Probabilistic Occlusion Boundary Detection on Spatio-Temporal Lattices

M.E. Sargin, L. Bertelli, B.S. Manjunath and K. Rose
Department of Electrical and Computer Engineering
University of California, Santa Barbara, CA 93106
{msargin,lbertelli,manj,rose}@ece.ucsb.edu

Abstract

In this paper, we present an algorithm for occlusion boundary detection. The main contribution is a probabilistic detection framework defined on spatio-temporal lattices, which enables joint analysis of image frames. For this purpose, we introduce two complementary cost functions for creating the spatio-temporal lattice and for performing global inference of the occlusion boundaries, respectively. In addition, a novel combination of low-level occlusion features is discriminatively learnt in the detection framework. Simulations on the CMU Motion Dataset provide ample evidence that proposed algorithm outperforms the leading existing methods.

1. Introduction

Detection of object boundaries on natural images has been studied extensively and found numerous applications in computer vision. In these applications, salient objects are often located *on* other objects or background when projected on the image plane. Therefore, occlusion provides crucial information on detection and localization of boundaries. More specifically, occlusion information introduces evidence to keep the relevant boundaries while rejecting the irrelevant ones within the objects. This effect is illustrated in Fig. 1, where the boundaries of occluding objects are of interest.

Recently, there has been much interest in 3D scene reconstruction from a single monocular image [21, 22, 6]. These studies employed a graphical model driven by monocular cues for reconstruction. The cues were designed to mimic human perception. However, for some cases, even human observers cannot infer depth ordering from a single image. We believe that motion of the camera or the objects help to resolve these disambiguities by providing occlusion information.

Detection of occluded regions is critical for accurate optical flow computation. For this purpose, Lim *et al*. [10] and Xiao *et al*. [27] integrated the motion estimation error, the former within a bi-directional Bayesian framework and the latter within a variational formulation. Later on, Sand and Teller [20] added divergence of the motion field in their variational framework. Their work was motivated from the well-known fact that motion discrepancies occur at the occlusion boundaries. Thompson *et al*. applied filtering to find these discrepancies [26]. Then, Black and Fleet introduced a Bayesian framework to track motion discontinuities [3].

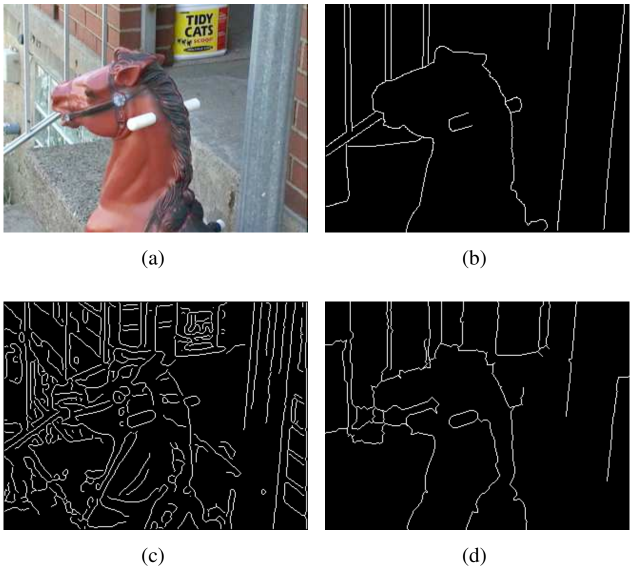


Figure 1. (a) An example frame from "rocking_horse" sequence [25]. (b) Corresponding ground truth. (c) Canny edges. Notice that not all the apperiance boundaries are occlusion boundaries. (d) The output of proposed occlusion boundary detector.

Layered motion segmentation is another major application where occlusion detection has received significant attention [28, 23, 16]. These studies used occluded regions to improve segmentation accuracy. In a similar scenario, edges, corners and t-junctions on the spatio-temporal domain are used to detect occlusions [1, 5], considering that the image sequence is represented as a spatio-temporal volume.

In this paper we introduce a probabilistic occlusion boundary detection framework defined on spatio-temporal lattices. The proposed framework enables combining occlusion information across multiple frames for robust detection. The main contributions can be summarized as follows.

- We define and minimize a probabilistic cost function for generating a spatio-temporal lattice, to be used as a sparse representation of the input data. In this setting, motion information is introduced from the first stage of the computational pipeline and cues extracted from multiple frames are combined in a principled way.
- A discriminative learning method is used to discover a novel combination of low-level occlusion features that help differentiate between occlusion and appearance boundaries.
- We exploit the data representation as a lattice structure, to introduce a probabilistic cost function for performing global inference of the occlusion boundaries. Junctions on the lattice are used as the elements of a graphical model, where the discriminative decisions are used as unary potentials and pairwise potentials are learnt from the data.

The rest of this paper is organized as follows. In Section 2, we introduce the probabilistic cost function to extract the spatio-temporal lattice. Low-level occlusion features are described in Section 3. In Section 4, we explain the probabilistic cost function to perform global inference. Experimental results are presented in Section 5, followed by conclusion in Section 6.

2. Spatio-Temporal Lattice Generation

We initially generate a sparse representation of the image sequence through over-segmentation. This representation provides coherent local regions in the spatial domain which are linked in the temporal domain. We will refer to these elements as *stempels* (Spatio-TEMPoral-ELements). We require stempels to have a rectangular lattice topology, to enable efficient inference over a graphical model. Therefore, stempel representation is a major departure from the earlier approaches [25], which allows us to perform feature extraction and specific higher level reasoning on the temporal domain as well.

Moore *et al.* introduced an algorithm in [15] for generating super-pixel lattices of still images, and proposed the 3D extension. The algorithm consists of successively partitioning the image into horizontal and vertical layers. The image is partitioned into layers recursively using a graph-cut based greedy scheme. This approach, the so-called *divide and conquer* heuristic, first partitions the image into two layers. The layers are then recursively partitioned into two, resulting in four layers and so on. The heuristic tries to solve easy local problems and lacks a cost function for joint partitioning of all layers. Instead, we propose a novel layer partitioning algorithm that optimizes a *globally defined* probabilistic cost-function. This cost function also enables us to fuse the optical flow information into the framework such that temporal coherence of the stempels is achieved.

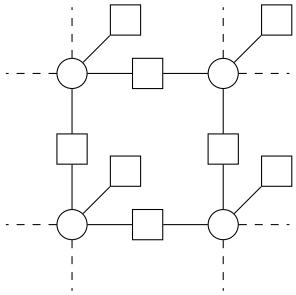


Figure 2. The factor graph: Variables and the factors are represented by circles and squares, respectively.

We make use of the factor graphs while defining the probabilistic cost-function used to compute the spatio-temporal lattice. We will first introduce a generic notation for the factor graphs and then provide the application specific components, since a similar cost-function will be applied for higher level reasoning in Section 4.

2.1. Factor Graphs

In graphical models, *global* joint probability of the variables can be written as a product of *local* joint probabilities that involves only a subset of the variables. These simpler functions are referred to as *factors*. Such factorization can be visualized using a factor graph where the factors are connected to its variables. The reader is referred to [8] for detailed review on factor graphs.

In this paper, we use the factor graph depicted in Fig. 2.1 where each variable at node i takes value in a finite set representing "states": $q_i \in \{1, \ldots, N\}$. The variables are associated with two types of factors. The first type is denoted by ϕ_i, which is associated with a single variable at node i. The second one is denoted by $\psi_{i,j}$, which is associated with a pair of variables at nodes i and j. In the factor graph, we wish to find the optimal state of the system, $\mathbf{q}^*$, that maximizes the joint probability of the variables $\mathbf{q} = \{q_i\}$:

$$\mathbf{q}^* = \underset{\mathbf{q}}{\operatorname{argmax}}\, P(\mathbf{q}) \tag{1}$$

$$P(\mathbf{q}) = \frac{1}{Z} \prod_i \phi_i(q_i) \prod_{i,j} \psi_{i,j}(q_i, q_j). \tag{2}$$

Here, Z is the normalization constant, which is also known as the partition function of statistical physics. This maximization problem can be solved effectively using well known methods such as loopy belief propagation [8], loop-corrected belief propagation [14] or turbo decoding [17].

2.2. Factors for Probabilistic Layer Partitioning

Assume that we want to partition an image into L vertical layers. We perform this partitioning by assigning to each pixel i a state $q_i \in \{1, \ldots, 2L-1\}$. In this representation, odd states are for layer pixels and even states are for the layer boundary pixels. The factors are defined such that the layer boundaries cover as many boundary pixels as possible, while maintaining the layer topology and 8-connectivity.

The probability that a pixel i is on a boundary, denoted $P_b(i)$, can be obtained using the derivation of [11] or [12]. Since we want the layer boundaries cover as many boundary pixels as possible, we define the single variable factors as follows:

$$\phi_i(q_i) = \begin{cases} P_b(i)P_p(i|q_i) & q_i \text{ even} \\ 1 & q_i \text{ odd} \end{cases} \tag{3}$$

Here, $P_p(i|q_i)$ is the prior probability that node i is on a boundary that separates layers $q_i/2$ and $q_i/2+1$ (by introducing this term, lattice regularization is obtained). Notice that we defined the single variable factors in terms of probabilities only for the layer boundary states. The factors for the layer states are simply set to 1. The latter is in fact analogous to *non-emitting* states in the Hidden Markov Models [18].

The layer topology is maintained using the factors connecting the horizontal pairs of nodes. Assuming that we are moving on a row from left to right, we only allow the states of pixels to increase by at most one and to maintain the same value only in the case of layer states. This guarantees that the boundaries are one pixel thick. More formally, we can write the horizontal factors as follows:

$$\psi^h_{i,j}(q_i, q_j) = \begin{cases} \delta_{q_i,q_j-1}, & q_i \text{ even} \\ \alpha\delta_{q_i,q_j} + (1-\alpha)\delta_{q_i,q_j-1}, & q_i \text{ odd} \end{cases}. \tag{4}$$

Here, $\delta_{m,n}$ is the Kronecker delta function:

$$\delta_{m,n} = \begin{cases} 1 & m = n \\ 0 & m \neq n \end{cases}. \tag{5}$$

The parameter α, is set considering the expected thickness of layers. The effect of horizontal factors is illustrated as a state diagram of a finite state machine in Fig. 3.

The vertical factors enforce the 8-connectivity of the vertical layer boundaries by only allowing certain state transitions, while moving from top to bottom on each column. As illustrated in Fig. 4, given the horizontal position of the vertical boundary state on the middle row, there are only three possible positions for the vertical boundary state on the last row that the 8-connectivity is satisfied. This is reflected on the vertical factors such that, while moving from top to bottom on each column, states of pixels can change by at most one. More formally, we can write the vertical factors as follows:

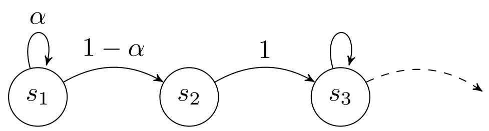


Figure 3. State diagram for vertical layer partitioning while moving from left to right on each row. States are named so that the s_k is the k^{th} state.

s_1	s_1	s_2	s_3	s_3
s_1	s_1	s_2	s_3	s_3
s_1	s_1	s_2	s_3	s_3

s_1	s_1	s_2	s_3	s_3
s_1	s_1	s_2	s_3	s_3
s_1	s_2	s_3	s_3	s_3

s_1	s_1	s_2	s_3	s_3
s_1	s_1	s_2	s_3	s_3
s_1	s_1	s_1	s_2	s_3

Figure 4. Three possible labeling schemes for the last row, given the middle row. Note that the system is first order Markovian since the allowed horizontal positions for the s_2 on the last row is independent of the first row given the middle row.

$$\psi^v_{i,j}(q_i, q_j) = \beta\delta_{q_i,q_j} + 0.5(1-\beta)(\delta_{q_i,q_j-1} + \delta_{q_i,q_j+1}). \tag{6}$$

Here, the parameter β adjusts the rigidity of the vertical layer boundaries. High valued β forces the boundaries to follow straight lines on the vertical axis. Similar to the horizontal factors, the effect of vertical factors is illustrated in Fig. 5.

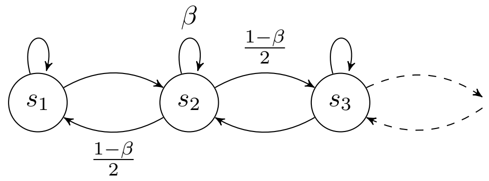


Figure 5. State diagram for vertical layer partitioning while moving from top to bottom on each column.

2.3. Joint Layer Partitioning of the Image Sequence

Given a sequence of images, we apply the horizontal and vertical probabilistic layer partitioning successively on a reference image, where overlapping stripes are used to compute the regularization prior $P_p(i|q_i)$ in (3). The reference frame can be selected arbitrarily, although a common practice is selecting the one in the middle of the sequence.

Once the layers for the reference image are obtained, the rest of the images can be partitioned using the optical flow information to compute the prior probability $P_p(i|q_i)$, *i.e.*, the probability that pixel i is on a layer boundary with state q_i. Let d_{i,q_i} be the distance of pixel i to the nearest pixel in the current image, which is covered by any one of the optical flow vectors pointing from pixels with state q_i in the reference image. Subsequently, $P_p(i|q_i)$ is obtained from d_{i,q_i} as follows:

$$P_p(i|q_i) \propto \exp(-d_{i,q_i}). \tag{7}$$

A sample spatio-temporal lattice, which is obtained using the proposed algorithm, is illustrated in Fig. 6.

3. Feature Extraction

Let $I(\mathbf{x}, t)$ be the image at time t where $\mathbf{x}$ is the spatial coordinates. We extract dense low-level occlusion features from $I(\mathbf{x}, t)$ based on the optical flow vector field, $D(\mathbf{x})$, calculated between $I(\mathbf{x}, t)$ and $I(\mathbf{x}, t + \Delta t)$. We use the method of [2] for robust optical flow calculation.

3.1. Low-Level Occlusion Features

The first low-level occlusion feature is inspired by the work in [27], where occluded pixels are detected in variational optical flow scenario by simply considering the following quantity:

$$f_1(\mathbf{x}, t, t + \Delta t) = \|I(\mathbf{x}, t) - I(\mathbf{x} + \mathbf{D}(\mathbf{x}), t + \Delta t)\|. \tag{8}$$

The second feature is computed following [20], where the divergence of the optical flow vector field is used to detect the occluded pixels:

$$f_2(\mathbf{x}, t, t + \Delta t) = |\nabla \cdot \mathbf{D}(\mathbf{x})|. \tag{9}$$

The third feature is inspired by [5], where the authors suggest that the edges and corners in the spatio-temporal domain correspond to the occluded pixels. These points of interest are detected using the minimum eigenvalue $\lambda_{\min}(\cdot)$ of the gradient structure tensor:

$$f_3(\mathbf{x}, t, t+\Delta t) = \lambda_{\min}((\nabla I(\mathbf{x}, t) \otimes \nabla I(\mathbf{x}, t)) * K(\mathbf{x}, \sigma)). \tag{10}$$

The operators $\otimes$ and $*$ represent outer product and convolution with a kernel, respectively, and the kernel is Gaussian:

$$K(\mathbf{x}, \sigma) = \frac{1}{2\pi\sigma^2} \exp(-\frac{\mathbf{x}^T\mathbf{x}}{2\sigma^2}). \tag{11}$$

We also propose to use the Frobenius norm of the gradient of the optical flow field in order to capture motion discontinuities:

$$f_4(\mathbf{x}, t, t + \Delta t) = \|\nabla \mathbf{D}(\mathbf{x})\|_F. \tag{12}$$

Together with the motion-based occlusion features, we use the probability of a pixel at $(\mathbf{x}, t)$ being a boundary pixel, $P_b(\mathbf{x}, t)$ as the last occlusion feature using the method of [11].

3.2. Features on the Spatio-Temporal Lattice

Let the image sequence consist of $2T + 1$ frames, where the frames are indexed by $t \in \{-T, \ldots, T\}$. We obtain the stempel boundaries using the the algorithm described in Section 2 with the reference frame at $t = 0$. Since the stempel boundaries coincide in the temporal domain, we extract low-level occlusion features on multiple reference frames and combine the corresponding features in the temporal domain. This makes the system more robust over the single reference frame case, which will be shown in Section 5.

Low-level occlusion features are defined over a pair of frames and extracted per-pixel. The pairs considered for feature extraction are illustrated in Fig. 7. For each pair, we combine each feature in the spatial domain by averaging them on stempel boundaries. Then, all the corresponding features in the temporal domain are combined by concatenation.

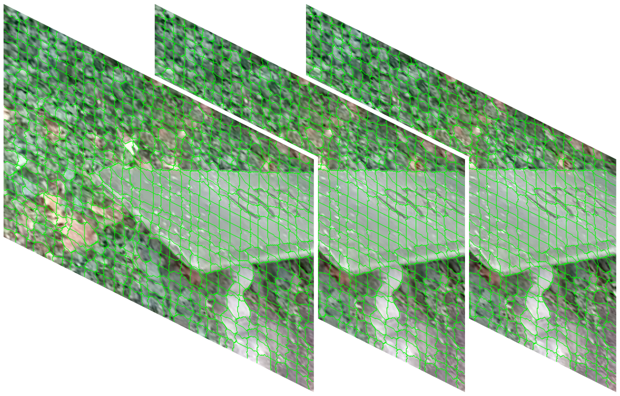


Figure 6. Stempel boundaries.

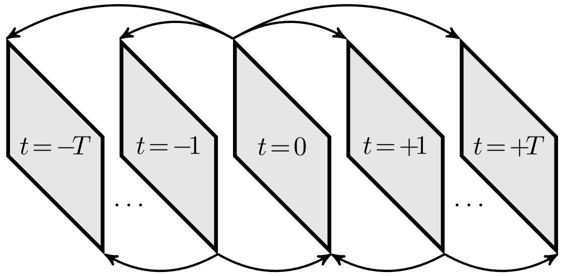


Figure 7. Pairs of frames considered for feature extraction. Ideally, there are $4T^2 + 2T$ possible pairs. Although, practically, some of the pairs can be pruned to reduce complexity.

4. Learning and Global Inference

We consider two classes, namely occlusion (o) and no-occlusion ($\bar{o}$). We train a Support Vector Machine (SVM) classifier [4] using the features extracted on the stempel

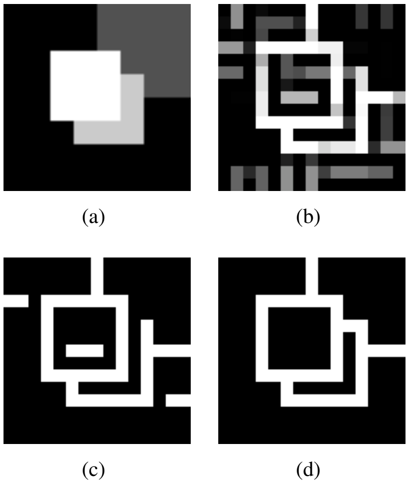


Figure 8. Toy problem for the global inference. (a) Square objects. The occlusion order is encoded by the intensity of objects. (b) $P(o|\mathbf{f}_k)$ corrupted by additive white Gaussian noise. (c) Maximum likelihood detection of the occlusion boundaries. Notice the spurious boundaries forming open contours and the gaps on the desired contours. (d) Output of the global inference.

boundaries. Let $\mathbf{f}_k$ be the occlusion features extracted on the stempel boundary k. Given $\mathbf{f}_k$ and the SVM model, the posterior probability of k being an occlusion boundary, $P(o|\mathbf{f}_k)$, is obtained using the method of [7].

Decision making on each stempel boundary using $P(o|\mathbf{f}_k)$ assumes that all boundaries are independent. This assumption does not hold in natural images since the boundaries of occluding objects tend to form closed contours. The dependency can be exploited to eliminate mislabeled spurious boundaries or to complete contours. A simple example for this is illustrated in Fig. 8.

Global dependency between the stempel boundaries can be decomposed into local dependencies of neighboring boundaries with the help of the lattice topology. In the rectangular lattice, junctions are the points where 4 stempel boundaries meet. Accordingly, a junction can be in 2^4 states considering the configurations of the stempel boundaries (See Fig. 9).

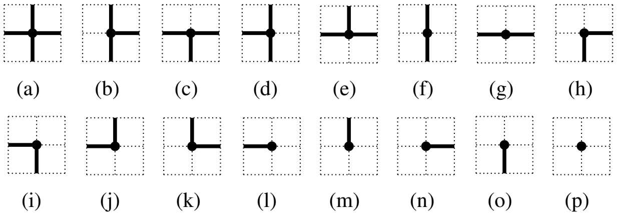


Figure 9. Possible states for a junction. Dark and dashed boundaries represent the boundaries labeled as o and $\bar{o}$ respectively.

Note that this representation brings redundant information since the stempel boundaries are shared by two junctions. Spatial continuity of the boundary labels is in fact maintained with this redundancy as we will see later.

4.1. Factors for Global Inference

Decomposition of the global dependency of the stempel boundaries into local dependency of the junctions enables us to use the factor graph described in Section 2.1 for solving the global inference problem. In this case, single variable factors represent the probability of a junction i being in state q_i. This probability can further be decomposed into product of the boundary probabilities joining at junction i, which are indexed with $N(i)$.

$$\phi_i(q_i) = \prod_{k \in N(i)} P(q_i|\mathbf{f}_k) \qquad (13)$$

The factors with pair of variables represent the joint probability of the neighboring junctions such that the spatial continuity of the boundary is maintained. The potentials of joint junction configurations are learnt from the data considering the rates of occurrence. The horizontal and the vertical factors are again considered separately since the dependency between states are different for vertical and horizontal neighbors.

5. Experimental Results

In this section, we demonstrate the performance of the proposed framework. First, we incrementally illustrate the gain of each contribution listed in Section 1. We then compare the proposed algorithm with leading existing methods.

Recently, Stein *et al.* introduced the CMU Motion Dataset [25] to show the effectiveness of motion cues for boundary detection. The dataset is publicly available [24] and it contains 30 short image sequences with the ground truth indicating the occlusion boundaries. The experimental procedure can be summarized as follows. First, spatio-temporal lattices with 24×32 stempels are generated on the image sequences. On the stempel boundaries, low-level occlusion features are extracted using the methods described in Section 3. These features are then used to train an SVM model with radial basis function kernel. Finally, the independent decisions for stempel boundaries are combined in the global inference module as explained in Section 4.

Turbo decoding [17] is used to perform inference on the factor graphs described in Section 2 and Section 4. In turbo decoding, rows and columns of the factor graph are alternately processed, and these horizontal and vertical inference processes "communicate" through their posterior probabilities. While the method does not guarantee convergence to the global optimum, the inference is effective and highly parallelizable since each row (or column) can be processed independent of the other rows (or columns).

The accuracy of the detection is quantified by precision, p, and recall, r. Precision is the probability that a pixel indicated as a boundary pixel by the segmentation algorithm is truly a boundary pixel. Recall is the probability that a true boundary pixel in the ground truth is correctly detected by the algorithm. These two measures are obtained using the algorithm described in [13]. One can finally combine p and r to get the F-measure, $F = 2pr/(p + r)$.

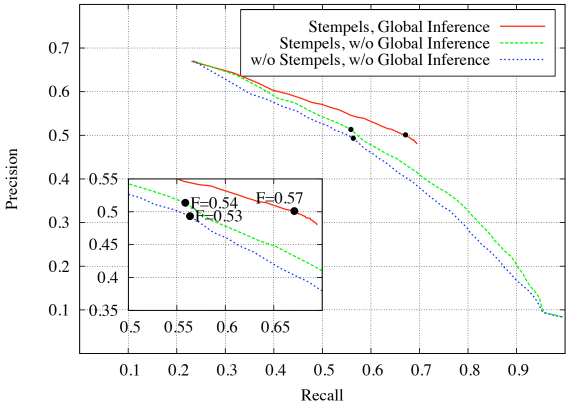


Figure 10. Precision and recall curves on the CMU Motion Dataset [25]. The results show that combining low-level occlusion features on the spatio-temporal lattice, improves the performance over the features with single reference frame. The performance is further increased by the global inference.

The gain of each contribution on the overall performance is illustrated in Fig. 10. The baseline is the independent decisions (without global inference) based on the features extracted only on the reference frame (without spatio-temporal feature combination). Combining the features, which are extracted on multiple reference frames improved the performance. The improvement was expected because multiple observations on the spatio-temporal lattice make the system more robust to noise, motion estimation error, *etc*. The best results are obtained when the global inference is applied on the decisions with the spatio-temporal feature combination. The performance improvement is primarily due to the fact that the spurious and missing stempel boundaries are addressed in the global inference by the configurations of neighboring stempel junctions (See Fig. 12 and Fig. 13).

We also compared the proposed method with leading existing methods on the same dataset. The precision and recall curves of [25] and [19] are roughly illustrated in Fig. 11 together with our best configuration. Here, note that [19] focuses on generic boundary detection and does not incorporate the temporal information to detect occlusions. As an additional reference point, the matting algorithm described in [9] was tested to yield ($p = 0.35$, $r = 0.20$). This inferior performance is primarily due to the fact that the matting algorithm attempts to cut out the object from the scene without considering occlusion information. In this sense, one can think of it as "region based" segmentation. However, our application is "edge based" i.e., only the occluding boundaries are of interest.

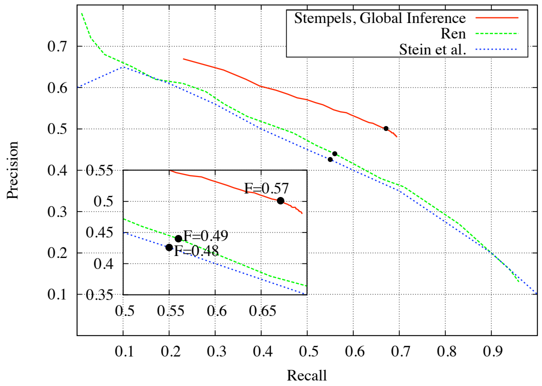


Figure 11. Precision and recall curves from Ren [19] and Stein *et al.* [25] together with the proposed method.

6. Conclusion

We presented an algorithm to detect the occlusion boundaries given a sequence of frames. Our contribution is three-fold. First, a probabilistic cost function is introduced to generate the spatio-temporal lattice on a sequence of frames. Second, a novel combination of features on the spatio-temporal lattice is learnt discriminatively. Finally, the independent decisions for boundaries are combined in a probabilistic model to perfom global inference.

Acknowledgements

This work was supported by the NSF Information Technology Research grant ITR-0331697

References

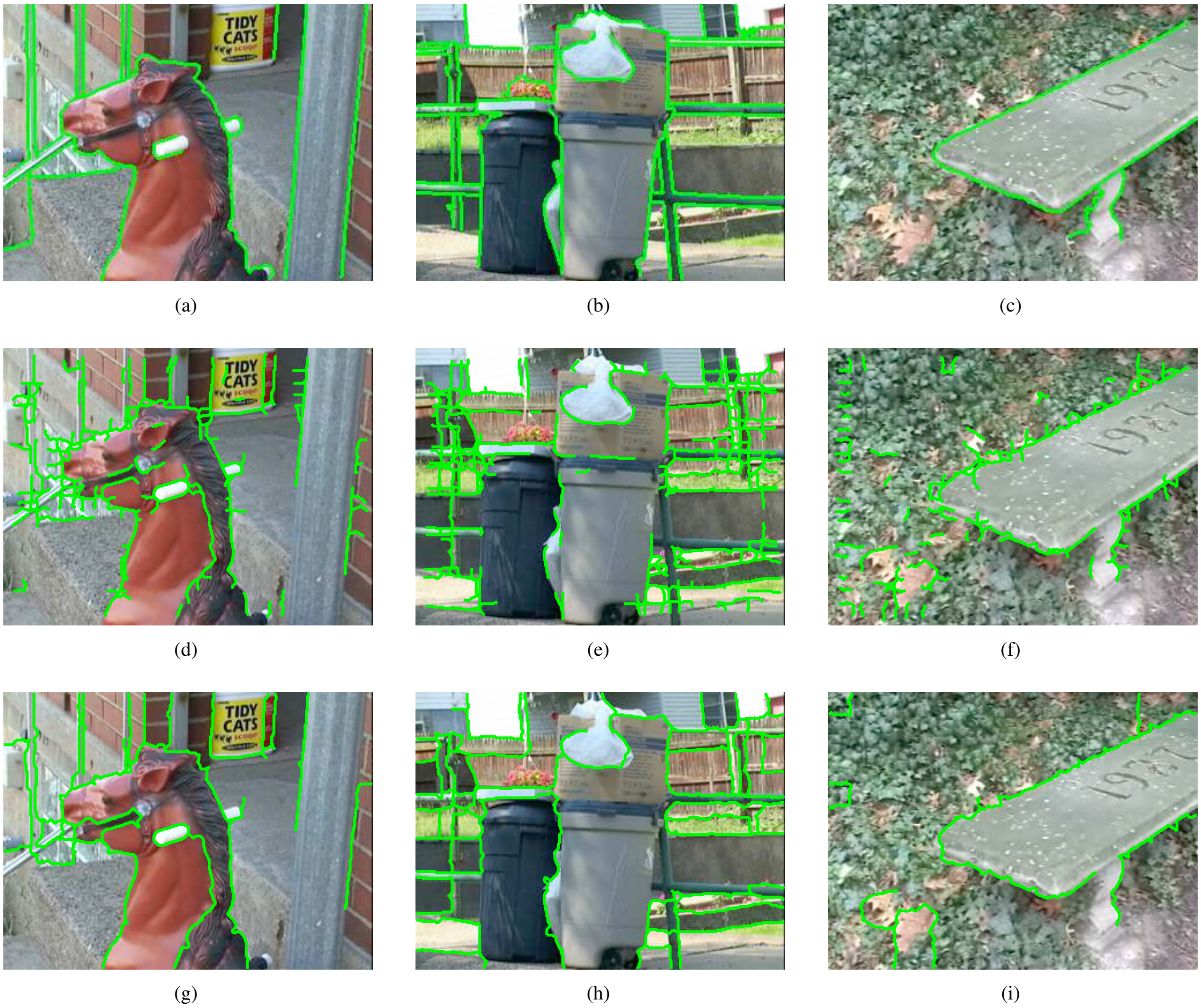
[1] N. Apostoloff and A. Fitzgibbon. Learning spatiotemporal t-junctions for occlusion detection. In *CVPR*, volume 2, pages 553–559, 2005.

[2] M. Black and P. Anandan. A framework for the robust estimation of optical flow. In *ICCV*, pages 231–236, 1993.

[3] M. J. Black and D. J. Fleet. Probabilistic detection and tracking of motion discontinuities. In *ICCV*, volume 1, pages 551–558, 1999.

[4] R. Fan, P. Chen, and C. Lin. Working set selection using second order information for training support vector machines. *The Journal of Machine Learning Research*, 6:1889–1918, 2005.

[5] D. Feldman and D. Weinshall. Motion segmentation and depth ordering using an occlusion detector. *TPAMI*, 30(7):1171–1185, 2008.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 12. Sample results on the CMU Motion Dataset. The ground truth is illustrated on the first row. The output with independent decisions based on the spatio-temporal feature combination is presented in the second row. Notice that the independent decisions result in missing and spurious boundaries. This effect is addressed by performing the global inference on the independent decisions, which is illustrated in the last row.

[6] D. Hoiem, A. N. Stein, A. A. Efros, and M. Hebert. Recovering occlusion boundaries from a single image. In *ICCV*, pages 1–8, 2007.

[7] T. Huang, R. Weng, and C. Lin. Generalized bradley-terry models and multi-class probability estimates. *The Journal of Machine Learning Research*, 7:85–115, 2006.

[8] F. Kschischang, B. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory*, 47(2):498–519, 2001.

[9] A. Levin, A. Rav-Acha, and D. Lischinski. Spectral matting. In *CVPR*, pages 1–8, 2007.

[10] K. P. Lim, A. Das, and M. N. Chong. Estimation of occlusion and dense motion fields in a bidirectional bayesian framework. *TPAMI*, 24(5):712–718, 2002.

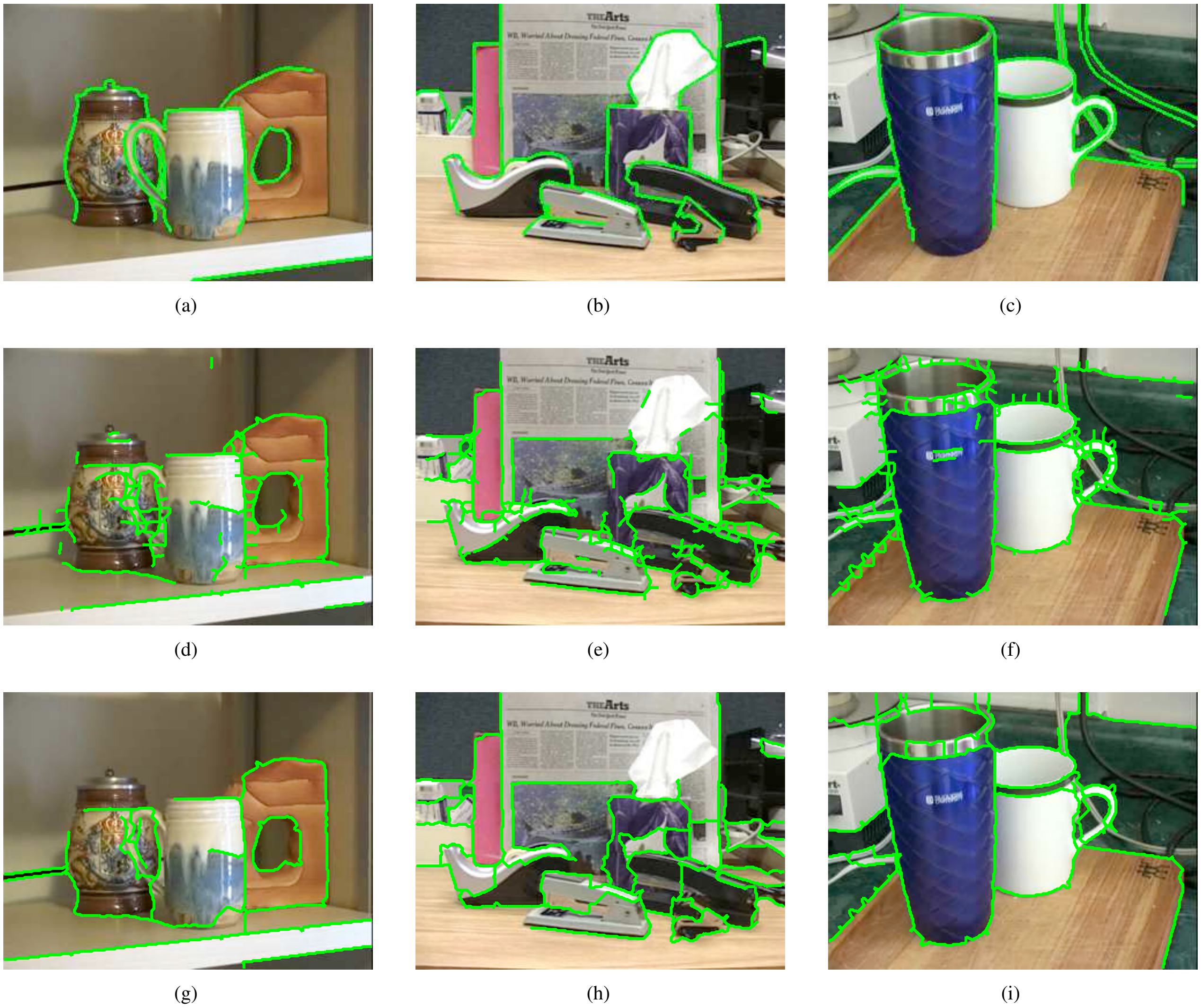
[11] M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik. Using contours to detect and localize junctions in natural images. In *CVPR*, pages 1–8, 2008.

[12] D. Martin, C. Fowlkes, and J. Malik. Learning to detect natural image boundaries using local brightness, color, and texture cues. *TPAMI*, 26(5):530–549, 2004.

[13] 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 *ICCV*, volume 2, pages 416–423, 2001.

[14] J. Mooij and H. Kappen. Loop corrections for approximate inference on factor graphs. *The Journal of Machine Learning Research*, 8:1113–1143, 2007.

[15] A. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones. Superpixel lattices. In *CVPR*, pages 1–8, 2008.



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 13. Sample results on the CMU Motion Dataset.

[16] A. S. Ogale, C. Fermuller, and Y. Aloimonos. Motion segmentation using occlusions. *TPAMI*, 27(6):988–992, 2005.

[17] F. Perronnin, J. L. Dugelay, and K. Rose. Iterative decoding of two-dimensional hidden markov models. In *ICASSP*, volume 3, pages III–329–32, 2003.

[18] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. IEEE*, 77(2):257–286, 1989.

[19] X. Ren. Multi-scale improves boundary detection in natural images. In *ECCV*, pages 533–545, 2008.

[20] P. Sand and S. Teller. Particle video: Long-range motion estimation using point trajectories. In *CVPR*, volume 2, pages 2195–2202, 2006.

[21] A. Saxena, M. Sun, and A. Ng. Make3d: Learning 3d scene structure from a single still image. *TPAMI*, 31(5):824–840, May 2009.

[22] A. Saxena, M. Sun, and A. Y. Ng. Learning 3-d scene structure from a single still image. In *ICCV*, pages 1–8, 2007.

[23] P. Smith, T. Drummond, and R. Cipolla. Layered motion segmentation and depth ordering by tracking edges. *TPAMI*, 26(4):479–494, 2004.

[24] A. Stein. CMU Motion Dataset. Available at `http://www.cs.cmu.edu/~stein/occlusion_data`.

[25] A. Stein, D. Hoiem, and M. Hebert. Learning to find object boundaries using motion cues. In *ICCV*, pages 1–8, 2007.

[26] W. B. Thompson, K. M. Mutch, and V. A. Berzins. Dynamic occlusion analysis in optical flow fields. *TPAMI*, 7(4):374–383, 1985.

[27] J. Xiao, H. Cheng, H. S. Sawhney, C. Rao, and M. A. Isnardi. Bilateral filtering-based optical flow estimation with occlusion detection. In *ECCV*, pages 211–224, 2006.

[28] J. Xiao and M. Shah. Motion layer extraction in the presence of occlusion using graph cuts. *TPAMI*, 27(10):1644–1659, 2005.