Exemplar-Based Face Parsing

Brandon M. Smith Li Zhang
University of Wisconsin–Madison

Jonathan Brandt Zhe Lin Jianchao Yang
Adobe Research

http://www.cs.wisc.edu/~lizhang/projects/face-parsing/

Abstract

In this work, we propose an exemplar-based face image segmentation algorithm. We take inspiration from previous works on image parsing for general scenes. Our approach assumes a database of exemplar face images, each of which is associated with a hand-labeled segmentation map. Given a test image, our algorithm first selects a subset of exemplar images from the database, Our algorithm then computes a non-rigid warp for each exemplar image to align it with the test image. Finally, we propagate labels from the exemplar images to the test image in a pixel-wise manner, using trained weights to modulate and combine label maps from different exemplars. We evaluate our method on two challenging datasets and compare with two face parsing algorithms and a general scene parsing algorithm. We also compare our segmentation results with contour-based face alignment results; that is, we first run the alignment algorithms to extract contour points and then derive segments from the contours. Our algorithm compares favorably with all previous works on all datasets evaluated.

1. Introduction

In face image analysis, one common task is to parse an input face image into facial parts, *e.g.*, left eye and upper lip. Most previous methods accomplish this task by marking a few landmarks [1, 22] or a few contours [4, 18] on the input face image. In this paper, we seek to mark each pixel on the face with its semantic part label; that is, our algorithm parses a face image into its constituent facial parts.

Compared to segment-based representations, we argue that landmark- and contour-based representations have several key limitations.

- Other than eye corners and mouth corners, most landmarks are *not* well-defined. For example, it is unclear how many landmarks should be defined on the chinline, or how noses should be represented: should there be a line segment along the nose ridge, or a contour around the nostrils? Due to the lack of agreement, different datasets have different contour models. This creates difficulty for practitioners interested in unifying different datasets for robust algorithm development.

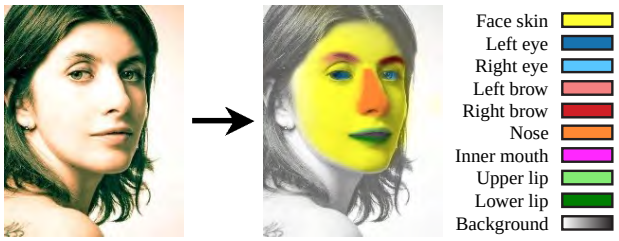

Figure 1. Our exemplar-based algorithm parses a face image into its constituent facial parts using a soft segmentation.

- Contour-based representations are not general enough to model several facial parts useful for robust face analysis. For example, teeth are important cues for analyzing open-mouth expressions; ears are important cues for analyzing profile faces; strands of hair are often confused by algorithms as occluders. It would be difficult to model these important parts using contours, and it is unclear how many landmark points should be used.

- It is difficult to encode uncertainty in contour-based representations. For example, the precise location of the tip of an eyebrow or the contour of a nose ridge are difficult to determine, even for human labelers. Such uncertainty leads to errors in human labeled face data that are used in both the training and evaluation of algorithms.

Segment-based representations alleviate the aforementioned limitations: segments can represent any facial part, be they hair or teeth, and soft segmentation can model uncertain transitions between parts. Although semantic segmentation for general scenes has received tremendous attention in recent years [7, 8, 12, 19], there has been relatively little attention given specifically to face part segmentation, with the exception of [15, 21]. Since facial parts have special geometric configurations compared to general indoor and outdoor scenes, we propose an exemplar-based face image segmentation algorithm, taking inspiration from previous work in image parsing for general scenes.

Specifically, our approach assumes a database of face images, each of which is associated with a hand-labeled segmentation map and a set of sparse keypoint descriptors. We emphasize that these keypoints need not correspond between dif-

ferent images in the database; we extract keypoints and their descriptors independently from each image using SIFT [14]. Given a test image, our algorithm first employs Belhumeur *et al*. [1] to select m top exemplar images from the database as input. Our algorithm then computes a nonrigid warp for each top exemplar; each nonrigid warp aligns the exemplar image to the test image by matching the set of sparse precomputed exemplar keypoints to the test image. Finally, we propagate labels from the exemplar images to the test image in a pixel-wise manner, using trained weights that modulate and combine label maps differently for each part type.

We evaluate our method on two challenging datasets [6, 9] and compare with two face parsing algorithms [15, 21] and a general scene parsing algorithm [12]. We also compare our segmentation results with contour-based face alignment results: that is, we first run the alignment algorithms [4, 18, 22] to extract contour points and then derive segments from the contours. Our algorithm compares favorably with these previous works on all datasets tested. As a byproduct, our algorithm can recover contours of facial parts as well, although we do not focus on this representation. In summary, this paper makes the following contributions.

1. *A novel algorithm for robustly segmenting face parts.* We recover a soft segmentation, which naturally encodes the segment class uncertainty in the image. Rather than artificially placing a hard boundary between, *e.g*., skin and eyebrow regions, we recover label probabilities at each pixel. Our algorithm is exemplar-based; exemplars can also encode continuous, probabilistic segment labels.

2. *A learning algorithm for finding optimal parameters for calibrating exemplar label types.* Some part labels occur more frequently than others (*e.g*., "skin" labels occur more frequently than "eye" labels), and tend to dominate adjacent labels that occur less frequently. To correct these biases, we train a set of label weights that adjust the relative importance of each label type.

3. *A rich training dataset for face segmentation.* Our dataset is built as an extension of the recent Helen Facial Feature Dataset [9]. The images are high resolution, and our dataset features segments that are created from densely-sampled, hand-labeled contours. Hair mattes are also included for future work in hair segmentation.

2. Related Work

We are inspired by the recent work of Luo *et al*. [15], who proposed a hierarchical technique for face parsing. In their work, they recover pixel-wise labels for eyes, eyebrows, nose, mouth, and background, which includes the skin. Their approach can be divided into three stages: (1) detect face parts (*e.g*., upper face, lower face), (2) detect face components (*e.g*., mouth, eyes), and (3) use component-specific segmentors on each component to estimate pixel-wise labels. There are two aspects of this approach that can be improved. First, because they operate at the lowest level, the component-specific segmentors do not generalize well to labeling larger and/or less distinct regions of the face, such as the cheeks or the chin. Second, because they produce only a binary classification, the component-specific segmentors do not generalize well to more complicated label interactions, such as those that exist between the inner mouth region, the lips, and the skin around the lips, for example. Instead, we propose a nonparametric approach that naturally extends to these difficult cases.

Previously, Warrell and Prince [21] introduced *LabelFaces* based on a scene parsing approach applied to faces. Warrell and Prince argued that the scene parsing approach is advantageous because it is general enough to handle unconstrained face images, where the shape and appearance of features vary widely and relatively rare semantic label classes exist, such as moustaches and hats. As part of their contribution, they introduced priors to loosely model the topological structure of face images (so that mouth labels do not appear in the forehead, for example). However, the labels they generate are often coarse and inaccurate, especially for small face components like eyes and eyebrows. We show in this work that our approach produces accurate, fine-scale label estimates in unconstrained face images.

There are several recent scene parsing approaches in the literature that do not target faces, but nonetheless do share some aspects with our approach ([7, 8, 12, 19] to name just a few). A full review is outside the scope of this paper. Of these approaches, Liu *et al*. [12] is particularly relevant. Like our approach, they propose a nonparametric system that transfers labels from exemplars in a database to annotate a test image. At its core, Liu *et al*.'s system relies on the SIFT Flow [13] algorithm to densely transfer labels from exemplars to the test image. Unfortunately, SIFT Flow is slow, even with a coarse-to-fine strategy. For example, [13] reported a runtime of 31 seconds for 256×256-pixel image pairs with a C++ implementation running on modern hardware.

Targetting face image matching, we employ an efficient sparse matching approach that does not require global optimization, and therefore requires much less computation for each image pair. This savings allows us to use a large set of top exemplar images for label transfer (in [12] they use $m \leq 9$ top exemplar images; we use $m = 100$), which is important in our approach for two reasons. First, by aggregating label votes from many exemplars, our approach is robust to outliers and noise. Second, it partially explains why we can avoid global optimization of the flow field: by using a large number of top exemplars, the sparse keypoint matches cover the test image well and good matches occur almost everywhere.

Furthermore, our algorithm produces soft segmentations while [12] produces hard segmentations; specifically, our algorithm assigns each pixel a probability value for each label type. To this end, we propose a training algorithm for estimat-

ing a set of weights that convert label maps from exemplars to label probabilities on the test image. We remark that soft segmentation is useful for future work on hair segmentation, among other applications.

3. Our Approach

In this section we first give an overview of our approach and then present technical details of each step. We start with some notation. Let $\mathbf{p}_i$ be a probability vector for pixel i:

$$\begin{aligned} \mathbf{p}_i &= [p_{i,1}, p_{i,2}, \ldots, p_{i,K}]^\mathsf{T} \\ \text{s.t.} &\quad \sum_k^K p_{i,k} = 1, \quad 0 \le p_{i,k} \le 1, \end{aligned} \tag{1}$$

where $p_{i,k}$ is the probability that pixel i belongs to segment class k. Each $\mathbf{p}_i$ encodes label uncertainty at the pixel level, which reflects the natural indistinctness of some facial features (*e.g.*, light eyebrows, chin line). We seek to estimate $\mathbf{p}_i$ for all pixels $i = 1, 2, \ldots, N$ in the test image I.

3.1. Overview

Database Construction Our database $\mathcal{M}$ is composed of a set of exemplars $\{M_j\}_{j=1}^J$. Each exemplar M_j has four parts: an image, a label map, a very sparse set of facial landmark points, and a sparse set of SIFT [14] keypoint descriptors. Each label map can either be soft (*i.e.*, so that each $\mathbf{p}_i$ has several non-zero components), or hard (*i.e.*, so that each $\mathbf{p}_i$ has exactly one nonzero component). We use 12 landmark points: 2 mouth corners, 4 eye corners, 2 points on the eyebrows (each centered on the top edge), 2 points on the mouth (one on the top edge of the upper lip and one on the bottom edge of the bottom lip), 1 point between the nostrils, and 1 chin point. About 150 SIFT keypoints are automatically extracted from each image independently in the database. We make a distinction here between *landmarks*, which are defined consistently across images (*e.g.*, the mouth corners), and *keypoints*, which are not necessarily consistent across images. Each SIFT keypoint descriptor is computed using a window radius of approximately 1/4 the inter-ocular distance (IOD).

Runtime Pre-processing Given a test image, we first use a face detector (*i.e.*, [20]) to roughly locate the face and estimate its scale. The test image is then rescaled so that the face has an IOD of approximately 55 pixels, which is the size of the exemplar faces. Second, *dense* SIFT decriptors [14] are extracted using the same window size (1/4 IOD) across all pixels. To search for a subset of m top exemplar faces in the database, we use Belhumeur *et al*. [1] on our 12 landmark points. The output of the pre-processing is a set of m exemplars, each of which is associated with a similarity transformation that aligns the exemplar to the face in the test image.

Step 1: Nonrigid exemplar alignment For each keypoint in each of the top m exemplars, search within a small window in the test image to find the best match; record the matching score and the location offset of the best match for each keypoint. Warp the label map of each top exemplar nonrigidly using a displacement field interpolated from the location offsets.

Step 2: Exemplar label map aggregation Aggregate warped label maps using weights derived from the keypoint matching scores in Step 1. The weights are spatially varying among exemplar pixel locations and favor exemplar pixels near keypoints that are matched well with the test image.

Step 3: Pixel-wise label selection Produce a label probability vector at each pixel by first attenuating each channel in the aggregated label map and then normalizing it. The attenuating weights are trained offline in order to correct for label population biases and maximize labeling accuracy. Hard segmentation can be generated by selecting the highest probability label channel.

3.2. Step 1: Nonrigid Exemplar Alignment

Due to local deformation, a similarity transformation is not sufficient to align an exemplar with the testing image. The goal of Step 1 is to refine the registration using a nonrigid warp between each top exemplar label map and the test image. Dense per-pixel correspondence algorithms like SIFT Flow [13] are one strategy for this purpose. However, per-pixel correspondence algorithms often perform global optimization, which is computationally intensive and does not scale well to the many exemplars we use for our task. In particular, we would like to aggregate a large number of top exemplars (we use 100) in order to be robust against outliers in one or a small number of exemplars. Therefore, for efficiency reasons, we instead rely on about 150 SIFT keypoints to compute the nonrigid warp between each exemplar and the test image.

Given the similarity transformation estimated in the preprocessing step, for each keypoint in one top exemplar, its true correspondence in the test image is usually within a small window centered at the location predicted by the similarity transformation. Therefore, we adopt a local search within the window to find its best match. To make the search robust to untextured regions, we encourage the best match to be close to the window center. Specifically, for each keypoint f with SIFT descriptor $\mathbf{s}_f$, we search for the location offset $\Delta\mathbf{x}_f$ that produces the best match in the window by using the following objective function:

$$r(\Delta\mathbf{x}_f) = g_{\mathsf{spatial}}\left(\frac{\Delta\mathbf{x}_f}{\sigma_{\mathsf{spatial}}}\right) \cdot g_{\mathsf{desc}}\left(\frac{\mathbf{s}(\Delta\mathbf{x}_f) - \mathbf{s}_f}{\sigma_{\mathsf{desc}}}\right), \tag{2}$$

where g_{spatial} and g_{desc} are Gaussian functions and $\mathbf{s}(\Delta\mathbf{x}_f)$ is the SIFT descriptor at the offset location. In our implementation, we set $\sigma_{\mathsf{spatial}} = 10$ to be the same as the search window radius and $\sigma_{\mathsf{desc}} = a \cdot b$, where a is the length of the SIFT descriptor and b is the scale of the descriptor elements.

Our algorithm computes a nonrigid warp for each exemplar label map by interpolating the displacements $\{\Delta\mathbf{x}_f\}_{f=1}^{F}$, where F is the number of SIFT keypoints in the exemplar. The interpolation is implemented using a linear combination of Gaussian Radial Basis Functions (RBF) [17] centered at each keypoint, where each RBF bandwidth is proportional to the distance to the nearest neighboring keypoint.

3.3. Step 2: Exemplar Aggregation

For each exemplar label map, we interpolate the matching scores $r(\Delta\mathbf{x}_f)$ in Eq. (2) between its keypoints to generate a matching score for each pixel; the interpolation uses the same Gaussian RBFs as we use at the end of Step 1 to generate the nonrigid warping field. Note that $r(\Delta\mathbf{x}_f) \in [0, 1]$, where higher values suggest better matches. Now, each nonrigidly warped exemplar label map is associated with a per-pixel matching score map. We aggregate these label maps by taking a weighted sum as follows:

$$\mathbf{p}_i \propto \sum\nolimits_j^m r_{i,j} \cdot \mathbf{p}_{i,j}^{\text{exemplar}}, \tag{3}$$

where $\mathbf{p}_{i,j}^{\text{exemplar}}$ is the label probability vector at pixel i in exemplar j, and $r_{i,j}$ is the corresponding matching score.

3.4. Step 3: Pixel-wise Label Selection

The results from the previous stage are imperfect. Near smaller regions, like the eyes, eyebrows, and lips, we observe that, if the aggregated label probabilities are incorrect, they tend to be incorrect in the direction of the larger surrounding regions, namely the face skin and background. Consider, for example, a region near the eyebrow edge. Assuming noise prevents perfect correspondences, "skin" label correspondences will occur more frequently than "eyebrow" label correspondence simply because there are many more skin labels than eyebrow labels. A common symptom of this label bias is that estimated eyebrow regions (and other small regions of the face) tend to be too small.

We compensate for this bias by re-weighting each component of the aggregated label probability vector, and then renormalizing each pixel's label probability vector afterward. Given a tuning set with ground truth label probabilities, we find label component weights $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_K]$ by minimizing the following function Θ:

$$\begin{aligned} \Theta(\boldsymbol{\alpha}) \;&=\; \sum_k \theta_k(\boldsymbol{\alpha}) + \lambda \sum_{k'>k} \|\theta_k(\boldsymbol{\alpha}) - \theta_{k'}(\boldsymbol{\alpha})\|^2 \\ \text{s.t.} \quad &\sum\nolimits_k^K \alpha_k = 1, \quad \forall k:\ \alpha_k > 0, \end{aligned} \tag{4}$$

where

$$\theta_k(\boldsymbol{\alpha}) = \frac{1}{C_k} \sum_j^J \sum_{i \in \phi_{j,k}} \left\| p_{i,j,k}^{\text{gt}} - \frac{\alpha_k p_{i,j,k}}{\sum_k \alpha_k p_{i,j,k}} \right\|^2, \tag{5}$$

$p_{i,j,k}$ and $p_{i,j,k}^{\text{gt}}$ are estimated and the ground truth label probabilities for image j, pixel i, and class k, respectively; $\phi_{j,k}$ is the set of pixels in image j for which $p_{i,j,k}^{\text{gt}} > 0$; C_k is a normalization parameter; and λ is a scalar regularization parameter.

Minimizing Eq. (4) is complicated by the normalization in $\theta_k(\boldsymbol{\alpha})$, which makes it nonlinear, and the sum over images and pixels, which makes it large. However, Eq. (4) only needs to be minimized offline once. In our implementation, we minimize Eq. (4) using MATLAB's fmincon function, which uses the interior point method for large-scale constrained nonlinear optimization problems [2].

We can use Eq. (4) to find optimal weights to maximize different evaluation metrics. For example, setting C_k to the total number of pixels in all images with label k according to ground truth and $\lambda = 0.5$, we can maximize accuracy with respect to a confusion matrix; setting $C_k = 1$ and $\lambda = 0$, we can maximize accuracy with respect to F-measures. We will discuss the pros and cons of different settings in Section 4.2.

After the label component weights have been found, we adjust each label probability vector. Optionally, we can select the component with the largest probability value to generate hard segmentation results. Hard segmentations are used in our quantitative experiments for accuracy evaluation.

4. Results and Discussion

We have evaluated our method on two different datasets, and we show that it clearly improves upon a recent general scene parsing approach and existing face parsing approaches. Additionally, we adapt a recent landmark localization method and two face alignment algorithms to produce segmentation results, and show that our method is more accurate.

4.1. Experimental Datasets

Our first experimental dataset is LFW [6]. Luo *et al.* [15] showed segmentation results on 300 randomly selected images from LFW. To compare with their results, we use the same subset of images in our experiments. Following their procedure to evaluate accuracy, we generated ground truth by annotating each face with contour points around each segment.

Our second (primary) dataset is Helen [9], which is composed of 2330 face images with densely-sampled, manually-annotated contours around the eyes, eyebrows, nose, outer lips, inner lips, and jawline. We use Helen because it features high-quality, real-world photographs of people with a more balanced proportion of genders, ages, and ethnicities than other face datasets. We separated Helen into three parts for our experiments: exemplar, tuning, and test sets. Our exemplar set was used for all experiments, including experiments on LFW images. Our Helen tuning and test sets were formed by taking the first 330 images in the dataset; they include no subjects from the exemplar set. Our Helen test set is composed of 100 randomly selected images from the first 330, and our tuning set comes from the remaining images.

We generated ground truth eye, eyebrow, nose, inside mouth, upper lip, and lower lip segments automatically by us-

	left eye	right eye	nose	left brow	mouth	right brow	background
left eye	**.90**			.01			.09
right eye		**.93**				.01	.06
nose			**.88**		.01		.11
left brow	.03			**.91**			.06
mouth					**.90**		.10
right brow		.02				**.89**	.09
background			.01		.04		**.95**

(a) Results from [15]

	left eye	right eye	nose	left brow	mouth	right brow	background
left eye	**.990**			.003			.007
right eye		**.990**				.002	.008
nose			**.992**	.001		.001	.006
left brow	.002			**.988**			.010
mouth			.001		**.983**		.016
right brow		.003				**.982**	.015
background	.002	.002	.004	.006	.005	.006	**.975**

(b) Our results

Figure 2. Luo *et al*. [15] presented the accuracy of their segmentors using the confusion matrix shown in (a). We repeated their experiment using our method; our results are shown in (b) for comparison. Based on the confusion matrix, our results look much more accurate. However, this metric can be deceiving, as discussed in Section 4.2.

ing the manually-annotated contours as segment boundaries. For face skin, we used the jawline contour as the lower boundary; for the upper boundary, we separated the forehead from the hair by manually annotating forehead and hair scribbles and running an automatic matting algorithm [11] on each image. Although we do not focus on hair segmentation in this work, we also recovered "ground truth" hair regions using this approach. The hair mattes from [11] are usually accurate, but mistakes are inevitable. Therefore, to ensure fair accuracy measurements, we manually annotated the face skin in all test images.

4.2. Comparisons on LFW

We first show results on LFW, and compare with two recent face parsing techniques [15, 21], two recent face alignment algorithms [4, 18] and a face landmark localization algorithm [22], which we have adapted for face parsing; that is, we derived segmentation results from the estimated contour points using the same approach used to generate ground truth segments in Helen. Warrell and Prince [21] also showed results on LFW, but their 150-image test subset and source code are no longer available. We therefore simply report their numbers, and acknowledge that their results were computed on a different (but qualitatively similar) set of images.

Luo *et al*. [15] presented the accuracy of their segmentors using a confusion matrix. We repeated their experiment using our method. First, we computed results on our Helen tuning set, and used our continuous results to train the label weights according to Section 3.4 (we matched the LFW segment representation by grouping the Helen mouth components, and treating face skin as background). Figure 2 shows a comparison of our results using our trained label weights, and the results reported in [15].

Using the confusion matrix for comparison, our results look much more accurate than [15]. However, Figure 3 shows a result that exemplifies a problem with the label weights found by minimizing Eq. (4). Effectively, if we use $C_k = \sum_j |\phi_{j,k}|$ and $\lambda = 0.5$, Eq. (4) finds label weights that maximize the recall rates of eye, eyebrow, nose, and mouth pixels, which are relatively few and sensitive to errors, by sacrificing the recall rate of background pixels, which are numerous and insensitive to errors. The confusion matrix in Figure 2 (b) reflects maximized recall rates, but does not give any indication of the problem exemplified in Figure 3 (a). We therefore instead show accuracy in Table 1 using the F-measure, which is the harmonic mean of both recall *and* precision. We find that setting $C_k = 1$ and $\lambda = 0$ in Eq. (4) results in a set of learned label weights that give good performance with respect to the F-measure, shown below.

(a) Estimated (b) Ground truth

Figure 3. The result on the left exemplifies the problem with the label weights found by minimizing Eq. (4) using $C_k = \sum_j |\phi_{j,k}|$ and $\lambda = 0.5$: the eye, eyebrow, nose, and mouth regions are too dilated compared to the ground truth shown on the right despite maximizing the diagonal of the confusion matrix in Figure 2 (b). Based on this problem with the confusion matrix, we instead show accuracy using the F-measure, and set $C_k = 1$ and $\lambda = 0$ in Eq. (4).

F-Measures for LFW Images					
Method	Eyes	Brows	Nose	Mouth	Overall
Warrell & Prince [21]	0.443	0.273	0.733	0.653	n/a
Zhu & Ramanan [22]	0.520	n/a	n/a	0.635	n/a
Saragih *et al*. [18]	0.684	0.651	0.903	0.753	0.793
Gu & Kanade [4]	0.735	0.722	0.900	0.801	0.820
Ours	**0.765**	**0.752**	**0.914**	**0.881**	**0.863**

Table 1. The top row is copied from [21]. Zhu & Ramanan [22] is a landmark localization method, Saragih *et al*. [18], and Gu & Kanade [4] are face alignment methods. Segments were derived from the contours generated by these methods. We used the model provided with Zhu & Ramanan's implementation, which was trained on the Multi-PIE face database [3]. We used our implementations of Saragih *et al*. [18] and Gu & Kanade [4], both trained on Helen. The "overall" values are computed using all eye, eyebrow, nose, and mouth pixels. We see that our algorithm compares favorably to all previous works on LFW.

Label Weights			
Face skin	0.0765	Nose	0.1000
Left eye	0.0925	Inner mouth	0.2132
Right eye	0.0925	Upper lip	0.1114
Left brow	0.0615	Lower lip	0.1067
Right brow	0.0615	Background	0.0841

We compare the accuracy of our method with several other face parsing and alignment methods [4, 18, 21, 22] in Table 1. Our algorithm compares favorably to these works on LFW.

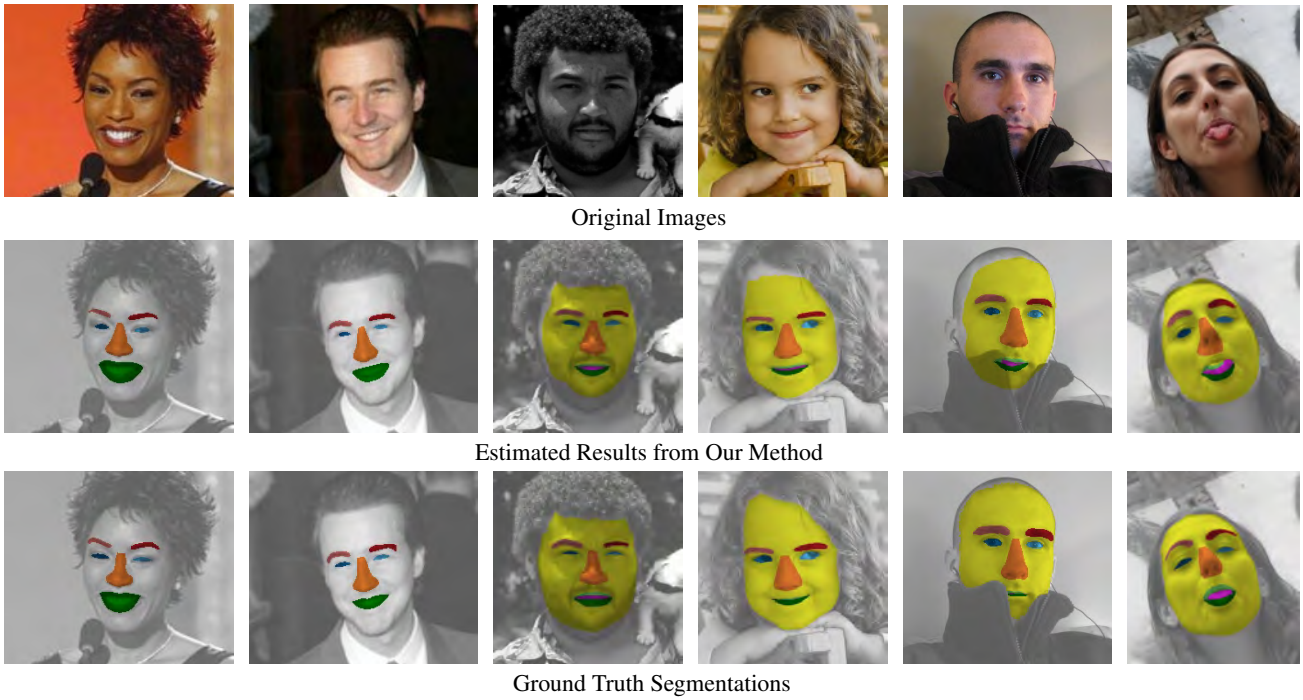

Figure 4. Selected results on LFW images (left two columns) and Helen images (right four columns). The inside of the mouth is not given as ground truth for the LFW images, and so we show only the entire mouth segment. We observe that our algorithm works well on grayscale images (third column) and images of young faces (fourth column). Our algorithm does not currently detect occlusions, and instead "hallucinates" occluded parts of the face (fifth column). The last column shows a failure case due to the unusual expression. We note that our algorithm generally produces excellent results. **Best viewed in color.**

F-Measures for Helen Images									
Method	Eyes	Brows	Nose	In Mouth	Upper Lip	Lower Lip	Mouth(all)	Face Skin	Overall
Zhu & Ramanan [22]	0.533	n/a	n/a	0.425	0.472	0.455	0.687	n/a	n/a
Saragih *et al*. [18]	0.679	0.598	0.890	0.600	0.579	0.579	0.769	n/a	0.733
Liu *et al*. [12]	0.770	0.640	0.843	0.601	0.650	0.618	0.742	**0.886**	0.738
Gu & Kanade [4]	0.743	0.681	0.889	0.545	0.568	0.599	0.789	n/a	0.746
Ours, Steps 1 & 3 omitted	0.766	0.687	0.896	0.678	0.637	0.703	0.853	0.861	0.779
Ours, Step 3 omitted	0.772	0.708	0.914	0.659	0.639	0.697	0.850	0.872	0.790
Ours, full pipeline	**0.785**	**0.722**	**0.922**	**0.713**	**0.651**	**0.700**	**0.857**	0.882	**0.804**

Table 2. Zhu & Ramanan [22], Liu *et al*. [12], Saragih *et al*. [18], and Gu & Kanade [4] were trained as described in Section 4.2. In this case, the "overall" measure is computed over eye, eyebrow, nose, inner mouth, upper lip, and lower lip segments; face skin is excluded in the overall measure, as it cannot be computed for Zhu & Ramanan, Saragih *et al*., or Gu & Kanade. The only area where Liu *et al*.'s system is more accurate than ours is on the face skin. The difference is minimal and is primarily due to our algorithm incorrectly "hallucinates" skin in hair regions, while Liu *et al*.'s system does not. In general, we see that our algorithm compares favorably to all previous works on this dataset, and our full pipeline performs best overall.

4.3. Comparisons on Helen

We repeated the LFW experiment on the Helen test set. Table 2 shows a comparison of the accuracy from three variants of our algorithm, Liu *et al*. [12], which is a general scene parsing method, and the adapted output from Zhu *et al*. [22], Saragih *et al*. [18], and Gu and Kanade [18], all ordered by overall F-measure.

A large disparity in accuracy can be seen between the results from Zhu *et al*. [22] and the other methods. We conjecture that this is due in part to the fact Zhu *et al*.'s provided model was trained on MultiPIE, whereas the other methods were trained on Helen, which includes a much richer set of landmarks and faces. Regardless, we see that our approach improves upon the segments generated by recent face alignment algorithms.

We used Liu *et al*.'s code trained on Helen to generate the values in Table 2. Liu *et al*.'s algorithm requires four parameters. We set $\alpha = 0.06$ (spatial prior weight) and $\beta = 1$ (smoothness weight) by performing a parameter sweep and selecting the values that maximized the overall F-measure. In [12] they suggest using $K = 85$ nearest neighbors and $M = 9$

voting candidates. However, for a fairer comparison, we set $K = 100$ and $M = 100$ to match the number of m top exemplars used by our method. To ensure that this change of K and M did not artificially reduce the performance of their method, we varified that $K = M = 100$ produced more accurate results than $K = 85$ and $M = 9$. We see in Table 2 that our algorithm is much more accurate than Liu *et al*. in general. Qualitatively, the segments generated by Liu *et al*. are also less accurate; please see our supplementary material for a visual comparison.

By comparing the fifth and sixth rows of Table 2, we observe that the local search and nonrigid exemplar alignment from Step 1 of our algorithm modestly improves the quantitative accuracy of our results. However, the improvement from Step 1 is mostly visible in our continuous, probabilistic results, which we cannot adequately judge quantitatively (*i.e*., to compute the F-measures, we must first quantize our results).

We see a noticeable improvement from row six to row seven in Table 2, especially in the inner mouth region, due to the label weights. In our view, the mouth is the most challenging region of the face to segment. The shape and appearance of lips vary widely between subjects, mouths deform significantly, and the overall appearance of the mouth region changes depending on whether the inside of the mouth is visible or not. Unusual mouth expressions, like the one shown in the rightmost column of Figure 4, are not represented well in the exemplar images, which results in poor label transfer from the top exemplars to the test image. Despite these challenges, our algorithm generally performs well on the mouth, with large segmentation errors occurring infrequently. Our improvement over other algorithms demonstrates the advantages of using segments to parse face parts. For example, the inside of the mouth is not well modeled using a classical contour based representation.

4.4. Runtime

Our experimental implementation was written in MATLAB, which makes it difficult to judge the true runtime of our approach. However, we can roughly estimate it as follows. Belhumeur *et al*. [1] report a runtime of less than one second per fiducial and they note that most of the time is spent evaluating the local detectors. For 12 landmarks, their algorithm should take approximately 12 seconds. Our MATLAB implementation of their algorithm represents 56% of the total computation in our pipeline. We therefore estimate that the true runtime of our algorithm would be approximately 21 seconds with a C++ implementation. Furthermore, much of the computation is devoted to interpolation and image warping, which can be made very fast on the GPU.

To be more concrete, we can compare the actual runtime of our current implementation with that of Liu *et al*. [12]. Liu *et al*. use global optimization, which makes their approach very slow. On a workstation with two quad-core 3.00 GHz Intel Xeon CPUs and 32 GB memory, their implementation took an average of 18.5 minutes per test image with $M = K = 100$, whereas our implementation took an average of 1.6 minutes per test image with $m = 100$ after the runtime pre-processing step (*i.e*., after Belhumeur *et al*.'s algorithm).

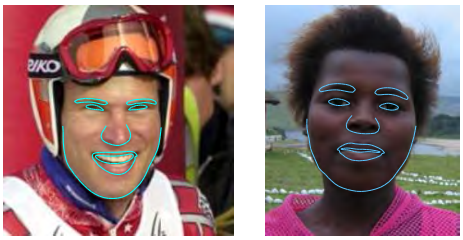

Figure 5. A simple extension of our approach is contour estimation. Please see Section 4.5 for details.

4.5. Extensions of Our Approach

Here we also show preliminary results for several extensions of our approach.

Contour Estimation Estimating contours is not the focus of this work. However, we can recover contours by treating contour points in the exemplars in almost the same way that we treat segment labels. That is, in Step 1, we warp the contour point from each exemplar in the same way that we warp the exemplar label maps. Then, in Step 2, we can aggregate the contour points using an approach similar to Belhumeur *et al*. [1]. Specifically, each contour point is found by computing the weighted average location of the warped exemplar contour points; each weight j is given by the match scores in R_j closest to the contour point. Figure 5 shows two selected results using this approach.

Hair Segmentation Several approaches for hair segmentation start by estimating a set of hair / not hair seed pixels in the image, and then refine the hair region using a matting algorithm ([16] is one example). We can also generate seeds by counting the votes from hair / not hair labels from the top exemplars, and thresholding the counts. Figure 6 shows seeds generating using this approach, and hair mattes computed from these seeds using [11].

Face Image Reconstruction and Synthesis Examplar-based face image reconstruction/synthesis is applicable for various face image editing tasks, including grayscale image colorization [10] and automatic face image retouching [5]. We can create a synthetic version of the input face by propagating color and intensity information from the exemplar images to the input image; this can be easily accomplished by replacing the label vectors with the color (or intensity) channels of the exemplar images. Figure 7 shows two examples.

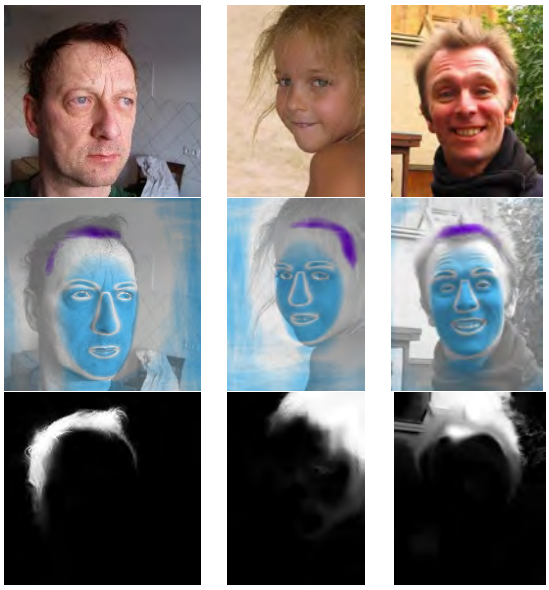

Figure 6. Preliminary results using an extension of our approach for hair segmentation. Top row: input. Middle row: our automatically generated "seeds" for hair (purple) and background (blue). Bottom row: automatic matting results from [11]. We can recover accurate hair mattes in many cases (first two columns), but the procedure often fails on difficult cases (third column). **Best viewed in color.**

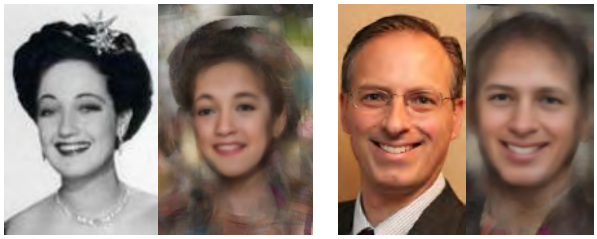

Figure 7. We can synthesize the input face by replacing the exemplar label vectors with the color channels from the exemplar images. See Section 4.5 for more details.

5. Conclusion and Future Work

In this paper, we have proposed an automatic face parsing technique that recovers a soft segment-based representation of the face, which naturally encodes the segment class uncertainty in the image. We argue that segments, unlike contours or landmarks, can represent any facial part, including hair, teeth, and cheeks. Second, we proposed a learning algorithm for finding optimal label calibration weights, which remove biases between label types. Third, we offer a new face segmentation dataset built as an extension of the recent Helen face dataset [9], which offers ground truth pixel-wise labels for face parts in high quality images. Finally, we showed in our experiments section that our approach parses faces more accurately than other approaches.

In the future, we plan to introduce additional label types, including ears, teeth, and neck. We also hope to explore one or more of the extensions in Section 4.5.

6. Acknowledgements

This work is supported in part by NSF IIS-0845916, NSF IIS-0916441, a Sloan Research Fellowship, a Packard Fellowship for Science and Engineering, Adobe Systems Incorporated, and an NSF Graduate Research Fellowship.

References

[1] P. N. Belhumeur, D. W. Jacobs, D. J. Kriegman, and N. Kumar. Localizing parts of faces using a consensus of exemplars. In *CVPR*, 2011.

[2] R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization*, 1999.

[3] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. In *Proc. Int. Conf. Autom. Face Gesture Recognit.*, 2010.

[4] L. Gu and T. Kanade. A generative shape regularization model for robust face alignment. In *ECCV*, 2008.

[5] D. Guo and T. Sim. Digital face makeup by example. In *CVPR*, 2009.

[6] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[7] P. Kontschieder, S. R. Bulò, H. Bischof, and M. Pelillo. Structured class-labels in random forests for semantic image labelling. In *ICCV*, 2011.

[8] D. Kuettel and V. Ferrari. Figure-ground segmentation by transferring window masks. In *CVPR*, 2012.

[9] V. Le, J. Brandt, Z. Lin, L. Bourdev, and T. S. Huang. Interactive facial feature localization. In *ECCV*, 2012.

[10] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. In *SIGGRAPH*, 2004.

[11] A. Levin, A. Rav-Acha, and D. Lischinski. Spectral matting. In *PAMI*, October 2008.

[12] C. Liu, J. Yuen, and A. Torralba. Nonparametric scene parsing via label transfer. In *PAMI*, December 2011.

[13] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. In *PAMI*, May 2011.

[14] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2003.

[15] P. Luo, X. Wang, and X. Tang. Hierarchical face parsing via deep learning. In *CVPR*, 2012.

[16] C. Rousset and P. Coulon. Frequential and color analysis for hair mask segmentation. In *ICIP*, October 2008.

[17] D. Ruprecht and H. Müller. Image warping with scattered data interpolation. *IEEE Comp. Graphics and Applications*, 1995.

[18] J. M. Saragih, S. Lucey, and J. F. Cohn. Face alignment through subspace constrained mean-shifts. In *CVPR*, 2009.

[19] J. Tighe and S. Lazebnik. Superparsing: Scalable nonparametric image parsing with superpixels. In *ECCV*, 2010.

[20] P. Viola and M. J. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[21] J. Warrell and S. J. Prince. Labelfaces: Parsing facial features by multiclass labeling with an epitome prior. In *ICIP*, 2009.

[22] X. Zhu and D. Ramanan. Face detection, pose estimation and landmark localization in the wild. In *CVPR*, 2012.