Hierarchical Object Parsing from Structured Noisy Point Clouds

Adrian Barbu

Abstract—Object parsing and segmentation from point clouds are challenging tasks because the relevant data is available only as thin structures along object boundaries or other features, and is corrupted by large amounts of noise. To handle this kind of data, flexible shape models are desired that can accurately follow the object boundaries. Popular models such as Active Shape and Active Appearance models lack the necessary flexibility for this task, while recent approaches such as the Recursive Compositional Models make model simplifications in order to obtain computational guarantees. This paper investigates a hierarchical Bayesian model of shape and appearance in a generative setting. The input data is explained by an object parsing layer, which is a deformation of a hidden PCA shape model with Gaussian prior. The paper also introduces a novel efficient inference algorithm that uses informed data-driven proposals to initialize local searches for the hidden variables. Applied to the problem of object parsing from structured point clouds such as edge detection images, the proposed approach obtains state of the art parsing errors on two standard datasets without using any intensity information.

Index Terms—object parsing, hierarchical models, MRF optimization, active shape model.

1 Introduction

Object parsing and segmentation are important problems with many applications in computer vision and medical imaging. While object segmentation is only directed towards labeling the object pixels, object parsing is aimed at identifying the object parts such as head, body, legs, etc.

The object parsing problem presents challenges in both modeling and computing. It is difficult to find accurate probability or energy models that have high probability on correct object parsings of the input image and low probability everywhere else. Moreover, it is difficult to design inference algorithms that find the correct object parsing from the image in a reasonable amount of time. Most inference problems for non-tree Markov Random Field (MRF) based models are NP-hard (proved in [8] for the Potts model) so an exact solution cannot be expected in polynomial time.

Because of the computational challenges, different trade-offs between model accuracy and computational feasibility have been made in previous works.

A. Barbu is with the Department of Statistics, Florida State University, Tallahassee, Florida 32306, USA, Phone: 850-980-2516, Fax: 850-644-5271, Email: abarbu@stat.fsu.edu.

The Active Shape [11] and Active Appearance Models [10] use a simplified object representation using Principal Component Analysis (PCA) and use local information to search for a solution. The Active Shape Models (ASM) contain only a PCA shape model and alternate one step that searches for local boundary evidence on the shape normals with another step that reprojects the evidence onto the PCA hyperplane, until convergence. The trade-off made by the ASM is the local search for the solution, based on partial image information existent on the shape normals. Because of this trade-off, the result depends on initialization.

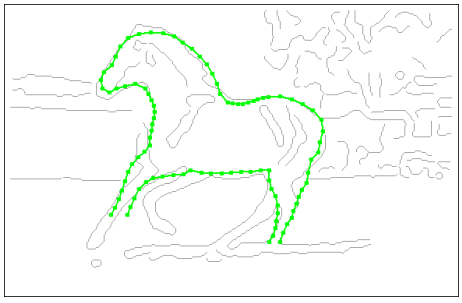

Fig. 1. Motivation for the hierarchical model. A shape described by PCA (shown with green dots) is not flexible enough to accurately follow the object boundary, but can serve as a backbone to limit the variability of the model.

The Active Appearance Models (AAM) also model the object appearance by PCA and employ a trained iterative algorithm that deforms an initial shape until convergence, guided by the image. The AAM uses more image information than the ASM, but the algorithm is still greedy, obtaining a local optimum that is dependent on initialization. In [38], [40] the radius of attraction of the AAM modes was used together with a measure of error based on sum of squares to obtain a globally optimal solution of the AAM.

Another limitation of the ASM/AAM models is the rigidity, as a low dimensional PCA shape cannot accurately describe the shape variability existent in real images, and is limited only to the main deformations. This is illustrated in Figure 1, where a 10-dimensional PCA shape shown in green cannot accurately follow the horse boundary and is off by a few pixels around the ears, back, legs, etc.

Many other previous works [27], [46] make trade-offs

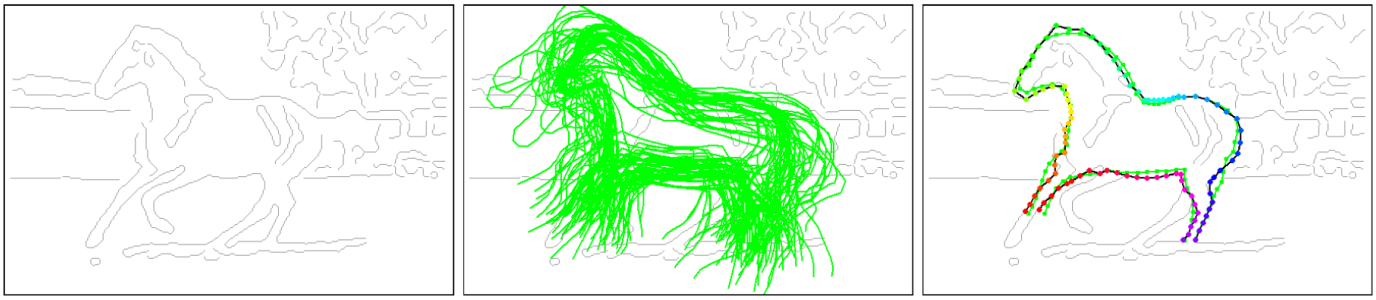

Fig. 2. Our approach starts by tracing the points into chains (left), finds data-driven PCA candidates (middle) that are used to initialize local optimizations of the model parameters. The parameters of lowest energy give the parsing result (right, black with colored dots) and associated PCA shape (right, green).

in the model in order to be able to cast it into a class for which efficient inference algorithms exist, and will be discussed in more detail in the next section.

The approach introduced in this paper tries to avoid making any modeling compromises, at the expense of having no off-the-shelf algorithm for inference. Instead, we introduce a novel inference algorithm that searches for many local optima using data-driven techniques and is more likely to find a solution close to the global optimum than a local optimization algorithm. The quantitative experiments will tell us how successful this strategy was in obtaining good parsing results. We observe that the proposed model together with the suboptimal algorithm can obtain state of the art object parsing results without using any intensity information. This finding is in line with our previous work [3] where a properly trained model compensated for a fast suboptimal algorithm for image denoising.

The contributions of the paper are the following:

First, it presents a hierarchical generative model that represents the object shape as a MRF-based deformation from a PCA backbone, obtaining a more accurate boundary delineation than with the PCA model alone. The shape model can be sampled if desired and used for numerical integration or to compute marginal statistics. The generative model also contains a data term that connects the image information with the shape model. Due to the high accuracy of the shape description, this model can be used for object parsing from point clouds (such as those obtained from edge detection), where the data information is one pixel wide.

Second, it presents an optimization algorithm for finding a strong optimum for the hierarchical model. The algorithm, illustrated in Figure 2, uses a data-driven set of PCA candidates to initialize local searches for optimizing a unique energy based on the MRF deformation and PCA parameters.

Third, it presents an evaluation of the proposed approach to parsing horses [6], cows [24] and faces [39] from point clouds obtained from edge detection. The evaluation revealed that the proposed algorithm obtains state of the art results on two of the datasets without using any intensity information.

The proposed shape model and inference algorithm could in principle be adapted for other object parsing and segmentation applications by using appropriate data terms, or even by employing more accurate backbone shape models than the PCA.

2 Related Work

Matching point clouds dates back to the Softassign [31] and the Robust Point Matching Algorithm [9]. These methods are capable of finding correspondences and transformations between unordered point clouds. However, for the problem of object parsing, their shape model can be considered a template plus deformation, which might not be an accurate enough model for real objects. It would be interesting to see how these methods perform in finding real objects from real edge detection images such as those presented in this paper.

An ASM based approach containing a dynamic programming step similar to the one in this paper was proposed in [4]. However, the appearance model from [4] is based on learned intensity profiles, thus it relies heavily on the intensity information. Moreover, the approach depends on an initialization step and does not have a hierarchical energy formulation as the one proposed in this paper.

The Oriented ASM [29] also uses dynamic programming to find the minimum cost boundary. The cost is based on intensity profiles perpendicular to the boundary at the landmark points and a cost along a contour obtained by the live wire method [14] between consecutive landmark points. The regularized PCA shape is used to constrain the obtained boundary, while in our approach it is part of the energy formulation, allowing deformations from the PCA shape.

A probabilistic model of shape and appearance was used in the Constrained Local Models (CLM) [13]. The appearance was modeled using local AAM templates at a number of keypoints on the object to be parsed. A Bayesian model combined a measure of appearance similarity of the local image templates with a global PCA shape prior similar to the one used in this paper. A local optimum of the joint model was obtained using the simplex method.

A refinement of the CLM is the cascade of Combined Shape Models (c-CSM) [42]. This approach alternates two steps, one maximizing a posterior probability $p(Y|I)$ of an approximate solution Y and one of obtaining

a regularized solution X from Y, which maximizes $p(X|Y)$. Using this terminology, our approach optimizes a single energy function $E(X,Y)$ that has a likelihood term $p(I|Y)$ and a prior $p(Y|X)$ based on the regularized shape X, and a prior $p(X)$. In c-CSM the final result is the regularized shape X while in our approach the final result is Y. Our inference algorithm is also different from the c-CSM in that we use data-driven candidates.

A robust hierarchical shape model was constructed for multi-view car alignment [27]. The model is a probabilistic PCA, which is a PCA model plus i.i.d Gaussian deformation of each vertex. It allows large deformations of the observed shape points and can also handle missing points due to occlusion or failures of the part detectors. Similar to our work, shape candidates are constructed from partial information obtained from part detectors to initialize a local search. However, these candidates have been directly generated using RANSAC, as the correspondence between the part detections and the model points was known. In the object parsing from point clouds, the correspondence between input points and the object points is not known and the fraction of outliers is usually higher than 90%, making RANSAC computationally prohibitive (a computation using eq. (13) from [27] gives about $4 \cdot 10^5$ candidates required for 99% certainty).

The Recursive Compositional Models (RCM) [46] represents the object shape in a hierarchical fashion using multiple levels of rotation-invariant models based on triplets of elements. The first level elements are the detected image edges, while the elements for each subsequent level are summaries of the triplets from the previous level. This hierarchical model allows inference by a version of dynamic programming with pruning. Through the hierarchy, the model enforces long range interactions between the shape elements, but at the same time some desired short-range interactions are missing. In contrast, our hierarchical model represents the shape using a PCA model plus MRF deformations along the normals, with the PCA model providing the long range interactions while the MRF allowing for smooth deformations. Because of the high connectivity of our proposed model, exact inference algorithms based on dynamic programming are not applicable. Instead, we propose a smart search algorithm that makes local searches at a number of locations dictated by a bottom-up data-driven process. The advantage of our approach is the simplicity of the model, that can be easily learned from training examples. Our evaluation shows that the errors obtained by our approach are similar to the RCM on two of the three datasets, without using any image intensity information.

The knowledge based segmentation [5] uses a shape prior based on pairwise cliques between the shape points and a primal-dual algorithm for inference. In contrast, our framework uses a PCA-based model that cannot be decomposed in pairwise cliques and could in principle be extended to work with non-linear shape models.

Torresani et al, [41] model the shape as a rigid transformation plus PCA, without the MRF deformation from our formulation.

Interactive Object Segmentation with Graph Cuts [19] imposes a shape prior on a Graph Cut energy. However, the shape prior is based on a template with similarity transformation without any deformation and the Graph Cut energy is on pixels, so no object parsing or aligned boundary is obtained. This work has been extended with a Kernel PCA shape prior [30], but still depends on manual initialization and obtains just a segmentation without boundary alignment. In contrast, our method obtains object parsing and boundary alignment, hence not only the object boundary but also the object parts are obtained.

The work of Ren et al, [32] is targeted to object boundary detection. If does not obtain a clear object segmentation or a parsing into object parts. Furthermore, it uses both edge and gradient information as input data.

Felzenszwalb and Schwartz, [15] use a shape tree as a model and focus on shape matching and retrieval, without evaluating the parsing error.

Zhu et al, [47] use a circularity measure to find cycles with good continuation in edge detection images. However, it does not have any global shape model so it addresses a different problem than ours.

The Active Skeleton [2] uses a skeleton-based shape model to detect objects from edge detection images. Even though in principle the method could be used for object parsing, it has not been evaluated for this purpose.

Groups of nearby contour segments are used in [17], [35] to construct features for object detection. Our paper uses similar contour fragments to construct bottom-up data driven candidates for searching the shape space. The features from [17], [35] could be further used to obtain better discriminative object models. Currently we use a generative model, with some parameters trained in a discriminative manner.

The part-based constellation model from [37] uses an extension of the contour segment network from [17] to construct object parts and a Metropolis-Hastings stochastic algorithm for inference. However, the method is used for object detection, where the precise location of the boundary is not as important as in segmentation or parsing.

Another closely related work is the unsupervised learning of shape models [18], which uses pairs of adjacent contours [17] as features and a voting scheme to find the object parameters. A separate deformation step is then performed using Thin Plate Splines. In contrast, the inference algorithm from our work optimizes a single criterion that combines the shape and deformation into a single hierarchical model. Moreover, our work is aimed towards object parsing, whereas [18] is used for object and boundary detection.

The Active Basis Model [44] can obtain a sketch of an object using Gabor filters and has been successfully used for object detection. However, it has not been

evaluated for object parsing or segmentation, and the sketch elements are not subject to a smoothness prior.

Our approach is inspired by [16], where an efficient version of the Hough transform for line detection is obtained by voting at locations given by least squares line fitting of clusters of approximately collinear pixels.

Generating candidates based on partial information is similar to the beta channel from [45], where partially occluded faces are detected by combining eye, nose and mouth detections.

3 A Hierarchical Approach to Object Parsing

We propose a hierarchical generative model with two levels of hidden variables that need to be inferred from the input data. The first level C is the actual object parsing while the second level is a PCA shape model that limits the degree of variability of the first level.

The PCA shape is controlled by variables (A, β) consisting of a similarity transformation A and the PCA coefficients $\beta \in \mathbb{R}^p$. We abuse the notation by denoting A as both the transformation parameters $A = (u, v, s, \theta)$, with rotation θ, translation (dx, dy) and scale s, and the actual transformation

$$A(x, y) = (sx \cos\theta + sy \sin\theta + u, -sx \sin\theta + sy \cos\theta + v)$$

The PCA shape is

$$S(A, \beta) = A(\mu_x + P_x\beta, \mu_y + P_y\beta) = (S_1, ..., S_N)' \quad (1)$$

where $\mu = (\mu_x, \mu_y) \in \mathbb{R}^N \times \mathbb{R}^N$ is the mean shape and $P = (P_x, P_y), P_x, P_y \in \mathcal{M}_{N,p}$ are the PCA eigenvectors ($\mathcal{M}_{N,p}$ being the space of $N \times p$ real matrices).

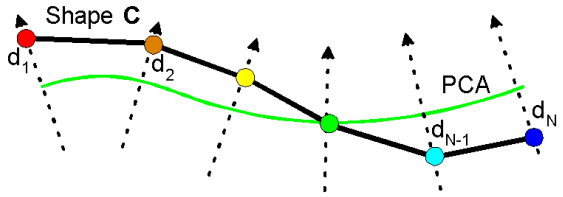

Fig. 3. The shape C is obtained as a deformation $\mathbf{d} = (d_1, ..., d_N)$ of a PCA shape (A, β) along the normals.

The representation is illustrated in Figure 3. The shape C (black) is obtained from the PCA shape (green) using a vector of displacements $\mathbf{d} = (d_1, ..., d_N) \in [-d_{max}, d_{max}]^N$ along the normals to shape. More exactly, the shape $C = C(\mathbf{d})$ consists of a number of line segments $\overline{C_i, C_{i+1}}$ where $C_i = S_i + n_i d_i, i = 1, ..., N$ and n_i is the normal to the PCA shape at S_i.

3.1 The Hierarchical Generative Model

The model can be represented either as a probability or an energy. For simplicity, we use an energy formulation of the model, illustrated in Figure 4,

$$E(C, A, \beta) = E_{data}(C) + E_{shape}(C, \beta|A) + E_p(A), \quad (2)$$

containing a data term $E_{data}(C)$ that relates the input data with the parsing result C, a shape deformation

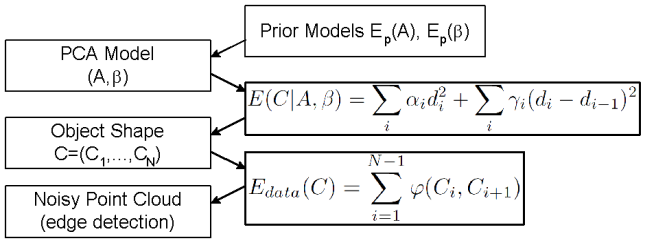

Fig. 4. Diagram of the proposed hierarchical model.

term $E_{shape}(C, \beta|A)$ and a prior $E_p(A)$ on the possible transformations A. The data term $E_{data}(C)$ is application specific and is based on the exact location of the shape $C = (C_1, ..., C_N)$. When the input data consists of noisy point clouds such as edge detection, the input points are traced into point chains based on the 8-neighborhood. The data term $E_{data}(C)$ encourages consecutive points C_i, C_{i+1} to be on the same point chain:

$$E_{data}(C) = \sum_{i=1}^{N-1} \varphi(C_i, C_{i+1}) \quad (3)$$

where $\varphi(C_i, C_{i+1}) = -\delta$ if and only if C_i, C_{i+1} are on the same point chain and $\varphi(C_i, C_{i+1}) = 0$ otherwise.

The shape term

$$E_{shape}(C, \beta|A) = E(C|A, \beta) + E_p(\beta) \quad (4)$$

consists of a deformation term $E(C|A, \beta)$ that connects the parsed shape C with the underlying PCA model with parameters (A, β), and a prior $E_p(\beta)$ on the PCA coefficients β (which are assumed to be independent of the transformation A).

The deformation term $E(C|A, \beta)$ is a Gaussian MRF [1] that encourages the curve (or curves) to be parallel and close to the PCA shape

$$E(C|A, \beta) = \sum_i \alpha_i d_i^2 + \sum_i \gamma_i (d_i - d_{i-1})^2 \quad (5)$$

and is defined in terms of the displacements d_i of the curve points C_i from the corresponding PCA shape points S_i. The coefficients α_i, γ_i represent the amount of penalty for the deformation at different points along the shape.

If the object contains multiple contour segments, they are concatenated into a single contour C and the coefficients γ_i connecting points on different contour segments will be zero. Similarly, there will be no data term $\phi(C_i, C_{i+1})$ between the contour segments.

In our applications all α_i have the same value $\alpha_i = \alpha$ and similarly $\gamma_i = \gamma$ (except those connecting different contour segments which are 0). This simplification could result in a decreased performance. For example the α_i for points on the horse head could have smaller values because there is more variability for those points.

The prior $E_p(\beta)$ on the PCA parameters is a Gaussian prior based on the PCA eigenvalues λ_i

$$E_p(\beta) = \rho \sum_{i=1}^{N} \frac{\beta_i^2}{\lambda_i} \quad (6)$$

The prior $E_p(A)$ for $A = (u, v, s, \theta)$ forces the scale and rotation within a range and discourages translations away from the image center (x_c, y_c):

$$E_p(A) = \begin{cases} \infty \text{ if } s \notin [s_{min}, s_{max}] \text{ or } |\theta| > \theta_{max} \\ r|u - x_c| + r|v - y_c| \quad \text{else} \end{cases} \quad (7)$$

The model parameters $\Theta = (\alpha, \gamma, \delta, \rho, r)$ are learned in a supervised manner on the training set through a procedure described in Section 3.5.

One advantage of the generative model described in eq. (4) is that one could easily obtain samples from the shape model $E(C|A, \beta)$, by sampling the PCA coefficients β from the Gaussian prior $\beta \sim \frac{1}{Z_1}\exp(-E_p(\beta))$ and the deformation field $\mathbf{d}$ from the Gaussian MRF $\mathbf{d} \sim \frac{1}{Z_2}\exp(-\sum_i \alpha_i d_i^2 + \sum_i \gamma_i (d_i - d_{i-1})^2)$.

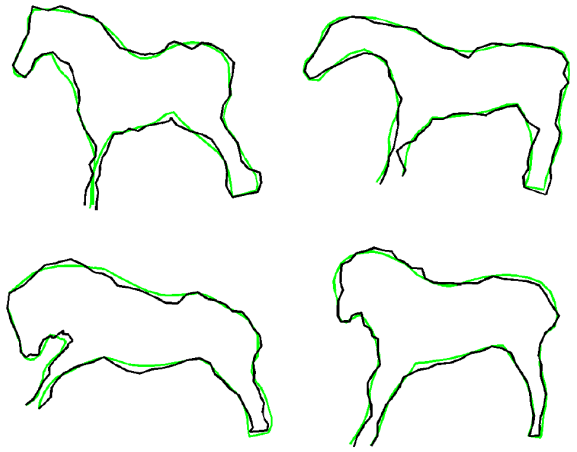

Fig. 5. Sample shapes from the shape model (4) with the parameters of the learned horse model. PCA shape (green) and sampled shape (black).

In Figure 5 are shown a few samples from the learned horse model $E(C|A, \beta)$ with the PCA shape S shown in green and the sampled shape C in black.

3.2 Inference Algorithm

Finding the object parsing C and the PCA parameters (A, β) is a nontrivial optimization problem.

The hidden variables are connected through a MRF, as illustrated in Figure 6. The PCA points (green) form a large fully connected clique in the MRF energy, and each contour point is connected to a PCA point and with its neighbors through pairwise cliques. The node labels represent the positions of the corresponding points in the image.

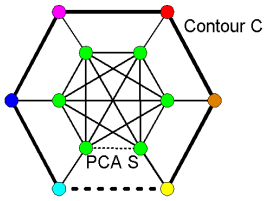

Fig. 6. The MRF interaction between the hidden variables has a large clique (between the PCA points S showed in green) and a number of binary cliques.

There exist recent advances in optimization for MRF energies with higher order cliques, such as [21] extending Graph Cuts [8] or [22], [20] based on Dual Decomposition. However, one could not even exhaust all possible combinations of labels on the nodes of the large clique, because it is computationally unfeasible even when the nodes have binary labels. For example, the large clique has 96 nodes for the horse parsing task.

We adopt a different strategy instead. If the PCA parameters (A, β) are known, the parsing C is uniquely determined by the displacement vector $\mathbf{d} = (d_1, ..., d_N)$, hence $C = C(\mathbf{d})$. In this case finding the optimal $C(\mathbf{d})$ is equivalent to finding $\mathbf{d}$ that minimizes $E(C(\mathbf{d}), A, \beta) = E(\mathbf{d})$. This can be done efficiently by dynamic programming, due to the additive nature of the model when the PCA shape (A, β) is fixed.

$$E(\mathbf{d}) = \sum_{i=1}^{N-1} \varphi(C_i, C_{i+1}) + \sum_i \alpha_i d_i^2 + \sum_i \gamma_i (d_i - d_{i-1})^2 + ct.$$

If the parsing C is fixed, an approximate minimum of $E(C, A, \beta)$ can be obtained by least square fitting of the PCA shape parameters (A, β).

Therefore, if the PCA shape parameters (A, β) are initialized close to their optimal values, an approach that alternates the above two steps, namely the computation of the parsing C by dynamic programming and the estimation of the PCA parameters (A, β) by least squares, will converge to an approximate local optimum of $E(C, A, \beta)$ in a few iterations. This approach is similar in spirit to the Active Shape Model, with the difference that a smooth contour C is found by optimization in our method instead of finding data evidence on each normal independently as the ASM does.

We will use a data-driven approach described in Section 3.3 to obtain a number of PCA candidate shapes $(A_i, \beta_i), i = 1, ..., N^{cand}$ for initialization of the local search described above. The final solution is obtained as the lowest energy configuration (C, A, β) among the N^{cand} local optima obtained. The whole optimization algorithm is described in Algorithm 1.

As the model energy (2) is just an approximation of the true object shape model, it is possible that other ways to combine the candidates such as weighted averaging [36] might be better than choosing the lowest energy one. This is subject to further investigation.

Algorithm 1 Optimization Algorithm

Input: Noisy point cloud e.g. edge detection image, PCA candidates $(A_i, \beta_i), i = 1, ..., N^{cand}$.
Output: Near-optimal hidden variables $(\hat{C}, \hat{A}, \hat{\beta})$.
for $i = 1$ **to** N^{cand} **do**
  for $j = 1$ **to** N_{iter} **do**
    Find displacement vector $\mathbf{d}$ using dynamic programming

$$\mathbf{d} = \underset{\mathbf{d}}{\text{argmin}}\, E_{data}(C(\mathbf{d})) + E(\mathbf{d}|A_i, \beta_i). \quad (8)$$

    Refit (A_i, β_i) by least squares on $C(\mathbf{d})$
  end for
  Obtain $C_i = C(\mathbf{d})$.
end for
Find $j = \text{argmin}_i\, E(C_i, A_i, \beta_i)$
Obtain $(\hat{C}, \hat{A}, \hat{\beta}) = (C_j, A_j, \beta_j)$

3.3 PCA Candidate Generation

The PCA shape candidates $(A_i, \beta_i), i = 1, ..., N^{cand}$ are obtained by matching one or more contour fragments to parts of the PCA model using the Weighted PCA method from [33], described in Section 3.3.3. The contour fragments are similar to [17], [35] and are obtained in a preprocessing step described in Section 3.4 below.

An initial set of PCA candidates can obtained from one contour fragment, as described in Section 3.3.1. If more accuracy is desired, these initial candidates can be refined by matching other contour fragments near the candidates to other parts of the PCA model, as described in Section 3.3.2.

3.3.1 Candidate Generation from One Contour Fragment

The PCA candidates are obtained by matching a contour fragment to different parts of the PCA model. A non-maximal suppression step is performed to obtain candidates that are different from each other, since too similar candidates are likely to end up in the same local energy optimum.

To speed-up computation, an interval $[L(l), U(l)]$, representing the number of PCA points that match contour fragments of length l (rounded to the nearest integer), is obtained from the training set and the ground truth annotations.

The whole procedure is described in Algorithm 2.

The Weighted PCA [33], described in Section 3.3.3, is used to fit in a least square sense a given subset of a PCA shape to a number of points $p_1, ..., p_k$.

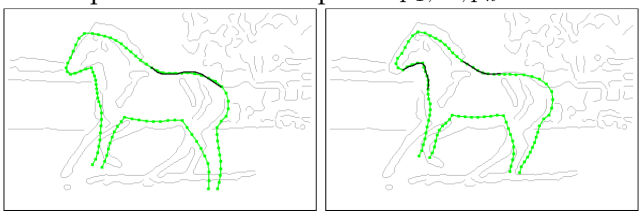

Fig. 7. The best candidate obtained from one (left) and two (right) contour fragments. The fragments that generated each candidate are shown in black.

The non-maximal suppression step finds the candidate of smallest fitting error and removes all candidates at average point-to-point distance at most D_1^{nms} from it, then adds the remaining candidate of smallest error, and so on.

For each obtained PCA shape candidate $(A_i, \beta_i), i = 1, ..., N_1^{cand}$ we also remember the contour fragment c_i and match location (b_i, k_i) that were used to generate it.

In Figure 7, left is shown the closest candidate to the ground truth among $N_1^{cand} = 200$ candidates obtained by Algorithm 2.

3.3.2 Candidate Generation from Two Contour Fragments

Usually images contain more than one contour fragment of the object to be segmented. We can refine a candidate obtained by **CG1** by fitting it simultaneously to the contour fragment it was obtained from and to another fragment close to the shape. The details of this strategy are given in Algorithm 3. In Figure 7, right is shown the closest candidate to the ground truth among $N_2^{cand} = 400$ candidates obtained by Algorithm 3. Experiments in Section 4 show that **CG2** can improve the quality of the candidates and of the final result.

Algorithm 2 CG1(N^{cand})

Input: Contour fragments c of length $len(c) \in [l_{min}, l_{max}]$
Output: At most N_1^{cand} different PCA shape candidates (A_i, β_i) with matches (c_i, b_i, k_i).
for any contour fragment c **do**
  for any k with $L(l) \le k \le U(l)$ where $l = [len(c)]$ **do**
    Subsample c evenly to have k points $p_1, ..., p_k$.
    for $1 \le b \le N$ **do**
      Fit points $b, ..., b + k - 1$ of PCA shape (A, β) to $p_1, ..., p_k$ in a least square sense.
      Discard (A, β) if the matching error is above a threshold.
    end for
  end for
end for
Perform Non-Max Suppression to keep at most N_1^{cand} candidates.

Algorithm 3 CG2(N^{cand})

Input: PCA shape candidates (A_i, β_i) with matches (c_i, b_i, k_i) from **CG1** and contour fragments c.
Output: At most N^{cand} different PCA shape candidates (A_i, β_i).
for $i = 1$ **to** N_1^{cand} **do**
  Set $(P_1, ..., P_N)' = S(A_i, \beta_i)$ from Eq. (1).
  for any contour fragment c **do**
    Find $P_j, P_k, 1 \le j, k \le N$ closest to the beginning and end of c
    if $d(c, P_j) + d(c, P_k) < 2d^{max}$ and $[j, k]$ does not overlap with $[b_i, b_i + k_i - 1]$ **then**
      Subsample c to have $m = k - j + 1$ points $p_1, ..., p_m$.
      Find PCA shape (A, β) that fits points $b_i, ..., b_i + k_i - 1$ through c_i and $j, ..., k$ through $p_1, ..., p_m$ in a least square sense.
      Discard (A, β) if the matching error is above a threshold.
    end if
  end for
end for
Perform Non-Max Suppression to keep at most N^{cand} candidates.

3.3.3 Weighted PCA

A partial PCA model can be fit to a number of points using the weighted least squares method [33], summarized

in Algorithm 4. The weights of missing PCA points are set to zero. The weighted alignment between the shapes S_1, S_2 has been described in Appendix A from [11].

Algorithm 4 FitWeightedPCA

Input: Shape $S_1 = (\mathbf{x}_1, \mathbf{y}_1)$, weight vector $\mathbf{w} = (w_1, ..., w_N)'$, $\|\mathbf{w}\|_1 = \sum_{i=1}^{N} w_i = 1$.
Output: Weighted least-square fit parameters (A, β)
Set $W = diag(w_1, ..., w_N)$
Set $K_x = (P_x' W^2 P_x)^{-1} P_x' W^2, K_y = (P_y' W^2 P_y)^{-1} P_y' W^2$
Set $\beta = 0$
for $i = 1$ **to** N_{it} **do**
Set $S_2 = (\mathbf{x}_2, \mathbf{y}_2)$, $\mathbf{x}_2 = \mu_x + P_x\beta$, $\mathbf{y}_2 = \mu_y + P_y\beta$
Solve

$$\begin{pmatrix} S_{xx}+S_{yy} & 0 & S_x & S_y \\ 0 & S_{xx}+S_{yy} & -S_y & S_x \\ S_x & -S_y & S_w & 0 \\ S_y & S_x & 0 & S_w \end{pmatrix} \begin{pmatrix} a \\ b \\ dx \\ dy \end{pmatrix} = \begin{pmatrix} S_1 \\ S_2 \\ \mathbf{x}_2'\mathbf{w} \\ \mathbf{y}_2'\mathbf{w} \end{pmatrix}$$

with $S_x = \mathbf{x}_1'\mathbf{w}, S_y = \mathbf{y}_1'\mathbf{w}, S_w = \|\mathbf{w}\|_1 = 1$
$S_{xx} = \mathbf{x}_1' W \mathbf{x}_1, S_{yy} = \mathbf{y}_1' W \mathbf{y}_1$
$S_1 = \mathbf{x}_1' W \mathbf{x}_2 + \mathbf{y}_1' W \mathbf{y}_2$
$S_2 = \mathbf{y}_1' W \mathbf{x}_2 - \mathbf{x}_1' W \mathbf{y}_2$
Obtain $A(\mathbf{x}, \mathbf{y}) = (a\mathbf{x} + b\mathbf{y} + dx, -b\mathbf{x} + a\mathbf{y} + dy)$
Find $(\mathbf{x}_o, \mathbf{y}_o) = A^{-1}(S_1)$
Set $\beta = K_x(\mathbf{x}_o - \mu_x) + K_y(\mathbf{y}_o - \mu_y)$
end for
Set $s = \sqrt{a^2 + b^2}, \theta = \arctan(b/a)$.
Obtain $A = (dx, dy, s, \theta)$

3.4 Preprocessing

Preprocessing begins with tracing the input points into point chains based on the 8-neighborhood. The point chains are then subsampled every 5-6 pixels to reduce the number of contour fragments obtained, as illustrated in Figure 8, left.

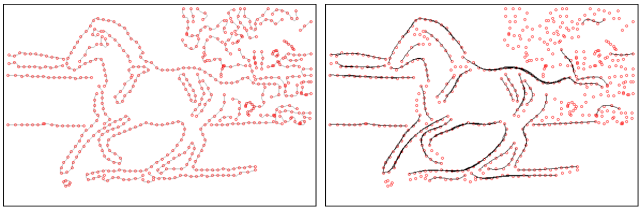

Fig. 8. Left: the input points are traced into point chains and subsampled every 5-6 pixels. Right: smooth contour fragments (black) are fitted through the point chains starting and ending in the subsampled pixels.

The contour fragments used by the candidate generators are represented as a polynomials of degree three relative to a system of coordinates aligned with the contour's endpoints, as illustrated in Figure 9.

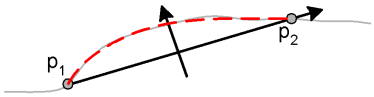

Fig. 9. A contour fragment (red dashed) is a polynomial fit of a subset of a chain of points (shown in gray).

The contour fragment endpoints are two of the subsampled points of the same traced point chain and the polynomial is fitted in a least square sense through all the chain points in between. The fragments are restricted in length to a range $[l_{min}, l_{max}]$. Only the fragments with a maximum error at most $e_{max} = 1.5$ pixels are kept.

The contour fragments obtained this way have a partial order inherited from the partial order between the sets of chain points they were constructed from.

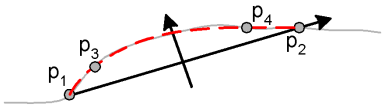

Fig. 10. The contour fragment between p_3 and p_4 is a subset of the contour fragment between p_1 and p_2, i.e. $\overline{p_3p_4} \subset \overline{p_1p_2}$.

For example $\overline{p_3p_4} \subset \overline{p_1p_2}$ in Figure 10. Based on this partial order, non-maximal contour fragments (e.g. $\overline{p_3p_4}$ from Figure 10) are removed.

An example of obtained contour fragments is shown in Figure 8, right.

3.5 Learning the Model and Algorithm Parameters

The proposed model is very simple. It consists of a PCA model that in practice has at most 10 principal directions plus a small number (<20) of parameters. Because the model is small, it should be expected that it generalize well to unseen data if the training data is representative.

The PCA model is learned in the standard way using Procrustes analysis to align the training shapes.

To learn the rest of the parameters, we adopt a supervised approach previously used successfully in other MRF-based methods [3], [28], [34], namely learning the parameters by optimizing a loss function on the training set. To speed-up the parameter learning, for candidate generators we employ loss functions that directly evaluate the generated candidates instead of the final result.

The parameters of the candidate generators are learned first, in the order **CG1** and **CG2**, using the minimum of the average point-to-point distances from the candidates to the ground truth annotation (described in Section 4) as loss function. This speeds-up the learning process since the **CG** parameters are this way decoupled from the later modules. Other measures, such as detection rate/false positive rate for the contour fragments, could be used instead and are subject to further investigation. The number of PCA components was fixed to $p = 4, 8$ for **CG1** respectively **CG2** except for the faces where we used $p = 2, 4$ for **CG1** respectively **CG2**.

Similar to [3], we adopted a coordinate descent optimization of the loss function, where at each step one parameter is perturbed by an increment and the change is kept if the loss function decreases. For **CG1** and **CG2** we restricted the number of candidates to balance speed and accuracy. The obtained parameters for **CG1** are $l_{min} = 20, l_{max} = 60, N_1^{cand} = 200, D_1^{nms} = 5$, and $N_2^{cand} = 400, D_2^{nms} = 8, d^{max} = 20$ for **CG2**.

In Figure 11 are shown the average errors of the closest candidate obtained by **CG1-2** on the training and test sets vs the number of candidates N^{cand}.

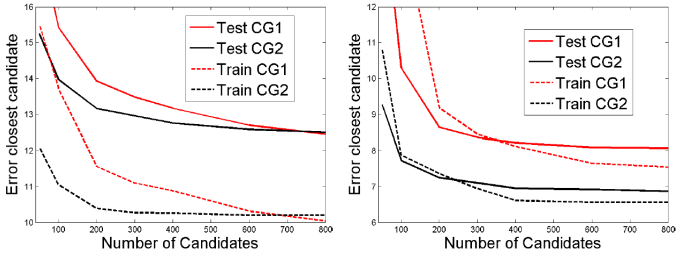

Fig. 11. Candidate generator error vs. number of candidates for the horse (left) and cow (right) datasets.

Figure 11 shows that the test error decreases as the the training error decreases, which means that there is minimal overfitting for the candidate generators. Similar behaviors were observed for the D_1^{nms}, D_2^{nms} parameters and for the d^{max} parameter of **CG2**. It is worth noting that the candidates from **CG2** are usually better than those of **CG1** (closer to the ground truth), and the difference is larger for the cow images.

The model parameters $\Theta = (\alpha, \gamma, \delta, \rho, r)$ are learned based on the average point-to-point distance between the obtained parsing results and the ground truth annotation.

$$Err(\Theta) = \frac{1}{n}\sum_{i=1}^{n} err_i(\Theta) \tag{9}$$

where $err_i(\Theta)$ is the average point-to-point error of the parsing result obtained with parameters Θ on example i using **CG1**.

TABLE 1
Learned parameters for Algorithm 1.

Dataset	α_i	δ	ρ	r	p
Weizmann horses [6]	0.04	2	2	1	10
Cows [24]	0.04	2	2	0.5	10
IMM Faces [39], [40]	0.04	6	2	0.5	5

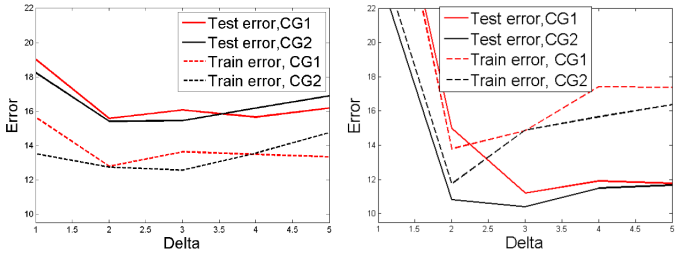

Fig. 12. The parsing error measure (9) vs δ for the horses (left) and cows (right).

The model parameters $\delta, \rho, \alpha_i = \alpha, r, p$ were obtained by optimizing the error measure (9) on the training set by coordinate descent, with fixed parameters $\gamma_i = 0.1$ (to fix the scale of the energy function (2)) and $N_{iter} = 10$. The obtained values are given in Table 1.

The dependence of the error on the values of δ is shown in Figure 12 for the horses and cows. Again, the test errors follow the training error.

4 Experimental Results

We evaluated this model and algorithm combination on three datasets: the Weizmann dataset [6] containing 328 horse images with object segmentations as binary masks, the Cows dataset [24] with 111 cow images and the IMM face dataset [39], [40]. We used the same subsets of images as [46] for training and testing the Weizmann dataset and the first 25 images for training the Cows dataset and tested on all 111 images.

Many works [7], [12], [23], [25], [32], [43], [46] report segmentation results on the Weizmann horses in terms of percentage of correctly classified pixels. However, object segmentation is a different problem than object parsing since a pixel segmentation has no information on the position of the object parts. Moreover, these works make use of the intensity information in different ways in obtaining the segmentation. We cannot do the same in our problem as we have no intensity information available from the edge detection images.

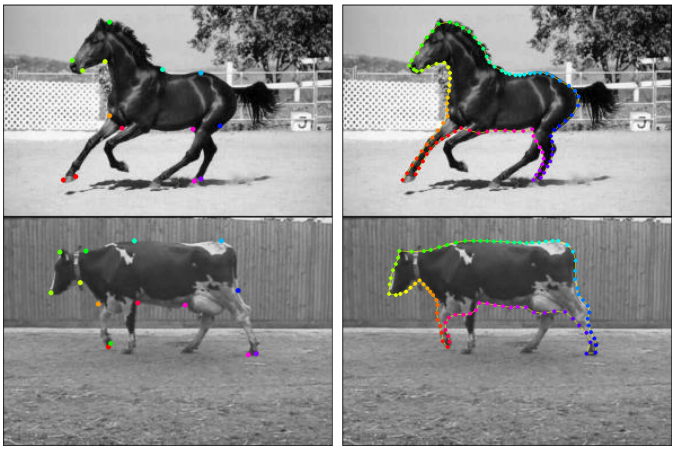

Fig. 13. Left: The horses and cows are annotated with 14 control points on the contour and the two outer legs. Smooth curves (yellow) are obtained between the control points using the ground truth segmentation. Right: The obtained boundary annotation with 96 points (horses) and 87 points (cows).

Each horse and cow was manually annotated with 14 control points on the boundary, as illustrated in Figure 13, left. For fairness of comparison, the same horse and cow legs (usually the outer legs) were annotated as in [46]. This is not an extreme case. If the legs for annotation were chosen at random, there would be larger shape variability and a decreased performance. If the legs were chosen to minimize the shape variability, a better performance could be obtained.

Smooth curves were obtained between the control points by dynamic programming to minimize the average distance to the object boundary from the binary mask. Intermediate points were obtained by dividing the smooth curves into equal parts. Examples of obtained annotations are shown in Figure 13, right, with 96-points for the horses and 87 points for the cows.

We also evaluated a standard Active Shape Model [11] initialized in the center of the image with average scale, no rotation $\theta = 0$, and 20 update iterations.

The results are summarized in Tables 2, 3 and 4. In Fig. 14 are plotted the sorted errors on the datasets, from which different error percentiles can be obtained.

TABLE 2
Performance of different methods on the Weizmann Horse dataset.

Method	Train images	Test images	Contour points	Train error	Test error	Time/img (sec)
Active Shape Model [11]	50	227	96	25.35	29.05	<1
Recursive Compositional Models [46]	1	227	27	-	18.7	3
Recursive Compositional Models [46]	50	227	27	-	16.04	23
Ours, with CG1	50	227	96	12.79	15.58	44
Ours, with CG2	50	227	96	**12.74**	**15.36**	69
Ours, with CG2, no head or legs	50	227	60	8.21	11.42	20

TABLE 3
Performance of different methods on the Cows dataset.

Method	Train images	Test images	Contour points	Train error	Test error	Time/img (sec)
Active Shape Model [11]	25	111	87	48.81	49.23	<1
Recursive Compositional Models [46]	1	111	27	-	15.8	3.5
Ours, with CG1	25	111	87	13.78	14.98	14
Ours, with CG2	25	111	87	**11.73**	**10.81**	28

TABLE 4
Performance of different methods on the IMM Face dataset at 320×240 resolution.

Method	Number of images	Uses intensity	Automatic initialization	Crossval. folds	Contour points	Train error	Test error	Time/img (sec)
Active Shape Model [11]	40	No	Yes	4	58	21.47	21.56	0.08
Stegman [40]	37	B/W	No	37	58	-	3.14	0.13
Stegman [40]	37	Color	No	37	58	-	**3.08**	0.28
Ours, with CG1	40	No	Yes	4	58	6.54	6.64	0.33
Ours, with CG2	40	No	Yes	4	58	5.30	5.57	0.43

The Recursive Compositional Model [46] also reports average point-to-point errors on the Weizmann and cow datasets but uses both edge and intensity information, unlike our approach, which only uses edge information. Table 2 shows two results from [46]. The first result and the cow result from Table 3 are based on models trained on a single image without annotation, which are not as good as models trained with manual annotation. The second result from [46] shown in Table 2 uses a model trained on 50 images manually annotated with 27-point contours similar to ours.

The face model is evaluated on the IMM face dataset [39], [40] containing 240 face images of 40 people, each with six different poses, varying illumination and expressions. The AAM model from [40] is evaluated on 37 out of the 40 frontal faces (probably excluding the three grayscale images). We evaluate our model on all 40 frontal faces of the dataset with four fold cross-validation. The results are summarized in 4. The best AAM errors from [40] are 3.08 using color information while our algorithm with **CG2** obtains 5.57. However, our algorithm does not use any intensity information and is fully automatic, whereas [40] initializes their algorithms at locations close to the true location and report the error after convergence. The argument is that a face detector can be used to initialize the AAM. However, both the face detector and the AAM use intensity information, so they cannot work on the edge detection images. We are not aware on any fully automatic face alignment results on this dataset. The Recursive Compositional Model [46] reports 6 pixel average point-to-point errors on the face dataset [26], so comparable or better results should be expected for the IMM face dataset.

One factor that greatly influences performance is our data term that works best when the contour points are close to each other (about 5 pixels or so), because when the contour points are far from each other it is more likely that the edge chain is broken between consecutive points. This disadvantage could be alleviated by increasing the density of the contour points, as we did for the horses and cows, or by using a more elaborate data term that also considers the gaps between contours.

The PCA model has difficulties modeling the shape variability of the horse head and legs. If the head and leg points are removed both from the model and from the evaluation, the training and test errors decrease substantially, as it can be seen in the last row of Table 2. This experiment suggests that part-based shape models with free parameters for the head and leg positions might be more appropriate than the PCA for the higher level model. Such models are subject to further investigation.

Parsing examples using **CG2**, are shown in Figure 15.

5 Conclusion and Future Work

This paper proposes a novel approach to object parsing and applies it to data coming from structured noisy point clouds such as edge detection images. The object shape is modeled as a MRF deformation of a hidden PCA model. The model energy is minimized through many local searches starting from a number of data-driven initializations. Based on the experimental evaluation we conclude that the proposed model is quite accurate,

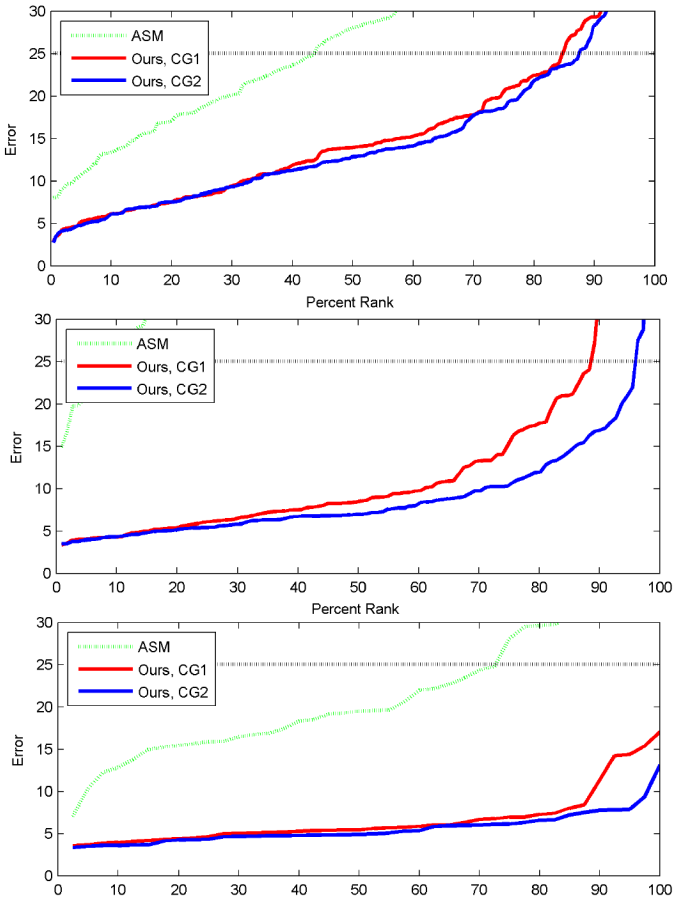

Fig. 14. The sorted errors (9) of our algorithm on the 227 test images from the Weizmann dataset (top), on the 111 images of the cow dataset (middle) and on the 40 frontal images of the IMM face dataset (bottom).

and even though the inference algorithm is suboptimal, our method is competitive with modern approaches for object parsing from point clouds such as the Recursive Compositional Models [46] and Active Shape Models [11].

The candidate generators and the object parsing algorithm can be easily parallelized, expecting a 10-100 times speedup from a GPU implementation.

In the future, we plan to investigate more accurate three-level part-based models, with separate parameters for the head and leg shapes and positions. We also plan to extend the method to 3D object parsing using approximate inference methods such as Graph Cuts or Belief Propagation for the deformation inference.

REFERENCES

[1] Y. Amit, U. Grenander, and M. Piccioni. Structural image restoration through deformable templates. *Journal of the American Statistical Association*, pages 376–387, 1991.

[2] X. Bai, X. Wang, L. Latecki, W. Liu, and Z. Tu. Active skeleton for non-rigid object detection. In *ICCV*, pages 575–582, 2009.

[3] A. Barbu. Training an Active Random Field for Real-Time Image Denoising. *IEEE Trans. Image Processing*, 18(11):2451–2462, 2009.

[4] G. Behiels, D. Vandermeulen, F. Maes, P. Suetens, and P. Dewaele. Active shape model-based segmentation of digital x-ray images. In *MICCAI*, pages 128–137, 1999.

[5] A. Besbes, N. Komodakis, G. Langs, and N. Paragios. Shape priors and discrete mrfs for knowledge-based segmentation. In *CVPR*, pages 1295–1302, 2009.

[6] E. Borenstein and S. Ullman. Class-specific, top-down segmentation. *ECCV*, pages 639–641, 2002.

[7] E. Borenstein and S. Ullman. Combined Top-Down/Bottom-Up Segmentation. *IEEE Trans. PAMI*, 30(12):2109–2125, 2008.

[8] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. PAMI*, pages 1222–1239, 2001.

[9] H. Chui and A. Rangarajan. A new point matching algorithm for non-rigid registration. *Computer Vision and Image Understanding*, 89(2):114–141, 2003.

[10] T. Cootes, G. Edwards, and C. Taylor. Active appearance models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(6):681–685, 2001.

[11] T. Cootes, C. Taylor, D. Cooper, J. Graham, et al. Active shape models-their training and application. *CVIU*, 61(1):38–59, 1995.

[12] T. Cour and J. Shi. Recognizing objects by piecing together the segmentation puzzle. In *CVPR*, 2007.

[13] D. Cristinacce and T. Cootes. Automatic feature localisation with constrained local models. *Pattern Recognition*, 41(10):3054–3067, 2008.

[14] A. Falcão, J. Udupa, S. Samarasekera, S. Sharma, B. Hirsch, and R. Lotufo. User-steered image segmentation paradigms: Live wire and live lane. *Graphical models and image processing*, 60(4):233–260, 1998.

[15] P. Felzenszwalb and J. Schwartz. Hierarchical matching of deformable shapes. In *CVPR*, 2007.

[16] L. Fernandes and M. Oliveira. Real-time line detection through an improved Hough transform voting scheme. *Pattern Recognition*, 41(1):299–314, 2008.

[17] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid. Groups of adjacent contour segments for object detection. *IEEE Trans. PAMI*, pages 36–51, 2007.

[18] V. Ferrari, F. Jurie, and C. Schmid. From Images to Shape Models for Object Detection. *IJCV*, 87(3):284–303, 2010.

[19] D. Freedman and T. Zhang. Interactive graph cut based segmentation with shape priors. In *CVPR 2005*, pages 755–762.

[20] V. Jojic, S. Gould, and D. Koller. Accelerated dual decomposition for map inference. In *Proc. of ICML*, 2010.

[21] P. Kohli, L. Ladický, and P. Torr. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision*, 82(3):302–324, 2009.

[22] N. Komodakis and N. Paragios. Beyond pairwise energies: Efficient optimization for higher-order mrfs. In *CVPR*, pages 2985–2992.

[23] M. Kumar, P. Torr, and A. Zisserman. Obj cut. In *CVPR*, pages 18–25, 2005.

[24] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision, ECCV*, pages 17–32, 2004.

[25] A. Levin and Y. Weiss. Learning to combine bottom-up and top-down segmentation. *IJCV*, 81, 2009.

[26] H. Li, S. Yan, and L. Peng. Robust non-frontal face alignment with edge based texture. *Journal of Computer Science and Technology*, 20(6):849–854, 2005.

[27] Y. Li, L. Gu, and T. Kanade. A robust shape model for multi-view car alignment. In *CVPR*, 2009.

[28] Y. Li and D. P. Huttenlocher. Learning for optical flow using stochastic optimization. *ECCV*, 2008.

[29] J. Liu and J. Udupa. Oriented active shape models. *Medical Imaging, IEEE Transactions on*, 28(4):571–584, 2009.

[30] J. Malcolm, Y. Rathi, and A. Tannenbaum. Graph cut segmentation with nonlinear shape priors. In *ICIP*, pages IV–365, 2007.

[31] A. Rangarajan, H. Chui, and F. Bookstein. The softassign procrustes matching algorithm. In *Information Processing in Medical Imaging*, pages 29–42. Springer, 1997.

[32] X. Ren, C. Fowlkes, and J. Malik. Cue integration for figure/ground labeling. *NIPS*, 18:1121, 2006.

[33] M. Rogers and J. Graham. Robust active shape model search. *ECCV*, pages 289–312, 2006.

[34] D. Scharstein and C. Pal. Learning conditional random fields for stereo. In *CVPR*, pages 1–8, 2007.

[35] J. Shotton, A. Blake, and R. Cipolla. Multi-Scale Categorical Object Recognition Using Contour Fragments. *IEEE Trans. PAMI*, 2008.

[36] M. Sofka, J. Zhang, S. Zhou, and D. Comaniciu. Multiple object detection by sequential monte carlo and hierarchical detection network. In *CVPR*, pages 1735–1742.

[37] M. Stark, M. Goesele, and B. Schiele. A shape-based object class model for knowledge transfer. In *ICCV*, pages 373–380, 2010.

[38] M. B. Stegmann. Active appearance models: Theory, extensions

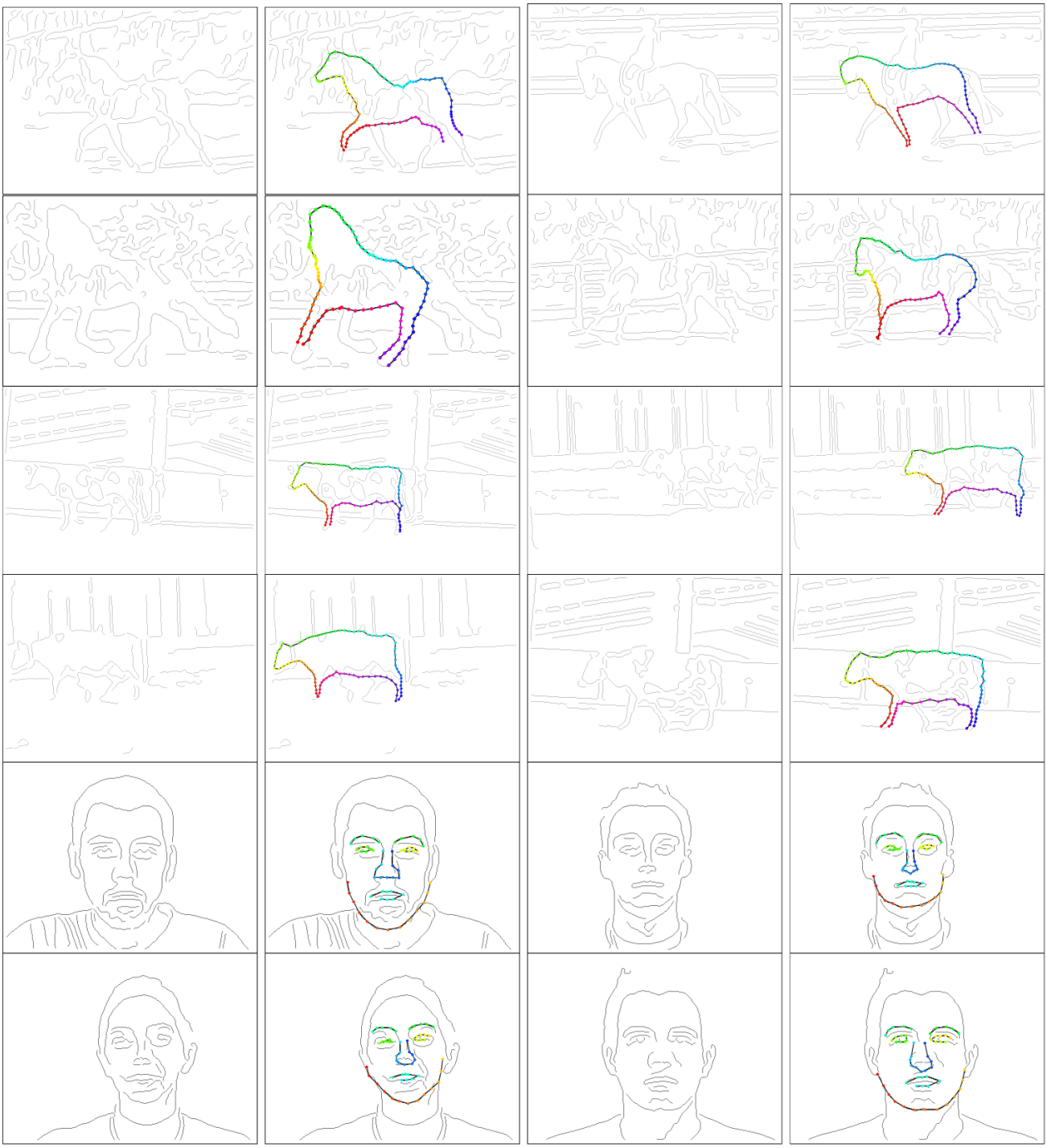

Fig. 15. Example of results on the horse, cow and face datasets.

and cases, Aug 2000.
[39] M. B. Stegmann. Analysis and segmentation of face images using point annotations and linear subspace techniques. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, 2002.
[40] M. B. Stegmann, B. K. Ersbøll, and R. Larsen. FAME – a flexible appearance modelling environment. *IEEE Trans. on Medical Imaging*, 22(10):1319–1331, 2003.
[41] L. Torresani, A. Hertzmann, and C. Bregler. Learning non-rigid 3d shape from 2d motion. In *NIPS*, 2003.
[42] P. Tresadern, H. Bhaskar, S. Adeshina, C. Taylor, and T. Cootes. Combining local and global shape models for deformable object matching. In *Proceeding of the BMVC*, 2009.
[43] J. Winn and N. Jojic. Locus: Learning object classes with unsupervised segmentation. In *ICCV*, 2005.
[44] Y. Wu, Z. Si, H. Gong, and S. C. Zhu. Learning active basis model for object detection and recognition. *IJCV*, pages 1–38, 2009.
[45] X. Yang, T. Wu, and S. C. Zhu. Evaluating information contributions of bottom-up and top-down processes. In *ICCV*, pages 1042–1049, 2010.
[46] L. Zhu, Y. Chen, and A. Yuille. Learning a hierarchical deformable template for rapid deformable object parsing. *IEEE Trans. PAMI*, 2009.
[47] Q. Zhu, G. Song, and J. Shi. Untangling cycles for contour grouping. In *ICCV*, pages 1–8, 2007.

Adrian Barbu received his BS degree from University of Bucharest, Romania, in 1995, a Ph.D. in Mathematics from Ohio State University in 2000 and a Ph.D. in Computer Science from UCLA in 2005. From 2005 to 2007 he was a research scientist and later project manager in Siemens Corporate Research, working in medical imaging. He received the 2011 Thomas A. Edison Patent Award with his co-authors for their work on Marginal Space Learning. Since 2007 he has been an assistant professor in Statistics at Florida State University. His research interests are in medical imaging, computer vision and machine learning.