On Partial Least Squares in Head Pose Estimation: How to simultaneously deal with misalignment

Murad Al Haj[1] Jordi Gonzàlez[1] Larry S. Davis[2]

[1]Centre de Visió per Computador
Universitat Autònoma de Barcelona
Bellaterra, Barcelona 08193, Spain
{malhaj, poal}@cvc.uab.es

[2]Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742, USA
lsd@umiacs.umd.edu

Abstract

Head pose estimation is a critical problem in many computer vision applications. These include human computer interaction, video surveillance, face and expression recognition. In most prior work on heads pose estimation, the positions of the faces on which the pose is to be estimated are specified manually. Therefore, the results are reported without studying the effect of misalignment. We propose a method based on partial least squares (PLS) regression to estimate pose and solve the alignment problem simultaneously. The contributions of this paper are two-fold: 1) we show that the kernel version of PLS (kPLS) achieves better than state-of-the-art results on the estimation problem and 2) we develop a technique to reduce misalignment based on the learned PLS factors.

1. Introduction

Head pose is an extremely powerful communication tool that conveys important nonverbal messages about subjects. The work of Langton *et al*. [10] showed that head pose is highly correlated with gaze estimation. The main challenges to accurate head pose estimation include: presence or absence of structural components (beards, mustaches, glasses, ...), facial expressions, occlusion, image orientation, and imaging conditions. Numerous papers have been published describing algorithms for head pose estimation and a good recent survey can be found in [15]. It divides the different methods into categories, including: appearance template methods such as [17], detector array methods where a dedicated face detector is trained for every pose as in [29], regression methods like [4], manifold embedding as [19] and geometric methods akin to [26].

The most successful methods for monocular head pose estimation are those using nonlinear regression [8, 15]. Work in this area include neural networks with locally linear maps [18] and multilayer perceptrons [23], in addition to support vector machine regression after PCA projection [12]. However, these nonlinear regression methods are especially sensitive to alignment errors; therefore, their performance diminishes with small localization error.

Alignment is a well-known problem in many recognition algorithms and the authors of [9] attribute the scarcity of fully automated recognition systems to the difficulty of alignment. Alignment is by now well understood as a major subproblem of face recognition [27]. However, it is rarely considered in evaluations of pose estimation; results are typically reported on manually aligned data. A notable exception is Murphy-Chutorian and Trivedi [16]. They developed a system for measuring the position and orientation of a driver's head, and propose the use of localized gradient orientation (LGO) histograms to offset some of the localization error of the underlying face detector.

We present a regression-based pose estimation method that achieves better than state-of-the-art results and handles misalignment effects without the need to include any misaligned sample during training. Given a set of candidate windows from a noisy face detector, we develop a technique that predicts which of those windows is best aligned with the model based on partial least squares (PLS) analysis. The best aligned window is the one to which the pose regression coefficients are then applied. The remainder of the paper is organized as follows: section 2 discusses both linear and kernel PLS regression methods; section 3 shows the results of the two methods on Pointing'04 and CMU Multi-PIE databases. In section 4, we show how PLS can be used to deal with misalignment as well as demonstrate the results of this framework on simulated noisy detections, and section 5 concludes the paper.

2. Partial Least Squares

Although it has been more than three decades since its introduction [28] and more than two decades since its use in the domain of chemometrics [3], partial least squares (PLS) analysis has only recently been attracting attention in computer vision [2, 7, 21]. In its most general form, PLS models the relationship between sets of observed variables by projecting them into a latent space; hence, some researchers refer to PLS as "Projection to Latent Structures". The modeling is done by selecting orthogonal score vectors (a.k.a. latent vectors) that maximize the covariance between the different sets of variables while, at the same time, keeping most of the variance of each set. PLS can be effectively applied to solve regression problems where the number of samples is less than the number of independent variables, as well as in the presence of high collinearity of those variables.

2.1. Linear PLS Regression

Consider a matrix of independent variables $\mathbf{X}$ formed from n observations of N dimensional vectors and a matrix of dependent variables $\mathbf{Y}$ obtained as a response to $\mathbf{X}$ and formed of n observations of M dimensional vectors. PLS decomposes the zero-mean nxN matrix $\mathbf{X}$ and the zero-mean nxM matrix $\mathbf{Y}$ as follows:

$$\mathbf{X} = \mathbf{T}\mathbf{P}^T + \mathbf{E} \quad (1)$$
$$\mathbf{Y} = \mathbf{U}\mathbf{Q}^T + \mathbf{F} \quad (2)$$

where $\mathbf{T}$ and $\mathbf{U}$ are nxd matrices of the d extracted score vectors, i.e. d factors or components. The Nxd matrix $\mathbf{P}$ and the Mxd matrix $\mathbf{Q}$ represent the loadings. The nxN matrix $\mathbf{E}$ and the nxM matrix $\mathbf{F}$ are residual matrices. There exist many methods to obtain the decomposition in equations 1 and 2, the most classical of which is based on the nonlinear iterative partial least squares (NIPALS) algorithm [28], which finds normalized weights $\mathbf{w}$ and $\mathbf{c}$ that maximize the covariance between the score vectors $\mathbf{t}$ and $\mathbf{u}$. In the modification proposed in [11], the normalization of $\mathbf{t}$ and $\mathbf{u}$, rather than the normalization of $\mathbf{w}$ and $\mathbf{c}$, is used and the computation is done in d-iterations where each iteration is as follows:

1. randomly initialize $\mathbf{u}$;
2. $\mathbf{w} = \mathbf{X}^T\mathbf{u}$; $\mathbf{t} = \mathbf{X}\mathbf{w}$; $\mathbf{t} \leftarrow \mathbf{t}/\|\mathbf{t}\|$;
3. $\mathbf{c} = \mathbf{Y}^T\mathbf{t}$; $\mathbf{u} = \mathbf{Y}\mathbf{c}$; $\mathbf{u} \leftarrow \mathbf{u}/\|\mathbf{u}\|$;
4. repeat steps 2-3 until convergence;
5. deflate $\mathbf{X}$: $\mathbf{X} \leftarrow \mathbf{X} - \mathbf{t}\mathbf{t}^T\mathbf{X}$; deflate $\mathbf{Y}$: $\mathbf{Y} \leftarrow \mathbf{Y} - \mathbf{t}\mathbf{t}^T\mathbf{Y}$;

The matrices $\mathbf{T}$, $\mathbf{U}$, $\mathbf{W}$ and $\mathbf{C}$ are formed by columns of of the vectors $\mathbf{t}$, $\mathbf{u}$, $\mathbf{w}$ and $\mathbf{c}$ respectively, obtained at every iteration.

Once the two sets of variables, $\mathbf{X}$ and $\mathbf{Y}$, are projected to latent subspaces, what is left is to find the NxM regression coefficients matrix $\mathbf{B}$ such that:

$$\mathbf{Y} = \mathbf{X}\mathbf{B} + \mathbf{F}^* \quad (3)$$

where $\mathbf{F}^*$ is a residual matrix. From [20], $\mathbf{B}$ can be computed as follows:

$$\mathbf{B} = \mathbf{W}(\mathbf{P}^T\mathbf{W})^{-1}\mathbf{C}^T \quad (4)$$

where the following equalities hold:

$$\mathbf{W} = \mathbf{X}^T\mathbf{U} \quad (5)$$
$$\mathbf{P} = \mathbf{X}^T\mathbf{T}(\mathbf{T}^T\mathbf{T})^{-1} \quad (6)$$
$$\mathbf{C} = \mathbf{Y}^T\mathbf{T}(\mathbf{T}^T\mathbf{T})^{-1}. \quad (7)$$

Given the orthonormality of $\mathbf{T}$, i.e. $\mathbf{T}^\mathbf{T}\mathbf{T} = \mathbf{I}$, and substituting equations 5, 6 and 7 to 4, $\mathbf{B}$ can be expressed as:

$$\mathbf{B} = \mathbf{X}^T\mathbf{U}(\mathbf{T}^T\mathbf{X}\mathbf{X}^T\mathbf{U})^{-1}\mathbf{T}^T\mathbf{Y}. \quad (8)$$

The NIPALS algorithm can be executed in a manner involving only matrix-vector multiplications, rendering the complexity in the order of $O(n^2)$.

2.2. Kernel PLS

Consider a nonlinear transformation of each input vector $\mathbf{x}$ into a feature space $\mathcal{F}$, i.e. mapping Φ: $\mathbf{x}_i \in R^N \rightarrow \Phi(\mathbf{x}_i) \in \mathcal{F}$. Denoting all the mapped vectors $\mathbf{x}$, i.e. $\{\Phi(\mathbf{x}_i)\}_{i=1}^n$, by $\mathbf{\Phi}$ and using the theory of Reproducing Kernel Hilbert Spaces (RKHS) [20], the kernel NIPALS algorithm is:

1. randomly initialize $\mathbf{u}$;
2. $\mathbf{t} = \mathbf{\Phi}\mathbf{\Phi}^T\mathbf{u}$; $\mathbf{t} \leftarrow \mathbf{t}/\|\mathbf{t}\|$;
3. $\mathbf{c} = \mathbf{Y}^T\mathbf{t}$; $\mathbf{u} = \mathbf{Y}\mathbf{c}$; $\mathbf{u} \leftarrow \mathbf{u}/\|\mathbf{u}\|$;
4. repeat steps 2-3 until convergence;
5. deflate $\mathbf{\Phi}\mathbf{\Phi}^T$: $\mathbf{\Phi}\mathbf{\Phi}^T \leftarrow (\mathbf{\Phi} - \mathbf{t}\mathbf{t}^T\mathbf{\Phi})(\mathbf{\Phi} - \mathbf{t}\mathbf{t}^T\mathbf{\Phi})^T$; deflate $\mathbf{Y}$: $\mathbf{Y} \leftarrow \mathbf{t}\mathbf{t}^T\mathbf{Y}$;

Using kernel mapping $K(.)$ and the "kernel trick", one can notice that $\mathbf{\Phi}\mathbf{\Phi}^\mathbf{T}$ represents the kernel Gram matrix $\mathbf{K}$ of the cross dot products between all mapped input, $\{\Phi(\mathbf{x}_i)\}_{i=1}^n$. The deflation of $\mathbf{\Phi}\mathbf{\Phi}^T$ in step 5 is now given by:

$$\mathbf{K} \leftarrow (\mathbf{I} - \mathbf{t}\mathbf{t}^T)\mathbf{K}(\mathbf{I} - \mathbf{t}\mathbf{t}^T) \quad (9)$$
$$\mathbf{K} \leftarrow \mathbf{K} - \mathbf{t}\mathbf{t}^T\mathbf{K} - \mathbf{K}\mathbf{t}\mathbf{t}^T + \mathbf{t}\mathbf{t}^T\mathbf{K}\mathbf{t}\mathbf{t}^T \quad (10)$$

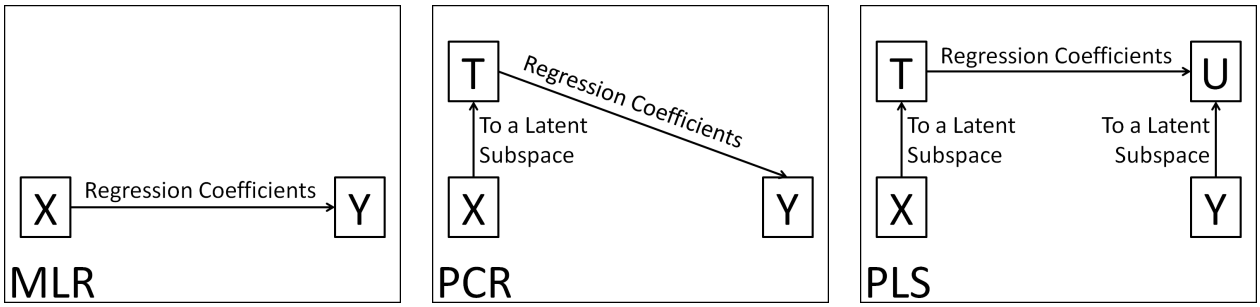

Figure 1. MLR, PCR and PLS coefficients calculation.

and the regression coefficients by:

$$\mathbf{B} = \mathbf{\Phi}^T\mathbf{U}(\mathbf{T}^T\mathbf{K}\mathbf{U})^{-1}\mathbf{T}^T\mathbf{Y}. \quad (11)$$

In our experiments, an rbf kernel was used; therefore, the complexity was increased, from the quadratic order required by NIPALS, to the order of $O(n^2N)$.

2.3. PLS, MLR and PCR

Like any regression method, the aim of PLS is to find a set of coefficients modeling the relationship between the input data $\mathbf{X}$ and its response $\mathbf{Y}$. Other relevant regression methods include Multiple Linear Regression (MLR) and Principal Component Regression (PCR). MLR solves for the regression coefficients directly by establishing a linear relationship between the input and the output. MLR cannot be applied when the inverse of $\mathbf{X}\mathbf{X}^T$ does not exist. PCR determines the coefficients based on the score (or latent) vectors after projecting $\mathbf{X}$ to a subspace determined by the principal components. Since these components are computed only on $\mathbf{X}$, without any consideration of $\mathbf{Y}$, some of them might be irrelevant in predicting the response. PLS projects both $\mathbf{X}$ and $\mathbf{Y}$ each to its latent subspace before computing the regression coefficients. This can be seen in Figure 1. As a rule of thumb, MLR models the maximum correlation between $\mathbf{X}$ and $\mathbf{Y}$, PCR models the maximum variance in $\mathbf{X}$ while PLS models the maximum covariance between $\mathbf{X}$ and $\mathbf{Y}$.

3. Head Pose Estimation

In this section, we will show the results of applying linear and kernel PLS regression to estimate the head pose in two datasets: Pointing'04 and CMU Multi-PIE. We will compare the results with state-of-the-art methods. The feature vector, for each face, is composed of 3-level pyramid Histogram of Oriented Gradients (HOG) extracted from the bounding box and quantized into 8 bins. Therefore, each row of the independent variable $\mathbf{X}$ is composed of 640 dimensions representing the HOG features of the corresponding face while each row of the dependent variable $\mathbf{Y}$ is composed of the corresponding pose; it is two dimensional for Pointing'04 since the dataset contains values for both pitch and yaw while one dimensional, yaw, for CMU Multi-PIE.

3.1. Results on Pointing'04

Per subject, the Pointing'04 database [4] contains poses discretized to 9 angles of pitch: $\{-90°, -60°, -30°, -15°, 0°, 15°, 30°, 60°, 90°\}$ and 13 angles of yaw: $\{-90°, -75°, -60°, -45°, -30°, -15°, 0°, 15°, 30°, 45°, 60°, 75°, 90°\}$. However, when the pitch angle is $-90°$ or $90°$, the yaw angle is always $0°$. Therefore, the total number of poses is: $7x13 + 2x1 = 93$ poses. The total number of subjects is 15, each of whom is photographed twice resulting in 2790 images forming the database. The bounding box containing the face for each image is provided. As indicated before, the features that were used in these experiments are 3-level pyramid HOG and 5-level cross validation is employed, where every sample is tested using a model trained on 80% of the remaning samples. The optimal number of factors was found to be 25 for linear PLS and 40 for kPLS. For kPLS, a radial basis function (rbf) kernel was used with a kernel width of $\sigma = 0.05$. A comparison between PLS and other state-of-the-art methods is shown in table 1. It is interesting to see that kPLS outperforms all other methods while, at the same time, reducing the feature space significantly, from 640 dimensions to 40 latent dimensions. The error shown in table 1 is the mean absolute error (MAE) between the continuous predicted pose and the discrete ground truth pose.

The yaw 'box and whisker' plot for kPLS regression is shown in figure 2, while that for pitch is shown in figure 3. In the lateral figure, the less accurate predictions at pitch poses $-90°$ and $90°$ are due to the much smaller number of training samples at those poses compared to the others; between the two, pitch $-90°$ is less accurate due to the higher variations in the images at this pitch (heads down) compared to those at $90°$ (heads up). Looking at all the poses in pitch and yaw, one concludes that the regression is able to accurately predict head pose with little variance and few outliers. The mean absolute error vs. the number of factors is shown in figure 4 and vs. the rbf kernel width in figure 5, justifying the use of 40 factors and $\sigma = 0.05$.

Method	Yaw Error	Pitch Error	Accuracy (Yaw,Pitch)	Notes
Ours (kernel PLS)	$\mathbf{6.56°}$	$\mathbf{6.61°}$	$\mathbf{(67.36\%, 80.36\%)}$	-
Stiefelhagen [22]	$9.5°$	$9.7°$	$(52.0\%, 66.3\%)$	1
Ours (linear PLS)	$\mathbf{11.29°}$	$\mathbf{10.52°}$	$\mathbf{(45.57\%, 58.70\%)}$	-
Human Performance [5]	$11.8°$	$9.4°$	$(40.7\%, 59.0\%)$	2
Gourier (Associative Memories) [5]	$10.1°$	$15.9°$	$(50.0\%, 43.9\%)$	3
Tu (High-order SVD) [24]	$12.9°$	$17.97°$	$(49.25\%, 54.84\%)$	4
Tu (PCA) [24]	$14.11°$	$14.98°$	$(55.20\%, 57.99\%)$	4
Tu (LEA) [24]	$15.88°$	$17.44°$	$(45.16\%, 50.61\%)$	4
Voit [25]	$12.3°$	$12.77°$	–	-
Li (PCA) [13]	$26.9°$	$35.1°$	–	5
Li (LDA) [13]	$25.8°$	$26.9°$	–	5
Li (LPP) [13]	$24.7°$	$22.6°$	–	5
Li (Local-PCA) [13]	$24.5°$	$37.6°$	–	5
Li (Local-LPP) [13]	$29.2°$	$40.2°$	–	5
Li (Local-LDA) [13]	$19.1°$	$30.7°$	–	5

Notes:
1) Used 80% of Pointing'04 images for training, 10% for cross-evaluation, and 10% for testing.
2) Human performance with training.
3) Best results over different reported methods.
4) Better results have been obtained with manual localization.
5) Results for 32-dim embedding.

Table 1. Comparison of our PLS results to state-of-the-art methods (from [15]) in terms of mean absolute error and classification accuracy.

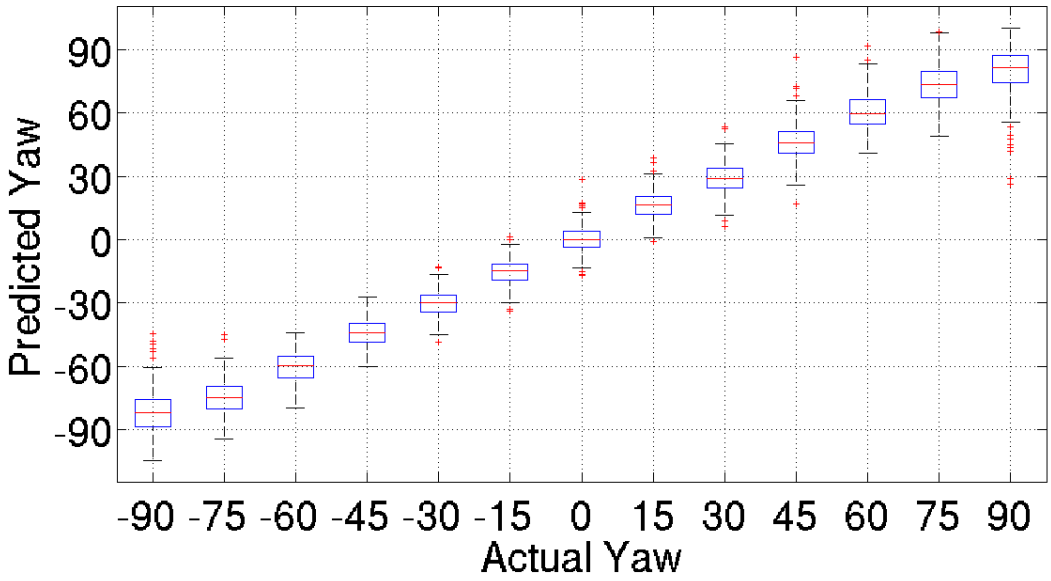

Figure 2. Detailed results of kPLS on Pointing'04 yaw regression.

3.2. Results on Multi-PIE

In this experiment, 2700 face images from the CMU Multi-PIE database [6] were manually annotated. These images belong to 144 subjects, under frontal illumination and varying expressions. Multi-PIE yaw angles range between $-90°$ and $90°$ with increments of $15°$ resulting in 13 discrete poses. Sampled cropped faces are shown in figure 6. We employed a 2-fold cross validation, where one half of the data was used for training while the other half for testing and vice versa. No tuning was done for the database, so the same parameters that were found to optimize the results

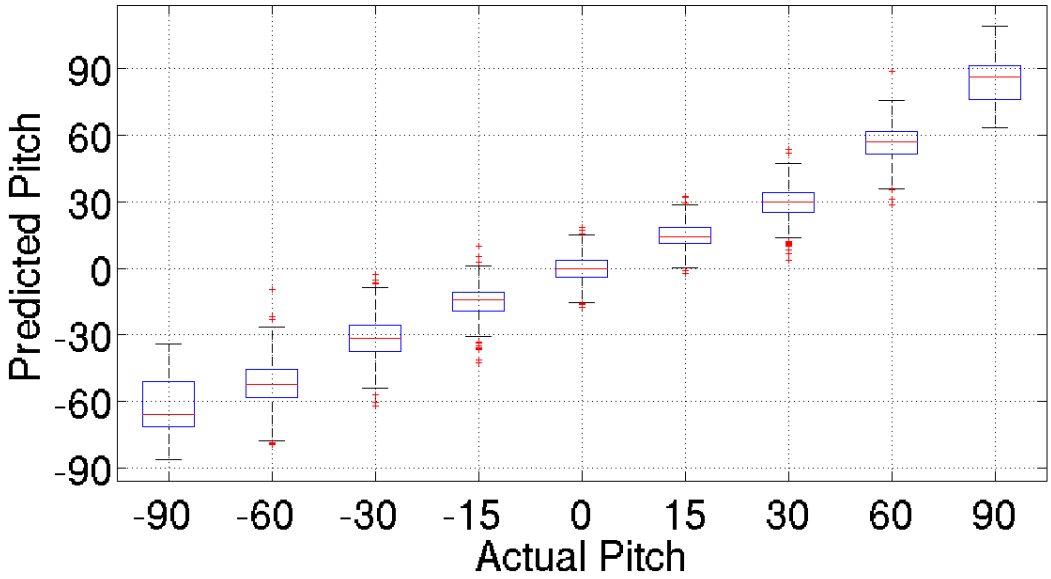

Figure 3. Detailed results of kPLS on Pointing'04 pitch regression.

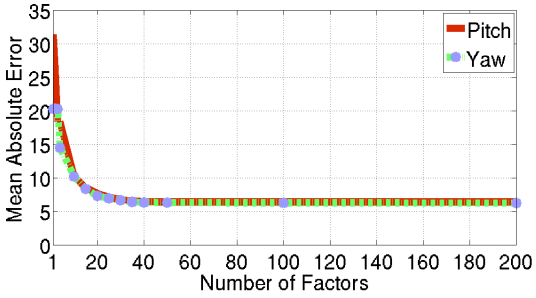

Figure 4. Mean absolute error vs. number of factors. This figure is best viewed in color.

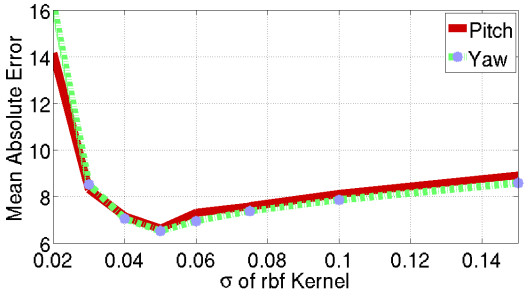

Figure 5. Mean absolute error vs. σ. This figure is best viewed in color.

on Pointing'04 were used, i.e. 25 factors for linear PLS and 40 factors with $\sigma = 0.05$ for kernel PLS. The results for kPLS and linear PLS regression are shown in Table 2

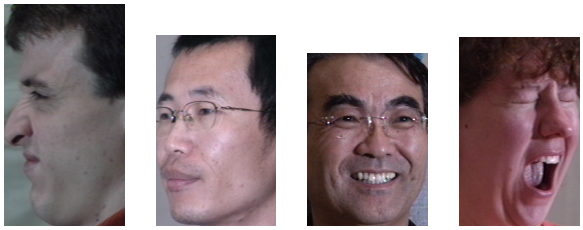

Figure 6. Sample unnormalized cropped faces under different expressions from CMU Multi-PIE.

	kPLS	linear PLS	PCR
MAE	**5.31°**	9.11°	11.03°
Accuracy	**79.48%**	57.22%	48.33%

Table 2. Comparison of the different algorithms in terms of mean absolute error and classification accuracy.

along with those of PCR (also 25 factors were used). MLR could not be applied due to the multicollinearity in the data. kPLS achieved the best results with a mean absolute error of 5.31°.

4. Misalignment

Regression (as well as classification) algorithms are generally sensitive to localization error. If the object is not accurately registered with the learned model, the comparison between the object features and the model features leads to errors. Most pose estimation methods are evaluated on well-annotated data and no analysis of the sensitivity to misalignment is typically reported. To study this problem, we

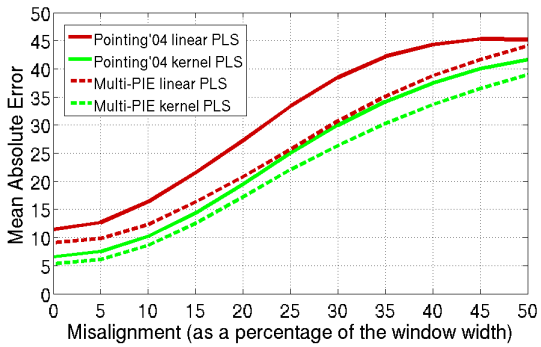

Figure 7. Mean absolute error vs. misalignment.

conducted experiments where the training data is kept unchanged while the test data is regenerated from the images through shifting the face bounding box by a certain percentage of its width. The MAE in yaw pose estimation versus the shift percentage is shown in figure 7. As expected, the greater the misalignment, the worse the pose estimation results. It is also interesting to see that the effect in the kernel version is close to that in the linear version.

To deal with misalignment, we propose the following: given that $\mathbf{T}$ and $\mathbf{U}$ of the PLS model are correlated and given a set of noisy observations, the observation that produces the minimum residual when projected to the latent subspace of $\mathbf{X}$ should have the minimum error between its predicted response and actual response. Therefore, to estimate pose on a candidate face produced by a noisy detection process, we consider not only the detected location of the face but also a set of shifted versions of the face, and choose the instance in the set that produces the minimum residual when projected to the latent subspace. The estimated pose for that instant is the face pose. In the remainder of this section, we derive the equations to calculate this residual for both linear and kernel PLS, and show how regressing on the minimum residual instance can reduce misalignment problems.

4.1. Linear PLS Residual

Given a new vector $\mathbf{x}$ and a trained PLS model, as described in section 2, $\mathbf{x}$ can be approximated as:

$$\mathbf{x} \approx \mathbf{t}\mathbf{P}^T \tag{12}$$

where $\mathbf{t}$ is the score vector corresponding to $\mathbf{x}$ and $\mathbf{P}$ represents the learned loadings. Using equation 6 and the fact that $\mathbf{T^T T} = \mathbf{I}$, $\mathbf{x}$ can be rewritten as:

$$\mathbf{x} \approx \mathbf{t}\mathbf{T}^T\mathbf{X}. \tag{13}$$

To approximate $\mathbf{t}$, the following derivation can be made:

$$\begin{aligned}
\mathbf{x}\mathbf{X}^T &\approx \mathbf{t}\mathbf{T}^T\mathbf{X}\mathbf{X}^T && (14)\\
\mathbf{x}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1} &\approx \mathbf{t}\mathbf{T}^T && (15)\\
\mathbf{x}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{T} &\approx \mathbf{t}, && (16)
\end{aligned}$$

substituting equation 16 to equation 13, $\mathbf{x}$ becomes:

$$\mathbf{x} \approx \mathbf{x}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{T}\mathbf{T}^T\mathbf{X}, \tag{17}$$

and the residual is the error in the approximation, given by:

$$\mathbf{e} = \mathbf{x} - \mathbf{x}\mathbf{X}^T(\mathbf{X}\mathbf{X}^T)^{-1}\mathbf{T}\mathbf{T}^T\mathbf{X}. \tag{18}$$

However, as we stated earlier $(\mathbf{X}\mathbf{X}^T)^{-1}$ might not exist due to multicollinearity in the data. Therefore, a better derivation, that makes use of equation 6, is:

$$\begin{aligned}
\mathbf{x} &\approx \mathbf{t}\mathbf{P}^T && (19)\\
\mathbf{x}\mathbf{P} &\approx \mathbf{t}\mathbf{P}^T\mathbf{P} && (20)\\
\mathbf{x}\mathbf{P}(\mathbf{P}^T\mathbf{P})^{-1} &\approx \mathbf{t} && (21)\\
\mathbf{x}\mathbf{X}^T\mathbf{T}(\mathbf{T}^T\mathbf{X}\mathbf{X}^T\mathbf{T})^{-1} &\approx \mathbf{t}, && (22)
\end{aligned}$$

replacing equation 22 in equation 13, $\mathbf{x}$ becomes:

$$\mathbf{x} \approx \mathbf{x}\mathbf{X}^T\mathbf{T}(\mathbf{T}^T\mathbf{X}\mathbf{X}^T\mathbf{T})^{-1}\mathbf{T}^T\mathbf{X}, \tag{23}$$

and the residual is:

$$\mathbf{e} = \mathbf{x} - \mathbf{x}\mathbf{X}^T\mathbf{T}(\mathbf{T}^T\mathbf{X}\mathbf{X}^T\mathbf{T})^{-1}\mathbf{T}^T\mathbf{X}. \tag{24}$$

Therefore, for a candidate face position, a search is done in its vicinity to find the best aligned window, i.e. the one whose feature vector produces the minimum residual in equation 24. Starting with the original ground-truth face windows, we shifted each of these windows by 5% of its width in the four directions (up, down, right, left), creating a bag containing those four misaligned samples in addition to the original sample. We gradually increased the shifts from 5% to 50% of the width in steps of 5%; in each step, four new samples, whose misalignment is worse than the previous four, are added to each bag. At 50%, each bag contained 41 feature vectors (40 misaligned and the original). The MAE of applying the regression on the selected mimimum residual sample of each bag, at each step, is shown in figure 8. Despite the huge increase in the added noise, the effect on our algorithm is negligibale; the maximum difference in the lateral figure is for Pointing'04 pitch which goes from $10.52°$ at 0% shift to $11.68°$ at 50% shift.

4.2. kPLS Residual

Similar to the linear case, and given the formulation developed in subsection 2.2, the mapped version of $\mathbf{x}$ can be approximated as:

$$\Phi(\mathbf{x}) \approx \mathbf{t}\mathbf{T}^T\mathbf{\Phi}; \tag{25}$$

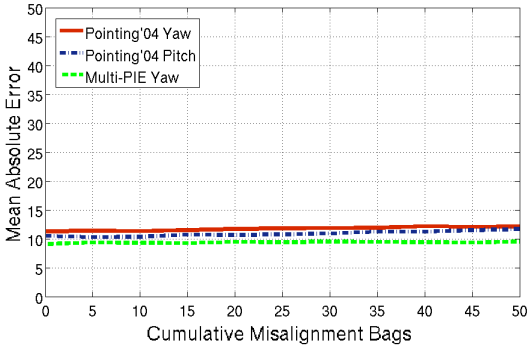

Figure 8. Mean absolute error as more misaligned samples are added to the bags (linear algorithm). This figure is best viewed in color.

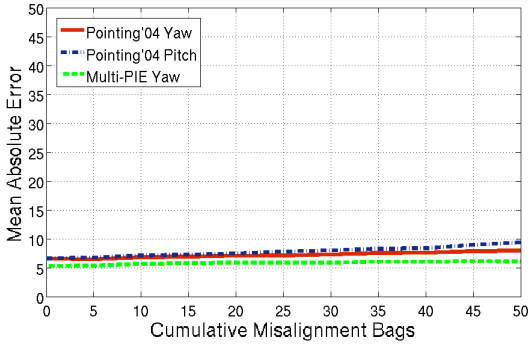

Figure 9. Mean absolute error as more misaligned samples are added to the bags (kernel algorithm). This figure is best viewed in color

also, in a similar manner to deriving equation 16, $\mathbf{t}$ in this case can be approximated as:

$$\mathbf{t} \approx \Phi(\mathbf{x})\mathbf{\Phi}^T(\mathbf{\Phi}\mathbf{\Phi}^T)^{-1}\mathbf{T}, \quad (26)$$

and after the kernel mapping $K(.)$:

$$\mathbf{t} \approx K(\mathbf{x},\mathbf{X})\mathbf{K}^{-1}\mathbf{T}. \quad (27)$$

Unlike the linear product, $\mathbf{X}\mathbf{X}^T$, we can assume that $\mathbf{K}$ is invertible due to the mapping to a much higher dimensional space that eliminates linear dependencies.

Starting from equation 25, the following derivation can be made:

$$(\Phi(\mathbf{x}) - \mathbf{t}\mathbf{T}^T\mathbf{\Phi})(\Phi(\mathbf{x}) - \mathbf{t}\mathbf{T}^T\mathbf{\Phi})^T \approx 0 \quad (28)$$

$$(\Phi(\mathbf{x}) - \mathbf{t}\mathbf{T}^T\mathbf{\Phi})(\Phi^T(\mathbf{x}) - \mathbf{\Phi}^T\mathbf{T}\mathbf{t}^T) \approx 0 \quad (29)$$

$$\Phi(\mathbf{x})\Phi^T(\mathbf{x}) - \Phi(\mathbf{x})\mathbf{\Phi}^T\mathbf{T}\mathbf{t}^T - \mathbf{t}\mathbf{T}^T\mathbf{\Phi}\Phi^T(\mathbf{x}) + \mathbf{t}\mathbf{T}^T\mathbf{\Phi}\mathbf{\Phi}^T\mathbf{T}\mathbf{t}^T \approx 0 \quad (30)$$

$$K(\mathbf{x},\mathbf{x}) - K(\mathbf{x},\mathbf{X})\mathbf{T}\mathbf{t}^T - \mathbf{t}\mathbf{T}^T K^T(\mathbf{x},\mathbf{X}) + \mathbf{t}\mathbf{T}^T\mathbf{K}\mathbf{T}\mathbf{t}^T \approx 0. \quad (31)$$

The same experimental setup as in the previous subsection is used here and the vector with minimum residual is defined as the one minimizing equation 31. The results are shown in figure 9, demonstrating that the method can also be successfully applied in kernel regression.

4.3. Comparison with MIL

Multiple instance learning (MIL), where a label is associated with a bag of instances rather than just a single instance, has been proposed to handle misalignment [1, 14]. We compare our algorithm against Multi-Instance Multi-Label SVM (MIMLSVM), which was shown to outperform other well-known multi-instance learning algorithms [30]. Our kPLS results obtained using bags with up to 50% shifts are compared against MIMLSVM on those same bags and the comparison is shown in table 3. For MIMLSVM, each dataset was divided equally for training and testing and the number of medoids was set to 20% of the training data. After computing the Hausdorff distance, the SVM cost was set to 20 and its rbf kernel width to 90 (these values where experimentally found to provide the best results). Our method outperformed MIMLSVM on average, despite not having any misalined sample in the training data. It is also worth mentioning that computing the Hausdorff distance on the training data took around 6 hours while our training process took around 3 minutes in total, on the same machine.

	Ours	**MIMLSVM**
MAE Pointing'04 Yaw	$7.94°$	$10.72°$
MAE Pointing'04 Pitch	$9.35°$	$12.32°$
MAE Multi-PIE Yaw	$6.06°$	$5.40°$

Table 3. Comparison between MIMLSVM and our kPLS method when dealing with misalignment

5. Conclusions

In summary, we presented a PLS-based regression method for head pose estimation that significantly reduces sensitivity to misalignment. The method outperforms state-of-the-art methods while simultaneously dealing with misaligned faces.

6. Acknowledgments

This work has been supported by the Spanish Research Programs: Consolider-Ingenio2010 MIPRCV (CSD200700018), Avanza I+D ViCoMo (TSI-020400-2009-133), and DiCoMa (TSI-020400-2011-55); along with the Spanish projects TIN2009-14501-C02-01 and TIN2009-14501-C02-02. Also, this research was partially supported by the ONR MURI grant N00014-08-10638.

Murad Al Haj acknowledges the support of AGAUR, Generalitat de Catalunya, through FI scholarship (2008 FIB 01165) and BE mobility grant (2010 BE1 00302) and thanks Radu Dondera and Dr. Vlad Morariu for the insightful discussions at UMD.

References

[1] B. Babenko, P. Dollár, Z. Tu, and S. Belongie. Simultaneous learning and alignment: Multi-instance and multi-pose learning. In *ECCV 2008: Faces in Real-Life Images*, October 2008.

[2] R. Dondera and L. S. Davis. Kernel PLS regression for robust monocular pose estimation. In *CVPR 2011 workshop on Machine Learning for Vision-based Motion Analysis (MLVMA'11)*, pages 24–30, 2011.

[3] P. Geladi and B. Kowalski. Partial least-squares regression: A tutorial. *Analytica Chimica Acta*, 185(1):1–17, 1986.

[4] N. Gourier, D. Hall, and J. L. Crowley. Estimating face orientation from robust detection of salient facial features. In *Proceedings of Pointing 2004, ICPR, International Workshop on Visual Observation of Deictic Gestures*, 2004.

[5] N. Gourier, J. Maisonnasse, D. Hall, and J. L. Crowley. Head pose estimation on low resolution images. In *CLEAR Workshop, In Conjunction with Face and Gesture*, April 2006.

[6] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010.

[7] G. Guo and G. Mu. Simultaneous dimensionality reduction and human age estimation via kernel partial least squares regression. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 657 –664, june 2011.

[8] D. Huang, M. Storer, F. De la Torre, and H. Bischof. Supervised local subspace learning for continuous head pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[9] G. B. Huang, V. Jain, and E. Learned-Miller. Unsupervised joint alignment of complex images. In *IEEE 11th International Conference on Computer Vision (ICCV)*, 2007.

[10] S. Langton, H. Honeyman, and E. Tessler. The influence of head contour and nose angle on the perception of eye-gaze direction. *Percept. Psychophys.*, 66(5):752–771, 2004.

[11] P. Lewi. Pattern recognition, reflections from a chemometric point of view. *Chemometrics and Intelligent Laboratory Systems*, 28:23–33, 1995.

[12] Y. Li, S. Gong, and H. Liddell. Support vector regression and classification based multi-view face detection and recognition. In *Proc. IEEE Int. Conf. Autom. Face Gesture Recog.*, pages 300–305, 2000.

[13] Z. Li, Y. Fu, J. Yuan, T. S. Huang, and Y. Wu. Query driven localized linear discriminant models for head pose estimation. In *ICME'07*, pages 1810–1813, 2007.

[14] Z. Lin, G. Hua, and L. S. Davis. Multiple Instance Feature for Robust Part-based Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[15] E. Murphy-Chutorian and M. M. Trivedi. Head pose estimation in computer vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:607–626, 2009.

[16] E. Murphy-Chutorian and M. M. Trivedi. Head pose estimation and augmented reality tracking: An integrated system and evaluation for monitoring driver awareness. *IEEE Transactions on Intelligent Transportation Systems*, 11:300–311, 2010.

[17] J. Ng and S. Gong. Composite support vector machines for detection of faces across views and pose estimation. *Image and Vision Computing*, 20(5–6):359–368, 2002.

[18] R. Rae and H. Ritter. Recognition of human head orientation based on artificial neural networks. *IEEE Transactions on Neural Networks*, 9(2):257–265, 1998.

[19] B. Raytchev, I. Yoda, and K. Sakaue. Head pose estimation by nonlinear manifold learning. In *IEEE Conference on Pattern Recognition (ICPR)*, 2004.

[20] R. Rosipal and L. J. Trejo. Kernel partial least squares regression in reproducing kernel hilbert space. *Journal of Machine Learning Research*, 2:97–123, 2001.

[21] A. Sharma and D. W. Jacobs. Bypassing synthesis: PLS for face recognition with pose, low-resolution and sketch. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, june 2011.

[22] R. Stiefelhagen. Estimating head pose with neural networks - results on the pointing04 icpr workshop evaluation data. In *Proc. Pointing 2004 Workshop: Visual Observation of Deictic Gestures*, 2004.

[23] R. Stiefelhagen, J. Yang, and A. Waibel. Modeling focus of attention for meeting indexing based on multiple cues. *IEEE Transactions on Neural Networks*, 13(4):928–938, 2002.

[24] J. Tu, Y. Fu, Y. Hu, and T. S. Huang. Evaluation of head pose estimation for studio data. In *CLEAR Workshop, In Conjunction with Face and Gesture*, pages 281–290, April 2006.

[25] M. Voit, K. Nickel, and R. Stiefelhagen. Neural network-based head pose estimation and multi-view fusion. In *CLEAR Workshop, In Conjunction with Face and Gesture*, April 2006.

[26] J. Wang and E. Sung. EM enhancement of 3d head pose estimated by point at infinity. *Image and Vision Computing*, 25(12):1864–1874, 2007.

[27] P. Wang, L. C. Tran, and Q. Ji. Improving face recognition by online image alignment. *International Conference on Pattern Recognition (ICPR2006)*, pages 311–314, 2006.

[28] H. Wold. Soft modeling by latent variables; the nonlinear iterative partial least squares approach. *Perspectives in Probability and Statistics. Papers in Honour of M. S. Bartlett*, 1975.

[29] Z. Zhang, Y. Hu, M. Liu, and T. Huang. Head pose estimation in seminar room using multi view face detectors. In *CLEAR Workshop, In Conjunction with Face and Gesture*, April 2006.

[30] Z. Zhou and M. Zhang. Multi-instance multilabel learning with application to scene classification. In *In Advances in Neural Information Processing Systems (NIPS)*, 2007.