Concurrent Object Recognition and Segmentation by Graph Partitioning

Stella X. Yu[†‡], **Ralph Gross**[†] **and Jianbo Shi**[†]
Robotics Institute[†]
Carnegie Mellon University
Center for the Neural Basis of Cognition[‡]
5000 Forbes Ave, Pittsburgh, PA 15213-3890
{stella.yu, rgross, jshi}@cs.cmu.edu

Abstract

Segmentation and recognition have long been treated as two separate processes. We propose a mechanism based on spectral graph partitioning that readily combine the two processes into one. A part-based recognition system detects object patches, supplies their partial segmentations and knowledge about the spatial configurations of the object. The goal of patch grouping is to find a set of patches that conform best to the object configuration. This process is integrated with the pixel grouping based on low-level feature similarity, through pixel-patch interactions and patch competition that is encoded as constraints in the solution space. The globally optimal partition is obtained by solving a constrained eigenvalue problem. We demonstrate that the resulting object segmentation eliminates local false positives at the high level of part detection, while overcoming occlusion and weak contours at the low level of edge detection.

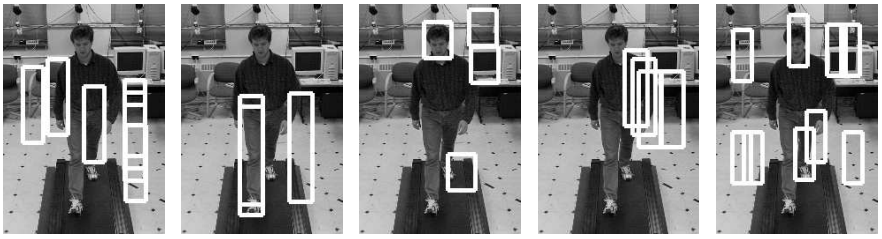
1 Introduction

A good image segmentation must single out meaningful structures such as objects from a cluttered scene. Most current segmentation techniques take a bottom-up approach [5], where image properties such as feature similarity (brightness, texture, motion etc), boundary smoothness and continuity are used to detect perceptually coherent units. Segmentation can also be performed in a top-down manner from object models, where object templates are projected onto an image and matching errors are used to determine the existence of the object [1]. Unfortunately, neither approach alone produces satisfactory results.

Without utilizing any knowledge about the scene, image segmentation gets lost in poor data conditions: weak edges, shadows, occlusions and noise. Missed object boundaries can then hardly be recovered in subsequent object recognition. Gestaltlists have long recognized this issue, circumventing it by adding a grouping factor called *familiarity* [6].

Without subject to perceptual constraints imposed by low level grouping, an object detection process can produce many false positives in a cluttered scene [3]. One approach is to build a better part detector, but this has its own limitations, such as increase in the complex-

ity of classifiers and the number of training examples required. On the other hand, another approach which we adopt in this paper, is based on the observation that the falsely detected parts are not perceptually salient (Fig. 1), thus they can be effectively pruned away by perceptual organization.



Right arm: 7 Right leg: 3 Head: 4 Left arm: 4 Left leg: 9

Figure 1: Human body part detection. A total of 27 parts are detected, each labeled by one of the five part detectors for arms, legs and head. False positives cannot be validated on two grounds. First, they do not form salient structures based on low-level cues, e.g. the patch on the floor that is labeled left leg has same features as its surroundings. Secondly, false positives are often incompatible with nearby parts, e.g. the patch on the treadmill that is labeled head has no other patches in the image to make up a whole human body. These two conditions, low-level image feature saliency and high-level part labeling consistency, are essential for the segmentation of objects from background. Both cues are encoded in our pixel and patch grouping respectively.

We propose a segmentation mechanism that is coupled with the object recognition process (Fig. 2). There are three tightly coupled processes. 1) Top-level: part-based object recognition process. It learns classifiers from training images to detect parts along with the segmentation patterns and their relative spatial configurations. A few approaches based on pattern classification have been developed for part detection [9, 3]. Recent work on object segmentation [1] uses image patches and their figure-ground labeling as building blocks for segmentation. However, this is not the focus of our paper. 2) Bottom-level: pixel-based segmentation process. This process finds perceptually coherent groups using pairwise local feature similarity. 3) Interactions: linking object recognition with segmentation by coupling patches and corresponding pixels. When a part is detected, it back projects foreground-background constraints onto the image data. When a pixel group is formed, it validates parts of the object model. With such a representation, we create a parallel object recognition and image segmentation process, the goal of which is to yield mutually consistent high-level part configuration and low-level self-coherent segmentation results.

We formulate our object segmentation task in a graph partitioning framework. We represent low-level grouping cues with a graph where each pixel is a node and edges between the nodes encode the affinity of pixels based on their feature similarity [4]. We represent high-level grouping cues with a graph where each detected patch is a node and edges between the nodes encode the labeling consistency based on prior knowledge of object part configurations. There are also edges connecting patch nodes with their supporting pixel nodes. We seek the optimal graph cut in this joint graph, which separates the desired patch and pixel nodes from the rest nodes. We build upon the computational framework of spectral graph partitioning [7], and achieve patch competition using the subspace constraint method proposed in [10]. We show that our formulation leads to a constrained eigenvalue problem, whose global-optimal solutions can be obtained efficiently.

2 Segmentation model

We illustrate our method through a synthetic example shown in Fig. 3a. Suppose we are interested in detecting a human-like configuration (Fig. 3b). Furthermore, we assume that

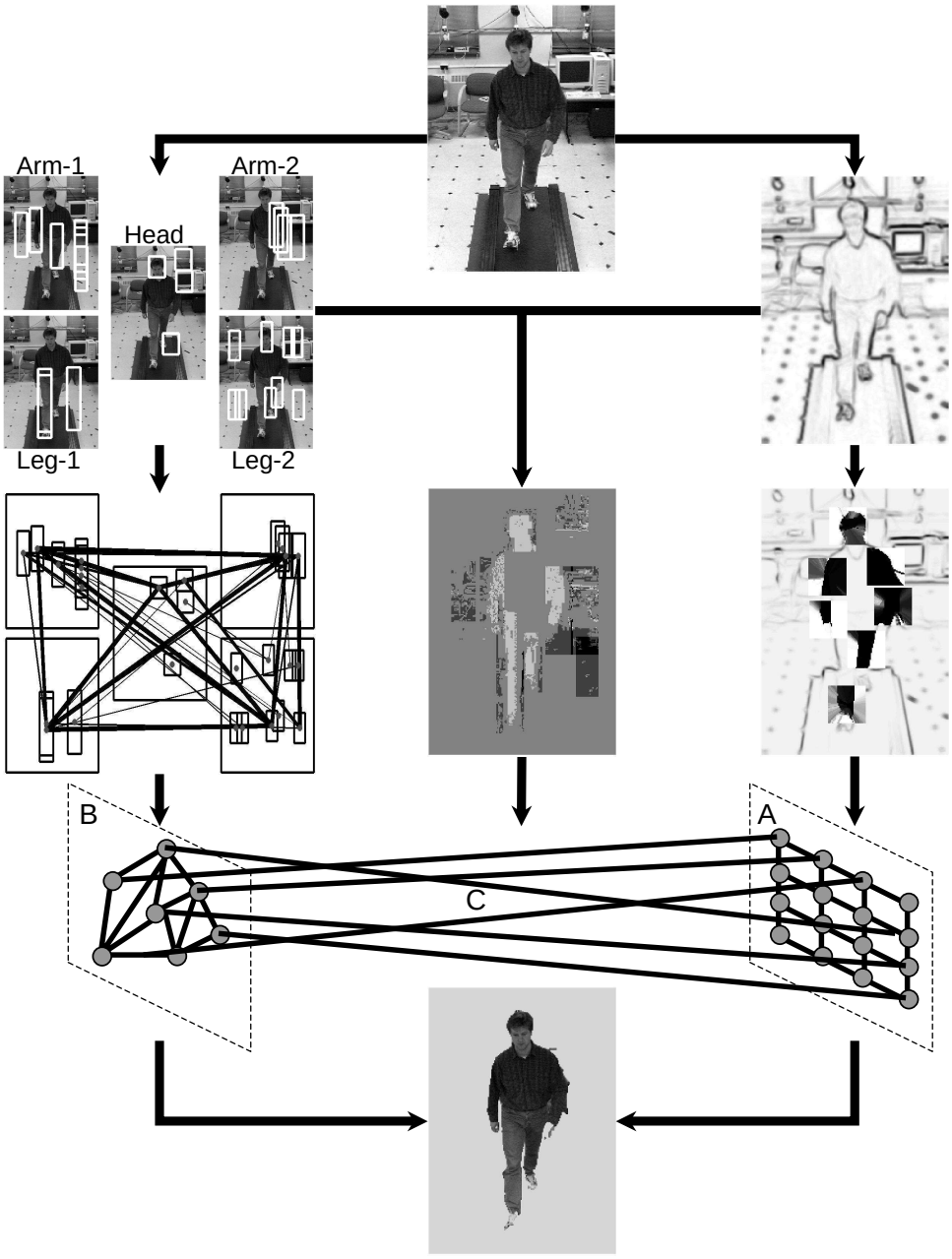
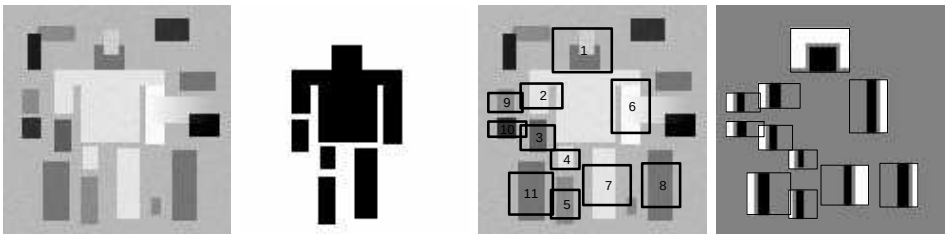


Figure 2: Model of object segmentation. Given an image, we detect edges using a set of oriented filter banks. The edge responses provide low-level grouping cues, and a graph can be constructed with one node for each pixel. Shown on the middle right is affinity patterns of five center pixels within a square neighbourhood, overlaid on the edge map. Dark means larger affinity. We detect a set of candidate body parts using learned classifiers. Body part labeling provides high-level grouping cues, and a consistency graph can be constructed with one node for each patch. Shown on the middle left are the connections between patches. Thicker lines mean better compatibility. Edges are noisy, while patches contain ambiguity in local segmentation and part labeling. Patches and pixels interact by mutual ownerships based on object knowledge and data coherence, as shown in the middle image. A global partitioning on the coupled graph outputs an object segmentation that has both pixel-level saliency and patch-level consistency.

some object recognition system has labeled a set of patches as object parts (Fig. 3c). Every patch has a local segmentation according to its part label (Fig. 3d). The recognition system has also learned the statistical distribution of the spatial configurations of object parts.



a) Image b) Object c) Patches d) Pixel-Patch

Figure 3: Object segmentation with part detection. Given 120×120 image in a), we want to detect the human-like configuration in b). 11 patches of various sizes are detected in c). They are labeled as head(1), left-upper-arm(2, 9), left-lower-arm(3, 10), left-leg (11), left-upper-leg(4), left-lower-leg(5), right-arm(6), right-leg(7, 8). Each patch has a *partial* local segmentation as shown in d). Dark pixels for figure, light for ground, and gray for no preference. The goal of segmentation is to find the best patch-pixel combinations that conform to the object knowledge and data coherence.

Given such information, we need to address two issues. One is the cue evaluation problem, i.e. how to evaluate low-level pixel cues, high-level patch cues and their correspondence. The other is the integration problem, i.e. how to fuse partial and imprecise object knowledge with most often unreliable low-level cues to segment out the object of interest.

2.1 Representations in graphs

We build coupled graphs for the object segmentation model (Fig.2). Formally, we denote a graph by $\mathsf{G} = (\mathsf{V}, \mathsf{E}, W)$. Let N be the number of pixels in the image, M the number of patches in the model. The complete node set is $\mathsf{V} = \{1, \cdots, N, N+1, \cdots, N+M\}$. The weight matrix for pairwise edge set E is:

$$W(A, B, C; \beta_B, \beta_C) = \begin{bmatrix} A_{N\times N} & \beta_C \cdot C^T_{N\times M} \\ \beta_C \cdot C_{M\times N} & \beta_B \cdot B_{M\times M} \end{bmatrix}, \quad (1)$$

where A is the pixel-pixel affinity matrix, B is the patch-patch affinity matrix, and C is the patch-pixel affinity matrix. β_B and β_C are scalars reflecting the relative importance of B and C with respect to A. All the weights are *nonnegative*.

Object segmentation corresponds to a node bipartitioning problem, where $\mathsf{V} = \mathsf{V}_1 \cup \mathsf{V}_2$ and $\mathsf{V}_1 \cap \mathsf{V}_2 = \varnothing$. We assume V_1 contains a set of pixel and patch nodes that correspond to the object, and V_2 is the rest of the background pixels and patches that correspond to false positives and alternative labelings. Let X_1 be an $(N+M) \times 1$ vector, with $X_1(k) = 1$ if node $k \in \mathsf{V}_1$ and 0 otherwise. It is convenient to introduce the indicator for V_2, where $X_2 = \mathsf{1} - X_1$ and $\mathsf{1}$ is the vector of ones.

We only need to process the image region enclosing all the detected patches. The rest pixels are associated with a virtual background patch, which we denote as patch $N+M$, in addition to $M-1$ detected object patches. Restriction of segmentation to this region of interest (ROI) helps binding irrelavent background elements into one group [10].

2.2 Initial evaluation of affinity matrices

Our initial estimation of the pixel affinity matrix A_S is based on edge detection. We first convolve the image with quadrature pairs of oriented filters to extract the magnitude of edge responses OE [4]. Let i denote the location of pixel i. Pixel affinity A_S is inversely

correlated with the maximum magnitude of edges crossing the line connecting two pixels:

$$A_S(i,j) = \exp\left(-\frac{1}{2\sigma_e^2}\cdot\left[\frac{\max_{t\in(0,1)} OE(\underline{i}+t\cdot\underline{j})}{\max_k OE(k)}\right]^2\right). \quad (2)$$

$A_S(i,j)$ is low if i, j are on the two sides of a strong edge.

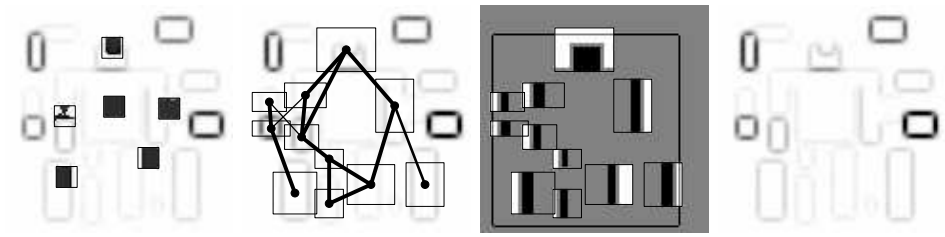
For object patches, we evaluate their position compatibility according to learned statistical distributions. For object part labels a and b, in principle, we can model their spatial distribution using a Gaussian, with mean μ_{ab} and variance Σ_{ab} estimated from training data. Let $\acute{p}$ be the object label of patch p. Let $\underline{p}$ be the center location of patch p. Given patches p and q, their affinity B_S:

$$B_S(p,q) = \exp\left(-\frac{1}{2}(\underline{p}-\underline{q}-\mu_{\acute{p}\acute{q}})^T\Sigma_{\acute{p}\acute{q}}^{-1}(\underline{p}-\underline{q}-\mu_{\acute{p}\acute{q}})\right). \quad (3)$$

$B_S(p,q)$ is low if p, q form rare configurations for their part labels $\acute{p}$ and $\acute{q}$. We manually set these values for our image examples.

Every object part label also projects an expectation of pixel segmentation within the patch window. We create pixel-patch association matrix C_S. Each column of C_S corresponds to N pixels, with $+1$ for the object pixels, -1 for non-object pixels, 0 for uncertain pixels.

As to the virtual background patch node, it only has affinity of 1 to itself in B_S, and 1 to a set of pixels outside of ROI in C_S. With these initial affinities, shown in Fig. 4, we couple their interactions to derive the final affinity matrices A, B and C.



a) A_S b) B_S c) C_S d) OE

Figure 4: Initial affinities and edges. A_S is evaluated for pixels within a distance of 5 units based on the edge responses in d). The affinity patterns of 6 pixels within their square neighbourhoods are superimposed on edges in a), with darker pixels for larger affinity. Patch graph B_S is drawn in b), with thicker lines for larger affinity. A summary of pixel-patch affinity C_S is shown in c), with object pixels marked black and non-object pixels white. The virtual background patch is associated with a set of pixels in the periphery.

2.3 Interactions between affinity matrices

Let α_A and α_B be integration weighting factors. Let f be a rectifying function: $f(x) = x$ for $x \geq 0$ and 0 otherwise. Let D_W be the degree matrix of affinity matrix W. It is a diagonal matrix with $D_W(i,i) = \sum_j W(i,j), \forall i$. With these notations, we define (Fig.5):

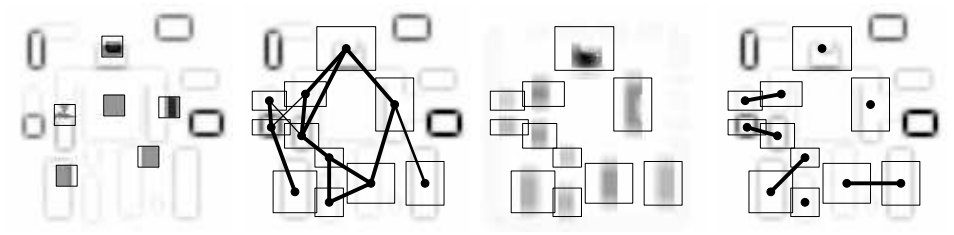
$$A = f(A_S + \alpha_A \cdot C_S B_S C_S^T), \quad (4)$$

$$B = f(B_S + \alpha_B \cdot C_S^T A_S C_S), \quad (5)$$

$$C = f(A_S C_S D_{B_S}). \quad (6)$$

Their explanations are as follows. For pixel affinity matrix A, we need to encode top-down feedback that can correct data by either breaking local coherence, e.g. enhancing weak contours (Fig. 3c patches 1 and 6), or promoting local coherence, e.g. ignoring occluding edges (Fig. 3c patch 1). This expected pixel affinity from object models is

captured by $C_S B_S C_S^T$. The final pixel affinity is thus a weighted average between the affinity computed from data and that expected from object models. Likewise, for patch affinity matrix B, we need to encode bottom-up feedforward evidence to corroborate or weaken the patch compatibility. The resulting patch affinity induced by the pixel affinity is captured by $C_S^T A_S C_S$. The final patch affinity is a weighted average between the affinity computed from object models and that expected from data. The pixel-patch affinity matrix C is obtained by averaging B_S through pixel and patch affinity, and then sorting out object pixels according to their signs. This operation takes both pixel affinity and patch affinity into account, so that the initial top-down projected segmentation is propagated and refined. The use of D_{B_S} instead of B_S is to keep each patch interacting with a small set of pixels, while having their strengths modulated by the overall patch affinity.



a) A b) B c) C d) S

Figure 5: Final affinities and exclusion constraints. a,b,c)Same conventions as in Fig. 4. d)Four linked pairs of patches compete to enter the object group.

The corrections made to initial pixel and patch affinities are restricted to local neighbourhoods of pixels and patches. With the pixel-patch affinity, patches of good configurations bring their pixels into one group. Such long-range binding is essential for popping out the object of interest among many equally good low-level segmentations.

The choice of these parameters dictates what we desire in the optimal object segmentation. For the integration parameters α_A and α_B, we simply set them to be unit-compatible. For the weighting factors we want β_B to balance the total weights between pixel and patch grouping so that the smaller number of patch nodes does not render patch grouping insignificant, while we want β_C to be large enough so that the results of patch grouping can bring their associated pixels along with them:

$$\alpha_A = \frac{\max A_S}{\max C_S B_S C_S^T},\ \alpha_B = \frac{\max B_S}{\max C_S^T A_S C_S};\quad \beta_B = 0.01\frac{1^T A 1}{1^T B 1},\ \beta_C = \frac{\beta_B}{\max C}. \quad (7)$$

2.4 Encoding exclusion for patch competition

The formulation presented so far does not prevent the desired pixel and patch group from including falsely detected patches and their pixels, nor does it favor the true object pixels to be away from unlabeled background pixels. We need further constraints to restrict a feasible grouping. This is done by constraining the partition indicator X.

In Fig. 5d, there are four pairs of patches with the same object part labels. To encode mutual exclusion between patches, we enforce one winner among patch nodes in competition. For example, only one of the patches 2 and 9 can be validated to the object group: $X_1(N+2) + X_1(N+9) = 1$. We also set an exclusion constraint between a reliable patch and the virtual background patch so that the desired object group stands out alone without these unlabeled background pixels, e.g $X_1(N+1) + X_1(N+M) = 1$. Formally, let S be a superset of nodes to be separated and let $|\cdot|$ denote the cardinality of a set. We have:

$$\sum_{k \in S_m} X_1(k) = 1,\ m = 1 : |S|. \quad (8)$$

2.5 Segmentation as an optimization problem

We apply the normalized cuts criterion [7] to the joint pixel-patch graph in Eq. (1) and formulate a constrained optimization problem:

$$\max \epsilon(X_1) = \sum_{t=1}^{2} \frac{X_t^T W X_t}{X_t^T D X_t}, \quad \text{s.t.} \sum_{k \in S_m} X_1(k) = 1,\ m = 1 : |S|. \tag{9}$$

Let $x = X_1 - \frac{X_1^T D X_1}{1^T D 1}$. By relaxing the constraints into the form of $L^T x = 0$ [10], we can show [10] that Eq. (9) becomes a constrained eigenvalue problem, the maximizer of which is given by the nontrivial leading eigenvector:

$$x^* = \arg\max \frac{x^T W x}{x^T D x}, \quad \text{s.t. } L^T x = 0. \tag{10}$$

$$Q D^{-1} W x^* = \lambda x^*, \tag{11}$$

$$Q = I - D^{-1} L (L^T D^{-1} L)^{-1} L^T. \tag{12}$$

Once we get the optimal eigenvector, we compare 10 thresholds uniformly distributed within its range and choose the discrete segmentation that yields the best criterion ϵ. Below is an overview of our algorithm.

1: Compute edge response OE and calculate pixel affinity A_S, Eq. (2).
2: Detect parts and calculate patch affinity B_S, Eq. (3).
3: Form initial pixel-patch affinity C_S.
4: Formulate constraints L among competing patches.
5: Compute A, B, C by coupling interactions between A_S, B_S and C_S, Eq. (4, 5, 6, 7).
6: Form W and calculate its degree matrix D, Eq. (1).
7: Solve $QD^{-1}Wx^* = \lambda x^*$, Eq. (12).
8: Threshold x^* to get a discrete segmentation.

We avoid computing Q directly by taking advantage of its low rank. It can be shown that the increase in computational complexity is negligible given $M \ll N$.

3 Results and conclusions

In Fig. 6, we show results on the synthetic image. Image segmentation alone gets lost in a cluttered scene. Even given pixel-patch interactions, a large object group can still not be formed without patch grouping cues. Given both patch grouping and patch-pixel interactions, a subset of false-positive patches form a group against other patches and background pixels. Only when exclusion constraints are used, the object of interest can be segmented out. With feedback from object models, unwanted edges (caused by occlusion) and weak edges (illusory contours) are corrected in the final segmentation.

We apply our method to human body detection in a single image. We manually label five body parts (both arms, both legs and the head) of a person walking on a treadmill in all 32 images of a complete gait cycle. Using the magnitude thresholded edge orientations in the hand-labeled boxes as features, we train linear Fisher classifiers [2] for each body part. In order to account for the appearance changes of the limbs through the gait cycle, we use two separate models for each arm and each leg, bringing the total number of models to 9. Each individual classifier is trained to discriminate between the body part and a random image patch. We iteratively re-train the classifiers using false positives until the optimal performance is reached over the training set. In addition, we train linear color-based classifiers for each body part to perform figure-ground discrimination at the pixel level. Alternatively a general model of human appearance based on filter responses as in [8] could be used.

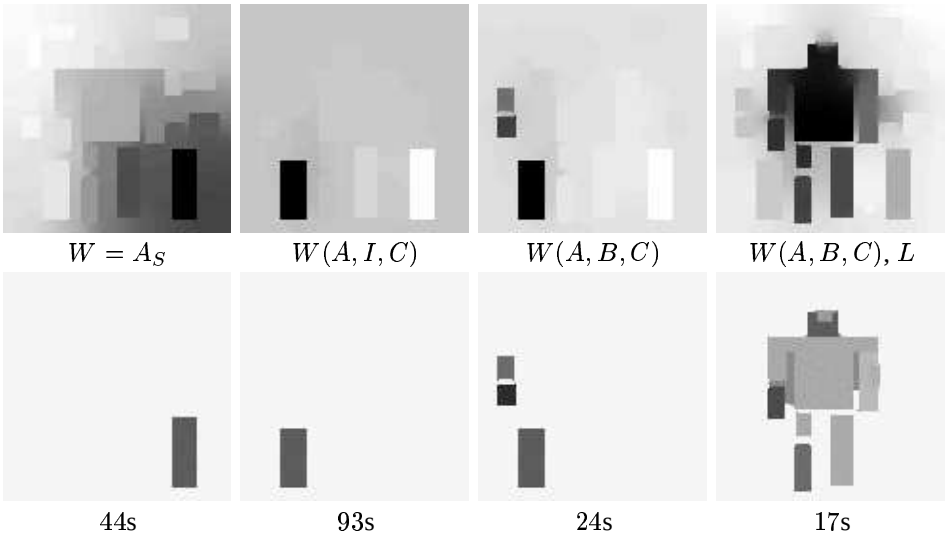


Figure 6: Eigenvectors and their segmentations for Fig. 3. From left to right are: 1) pixel grouping alone; 2)pixel grouping with pixel-patch interactions (patch affi nity set to the identity matrix); 3)pixel-patch grouping with their interactions; 4)pixel-patch grouping with interactions and exclusion constraints. Computation times are obtained in MATLAB on a PC with 1GHz CPU.

In Fig. 2, we show the results on a test image. Though the pixel-patch affinity matrix C, derived from the color classifier, is neither precise nor complete, and the edges are weak at many object boundaries, the two processes complement each other in our pixel-patch grouping system and output a reasonably good object segmentation.

Finally, we point out that it takes less time to compute the solutions in pixel-patch grouping since the size of the solution space is greatly reduced. This provides an explanation for the speed of biological vision systems given their slow neurochemical substrates.

Acknowledgments. We thank Shyjan Mahamud and anonymous referees for valuable comments. This research is supported by ONR N00014-00-1-0915 and NSF IRI-9817496.

References

[1] E. Borenstein and S. Ullman. Class-specifi c, top-down segmentation. In *European Conference on Computer Vision*, 2002.

[2] K. Fukunaga. *Introduction to statistical pattern recognition*. Academic Press, 1990.

[3] S. Mahamud, M. Hebert, and J. Lafferty. Combining simple discriminators for object discrimination. In *European Conference on Computer Vision*, 2002.

[4] J. Malik, S. Belongie, T. Leung, and J. Shi. Contour and texture analysis for image segmentation. *International Journal of Computer Vision*, 2001.

[5] D. Marr. *Vision*. CA: Freeman, 1982.

[6] S. E. Palmer. *Vision science: from photons to phenomenology*. MIT Press, 1999.

[7] J. Shi and J. Malik. Normalized cuts and image segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 731–7, June 1997.

[8] H. Sidenbladh and M. Black. Learning image statistics for Bayesian tracking. In *International Conference on Computer Vision*, 2001.

[9] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

[10] S. X. Yu and J. Shi. Grouping with bias. In *Neural Information Processing Systems*, 2001.