Pose-invariant Face Recognition with a Two-Level Dynamic Programming Algorithm

Harald Hanselmann[1], Hermann Ney[1] and Philippe Dreuw[1,2]

[1] Human Language Technology and Pattern Recognition Group
RWTH Aachen University, Aachen, Germany
`<surname>@cs.rwth-aachen.de`
[2] Robert Bosch GmbH, Hildesheim, Germany
`philippe.dreuw@de.bosch.com`

Abstract. In this paper, we propose a novel algorithm for general 2D image matching, which is known to be an NP-complete optimization problem. With our algorithm, the complexity is handled by sequentially optimizing the image columns from left to right in a two-level dynamic programming procedure. On a local level, a set of hypotheses is computed for each column, while on a global level the best sequence of these hypotheses is selected. The optimization on the local level is guided by a lookahead that gives an estimate about the not yet optimized part of the image. We evaluate the algorithm on the task of pose-invariant face recognition in an automatic setup and show that the suggested method is competitive and achieves very good recognition accuracies on the popular face recognition databases CMU-PIE and CMU-MultiPIE.

Keywords: Face recognition, Image warping, Dynamic programming

1 Introduction

The task of face recognition is very challenging for image recognition algorithms. Many inter- and intra-class variations such as lighting, facial expressions or different poses make this task difficult. Especially the recognition across varying poses is a complex problem due to the variation in a 3D space. Over the last couple of years, many different approaches to face recognition in general [1, 2] and pose-invariant face recognition in particular [3] have been proposed. Here, we focus on the technique to solve the problem by using 2D image matching (2D warping) in a nearest-neighbor framework [4]. The minimized cost of matching training and testing image (in the context of face recognition often referred to as gallery and probe image, respectively) is used as a similarity measure. By this procedure, deformations caused by the previously mentioned variations can be taken into account. It has been shown that structural constraints on the image matching lead to smoother deformations and an increased recognition accuracy [5, 6]. In order to incorporate such constraints and to apply relative penalty functions, the dependencies defined by a 2D grid need to be maintained. Consequently, the warping of a specific pixel effectively depends on its neighbors in both dimensions. Unfortunately, image matching with 2D dependencies is NP-complete [7] and therefore infeasible. Thus, most 2D warping algorithms approximate the optimal solution or optimize a relaxed criterion.

The Pseudo 2D Warping (P2DW) approach presented in [8] reduces the 2D problem to two 1D problems. On a global level all pixels within one column are combined to a super-pixel and the optimal sequence of such super-pixels is computed. On a local column level the matching of two super-pixels is optimized. This is implemented by a two-level dynamic programming algorithm. Tree-Serial Dynamic Programming (TSDP) [9] solves the problem by optimizing a set of trees. For each column one tree is constructed, where vertical dependencies represent the stem and the horizontal dependencies represent the branches. In [6] this approach is extended by using structural constraints [5] (CTSDP). The before mentioned approaches suffer from the drawback that the final solution can only maintain vertical dependencies, due to the independent optimization of the columns. Horizontal dependencies are only included implicitly.

Similar to P2DW, in [10] the problem is also divided into a global and a local level. However, in this case the local level is approximated by linear interpolation, whereas on a global level the optimization is reduced to specific pivot pixels. While this algorithm is efficient for a small number of pivot pixels, the interpolation allows only for very simplified warpings.

Other approaches are based on using an iterative inference algorithm on Markov Random Fields (MRF) such as Sequential Tree-Reweighted Message Passing (TRW-S) [11]. In [12] displacement vectors are used as labels in the MRF framework, but to reduce complexity horizontal and vertical displacements are decoupled. The matching is then computed using TRW-S in a multi-resolution setup. In contrast to this, the approach presented in [13] (CTRW-S) uses actual pixels as labels. Feasible complexity is reached by applying structural constraints as in CTSDP effectively speeding up the message updates in TRW-S. Nevertheless, the quality of the warping depends on the number of iterations which are still computationally expensive.

In this paper, we propose a novel approximative 2D warping algorithm based on dynamic programming (2LDP-LA). Unlike P2DW and TSDP/CTSDP, our algorithm is capable of fully maintaining 2D dependencies. The algorithm is not iterative and achieves high efficiency by optimizing column-wise and considering only a number of hypotheses for each column (c.f. Section 3). This optimization is done sequentially in dependence of the hypotheses computed for the previous columns. Furthermore, a lookahead is integrated that adds an estimate of the cost of not yet optimized columns (c.f. Section 3.2). Finally, we evaluate our algorithm on the popular CMU-PIE [14] and CMU-MultiPIE [15] databases (c.f. Section 4).

2 2D Warping

We start by briefly reviewing the problem formulation of 2D warping in correspondence with [5] and [6]. Given a source image $X = \{x_{ij}\}$ with resolution $I \times J$ and a target image $R = \{r_{uv}\}$ with resolution $U \times V$, we search for a mapping

$(i, j) \to w_{ij} = (u, v)$ that assigns each pixel of the source image to a pixel of the target image with minimal energy cost. The energy $E(X, R, w_{ij})$ is defined as:

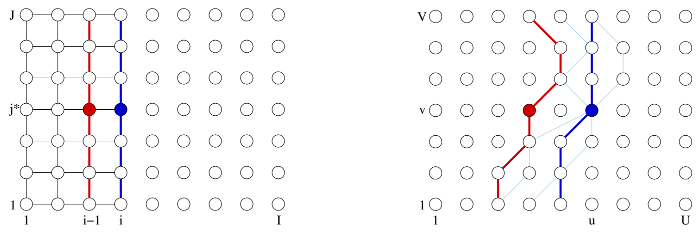
$$E(X, R, \{w_{ij}\}) = \sum_{i,j} \left[d_{ij}(w_{ij}) + T(w_{i-1,j}, w_{ij}) + T(w_{i,j-1}, w_{ij}) \right] \qquad (1)$$

The term $d_{ij}(w_{ij})$ describes the data cost of mapping (i, j) to w_{ij}, defined by a cost function such as the absolute distance. Local context can be included by computing the cost-function over a small patch centered around the pixel (i, j) and the distances are pre-computed and cached [6]. To prevent outliers from having too much influence, the distance function is truncated by a threshold τ [6]. The penalty function $T(\cdot)$ encodes the 2D dependencies by adding cost depending on the mapping of neighboring pixels. Differences in the displacement of a pixel and its horizontal and vertical neighbors are penalized leading to a smoother matching. For this we use the Euclidean distance weighted by a factor α. Additionally, structural constraints [5] are included that enforce monotony and continuity in the matching.

For classification, the nearest-neighbor rule is applied using the minimized energies as similarity measure [4]. In the case of face recognition across poses with frontal face images as gallery images, it is helpful to match the probe image with the left and right half of the gallery image instead of using the complete gallery image [12]. This is due to the fact that in profile and near profile images roughly half of the face is occluded. In [12] the half images are only used for images with much difference in yaw and the full gallery images are used otherwise. We found that this distinction is not necessary and simply matching both half-images and selecting the lower resulting energy is sufficient for all poses. As in [12] and [6] we use the gallery image as source and the probe image as target.

3 Two-Level Dynamic Programming with Lookahead

The proposed 2D warping algorithm is designed to maintain the full 2D dependencies defined in Eq. (1). Similar to Pseudo 2D Warping we divide the problem into a global and a column level, but unlike the work in [8] we do not optimize the mapping of the source columns independently, since the latter only allows to maintain vertical dependencies. Instead, we sequentially compute a set of hypotheses for each column, i.e. a set of possible paths to map the column to the target image, in dependence of the previous columns. This set is defined by a representative row j^* similar to the pivot pixels in [10], where we choose the middle row ($j^* = J/2$). As result, each column has one representative pixel and for every possible matching of this pixel the best corresponding path is computed and included in the set of hypothesis. Finally, in the global optimization, the best sequence of the hypotheses is selected as solution. An illustration of this procedure is given in Fig. 1. We will now define the algorithm in more detail.



(a) Source Image (b) Target Image

Fig. 1. Two-level DP algorithm: Column i is optimized in dependence of column $i-1$. Out of several options (blue paths in 1(b)), the optimal is selected (darker blue).

3.1 Basic Algorithm

In the following, w_1^J denotes a mapping path of a fixed column i, where (i, j^*) is mapped to w. Accordingly, $\tilde{w}_1^J$ represents a path of the predecessor column $i-1$, where $(i-1, j^*)$ is mapped to $\tilde{w}$.

$$w_1^J = \{w_{i,1}, \ldots, w_{i,j^*} = w, \ldots, w_{i,J}\} \tag{2}$$
$$\tilde{w}_1^J = \{w_{i-1,1}, \ldots, w_{i-1,j^*} = \tilde{w}, \ldots, w_{i-1,J}\} \tag{3}$$

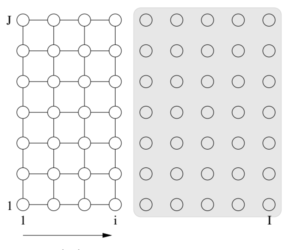
To define the algorithm we start with the dynamic programming recursion on the global level, where for each column a set of hypotheses is computed:

$$D(i, w) = \min_{\tilde{w}} \Big\{ D(i-1, \tilde{w}) + \hat{C}(i, \tilde{w}, w) \Big\}. \tag{4}$$

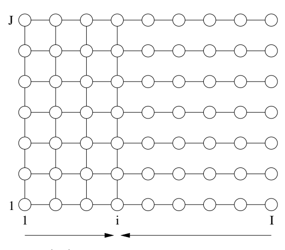
Each entry of the dynamic programming table $D(i, w)$ describes the optimal cost of matching column i with the restriction that the representative pixel (i, j^*) is mapped to $w = (u, \cdot)$. The v-component of w is left variable, indicating that the global optimization is done only over the u-component of the representative mapping. This simplification is done for complexity reasons. As defined in Eq. (4), the entries of $D(i, w)$ are computed by minimizing over the best predecessor and the local cost $\hat{C}(\cdot)$ of the current column. For the initialization $D(0, w)$ the best paths of mapping the first column to the target image are computed without horizontal dependencies. The local cost is defined by

$$\hat{C}(i, \tilde{w}, w) = \min_{w_1^J : w_{j^*} = w} \Big\{ C(i, \hat{w}_1^J(i-1, \tilde{w}), w_1^J) \Big\}, \tag{5}$$

where we optimize over the possible paths for the current column given the restriction by the mapping of the representative and the best path for the predecessor. The latter is read from $\hat{w}_1^J(i, w)$, which can be seen as data structure that stores the best paths corresponding to the entries $D(i, w)$ in the global dynamic programming table. The actual cost of mapping column i to w_1^J in dependence



(a) No lookahead



(b) With lookahead

Fig. 2. Illustration of the lookahead. Without lookahead, the grey area in (a) has no impact on the optimization of column i. By including the lookahead in (b), the optimization of i is influenced by an estimate of the cost for the rest of the image.

of the predecessor path is obtained by

$$C(i, \hat{w}_1^J(i-1, \tilde{w}), w_1^J) = \sum_j \big[d_{ij}(w_j) + T(w_{j-1}, w_j) + T(\hat{w}_j(i-1, \tilde{w}), w_j)\big]. \quad (6)$$

Here we can find the data cost and penalty functions from Eq. (1). The final energy, which is used for the nearest-neighbor classification, is found by

$$\hat{E}(X, R, \{w_{ij}\}) = \min_w \Big\{D(I, w)\Big\}. \quad (7)$$

The actual deformation of the image can be computed by using back-tracing.

3.2 Lookahead

The algorithm as introduced above is strongly influenced by the matching of the first columns. When the first columns are optimized, large parts of the image have no impact (c.f. Fig. 2(a)). The complete image is only considered in the global level when finally the best hypotheses are selected as solution. However, if the first columns are matched too greedy, the global result might suffer severely. For this reason we implement a lookahead that gives an estimate of the cost for the not yet optimized part of the image, similar to the branches during the optimization of the stems in TSDP or the lookahead techniques used in beam search for speech recognition [16]. Consequently the optimization of a column is guided by the previously computed hypotheses from the one side and the lookahead from the other side (c.f. Fig. 2(b)).

To integrate the lookahead into the two-level dynamic programming algorithm, we replace the cost function $C(\cdot)$ by

$$C_{LA}(i, \hat{w}_1^J(i-1, \tilde{w}), w_1^J) = \\ \sum_j \big[d_{ij}(w_j) + T(w_{j-1}, w_j) + T(\hat{w}_j(i-1, \tilde{w}), w_j) + D_{LA}(i, j, w_{ij})\big], \quad (8)$$

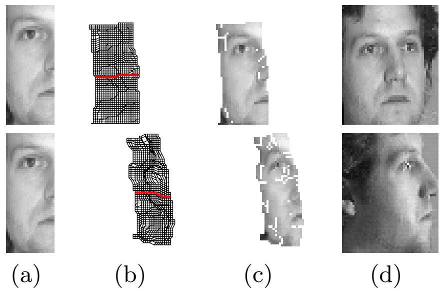


Fig. 3. Example of image matching with 2LDP-LA. The left half image (a) is matched to image (d). Image (c) is the resulting deformed source image, while in (b) the distortion grid is shown. The red line indicates the matching of the representative row.

where $D_{LA}(i, j, w_{ij})$ contains the cost estimated by the lookahead for the remaining part of row j in case (i, j) is mapped to w_{ij}. However, this new cost-function is only used in the local level during the optimization of the paths. In the global dynamic programming table the cost without lookahead is stored such that the entries $D(i, w)$ only capture the cost up to column i. To compute the lookahead table $D_{LA}(\cdot)$ we use the tree optimization approach with structural constraints (CTSDP) from [6], because the authors show that this algorithm achieves good results on the task of face recognition. Since we need a row-wise lookahead we use the rows instead of the columns as stems of the trees and thus the columns as branches (CTSDP-R).

3.3 Complexity

CTSDP as well as CTSDP-R is implemented by using one forward and one backward run of dynamic programming for the branches and one bottom-up run for the stems resulting in a complexity of $\mathcal{O}(IJUV)$ [6]. These calculations have to be done as well during 2LDP-LA with CTSDP-R as lookahead. Apart from that, the global level of 2LDP-LA only depends on the number of columns I and U and thus has a complexity of $\mathcal{O}(IU)$. On the local level for each entry in the global dynamic programming table the respective column has to be optimized leading to a complexity of $\mathcal{O}(IJU)$. An exception is defined by the first column, where no predecessor is available. Here, for each $w = (u, v)$ one path has to be optimized which has a complexity of $\mathcal{O}(JUV)$. As a result, the overall complexity of 2LDP-LA is only slightly higher than the complexity of CTSDP-R. As an example, for optimizing the mapping of a 34×86 image to a 68×86 image we measured a runtime of 16.2 seconds with CTSDP-R on a processor with 3.2 GHz. Optimizing with 2LDP-LA took only additional 0.75 seconds. It should be noted that both algorithms are well suited for parallel implementation, which would decrease the runtime significantly. Additionally, pruning techniques can be used to speed up the nearest-neighbor classification as proposed in [6].

4 Experimental Evaluation

The proposed algorithm[3] is evaluated on the CMU-PIE [14] and the more challenging CMU-MultiPIE [15] databases. Since we use CTSDP-R as lookahead for our algorithm, a direct comparison of CTSDP-R and 2LDP-LA is interesting. For this reason, we evaluate both algorithms with the same experimental setup extending the implementation provided with [6]. For both databases, we use a setup where only one image per class is used as gallery image. For this, we choose the frontal pose images, which are aligned by rotating the images such that hand-marked eye-centers are aligned horizontally. For the probe images a publicly available face detector [17] is used as rough alignment. To keep the process fully automatic, we do not assume knowledge of the poses of the probe images. Hence, we apply the left, right and frontal face detector and select the resulting bounding box with the highest confidence score. All images are resized to 204×256 as in [18], and the Tan-Triggs preprocessing [19] is applied using the authors code. As feature descriptor we use PCA-reduced U-SIFT [20, 21] descriptors with a PCA dimension of 30.[4] They are extracted on a grid with a step-size of 3 and normalized using the ℓ^1-norm [6]. By this procedure we obtain 68×86 descriptors arranged in a regular 2D grid for each image.

We roughly optimized one parameter set for our algorithm and one for CTSDP-R that work well on the most difficult poses of both databases. For 2LDP-LA we use 5×5-patches as local context, the distance threshold τ is set to 1.3 and the penalty weight α is set to 0.04. For CTSDP-R we use the same parameters except for a slightly higher distance threshold of 1.4.

4.1 CMU-PIE

The CMU-PIE [14] database contains images of 68 different subjects with varying pose, illumination and facial expressions. Here, we use a subset containing all poses with neutral illumination and facial expression (c.f. Fig. 4). The frontal pose (c27) is used for the gallery images while the remaining 12 near-frontal and near-profile poses are used for testing. This leads to a setup with 68 gallery and 816 probe images.

We can observe in Table 1 that on the near frontal as well as the near profile poses 2LDP-LA outperforms CTSDP-R. Overall, the recognition accuracy is improved from 92.2% to 94.6%. To the best of our knowledge, this is also the best overall accuracy reported on this full subset of the CMU-PIE for a system without manual alignment of the test data. Other fully automatic approaches, such as the pose normalization methods presented in [22] and [23], achieve good results on the near-frontal poses, but do not report results for the difficult near-profile poses. The differences between our results for CTSDP-R and the results in [6] can be explained mainly by the different alignment, since in [6] manual instead of automatic alignment is used. Additionally, in [6] the authors use no

[3] Implementation available at `http://www.hltpr.rwth-aachen.de/w2d`
[4] U-SIFT features are extracted using a modified version of OpenCV 2.2.0.

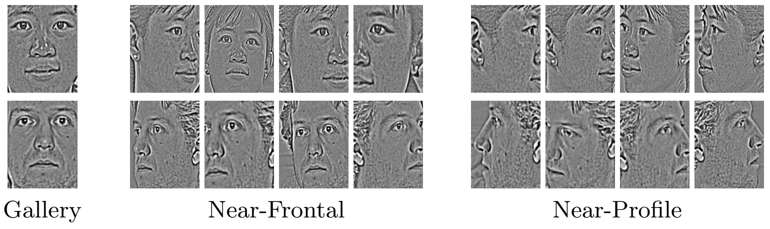


Fig. 4. Examples images used for evaluation after face detection and pre-processing. Upper row: CMU-PIE, lower row: CMU-MultiPIE.

Table 1. Recognition accuracies on the CMU-PIE database. The set of near-frontal (NF) poses contains the six poses with the lowest variation in yaw, the set of near-profile (NP) poses contains the six poses with the highest variation in yaw.

Method	Autom. align.	NF[%]	NP[%]	Average[%]
CTSDP-R	yes	96.6	87.7	92.2
2LDP-LA	yes	97.1	92.2	94.6
Multi-Resolution MRF [12]	yes	96.0	88.5	92.3
3D Pose-Norm. [22]	yes	99.0	-	-
Continuous Pose-Norm. [23]	yes	100.0	-	-
Ridge Regression [18]*	no	100.0	82.7	91.4
CTSDP [6]	no	99.8	92.7	96.2
CTRW-S [6]	no	99.5	93.6	96.6

* Numbers are estimated from a graph.

preprocessing, column-wise optimization and a different grid for the extraction of the feature descriptors.

4.2 CMU-MultiPIE

As an extension of the CMU-PIE database, the CMU-MultiPIE [15] covers more subjects, several recording sessions and more variation in pose and facial expression. We use the images from the first recording session with neutral expression and frontal lighting (c.f. Fig. 4). We choose pose 05_1 as gallery image and the remaining 14 poses for testing leading to 249 gallery and 3486 probe images.

As the results in Table 2 show, also on this database 2LDP-LA outperforms CTSDP-R, although the margin between the results is smaller than on the CMU-PIE database. Compared to the latter, the average recognition accuracy of both algorithm degrades. This is rooted mainly in the increased variation in yaw. Especially the two most difficult poses with $-90°$ and $+90°$ yaw decrease the accuracy. By excluding these two poses from the evaluation, the average accuracy rises to 96.1% for 2LDP-LA and 94.8% for CTSDP-R, respectively. Compared to

Table 2. Recognition accuracies on the CMU-MultiPIE database. The near-frontal (NF) and near-profile (NP) sets are defined analogously to Table 1, except in this case there are eight near-profile poses.

Method	Autom. align.	NF[%]	NP[%]	Average[%]
CTSDP-R	yes	99.8	72.2	84.0
2LDP-LA	yes	99.9	74.8	85.6
3D-GEM [24]	yes	81.8	-	-
Ridge Regression [18]*	no	98.3	61.6	77.3

* Numbers are estimated from a graph.

state-of-the-art approaches that use a very similar experimental setup such as 3D Generic Elastic Models [24] and Ridge Regression [18], our algorithm achieves the best recognition accuracy. Other results for pose-invariant face recognition have been reported on this database, e.g. [25, 22, 26], but the differences in the setup do not allow a quantitative comparison with our results.

5 Conclusion

In this paper we have introduced a novel algorithm for image recognition in a nearest-neighbor framework. The algorithm is capable to maintain full 2D dependencies by sequentially computing a number of hypotheses for each column defined by a representative row. This process is guided by a lookahead computed in advance as an estimate of the not yet optimized columns. While there are several options on how to compute the lookahead, we use CTSDP-R which already achieves good recognition results on its own. We evaluated our algorithm in a difficult, fully automatic setup with only one gallery image per subject. We obtained very competitive results on pose subsets of the CMU-PIE and the CMU-MultiPIE database, even compared to explicit 3D face modeling approaches and without dependence on facial landmarks. The comparison with CTSDP-R shows that in both cases our algorithm improves the recognition accuracy while increasing the complexity only marginally.

While in this paper the evaluation is done on pose-invariant face recognition, the method is also easily applicable on other image recognition tasks due to its generality and the lack of assumptions about the subjects to classify.

References

1. Zhao, W., Chellappa, R., Phillips, P., Rosenfeld, A.: Face recognition: A literature survey. Acm Computing Surveys (CSUR) **35**(4) (2003) 399–458
2. Bowyer, K.W., Chang, K., Flynn, P.: A survey of approaches and challenges in 3d and multi-modal 3d+ 2d face recognition. Computer Vision and Image Understanding **101**(1) (2006) 1–15

3. Zhang, X., Gao, Y.: Face recognition across pose: A review. Pattern Recognition **42**(11) (2009) 2876–2896
4. Keysers, D., Deselaers, T., Gollan, C., Ney, H.: Deformation models for image recognition. IEEE T-PAMI (2007) 1422–1435
5. Uchida, S., Sakoe, H.: A monotonic and continuous two-dimensional warping based on dynamic programming. In: ICPR. (1998) 521–524
6. Gass, T., Pishchulin, L., Dreuw, P., Ney, H.: Warp that smile on your face: optimal and smooth deformations for face recognition. In: FG. (2011) 456–463
7. Keysers, D., Unger, W.: Elastic image matching is NP-complete. Pattern Recognition Letters **24**(1-3) (2003) 445–453
8. Kuo, S., Agazzi, O.: Keyword spotting in poorly printed documents using pseudo 2-D hidden Markov models. IEEE T-PAMI **16**(8) (1994) 842–848
9. Mottl, V., Kopylov, A., Kostin, A., Yermakov, A., Kittler, J.: Elastic transformation of the image pixel grid for similarity based face identification. In: ICPR. (2002) 549–552
10. Uchida, S., Sakoe, H.: Piecewise linear two-dimensional warping. In: ICPR. (2000) 534–537
11. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. IEEE T-PAMI (2006) 1568–1583
12. Arashloo, S., Kittler, J., Christmas, W.: Pose-invariant face recognition by matching on multi-resolution mrfs linked by supercoupling transform. Computer Vision and Image Understanding **115**(7) (2011) 1073–1083
13. Gass, T., Dreuw, P., Ney, H.: Constrained energy minimization for matching-based image recognition. In: ICPR. (2010) 3304–3307
14. Sim, T., Baker, S., Bsat, M.: The cmu pose, illumination, and expression (pie) database. In: FG. (2002) 46–51
15. Gross, R., Matthews, I., Cohn, J., Kanade, T., Baker, S.: Multi-pie. Image and Vision Computing **28**(5) (2010) 807–813
16. Ortmanns, S., Ney, H.: Look-ahead techniques for fast beam search. Computer Speech & Language **14**(1) (2000) 15–32
17. Kalal, Z., Matas, J., Mikolajczyk, K.: Weighted sampling for large-scale boosting. BMVC (2008)
18. Li, A., Shan, S., Gao, W.: Coupled bias–variance tradeoff for cross-pose face recognition. Image Processing, IEEE Transactions on **21**(1) (2012) 305–315
19. Tan, X., Triggs, B.: Enhanced local texture feature sets for face recognition under difficult lighting conditions. AMFG (2007) 168–182
20. Ke, Y., Sukthankar, R.: Pca-sift: A more distinctive representation for local image descriptors. In: CVPR. (2004) II–506
21. Dreuw, P., Steingrube, P., Hanselmann, H., Ney, H.: Surf-face: Face recognition under viewpoint consistency constraints. In: BMVC. (2009)
22. Asthana, A., Marks, T., Jones, M., Tieu, K., Rohith, M.: Fully automatic pose-invariant face recognition via 3d pose normalization. In: ICCV. (2011) 937–944
23. Ding, L., Ding, X., Fang, C.: Continuous pose normalization for pose-robust face recognition. IEEE Signal Process. Lett. **19**(11) (2012) 721–724
24. Prabhu, U., Heo, J., Savvides, M.: Unconstrained pose-invariant face recognition using 3d generic elastic models. IEEE T-PAMI **33**(10) (2011) 1952–1961
25. Sharma, A., Haj, M.A., Choi, J., Davis, L.S., Jacobs, D.W.: Robust pose invariant face recognition using coupled latent space discriminant analysis. Computer Vision and Image Understanding **116**(11) (2012) 1095–1110
26. Heo, J., Savvides, M.: Face recognition across pose using view based active appearance models (vbaams) on cmu multi-pie dataset. In: ICVS. (2008) 527–535