Segmentation According to Natural Examples: Learning Static Segmentation from Motion Segmentation (pre-print)
Michael G. Ross and Leslie Pack Kaelbling

Copyright ©2008 IEEE. Reprinted from IEEE Transactions on Pattern Analysis and Machine Intelligence.

This material is posted here with permission of the IEEE. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org.

By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

Segmentation According to Natural Examples: Learning Static Segmentation from Motion Segmentation

Michael G. Ross and Leslie Pack Kaelbling *Member, IEEE*

***Abstract*— The Segmentation According to Natural Examples (SANE) algorithm learns to segment objects in static images from video training data. SANE uses background subtraction to find the segmentation of moving objects in videos. This provides object segmentation information for each video frame. The collection of frames and segmentations forms a training set that SANE uses to learn the image and shape properties of the observed motion boundaries. When presented with new static images, the trained model infers segmentations similar to the observed motion segmentations.**

SANE is a general method for learning environment-specific segmentation models. Because it can automatically generate training data from video, it can adapt to a new environment and new objects with relative ease, an advantage over untrained segmentation methods or those that require human-labeled training data. By using the local shape information in the training data, it outperforms a trained local boundary detector. Its performance is competitive with a trained top-down segmentation algorithm that uses global shape. The shape information it learns from one class of objects can assist the segmentation of other classes.

***Index Terms*— segmentation, machine learning, motion, computer vision, Markov random field**

I. Introduction

A PICTURE is worth a thousand words, but a pixel is nearly worthless. Its brightness and color might be due to an object surface, lighting conditions, or instrument sensitivity. Its value may be corrupted by sensor noise. It provides no depth information. There are few useful vision tasks that can be performed with a single pixel. Therefore, optimally grouping pixels is a primary challenge in computer vision. This paper describes Segmentation According to Natural Examples (SANE), an algorithm that learns to segment objects in still images by training on videos of moving objects.

Many image segmentation algorithms have been developed in the last 40 years, but the endeavor has always been troubled by what David Marr called the "almost... philosophical problems" of specifying a useful region definition [20]. Image properties that might indicate a significant image boundary in one environment might be an unimportant local variation in another. If the segmentation is a pre-processing step, the regions useful for one algorithm might be useless to another. Learning approaches based on human-labeled boundaries have been explored ([3], [14], [22]), but acquiring the training data is expensive and the utility of these subjectively defined segmentations is unclear.

M.G. Ross is with the Department of Brain and Cognitive Sciences and L.P. Kaelbling is with the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Institute of Technology, 77 Massachusetts Ave, Cambridge, MA 02139. Emails: mgross@mit.edu, lpk@csail.mit.edu

SANE addresses these concerns. It learns a model of region boundaries by observing object motion in videos, and then applies this model to segment new static images. It produces segmentations with closed boundaries; uses an objective, useful, region definition; and can be trained to segment new environments with relatively low cost.

Motion provides a concrete, general-purpose region definition for segmentation. Defining a region as a collection of elements that undergo coherent motion corresponds well to intuition. All the visually distinct elements of a car can be considered a single region because they frequently move together, but a pile of leaves is composed of many elements that move independently on a windy day. This motion definition is clearly useful; for example, a robot manipulator must know whether a collection of pixels represents a single entity or something that will disintegrate at the slightest touch.

Apart from their utility, motion-defined regions are desirable because their test and training example sets can be automatically generated. A video camera on a sidewalk can record hundreds of moving cars, people, or animals every hour, and software can use their motion to separate them from their surroundings. Thus motion-defined regions combine utility and practicality.

SANE uses background subtraction to find the segmentation of moving objects in videos. This provides segmentation information for each video frame. The collection of frames and segmentations forms a training set that SANE uses to learn the image and shape properties that correspond to the observed motion boundaries. Then, when presented with new static images, the model infers segmentations similar to the observed motion segmentations.

SANE contributes a framework that uses video motion to generate training data for new segmentation environments, and learning and inference algorithms that translate the data's shape and image information into a useful still-image segmentation algorithm. Because it does not require human-labeled training data, SANE can adapt to a new environment and new objects with relative ease. The large, automatically labeled training sets also allow SANE to use simple learning algorithms that rely on counting and non-parametric probability density estimates. Comparisons to the well known normalized cuts segmentation algorithm [35] and to the Martin et al. [22] learned edge detector reveal performance advantages due to SANE's adaptability and shape modeling. SANE's output is competitive with the output of the top-down LOCUS segmentation algorithm [45], and it learns useful shape information that generalizes well to the segmentation of novel object classes and environments.

II. Related work

A. Image segmentation

Modern scientific interest in image segmentation dates from the work of Max Wertheimer and other Gestalt psychologists, who studied the visual properties that cause humans to perceive multiple image elements as unified groups. They identified grouping factors such as proximity, color, orientation, and common motion [26]. Many of the features used in computer image segmentation algorithms correspond to these classical principles.

The work of Spelke et al. on the development of image-segmentation abilities in children is an inspiration for SANE. They determined that the ability to distinguish object boundaries by motion and depth perception developmentally precedes the ability to segment based on cues such as color, brightness, and texture [36]. These results suggest that segmentation with these cues can be learned from the more primitive modes of segmentation.

The multitude of approaches to image segmentation can be usefully described along many dimensions. One of the fundamental differences is between methods that output edge maps and those that output region labels. Edge map methods, such as the Canny edge detector [5], typically do not constrain their output to only contain closed contours, so it may not be possible to determine the image regions from the edge map. Both the Shashua and Ullman [34] and the Ren et al. [28] algorithms attempt to improve edge map results by removing spurious edges and completing broken contours, but still do not guarantee closed boundaries. Segmentation algorithms, such as Shi and Malik's normalized cuts method [35], explicitly assign region labels to all image pixels, eliminating this ambiguity. The Martin et al. [22] and Konishi et al. [14] algorithms are two modern edge map methods, while Tu and Zhu's data-driven Markov chain Monte Carlo algorithm [39] outputs image regions, as does SANE.

Early segmentation algorithms required human initialization [13] or assumed human-tuned model parameters. Most modern approaches to the segmentation problem use machine learning techniques, but vary greatly in the type of training data they require. The Martin [22] and Konishi [14] algorithms both learn to detect edges from manually-labeled training data, as does the Levin and Weiss [19] method. Several region algorithms, such as Borenstein and Ullman [4] and Winn and Jojic's LOCUS [45] can learn from sets of images that each contain instances of a particular object class. Some manual cropping, selection, or other manipulation may be required, but not specific segmentation or object localization. An earlier version of SANE introduced the concept of learning to detect boundaries in static images by observing moving objects [30], [31]. The motion-learning approach makes the generation of training data cheap and automatic, and it does not rely on subjective human definitions of "object," "region," or "boundary." The Obj Cut algorithm [15] also learns by automatically extracting motion regions in videos.

There is also considerable variance in the models used to perform segmentation. Some segmentation algorithms, like normalized cuts, are primarily "bottom-up"—they infer segmentation from pixel values and local brightness and color gradients. Several of the recent learning methods, such as LOCUS and Obj Cut rely on very strong models of the objects' global shape and are best characterized as "top-down" methods. Levin and Weiss attempt to merge both approaches, while others, such as Borenstein and Ullman operate on mid-sized image chunks in an attempt to split the difference. SANE operates between the extremes of "bottom-up" and "top-down". It learns local boundary likelihoods using a simple set of features and combines that with a shape model based on local neighborhoods. It demonstrates that the combination of two independently inadequate sources of segmentation information can produce a successful segmentation algorithm.

A relatively small, but promising, category of algorithms attempts to integrate segmentation with related processes, such as object detection and classification. Leibe et al. [18] and Tu et al. [38] are two examples of the integrated approach.

Preliminary results from an earlier version of SANE were reported by Ross and Kaelbling [32].

B. Markov random fields

The segmentation models introduced in Section III are based on Markov random fields (MRFs), a class of probabilistic graphical models that encode the relationships between multiple variables. Markov random fields have been popular in computer vision since their use by Geman and Geman [10]. Recently, conditional random fields, a related family of graphical models proposed by Lafferty et al. [17] and applied to vision by Kumar and Hebert [16], have become prominent because they allow the use of more complex contextual features and avoid the overhead of generative modeling. The SANE MRFs also make extensive use of contextual image information.

A Markov random field is an undirected graph whose nodes represent variables of interest. If a graph contains variables X, and the neighbors of a particular variable X_i are denoted $N(X_i)$, the Markov property holds that $P(X_i|X \setminus X_i) = P(X_i|N(X_i))$; a variable is independent of the remainder of the graph given the values of its neighbors.

Unlike Bayesian networks, the joint distribution of the variables in an MRF cannot be found by multiplying local conditional probability functions. Instead, the Hammersley-Clifford theorem [1] states that every Markov random field is a Gibbs distribution, in which $P(X = x) = Z^{-1} \prod_C \Psi_c(X_c = x_c)$, where C is the set of cliques, or mutual neighbors, in the graph; Ψ_c are positive-valued clique potential functions; and $Z = \sum_X \prod_C \Psi_c(X_c = x_c)$ is a normalization constant called the "partition function." For a particular assignment to the variables in clique c, the larger the value of Ψ_c, the more likely the combination.

The necessity of normalizing the Markov random field energy function by Z makes inference in these models challenging. One popular approximate inference method is based on the belief propagation algorithm. Discovered by Pearl, belief propagation [27] is a pair of related algorithms that allow for efficient calculation of local marginal probabilities and global maximum *a posteriori* (MAP) estimates in tree-structured MRFs and Bayesian networks. Although their accuracy are only guaranteed for tree-structured MRFs, the algorithms can be applied to graphs containing loops. Commonly called "loopy belief propagation," this approach has found great favor among computer-vision researchers because it often produces useful approximate inference [43]. To improve the convergence of loopy belief propagation, researchers have investigated altering its "step size." Murphy et al. found that using a linear combination of the old and new messages improved convergence in some cases, but note that "in several cases the beliefs to which the algorithm converged were quite inaccurate" [25]. Heskes described "damping" the steps of loopy

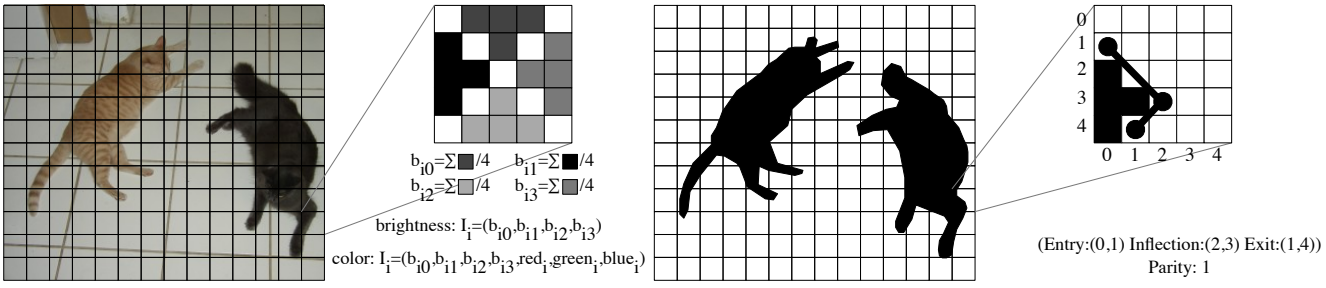

Fig. 1. An image and its segmentation are represented by patches representing local pixel, boundary, and segmentation values.

belief propagation by linearly combining the logarithms of new and old messages [11].

Besag [2] described iterative conditional modes (ICM), a simple hill-climbing method that produces adequate assignments to MRFs in many instances. ICM takes an initial assignment and scans through all the variables, making any local change that increases the probability of the overall assignment. Because each change only affects a few clique potentials, the algorithm is extremely efficient, although it is most useful when the initial assignment is nearly optimal.

Creating a Markov random field model for a set of variables requires choosing values for the clique potential functions. There is no simple translation between clique potential functions and the joint probability distributions of the clique's component variables because the normalizing partition function Z is necessary to all marginalizations and is influenced by all of the MRF's potentials. As a result, approximations are frequently used.

One parameter-fitting procedure is iterative proportional fitting (IPF), which selects parameters that match the marginal probabilities of the model to a set of observed marginal probabilities [12]. However, each IPF update requires inferring clique marginals, which is intractable due to the partition function. One solution is to use approximate inference to efficiently compute the necessary marginals. Wainwright and Sudderth have proven that in an MRF with single and pairwise clique potentials, $\Psi_i(X_i) = P(X_i)$ and $\Psi_{ij}(X_i, X_j) = \frac{P(X_i,X_j)}{P(X_i)P(X_j)}$ are fixed points of IPF using belief propagation inference [42]. Subsequently, Wainwright et al. demonstrated that these parameter settings are also approximate maximum-likelihood estimates [41] and noted that they are exact clique potential functions on a tree-structured MRF [40]. The two-element clique potential functions have an intuitive interpretation. If X_i and X_j are independent of each other, $\Psi_{ij}(X_i = x_i, X_j = x_j) = 1$ for all values x_i and x_j. Values for which $\Psi_{ij} > 1$ are positively correlated and values for which $\Psi_{ij} < 1$ are negatively correlated. These approximations will be the bases for the clique potential functions used in SANE. They have been used in MRF models prior to Wainwright and Sudderth's theoretical justification. For example, they are equivalent to the conditional probability potential functions used by Freeman et al. [8].

III. The SANE Algorithm

A. Segmentation model

SANE models images and their segmentations by dividing them into a grid of 5 pixel by 5 pixel, non-overlapping patches. The segmentation between an isolated object and its surroundings can be represented in each patch by the shape of the local boundary and a parity bit indicating which of the two segmentation labels ("0" and "1") is on each side of the boundary (Figure 1). The absence of a boundary in a patch is represented by a special "empty edge" value. The segmentation of patch (i, j) is denoted $S_{i,j} = (E_{i,j}, P_{i,j})$, with $E_{i,j}$ representing the local boundary (or "edge") and $P_{i,j}$ indicating the parity.[1] The corresponding image patch is $I_{i,j}$. The use of patches rather than pixels as the basic image unit is inspired by Freeman et al.'s [8] super-resolution MRF model.

The parity bits might seem unnecessary because the detection of closed boundaries are equivalent to the explicit labeling of segmented regions. But errors produced by the use of loopy belief propagation ([25], [44]) can result in the inference of broken boundaries even if the clique potential functions require closed boundaries. The parity bits, which enable labeling the pixels on each side of every edge as belonging to group "0" or group "1", allow SANE to always output a segmentation.

Parities are binary and edge variables are discrete and range over a parameterization of all possible boundary edges that can pass through a 5 by 5 patch. The patch size was chosen so the edge variables would represent significant local shape information, but have few enough possible values to allow for discrete representation. Because region boundaries are closed, their constituent edges cannot terminate in the middle of a patch. Therefore, local edges are represented as an entry point on the border of the patch, an optional inflection point inside the patch, and an exit point on the border of the patch. The permitted coordinates for these locations are the integer-valued pixel locations in the patch. Figure 1 shows an example segmentation and one of its constituent edge values. Enumerating the possible combinations, ignoring parameter settings that produce duplicate edges (such as swaps of the entry and exit coordinates), and adding the empty edge as a special case produces 2717 unique possible edge assignments.

Some local edge shapes cannot be exactly represented with this parameterization. For example, any local shape containing two or more inflections would be approximated by the nearest single-inflection edge. The small size of the patches compared to the object boundaries makes the representation an acceptable approximation that appears to cause no significant problems.

Each image patch $I_{i,j}$ is a real vector-valued variable representing useful local image features. In SANE, all models use the average brightness of four patch regions as image features, as shown in Figure 1. Color models also use the average red, green, and blue values of all the patch's pixels.

In a traditional MRF model, variables representing the underlying image data are separate from the hidden variables whose value is to be inferred. That arrangement is unsatisfactory for image segmentation because it prevents the model from representing certain useful relationships. For example, it might be desirable to

[1] Variables will be single-indexed when image location is unimportant.

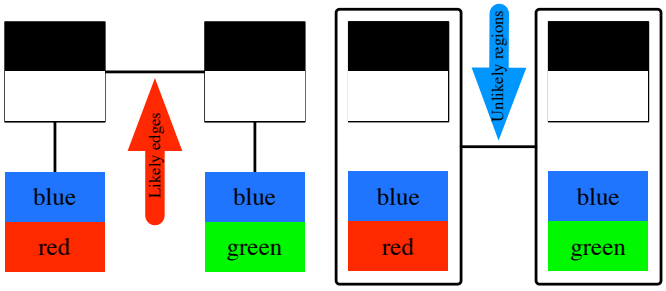

Fig. 2. Merging image and segmentation variables increases flexibility.

construct a segmentation model such that red and green pixels are never combined in the same segment and in which red-blue and green-blue transitions generally indicate boundaries. Consider the example in Figure 2, in which these two properties conflict. The compatibility between the neighboring assignments only depends on the fact that they are neighboring boundaries that continue a straight line. There is no clique potential function that prevents the combination of red and green pixels. To counter this problem, SANE combines the image and segmentation variables into the same nodes, so the MRF for a particular segmentation consists of nodes $X_{i,j} = (S_{i,j}, I_{i,j})$. This is similar to the conditional random field (CRF) models proposed by Lafferty et al. [17].

The nodes are connected to their neighbors in a first-order neighborhood. The neighbors of $X_{i,j}$ are $N(X_{i,j}) = \{X_{i+1,j}, X_{i-1,j}, X_{i,j-1}, X_{i,j+1}\}$. As discussed in Section II-B, two-element clique function values can be approximated by the ratio between the neighbors' joint marginal probability and the product of their individual marginal probabilities. SANE combines these potentials with a requirement that segmentation assignments correctly continue their neighbors.

To produce valid segmentations, neighboring patches must be assigned compatible values. Every possible local edge assignment requires the edge to enter from one neighboring patch and exit onto another and the assignments to these neighbors must have entrances and exits that continue the boundary without any break. The second requirement is that the parities of neighboring patches must produce consistent label assignments. Pixels labeled "0" cannot be next to pixels labeled "1" without an intervening edge and vice-versa. Both requirements are enforced by the continuity function $C(S_i, S_j)$, which returns 1 if the assignment to neighbors X_i and X_j match, and 0 if they are incompatible. Edge continuity is not enforced at patch corners because which neighboring patch should continue the edge assignment is ambiguous, but the label consistency requirements ensure a consistent assignment in those cases.

Adding the continuity requirement to the clique potential functions discussed in Section II-B, the parameters of a SANE MRF are estimated by

$$\Psi(X_i, X_j) = C(S_i, S_j)\frac{P(X_i, X_j)}{P(X_i)P(X_j)} + (1 - C(X_i, X_j))\epsilon,$$

where $\epsilon > 0$ is a constant indicating low compatibility ($\epsilon = 10^{-10}$ in the experiments described in Section IV). Zero-valued pairwise clique potentials are forbidden by the Hammersley-Clifford theorem [1] and would prevent the use of the ICM method to fix incompatible assignments that result from loopy belief propagation, which is discussed in Section III-D.

The X variables consist of discrete edge assignments and continuous image data. Therefore, it will be convenient to factor the probabilistic term in the clique potentials into

$$\begin{aligned}\frac{P(X_i, X_j)}{P(X_i)P(X_j)} &= \frac{P(I_i, I_j|S_i, S_j)}{P(I_i|S_i)P(I_j|S_j)}\frac{P(S_i, S_j)}{P(S_i)P(S_j)} \\ &= \frac{P(I_i, I_j|E_i, E_j)}{P(I_i|E_i)P(I_j|E_j)}\frac{P(E_i, E_j)}{P(E_i)P(E_j)}.\end{aligned}$$

The second equality results because the parities of the pixels are only used for continuity and interpreting the output as a segmentation. One label does not represent "object" and the other "background." The factorization is aesthetically pleasing because the edge factors represent the shape information learned from the data, while the image factors represent the probability of the image data given different shapes and the probability of inter-patch regional properties, such as the grouping of red and green pixels discussed earlier. It also divides the problem of estimating the necessary marginal distributions into discrete and continuous components.

Merging the image and segmentation variables into single nodes does not increase the computational complexity of inference. Since the image values are observed, there is no increase in the number of hidden variables or in their possible assignments.

Because the parities only affect the clique potentials between neighbors, the model described above always has at least two equivalent MAP assignments, both with the same edge assignments, but with opposite parity bits at every node. Belief propagation cannot resolve MAP ties [46], so SANE fixes the parity of the upper-left node, which breaks the tie.

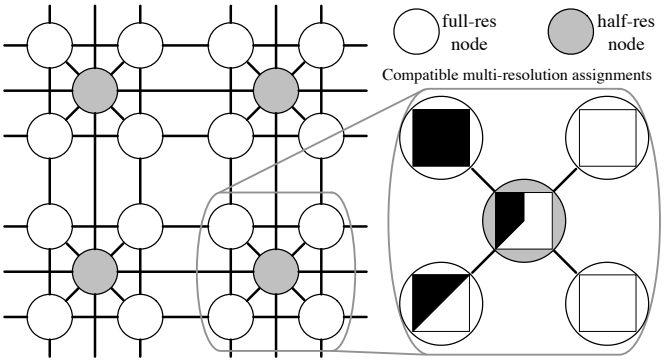

Fig. 3. Multiresolution SANE links full-resolution and half-resolution nodes.

B. Multiresolution segmentation

Many computer vision algorithms use multiresolution representations to combine the long-distance image relationships best captured at a low resolution with the precision possible at full resolution. Similarly, the multiresolution version of SANE consists of two linked MRFs, one defined on the original image and the other on the image at half resolution. Constructing them requires two models, one trained on full-scale images and the other trained on half-scale images.

As seen in Figure 3, the nodes of the two Markov random fields are intraconnected within each level as described previously and, additionally, each low resolution node is connected to the four full resolution nodes that are responsible for the equivalent area of the full-resolution image. The intralevel clique potential functions are the same as they would be for two independent MRFs, and the interlevel clique potentials enforce consistency between each low-resolution assignment and the associated full-resolution assignments for each of its four corners. For example, an assignment

Fig. 4. Images and background subtraction masks from each of the four data sets. From left to right: traffic, walking, mwalkA, and mwalkC.

to the full-resolution upper-left variable S_{ul} implies a 5x5 array of pixel labels L_{ul} and it is attached to a half-resolution node that assigns L_h to its pixels. Consistency demands that the sum of the squared differences between overlapping full and half-resolution labels, $M(L_{ul}, L_h) = \sum_{i,j}(L_{ul}(i,j) - L_h(i/2, j/2))^2$ is minimized. In order to avoid the complications of learning this relationship, and to allow flexibility for the inevitable minor variations between full and half-resolution assignments, multiresolution SANE sets the clique potential between the X_{ul} and X_h to 1 if $M(L_{ul}, L_h) <= 4$ and ϵ otherwise. The threshold of 4 was determined by trial and error. The clique potential functions for the other three multiresolution relationships, upper-right, lower-left, and lower-right, are defined analogously.

C. Training

Training the segmentation model requires a video of moving objects against a still background. The video is processed with the Stauffer and Grimson or Migdal and Grimson background-subtraction algorithms ([24], [37]), which produce a binary image for every video frame in which "1" pixels indicate a moving object and "0" pixels indicate background.

The background subtraction output only provides information about moving objects, so it produces a partially labeled dataset. For example, in a video of moving cars on a highway, the segmentation of disabled cars along the roadside will not be detected because static objects become part of the background. Training could be negatively affected by presenting the boundaries of such static objects as internal parts of a region. To alleviate this, a bounding box containing the foreground pixels and their immediate surroundings is computed and the remainder of the image is excluded from the training set. If multiple moving objects are present, a single bounding box containing all of them is computed.[2] Next, the regions inside the bounding box are extracted from the original video frame and the binary background subtraction image. The background subtraction image is converted into a binary edge image by scanning every row and column and labeling any transition between foreground and background as a boundary point.

The segmentation model requires estimates of discrete and continuous probability functions. The edge values are discrete, with 2717 possible values, while the image data are continuous, with a dimensionality and range dependent on the image features used. The edge-value distributions are computed by counting observed instances. The image-feature distributions are represented using Gaussian kernel density estimates. The ease of generating motion-labeled training data makes it possible to gather enough samples to use these simple learning algorithms. The training data is processed to make the model invariant to translation and rotation.

[2]In future applications to data sets that contain multiple moving objects intermixed with many static objects, per-object bounding boxes may be more appropriate.

The first task is estimating the edge probabilities $P(E_i)$ and $P(E_i, E_j)$ from the examples. The video processing steps described above result in a binary edge image. Dividing any edge image into 5x5 patches produces a set of edge and edge-pair samples. Shifting the patch boundaries right or down by 1–4 pixels produces an additional set of samples from the same image. Rotating each example patch and its immediate surroundings through 360 degrees (by 22.5 degree increments) further increases the number of sample edge and edge pairs harvested from each edge image and imparts rotational invariance to the resulting model.

To make the model translationally invariant, the individual edge samples are pooled into a single set without regard to their image locations. The edge pairs can be similarly pooled without regard to location, but they divide into two separate sets—vertical and horizontal neighbors. Assuming rotational invariance, a -90° rotation maps any horizontal pair into an equivalent vertical pair. Transforming all of the horizontal pairs in this manner represents all the pair samples as vertical relationships, concentrating them into fewer categories and making efficient use of their information.

After marshaling the training examples into their proper orientations, the problem of matching them to the appropriate edge parameterization remains. A 5x5 binary image patch can represent 2^{25} possible values, but the parameterization described in Section III-A only contains 2717 distinct edges. Therefore, each raw binary edge sample is represented by the most similar parameterized edge. The best match is found by comparing a rendered version of each parameterized edge to the binary image patch in question. The rendering of edge E is a 5x5 matrix $S(E)$ where location $S(E)(i,j) = \exp(-d(E,(i,j)))$ and $d(E,(i,j))$ is the distance between location (i,j) and the nearest location on edge E. For these calculations, the integer pixel and edge coordinates are considered to be in the center of pixels. For example, $(0,0)$ represents the center of the top-left pixel in the patch. Given a binary edge image patch, B, the error for parameterized edge E is $\sum_{i,j}(S(E)(i,j) - B(i,j))^2$. In order to discourage bending the local boundary to slightly reduce the error, an edge containing an inflection point is only considered best if its error is at least a 30% improvement over the lowest-error uninflected edge.

In some cases, a 5x5 binary image patch provides too little support to select the correct parameterized edge. Because edges must be continuous with their neighbors in inference, all edge-fittings occur in pairs, choosing the pair of continuous neighboring edges that best fit the binary edge image data. At each location the software finds the lowest-scoring continuous matches for that location and each of its four neighboring patches, provided they are not off the edge of the image. Each pair is stored as an example of neighboring edge assignments and each element as a single edge sample. Any overcounting that results should be almost evenly distributed and is not a cause for concern.

Given sets of sample individual edges, the next step is to estimate the probability distributions $P(E_i)$ and $P(E_i, E_j)$. These probabilities are estimated by counting observed edges and edge pairs, initializing each count to 1. This is equivalent to finding a MAP posterior estimate of the edge probabilities, assuming a Dirichlet prior that makes every edge and edge pair equiprobable [9]. This prevents zero probabilities, which are forbidden by the Hammersley-Clifford theorem and often cause inference

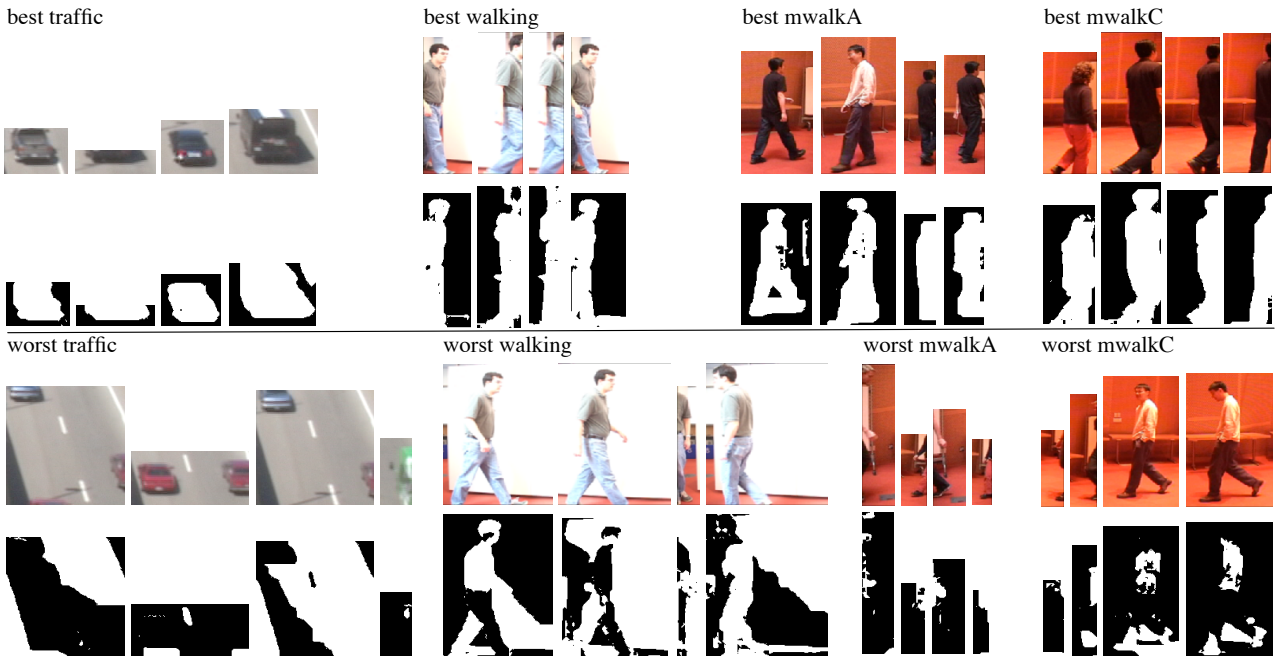

Fig. 5. The four best and worst color, multiresolution SANE results on the four data sets, computed using the step-size search (bp,ss) method.

problems. Edge values that are never observed in training are discarded from both the edge and edge-pair possible values.

The image probabilities, $P(I_i|E_i)$ and $P(I_i, I_j|E_i, E_j)$ are continuous and also estimated by collecting samples. The number of available samples is enhanced by observing multiple rotations of the underlying image data at each image location and by making the probabilities invariant to horizontal and vertical reflection. The single and pair distributions are related by $P(I_i|E_i) = \sum_{E_j} \int_{I_j} P(I_i, I_j|E_i, E_j)P(E_j|E_i)$, therefore SANE builds an estimate for the pair distribution and uses it to compute the single distribution. It would be more computationally efficient to learn a separate estimate of the single distribution, but experience has demonstrated that the quality of the output is degraded by any inconsistencies between the two distributions.

Although collecting enough image patches to provide a conditional image distribution for every edge pair is difficult, in some cases there is too much data rather than too little. For example, the empty-edge assignment appears far more often than any other value because it is used to label all non-boundary image regions. SANE solves this problem by adaptively lowering the patch sampling rate for these edge assignments. The image-pair probabilities are represented by Gaussian kernel density estimates [33]. For each edge pair, the variance of the kernels of its density estimate is selected by randomly dividing the collected samples in half and searching for a variance that maximizes the probability of one half when the remainder is used as the kernel means. The final estimator uses this kernel variance and all the samples as kernel means.

D. Inference

Once all the appropriate probability functions have been estimated from training data, the segmentation of new images can be performed by constructing an MRF and using the belief propagation inference algorithm.

Given an input image, the algorithm divides it into a grid of non-overlapping 5x5 patches. Each patch is assigned to a node $X_i = (S_i, I_i)$ as described previously. The computational costs of the belief propagation inference algorithm are $O(|e|^2)$ where $|e|$ is the maximum number of possible edge assignments at each node. Therefore, using the full set of 2717 possible edge assignments is impractical and each node is instead given a set of the locally most likely edge assignments. For each node X_i, the algorithm selects the empty edge and 19 other edge assignments that maximize $P(E_i = e_i|I_i)$. After these initial sets of possible assignments are selected at each node, the assignments at every neighbor and every pair of neighbors are examined to discover neighboring edge assignments that cannot be continued correctly by any currently available assignment at X_i. For each uncontinuable neighbor or paired-neighbor assignment, the most locally likely edge that continues it is added to the set of possible edges. Unlike the C function, continuity of edges that enter and exit through patch corners is considered because it is necessary to provide completing possible assignments at both of the bordering neighbors. This enhancement process is repeated until it converges and no more edges are added at any node. The local edge sets are also extended with edges corresponding to the patches' sides and corners to help close segmentation boundaries. Finally, the edges (except at X_0, see Section III-A) are paired with each of the two parity values, doubling the number of possible assignments at each node.

The approximate MAP estimates of the S variables are calculated via the max-product algorithm. There is no guarantee of convergence for loopy MRFs, so SANE stops belief propagation after 200 iterations and takes the approximate MAP estimate available at that point. Results can sometimes be improved by using Besag's iterative conditional modes (ICM) algorithm [2] to improve parity assignments as a post-processing step. Another approach is to try ten Heskes [11] belief propagation step sizes ranging from 0.1 to 1 and choose the result with the maximum MRF configuration value. More details of the development and implementation of the SANE algorithm can be found in Ross's doctoral dissertation [29].

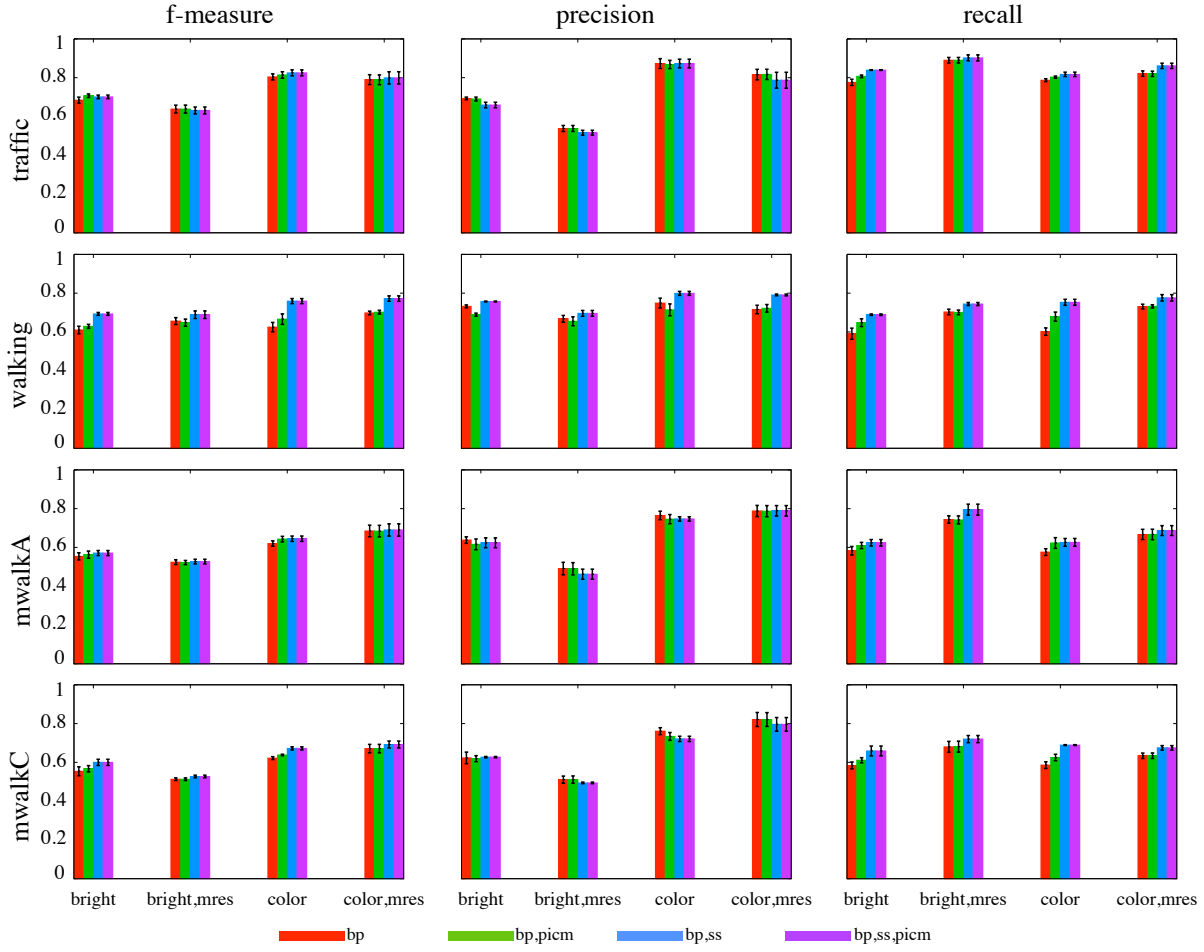

Fig. 6. The f-measures, precisions, and recalls of SANE models and inference methods on the data sets.

IV. Experiments

A. Data sets

The experiments in this section illuminate the performance of different SANE models and algorithms on four data sets. They demonstrate its advantages over the standard normalized cuts segmentation algorithm [35] and the Martin trained boundary detection algorithm [22]. The experiments also find that SANE's output on two of the data sets is competitive with the results of the LOCUS [45] top-down segmentation approach. Finally, additional experiments measure SANE's ability to generalize to other environments and lighting conditions, and show that the shape information it learns is surprisingly general and can be applied to object classes significantly different than those in the training set.

SANE was tested on four data sets (Figure 4). The *traffic* data consists of a single video recording of cars traveling down two roads. The original video was spatially divided to produce two videos, each covering traffic on a different road. The *walking* video shows one of the authors walking back and forth in front of a whiteboard. Two other videos of many people individually walking across a room form the *mwalkA* and *mwalkC* data sets. These sequences were both filmed in the same room, with the same camera position and nearly the same subjects, but each has different lighting conditions.

Each data set was divided into training and testing examples. On the traffic data, the video of the left road became the training set, and the video of the right road became the testing set. The other videos were split temporally into training and testing sets—all frames prior to a certain time index were for training and the remainder for testing. In the traffic and mwalk data sets, the test sets contain some objects that do not appear in the training sets, so success requires generalization.

Discarding the frames in which background subtraction detected no moving objects, the traffic data contains 1432 training frames and 2285 testing frames, the walking data contains 401 training and 200 testing frames, and both the mwalkA and mwalkC data sets contain 1201 training and 1200 testing frames. After discarding image areas devoid of moving objects, the average image size across all data sets was approximately 13,000 pixels.

Segmentation performance on each data set was measured by randomly selecting three training and testing subsets. Every training subset contained 200 images and every testing subset contained 40 images. Performance was measured by averaging the results across the subsets. For example, on the traffic data, a SANE model was trained on the first training subset and tested on the first testing subset, another SANE model was trained on the second training subset and tested on the second testing subset, and so on. Performance statistics were computed for each testing subset and the averages are reported in the graphs of this section, along with error bars indicating the standard error.

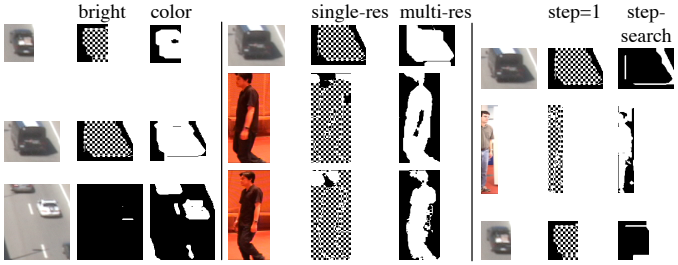

Fig. 7. The largest gains from color, multiresolution, and step-size search.

Although the motion segmentation pixels are all labeled either "0," indicating background, or "1," indicating moving object, multiple moving objects can be represented. Two non-overlapping moving objects in a frame will produce two groups of intra-connected "1" pixels, separated from one another by surrounding "0" pixels, each of which should be represented by a unique connected component in the inferred segmentation. Two pixels are in the same connected component if they are horizontal or vertical neighbors and have the same label. For comparison to an inferred segmentation, the pixels of each connected component are given a unique label. The surrounding "0" pixels are left unmodified.

Similarly, the output of SANE can contain multiple connected components. Because the "0" and "1" labels carry no meaning, all pixels are relabeled by connected component. An accurate segmentation should match one SANE component to each motion component and all the remaining SANE components should correspond to the non-moving pixels. There is no penalty for subdividing the background because there is no information available for judging the true segmentation of static parts of the scene.

For an image, consider an assignment such that each motion object component is paired with a single unique SANE component or else with no SANE component. Precision (P) is the fraction of all the matched SANE component pixels that cover the same image locations as pixels from their corresponding motion component. Recall (R) is the fraction of all motion component pixels that are covered by pixels from their corresponding motion component, also known as the true positive rate. The f-measure, $F = 2PR/(P + R)$, is a harmonic average of both factors. A perfect segmentation produces a precision, recall, and f-measure of 1, the worst possible values are 0s. There are many possible pairings of SANE and motion components; the segmentation quality is measured using the combination that maximizes the f-measure. The background subtraction process is imperfect, so even an apparently perfect segmentation might not score perfectly. The precision, recall, and f-measure are calculated for each image and then averaged to produce values for each data subset.

B. SANE results

After taking advantage of all the rotational and reflection invariances, the mwalkA and mwalkC SANE models averaged 10,700 edge pairs and averaged 187 patches per pair. Traffic averaged 32,201 pairs and 63 patches per pair. Walking averaged 26,332 pairs and 115 patches per pair. Naturally, each model has a large number of edge combinations that were only observed once or twice during training, but because they were infrequently observed, these pairs have a low prior probability of appearing in a finished segmentation and the inability to adequately represent their associated patch distributions is unlikely to affect performance.

There are four types of SANE model and four inference algorithms. Figure 6 shows the f-measure, precision, and recall for each combination on each of the four data sets. The four model types are brightness-only, single resolution (*bright*); brightness-only, multiresolution (*bright,mres*); color, single-resolution (*color*); and color, multiresolution (*color,mres*). The four inference algorithms are standard belief propagation (*bp*), belief propagation followed by parity-flipping iterative conditional modes (*bp,picm*), belief propagation using step-size search (*bp,ss*), and the combination of all three methods (*bp,ss,picm*). The *bp,ss* algorithms choose the best step size by maximizing each MRF's configuration value, not by comparing its outputs to ground truth.

The SANE algorithm was more successful at segmenting the walking and traffic data sets than the mwalk sets. This is not surprising since the walking and traffic sets feature objects against relatively uncluttered, untextured backgrounds. The cars demonstrate less internal variation than walking people, and it's easier to learn to segment a single walking person (walking) than multiple people (mwalk), some of whom do not appear in the training data. The highest average per-image f-measure on each data set ranged from 0.690 (mwalkA) to 0.825 (traffic). Adding color information typically boosted precision. Figure 5 contains several of the best and worst color, multiresolution results on each data set.

Multiresolution models using only brightness features almost universally underperformed their single resolution counterparts, trading too much precision for increased recall. The half-resolution layers' patches cover greater image area, and perhaps that reduced the effectiveness of the brightness features, since their average distances from an object boundary will subsequently increase. On the walking and mwalk data sets, the color multiresolution models were roughly equal to or slightly better than color single resolution models on the same data, while they were slightly worse than single-resolution color on traffic data. Figure 7 contains several examples in which multiresolution modeling or the addition of color features corrected a checkerboard segmentation pattern, which can be produced by belief propagation divergence.

None of the four inference algorithms dominated the others. Step-size search improved all the walking results, but did not always help on other data sets. It never appears to make results significantly worse, so it can be considered the best inference algorithm. The parity-flipping ICM step, which was originally added to fix the checkerboard patterns, almost never made a substantial difference. This is not surprising, since it can easily make a checkerboard worse by making a half-wrong area completely mislabeled if it picks the wrong square for its first flip.

Figure 5 presents the four best and four worst results for the color, multiresolution models of each data set, inferred using the step-search method. Unsurprisingly, clear foreground-background separations, such as a single dark car or an individual surrounded by white pixels, appear in the best collection, while multiple objects and greater clutter produced some of the poorer output. In the mwalk data, many of the bad examples contain objects at the edge of the frame, or only small object pieces, such as an arm or leg. These cases might indicate that the absence of part of the object boundary makes inference difficult, perhaps because that situation is not well-represented by the learned shape model or because a small input image makes the creation of the half-

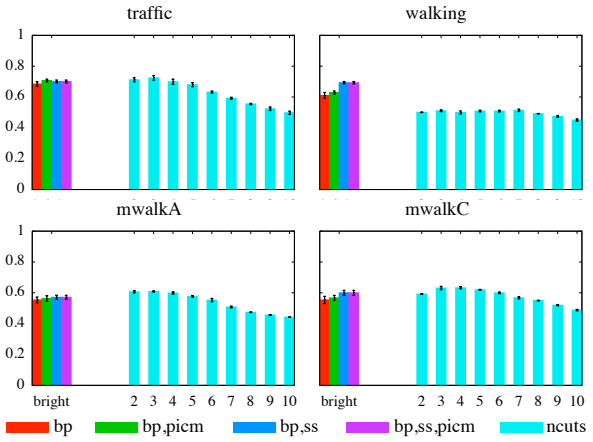

Fig. 8. The f-measures of SANE and normalized cuts set for 2-10 segments.

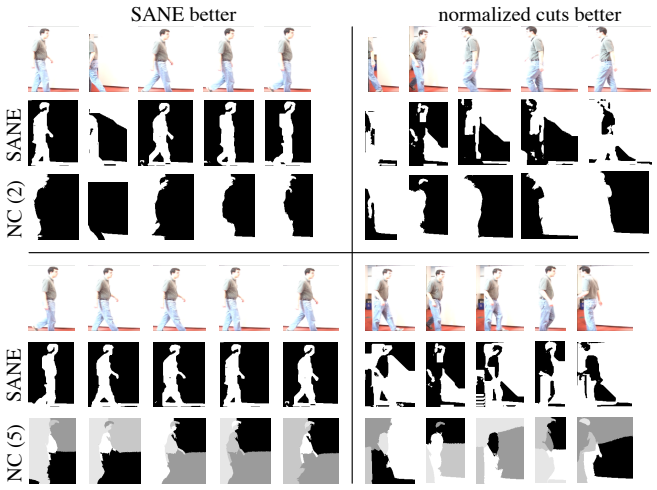

Fig. 9. Walking examples in which SANE most outperformed two and five-region normalized cuts and vice-versa.

resolution MRF level difficult. Weiss analyzed the performance of loopy belief propagation by considering the number of times evidence is over-counted [43]; perhaps the different distribution of message over-counting at the image boundary might cause some of these problems. Only the traffic data contains multiple moving objects in some example frames, and the three worst examples contain two full or partial cars. Because there are only two labels, segmenting multiple objects correctly requires an unforgiving alternation between foreground and background labels. Failing to close the boundary of one car may make it extremely difficult to correctly label the other. Increasing the number of available segmentation labels may alleviate this problem.

Discarding ICM, which did not affect the results, the color, multiresolution and step-size search variations all fixed some checkerboard patterns, the most visible effect of belief propagation non-convergence. In Figure 7, all but one of the three most improved examples from each technique contain examples of fixed checkerboards. In other cases, the changes improved the detection of object boundaries and the ability to distinguish them from internal borders. The model might benefit from more sophisticated local features, which could reduce reliance on belief propagation accuracy, or a higher-level shape model, which could use stronger priors to overcome uncertain image information.

C. Comparing SANE and normalized cuts

Comparing SANE to normalized cuts reveals that a trained segmentation algorithm can outperform a good general-purpose segmentation algorithm. Although normalized cuts is an excellent generic image segmentation algorithm, SANE offers comparable performance on the traffic, mwalkA, and mwalkC data sets, and greatly outperforms it on the walking data.

Normalized cuts [35] is a well known general-purpose segmentation algorithm. It computes a matrix that represents the similarity between all pairs of image pixels and then segments the image by solving a related eigenproblem. The implementation used for these experiments, developed by Cour, Yu, and Shi [6], measures the difference between image pixels by searching for the presence of brightness contours found by the Canny edge detector [5] between every pair of pixels. It was not trained or parameter-tuned for the test data set.

Normalized cuts uses a user-specified number of region labels. Just as the binary SANE outputs were processed into connected components, so are the normalized cuts region labels, so both algorithms are judged by how well they cover the moving objects and how well they separate them from their surroundings and each other. The assignments between normalized cuts and motion components with the best f-measures were used.

Because the normalized cuts code only uses grayscale images, for comparison, SANE was also restricted to brightness features. Since multiresolution only harms the performance of brightness-only models, only comparisons to the single-resolution brightness SANE are presented. Color features improved SANE's output on all the data sets, but it would have been unfair to graft crude color features onto the publicly available normalized cuts code for the purposes of this comparison. Presumably color features could improve normalized cuts' performance too, but those are best added by researchers with expertise in that code, since badly-designed color features could unfairly favor SANE.

In Figure 8, it is clear that normalized cuts closely matches SANE's performance on the traffic and the two mwalk data sets over a range of user-specified label-set sizes. But on the walking data, no normalized cuts setting matched SANE's performance. Figure 9 makes it clear that normalized cuts does a poor job on many examples because it considers the external boundaries of the person to be less significant than his internal visual contrasts, or because it fails to detect pieces of the boundary, perhaps due to the saturation of the pixels bordering the whiteboard. This highlights the advantage of a segmentation algorithm that can automatically adjust to a particular visual environment and that has a strong shape model to compensate for weak local information.

D. Comparing SANE and Martin's edge detector

Martin et al.'s learned boundary detector [22] is a good contrast to the SANE approach. Like SANE, the Martin algorithm can be trained on a database of example segmentations, but unlike SANE it attempts to discover the best local boundary detector without learning a shape model or propagating information to neighboring sites. Martin detectors attempt to classify each point as "boundary" or "non-boundary" based only on the local image data. Thus the trained Martin detectors, which outperform many standard edge detection algorithms on the Berkeley Segmentation Database [23], rely on a strong local detection model, while

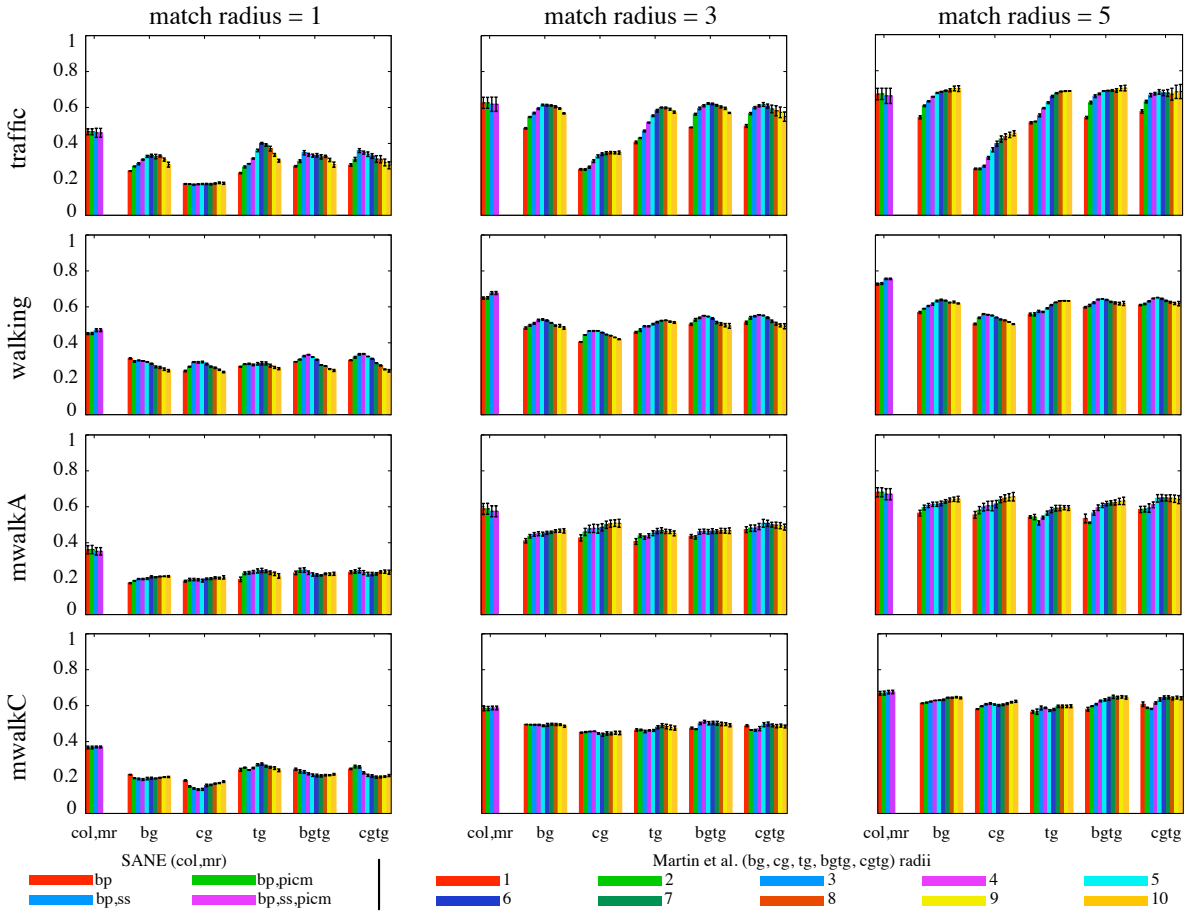

Fig. 10. The f-measures of color, multiresolution SANE and Martin et al. models for different match radii.

SANE uses weak local detectors and relies on shape information for joint boundary determination.

The comparison is difficult because the Martin algorithm outputs a map of boundary detections, not a segmentation. Each pixel in the boundary map has a value between 0 and 1 that indicates the probability of it being a boundary pixel; the map must be thresholded to produce a binary boundary detection. For these comparisons, SANE outputs are converted to binary boundary images by labeling all horizontal and vertical transitions between "0" and "1" segment labels as boundary locations.

The Martin and Fowlkes implementation of the edge detection training and testing algorithms [21] was used to compare the Martin detectors to SANE. Five detector classes were trained on the SANE data sets: brightness gradient (*bg*), color gradient (*cg*), texture gradient (*tg*), brightness and texture gradient (*bgtg*), and color and texture gradient (*cgtg*). The *cg* detector contains no brightness features, but the *cgtg* detector does. Each detector uses image data from a fixed, prespecified radius around the location to be classified.

For these experiments, the Martin detectors were trained on each SANE training subset, just as the SANE models were, and tested on the corresponding testing subsets. In the original implementation, the detector radius varied according to the size of the input image. In the Berkeley Segmentation Database, the image size indicates the scale of the image boundaries. This is not consistently true in the SANE data sets, and fixed radii performed better in early trials. The *bg*, *cg*, and *tg* detectors were trained and tested with radii ranging from 1 to 10 pixels. The original Martin and Fowlkes code used a 1:2 ratio of *bg* radius to *tg* radius in the bgtg detector, so these detectors ranged from 1:2 to 10:20 pixels. For the same reason, the brightness, color, and texture radii in the *cgtg* detector ranged from 1:2:2 to 10:20:20 pixels.

Comparing boundary images is a difficult task. Simply overlapping two binary boundary maps and counting matches is not useful because a small misalignment can cause a complete mismatch, and the resulting score could indicate that two very similar boundaries are extremely different from one another. Martin et al. compared boundaries by solving a bipartite graph matching problem. Given a binary detected boundary image and binary ground-truth boundary image, each boundary pixel in each image is considered as a graph node. The goal is to match each detected node to a unique ground-truth node. In turn, each ground-truth node can match at most one detected node. If a detected node matches a ground-truth node, it is considered to be an accurate detection. If it cannot be matched, it is inaccurate. The distance between two matched nodes cannot exceed a specified maximum. As with the detector radii, the original implementation varied these match distances relative to image size, but the modified implementation does not. Again, the reason is that the sizes of our images do not indicate the scale of the boundaries.

In order to transform the soft boundary maps of the detectors into binary boundary maps, the Martin and Fowlkes error-measurement code chooses the threshold that maximizes the f-measure across the testing set. In a deployed application that required binary boundary detection, this threshold would need to be chosen during training. This extra degree of freedom benefits

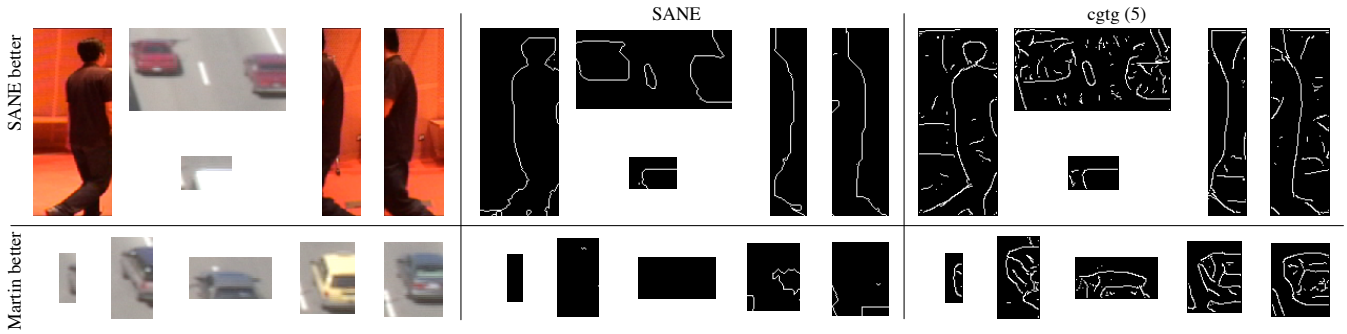

Fig. 11. The examples for which color, multiresolution SANE most outperformed *cgtg* Martin with features radius 5 using match distance 3 and vice-versa.

the Martin detectors in any comparison to SANE.

The results in Figure 10 make it clear that SANE outperforms Martin on these data sets. The examples in Figure 11 illustrate that SANE's advantage lies in its higher precision. While the Martin detectors have little trouble finding object boundaries, they also detect many other non-object boundaries in the process. In all four data sets, the color, multiresolution SANE f-measures outperform all Martin detectors at all feature radii using the strict matching distance of 1. As the match maximum distance relaxes to 3 and 5, the Martin results improve, especially their recall. At radius 3, the best Martin results only match SANE on the traffic data. At radius 5, the best results match or slightly surpass SANE on traffic, and match mwalkA and mwalkC, but still lag behind on the walking data. Figure 11 contains examples in which SANE's performance was superior to *cgtg* with feature radius 5 (one of the overall best-performing Martin detectors) using match maximum distance 3. Most of the Martin results are riddled with non-object boundaries compared to the SANE output. The examples for which Martin performed better demonstrate that extreme SANE recall failures are responsible for several of the cases in which *cgtg* is superior.

Much of the performance disparity could derive from the data sets that each algorithm was originally designed for. Although the Martin detectors were retrained on the SANE data sets, the Berkeley Segmentation Database contains low-noise, high-resolution images, and much of the success of the Martin detectors on that set appears to be due to the use of the sophisticated texture gradient features [22]. The SANE sample images have much lower resolution and much poorer quality. There is less texture information available in each image, and boundaries are not as sharp. SANE's local edge detectors are simple, but the shape model compensates. Also, the segmentations in the Berkeley database can include visually significant internal object regions, and it's unsurprising that a detector designed to detect these boundaries will have difficulty learning to ignore them when they do not correspond to object boundaries. The shape information stored in the SANE model appears to be more useful on this data than the sophisticated local Martin detectors. The fact that the SANE data requires different segmentation skills than the Berkeley database is a strong argument for its potential utility, in addition to the aforementioned advantages of learning from motion-labeled training data.

E. Comparing SANE and LOCUS

To further investigate SANE's performance, we compared it to LOCUS, a learning-based, top-down segmentation algorithm developed by Winn and Jojic [45]. LOCUS segments a group of images based on the assumption that they each contain an instance of the same object class. Unlike SANE, which segments each image independently, LOCUS jointly segments a collection of images, allowing information in any image to influence the segmentation of the others.

LOCUS provides a good comparison to SANE. Its shape model, which includes a probabilistic template describing the expected overall shape of the objects, is much more global than SANE's shape model, which learns the relationships between neighboring boundary segments. Furthermore, LOCUS does not attempt to learn a common image model, while SANE segments new images using previously learned models of the image properties of object boundaries.

LOCUS typically operates without supervision, but to provide a fair comparison with SANE it was also run in a supervised mode. This was accomplished by adding the training examples to the collection of images and fixing their segmentations to be the motion-defined ground truth. In this way, their shape information influences the joint segmentation process on the other images. LOCUS was run on the walking and mwalkA data sets. It could not be run on the traffic data because many of those images contain multiple cars and LOCUS is not designed to segment multiple objects in an image.

The LOCUS algorithm requires a set of equal-sized images. Therefore, it was provided with the uncropped images and bounding boxes equivalent to the crops used to create the SANE data sets. On each test image, the pixels external to the bounding box were assigned the "background" label, while the segmentation of the interior pixels was inferred by the algorithm. As a result, the LOCUS algorithm used the areas in each test image that lacked moving objects as labeled training data for the construction of the background image model for that image. When training and testing the SANE models, these regions were simply discarded on the assumption that in the absence of motion the true segmentation was unknown and unavailable to train or test against.

The results of the comparison are in Figure 12. On the walking data sets, performance of unsupervised and supervised LOCUS was very similar to that of color, multiresolution SANE. LOCUS had higher recall, but lower precision. The small advantage of SANE in f-measure was due to a few cases in which LOCUS failed to find any object in images — several examples can be seen in the figure. These examples typically occurred when the figure was far over to the side of the bounding box, which caused LOCUS to fail because it became stuck in a local minimum that labeled all pixels as "background."

LOCUS outperformed SANE on the mwalkA data, especially in supervised mode. Examining the three images on which supervised LOCUS most outperformed SANE, it appears that LOCUS's strong global shape model enabled successful segmen-

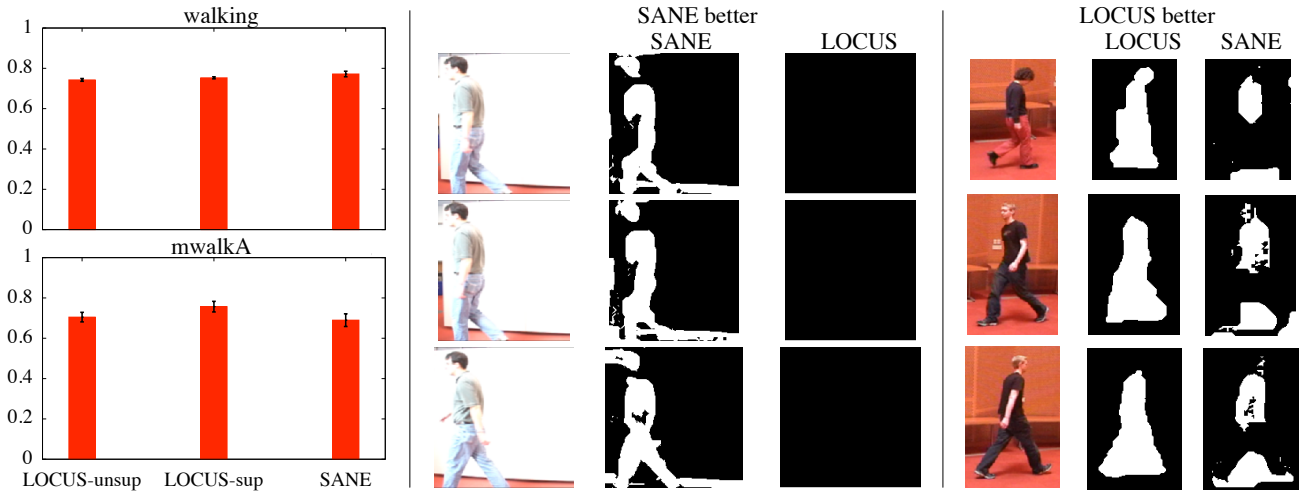

Fig. 12. F-measures for SANE and supervised and unsupervised LOCUS and the examples for which SANE most outperformed LOCUS and vice-versa.

tation of some parts of the figures that had color values similar to surrounding background region.

SANE's performance was competitive with LOCUS's on these data sets. LOCUS had the advantage of joint segmentation, which enables more sharing of shape information, and a global shape representation. It also relied on the Canny edge detector, a robust local edge detector, as an image feature. On the other hand, SANE used only local shape information, simpler image features, and segmented each image independently. The fact that this mid-level approach achieves performance near or in some cases exceeding LOCUS's performance indicates that it can be a useful enhancement to or replacement for top-down segmentation methods. Additionally, as described in the next section, SANE's local shape information can generalize well to other object classes, while LOCUS's global shape templates are usually class specific.

F. Generality

The previously discussed experiments have all involved testing and training data drawn from the same location and environmental conditions. It is also fair to judge a learning-based segmentation system on its generality. Generality is the effectiveness of a segmentation model learned in a particular environment or on a particular class of objects when applied to new environments or classes. Experiments testing SANE's generality indicate that the shape model, SANE's central feature, is both highly general and crucial to its performance.

The simplest generality experiment is to apply a model trained on one type of data to another. In these crossed-model experiments, single-resolution models trained on the traffic training subsets segmented examples from the walking testing subsets, and vice-versa. The mwalkA and mwalkC models were also crossed.

Unsurprisingly, running the walking models on the traffic data and vice-versa produced much worse results than running on the matching test set, as seen in Figure 13. Crossing the mwalkA and mwalkC models produced roughly no change in the brightness results and a relatively small decline in color model performance. The two sequences came from the same environment and contained many of the same people, but were filmed under different lighting conditions. Given the well known sensitivity of color values to lighting, it is unsurprising that the color results were affected more by the swap. The grayscale results indicate that with the appropriate selection of image features, SANE can be robust to lighting variations.

SANE uses simple brightness and color image features which may not be robust to environmental change, but could be replaced with more sophisticated features, such as those used in the Martin detectors. A more fundamental issue is the generality of the shape information in each model. Is SANE learning shape probabilities specific to particular environments or particular object classes, or are they useful for generic segmentation tasks? What aspects of the shape information are most important to its performance?

Because SANE's compatibility functions factorize into image and shape components, it is possible to explore the generality of the shape model independently from that of the image features. The image features of a trained model can be left unmodified while the probability of edges and neighboring edge pairs are altered. Observing the effects of these alterations on the segmentation results will reveal the influence of shape information on the results and the degree of generality of each trained shape model.

This shape-transfer experiment was performed on the color, single-resolution traffic, walking, and mwalkA models, and on the color, multiresolution mwalkA model. The bp,ss algorithm was used to find the segmentations. The shape information (the $P(E_i, E_j)$ and $P(E)$ probability functions) in each model was replaced to create four alternative models. The generic-shape (gs) model contains no edge probability information, with the exception of the requirement that neighboring patches form closed contours. The generic-pair (gp) model uses the original $P(E_i)$ functions, but defines $P(E_i, E_j) = P(E_i)P(E_j)$. Examining its performance can indicate whether the pairwise statistics collected in training are useful in producing accurate segmentations. Finally, the other two trained shape models were substituted to examine whether shape learned for one class of objects or in one environment would be useful in a different situation. For example, a traffic model's shape information would be replaced by the shape information from an mwalkA or walking model. All the shape-modified models were tested on test sets appropriate to the source of their *image* training, therefore any variation in performance is solely due to shape factors. Comparisons of the modified models with each other and with the original shape model are in Figure 14.

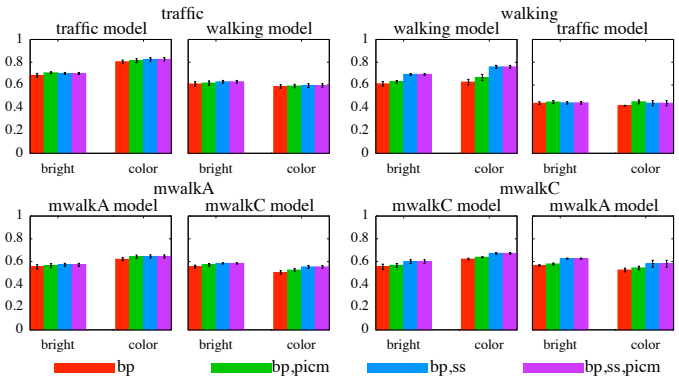

Fig. 13. F-measures from crossing traffic and walking models, and from crossing mwalkA and mwalkC models.

The overall pattern of results reveals that shape information is important to the results and exhibits surprising generality. The most striking confirmation of this hypothesis is in the walking data, in which the gs and gp models severely underperformed all three of the learned models, which performed very similarly to one another. It is not surprising that shape is important on this data set, given the saturated background that tends to wash out local image edge evidence, and it is to be expected that the wide variety of people in the mwalkA data lends itself to the task of segmenting the figure in walking. But it's shocking that the traffic shape information provides almost equal performance. Furthermore, it is clear from the relatively poor performance of the gp model that the pairwise information was crucial on this data set. Even learning the pairwise shape probabilities of the blob-like cars substantially boosted performance, indicating that this knowledge is both important and general.

The traffic and mwalkA data sets also produced performance differences. Across all data sets, the learned shape models had higher levels of precision than the gs and gp models. Walking is the only data set where there is a similarly dramatic improvement in recall, but the superiority of learned models is still reflected in higher f-measures in most cases. The mwalkA single-resolution results do not show much of a pattern because the learned shape models tend to decrease recall compared to gs, but in the multiresolution results there is a consistent pattern indicating an advantage of the learned models due to better recall performance.

Overall, the data indicate that the shape information transferred extremely well between object classes and environments. Additionally, the gp model was the clear worst performer in every case except the walking data, and that case is most likely due to the extremely poor performance of gs, which probably floundered because the poor image data was combined with the absence of any local prior probability on edge assignments.

These results indicate that SANE is learning generic properties about object boundaries. The information is extremely useful when image data is poor and the pairwise statistics are important. Shape information from a single class example (walking) can be useful on a variety of other examples in a different environment (mwalkA), and vice-versa. The performance of traffic shape suggests that even if shape is not very important in the environment it is learned in, it can be useful in a new environment. The contrast between the shape-transfer results and the crossed-model results suggest that future work on learning more generalized SANE models should focus on improving the image features.

G. Future work

Future extensions to SANE might include the use of more sophisticated local features, such as those used in the Martin edge detectors [22] and extending the model to use four segmentation labels, which would allow for the production of arbitrary segmentations, even separating overlapping objects. More accurate inference and learning algorithms might also improve the results.

The current implementation of SANE requires a few minutes to segment images composed of a few thousand pixels, but can take up to a few hours for images containing tens of thousands of pixels. However, these running times are dominated by the construction of the MRFs, rather than by inference, and they could be greatly reduced by software optimizations.

Currently, SANE can learn a scale-robust model by being exposed to example objects of different scales such as differently sized people or cars. This could be enhanced by changing the training procedure so the images are presented at a number of different scales, or by modifying the model to account for scale more directly, perhaps in the context of the multiresolution approach. The related issue of image resolution might be more difficult to tackle—the relatively poor performance of the Martin edge detectors suggest that different features are useful for segmentation on high and low-resolution images.

SANE acquires its training data through background subtraction, but the Stauffer algorithm cannot handle videos with moving backgrounds, or separate multiple moving objects whose paths intersect. In order to learn under these conditions, more sophisticated background subtraction or motion segmentation algorithms would be required, particularly those able to use optical flow.

Further work might integrate SANE with other algorithms, such as manipulation or object recognition, in which segmentation can be optimized against other objective criteria, such as its utility in assisting the performance of another task. The concept of using manipulation to provide segmentation knowledge is similar to recent work by Fitzpatrick [7], but with the goal of learning better segmentation algorithms rather than focusing on detecting specific objects and their boundaries. Extra sources of segmentation information might allow a future SANE algorithm to distinguish between objects and shadows and to learn to segment a larger class of objects.

V. Conclusions

The results demonstrate that the SANE algorithm provides advantages over a prominent non-learned image segmentation algorithm and over a trained local boundary detector. SANE's output is competitive with a learning-based, top-down segmentation algorithm despite using local neighborhood shape information instead of a global class-specific model. Its ability to learn the visual features appropriate to segmentation in a particular visual environment and to combine those features with a strong learned shape model had clear performance benefits. It demonstrates that simple boundary detection features and local neighborhood shape information can be combined into a powerful segmentation algorithm. The automatically labeled training data made the use of simple and powerful non-parametric density estimators practical.

Furthermore, experiments on generality demonstrate that the shape information learned is not specific to a particular object class or environment, and that it is responsible for much of SANE's performance. This validates the utility and practicality of SANE's shape model, the central structure of the model.

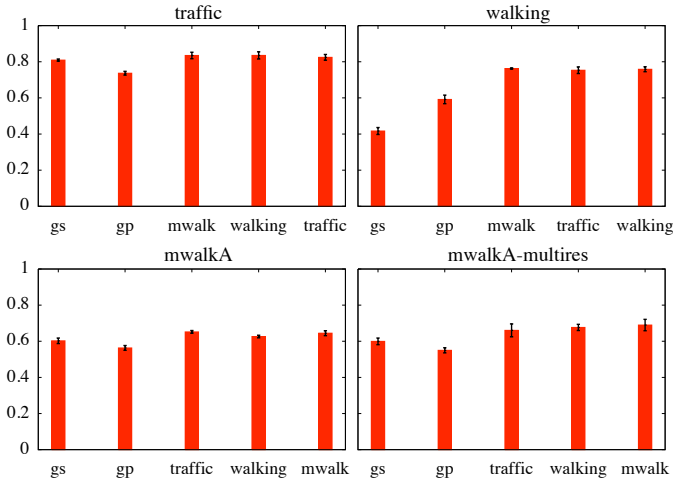

Fig. 14. The f-measures from the shape-transfer experiments.

Most importantly, this learning was done with self-labeled training data, optimizing against an objective and useful segmentation ground truth. Therefore it is practical to train SANE to be a useful component of a deployed vision system in a new environment.

ACKNOWLEDGMENTS

The authors thank John Winn for adapting his LOCUS code and running it on the SANE data, and thank David Martin, Charless Fowlkes, Chris Stauffer, Joshua Migdal, Timothée Cour, Stella Yu, and Jianbo Shi for providing software and (in many cases) help with adapting it to the data. They also thank Andrew Cohen and Aude Oliva for post-doctoral support of Michael Ross, and thank Bill Freeman, Tomás Lozano-Pérez, Alan Yuille, and many others for their advice and encouragement. Additionally, the authors thank the anonymous reviewers and the TPAMI editors, whose questions, comments, and suggestions greatly improved the paper. This work was funded in part by the Defense Advanced Research Projects Agency (DARPA), through the Department of the Interior, NBC, Acquisition Services Division, under Contract No. NBCHD030010, and in part by the Singapore-MIT Alliance agreement dated 11/6/98.

This paper contains material previously published and copyrighted by the Massachusetts Institute of Technology [29] and the Association for the Advancement of Artificial Intelligence [32].

REFERENCES

[1] J. Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society, Series B(Methodological)*, 36(2), 1974.

[2] J. Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society, Series B(Methodoloical)*, 48(3), 1986.

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

[4] E. Borenstein and S. Ullman. Learning to segment. In *European Conference on Computer Vision*, 2004.

[5] J. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), November 1986.

[6] T. Cour, S. Yu, and J. Shi. Matlab normalized cuts segmentation code. http://www.cis.upenn.edu/∼jshi/software/, 2004.

[7] P. Fitzpatrick. *From First Contact to Close Encounters: A developmentally deep perceptual system for a humanoid robot.* PhD thesis, Masschusetts Institute of Technology, 2003.

[8] W.T. Freeman, E.C. Pasztor, and O.T. Carmichael. Learning low-level vision. *International Journal of Computer Vision*, 40(1), 2000.

[9] A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC, 2003.

[10] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6), November 1984.

[11] T. Heskes. On the uniqueness of loopy belief propagation fixed points. *Neural Computation*, 16, 2004.

[12] R. Jiroušek and S. Přeučil. On the effective implementation of the iterative proportional fitting procedure. *Computational Statistics & Data Analysis*, 19:177–189, 1995.

[13] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *International Conference on Computer Vision*, 1987.

[14] S.M. Konishi, A.L. Yuille, J.M. Coughlan, and Song Chun Zhu. Statistical edge detection: Learning and evaluating edge cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(1), 2003.

[15] M. Kumar, P. Torr, and A. Zisserman. Obj cut. In *Computer Vision and Pattern Recognition*, 2005.

[16] S. Kumar and M. Hebert. Discriminative random fields: a discriminative framework for contextual interaction in classification. In *International Conference on Computer Vision*, 2003.

[17] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, 2001.

[18] B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *European Conference on Computer Vision Workshop on Statistical Learning in Computer Vision*, 2004.

[19] A. Levin and Y. Weiss. Learning to combine top-down and bottom-up segmentation. In *European Conference on Computer Vision*, 2006.

[20] D. Marr. *Vision*. W.H. Freeman and Company, 1982.

[21] D. Martin and C. Fowlkes. Matlab boundary detection code. http://www.cs.berkeley.edu/projects/vision/grouping/segbench/, 2004.

[22] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(5), 2004.

[23] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *International Conference on Computer Vision*, 2001.

[24] J. Migdal and W.E.L. Grimson. Background subtraction using markov thresholds. In *IEEE Workshop on Motion and Video Computing*, 2005.

[25] K.P. Murphy, Y. Weiss, and M.I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Uncertainty in Artificial Intelligence*, 1999.

[26] S.E. Palmer. *Vision Science: Photons to Phenomenology*. The MIT Press, 1999.

[27] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

[28] X. Ren, C. Fowlkes, and J. Malik. Scale-invariant contour completion using conditional random fields. In *International Conference on Computer Vision*, 2005.

[29] M.G. Ross. *Learning Static Object Segmentation from Motion Segmentation*. PhD thesis, Massachusetts Institute of Technology, 2005.

[30] M.G. Ross and L.P. Kaelbling. Learning object segmentation from video data. Technical Report AIM-2003-022, MIT Artificial Intelligence Laboratory, 2003.

[31] M.G. Ross and L.P. Kaelbling. A systematic approach to learning object segmentation from motion. In *NIPS 2003 Workshop on Open Challenges in Cognitive Vision*, 2003.

[32] M.G. Ross and L.P. Kaelbling. Learning static object segmentation from motion segmentation. In *Twentieth National Conference on Artificial Intelligence*, 2005.

[33] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2003.

[34] A. Shashua and S. Ullman. Structural saliency: The detection of globally salient structures using a locally connected network. In *International Conference on Computer Vision*, 1988.

[35] J. Shi and J. Malik. Normalized cuts and image segmentation. In *Computer Vision and Pattern Recognition*, June 1997.

[36] E.S. Spelke, P. Vishton, and C. von Hofsten. Object perception, object-directed action, and physical knowledge in infancy. In *The Cognitive Neurosciences*, pages 165–179. The MIT Press, 1994.

[37] C. Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition*, 1999.

[38] Z. Tu, X. Chen, A.L. Yuille, and S. Zhu. Image parsing: Unifying segmentation, detection, and recognition. In *International Conference on Computer Vision*, 2003.
[39] Z. Tu and S. Zhu. Image segmentation by data-driven markov chain monte carlo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), 2002.
[40] M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. Tree-based reparameterization framework for analysis of sum-product and related algorithms. *IEEE Transactions on Information Theory*, 49(5), May 2003.
[41] M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. Tree-reweighted belief propagation and approximate ML estimation by pseudo-moment matching. In *Workshop on Artificial Intelligence and Statistics*, 2003.
[42] M.J. Wainwright and E.B. Sudderth. Personal communication, 2002.
[43] Y. Weiss. Belief propagation and revision in networks with loops. Technical Report 1616, MIT Artificial Intelligence Laboratory, November 1997.
[44] Y. Weiss and W.T. Freeman. On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, 47(2), 2001.
[45] J. Winn and N. Jojic. Locus: Learning object classes with unsupervised segmentation. In *International Conference on Computer Vision*, 2005.
[46] C. Yanover and Y. Weiss. Finding the M most probable configurations using loopy belief propagation. In *Advances in Neural Information Processing Systems*, 2003.

Michael G. Ross received the A.B. degree in Computer Science from Dartmouth College in 1998, and the Ph.D. in Computer Science from the Massachusetts Institute of Technology in 2005. From 2005 to 2007 he was a postdoctoral researcher at the Psychology Department of the University of Massachusetts Amherst, and he is currently a postdoctoral associate at the Department of Brain and Cognitive Sciences at MIT. His primary research interest is the use of machine learning techniques to improve the understanding of human vision and the development of computer vision.

Leslie Pack Kaelbling received the A.B. degree in Philosophy in 1983 and the Ph.D. Degree in Computer Science in 1990, both from Stanford University. From 1991 to 1999 she was a faculty member of the Department of Computer Science at Brown University. Since 1999, she has been a Professor of Computer Science and Engineering at the Massachusetts Institute of Technology. Her research interests include robotics, reinforcement learning, planning, computer vision, and machine learning. She is the founding editor-in-chief of the Journal of Machine Learning Research and a fellow of the Association for the Advancement of Artificial Intelligence.