PRECISE HEAD SEGMENTATION ON ARBITRARY BACKGROUNDS

David C. Schneider, Benjamin Prestele, Peter Eisert

Fraunhofer Institute for Telecommunications, Heinrich-Hertz-Institut
Einsteinufer 37, 10587 Berlin, Germany
{david.schneider, benjamin.prestele, peter.eisert}@hhi.fraunhofer.de

ABSTRACT

We propose a method for segmentation of frontal human portraits from arbitrary unknown backgrounds. Semantic information is used to project the face into a normalized reference frame. A shape model learned from a set of manually segmented faces is used to compute a rough initial segmentation using a fast iterative algorithm. The rough initial cutout is refined with a boundary based algorithm called "Cluster Cutting". Cluster Cutting uses a cost function derived from clustering pixels along the normal of the initial segmentation path with a tree-building algorithm. The result can be refined by the user with an interactive variant of the same algorithm.

1. INTRODUCTION AND RELATED WORK

We propose a method for the segmentation of frontal human portraits including hair from arbitrary unknown backgrounds. Most segmentation methods require the user to provide some form of seed and allow for interactive and iterative refinement of the initial segmentation result. Our approach exploits its limitation to a narrow class of objects—frontal human portraits—to determine a seed automatically. An initial segmentation is computed and the result can be refined interactively if necessary. Even on images hard to segment, typically very few clicks are sufficient to obtain a good cutout. In the best case, no user interaction is required at all. The algorithm comprises two stages: In the first, a rough, shape-constrained segmentation is computed. In the second stage the result of the first is refined using a technique we call "Cluster Cutting". In the supervised refinement stage an interactive variant of the same algorithm is employed.

The algorithm was developed as part of a larger system for producing cut-out animations from user-provided photos. Therefore it is designed to robustly handle arbitrary image backgrounds and low-quality content, e.g. from mobile phones or webcams.

Literature on segmentation is too extensive to review comprehensively. Roughly, there are two primary types of interactive segmentation schemes, region and boundary based. The former are, for example, initialized with regions that label pixels as certain fore- and background [1] or with a single bounding box enclosing the full foreground [2]. A statistical model (e.g. of color) is derived from the samples. The segmentation is then computed by minimizing an error function that penalizes cutting regions homogeneous with respect to the model. For a class of error functions investigated in [3], the optimization can be computed using graph cuts. While little user interaction is required, region based methods do not provide precise control over the segmentation boundary. As color and gradient are not always sufficient for a plausible segmentation, several attempts were made to integrate shape priors [4, 5, 6, 7]. Our algorithm's first stage is region-based and uses a shape model. However, the limited variance of head shapes and the use of a special coordinate system allow us to employ a simple one-dimensional optimization rather than computationally expensive graph cuts.

Boundary based segmentation methods require the user to roughly trace the full object contour [8, 9, 10]. The path is then "snapped" to nearby edges. Recently "Boundary Snapping" [11] was proposed, where the user specifies only a few points which accurately lie on the object boundary. The segmentation is computed under the assumption that the vertical profile along the object boundary matches an interpolation between the vertical profiles at consecutive control points. The amount of user interaction is thus reduced in comparison to full boundary tracing but the object contour is required to change smoothly between control points. Our refinement algorithm employed in the second stage adopts the idea of using few control points and optimizing the segmentation path in between, as well as the scheme of working in a channel of normals around an initial segmentation boundary. However, a different approach for optimizing the segmentation in this channel is used. A generic segmentation approach combining a region-based initialization with a boundary based refinement is given by [12].

2. ROUGH SEGMENTATION

While the segmentation computed in the rough stage need not be precise, it is crucial that it does not deviate extremely from the depicted head. This is achieved by restricting the segmentation boundary to a shape space learned from 40 manually segmented images. The restriction of the shape has a stabilizing effect in image regions that are difficult to separate by color.

2.1. Polar reference frame

The entire rough segmentation is performed on a face-specific polar projection of the image. The center of the polar reference frame is the midpoint of the line segment connecting the eyes. Locating eyes is an extensive research topic to which we do not wish to contribute in this work; we use a standard Adaboost based detector (see [13], [14]). The zero-angle axis points in the direction of the right eye, hence normalizing face rotation. The eye distance is used to normalize scale. The image is resampled in the polar frame with an angular resolution A of 420 steps and a radial resolution R of 400 steps using bilinear interpolation. Fig. 1 (b) shows an example of an image projected into the reference frame.

Normalizing rotation and scale by working in the polar frame has several advantages: All subsequent processing in the rough segmentation stage is performed on images of the same size (i.e. $A \times R$ pixels); computation time is thus controllable despite of varying input sizes. Moreover, a common reference frame simplifies computation of probabilistic information from a set of training images and

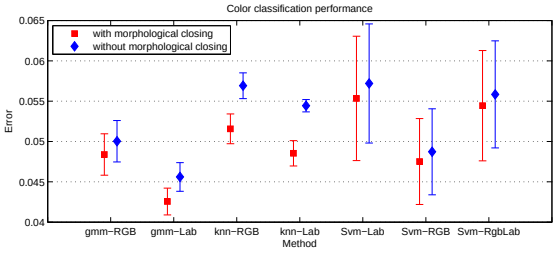

Fig. 2. Results of the color model type evaluation. Error measured as fraction of misclassified pixels.

relating this information to a new image. Finally, it facilitates the representation of a segmentation boundary as a fixed-length, one dimensional vector of radii, $[r_1 \dots r_A]^T$ which greatly simplifies the shape-constrained segmentation process described in section 2.3.

2.2. Color modelling

Rough segmentation is color-based. The color model is learned from the image. Let $\mathsf{L}_{\theta,r} \in \{'\mathrm{fg}', '\mathrm{bg}'\}$ be the label (or class) of a pixel at coordinates (θ, r) in a polar face image. Treating this as a random variable, a prior probability distribution $p(\mathsf{L}_{\theta,r})$ of $\mathsf{L}_{\theta,r}$ is computed from a set of manually segmented frontal face images projected into the polar reference frame. A statistical polar trimap of foreground, background and uncertain areas is determined by thresholding label probabilities.

When an image is segmented, two color models are learned from the certain background and certain foreground regions of the precomputed trimap. After evaluating several parametric and non-parametric model types empirically we chose Gaussian Mixture Models (GMMs) over L*a*b color space; the evaluation is described below. Denoting the color of a pixel at (θ, r) by $\mathbf{c}_{\theta,r}$ the GMMs give us the conditional probabilities $p(\mathbf{c}_{\theta,r}|\mathsf{L}_{\theta,r})$ for both possible values of $\mathsf{L}_{\theta,r}$. Hence the posterior probability of a pixel's label is given by applying Bayes theorem as

$$p(\mathsf{L}_{\theta,r}|\mathbf{c}_{\theta,r}) = \frac{p(\mathbf{c}_{\theta,r}|\mathsf{L}_{\theta,r}) \cdot p(\mathsf{L}_{\theta,r})}{p(\mathbf{c}_{\theta,r})} \tag{1}$$

with

$$p(\mathbf{c}_{\theta,r}) = p(\mathbf{c}_{\theta,r}|'\mathrm{fg}'_{\theta,r})p('\mathrm{fg}'_{\theta,r}) + p(\mathbf{c}_{\theta,r}|'\mathrm{bg}')p('\mathrm{bg}'_{\theta,r}) \tag{2}$$

To avoid clutter in notation we use $p_{\theta,r} := p(\mathsf{L}_{\theta,r}|\mathbf{c}_{\theta,r})$ for the posterior of eq. 1 in the following.

To select a color model we tested several model types, parametric and non-parametric, on a set of 40 highly diverse images that were segmented manually into a head region (with hair and throat) and a background. Each tested model type yields two numeric values for each pixel indicating the degree to which the pixel belongs to the foreground and to the background class; for some but not all models these are probabilities. A binary segmentation map was obtained by taking the larger value as the winner. The map was then compared to the ground truth. In total, seven model types were tested: Gaussian Mixture Models (GMMs) in RGB (1) as well as L*a*b color space (2), both normalized; k-nearest-neighbor models in normalized RGB (3) and L*a*b color space (4). Support vector machine (SVM) classification in normalized RGB (5) and L*a*b space (6) as well as in a combined, six-dimensional RGB+L*a*b space (7).

Fig. 2 shows the results. A second evaluation was computed after a morphological refinement of the class maps. In conclusion, the choice of model has no large effect on the color classification performance. GMMs in L*a*b color space perform best and have a reasonably small variance. They have the additional advantage of allowing a direct probabilistic treatment of color as described above.

2.3. Shape-constrained segmentation

Rough segmentation uses a linear shape model obtained from a set of manually segmented images. To build the model, the segmentation map of each image is projected into the polar reference frame. There, its boundary $(r_1, \theta_1),\ (r_2, \theta_2), \dots, (r_n, \theta_n)$ is determined. Assuming that the angular components θ_i are strictly and regularly increasing—i.e. $\theta_i = \frac{2\pi}{R} i$ where R is the angular resolution of the polar reference frame—the boundary can be represented as a single vector of radii $[r_1 \dots r_A]^T$ that has the same length for all images. This entails a simplification of the boundary shape which is, however, tolerable for head shapes. From the boundaries of the manual segmentations a linear subspace model is computed by Principal Component Analysis with mean vector $\boldsymbol{\mu}$ and principal component matrix $\mathbf{P}$. A subspace of six eigenvectors is used. To cover more variation, rotated and scaled copies of the training examples are included in the training set. Note that in the polar frame scaling is achieved by simply adding a constant to the r_i while rotation amounts to a circular shift of elements in the radius vector.

The shape-constrained segmentation process is iterative. The current segmentation boundary is always represented as a vector of radii $\mathbf{q} = [r_1 \dots r_A]^T$ as described above. For a pixel at location (θ, r) this implies the labeling

$$\mathsf{L}_{\theta,r|\mathbf{q}} = \begin{cases} '\mathrm{fg}' & \text{iff } r \leq \mathbf{q}_\theta \\ '\mathrm{bg}' & \text{otherwise} \end{cases}. \tag{3}$$

Now let $\mathbf{f}$ be a normalized histogram describing the distribution of foreground posterior probabilities in the foreground of the image given segmentation $\mathbf{q}$. Let similarly be $\mathbf{b}$ the normalized histogram of *foreground* probabilities in the background implied by $\mathbf{q}$. Segmentation is performed by maximizing the χ^2-distance between $\mathbf{f}$ and $\mathbf{b}$ defined as

$$\chi^2 = \sum_{i=1}^{K} \frac{(\mathbf{f}_i - \mathbf{b}_i)^2}{\mathbf{f}_i + \mathbf{b}_i} \tag{4}$$

while restraining the boundary to the shape model; K is the number of histogram bins. In each iteration the elements of $\mathbf{q}$—i.e. the radii of the segmentation boundary—are updated in order to increase χ^2. Therefore it is not necessary to recompute the full histograms. Rather a pixel (θ, q_θ) on the boundary is updated according to the rule

$$(\theta, q_\theta) \leftarrow \begin{cases} (\theta, q_\theta + 1) & \text{if } \mathbf{f}[\theta, q_\theta + 1] > \mathbf{b}[\theta, q_\theta + 1] \\ (\theta, q_\theta - 1) & \text{if } \mathbf{f}[\theta, q_\theta - 1] < \mathbf{b}[\theta, q_\theta - 1] \end{cases} \tag{5}$$

where $\mathbf{f}[\theta, r]$ and $\mathbf{b}[\theta, r]$ are the histogram values for the pixels in the brackets. Also, the histograms can be updated along with the local changes. Call this a maximization step. After several maximization steps the changed boundary $\mathbf{q}$ is projected into the shape space by

$$\mathbf{q} \leftarrow \boldsymbol{\mu} + \mathbf{P}\left(\mathbf{P}^+ (\mathbf{q} - \boldsymbol{\mu})\right) \tag{6}$$

denoting by $(\cdot)^+$ the pseudo-inverse of a matrix. Call this a projection step. After each projection step the histograms have to be updated. Maximization and projection steps are repeated until convergence, i.e. until the change of χ^2 falls below a threshold. The boundary used for initializing the optimization is ϕ.

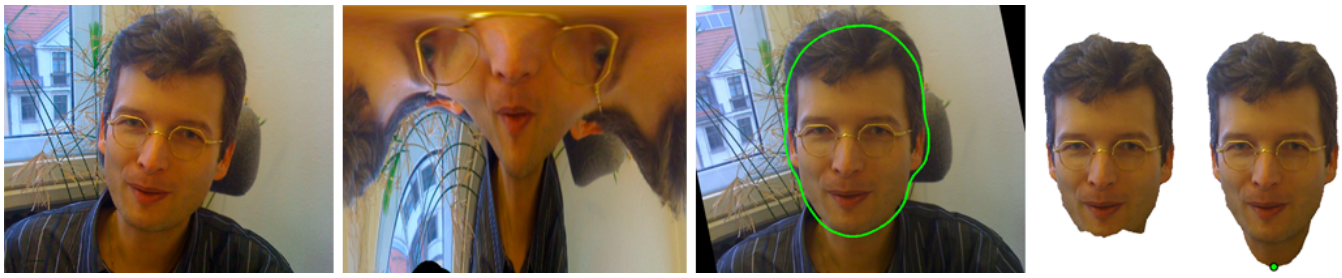

Fig. 1. (a) Input image. (b) Polar projection. (c) Rough segmentation boundary, rotation normalized. (d) Result after cluster cutting. (e) Result after one correction (clicked location indicated by green dot).

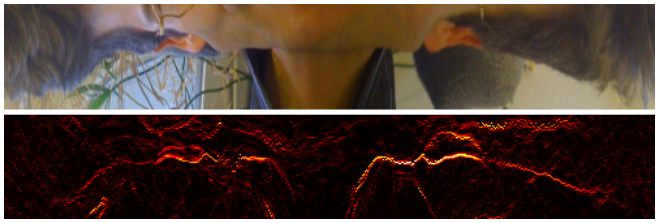

Fig. 3. Image channel (top) and corresponding cost function (bottom) used in Cluster Cutting.

The effect of the projection step is to constrain the segmentation path to the shape space learned from the manual segmentations. Thereby the initial segmentation always is a plausible head shape. Also, the projection step can bring parts of the path that pushed into a local similarity of foreground and background during the maximization step back "in line". An example of a rough segmentation is given in fig. 1 (c).

3. BOUNDARY REFINEMENT WITH "CLUSTER CUTTING"

To find a precise segmentation an automatic boundary-based method is applied using the result of the rough segmentation as input. Cluster Cutting is easily extended into an interactive segmentation tool that is finally used to correct remaining errors in the automatic segmentation.

As the algorithm requires an initial outline of the area to segment the rough segmentation boundary described in the previous section is projected back into the Euclidean frame and approximated by a natural cubic spline. The image is sampled at M points along each of N fixed-length, regularly distributed normals of that curve. This yields a rectangular projection of an M pixel wide channel around the rough segmentation boundary; see fig. 3.

The cost-function used by cluster-cutting is computed individually on the pixels along each normal. Therefore, the pixels along the normal are clustered using an iterative tree-building algorithm. The algorithm maintains a list $\mathsf{T}_1, \ldots, \mathsf{T}_M$ of trees represented by their roots. Initially the list contains M one-element trees each representing one pixel. Let $v(\mathsf{T}_i)$ be the color value of a tree. Initially, $v(\mathsf{T}_i)$ is set to the color vector of the pixel represented by the node in normalized L*a*b space. In each iteration, two trees T_i and T_j of the list satisfying two conditions are merged: T_i and T_j are adjacent (i.e. $|j-i|=1$) and the color similarity score

$$\alpha := ||v(\mathsf{T}_i) - v(\mathsf{T}_j)|| + \lambda ||p(v(\mathsf{T}_i)) - p(v(\mathsf{T}_i))|| \qquad (7)$$

is minimal over all pairs satisfying the first condition. Here $p(\cdot)$ is a color probability according to the model described in section 2.2 and the factor λ weights the influence of the color model against plain L*a*b color distance. Merged trees are replaced in the list by the common root T_i^* whose children they become. The color value of the new root is

$$v(\mathsf{T}_i^*) = \frac{v(\mathsf{T}_i)}{s(\mathsf{T}_i)} + \frac{v(\mathsf{T}_j)}{s(\mathsf{T}_j)} \qquad (8)$$

where $s(\cdot)$ denotes the number of leafs of a tree (i.e. the number of pixels it represents). The process is repeated until a single tree is left in the list.

Note that each of the $M-1$ non-leaf nodes in the tree corresponds with a two-partition of the pixels due to the adjacency condition. Hence each partition can be rated according to its α value defined in eq. 7. So when all trees have been built, an $M-1 \times N$ cost map of the α values in the segmentation channel can be set up; see fig. 3. A segmentation path is finally determined by computing a minimal cost through the cost map using dynamic programming (DP). To obtain a closed contour, the path must begin and end at the same vertical offset in the cost map which is not guaranteed by straighforward DP. Therefore, a common heuristic to close DP paths is used: DP is computed on several horizontally concatenated copies of the cost map. If closure fails, Dijkstra's shortest path algorithm is used as a fallback. Fig. 1 (d) shows a result of automatic cluster cutting.

Finally, for interactive refinement, the user clicks on a point he wants the segmentation path to pass through. The column of the cost map containing this point is determined and the cost map is set to $-\infty$ at all elements of the column except for the one clicked thereby forcing the path to pass through that pixel. The segmentation path is recomputed and displayed. Fig. 1 (e) shows a result of an interactive correction. Fig. 4 shows additional results, including those of the rough stage.

4. CONCLUSION & DISCUSSION

The proposed method was tested on a database of everyday frontal portraits with widely varying backgrounds. Generally, only a few user clicks were necessary to get a good segmentation of the face. In some cases no correction is required at all; examples are also shown in fig. 4.

The overall strategy of the algorithms can be summed up as follows: Limiting the segmentation problem to frontal human portraits allows us to use a special, semantically motivated coordinate frame. In this frame the problem is essentially one-dimensional in the sense that the segmentation boundary can be described as a function of polar angle. Therefore, a simple optimization can be used in the

Fig. 4. Exemplary results of the proposed segmentation method. Rough segmentation results are plotted in the original images. The two examples on the right required user interaction as indicated by the green dots.

first stage and the second stage can be constructed as a one dimensional path search given an appropriate cost function. Dividing the computation into a rough and a fine stage allows us to impose strict shape constraints in the first stage. Finally, the key idea behind the fine segmentation algorithm is to allow for precise user control while still enabling a fully automatic "first guess".

The approach is currently restricted to frontal views primarily due to the construction of the shape model and the use of eye detection. In principle a shape model could be set up for other views as well. Also, the result depends on eye detection which can, however, be replaced by an additional user interaction if necessary. As many other algorithms that rely on color models, our method requires the image to be principally color-separable. However, the shape model has a stabilizing effect if the color similarities between foreground and background are local.

Regarding future work, we want to further improve the shape model and the way it is fit to the image during rough segmentation. Also, segmentation is currently treated as a binary problem. Incorporating alpha matting to create a smooth transition between foreground and background should greatly improve the quality of the results; see, for example, the Bayesian method proposed in [15].

5. REFERENCES

[1] Y. Boykov and M.-P. Jolly, "Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images," in *Eighth IEEE International Conference on Computer Vision*, 2001.

[2] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," *ACM Transactions on Graphics*, vol. 23, pp. 309 – 314, 2004.

[3] Vladimir Kolmogorov and Ramin Zabih, "What energy functions can be minimized via graph cuts?," *IEEE Transactions On Pattern Analysis And Machine Intelligence*, vol. 26, pp. 147–159, 2004.

[4] Hang Chang, Qing Yang, and B. Parvin, "A bayesian approach for image segmentation with shape priors," in *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2008*, 2008.

[5] Nhat Vu and B.S. Manjunath, "Shape prior segmentation of multiple objects with graph cuts," in *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2008*, 2008.

[6] James Malcolm, Yogesh Rathi, and Allen Tannenbaum, "Graph cut segmentation with nonlinear shape priors," in *IEEE International Conference on Image Processing*, 2007.

[7] D. Freedman and Tao Zhang, "Interactive graph cut based segmentation with shape priors," in *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2005*, 2005.

[8] Patrick Pérez, Andrew Blake, and Michel Gangnet, "JetStream: Probabilistic contour extraction with particles," in *Proceedings of the Eighth International Conference On Computer Vision (ICCV-01)*, 2001.

[9] Eric N. Mortensen and William A. Barrett, "Intelligent scissors for image composition," in *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 1995.

[10] Michael Gleicher, "Image snapping," in *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 1995.

[11] E. Zadicario, S. Avidan, A. Shmueli, and D. Cohen-Or, "Boundary snapping for robust image cutouts," in *IEEE Conference on Computer Vision and Pattern Recognition CVPR 2008*, 2008.

[12] Yin Li, Jian Sun, Chi-Keung Tang, and Heung-Yeung Shum, "Lazy snapping," in *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers*, 2004.

[13] Yoav Freund and Robert E. Schapire, "A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting," *Journal of Computer and System Sciences*, vol. 55, pp. 119 – 139, 1997.

[14] Paul Viola and Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," in *IEEE Conference on Computer Vision and Pattern Recognition CVPR*, 2001.

[15] Yung-Yu Chuang, Brian Curless, David H. Salesin, and Richard Szeliski, "A Bayesian Approach to Digital Matting," in *Proceedings of CVPR*, 2001.