A Probabilistic Model for Object Recognition, Segmentation, and Non-Rigid Correspondence

Ian Simon
University of Washington

Steven M. Seitz
University of Washington

Abstract

We describe a method for fully automatic object recognition and segmentation using a set of reference images to specify the appearance of each object. Our method uses a generative model of image formation that takes into account occlusions, simple lighting changes, and object deformations. We take advantage of local features to identify, locate, and extract multiple objects in the presence of large viewpoint changes, nonrigid motions with large numbers of degrees of freedom, occlusions, and clutter. We simultaneously compute an object-level segmentation and a dense correspondence between the pixels of the appropriate reference images and the image to be segmented.

1. Introduction

We address the problem of explaining the pixels of a target image given a set of reference views specifying objects of interest. Each of these objects may appear in the target image among significant clutter, partially occluded, under significant deformation, or not at all. In explaining the target image, we segment the pixels by reference object and compute a dense correspondence between the target image pixels and the pixels in the matching reference views.

Our formal problem statement is as follows (see Figure 1):

> Given target image I to be explained and reference images R_j specifying the object appearances, determine which object (if any) each pixel in I belongs to, along with its corresponding location in one of the reference images.

Our approach places this problem into a generative model framework, where we assume that I is generated by transformations of the R_j and a background model B. The generative model also includes some other components, detailed in Section 3. See Figure 2 for a simplified overview. We solve for the segmentation and other model variables by attempting to find the maximum likelihood model state, which is equivalent to minimizing an energy function.

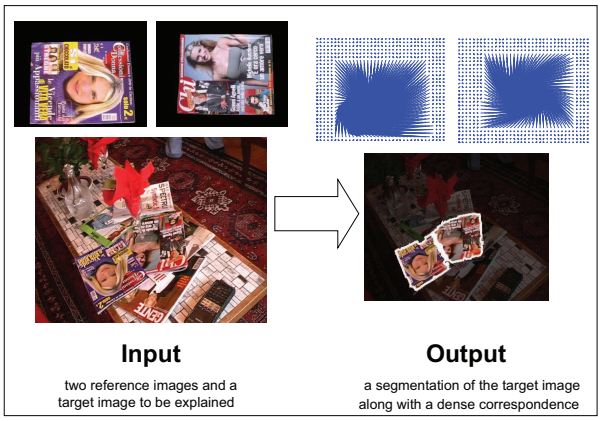

Figure 1. Our system uses a set of reference images to explain the pixels of a target image.

In the following section, we discuss related work. In Section 3, we present the generative image model. In Section 4, we describe our algorithm in detail. In Section 5, we show some results and discuss the significance of our work.

2. Related Work

Boykov and Jolly [3] address the problem of segmenting a grayscale image into foreground and background with limited user input. They use intensity histograms to model the foreground and background appearance distributions, and binary terms between neighboring pixels to enforce spatial smoothness. In their work, the object appearance is not specified in advance, but is dictated by user interaction. Blake *et al*. [2] extend this model to color images and produce an interactive matting application. Neither model takes the spatial properties of the object or background into account.

Kannan *et al*. [12] construct a generative model for image formation in which an image is composed of multiple transformed layers. Given only an input video, they decompose it into layers and represent each frame as an occluded set of layers, where each layer is transformed spatially and

1-4244-1180-7/07/$25.00 ©2007 IEEE

in appearance. Though their approach does not require that layer appearances are specified in advance, the transformation model they use is limited to translation plus a few pixels of local warping in order to make inference tractable. By using local features for initialization, our model is capable of handling a more extensive set of spatial transformations.

Kumar *et al*. [14] give an MRF-based generative model for object class recognition and segmentation using pictorial structures ([8]). He *et al*. [11], Levin and Weiss [15], and Winn and Shotton [22] all use conditional random fields to detect and segment object classes in images. Their approaches, like ours, attempt to merge high-level information (object appearance) with low-level information (smoothness). As the challenges of object class detection are different from those of object instance detection, their appearance models differ from ours. Also, they require multiple presegmented training images to learn the appearance models, and do not model the appearance of the object class from widely separate viewpoints.

Ferrari *et al*. [9] solve almost the same problem as we do, but use a different approach. They begin with a set of feature matches for each object and explicitly expand and contract that set, gradually exploring the image. Our approach is to construct a generative model of image formation and infer the values of the variables in our model. This involves minimizing an energy function that corresponds to the likelihood of our model, iteratively updating the segmentation and dense correspondence to object pixels. Many of the dependencies and priors in our model capture effects that Ferrari *et al*. achieve in a more procedural fashion.

3. Generative Image Model

We seek to explain the target image by a set of transformed reference images and a background distribution. Each pixel in the target image is generated by a transformation of a pixel in one of the reference images or by the background distribution. An index mask indicates which reference image (if any) generates each pixel in the target image, and there are several model variables describing the transformation. Instead of specifying the spatial transformation parametrically, we keep our approach general by using a smoothly varying non-parametric warp field.

Our model (Figure 3) uses the following components:

I - the image to be segmented

M - the index mask for image I

R_j - the j^{th} reference image

K_j - the color transformation from R_j to I

S_j - the spatial transformation from R_j to I

Φ_j - the pixel correspondence from I to R_j

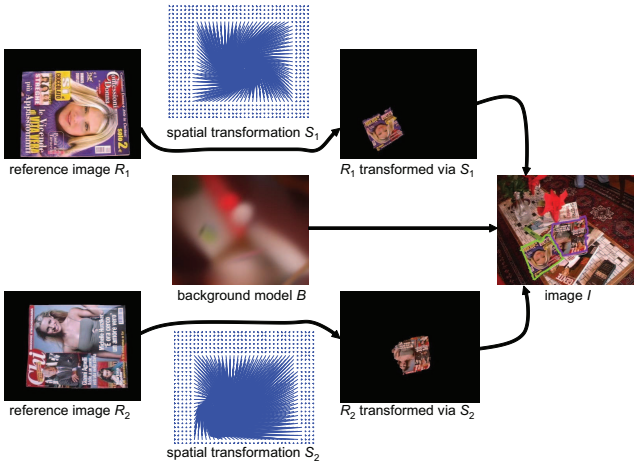

Figure 2. An example of the image generation process. The target image I is generated by transforming the two reference images R_1 and R_2 and placing them over a background model. Not all model components are shown, and the arrows in this figure do not indicate model dependencies. The background model B is represented as a mixture of Gaussians over RGBXY, depicted as an image here.

Ψ_j - the confidence image for R_j

Σ_j - the transformation covariance (a 5-by-5 matrix)

B - the background model for image I

We use the notation $I(p)$ to refer to the 5-dimensional RGBXY vector corresponding to pixel p in image I. To refer to only the color of $I(p)$, we use $I^{\text{RGB}}(p)$. To refer to only the location of $I(p)$, we use $I^{\text{XY}}(p)$. $M(p) = j$ means that pixel p comes from reference image R_j. $M(p) = 0$ means that pixel p comes from background model B.

I and the R_j are RGB images. M is an image of indices. K_j is a function that transforms $R_j^{\text{RGB}}(q)$. S_j is a function that transforms $R_j^{\text{XY}}(q)$. Φ_j is a function from $p \rightarrow q$, where p is a pixel in I and q is a pixel in R_j. Ψ_j is an image the size of R_j that indicates the probability that each pixel (after transformation) is present in I. Σ_j is a 5-by-5 RGBXY covariance matrix over the pixels of I with correspondences in R_j. B is a mixture of 5-dimensional Gaussians over the pixels of I belonging to the background.

The image I is conditionally distributed as follows:

$$P(I|M,R,K,S,\Phi,\Sigma,B) = \left(\prod_{p|M(p)=0} P(I(p)|B)\right) \times \left(\prod_{p|M(p)=j} P(I(p)|R_j,K_j,S_j,\Phi_j,\Sigma_j)\right)$$

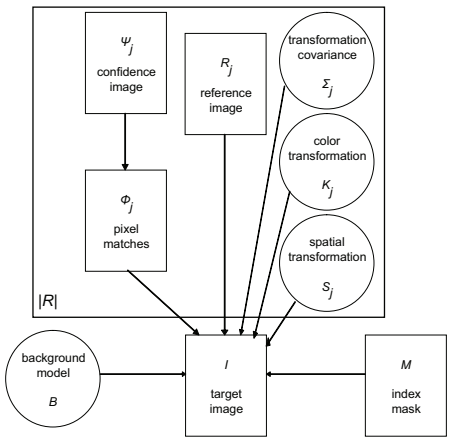

Figure 3. The model graph. I is directly dependent on M, B, and all R_j, S_j, K_j, Σ_j, and Φ_j. Φ_j is directly dependent on Ψ_j. The plate indicates that R_j, S_j, etc. are duplicated for each reference image.

$P(I(p)|B)$ is simply the density function for a mixture of Gaussians. $P(I(p)|R_j, K_j, S_j, \Phi_j, \Sigma_j)$ is a conditional Gaussian density function over RGBXY, with color mean defined by $K_j(R_j^{\mathrm{RGB}}(\Phi_j(p)))$, spatial mean defined by $S_j(R_j^{\mathrm{XY}}(\Phi_j(p)))$, and covariance matrix Σ_j.

The model therefore consists of two distributions, a background distribution which is a mixture of Gaussians, and a foreground distribution which is a single conditional Gaussian whose mean at pixel p depends only on $R_{M(p)}(\Phi_{M(p)}(p))$, the corresponding pixel in reference image $M(p)$. Pixels that are explained well using R_j are likely to belong to the j^{th} object, and pixels that are not explained well using any R_j are likely to belong to the background.

It still remains to define the transformations K_j and S_j. K_j is a linear transformation on R_j^{RGB}, such that each color channel is transformed to a linear combination of all three channels plus a constant. Thus, K_j is represented by a 3-by-3 matrix and a length 3 vector. S_j is a smooth mapping that transforms R_j^{XY} into the space of I. How this mapping and the other components are computed and updated will be discussed in the next section. We point out that S_j and Φ_j, the pixel correspondences, are not exact inverses. S_j represents the spatial transformation from R_j to I, on which smoothness is enforced, but color matching is not enforced. Φ_j represents the pixel correspondences from I to R_j, which are chosen to best satisfy both S_j and K_j, but on which smoothness is computationally difficult to enforce.

In addition, the model graph contains a dependence of Φ_j on Ψ_j. We use Ψ_j to represent the multinomial distribution from which $\Phi_j(p)$ is drawn:

$$P(\Phi_j(p) = q) = \Psi_j(q)$$

That is, the confidence image contains the probability that a pixel q in R_j is the corresponding pixel to some p in I, independent of the pixel colors and locations. The confidence image lets us represent which regions of the object in R_j are likely to be present in I.

When computing a segmentation and correspondence, we attempt to maximize the joint density of all model variables. This is equivalent to minimizing the negative log joint density, expressed as the following energy function:

$$\begin{aligned} E &= \sum_{p|M(p)=0} \big(\log P(M(p)=0) + \log P(I(p)|B)\big) \\ &+ \sum_j \sum_{p|M(p)=j} \big(\log P(M(p)=j) + \log P(\Phi_j(p)|\Psi_j) \\ &\qquad + \log P(I(p)|R_j, K_j, S_j, \Phi_j, \Sigma_j)\big) \\ &+ \sum_j \big(\log P(S_j) + \log P(\Psi_j) + \log P(\Sigma_j)\big) \end{aligned}$$

We use the terms in the last row to enforce priors on some of the model variables. Note that we have ignored $P(R_j)$, as the reference images are observed, and we have left out $P(K_j)$, as we do not specify a prior over color transformations.

Though the edge is not drawn in the model graph, Φ_j is also implicitly dependent on M, in that we are only accruing energy from $\log P(\Phi_j(p)|\Psi_j)$ for pixels p where $M(p) = j$. This is reasonable, as it makes little sense to penalize a tentative correspondence between target image pixel $I(p)$ and reference image pixel $R_j(q)$ when $I(p)$ is not classified as belonging to the object in reference image j.

We use the prior terms on all S_j and all Ψ_j to enforce smoothness on these variables. We don't enforce smoothness on the pixel correspondences Φ_j, but by forcing the spatial transformation S_j to be smooth, we encourage Φ_j to be smooth as well. Σ_j controls the scale of this effect. By increasing the color variance or decreasing the spatial variance, we encourage each pixel p in I to choose its match q in R_j such that $I^{\mathrm{XY}}(p)$ is closer to $S_j(R_j^{\mathrm{XY}}(q))$. The details of the smoothness functions we use are discussed in Section 4.2.

4. Algorithm

The algorithm consists of two main phases, initialization and iterative updates. In the iterative updates phase, each variable is updated in turn until convergence is reached. Each update causes the global energy to decrease. Since the algorithm is not guaranteed to find the global energy minimum, it is important that the initialization phase set the variables such that a good solution is reachable.

4.1. Initialization

The initialization is done in several steps:

1. First, we compute multiple types of local features in I and all R_j. We use two different affine invariant feature detectors: the Hessian-Affine detector [18] and the MSER detector [17]. We use SIFT [16] as our feature descriptor.

2. For each feature in I, we find the best match among the R_j using the Euclidean distance metric, using a ratio test with threshold 0.8 to remove potential outliers.

3. For each reference image R_j, if at least 4 matches were found in the image, use RANSAC [10] to robustly fit a homography to these matches. Throw out all outlier matches. Note that this may cause only a small region of the object to be initially detected. Initialize the spatial transformation S_j to be the resulting homography. Initialize the pixel correspondence Φ_j using the inverse of this transformation (rounding to integer pixels).

4. Initialize the segmentation using the inlier feature regions. All pixels belonging to an inlier feature matched with R_j are given the value j in M. All other pixels are assigned to the background.

5. Initialize the background model B by randomly assigning each background pixel to a Gaussian cluster, and run several iterations of the EM algorithm for mixtures of Gaussians. We use 20 clusters in our test, and run EM for 30 initialization iterations.

6. Initialize all K_j, all Ψ_j, and all Σ_j using the ordinary update method for these variables, discussed in the next subsection.

To achieve greater efficiency, we compute the features for the reference images beforehand and insert them into a nearest neighbor data structure ([1]).

4.2. Iterative Updates

After all of the variables are initialized, we update them in alternating fashion. Each variable is updated to minimize (or at least decrease) the global energy function with respect to that variable.

4.2.1 Updating the Segmentation

Without considering smoothness for the moment, we assign $M(p) = 0$ if

$$P(M(p) = 0)P(I(p)|B)$$

is greater than

$$P(M(p) = j)P(I(p)|R_j, K_j, S_j, \Phi_j, \Sigma_j)$$

for all j. Otherwise, we assign $M(p) = j$ in order to maximize:

$$P(M(p) = j)P(I(p)|R_j, K_j, S_j, \Phi_j, \Sigma_j)$$

This is equivalent to minimizing a unary error function where:

$$E(p) = \begin{cases} -\log P(M(p) = 0) + \\ -\log P(I(p)|B) & M(p) = 0 \\ \\ -\log P(M(p) = j) + \\ -\log P(I(p)|R_j, K_j, S_j, \Phi_j, \Sigma_j) & M(p) = j \end{cases}$$

To enforce the smoothness of M, we add a binary energy term $E(p, q)$ between each pixel p and all pixels q in its 4-neighborhood $N(p)$. We use the same ad-hoc function used by Blake *et al*. [2]:

$$E(p, q) = \begin{cases} 0 & M(p) = M(q) \\ \gamma e^{\frac{-\left(I^{\mathrm{RGB}}(p) - I^{\mathrm{RGB}}(q)\right)^2}{\beta}} & M(p) \neq M(q) \end{cases}$$

Note that this violates our model structure somewhat, as the smoothness terms are dependent on the pixel values in I. Most previous MRF approaches to segmentation, including Blake *et al*. [2], contain the same violation.

When an object has multiple reference views, we set $E(p, q) = 0$ if $M(p) \neq M(q)$ but $R_{M(p)}$ and $R_{M(q)}$ are two views of the same object. We use $\gamma = 50$ and we set β to $\frac{1}{2}$ of the mean squared color distance between neighboring pixels. This smoothness function encourages adjacent pixels with similar colors to be classified as belonging to the same object.

This type of binary energy function can be approximately minimized by a multiway graph cut algorithm as discussed in Boykov and Kolmogorov [4]. In our case, this is only one step in a larger energy minimization problem, so we only need to improve the current segmentation rather than find an optimal one. Therefore, we perform one α-expansion step per index j instead of iterating until convergence.

After updating the segmentation M, we also update the class probabilities $P(M)$ using pixel counts. We set $P(M(p) = j)$ to the number of pixels classified as belonging to object j divided by the total number of pixels.

4.2.2 Updating the Pixel Correspondences

Each set of pixel correspondences Φ_j (between I and R_j) is updated by matching each pixel p in I with its nearest neighbor in R_j (transformed by K_j and S_j) under the Mahalanobis distance defined by Σ^{-1}. Each pixel in R_j also has a constant penalty associated with it determined by Ψ_j. The match error we seek to minimize is:

$$\log P(\Phi_j(p)|\Psi_j) + \log P(I(p)|R_j, K_j, S_j, \Phi_j, \Sigma_j))$$

We set $\Phi_j(p) = q$ to minimize the above quantity. The best match for each pixel can be quickly computed with a nearest neighbor data structure ([1]).

4.2.3 Updating the Color Transformations

To update K_j, we solve a system of linear equations using the pixels p for which $M(p) = j$, along with Φ_p in R_j. Each pixel contributes 3 equations, and there are 12 unknowns, the 3-by-3 transformation matrix and the length 3 offset vector. The color transformation that minimizes the global energy can be recovered using least squares.

For our results, we restrict the 3-by-3 transformation matrix to be diagonal, effectively treating each color channel separately.

4.2.4 Updating the Spatial Transformations

We represent the spatial transformation S_j as an image of displacements the size of R_j. That is, we store $S_j(R_j^{\mathrm{XY}}(q))$ for all q in R_j. In a slight abuse of notation, let $S_j[q]$ be the displacement for pixel q. The warp we choose is constrained in two ways:

1. The warp should transform $R_j^{\mathrm{XY}}(\Phi(p))$ to $I^{\mathrm{XY}}(p)$, or close to it.

2. The warp should be smooth.

We set up a system of linear equations to encourage our warp to have these properties. To encourage the first property, for each pixel p where $M(p) = j$ we use the equations:

$$S_j[\Phi(p)] = I^{\mathrm{XY}}(p)$$

For the second property, we use equations that penalize local neighborhoods where the transformation is not affine. For each triple of horizontally or vertically consecutive pixels $q_1q_2q_3$, we use the equations (where ϵ weights smoothness relative to accuracy):

$$\epsilon\left(S_j[q_1] - 2S_j[q_2] + S_j[q_3]\right) = 0$$

For each square block of four pixels $\begin{matrix} q_1q_2 \\ q_3q_4 \end{matrix}$, we use the equations:

$$\epsilon\left(S_j[q_1] - S_j[q_2] + S_j[q_3] - S_j[q_4]\right) = 0$$

These equations encourage the spatial transformation to be locally affine. We solve the resulting large sparse system of equations using least squares.

4.2.5 Updating the Transformation Covariances

The transformation covariance Σ_j is updated easily, by computing the sample covariance of the $I(p)$ for p with $M(p) = j$ and mean specified by R_j, K_j, S_j, and Φ_j.

We also place a Wishart prior on Σ_j, to prevent it from growing too large. This is the conjugate prior for the covariance matrix of a Gaussian distribution. Without this prior, pixels in I may not be sufficiently penalized for being matched to pixels in R_j that are far away or not close in color.

4.2.6 Updating the Background Distribution

As the background distribution B is a mixture of Gaussians, we can perform a few steps of the EM algorithm for mixtures of Gaussians, using pixels p in I for which $M(p) = 0$.

4.2.7 Updating the Confidence Images

The confidence image Ψ_j is updated using Φ_j and M. For each pixel q in R_j, we count the number of pixels p in I for which $M(p) = j$ and $\Phi_j(p) = q$, and normalize by the total number of pixels p in I for which $M(p) = j$.

Unfortunately, this prevents any new regions of the object from being discovered, since if no pixel p in I corresponds to some pixel q in R_j, then no pixel in I can ever correspond to q. However, we would like to restrict matches to lie near regions that have already been discovered. In order to do this, we convolve the confidence image Ψ_j with a Gaussian after counting correspondences and before normalizing.

5. Results and Conclusions

We tested our method on the data set courtesy of Ferrari *et al*. [9]. The data set consists of 40 reference images covering 9 objects (Figure 4), and 22 test images containing different subsets of the objects.

After computing local features, we resize each image and reference view to one megapixel for efficiency purposes. We run our algorithm until the segmentation remains unchanged after an iteration. This can take anywhere from 5 to 30 iterations, depending on the initial feature coverage.

We show the segmentation extracted by our algorithm for several of these test images (Figure 5). The segmentation achieved is very good in most cases. Images (a1), (a2), and (a3) show simpler objects for which our segmentation is nearly perfect. Note the accurate occlusion boundary between the two magazines in (a3). A specular highlight was classified as background in (a1), as our color model doesn't handle such effects. Images (b1), (b3), and (c2) show more difficult occlusions that our algorithm still handles pretty well. Image (b2) shows our result on a scene with eight objects and multiple occlusions. Images (c1) and (c3) show

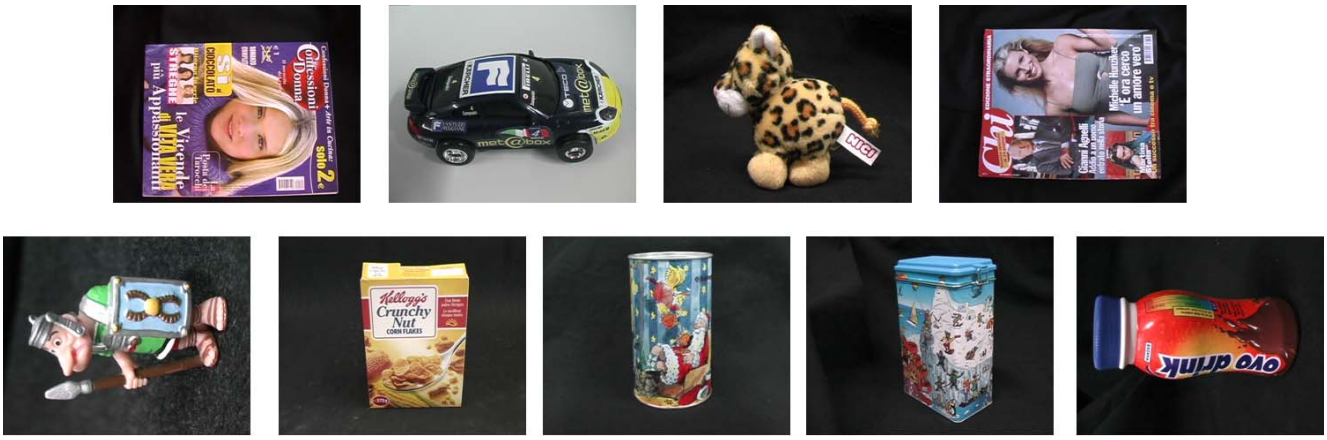

Figure 4. The 9 reference objects. Some objects used up to 7 additional reference images other than the one shown.

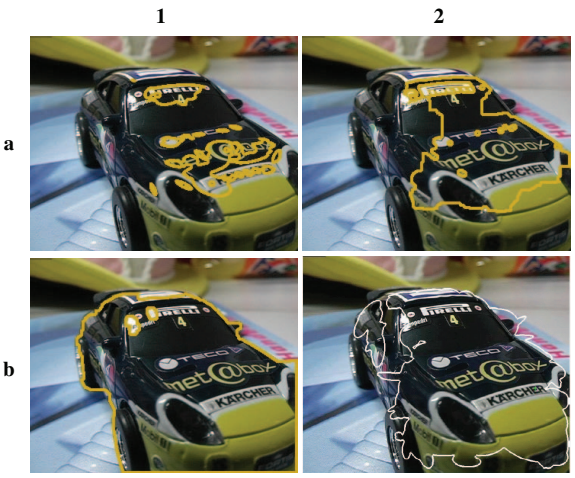

Figure 6. The segmentation computed by our algorithm after initialization (a1), several iterations (a2), and convergence (b1), and and the segmentation given by Ferrari *et al*. [9] (b2).

non-rigid transformations. In both cases, our algorithm is able to cope with the non-rigidity, though our global color model is unable to handle regions in shadow. One potential improvement to our model is to change the color representation to include local filter responses that are more robust to lighting changes, or to use a different color transformation model.

In Figure 6 we show the progress of our algorithm after initialization (a1), several iterations (a2), and convergence (b1). Image (b2) shows the segmentation given by Ferrari *et al*. [9]. Using graph cuts with a smoothness term at the pixel level helps us achieve smooth segmentation boundaries.

In this paper, we have presented a generative model for image formation based on a set of reference views, along with an algorithm that uses this model to compute a segmentation and dense correspondences in an unsupervised fashion. By explicitly representing and solving for multiple components of the image generation process, we are able to achieve competitive segmentation results on difficult images while placing the problem on a sound footing.

Acknowledgements

This work was supported in part by National Science Foundation grant IIS-0413198, the University of Washington Animation Research Labs, Adobe, Microsoft, and an endowment by Rob Short and Emer Dooley.

References

[1] S. Arya, D. Mount, N. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM*, 45(6):891–923, 1998.

[2] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive image segmentation using an adaptive gmmrf model. *Proc. ECCV*, pages 428–441, 2004.

[3] Y. Boykov and M. Jolly. Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images. *Proc. ICCV*, pages 105–112, 2001.

[4] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9):1124–1137, 2004.

[5] Y. Boykov, O. Veksler, and R. Zabih. Efficient approximate energy minimization via graph cuts. *PAMI*, 20(12):1222–1239, 2001.

[6] M. Brown and D. Lowe. Unsupervised 3D object recognition and reconstruction in unordered datasets. *Proc. 3DIM*, pages 56–63, 2005.

[7] G. Donato and S. Belongie. Approximate Thin Plate Spline Mappings. *Proc. ECCV*, pages 531–542, 2002.

[8] P. Felzenszwalb and D. Huttenlocher. Pictorial Structures for Object Recognition. *IJCV*, 61(1):55–79, 2005.

[9] V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous Object Recognition and Segmentation from Single or Multiple Model Views. *IJCV*, 67(2):159–188, 2006.

Figure 5. Some example segmentations using the objects from Figure 4. These images should be viewed in color. See text for discussion.

[10] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[11] X. He, R. Zemel, and D. Ray. Learning and Incorporating Top-Down Cues in Image Segmentation. *Proc. ECCV*, pages 338–351, 2006.

[12] A. Kannan, N. Jojic, and B. Frey. Generative model for layers of appearance and deformation. *Proc. AISTATS*, 2005.

[13] V. Kolmogorov and R. Zabih. What energy functions can be minimized via graph cuts? *PAMI*, 26(2):147–159, 2004.

[14] M. P. Kumar, P. H. S. Torr, and A. Zisserman. OBJ CUT. *Proc. CVPR*, pages 18–25, 2005.

[15] A. Levin and Y. Weiss. Learning to Combine Bottom-Up and Top-Down Segmentation. *Proc. ECCV*, pages 581–594, 2006.

[16] D. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 60(2):91–110, 2004.

[17] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. *Proc. BMVC*, pages 384–393, 2002.

[18] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[19] A. Opelt, A. Pinz, and A. Zisserman. A Boundary-Fragment-Model for Object Detection. *Proc. ECCV*, pages 575–588, 2006.

[20] T. Tuytelaars and L. Van Gool. Matching Widely Separated Views Based on Affine Invariant Regions. *IJCV*, 59(1):61–85, 2004.

[21] Y. Wexler, A. Fitzgibbon, and A. Zisserman. Bayesian estimation of layers from multiple images. *Proc. ECCV*, pages 487–501, 2002.

[22] J. Winn and J. Shotton. The Layout Consistent Random Field for Recognizing and Segmenting Partially Occluded Objects. *Proc. CVPR*, pages 37–44, 2006.

[23] S. Yu, R. Gross, and J. Shi. Concurrent object recognition and segmentation by graph partitioning. *Proc. NIPS*, pages 1383–1390, 2002.