Optimal Landmark Detection using Shape Models and Branch and Bound

Brian Amberg
University of Basel
brian.amberg@unibas.ch

Thomas Vetter
University of Basel
thomas.vetter@unibas.ch

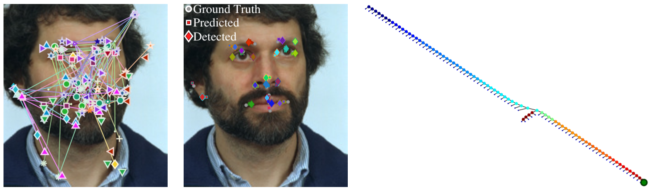
Abstract

Fitting statistical 2D and 3D shape models to images is necessary for a variety of tasks, such as video editing and face recognition. Much progress has been made on local fitting from an initial guess, but determining a close enough initial guess is still an open problem. One approach is to detect distinct landmarks in the image and initalize the model fit from these correspondences. This is difficult, because detection of landmarks based only on the local appearance is inherently ambiguous. This makes it necessary to use global shape information for the detections. We propose a method to solve the combinatorial problem of selecting out of a large number of candidate landmark detections the configuration which is best supported by a shape model. Our method, as opposed to previous approaches, always finds the globally optimal configuration.

The algorithm can be applied to a very general class of shape models and is independent of the underlying feature point detector. Its theoretic optimality is shown, and it is evaluated on a large face dataset.

1. Introduction and related work

Fitting two or three dimensional models of objects – such as faces – to images has been used to great effect in many applications, for example face recognition [8, 22, 30, 5] and video editing [10, 2, 6, 27, 25]. Statistical shape models are fitted by maximizing the posterior of the model parameters given an observed image. Even for simple models such as Active Appearance Models (AAM, [8, 17]) or 3D Morphable Models (3D-MM [3]), this posterior has a complex shape and is defined over a high dimensional space, making it impossible to find its global maximum. Instead, most algorithms are concerned with the efficient local maximization of the posterior starting from an initial guess [8, 17, 1, 28, 23]. In some applications the initial guess can be obtained from a face detector [26], but if the face is non-frontal, or a highly precise fit is required then it is necessary to start with a better initialization.



(a) Candidates (b) Optimal Selection (c) Search Tree

Figure 1. Our algorithm selects the optimal set of candidates (b) from a large number of candidate detections (a) while searching only a small fraction of the search space using a greedy search tree (c) in a branch and bound approach. The solution space (number of possible ways to select candidate points) for this example contains 2×10^{17} candidates, and the optimal one was chosen with only 82 tests. If for some landmarks the correct detection is not included in the candidates, it is automatically replaced with its most likely predicted position. This happened here to the left ear, and the lower lip. Note that the search tree is very degenerate, because our algorithm successfully prunes large areas of the search space.

One way to specify an accurate initialization is by detecting landmarks, correspondences between points in the model domain and points in the image. These are detected with sliding window detectors such as [26]. See [29] for a survey of this area. These detectors classify each patch of the image separetely as being one of the landmarks or the background, the positions that match are then typically clustered and the cluster centers returned. This works relatively well for such as as frontal faces which have a relatively unique appearance. But detecting landmarks such as the corners of the mouth or the tip of the nose is inherently more difficult, as these image patches are ambiguous. Many patches in an image look exactly like a corner of the mouth, if they are not seen in the larger context of the image.

A patch-based landmark detector will therefore return a number of false positive matches (detections of the landmark at the wrong positions), and for some landmarks also false negatives (no detection at the correct position). The algorithm presented here takes the output from a large (i.e. 23) number of landmark detectors and determines which are the correct detections. The candidates whose configuration can be best explained as resulting from a face are chosen. This is formalized as searching for the candidates which result in a shape model fit with minimal residual.

For N landmarks and K detections per landmark there are K^N possible combinations to consider. For typical images this is 20 landmarks and on average 7 detections per landmark, resulting in $7^{20} \approx 10^{16}$ combinations. Nonetheless, we are able to find the optimal configuration within less than a second by efficently discarding large areas of the search space by using the branch and bound framework introduced in [15]. Branch and bound has been used before in computer vision, for example for efficient object detection in [14] and to estimate camera parameters from matches between 2D image points and a 3D model in [12, 7] and [18]. The latter is more closely related to our work, as the determination of the camera parameters from a correspondence between 3D points and 2D landmarks is repeatedly solved as a subproblem within our algorithm. The difference is, that we *simultaneously* solve for the correct camera (and potentially shape) parameters and the image position.

The problem of choosing the right detections out of a candidate set has been addressed before with a stochastic search using RANSAC [9]. In [24] a RANSAC based algorithm with a fast rejection test was introduced which solves the same problem. The advantages of our algorithm are that we guarantee to find the globally optimal solution, and that our formulation is general enough to encompass different camera and shape models.

A closely related search method was presented in [16]. They formulate AAM fitting as an instance of the A* algorithm, which is itself an instance of branch and bound for graph search. Similar to our approach, [Lekadir and Yang] find an optimal fit by constraining the position of unknown landmarks with the help of landmarks which are already known. The algorithm presented here differs in that we can handle arbitrary shape models, and that we bound sets of landmark candidates, while [16] bound partial solutions where only a single landmark is picked from each set of candidates. Our method can therefore mimick the behaviour of [16], but more efficient search strategies can be implemented and are compared in this paper. Also, we show how to use branch and bound search for different models, instead of developing a solution for 2D AAMs.

2. Problem Formulation

We require a shape model, which is a function

$$M(\mathbf{\Theta}) = (m_1(\mathbf{\Theta}), \dots, m_N(\mathbf{\Theta})) \quad m_i : \mathbb{R}^{N_{\mathbf{\Theta}}} \to \mathbb{R}^2 \quad (1)$$

mapping the $N_{\mathbf{\Theta}}$ dimensional vector of model parameters $\mathbf{\Theta}$ to image positions $m_i(\mathbf{\Theta})$. This can for example be a 2D Point Distribution Model [8] or – as used throughout this paper – a fixed 3D Shape projected according to a weak perspective camera. It is also possible to use a full 3D shape model, but for expressionless faces this is not necessary to select the correct landmarks out of the candiates.

For each projected point m_i a set of candidate positions

$$L_i = \{\boldsymbol{l}_i^1, \boldsymbol{l}_i^2, \dots\} \qquad \boldsymbol{l}_i^j \in \mathbb{R}^2 \qquad (2)$$

is detected in the image, using any object detector. Detection is not the topic of this article, any classifier applied in a sliding window manner can be used. Obviously, the better the detector, the better the final results. Also, even though our formulation can handle a relatively large fraction of occluded or undetected points, we are unable to find the correct position if for more than 20% of the model vertices no correct detection is included in the candidate set.

The task is to assign to every model point one of the candidate positions such that the shape model can be best fit to the selection. Let us denote a *selection* $\mathbf{S}$ by the tuple

$$\mathbf{S} = (j_1, j_2, \dots, j_N) \qquad j_i \in \mathbb{N}, \qquad (3)$$

where j_i is the index of a candidate of landmark i. We choose the selection $\mathbf{S}^*$ which minimizes the distance between the shape model and the image landmarks:

$$\begin{aligned} \mathbf{S}^* &= \underset{\mathbf{S}=(j_1,\dots,j_N)}{\arg\min} f(\mathbf{S}) \\ f(\mathbf{S}) &= \min_{\mathbf{\Theta}} \sum_i \rho\left(\left\| m_i(\mathbf{\Theta}) - \boldsymbol{l}_i^{j_i} \right\|\right) \quad . \end{aligned} \qquad (4)$$

Here $\rho : \mathbb{R} \to \mathbb{R}$ is a robust function acting on the distance between the projected model vertices and the detected candidate points. We use the Huber distance [13], which behaves like the squared distance up to some point and then switches to the absolute distance. To some extent this allows us to handle missing detections, and points which are invisible due to occlusion.

3. Branch and Bound

The problem as formulated above is a discrete optimization problem. The number of possible selections $\mathbf{S}$ within the candidates is exponential in the number N of points of the model, growing as K^N for K candidates. Nonetheless, we are able to efficently find the optimal selection with the help of branch and bound [15]. In this section we recapitulate branch and bound in its general formulation using the terminology introduced in the previous section, and then flesh out the parts which constitute our algorithm.

Branch and bound finds the element $\mathbf{S}^*$ in a set $\mathfrak{S} = \{\mathbf{S}_1, \mathbf{S}_2, \dots\}$ which minimizes a function $f(\mathbf{S})$. The idea is to reason not over single elements, but over sets of elements, which can then be discarded in whole. It uses a function defined over subsets $\mathcal{P} \subseteq \mathfrak{S}$ which bounds the value of the cost function for the elements in the subset from below:

$$g(\mathcal{P}) \leq \min_{\mathbf{S} \in \mathcal{P}} f(\mathbf{S}) \quad . \qquad (5)$$

Additionally, we require that for sets consisting of only a single entry the lower bound is tight:

$$g(\{\mathbf{S}\}) = f(\mathbf{S}) \quad . \tag{6}$$

The general branch and bound procedure is

```
1. Start with the set of all elements $\mathcal{Q} = \{\mathfrak{S}\}$
2. **Repeat**:
    (a) Take the minimal subset
        $\mathcal{P}_i \leftarrow \arg\min_{\mathcal{P}_i \in \mathcal{Q}} g(\mathcal{P}_i) \quad ; \qquad \mathcal{Q} \leftarrow \mathcal{Q} \setminus \{\mathcal{P}_i\}$
    (b) **Return S if** $\mathcal{P}_i = \{\mathbf{S}\}$ is a single element.
    (c) Split $\mathcal{P}_i$ into
        $\mathcal{P}_i^1 \subset \mathcal{P}_i, \qquad \mathcal{P}_i^2 \subset \mathcal{P}_i \quad$ s.t. $\mathcal{P}_i = \mathcal{P}_i^1 \cup \mathcal{P}_i^2$.
    (d) Add the new subsets to the candidates
        $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{\mathcal{P}_i^1, \mathcal{P}_i^2\}$
```

A branch and bound algorithm for a specific problem, such as the one solved in this paper, needs to specify (1) the cost function f which is minimized (2) a bounding function g which is as tight as possible but efficient to evaluate, (3) the representation of the candidate sets, such that one does not have to store all members of $\mathcal{P}$ explicitly, and (4) a splitting strategy which splits a given $\mathcal{P}$ into new subsets $\mathcal{P}_i^1, \mathcal{P}_i^2$.

In practice, $\mathcal{Q}$ is implemented as a priority queue, such that it is cheap to select the set of candidates with the minimal value of the bounding function.

4. Landmark Detection with Branch and Bound

In this section we specify the four ingredients necessary to define the branch and bound algorithm for landmark detection. All sets defined here are finite, but to avoid the clutter of having to introduce a variable for the cardinality of every set we leave the count implicit.

4.1. Cost Function

The cost function was specified in Equation 4, it is the residual of the optimal fit of the model to the chosen candidate points.

4.2. Bounding function

Branch and bound requires a function $g(\mathcal{P})$ operating on sets of selections which bounds the cost function $f(\mathbf{S})$ from below, such that $g(\mathcal{P}) \leq \min_{\mathbf{S} \in \mathcal{P}} f(\mathbf{S})$. Calculating a g which exactly returns the value of the optimal selection is as hard as solving the original problem, we therefore need to construct a bounding function which can be efficiently evaluated but has a bound which is as tight as possible. Remember that f is defined as the minimum residual which can be reached when fitting the model to the candidates in a selection. We now relax $g(\mathcal{P})$ such that it does not minimize the distance towards the optimal selection within $\mathcal{P}$, but instead towards the convex hulls of the candidate points in the selections in $\mathcal{P}$. Denote the union of all candidate points of the ith landmark which are included in any selection $\mathbf{S} \in \mathcal{P}$ by $\mathcal{P}_i$, and by $\boldsymbol{l}_i^{\mathcal{P}_i} = \{\boldsymbol{l}_i^{j_i^1}, \boldsymbol{l}_i^{j_i^2}, \dots\}$ the corresponding landmarks. Then the sum of the distances towards the convex hull of the candidate points in $\mathcal{P}_i$

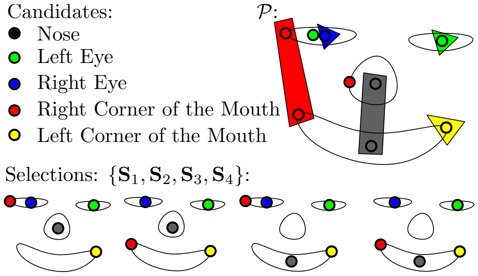

Candidates:
● Nose
● Left Eye
● Right Eye
● Right Corner of the Mouth
● Left Corner of the Mouth
$\mathcal{P}$:
Selections: $\{\mathbf{S}_1, \mathbf{S}_2, \mathbf{S}_3, \mathbf{S}_4\}$:

Figure 2. During the branch and bound search we consider sets of selections, which are defined by one subset of the candidate points for each landmark. The Cartesian product of the choosen candidate points for each landmark defines a subset of selections. The figure shows candidates for some landmark points (colored circles), and a choice of subsets of each landmark (denoted by the enclosing polygon). All four selections $\mathbf{S}_1, \dots, \mathbf{S}_4$ which are included in the set $\mathcal{P}$ are listed in the lower part of the figure. For real-world problems there are up to 16 candidate positions per landmark and up to 23 landmarks, leading to much larger sets, which still are compactly represented by the subsets of candidates points.

$$g(\mathcal{P}) = \min_{\boldsymbol{\Theta}} \sum_i \rho\left(d_{\text{convex hull}}(\boldsymbol{l}_i^{\mathcal{P}_i}, m_i(\boldsymbol{\Theta}))\right)$$
$$d_{\text{convex hull}}(\boldsymbol{l}_i^{\mathcal{P}}, \boldsymbol{x}) = \min_{c \in \text{convex hull}(\boldsymbol{l}_i^{\mathcal{P}})} \|x - c\| \quad , \tag{7}$$

is a lower bound on f, as we have only added more points towards which the distance is calculated. So for monotone ρ we have defined a suitable bounding function g, which can be evaluated efficiently by fitting to convex polygons instead of fitting to landmarks.

Our algorithm assumes that such a fit can be calculated efficiently, and that the fitting is convex, or at least that for all interesting poses the optimal fit can be obtained from the initial position. Our experiments show, that this is the case for the shape model and distance used as an example throughout this article. Recall, that we are using the Huber distance and a constant shape model with a weak perspective projection.

4.3. Representation of sets of selections

The use of convex hulls of the active candidate points naturally leads to a compact representation of sets of selections $\mathcal{P}$. We define the elements $\mathbf{S} \in \mathcal{P}$ to be the Cartesian product of *active candidates* $\mathcal{A}_i$ for each landmark i. That is, we encode the sets of selections as tuples of sets of active

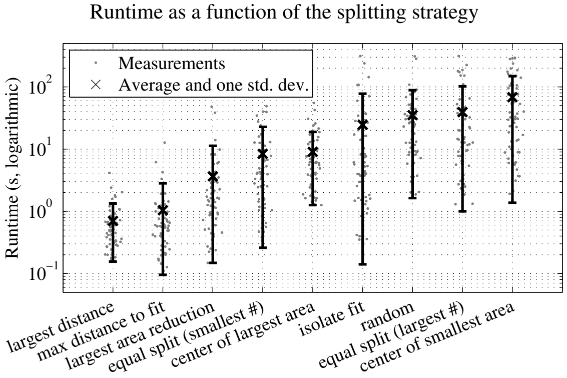


Figure 3. Different splitting strategies result in vastly different performance. Note that the often used strategie 'split into equal sized problems' is one of the worst strategies for branch and bound. Note the logarithmic scale.

candidates,

$$\mathcal{P} = (\mathcal{A}_1, \dots, \mathcal{A}_N) \qquad \mathcal{A}_i = \{j_i^1, j_i^2, \dots\}. \tag{8}$$

Such a set of selections contains all combinations of active landmarks in this set, such that we can also write

$$\mathcal{P} = \{\mathbf{S} = (j_1, \dots, j_n) \mid j_1 \in \mathcal{A}_1, \dots, j_N \in \mathcal{A}_n\}\} . \tag{9}$$

Even though we defined $\mathcal{P}$ twice, once as a set of selections in Equation 9 and once as its encoding in Equation 8, it will always be clear from context which definition we are using. Refer also to Figure 2 for a visualization of the encoding concept.

This is a very compact representation of combinatorially many solutions which can be implemented as a simple tuple of bitmasks.

4.4. Splitting Strategy

The choice of a suitable splitting strategy is crucial for a branch and bound algorithm to be efficient. A strategy is good, if it leads to a fast isolation of the correct solution and splits the remaining candidate sets into sets for which the lower bound is larger than the function value of the solution. Obviously, one can not efficiently search for the optimal splitting sequence, therefore a strategy has to be choosen which is expected to perform good, and is cheap to apply.

With our formulation a disjunct split of a candidate set $\mathcal{P}$ into two subsets $\mathcal{P}_1, \mathcal{P}_2$ can be achieved by choosing one landmark, and within this landmark splitting the candidates into two disjunct sets. We only considered splits where the candidate points were split either along the horizontal or vertical axis. Splitting along a line ensures that the convex hulls of the resulting partitions do not overlap, axis aligned splits were used because this reduces the number of possible splits to twice the number of landmarks, allowing us to define the splitting strategies as objective functions over splits, which are minimized by complete enumeration.

We tested a large number of splitting strategies as tabulated in Figure 3, and found that their behaviour differs a lot. The most efficient strategy we found is to divide the candidate point sets such that the distance of the convex polygons of the split landmark candidates was maximal. The worst performing methods split the problem into equally sized subproblems, while the best performing methods rapdily increase the lower bound. It is conceivable that even better strategies can be devised, or learned from example problems. This is a venue for further research.

5. Scale

The cost as formulated in Equation 4 prefers smaller faces over larger faces, if these are detected, because the residual is calculated from the image distances. This could be overcome by normalizing with respect to the scale, but the resulting cost function is then more expensive to optimize. Instead, we exploit that the landmark detector anyhow has to search over multiple scales, and we therefore know the approximate scale of the face in the image. The image is resized to a pyramid of scales, and at each pyramid level we perform a candidate point detection and landmark selection, where the landmark selection is constrained to faces which have a minimum size corresponding to the size at which the landmarks are detected. To constrain the search, we add a regularization term to the cost function, resulting in

$$f(\mathbf{S}) = \min_{\mathbf{\Theta}} \left(\sum_i \rho\left(\left\| m_i(\mathbf{\Theta}) - \boldsymbol{l}_i^{j_i} \right\| \right) + r(\mathrm{scale}(\mathbf{\Theta})) \right) \tag{10}$$

$$g(\mathcal{P}) = \min_{\mathbf{\Theta}} \left(\sum_i \rho\left(\min_{c \in \text{convex hull}(\boldsymbol{l}_i^{\mathcal{P}_i})} \| m_i(\mathbf{\Theta}) - c \| \right) + r(\mathrm{scale}(\mathbf{\Theta})) \right) \tag{11}$$

$$r(\sigma) = -\log\left(\frac{\sigma - \tau}{\tau}\right) + \frac{\sigma - \tau}{\tau} \quad . \tag{12}$$

The regularization assigns infinite cost to scales smaller than τ and increases slowly for larger scales. We are using a weak projective model, that is our point model is a function

$$m_i(\mathbf{\Theta} = (\boldsymbol{q}, \boldsymbol{t})) = \boldsymbol{R}_{\boldsymbol{q}} \boldsymbol{v}_i + \boldsymbol{t} \tag{13}$$

$$\boldsymbol{R}_{\boldsymbol{q}=(a,b,c,d)} = \begin{bmatrix} a^2 + b^2 - c^2 - d^2 & 2(bc - ad) & 2(ac + bd) \\ 2(ad + bc) & a^2 - b^2 + c^2 - d^2 & 2(cd - ab) \end{bmatrix} .$$

Here $\boldsymbol{R}$ is a matrix which describes a 3D rotation, projection onto the first two dimensions and scaling and $\boldsymbol{v}_i$ are the vertices of a 3D shape. The matrix is described in terms of an unnormalized quaternion $\boldsymbol{q}$ [11]. The scale is therefore just $\|\boldsymbol{q}\|^2$, making it easy to differentiate and minimize the above equations.

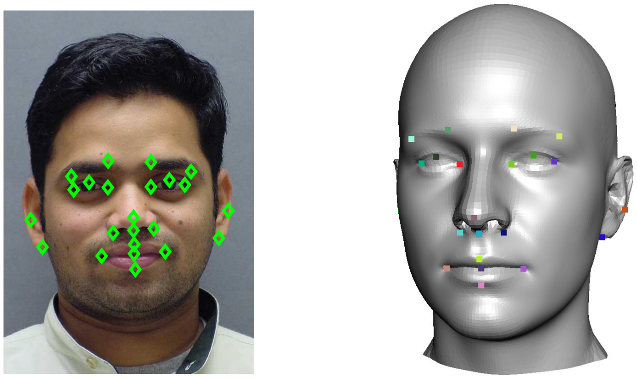


Figure 4. The landmarks used in the experiment on one of the training images and the corresponding vertices of the 3D shape that is fit to the image to select the correct landmark candidates.

6. Multiple Faces per Image

When detecting more than one image per face, it is possible to exploit the information from the search for the first face when searching for the second face. In this case, we return the first face found, and remove the candidate positions belonging to this face from all candidates in the queue. This keeps the lower bound constraint, because the minimal residual increases, when removing candidate positions. The search is then continued on the pre-filled queue, and rapidly finds the second face.

When searching only for a single face but at an unknown scale, then it is fastest to initialize the queue with one selection per scale, choosing all detections at that scale. The branch and bound algorithm will then stop when the face with the smallest cost at any scale has been found, without the need to continue the search at the other scales.

7. Candidate Detector

Even though the detector is not the topic of this article, a landmark detector is nonetheless neccessary to evaluate our algorithm in practice. No pretrained detector for the large amount of landmarks used in our experiments was available, we therefore describe in the following section the landmark detector used in our experiments. It is possible to replace this detector with any other detector in an application. We trained a detector for 23 landmarks, as shown in Figure 4. The 3D shape corresponding to these landmarks was read from the mean of the BFM 3D Model [19]. The landmark detector consists of two phases, in the first phase we use a decision forest [4] to classify image patches of size 5×5 into interestpoint or not interestpoint. This was used to extract only 1-3% of the image pixels as potential candidates for landmark positions. At these interestpoints we extracted 64 features by projecting patches of size 33×33 onto the first 64 eigenvectors of the covariance of all patches around interestpoints in the training data. This basis is shown in Figure 6. This reduced set of 64 features per patch was then classified with the help of another decision forest, and up to sixteen detections per image and landmark were kept as candidates, if the detection confidence was above a fixed threshold. The decision functions in the nodes of the forest are linear functions of the full dimensional space. The decision functions were learned by randomly drawing two samples from different classes, and taking the direction between these samples as the normal of the decision function, and the midpoint as its position. Twenty directions per node were tried, and the one which most decreased the entropy in the resulting classes was chosen.

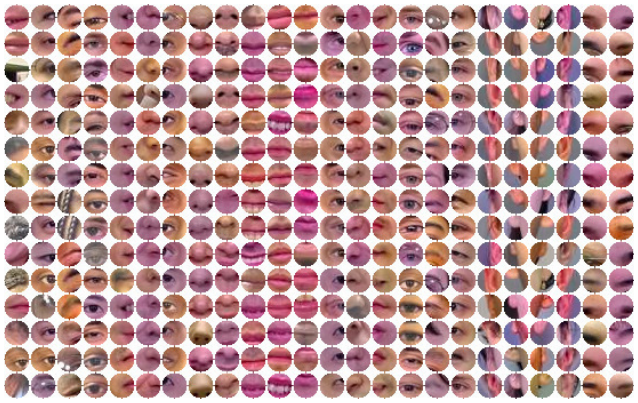


Figure 5. Some of the example patches used to train the landmark detector. Each column shows 15 randomly selected samples for one of the landmark classes, the first column contains background samples. This figure is best viewed in the digital copy.

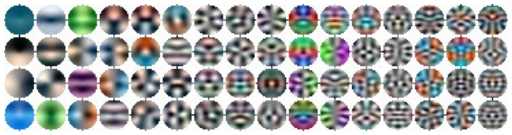


Figure 6. Linear Basis used to extract features for the classification. The basis was learned from example patches and contains explicit steerable color edge detectors at multiple scales.

8. Experiments

We present two types of experiments. First, on synthetic data we analyze the break-down points of our algorithm, without dependence on a good object detector. In a second experiment we show experimental results on a number of difficult images from the color feret database [20, 21]. These results depend on the candidate point detector, and will improve when a better tuned detector is used, but they are included to show that even with a suboptimal detector a useful system can be built with our method.

8.1. Synthetic data

To analyze the performance of the algorithm independently from the performance of the candidate point detector,

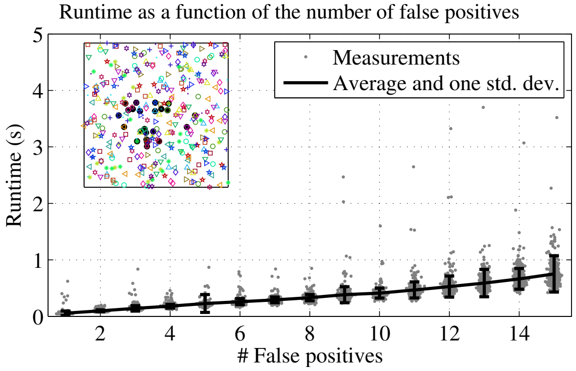


Figure 7. Our algorithm seems to scale only linearly in the number of false positives per landmark. We produced for a dense sampling of rotations adn scalings noise-free synthetic data with a uniformly distributed background set of false detections. The graph shows linear runtime behaviour, a typical example with 23 landmarks and 9 false positives per landmark is plotted in the inset.

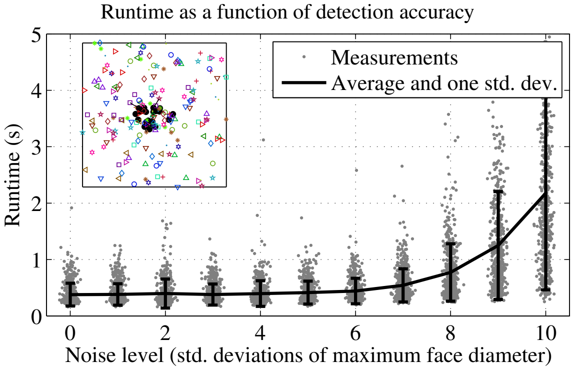


Figure 8. Our algorithm behaves well in the presence of noise. It only starts to require many iterations, once the amount of noise is larger than the distance between the landarks in the face. The graph shows the amount of noise (in std-deviations of the maximum face diameter) against runtime, and in the inset a typical example for a very situation with noise-level 6, where the lines show the displacement of the true landmarks to the noisy landmarks.

we performed experiments on synthetic datasets.

As a first experiment, we generated landmark coordinates from the model, which were perturbed with Gaussian noise of increasing levels, and added false detections at uniformly randomly distributed positions in the image. In this setting, we can evaluate (1) the effect on runtime of adding more candidates, (2) the point were a displacement from our rigid model is so large, that the optimum configuration no longer is the right choice, and (3) how many of the correct landmarks can be completely removed before the algorithm breaks down.

Number of false positives The runtime of our algorithm grows approximately linear in the number of added false positives. This is demonstrated in Figure 7, where we have also shown an example of a synthetic problem as described in the previous paragraph. IN this experiment all landmarks were available (no false negatives), and zero noise was used.

Amount of noise on the landmarks Next, we evaluated the effect of adding noise to the landmarks, for a fixed number of false positives. We found, that the distance between detections of the same class needs to be larger than the maximal noise on the landmarks, as otherwise the splitted subproblems have nearly identical costs and need to be enumerated completely, leading to a large number of evaluations. For real world data we achieve this by non-maximum suppression within regions of the size of the expected noise, and for synthetic data we create suitable datasets, were the minimum distance is 1.5 times the maximum noise. In this experiment all landmarks were observed, and Gaussian noise cut off at 2σ was added to the landmarks. We observed that the runtime does not depend on the amount of noise, until the deviations are larger than the distance between the landmarks. At this point we observed a rapid increase in the number of iterations necessary, and also an increase in the variability of the runtime. This highlights the necessity to choose the detected landmarks such that they can be located with higher accuracy than the distance to their neighbours, which also makes intuitive sense, because otherwise we can no longer distinguish the landmarks. The results are graphed in Figure 8, observe the sharp increase at $\sigma = 8\%$ of the maximum face diameter.

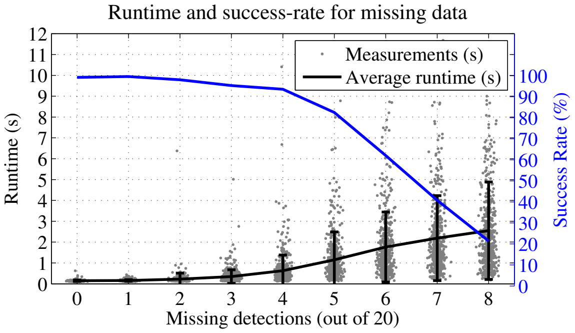


Figure 9. Missing detections are handled by the robust distance function. We observe, that the minimum of our cost function is at the expected position for up to 20% completely missed detections. We randomly selected 18 out of the 23 landmarks, and varied the amount of landmarks which had no correct detections but only false positives in their candidate set. The search was cut off, when the cost was above a threshold. The success rate (number of times that the returned globally minimal solution was the one we wanted to find) is shown in blue, and the runtime in black.

Missed Detections In practice, missing detections will occur. Our strategy to handle this is to detect a large number of landmarks (23) and to include only the 18 land-

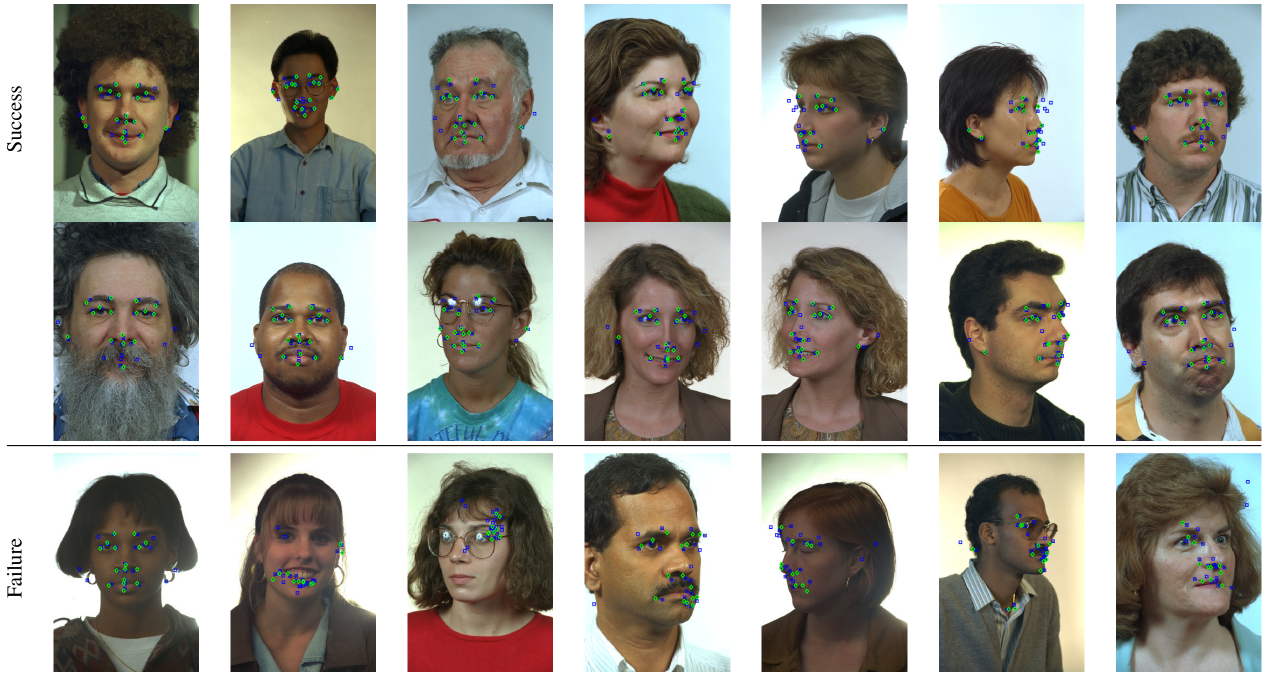


Figure 10. Some randomly chosen images from the color feret database for each pose, and the detected landmark positions. The first two rows are success cases, the last row shows a failure case. This figure is best viewed in the electronic copy.

Color Feret Pose		Correct detections (%)
fa	Frontal	92%
fb	Frontal	86%
ql	Quarter left	93%
qr	Quarter right	94%
rc	Random (10 deg)	91%
hl	Half left	69%
hr	Half right	72%

Figure 11. Success rate on the color feret dataset. Note that our algorithm always picks the globally optimal candidate set, but the detector used in our experiments performed suboptimally. We expect these rates to increase, once more attention can be devoted to the detector. Note that the half-left and half-right datasets contain also a number of profile views, for which we do not have a suitable detector.

marks wich contain the strongest response. This typically increases the number of correct detections, without loosing expressivity. Also, we use a relatively low threshold, detecting many candidates, such that the candidates are likely to contain the true landmark. But also with this strategy there will be a certain amount of completely undetected landmarks, which we handle by using the Huber distance measure in Equation 4. In Figure 9 we graph the effect of missing detections in a synthetic experiment. We observe that the cost no longer has its minimum at the correct selection, once more than 20% of the points are completely missing. And searching for the optimum becomes expensive, because many solutions have a similar cost. We mitigate this by setting an upper limit on the acceptable distance for a match to be a face, once the current lower bound raises above this level, we report that no face has been found.

8.2. Real world data

We detected landmarks in the non-profile poses of the color feret database. Our detector was not trained for profile views, so we did not test these subsets. We tested 100 randomly chosen images out of each class. A detection was labelled as correct, when the predicted positions of *all 23* landmarks were approximately correct, as judged by an experimenter. Some example detections and failures are shown in Figure 10, and the detection rates are tabulated in Figure 11 for the different experiments. We searched at multiple scales and kept the detection with the smallest residual. The runtime in these experiments was dominated by the detector. Generally, the most problematic images were those, were the ears were invisible or not detected. The inner face landmarks were detected correctly in most cases, but the algorithm can be trapped into the wrong pose, if the two landmarks at the ears have not been detected. This accounts for the majority of failed detections.

9. Conclusion

We presented a novel algorithm to find the globally best set of detections out of a number of candidate detections with the help of a shape model. The algorithm is applicable to a large number of shape models. As it is globally optimal we hope that it will supersede the use of stochastic algorithms such as RANSAC for this type of problem. The algorithm can be added as an additional step to existing systems to improve their performance and robustness. To stimulate the use of this algorithm we publish efficient source code with a matlab and c++ interface, and a pretrained detector for 23 facial landmarks.[1]

Acknowledgement: Portions of the research in this paper use the FERET database collected under the FERET program, sponsored by the DOD Counterdrug Technology Development Program Office.

References

[1] B. Amberg, A. Blake, and T. Vetter. On compositional image alignment, with an application to active appearance models. In *CVPR'09*, volume 0, pages 1714–1721. IEEE Computer Society, June 2009.

[2] V. Blanz, C. Basso, T. Vetter, and T. Poggio. Reanimating faces in images and video. In P. Brunet and D. W. Fellner, editors, *EUROGRAPHICS 2003*, volume 22, issue 3 of *Computer Graphics Forum*, pages 641–650. The Eurographics Association, Blackwell, 2003.

[3] V. Blanz and T. Vetter. A morphable model for the synthesis of 3D faces. In *SIGGRAPH'99: Computer Graphics and Interactive Techniques*, pages 187–194. ACM Press/Addison-Wesley Publishing Co., 1999.

[4] L. Breiman. *Classification and Regression Trees*. Chapman & Hall/CRC, Boca Raton, 1984.

[5] P. Breuer, K.-I. Kim, W. Kienzle, B. Schölkopf, and V. Blanz. Automatic 3D face reconstruction from single images or video. In *FG'08*, pages 1–8, Sept. 2008.

[6] H. W. Byun. Realistic facial modeling and animation based on high resolution capture. In *ACIVS'07: Advanced concepts for intelligent vision systems*, pages 417–426. Springer, 2007.

[7] K. Choi, S. Lee, and Y. Seo. A branch-and-bound algorithm for globally optimal camera pose and focal length. *Image and Vision Computing*, 28(9):1369–1376, Sept. 2010.

[8] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active Appearance Models. *PAMI*, 23(6):681–685, 2001.

[9] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

[10] P. Fua. Regularized Bundle-Adjustment to model heads from image sequences without calibration data. *Int. J. Comput. Vision*, 38:153–171, July 2000.

[11] W. R. Hamilton. On quaternions, or on a new system of imaginaries in algebra. *Philosophical Magazine*, Vol. 25(n 3):489–495, 1844.

[12] R. Hartley and F. Kahl. Global optimization through rotation space search. *International Journal of Computer Vision*, 82(1):64–79, 2009.

[13] P. J. Huber. Robust estimation of a location parameter. *The Annals of Mathematical Statistics*, 35(1):73–101, Mar. 1964.

[14] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Efficient subwindow search: A branch and bound framework for object localization. *PAMI*, 31(12):2129–2142, Dec. 2009.

[15] A. H. Land and A. G. Doig. An automatic method for solving discrete programming problems. In M. Jünger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, and L. A. Wolsey, editors, *50 Years of Integer Programming 1958-2008*, chapter 5, pages 105–132. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.

[16] K. Lekadir and G.-Z. Yang. Optimal feature point selection and automatic initialization in active shape model search. In D. Metaxas, L. Axel, G. Fichtinger, and G. Szkely, editors, *MICCAI 2008*, volume 5241 of *LNCS*, pages 434–441. Springer, 2008.

[17] I. Matthews and S. Baker. Active Appearance Models Revisited. *IJCV*, 60(2):135–164, Nov. 2004.

[18] C. Olsson, F. Kahl, and M. Oskarsson. Optimal Estimation of Perspective Camera Pose. In *ICPR'06*, pages 5–8, 2006.

[19] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter. A 3d face model for pose and illumination invariant face recognition. In *AVSS*, Genova, Italy, 2009. IEEE.

[20] P. Phillips, H. Moon, S. Rizvi, and P. Rauss. The FERET Evaluation Methodology for Face Recognition Algorithms. *PAMI*, 22:1090–1104, 2000.

[21] P. Phillips, H. Wechsler, J. Huang, and P. Rauss. The FERET database and evaluation procedure for face recognition algorithms. *Image and Vision Computing*, 16(5):295–306, 1998.

[22] S. Romdhani, J. Ho, T. Vetter, and D. J. Kriegman. Face recognition using 3-D models: Pose and illumination. *Proceedings of the IEEE*, 94(11):1977–1999, 2006.

[23] S. Romdhani. and T. Vetter. Estimating 3D shape and texture using pixel intensity, edges, specular highlights, texture constraints and a prior. In *CVPR'05*, volume 2, pages 986–993 vol. 2, 2005.

[24] S. Romdhani and T. Vetter. 3D Probabilistic Feature Point Model for Object Detection and Recognition. In *CVPR'07*, pages 1–8, 2007.

[25] N. Stoiber, R. Seguier, and G. Breton. Facial animation retargeting and control based on a human appearance space. *Computer Animation and Virtual Worlds*, 21(1):39–54, 2010.

[26] P. Viola and M. Jones. Robust real-time object detection. In *International Journal of Computer Vision*, 2001.

[27] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *ACM TOG*, 24(3):426–433, July 2005.

[28] M. Wimmer, F. Stulp, S. Pietzsch, and B. Radig. Learning local objective functions for robust face model fitting. *PAMI*, 30(8):1357–1370, 2008.

[29] C. Zhang and Z. Zhang. A survey of recent advances in face detection. Technical report, Microsoft Research, 2010.

[30] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *ACM Comput. Surv.*, 35(4):399–458, Dec. 2003.

[1] The code can be requested at: `http://www.cs.unibas.ch/personen/amberg_brian/bnb/`.