POSE ESTIMATION BY LOCAL PROCRUSTES REGRESSION

Bisser Raytchev, Kazuya Terakado, Toru Tamaki and Kazufumi Kaneda

Department of Information Engineering, Hiroshima University, Japan

ABSTRACT

In this paper we propose a new method for appearance-based pose estimation, called Local Procrustes Regression (LPR). In LPR, rather than learning a map between all available training samples and pose space, as is common for appearance-based pose estimation algorithms, the pose of an unknown sample is recovered locally from a small subset of the training samples, by utilizing their inter-point distances. This is accomplished by using Procrustes analysis to align the low-dimensional image subspace generated for the neighborhood of the test sample to its corresponding part of pose space. Experimental results obtained on the Object Pose Estimation Database (OPED) indicate that the proposed method performs on par with state-of-the-art methods like support vector regression.

Index Terms— Pose estimation, Regression, Multidimensional Scaling (MDS), Procrustes analysis, k-nearest neighbors

1. INTRODUCTION

Pose estimation is an important problem in computer vision, with many potential applications in object tracking and recognition, virtual and augmented reality, human-computer interfaces, robotic vision, etc. Although many different approaches have been proposed to solve this problem, most of them can be broadly classified as either model-based [1], appearance-based [2], or a mixture of these two. In this paper we focus on the appearance-based (also called view-based) pose estimation methods, which typically learn a map between a set of training samples and pose space. Usually a single map is learned, which characterizes this approach as a predominantly global one. Here we propose an alternative, local approach, based on a new method which we call Local Procrustes Regression (LPR). In our method, rather than learning a map between all available training samples and pose space, the pose of an unknown sample is recovered locally from a small subset of the training samples, by utilizing their inter-point distances.

The proposed method has the following advantages. It is simple and easy to implement (based on standard linear algebra operations); it is fast enough to permit real-time performance; it efficiently deals with high-dimensional images; experimental evaluation on what seems to be the most accurate (with sufficient accuracy for robot grasping) publicly available pose estimation database [7] suggests that the proposed method performs on par with state-of-the-art methods like support vector regression.

The rest of this paper is organized as follows. In section 2 we provide an outline of the proposed method. Sections 3 and 4 give the technical details. Section 5 reports on the experimental evaluation of the method, and section 6 concludes the paper.

Figure 1. Outline of the proposed pose estimation method (see text for details).

2. LOCAL PROCRUSTES REGRESSION

Here we give an outline of the proposed pose estimation method, Local Procrustes Regression (LPR). Let \( X_T = \{x_1, \ldots, x_N\} \) be a set of \( N \) images (in a column-vector form) of a certain object, where each image \( x_i \) represents a different view of the object. The pose of each image in \( X_T \) is known, and represented as \( P_T = [p_1, \ldots, p_N] \), where \( p_i \) is a parameter vector, giving the pose of \( x_i \). \( X_T \) and \( P_T \) taken together represent the training set \( T \). Then pose estimation amounts to finding the pose parameter \( p_0 \) corresponding to a test image \( x_0 \), where \( x_0 \) is generally not in \( X_T \), i.e. it might represent a view which is not included in the training set.

LPR solves the pose estimation problem in three steps. First, the \( k \)-nearest neighbors of \( x_0 \) are found among all \( x_i \) in \( X_T \). Let’s store them as rows in a matrix \( X^{(k)} = \begin{bmatrix} x_1 \cdots x_k \end{bmatrix} \), and let \( p^{(k)} = \begin{bmatrix} p_1 \cdots p_k \end{bmatrix} \) be their corresponding pose parameters. We form also \( X = \begin{bmatrix} x_0 \ x_1 \cdots x_N \end{bmatrix} \) by inserting one more row, \( x_0' \), in \( X^{(k)} \). Next, (see Fig. 1.), by using Multidimensional Scaling (MDS) [3, 4], \( X \) is mapped from the high-
dimensional image space $H$ to a low-dimensional subspace $L$, where the image of $X$ is $Y = (y_0, y_1, \ldots, y_k)$. The mapping between $H$ and $L$ is done in a distance-preserving way, i.e. the relative distances between any two vectors $x_i$ and $x_j$ in $H$ and between their images $y_i$ and $y_j$ in $L$ are preserved. During the last step of the algorithm, using the known $k$ correspondences between $X$ and $P^{(k)}$, and therefore between $Y$ and $P^{(k)}$, we find the optimal transformation between them, which minimizes the “goodness of fit” criterion (explained below), using Procrustes analysis [5]. Then $p_0$ can be obtained by using the same transformation applied to $y_0$. The details are given in the following two sections.

3. MULTIDIMENSIONAL SCALING (MDS)

Multidimensional scaling is a term used to denote a group of techniques which obtain a low-dimensional representation of a set of data points by analyzing the distance matrix of the data. Although many different types of MDS exist, here we use the so-called classical MDS [4] to obtain the low-dimensional mapping $Y$ of $X$, as explained in the previous section. First, we form the $(k+1) \times (k+1)$ distance matrix $D = (d_{ij})$, where $d_{ij}$ are the Euclidean distances between any two images in $X$. From $D$, the following matrix is formed:

$$A = (a_{ij}), \quad a_{ij} = -\frac{1}{2} d_{ij}^2.$$  \hspace{1cm} (1)

Next, the “doubly-centered” matrix $B = CAC$ is formed, where $C$ is the centering matrix

$$C = I_n - n^{-1} J_n, \quad J_n = 1_n 1_n'$$  \hspace{1cm} (2)

and $J_n$ is a $(k+1) \times (k+1)$ matrix of ones ($n = k+1$). Now, the eigenvectors $v_i$ corresponding to the $m$ largest positive eigenvalues $\lambda_i$ of $B$ are found ($m < n$), and the required $m$-dimensional low-dimensional mapping of the data is given by

$$Y = VA^{1/2} = (\sqrt{\lambda_1} v_1, \ldots, \sqrt{\lambda_m} v_m) = (y_0, y_1, \ldots, y_m)'$$  \hspace{1cm} (3)

where the $(m \times m)$ orthonormal matrix $A$ can be interpreted as the $m$-dimensional equivalent of rotation/reflection, $b$ is translation and $s$ the isotropic dilation, needed to align the two data sets $Y$ and $P^{(k)}$. (Note that if the dimensionality of $P^{(k)}$ is lower than $m$, we can add columns of zeros to make it equal to $m$.) Procrustes analysis (see [5]) shows that by differentiating (4) with respect to $A$, $b$ and $s$, the optimal solution is given by

$$A = VU', \quad Z = Y'P^{(k)} = VLU',$$  \hspace{1cm} (5)

$$s = \text{trace}(L)/\text{trace}(YY')$$  \hspace{1cm} (6)

$$b = \bar{p} - sA'y, \quad \bar{p} = \frac{1}{k} \sum_{i=1}^k y_i, \quad \bar{y} = \frac{1}{k} \sum_{i=1}^k y_i'.$$  \hspace{1cm} (7)

In (5), $Z = VLU'$ is the singular value decomposition of $YP^{(k)}$, so that $V$ and $U$ are orthogonal $(m \times m)$ matrices and $L$ is a diagonal matrix containing the singular values. The residual $R$, which provides a measure of how well $Y$ and $P^{(k)}$ are aligned is given by

$$R = \text{trace}(PP') + s^2 \text{trace}(YY') - 2s \text{trace}(P'YY'P)^{1/2}$$  \hspace{1cm} (8)

Once $A$, $b$ and $s$ are determined, $p_0$, the estimated pose parameter vector corresponding to the test image $x_0$ is obtained from its low-dimensional representation $y_0$ as

$$p_0 = sA'y_0 + b.$$  \hspace{1cm} (9)

4. PROCRUSTES ANALYSIS

Once we have obtained the low-dimensional representation $Y$ of the images in $X$, we need to find a transformation which would align $Y$ to $P^{(k)}$, optimizing at the same time some meaningful criterion. In LPR we adopt the “goodness of fit” measure obtained by moving all $y_i$ in $Y$ (excluding $p_0$ as its correspondence $p_0$ in $P$ is not known yet) relative to the corresponding $p_i$ in $P^{(k)}$ until the residual sum of squares is minimal:

$$R = \min_{A,b,s} \sum_{i=1}^k (p_i' - sA'y_i' - b)^2 (p_i' - sA'y_i' - b)$$  \hspace{1cm} (4)

where the $(m \times m)$ orthonormal matrix $A$ can be interpreted as the $m$-dimensional equivalent of rotation/reflection, $b$ is translation and $s$ the isotropic dilation, needed to align the two data sets $Y$ and $P^{(k)}$. (Note that if the dimensionality of $P^{(k)}$ is lower than $m$, we can add columns of zeros to make it equal to $m$.) Procrustes analysis (see [5]) shows that by differentiating (4) with respect to $A$, $b$ and $s$, the optimal solution is given by

$$A = VU', \quad Z = Y'P^{(k)} = VLU',$$  \hspace{1cm} (5)

$$s = \text{trace}(L)/\text{trace}(YY')$$  \hspace{1cm} (6)

$$b = \bar{p} - sA'y, \quad \bar{p} = \frac{1}{k} \sum_{i=1}^k y_i, \quad \bar{y} = \frac{1}{k} \sum_{i=1}^k y_i'.$$  \hspace{1cm} (7)

In (5), $Z = VLU'$ is the singular value decomposition of $YP^{(k)}$, so that $V$ and $U$ are orthogonal $(m \times m)$ matrices and $L$ is a diagonal matrix containing the singular values. The residual $R$, which provides a measure of how well $Y$ and $P^{(k)}$ are aligned is given by

$$R = \text{trace}(PP') + s^2 \text{trace}(YY') - 2s \text{trace}(P'YY'P)^{1/2}$$  \hspace{1cm} (8)

Once $A$, $b$ and $s$ are determined, $p_0$, the estimated pose parameter vector corresponding to the test image $x_0$ is obtained from its low-dimensional representation $y_0$ as

$$p_0 = sA'y_0 + b.$$  \hspace{1cm} (9)

Figure 2. Several views of the objects in the OPED database.
5. EXPERIMENTS

Here we report the results obtained by the proposed method on a typical pose estimation task. We have used the publicly available Object Pose Estimation Database (OPED), which is described in [6], and can be downloaded from [7]. This database contains 16 objects, each sampled at 5° angle increments along two rotational axes (pan and tilt: pan changes between 0° and 180°, and tilt between 0° and 90°). This means that for each object there are 703 different images/views available. Several representative views for each object are shown in Fig. 2. We randomly remove 100 images to be used as test views, and use the remaining 603 images as a training set. This procedure is repeated 100 times and the mean absolute error obtained for each object is reported in Table 1. We implement LPR with \( k \) = 5 and 10 (\( k \)-nearest neighbors being used) and compare with the results obtained when Linear regression or Support Vector Regression (SVR) [8] is used to learn a global map between the training images and pose space. Additionally, we have implemented a nearest neighbor (NN) pose estimation method, which selects the pose of the most similar image to the test image, among the images in the training set. Note that to ease comparison, the two pose angles (for pan and tilt) are combined into a vector on the unit sphere, so that the absolute angle error between ground-truth and estimated pose can be represented by a single value.

Table 1. Experimental results. Shows mean absolute pose angle errors for each object, averaged over 100 trials. The best result is highlighted in red.

<table>
<thead>
<tr>
<th>object</th>
<th>NN</th>
<th>SVR</th>
<th>Linear</th>
<th>LPR ( k=5 )</th>
<th>LPR ( k=10 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>adapter</td>
<td>3.39</td>
<td>1.70</td>
<td>1.87</td>
<td>1.65</td>
<td>1.11</td>
</tr>
<tr>
<td>bay</td>
<td>5.70</td>
<td>1.65</td>
<td>1.72</td>
<td>2.75</td>
<td>2.43</td>
</tr>
<tr>
<td>cable</td>
<td>3.39</td>
<td>0.82</td>
<td>0.79</td>
<td>1.80</td>
<td>0.81</td>
</tr>
<tr>
<td>cap</td>
<td>3.37</td>
<td>0.82</td>
<td>1.13</td>
<td>3.91</td>
<td>3.39</td>
</tr>
<tr>
<td>clamp</td>
<td>3.72</td>
<td>2.15</td>
<td>1.58</td>
<td>1.16</td>
<td>1.23</td>
</tr>
<tr>
<td>fuse</td>
<td>3.57</td>
<td>1.12</td>
<td>0.83</td>
<td>1.57</td>
<td>1.41</td>
</tr>
<tr>
<td>goldcar</td>
<td>3.40</td>
<td>1.36</td>
<td>1.24</td>
<td>1.64</td>
<td>1.13</td>
</tr>
<tr>
<td>house</td>
<td>4.11</td>
<td>1.19</td>
<td>1.94</td>
<td>1.66</td>
<td>1.86</td>
</tr>
<tr>
<td>ipipe</td>
<td>4.76</td>
<td>1.18</td>
<td>1.70</td>
<td>3.19</td>
<td>3.76</td>
</tr>
<tr>
<td>redcar</td>
<td>3.46</td>
<td>1.40</td>
<td>1.77</td>
<td>1.20</td>
<td>1.14</td>
</tr>
<tr>
<td>socketin</td>
<td>3.70</td>
<td>0.84</td>
<td>1.34</td>
<td>4.39</td>
<td>2.17</td>
</tr>
<tr>
<td>socketout</td>
<td>3.70</td>
<td>0.84</td>
<td>1.01</td>
<td>3.40</td>
<td>1.83</td>
</tr>
<tr>
<td>tpipe</td>
<td>4.56</td>
<td>1.74</td>
<td>1.74</td>
<td>0.68</td>
<td>1.21</td>
</tr>
<tr>
<td>trap</td>
<td>3.67</td>
<td>0.92</td>
<td>0.82</td>
<td>2.88</td>
<td>1.76</td>
</tr>
<tr>
<td>whitecar</td>
<td>3.82</td>
<td>1.15</td>
<td>1.53</td>
<td>0.34</td>
<td>0.48</td>
</tr>
<tr>
<td>wood</td>
<td>3.41</td>
<td>2.12</td>
<td>1.09</td>
<td>0.68</td>
<td>0.90</td>
</tr>
</tbody>
</table>

As can be seen from Table 1, LPR performs very well for \( k = 5 \) and \( k = 10 \), obtaining best accuracy (indicated in red) for 7 among the 16 objects, followed by SVR, which has best accuracy for 6 objects. Figure 3 (on last page) shows one typical instance of the aligned low-dimensional image subspace and pose space, generated by the LPR algorithm for one test sample from the object clamp. In the figure, the ground truth pose parameters (pan and tilt angles in degrees) for the training data are indicated by red circles, while the positions of the mapped training images are indicated by blue stars. For the test sample, the ground-truth pose parameters are indicated by a black cross, at position (135, 0) for the present example. The estimated pose parameters are indicated by a green cross at position (135.9, -1.1). The i-th nearest neighbor number of the test sample among the training samples is given in brackets before the position coordinates (in this case 10-nearest neighbors were used)

Regarding speed, our unoptimized MATLAB implementation of LPR needs less than half a second on a standard computer to estimate the pose of a single test sample (using 603 training samples). The most computationally intensive part is the search for the \( k \)-nearest neighbors among the training samples. Although a few hundred training samples are usually enough to provide a good accuracy pose estimation, for much larger training sets more efficient methods for searching for the nearest neighbors might be used to reduce computational time.

6. CONCLUSION

In this paper we have proposed a new method for appearance-based pose estimation, in which the pose of an unknown sample is recovered locally from a small subset of the training samples, by utilizing their inter-point distances. We have shown how Procrustes analysis can be used to align the low-dimensional image subspace generated by the neighborhood of the test sample to its corresponding part of pose space. The experimental results reported for the Object Pose Estimation Database (OPED) demonstrate the effectiveness of the proposed method. Further work includes considering alternative methods for distance-preserving low-dimensional mapping, for example, like manifold learning methods [9-12], and also experimenting with other possible ways for finding an alignment between the low-dimensional image subspace and parameter space.

REFERENCES


Figure 3. Figure generated by the LPR algorithm ($k = 10$) for one test sample from the object **clamp**. For the **training** data, the ground truth pose parameters are indicated by red circles, while the positions of the mapped images are indicated by blue stars. For the test sample, the ground-truth pose parameters are indicated by a black cross, at position (135, 0) for the present example. The estimated pose parameters are indicated by a green cross at position (135.9, -1.1). The $i$-th nearest neighbor number of the test sample among the training samples is given in brackets before the position coordinates.