from 2D feature points in the images, without constructing a 3D model or system training and learning before hand. The motion based particle filter is applied to improve the computational efficiency. SIFT and Sylvester's Equation are utilized to estimate the rotation parameters from feature points between successive frames. The application of our method can be extended beyond human head, such as cars, doors and buildings.

REFERENCES

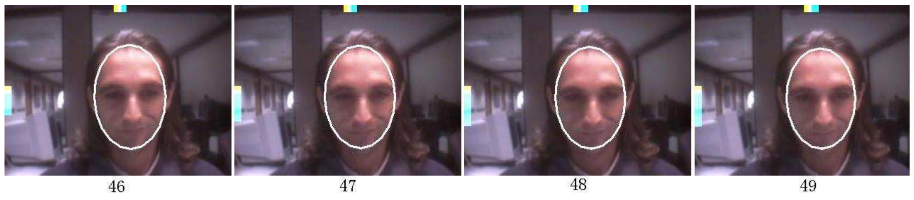
1. S. O. Ba and J. M. Odobez, "A probabilistic framework for joint head tracking and pose estimation," in *Proceedings of 17th International Conference Pattern Recognition*, **4**, pp. 264–267, 2004.
2. K. S. Arun, T. S. Huang, and S. D. Rlostein, "Least-squares fitting of two 3-d point sets," *IEEE Trunsactions on Pattern Analysis and Machine lntelligence* **9**(5), pp. 698–700, 1987.
3. M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for. online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions Signal Processing* **50**(2), pp. 174–188, 2002.
4. M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *Proceedings European Conference on Computer Vision*, pp. 343–356, 1996.



(a)



(b)



(c)

Figure 6. Pose Estimation results of the video Boy Nodding: (a) GM method[16]; (b) TP approach; (c) the proposed SE algorithm

5. N. Bouaynaya and D. Schonfeld, "Complete system for head tracking using motion-based particle filter and randomly perturbed active contour," in *Proceedings of SPIE, Image and Video Communications and Processing (IVCP'05)*, **5685**, pp. 864–873, 2005.
6. K. Nummiaroa, E. K. Meierb, and L. V. Gool, "An adaptive color-based particle filter," *Image and Vision Computing* **21**(11), pp. 99–110, 2002.
7. D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision* **60**(2), pp. 91–110, 2004.
8. M. A. Fischler and 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), pp. 381–395, 1981.
9. A. A. T. Jebara and A. Pentland, "3d structure from 2d motion," *IEEE Signal Processing Magazine* **16**(3), pp. 66–84, 1999.
10. S. S. H. Adams and D. Strelow, "An empirical comparison of methods for image-based motion estimation," in *IEEE/RSJ International Conference on Intelligent Robots and System*, **1**, pp. 123–128, 2002.
11. Q. Ji, "3d face pose estimation and tracking from a monocular camera," in *Image and Vision Computing*, 2002.
12. R. M. Haralick, C.-N. L. H. Joo, V. G. V. X. Zhuang, and M. B. Kim, "Pose estimation from corresponding point data," *IEEE Trunsactions on Systems, Man and Cybernetics* **19**(6), pp. 698–700, 1989.
13. C. F. V. Loan, "The ubiquitous kronecker product," *Journal of Computational and Applied Mathematics* **123**, pp. 85–100, 2000.
14. N. J. Higham, "Perturbation theory and backward error for ax+xb=c," in *Numerical Analysis Report No. 211, University of Manchester*, 1992.
15. X. Zhuang and 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), pp. 1492–1500, 1986.
16. S. O. Ba and J. M. Odobez, "Evaluation of multiple cues head pose tracking algorithm in indoor environments," in *Proceedings of International Conference on Multimedia and Expo, ICME 2005*, 2005.
17. "Pointing'04 icpr workshop: Head pose image database," in *http://www-prima.inrialpes.fr/Pointing04/data-face.html*, 2004.