Class Generative Models based on Feature Regression for Pose Estimation of Object Categories

Michele Fenzi, Laura Leal-Taixé, Bodo Rosenhahn, Jörn Ostermann
Institute for Information Processing (TNT), Leibniz University Hannover, Germany
fenzi,leal,rosenhahn,ostermann@tnt.uni-hannover.de

Abstract

In this paper, we propose a method for learning a class representation that can return a continuous value for the pose of an unknown class instance using only 2D data and weak 3D labelling information. Our method is based on generative feature models, i.e.*, regression functions learnt from local descriptors of the same patch collected under different viewpoints. The individual generative models are then clustered in order to create class generative models which form the class representation. At run-time, the pose of the query image is estimated in a maximum a posteriori fashion by combining the regression functions belonging to the matching clusters. We evaluate our approach on the EPFL car dataset [17] and the Pointing'04 face dataset [8]. Experimental results show that our method outperforms by 10% the state-of-the-art in the first dataset and by 9% in the second.*

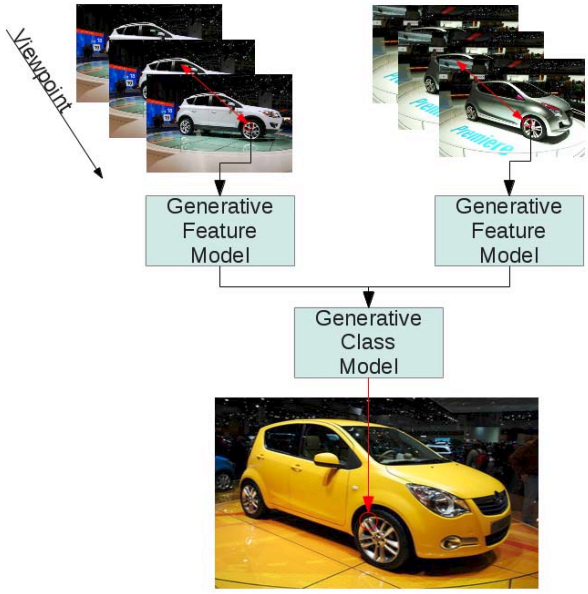

Figure 1. Generative feature models are learnt from different views of the same patch and clustered in appearance and pose space to form class generative models. Class generative models are then used to explain local parts of a test object. This shows to perform better than global approaches combining whole class instances.

1. Introduction

In the last few years, the computer vision community has seen a large increase in the interest dedicated to the problem of pose estimation for object classes. While pose estimation for individual objects has been addressed by a plethora of techniques [19, 11, 7], their extension to object classes is not straightforward. Intrinsic difficulties due to the nature of the problem, including the variability in both the appearance and the geometry of the class instances, makes the task hard to solve. The main challenge lies in designing a representation that is discriminative to differentiate among classes, yet general enough to cope with intra-class variability.

Most of the approaches proposed so far are based on local features extracted from multiple views. Local features show to generalize well across instances of the same class, as analogous local structures are repeatedly detected and similarly described. The previous works can be distinctively split into two groups: those that are based on an explicit 3D class model, either synthetic [12, 18] or reconstructed *ad hoc* [25, 6], and those that favour strategies based on solely 2D data [21, 22] or on 2D data combined with weak 3D annotation [17, 24]. However, the price to pay for avoiding an explicit 3D class representation is very often an estimation of the pose limited to a few discrete viewpoints.

The method we propose bridges this gap, as it returns a continuous value for the pose of an unknown class instance by using only 2D data combined to weak viewpoint labelling without using an explicit 3D model. Furthermore, we give a probabilistic formulation of our approach as a *maximum a posteriori* (MAP) estimation.

1.1. Contribution and Motivation

Contribution: First, we propose a way to learn a class representation based on feature generative models derived

from training class instances. Each model is based on a regression function learnt from descriptors of the same patch, and is designed to yield an estimate of the patch descriptor given a query pose.

To this end, we also advocate the use of local features but we propose to intentionally exploit one of their seeming weaknesses. That is, the slightly different description of the same patch when it undergoes geometric transformations to which the description method is not invariant. We show that the variation is smooth in the descriptor space as function of the pose change, and we demonstrate how to use this information profitably.

Finally, we propose a method to group the individual generative models by combining dynamic time warping and spectral clustering. Clustering is performed by taking into account reciprocal similarity both in appearance and viewpoint. The clusters are thus conceived as class generative models and used to provide a continuous MAP estimation for the pose of the query instance.

Motivation: Since pose estimation is ultimately a continuous problem, any method pursuing only a discrete pose estimation (*i.e.*, pose classification) is inherently inaccurate. Furthermore, the classifying methods proposed so far achieve high detection rates only for coarse quantizations. Our motivation to use regression in order to estimate a continuous value is thus a natural choice.

While our method exploits regression on local patches, [24] applies it to whole images as atomic units. In this regard, the motivation for our contribution is that a class instance is best explained as a combination of individual parts appearing separately on different instances, rather than a weighted combination of whole instances. This is represented in Figure 1. As shown in the experimental section, our method obtains more than 10% improvement in terms of pose accuracy with respect to [24].

In the next section, a review of related works is given. Section 3 introduces the generative feature model, while Section 4 provides a detailed description of our MAP method for class pose estimation. In Section 5, we present experimental results on two public datasets of cars and faces, and we give our final conclusion in Section 6.

2. Related Works

Feature-based pose estimation using multiple views can be targeted towards individual objects and object classes. In the individual case, the paradigm outlined in [19, 7] envisages first the construction of an explicit 3D object model from the training views. This is followed by a matching step between the query image and the model features, and the estimation of the object pose through the solution of the PnP problem. In the case of object classes, works can be split into two distinct groups: those that share this paradigm by employing a full 3D class model, and those relying only on 2D information or on combining 2D data with weak viewpoint annotation.

Among those using an explicit 3D model, [13, 12, 18] exploit synthetic CAD models to refine rough pose hypotheses determined by the voting of a bank of global and part-based SVM classifiers trained on appearance patches. Other works opt for a full 3D sparse reconstruction of each class instance through standard Structure for Motion techniques [25, 6]. In [25], the 3D models are separately matched to the test image and used to vote, via projection, in the image space, where the most prominent points are used for estimating the pose. In [6], the models are fused together and voting is performed directly in the pose space.

Among those using only 2D information, Savarese et al. [21, 22] propose a method that, by grouping spatially close 2D features over the training images, finds and links canonical object parts among different class instances by their mutual homographies. Their approach performs pose estimation only in terms of finding the most similar canonical view within the training set. The method proposed by [17] returns only a quantized pose value, as 16 bins are used to represent the one-dimensional pose space. In their method, a SVM classifier is trained for each discrete pose with spatial pyramids of histograms based on clustered dense features. Another approach still sharing a quantized output is provided by the so-called pose-indexed features [5, 1]. An Adaboost-based method is trained with features (edge ratios) that depend both on the image content and the candidate quantized pose. Pose estimation for object categories has also been addressed for video sequences [16]. Training sequences are split into short video segments represented by a collection of grouped patches which are characterized by their estimated appearance-geometry pdf. The quantized output pose is estimated by computing a joint pdf distance between test video parts and the training pdf's of different instances of the same class.

While all previous approaches return discrete pose estimates, two works which provide continuous values are [24, 9]. In [9], viewpoint classification is extended so that a continuous pose can be estimated by finding the maximum of a score function with respect to linearly deformed viewpoint templates. The authors in [24] employ regression on whole images as atomic units, whereas we perform regression on small local patches, as motivated in the contribution section. All image features are projected on a smaller dimensional manifold, driving the projection with feature appearance and location. Since the authors claim features to be view-point independent, the pose-dependent component is thus the feature location. This requires to have a uniformly scaled dataset, whereas our method is free from any scale dependency.

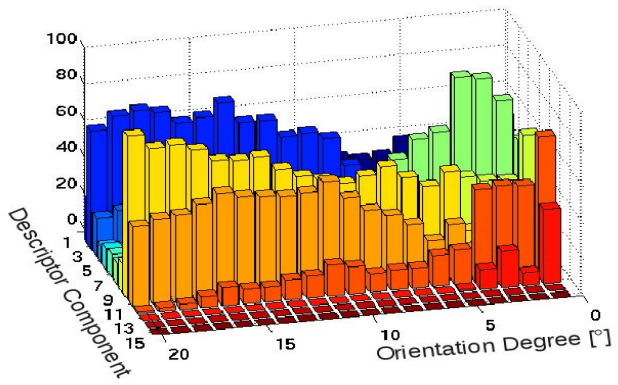

Figure 2. First 15 components (in different colors) of the feature descriptor of the same patch undergoing a 20°-wide rotational motion. The variation in the component amplitude as function of the pose change shows to be smooth. Figure best viewed in color.

3. Generative Feature Model

In this section, we introduce our first two contributions that form the building block of our approach, the so-called generative feature model. We were inspired by the work of [23], where regression is applied to grey-valued patches for robot localization. We use local feature descriptors, instead. The feature descriptor behaviour is modelled by means of a regression function learnt from training patches, as function of the object pose. Once learnt, a prediction of the patch appearance given a query pose can be obtained.

Let t_i be a feature track, *i.e.*, a set of n features all representing the same 3D planar patch from different views. Since the features are collected from pose-labelled training images, t_i is actually a set of feature-pose pairs, *i.e.*,

$$t^i = \{(f_1^i, \alpha_1^i), (f_2^i, \alpha_2^i), \ldots, (f_n^i, \alpha_n^i)\}, \tag{1}$$

where f_j^i is the k-dimensional descriptor representing the neighbourhood of the patch in pose α_j^i. As an example, Figure 2 shows how smoothly feature descriptors change along with the camera motion. So, it is possible to design a vector-valued function, our feature generative model, $F^i : \mathbb{R}^m \rightarrow \mathbb{R}^k$, where m is the dimensionality of the pose space, as a mapping between pose and feature descriptor space. Our goal is to learn such a function for each training track t^i in order to have a descriptor estimate $\hat{f}^i = F^i(\alpha)$ at pose α. We model this function as a linear combination of Gaussian kernels centred at each training pose,

$$\hat{f}^i = F^i(\alpha) = \sum_{j=1}^{n} w_j^i G(\alpha, \alpha_j^i) \tag{2}$$

where w_j^i are k-dimensional coefficients estimated from t^i during learning, and $G : \mathbb{R}^m \times \mathbb{R}^m \rightarrow \mathbb{R}$ is an exponential function,

$$G(\alpha, \alpha_j^i) = \exp\left(-\frac{d_p(\alpha, \alpha_j^i)^2}{2\sigma^2}\right) \tag{3}$$

where $d_p(\cdot, \cdot)$ is a pose distance metric.

According to the regularization theory, the optimal values for w_j^i can be obtained as the solution of the regularized linear least-square problem

$$(\mathbf{G}^i + \lambda \mathbf{I})\mathbf{W}^i = \mathbf{Z}^i \tag{4}$$

where $\mathbf{G}^i$ is a $n \times n$ matrix such that $\mathbf{G}_{lm}^i = G(\alpha_l^i, \alpha_m^i)$, $\mathbf{I}$ is the identity matrix, $\mathbf{W}^i$ and $\mathbf{Z}^i$ are $n \times k$ matrices containing the unknown coefficients and the feature descriptors of the track t^i stacked in row order, respectively. Instead of using a plain least-square approach, a regularization term must be introduced, as the training track can be affected by camera noise, imperfect pose labelling, and outliers.

4. Class Pose Estimation

In this section, we introduce the final contribution of our paper. Briefly, we first collect all generative models learnt from the training instances and we cluster them on the basis of their similarity both in descriptor and pose space. By doing so, we can separate tracks having similar appearance but inconsistent viewpoint. During run-time, the query features are set in correspondence with the clusters and the instance pose is estimated in a Bayesian fashion as the one maximizing the joint MAP probability of the observation of the query features.

This is motivated and experimentally verified by the assumption that as descriptors of analogous local structures generalize well across class instances, then also the modelling of their behaviour should. Since a class structure can be explained as a combination of its individual realizations, we can explain the class structure behaviour as a weighted combination of local patch behaviours, *i.e.*, as a weighted combination of regression functions.

4.1. Class Generative Feature Models

Given a set of N feature tracks $\mathcal{T} = \{t^i\}_{i=1}^N$ collected from different training instances, where each track is defined as in (1), the first step to build a class representation is to cluster the tracks. As we want to cluster feature tracks that are similar in both the appearance and in the viewpoint space, we have first designed a similarity score for pairs of tracks based on dynamic time warping. Dynamic time warping is a well-known technique from audio processing that allows to align two sequences given some restrictions and to have a measure of the alignment cost [20]. In our case, we impose two restrictions: (a) Tracks not overlapping in the pose space are assigned a null similarity score;

(b) The alignment must be monotonic, *i.e.*, the alignment cannot turn back on itself, as matching indices either stay the same or increase.

We use a dynamic programming implementation for our track similarity score computation. In this implementation, a cost accumulation matrix C is built in a way that each entry is the sum of the current cost and the minimum among its three adjacent past neighbouring costs, *i.e.*,

$$C(i,j) = G(\alpha_i^1, \alpha_j^2) \cdot \|f_i^1 - f_j^2\|_2 + \min(C(i-1,j), C(i,j-1), C(i-1,j-1)), \tag{5}$$

where $\|\cdot\|_2$ is the L2-norm, f_i^1 and f_j^2 are the i-th and j-th feature in track t^1 and t^2, respectively. The cost definition takes into account both the distance in the descriptor and in the pose space. The cost of aligning the two tracks is given by the last entry of the accumulation matrix C_{mn}, and we define the similarity score of t^1 and t^2 as $S_{12} = \exp(-C_{mn})$.

Given the similarity score for each pair of tracks, a square similarity matrix S can be built as follows

$$S = \begin{bmatrix} S_{11} & \cdots & S_{1N} \\ \vdots & \ddots & \vdots \\ S_{N1} & \cdots & S_{NN} \end{bmatrix}, \tag{6}$$

and clustering performed therewith. Among clustering techniques, spectral clustering algorithms have been shown to perform better than other traditional methods [3]. In order to determine k clusters with spectral clustering, we first find the k smallest eigenvectors of the Laplacian matrix L obtained from S,

$$L = I - D^{-1/2} S D^{-1/2} \tag{7}$$

where D is a diagonal matrix with $D_{ii} = \sum_{j=1}^{N} S_{ij}$. The k eigenvalues of L $\{\mathbf{v}\}_{i=1}^{k}$ are stacked in a $N \times k$ matrix V in column order, *i.e.*, $V = \begin{bmatrix}\mathbf{v}_1 \cdots \mathbf{v}_k\end{bmatrix}$. The matrix V is then normalized such that

$$U_{ij} = \frac{V_{ij}}{\sqrt{\sum_{r=1}^{k} V_{ir}^2}}, \tag{8}$$

and the N rows of U are finally clustered using k-means. A good choice for k is given by the total number of tracks N divided by the number of training exemplars.

Despite its better performance, spectral clustering can suffer from scalability problems when the similarity matrix is large and dense, as argued by [3]. However, in our case, even if N can be in the order of tenths of thousands, the matrix is very sparse because most of the elements are null as their viewpoints do not overlap.

4.2. Pose Estimation in a Bayesian Framework

If the feature descriptor f is found in the current image, we can define a probability function $p(\alpha, c|f)$ that expresses the likelihood of f being observed from pose α and matching to cluster c. This probability can be formulated with Bayes' Rule as

$$p(\alpha, c|f) = \frac{p(f|\alpha, c)p(\alpha, c)}{p(f)}. \tag{9}$$

The maximum a posteriori object pose α^* and match c^* are thus the ones maximizing this probability, *i.e.*

$$(\alpha^*, c^*) = \arg\max_{(\alpha,c)} p(\alpha, c|f) = \arg\max_{(\alpha,c)} \frac{p(f|\alpha, c)p(\alpha, c)}{p(f)}. \tag{10}$$

Since performing maximization over pose and matches can be computationally infeasible at run-time, we first match the test features against the cluster models. Matching is performed by finding the cluster center that is the nearest neighbour in the descriptor space to the query feature. Therefore, the MAP estimation reduces to

$$\alpha^* = \arg\max_{\alpha} \frac{p(f, c|\alpha)p(\alpha)}{p(f, c)} = \arg\max_{\alpha} p(f, c|\alpha)p(\alpha). \tag{11}$$

as $p(f, c)$ is independent from α due to matching. The prior $p(\alpha)$ can be defined, for example, given a SVM pose classifier, as uniformly distributed over the classifier output bin and null elsewhere. The conditional term $p(f, c|\alpha)$ can be expressed in terms of our generative feature model. We use the regressors of the tracks contained in each cluster to provide an estimation of the likelihood of the feature descriptor f, already matching with cluster c, being observed from pose α. This probability can be expressed as $p(f, c|\alpha) = p(f|c, \alpha)p(c|\alpha)$. The first term evaluates appearance similarity and the second term pose consistency.

Let $e^i = f - F^i(\alpha)$ be the error between the query feature and the descriptor estimated by the i-th regressor of cluster c. Then, the observation likelihood is defined as

$$p(f, c|\alpha) = \sum_{i:t^i \in c} \underbrace{\frac{W_i}{M} \exp(-\frac{1}{2}(e^i)^T R_i^{-1} e^i)}_{p(f|c,\alpha)} \cdot \underbrace{G(\alpha, \beta_i)}_{p(c|\alpha)}, \tag{12}$$

where $M$is a normalization constant; $W_i = \min_j \|f - f_j^i\|_2$, so that cluster tracks closer in descriptor space contribute more; $R_i = \frac{1}{n}\sum_{j=1}^{n} e_{ij} e_{ij}^T$ is the error covariance matrix of the regressor estimated by leave-one-out-cross-validation from the training samples; $\beta_i = \arg\min_j d_p(\alpha, \alpha_j^i)$, where d_p is defined in Section 3.

The extension to a set of features $\mathcal{F} = \{f_r\}_{r=1}^{R}$ extracted from the query image and the set of cluster matches $\mathcal{C}$ is

straightforward. By taking all query features into account and assuming stochastic independence, we have

$$p(\alpha|\mathcal{F},\mathcal{C}) = \frac{p(\mathcal{F},\mathcal{C}|\alpha)p(\alpha)}{p(\mathcal{F},\mathcal{C})} \propto \prod_r p(f_r, c_r|\alpha)p(\alpha). \quad (13)$$

However, this formulation is prone to cancellation in the presence of few outliers. Thus, we approximate it with a mixture model where each feature contributes equally,

$$p(\mathcal{F},\mathcal{C}|\alpha) \approx \frac{1}{R}\sum_{r=1}^{R} p(f_r, c_r|\alpha), \quad (14)$$

where R is the number of extracted features. Then, our method yields the MAP pose approximation as

$$\alpha^* = \arg\max_\alpha p(\alpha|\mathcal{F},\mathcal{C}) \approx \arg\max_\alpha \sum_r p(f_r, c_r|\alpha)p(\alpha). \quad (15)$$

5. Experiments

We provide here the experimental results of our method. We show that our algorithm performs better than the state of the art in estimating the pose of an unknown instance given the class representation built as above. For testing our method, we have considered two datasets: the EPFL multi-view car dataset [17] and the Pointing'04 face dataset [8]. The first dataset provides pose-labelled sequences of cars rotating on a floor pedestal. The sequences are densely sampled in the pose space, but many cars are uncommonly shaped, making the model generalization very challenging. The second dataset contains human faces with variable yaw and pitch values, it is sparsely sampled but with less variability in the exemplars. Even if the datasets present a 1D and 2D varying pose, respectively, the application of our method to 3-dimensional poses is straightforward.

Before showing the performance of our algorithm for class pose estimation, we also demonstrate that it can be also applied to the single exemplar case. To this end, we learn a model from a subset of training images of one instance and we evaluate the pose estimation performance on the remaining pictures.

5.1. Pose Estimation in the single case

In this section, we provide the results for the single exemplar case. We took the first 10 car sequences from the EPFL dataset and, for each, we learn a model using a 33% split, *i.e.*, one third of the images for learning and the rest for testing. Each sequence is different in length, ranging from 60 to 140 images, and with a sampling rate from 2.5° to 6°, approximately.

First, we extract local features from the training images and we form a set of model tracks $\mathcal{T}$ by tracking the features over the training views. Matches are verified and filtered through epipolar geometry with a very low threshold in order to reduce the number of outliers to a minimum. For each track, a regression function is learnt as described in Section 3. At run-time, a set of features $\mathcal{F}$ is extracted from the query image and matched against the set of model track representatives defined as the features corresponding to the middle viewpoint of the track. Finally, we return the maximum a posteriori estimate of the pose by using (15). In this single exemplar case, the set of clusters $\mathcal{C}$ corresponds to the set of model tracks $\mathcal{T}$, as only one sequence is considered. The pose prior is assumed to be uniformly distributed over the entire pose space.

We evaluated the performance of our method by using two different feature descriptors, SIFT [15] and SURF [2]. In Figure 3, we present a graph that shows the mean absolute error (MAE) that our method obtains. The MAE is defined as the average of the absolute difference between the returned value and the ground truth calculated on the test images of each sequence.

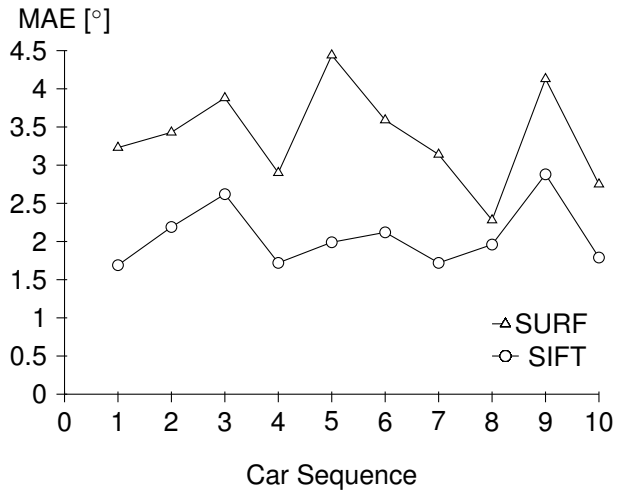

Figure 3. Single exemplar case

By using SIFT features, our method achieves an average MAE over the whole set of sequences of 2.06°. SURF features provide a less accurate estimation with a corresponding MAE of 3.37°, as the description of the patch gradient distribution is coarser. The results of our method can be considered very accurate estimations, if we take into account that the spacing between each training sample ranges from 7.5° to 18°. In comparison to [24], where only the result for the first sequence is provided, we improve the pose accuracy by almost 10% ($\mathbf{1.69}^\circ$ vs. $\mathbf{1.84}^\circ$) using the same amount of training images.

5.2. Pose Estimation in the Class case

In the following, we provide the experimental results of our method in the class case for two datasets, the EPFL Multi-view car dataset and the Pointing'04 face dataset.

Method	MAE [°] 90% percentile	MAE [°] 95% percentile	MAE [°]
Ozuysal et al. [17] (Baseline)	-	-	46.48
Torki et al. [24] - 50% split	19.4	26.7	33.98
5NN Track - 50% split	15.33	23.64	32.08
Regression 5NN - 50% split	15.17	23.49	31.93
Our Full Method - 50% split	**14.51**	**22.83**	**31.27**
Torki et al. [24] - Leave One Out split	23.13	26.85	34.90
5NN Track - Leave One Out split	15.17	23.48	31.92
Regression 5NN - Leave One Out split	15.10	23.42	31.85
Our Full Method - Leave One Out split	**14.41**	**22.72**	**31.16**

Table 1. Pose Estimation on the EPFL dataset. Comparison among full and partial implementations of our method, [17] and [24].

5.2.1 EPFL Multi-view car dataset

With regard to the car dataset, we have used the same testing framework as [17, 24] in order to compare our results with theirs. We evaluated our method with two different training/testing split set-ups:

- Training is performed on the first 10 sequences and testing on the remaining 10 (50% split), as [17, 24].
- Training is performed on 19 sequences in turn and testing on the remaining one (Leave One Out), as [24].

For both set-ups, we used only SIFT features because they provide a higher accuracy as demonstrated in the previous section. As in the single exemplar case, SIFT features are extracted from the training images, model tracks are collected by tracking the features over the training views and a regressor function is learnt for each track. Model tracks are then grouped in a set of clusters $\mathcal{C}$ as described in Section 4. At run-time, a pose interval for the query image is first estimated using the approach by [14]. It is an improved version of the Deformable Part Models (DPMs) introduced by [4] and it returns a discrete pose estimate over 4, 8 or 16 bins. For this experiment, we have used its 16-bin implementation, thus receiving a pose interval estimation of size 22.5°. Then, a set of features $\mathcal{F}$ is extracted from the query image and matched against the cluster representatives defined as the cluster centres. Eventually, the maximum a posteriori estimate of the pose is computed by using (15), where the pose prior is defined as uniformly distributed over the interval returned by the classifier and null elsewhere.

We considered [17] as baseline method because they introduced the dataset, and [24] as state-of-the-art method because they also employ a regression-based pose estimation. On our side, we provide an evaluation of our full method against two partial implementations of it and against the aforementioned works. The variants were chosen to highlight the beneficial contribution of each part of our method. The results of each variant of our method and of [17] and [24] are compared in Table 1.

Evaluated implementations:

5NN Track We find the 5 nearest neighbours among the model track representatives for each query feature. The returned pose is the mode calculated on the training viewpoints of the feature in each matching track which is closest to the query feature in descriptor space. This is a naive implementation providing results only in terms of the viewpoint of the five nearest neighbour tracks. Five was chosen as it represents the most frequent cluster size in the full method implementation. (No regression and no track clustering)

Regression 5NN We find again the 5 nearest neighbours among the track representatives for each query feature. The pose is estimated by using the track regression functions as described in the single exemplar case. This second variant shows the benefit of introducing regression for pose estimation. With respect to the naive 5NN variant, it shows that regression improves pose accuracy. (No track clustering)

Full Method We find the nearest neighbour cluster centre for each query feature. The maximum a posteriori pose is estimated as described in Section 4. It shows a further increase in the resulting pose accuracy.

In analysing Table 1, a common issue also present in [17] and [24] must be taken into account, as it strongly affects what seems to be a large error in the performance of all methods and the apparent small improvement of our contribution. In certain views, mainly the side views, the depicted cars show an almost perfect symmetry. This potentially generates 180° errors which strongly affect the final average. The influence of this flipping is less visible when the results are given in terms of their 90% and 95% percentile. In the corresponding two columns in Table 1, our method shows to perform better than the state-of-the-art, with an improvement given by the full method in the order of 25%. Furthermore, even if the 16-bin classifier we used has a worse performance than [24] (66% vs. 70.31%

detection rate for 16 bins), we are still able to obtain smaller errors. Finally, our modelling does not seem to suffer from over-fitting problems, like [24]. It must be noted that even our naive 5NN implementation is able to outperform them. This can be taken as a further confirmation of the superiority of local approaches for pose estimation over global ones.

For a deeper insight into the performance of our method, we give in Table 2 the results of a side experiment performed using the ground-truth bin instead of the actual classifier output. The naive implementation performs only slightly worse than the regression variant as the training images are very finely sampled in pose. On the contrary, the improvement in the accuracy given by our full method with respect to the naive and the only-regression variants comes out more evidently with a 20% and 15% increase, respectively. The better performance compared to the only-regression variant is due to the use of consistent clusters having appropriate size, instead of k nearest neighbours. In Figure 4, we provide visual examples of the performance of our method and of the appearance variability of the test exemplars.

Method	MAE [°]
5NN Track - 50% split	6.87
Regression 5NN - 50% split	6.64
Our Full Method - 50% split	**5.64**
5NN Track - Leave One Out split	6.63
Regression 5NN - Leave One Out split	6.53
Our Full Method - Leave One Out split	**5.48**

Table 2. Pose estimation using a ground-truthed classifier

5.2.2 Pointing'04 face dataset

The dataset is composed of 2790 face images whose pose is discretized in 9 and 13 angles of pitch and yaw, respectively. We choose this dataset as the pose sampling is coarser, with a minimum angular distances of 15° in both yaw and pitch.

We used the same testing framework as [8, 10] in order to compare our results with theirs. Evaluation is performed with 5-level cross validation, where each sample is in turn tested using a model trained on 80% of the remaining samples. For our method, we have evaluated three variants with a similar spirit and implementation as in the previous experiment. This time, we used only the first nearest neighbour (1NN) because samples of the same person are found both in training and testing. We report the results in Table 3.

Our method shows to have a better or comparable performance with respect to the state of the art on this dataset. The slightly worse performance in the pitch estimation for all methods is due to the coarser sampling of the pitch poses. Unlike the previous experiment, the benefit of using feature regression comes out more evidently when the pose sampling is coarser with an increase in pose accuracy of 42%. Again, the full method outperforms its variants by 57% and 25%, respectively. We can think that each class generative model explains the query pose by combining the descriptors of the same person in neighbouring poses and the descriptors belonging to other persons, potentially available at the same viewpoint of the query image. In Figure 5, we provide visual examples of the performance of our method on the face dataset.

Method	Yaw MAE [°]	Pitch MAE [°]
Gourier et al. [8]	10.1	15.9
Haj et al. [10]	6.56	**6.61**
1NN Track	13.82	19.27
Regression 1NN	7.89	9.43
Our Full Method	**5.94**	6.73

Table 3. Pose estimation on the Pointing'04 dataset

6. Conclusion

We proposed a novel method for learning a class representation in order to provide a continuous estimate for the pose of an unknown class instance. Our method is based on generative feature models, *i.e.*, regression functions learnt from features computed on different views of the same patch. Each model is designed so to predict the appearance of a local patch as a function of the viewpoint. By clustering individual models in the appearance and pose space, we obtain a set of consistent class generative models that form the class representation. The pose of a query instance is given as a maximum a posteriori estimation by using the matching class generative models.

Experiments show that our method provides a pose that is 10% more accurate than state-of-the-art methods, as a local-based approach explains a query instance better than a global one. By evaluating our method on two different and challenging class datasets, we show that our approach is general and can be applied to different object classes.

Acknowledgements Research was conducted inside the project ASEV (n. 13N10719) funded by the German Federal Ministry of Education and Research (BMBF).

References

[1] K. Ali, F. Fleuret, D. Hasler, and P. Fua. Joint Pose Estimator and Feature Learning for Object Detection. *ICCV*, 2009. 2

[2] H. Bay, T. Tuytelaars, and L. V. Gool. SURF: Speeded Up Robust Features. *ECCV*, 2006. 5

[3] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang. Parallel Spectral Clustering in Distributed Systems. *TPAMI*, 2011. 4

[4] P. F. Felzenszwalb, R. B. Girshick, and D. A. McAllester. Cascade Object Detection with Deformable Part Models. *CVPR*, 2010. 6

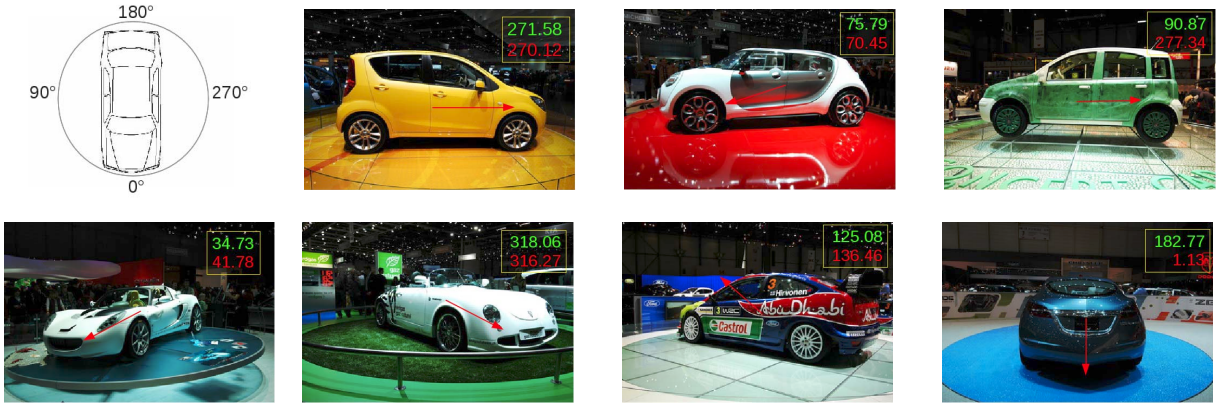

Figure 4. Results of our method on the EPFL car dataset. In the top left corner, a diagram with the car orientation. For each picture, the ground truth value (green) and the estimated one (red) are given in a box and a red arrow is overlaid on the car. In the last column, two examples of flipped estimation are provided. The pose of the car in the bottom right corner is hard to guess also for a human being. Figure best viewed in color.

Figure 5. Results of our method on the Pointing'04 face dataset. In this case, the returned pose value is two-dimensional, as it is given in terms of pitch and yaw. The ground truth pose (green) and the estimated one (red) are given in a box.

[5] F. Fleuret and D. Geman. Stationary Features and Cat Detection. *JMLR*, 2008. 2

[6] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich. Viewpoint-Aware Object Detection and Pose Estimation. *ICCV*, 2011. 1, 2

[7] I. Gordon and D. G. Lowe. What and Where: 3D Object Recognition with Accurate Pose. *Toward Category-Level Object Recognition*, 2006. 1, 2

[8] N. Gourier, D. Hall, and J. L. Crowley. Estimating Face Orientation from Robust Detection of Salient Facial Features. *ICPR*, 2004. 1, 5, 7

[9] C. Gu and X. Ren. Discriminative Mixture-of-templates for Viewpoint Classification. *ECCV*, 2010. 2

[10] M. Haj, J. Gonzalez, and L. Davis. On Partial Least Squares in Head Pose Estimation: How to Simultaneously Deal with Misalignment. *CVPR*, 2012. 7

[11] E. Hsiao, A. Collet Romea, and M. Hebert. Making Specific Features Less Discriminative to Improve Point-based 3D Object Recognition. *CVPR*, 2010. 1

[12] J. Liebelt and C. Schmid. Multi-View Object Class Detection with a 3D Geometric Model. *CVPR*, 2010. 1, 2

[13] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3D Feature Maps. *CVPR*, 2008. 2

[14] R. J. López-Sastre, T. Tuytelaars, and S. Savarese. Deformable Part Models Revisited: A Performance Evaluation for Object Category Pose Estimation. *ICCV Workshops*, 2011. 6

[15] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 2004. 5

[16] L. Mei, J. Liu, A. O. Hero, and S. Savarese. Robust Object Pose Estimation via Statistical Manifold Modeling. *ICCV*, 2011. 2

[17] M. Özuysal, V. Lepetit, and P. Fua. Pose Estimation for Category Specific Multiview Object Localization. *CVPR*, 2009. 1, 2, 5, 6

[18] B. Pepik, P. Gehler, M. Stark, and B. Schiele. 3d2pm - 3d deformable part models. *ECCV*, 2012. 1, 2

[19] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3D Object Modeling and Recognition Using Local Affine-Invariant Image Descriptors and Multi-View Spatial Constraints. *IJCV*, 2006. 1, 2

[20] H. Sakoe and S. Chiba. Dynamic Programming Algorithm Optimization for Spoken Word Recognition. *IEEE Trans. on Acoustics, Speech, and Signal Proc.*, 1978. 3

[21] S. Savarese and F.-F. Li. 3D Generic Object Categorization, Localization and Pose Estimation. *ICCV*, 2007. 1, 2

[22] S. Savarese and F.-F. Li. View Synthesis for Recognizing Unseen Poses of Object Classes. *ECCV*, 2008. 1, 2

[23] R. Sim and G. Dudek. Learning Generative Models of Scene Features. *IJCV*, 2004. 3

[24] M. Torki and A. M. Elgammal. Regression from Local Features for Viewpoint and Pose Estimation. *ICCV*, 2011. 1, 2, 5, 6, 7

[25] J. Xiao, J. Chen, D.-Y. Yeung, and L. Quan. Structuring Visual Words in 3D for Arbitrary-View Object Localization. *ECCV*, 2008. 1, 2