An Online Learning Approach to Occlusion Boundary Detection

Natan Jacobson, *Student Member, IEEE*, Yoav Freund, and Truong Q. Nguyen, *Fellow, IEEE*

***Abstract*—We propose a novel online learning-based framework for occlusion boundary detection in video sequences. This approach does not require any prior training and instead "learns" occlusion boundaries by updating a set of weights for the online learning Hedge algorithm at each frame instance. Whereas previous training-based methods perform well only on data similar to the trained examples, the proposed method is well suited for any video sequence. We demonstrate the performance of the proposed detector both for the CMU data set, which includes hand-labeled occlusion boundaries, and for a novel video sequence. In addition to occlusion boundary detection, the proposed algorithm is capable of classifying occlusion boundaries by angle and by whether the occluding object is covering or uncovering the background.**

***Index Terms*—Edge detection, motion estimation, occlusion boundaries, occlusion boundary detection, online learning.**

I. Introduction

Digital video has become ubiquitous over the last two decades. Demand is growing at an exceptional rate for mobile video, high-quality digital television, and even 3-D broadcast and movies. This demand has necessitated significant research in the fields of compression, motion-compensated interpolation, and disparity estimation. Fundamental to each of these research topics is the concept of occlusion boundary detection, where an occlusion is the region between two overlapping objects with disparate motion. Detecting these events is crucial because many of the video processing assumptions fail at occlusion boundaries. For example, the smoothness constraint of disparity estimation does not hold true across an occlusion boundary. However, if the occlusion boundary is detected, then the smoothness term can be set to zero at these locations.

Occlusions are omnipresent and are crucial for visual understanding in a 3-D world [1]. By strict definition, an occlusion event occurs whenever one object is covered or uncovered by another object that is spatially closer to the observer. The observer may be a human, a still camera, or a video camera. For humans, occlusion is crucial in order to infer the relative depth of objects in the world. In fact, occlusion boundaries play a central role in human stereopsis [2], which determines how the human visual system perceives three dimensions from stereo vision.

Manuscript received July 16, 2010; revised January 30, 2011 and June 21, 2011; accepted June 30, 2011. Date of publication July 22, 2011; date of current version December 16, 2011. This work was supported in part by the Center for Wireless Communications, University of California at San Diego, La Jolla, and in part by the University of California Discovery Program. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. A. G. Bors.

The authors are with the Department of Electrical and Computer Engineering and the Department of Computer Science, University of California at San Diego, La Jolla, CA 92093 USA (e-mail: njacobso@ucsd.edu; yoav@cse.ucsd.edu; nguyent@ece.ucsd.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2011.2162420

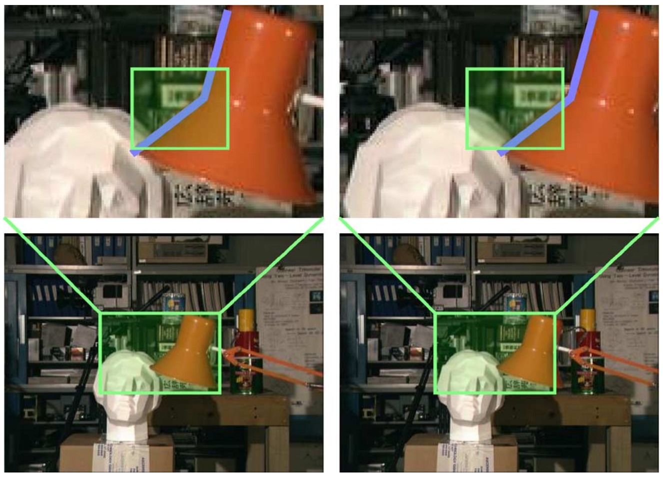

Fig. 1. Occlusions evident in Tsukuba stereo image pair. The lampshade occludes pixels in the left view, which are visible in the right view. Occlusion boundary in the selected window is highlighted in blue.

The distinction between *occlusion boundaries* and *appearance edges* is an important one. Here, an *appearance edge* refers to the typical output of an edge detection algorithm (e.g., Canny edge detector [3]) on luminance or color image data, whereas an *occlusion boundary* is explicitly created by objects covering or uncovering one another. An example of an occlusion boundary from the Tsukuba stereo image pair is demonstrated in Fig. 1. In this stereo pair, the lampshade in the foreground occludes the bookcase in the background. Whereas occlusion boundaries typically occur at appearance edges, the detection of an appearance edge is in no way sufficient to guarantee an occlusion boundary. For example, the significant edge on the nose of the ceramic bust in Fig. 1 is not an occlusion boundary; rather, it is due to the lighting of the scene. Likewise, edges in the motion field may be an indicator that an occlusion boundary is present; however, no guarantee is made. It is, however, evident that valuable information is provided by both the appearance cue and the motion field. In this paper, both cues will be considered. To calculate the motion cue, an off-the-shelf dense optical flow method is used [4]. This motion information is combined with pixel-domain information in an online learning framework.

The online learning framework is based on the idea of a panel of *experts*, where each expert is proficient at a distinct task. Each time a decision needs to be made, each expert is polled, and a loss is calculated based on how correct their prediction turned out to be. Over time, the best expert for each task will become evident. In this paper, the task of each expert is to detect a certain type of occlusion event, and the quality of each expert's detection is measured using a loss function in the pixel domain (similar to a standard distortion measure). At a specific location, an occlusion boundary is detected based on the best prediction over all experts. Contrary to all competing approaches, the proposed method does not require any training; rather, the classification of occlusion boundaries is based on the relative weighting of the experts, which occurs online.

This paper is organized as follows: Previous methods for the detection of occlusion boundaries are described in Section II. The underlying concepts of online learning are presented in Section III. Next, our algorithm is presented in Section IV along with a description of the feature set and experts used. Finally, results are presented in Section V, and this paper is concluded in Section VI.

II. Previous Work

The presence of occlusions in the image and video processing literature is astoundingly diverse. Research is conducted both to determine occlusion boundaries explicitly and to use this information implicitly in the pursuit of other results.

Implicit determination of occlusion boundaries has been cited in applications of object tracking [5], segmentation [6]–[8], and disparity estimation [9]–[11]. In all of these cases, occlusions are implicitly handled due to the information they provide about the scene. Two important concepts related to disparity estimation and pertinent to these works are the *uniqueness* and *ordering constraints* [11]. The uniqueness constraint states that features in the left and right images are in one-to-one correspondence. This concept is naturally extended to monocular video as objects generally do not appear or disappear during a single frame period. The ordering constraint states that the ordering of two objects in the left view is maintained in the right view. Because these concepts apply to objects in a stereo pair or video sequence, they must also apply to interactions between objects. For this reason, the two constraints must also apply for occlusion detection.

A number of methods have been applied to the problem of explicitly determining occlusion boundaries. One early approach determines occlusion boundaries for a set of known object types on a small pixel grid [12]. This method showed promising performance assuming no noise and a set of *a priori* known objects. Later, occlusion detection was combined with motion estimation to classify occluded areas based on a photometric mismatch between frames [13]. The drawback of this method is that any errors in the motion vector field (MVF) are likely to cause false detection of occlusion boundaries. Around the turn of the 21st century, learning-based research was conducted, which used two separate models to describe scene motion, i.e., a two-parameter translational model for regular motion and a six-parameter generative model for occlusion boundaries [14]. A graph-cuts approach has been also considered in which the uniqueness constraint is utilized to guarantee proper occlusion handling [15]. In a separate approach, occlusion events are determined

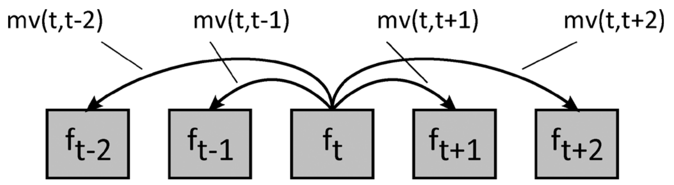

Fig. 2. Notation used for motion estimation. $mv(t,s)$ represents the motion field between frames f_t and f_s.

by the presence of T-junctions in a spatiotemporal volume created from a video sequence [16]. Geometric approaches have been also considered, analyzing the motion field alone to determine the presence of occlusions [17]. A more recent direction makes use of local appearance cues as well as motion information to detect occlusion boundaries [18], [19]. It is demonstrated that the combination of cues performs better than any cue independently used. In a separate project, researchers have applied a probabilistic framework for considering occlusion information across multiple frames [20]. The drawback of these learning-based methods is that, in order to obtain good detection performance, significant training data must be available. Additionally, this training data must be very similar in content to the test data in order for the trained features to contain sufficient discriminating power.

Additional motivation for this paper was obtained from the excellent "Particle Video" research paper [21]. In this work, motion estimation is posed as a particle-tracking problem using particle appearance and interparticle distortion. Here, a significant improvement is demonstrated over standard optical flow methods.

III. Online Learning

We pose occlusion boundary detection as a problem of prediction over a video sequence. This formulation is well suited for video as the correctness of each prediction at frame f_t will be revealed by subsequent frame f_{t+1}. The proposed framework is based on the Hedge online learning algorithm [22]. Rather than requiring a training stage, as most conventional learning algorithms do, the Hedge algorithm may be run against a video sequence without seeing any previous input. The Hedge algorithm is based on a panel of experts, where each expert is tuned to perform a simple decision at each frame instance. Decisions are made using the MVF and pixel-domain information. The MVF is computed using an optical flow technique [4]. The flow field between frames f_t and f_s will be denoted as $mv(t,s)$, as shown in Fig. 2. The correctness of each decision is then measured using a loss function. Experts are weighted based on their loss functions such that good performance (low loss) is rewarded with higher weighting. Finally, the Hedge algorithm makes its prediction based on the weighting of all experts. The flexibility of the algorithm allows us to choose the number of experts and to determine how each expert makes its decision. In this paper, the experts have been designed such that each one is tuned to detect a specific type of occlusion boundary. As will be presented, the loss function is measured using the sum of absolute differences (SAD) between predicted image patches and image patches in the current frame.

TABLE I
VARIABLE LIST

f_t	frame t of the input sequence
m	particle index
i	expert index
t	time-step
η	learning rate
$X(m)$	X and Y coordinates for particle m
$D(m)$	indicator vector for inactive particles
$C(m)$	occlusion classification for particle m
e_i	expert i
$l_{i,m}^t$	instantaneous loss
$L_{i,m}^t$	cumulative loss
$w_{i,m}^t$	attributed weight
$p_{i,m}^t$	normalized weight (probability)
c	size in pixels of ignored image boundary
θ	parameter to specify occlusion boundary angle
α	parameter to specify covering/uncovering
$\Gamma(\theta,\alpha)$	occlusion type for parameters θ, α
δ	patch size for calculation of expert loss
τ	multiplicative weight for no-occlusion expert

A. Pixels and Particles

In this paper, we distinguish between *pixels* and *particles* to be as explicit as possible. Whereas a pixel has the standard meaning, we use the term *particle* to refer to a picture element that will be tracked through the video sequence and will maintain certain properties. In the first time step of the video sequence, a large number of particles will be initialized such that one particle exists for each pixel in the image, minus the boundary. The online learning framework will then be applied to each particle, which will be classified based on the prediction and loss over the set of experts. Between time steps, each particle will be propagated based on the calculated optical flow. In this way, each particle maintains its history, and the classification will gain confidence over time. Special care is taken to ensure the particle tracking grid remains dense between time steps, as is discussed in Section IV.

B. Notation

The following notation is used for all further discussion involving Hedge and the proposed algorithm. Subscript i will be used to index the set of experts, whereas subscript m is used to index the set of particles. The time step is denoted by superscript t. For example, variable $w_{i,m}^t$ will refer to the weighting of expert i at time-step t for particle m. A full listing of variables is presented in Table I.

Further description and analysis of the experts is presented in the following section. A complete description of the proposed algorithm is then provided in Section IV.

C. Description of Experts

The goal of each expert is to detect a specific type of occlusion boundary. In the online learning framework, this is accomplished by each expert predicting content in the subsequent frame congruous with its specific occlusion boundary type. Parameters must be defined so that each occlusion boundary type is explicitly stated. Once this is accomplished, the calculation of *loss* can be defined, ultimately leading to the proposed algorithm for occlusion boundary detection.

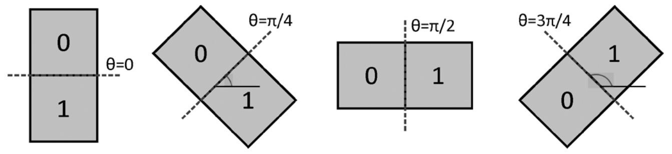

Fig. 3. Occlusion type parameter θ for the angle of the occlusion boundary. For each angle, the foreground object at the occlusion boundary can be either covering or uncovering the background. Labels 0 and 1 are used to make explicit the two sides of the occlusion boundary.

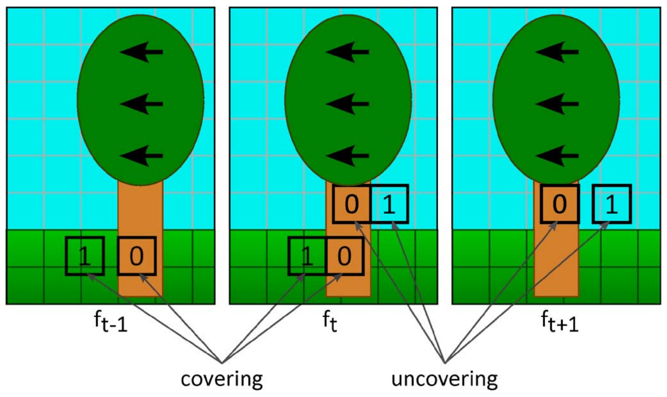

Fig. 4. Covering ($\alpha = 1$) and uncovering ($\alpha = 0$) occlusion boundaries for frame f_t. Both boundaries are vertical ($\theta = \pi/2$) with the tree in the foreground. The expert for the covering occlusion boundary uses frame f_{t-1} as reference with motion field $mv(t, t-1)$, whereas the expert for the uncovering case uses frame f_{t+1} with motion field $mv(t, t+1)$.

Each expert is associated with two parameters that determine its prediction. These parameters are the angle of the occlusion boundary and whether the foreground object is covering or uncovering the background. The first parameter is illustrated in Fig. 3, where the angle of occlusion boundary is in set $\theta \in \{0, (\pi/4), (\pi/2), (3\pi/4)\}$. Note that additional angles may be included if the user desires a larger set of experts. The next parameter distinguishes between a *covering* and an *uncovering* occlusion event, as illustrated in Fig. 4. Parameter $\alpha = 1$ denotes the foreground object covering the background, whereas $\alpha = 0$ denotes the foreground object uncovering the background. To make these concepts more concrete, a series of *occlusion types* are defined, where each is fixed to a specific set of parameters. Denote an occlusion type as $\Gamma(\theta, \alpha)$, where parameters θ and α are as described here. For example, the two experts in Fig. 4 detect a vertical covering occlusion type $(\Gamma((\pi/2), 1))$ and a vertical uncovering occlusion type $(\Gamma((\pi/2), 0))$, respectively. Patch labels 0 and 1 are included to explicitly distinguish between the two sides of the occlusion boundary.

It is clear from this formulation that the introduction of each additional angle will create two unique occlusion types. An analysis has been conducted, which demonstrates a decreasing marginal benefit to performance as the set of angles is increased. This is discussed further in Section V. For the time being, it should be mentioned that for results presented in this paper, the four occlusion boundary angles in Fig. 3 are considered. Therefore, a total of eight occlusion types are included in the framework, along with a default case that assumes no occlusion event. Because each of these types is predicted by a unique

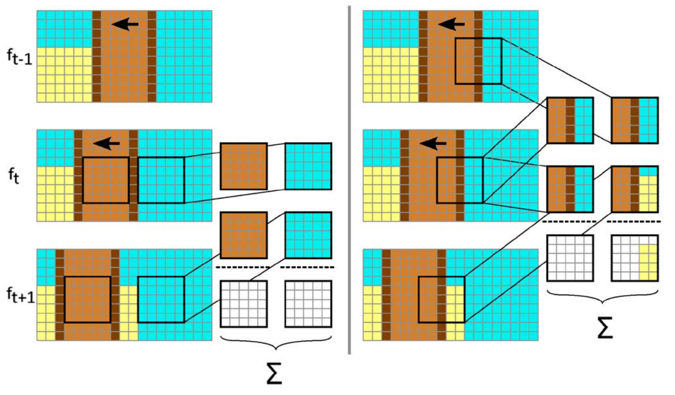

Fig. 5. Example of instantaneous loss calculation for two experts. The tree is moving to the left against a static background. On the left side of the figure, occlusion type $\Gamma((\pi/2), 0)$ is shown, and the instantaneous loss is computed for patches on either side of the occlusion boundary. On the right side of the figure, the null occlusion type is shown, and instantaneous loss is calculated for a centered patch. For both occlusion types, instantaneous loss is calculated as the SAD between patches in frame f_t and predicted patches in frames $f_{t\pm1}$. In the case of the null occlusion type, prediction error occurs for the background, indicating that this expert will have a larger instantaneous loss than the expert on the left.

expert, the current formulation of the problem includes a total of $N = 9$ experts.

D. Instantaneous Loss Calculation

The instantaneous loss of each expert is measured using the SAD error metric, as is demonstrated in Fig. 5. The SAD is computed between predicted and revealed image patches either adjacent to or centered on the particle. For all occlusion types that predict an occlusion boundary, the patches are adjacent to the boundary, with one patch on either side. For the null case that predicts no occlusion boundary, the patch will be centered on the particle. If, for example, the image patch is of size $\delta \times \delta$ and the expert is of type $\Gamma((\pi/2), 0)$, then the centers of the two patches will be located at $(x(m) \pm \lceil(\delta/2)\rceil, y(m))$, where $(x(m), y(m))$ is the location of particle m. Denote $x_0 = x(m) - \lceil(\delta/2)\rceil$ and $x_1 = x(m) + \lceil(\delta/2)\rceil$, and consider the motion vector pair u_0, v_0 to be the x and y components of the motion field at x_0 and u_1, v_1 to be the components at x_1. Finally, it should be mentioned that for an uncovering occlusion boundary event, motion field $mv(t, t+1)$ will be used with predicted patches in frame f_{t+1}. However, for a covering occlusion type, motion field $mv(t, t-1)$ will be used instead with frame f_{t-1}. For the null case, both motion fields will be used.

For occlusion type $\Gamma((\pi/2), 0)$, the SAD for the two patches, i.e., l_0 and l_1, are calculated as

$$l_0 = \sum_{a=-\lfloor\frac{\delta}{2}\rfloor}^{\lfloor\frac{\delta}{2}\rfloor} \sum_{b=-\lfloor\frac{\delta}{2}\rfloor}^{\lfloor\frac{\delta}{2}\rfloor} |f_t(x_0 + a, y(m) + b) - f_{t+1}(x_0 + a + u_0, y(m) + b + v_0)|$$
$$l_1 = \sum_{a=-\lfloor\frac{\delta}{2}\rfloor}^{\lfloor\frac{\delta}{2}\rfloor} \sum_{b=-\lfloor\frac{\delta}{2}\rfloor}^{\lfloor\frac{\delta}{2}\rfloor} |f_t(x_1 + a, y(m) + b) - f_{t+1}(x_1 + a + u_1, y(m) + b + v_1)|. \quad (1)$$

The total instantaneous loss for the expert is the sum of the two patches, i.e., $l_{i,m}^t = l_0 + l_1$. Instantaneous loss for the other experts is calculated likewise; however, the location and orientation of the patches will depend on angle θ of the occlusion boundary and user-defined parameter δ.

For the null expert, which predicts the lack of an occlusion boundary, instantaneous loss is calculated for a patch centered on the particle with reference frames f_{t-1} and f_{t+1}. Here, the SAD for the two patches are calculated as

$$l_0 = \sum_{a=-\lfloor\frac{\delta}{2}\rfloor}^{\lfloor\frac{\delta}{2}\rfloor} \sum_{b=-\lfloor\frac{\delta}{2}\rfloor}^{\lfloor\frac{\delta}{2}\rfloor} |f_t(x(m) + a, y(m) + b) - f_{t+1}(x(m)+a+u_0, y(m)+b+v_0)|$$
$$l_1 = \sum_{a=-\lfloor\frac{\delta}{2}\rfloor}^{\lfloor\frac{\delta}{2}\rfloor} \sum_{b=-\lfloor\frac{\delta}{2}\rfloor}^{\lfloor\frac{\delta}{2}\rfloor} |f_t(x(m) + a, y(m) + b) - f_{t-1}(x(m)+a+u_1, y(m)+b+v_1)| \quad (2)$$

and $l_{i,m}^t = \tau(l_0 + l_1)$. Parameter τ controls the sensitivity of the detector to occlusion boundaries. We vary this parameter in order to generate the precision–recall plots shown in Section V. For the proposed work, we fix parameter $\delta = 7$ for all experiments. This selection was made to capture sufficient local pixel-domain information without requiring a huge number of pixel calculations for each expert, which would dramatically increase the computational complexity.

IV. Proposed Algorithm

A flowchart in Fig. 6 depicts the three main portions of the proposed algorithm. In the first portion, the particle tracking grid is initialized (if it is the first time step), and optical flow fields are computed. Next, the Hedge algorithm is used to compute the probability distribution over the experts for each particle. In the third portion, each particle is classified, and the particle tracking grid is propagated.

In the first stage of the proposed algorithm, motion vector information is obtained for the video sequence using an optical flow technique [4]. As is standard in optical flow methods, the brightness constancy and gradient constancy assumptions are combined to obtain a motion field between two subsequent frames. A third term is included in [4], which enforces spatiotemporal smoothness while preserving spatial discontinuities. The resulting dense MVF $mv(t, t \pm 1)$ is defined for each pixel in the sequence.

The Hedge algorithm [22], which is presented in Algorithm 1, is responsible for detecting occlusion boundaries using a set of experts, each of which is tuned to detect a separate occlusion type. The input to the algorithm is the pixel-domain information f_t, $f_{t\pm1}$, and MVFs $mv(t, t \pm 1)$. The output is a probability distribution over the set of experts for each particle. This results in a classification of each particle into an occlusion type.

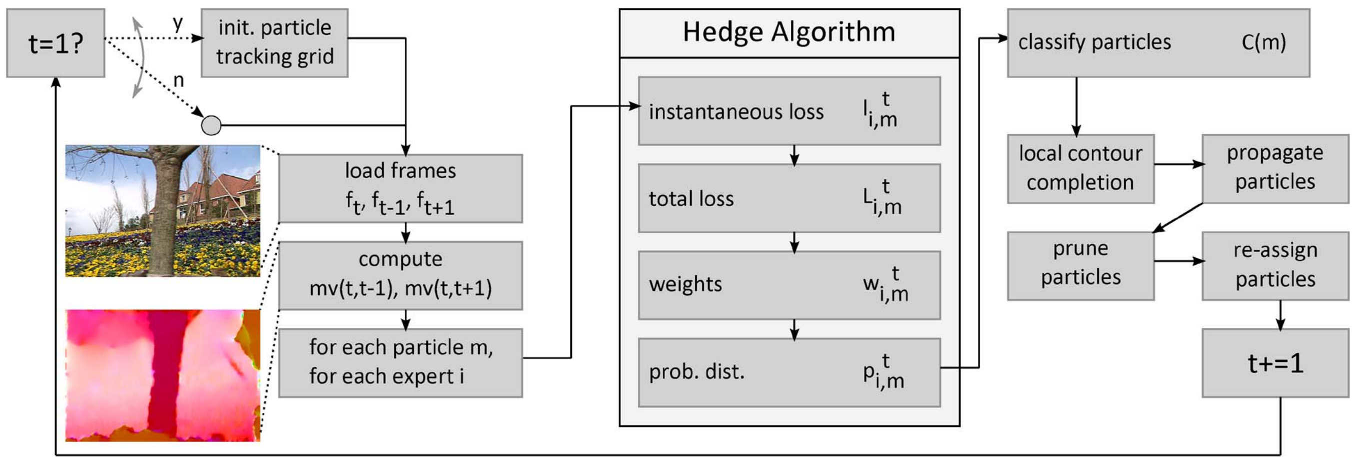

Fig. 6. Flowchart for the proposed occlusion boundary detection algorithm.

Algorithm 1 Hedge

```
initialize expert weights to be uniform: $w_i^1 = 1/N, \forall i$
for $t = 1, \dots, T$ do
  for $m = 1, \dots, M$ do
    for $i = 1, \dots, N$ do
      expert $e_i$ makes prediction
      instantaneous loss calculated: $l^t_{i,m}$
      if exponentially discounted loss then
        total loss: $L^t_{i,m} = (1-\alpha)\sum_{s=1}^{t-1} l^s_{i,m} + l^t_{i,m}$
      else
        total loss: $L^t_{i,m} = \sum_{s=1}^{t} l^s_{i,m}$
      end if
      weights updated: $w^t_{i,m} = w^1_{i,m} \exp^{-\eta L^t_{i,m}}$
      form prob. dist. over experts: $p^t_{i,m} = w^t_{i,m} / \sum_{j=1}^{N} w^t_{j,m}$
    end for
  end for
end for
```

The algorithm is initialized with a dense grid of particles in the first frame of the video sequence. The particle grid is the same resolution as the video sequence with a border of $c = 10$ pixels removed to avoid edge effects. Particles are indexed by $m \in \{1, \dots, M\}$, where $M = (W-2c)\times(H-2c)$ and W and H are the width and height in pixels, respectively, for the video sequence. Particle locations are stored in vector $\mathbf{X} \in \mathbb{Z}^{M\times 2}$, where each row of $\mathbf{X}$, denoted by $X(m)$ contains the x and y coordinates of particle m. A set of experts $e_1, \dots, e_N$ predicts local patch information in f_{t-1} or f_{t+1} based on the occlusion type. The instantaneous loss of each expert is calculated using the SAD error metric. Next, cumulative loss is calculated using an exponentially discounted loss function and stored in $\mathbf{L} \in \mathbb{R}^{M\times N}$. The exponentially discounted loss function is

$$L^t_{i,m} = (1-\alpha)L^{t-1}_{i,m} + l^t_{i,m} \tag{3}$$

where $L^t_{i,m}$ is the cumulative loss of expert i at time-step t for particle m. Each expert is reweighted at each time step based on the cumulative loss and tunable *learning rate* η, i.e.,

$$w^t_{i,m} = \frac{1}{N}\exp^{-\eta L^t_{i,m}}. \tag{4}$$

Weights $w^1_{i,m}$ are initialized to be uniform. That is, $w^1_{i,m} = 1/N, \forall i$. The learning rate is set to $\eta = \sqrt{(2\ln N)/T}$, as proposed in [22]. This learning rate is selected as it guarantees an upper bound on the cumulative loss of the Hedge algorithm as

$$L_{\text{hedge}}(\eta) \leq \min_i L_i + \sqrt{2T\ln N} + \ln N \tag{5}$$

where N is the total number of experts and T is the number of time steps. This means that Hedge will always achieve a cumulative loss close to that of the best expert at time t. Next, the weights are normalized to produce a probability distribution over the experts. Intuitively, this is a measure of how well each expert is performing, i.e.,

$$p^t_{i,m} = \frac{w^t_{i,m}}{\sum_{j=1}^{N} w^t_{j,m}}. \tag{6}$$

At each time step and for each particle, the weights attributed to the N experts are compared. Classification is performed based on the expert with the largest weighting at time-step t. The classification function is

$$C(m) = \arg\max_i \left(p^t_{i,m}\right). \tag{7}$$

Therefore, if expert e_i is tuned to detect occlusion type $\Gamma((\pi/4), 1)$, then particle m will be classified as a covering occlusion boundary of angle $\theta = \pi/4$ if $p^t_{i,m}$ has the largest value over the vector p.

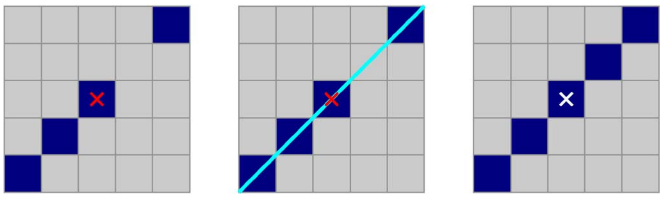

Fig. 7. Contour completion at angle $\theta = \pi/4$ using four neighbors and threshold of 50%. Center pixel marked with red "X."

A. Local Contour Completion

Subsequent to the Hedge update at each time step, a contour completion stage is conducted. This is similar to the binary morphological operation of dilation. We have determined that this operation significantly improves detection performance while adding little complexity. Performance improvement is observed because the Hedge method may result in small gaps in the detected occlusion boundary due to factors such as noisy pixel data and a slow learning rate. For each particle that has registered an occlusion event, a short segment of potential particles are aligned based on the classified occlusion type of the particle. An example is shown in Fig. 7 for an occlusion type of angle $\theta = \pi/4$. If more than half of these particles also observe an occlusion event tuned to the same angle, then the entire line segment is identically classified.

B. Particle Propagation

Between two adjacent time steps, the particles are propagated via the MVF. If particle m is located at position $\mathbf{X}(m) = (x(m), y(m))$ in frame f_t, then it will be propagated to

$$\mathbf{X}(m) = (x(m) + u, y(m) + v) \quad \text{in} \quad f_{t+1} \tag{8}$$

where u and v are the components of the motion vector, located in the MVF $\mathbf{mv}\ (t, t+1)$ at position $(x(m), y(m))$.

C. Particle Pruning and Reassignment

The number of particles is kept constant by merging particles that belong to the same pixel and introducing new particles where appropriate. In addition, particles that are propagated to within $c = 10$ pixels of the image boundary will be pruned as it is unnecessary to track particles that are exiting the frame. If multiple particles are propagated to the same pixel, all but one of them will be pruned, and the average of the cumulative losses will be assigned to the remaining particle. After particle pruning is complete, new particles will be assigned such that the particle tracking grid remains dense. For time-step $t+1$, the particle tracking grid is examined to determine if there are any pixel locations such that no particle exists. If one is found, a new particle is added to the tracking grid with uniform weights over the set of experts, i.e., $w_{i,m}^{t+1} = 1/N, \forall i$

V. Simulation Results

The proposed occlusion boundary detection algorithm has been tested against sequences obtained from the *CMU Video Data set for Occlusion/Object Boundary Detection* [18]. This data set is comprised of 30 short video sequences, each of which contains labeled ground truth occlusion boundaries. In addition, a synthetic sequence has been created, which is significantly different from the sequences available in the CMU data set. This will test the robustness of learning-based methods for which cross-validation is not possible. Additional results, including detection results for full video sequences, are included on the author's website.[1]

A. CMU Occlusion Data Set

Each sequence in the CMU data set consists of between 5 and 30 frames and includes hand-labeled ground truth occlusion boundaries for the center frame. An objective comparison is performed between each occlusion boundary detector and the ground truth data from the data set using precision–recall. An example of this is demonstrated in Fig. 8, where four frames from the data set are shown along with the ground truth occlusion boundaries and the result of the proposed online detection algorithm. Precision and recall scores are calculated as follows: Let $\{x_{gt}\}$ denote the set of occlusion boundary pixels in the ground truth data and $\{x_d\}$ denote the set of occlusion boundary pixels detected by the proposed algorithm. Then, precision, recall, and F-score values are calculated as

$$p = \frac{|x_{gt}| \wedge |x_d|}{|x_d|}, \quad r = \frac{|x_{gt}| \wedge |x_d|}{|x_{gt}|}, \quad F = \frac{2pr}{p+r} \tag{9}$$

where F-score F can be equivalently calculated as the ratio between the common ground truth and detected occlusion boundaries and the mean between the two, that is,

$$F = \frac{2(|x_{gt}| \wedge |x_d|)}{|x_{gt}| + |x_d|}.$$

In general, results are presented using this single metric. The proposed algorithm is compared with four competing methods from the occlusion literature. The first approach that we compare with is based on the photometric difference between adjacent frames [13] and is denoted as the *photometric* approach. Here, the mismatch in intensity between two adjacent frames is measured using the motion field. For two adjacent frames f_t and f_{t+1} and the forward and backward motion fields $d_f = mv(t, t+1)$ and $d_b = mv(t+1, t)$, the motion-compensated prediction errors are given by

$$\begin{aligned} \epsilon_f(\mathbf{x}) &= f_t(\mathbf{x}) - f_{t+1}(\mathbf{x} + d_f(\mathbf{x})) \\ \epsilon_b(\mathbf{x}) &= f_{t+1}(\mathbf{x}) - f_t(\mathbf{x} - d_b(\mathbf{x})). \end{aligned} \tag{10}$$

The absolute value of these two errors are compared with threshold Θ to determine the presence of occlusion boundaries. The precision–recall curve in Fig. 9 is produced by sweeping the threshold in the range $\Theta \in [0, 255]$. In the second *geometric* approach, uncovered regions are detected by locating areas in the reference frame for which no motion candidates exist in the current frame [17]. If we denote a sampling lattice in frame f_t as Λ and we denote $S = \{\mathbf{y} : \mathbf{y} = \mathbf{x} + d_f(\mathbf{x}), \mathbf{x} \in \Lambda\}$ as the set of spatial positions in f_{t+1} achieved by motion compensating the

[1]http://videoprocessing.ucsd.edu/~NatanHaim

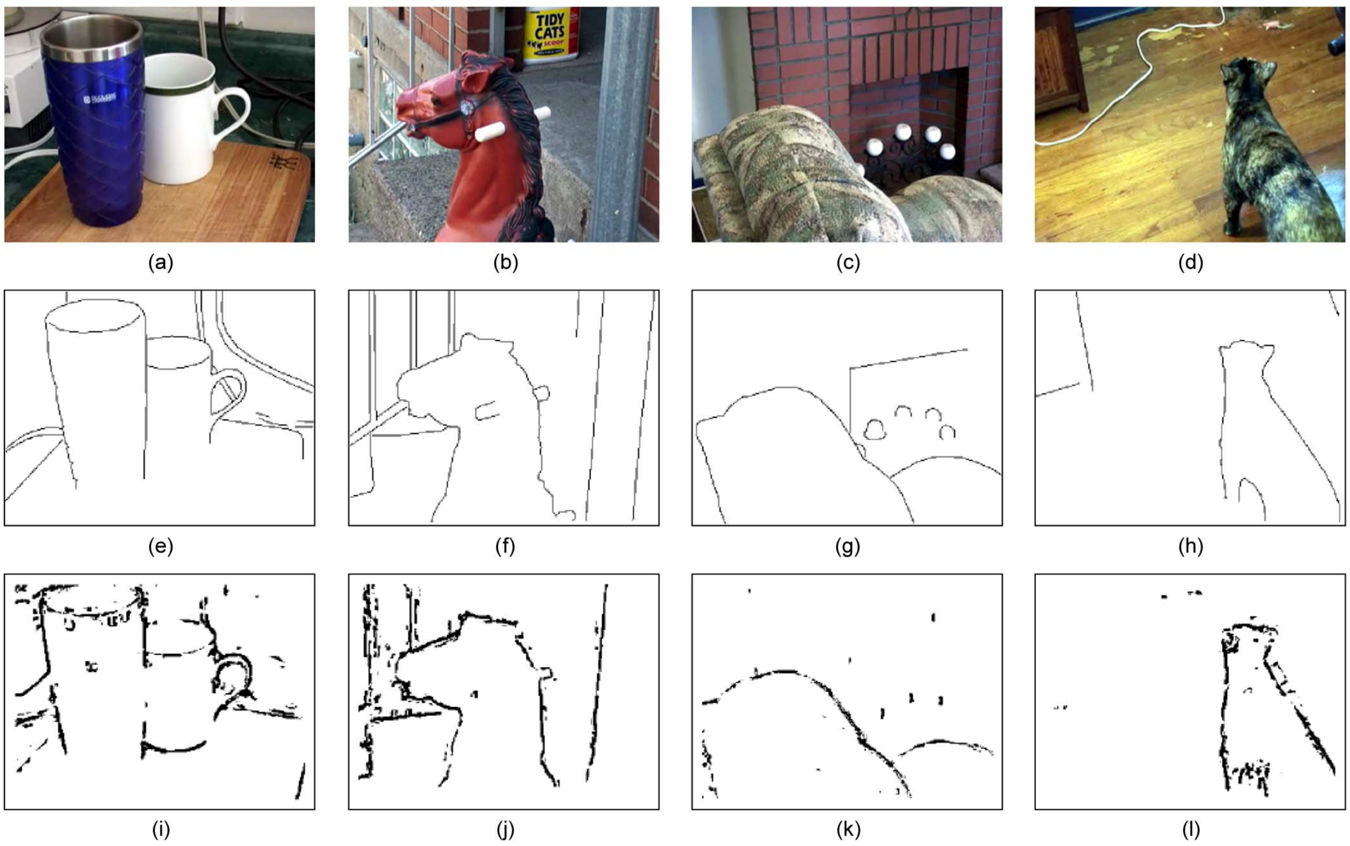

Fig. 8. Comparison of the proposed algorithm with ground truth occlusion boundaries for four sequences of the CMU database. For each sequence, the frame is shown in the top row, ground truth in the middle row, and the result of the proposed occlusion boundary detector in the bottom row. (Columns from left to right) *mugs2* ($F = 49.10\%$), *rocking horse* ($F = 54.50\%$), *couch corner* ($F = 54.31\%$), and *zoe1* ($F = 50.29\%$).

sampling lattice of f_t, then an indicator function may be defined as

$$\xi_i(\mathbf{x}) = \begin{cases} 1, & \|\mathbf{x} - \mathbf{z_i}\| \leq r \\ 0, & \text{otherwise} \end{cases} \quad \mathbf{x} \in \Lambda, \mathbf{z_i} \in S \tag{11}$$

where $\mathbf{x}$ and $\mathbf{z_i}$ are in f_{t+1} and radius r is defined by the user (in this paper, as in [17], $r = 2$). Finally, $M(\mathbf{x}) = \sum_{i=1}^{|S|} \xi_i(\mathbf{x})$ measures the density of projections. This value is thresholded to generate the precision–recall curve. In order to detect covered regions, the reverse process is computed from $f_{t+1} \rightarrow f_t$. The third approach considered is the training-based method of [19] in which numerous appearance and motion features are considered in the training of a binary occlusion boundary classifier. Finally, the method of [20] is explored in which a discriminative learning step is used to learn the relation between low-level features and labeled occlusion boundaries for the CMU data set.

Results for these methods as well as the proposed occlusion boundary detector are displayed in Fig. 9. Note that the proposed detector outperforms the other two nontraining-based methods by roughly 20%. In fact, our performance approaches that of [19], achieving within 5% of a method that was trained using the CMU data set. The further work of [20] is able to increase performance further but is not well suited to novel video sequences, as will be demonstrated in the next section. For the two training-based methods, learning was performed using ground truth on half of the CMU data set while testing on the other half. This was repeated such that all sequences were tested.

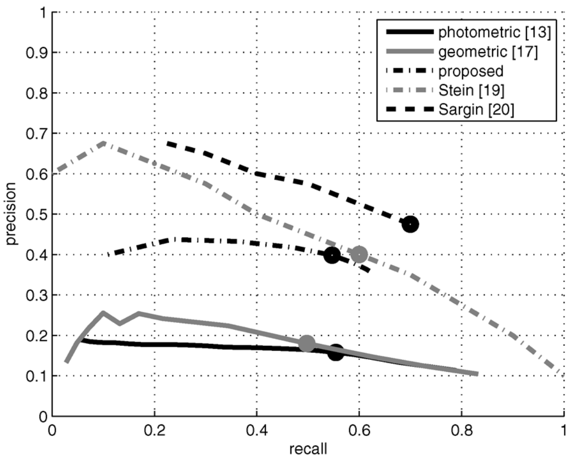

Fig. 9. Comparison of occlusion boundary detection performance using precision–recall. Point of maximum F-score marked for each curve. Best performance of each method. (Photometric) $F_{\max} = 22.637\%$. (Geometric) $F_{\max} = 24.636\%$. (Proposed) $F_{\max} = 43.857\%$. (Stein [19]) $F_{\max} = 48.000\%$. (Sargin *et al.* [20]) $F_{\max} = 56.596\%$.

B. Synthetic Sequence

To further assess the performance of the proposed method, we have constructed a 60-frame synthetic sequence with a dominant occlusion boundary. Textures for this sequence were obtained

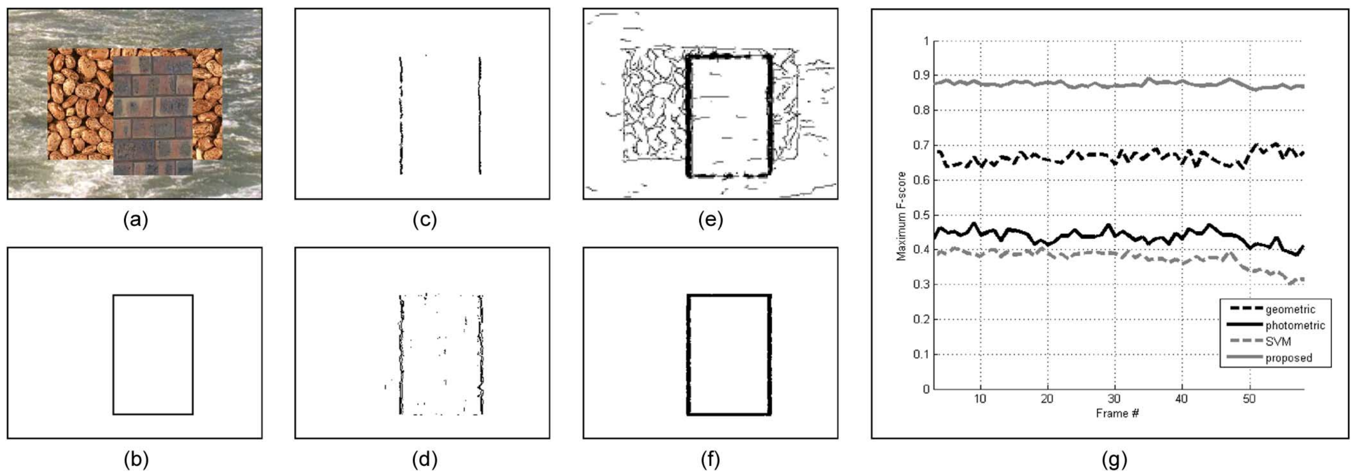

Fig. 10. Results for synthetic sequence: (a) frame 30; (b) ground truth; (c) geometric method [17]; (d) photometric method [13]; (e) trained SVM [20]; (f) proposed; and (g) comparison of methods, i.e., $F_{\max}$ versus frame index.

from the publicly available Vision Texture homepage [23]. The proposed method for occlusion boundary detection is compared with the methods of [13], [17], and [20] in Fig. 10. Here, objective results are provided in Fig. 10(g) by computing the maximum F-score for each method at each frame of the sequence. It is clear from this simulation that the best performance is obtained by the proposed detector. It is also clear that while the training-based method may perform well when cross-validation is possible, it may not perform as well in general. In particular, a large number of false positive occlusion boundaries are detected due to edges in the texture maps.

Results for the two competing nontraining-based methods are achieved using a threshold of $\Theta = 80$ for the photometric method [13] and $\Theta = 11$ for the geometric method [17]. These parameters were selected because they produced the maximum F-score for each frame. We implemented the method of [20] using a support vector classifier (LibSVM [24]), which is trained on the CMU database. The feature set for training is identical to that used in [20]. It is worth noting that the same optical flow field is used for both the proposed method and the feature set for the SVM classifier. In this way, it cannot be stated that one method of computing optical flow is superior to another, thus providing for an unfair comparison.

The SVM classifier is trained using the following procedure: First, the optical flow field is computed between each adjacent pair of frames using the method due to [4]. Denote the motion field in the x and y directions as u and v, respectively. The gradients of the optical flow field are then u_{dx}, u_{dy}, v_{dx}, and v_{dy}. The feature set is comprised of the following five features:

1) magnitude of the optical flow gradient, i.e.,

$$\sqrt{u_{dx}^2 + u_{dy}^2 + v_{dx}^2 + v_{dy}^2} \tag{12}$$

2) motion estimation error, i.e.,

$$|F_t - F_{t+1}(x+u, y+v)| \tag{13}$$

3) divergence of optical flow, i.e.,

$$|u_{dx} + v_{dy}| \tag{14}$$

4) minimum eigenvalue of the spatiotemporal structure tensor [see (15)], i.e.,

$$T = \begin{bmatrix} F_{dx}^2 & F_{dx}F_{dy} & F_{dx}F_{dt} \\ F_{dy}F_{dx} & F_{dy}^2 & F_{dy}F_{dt} \\ F_{dt}F_{dx} & F_{dt}F_{dy} & F_{dt}^2 \end{bmatrix} \tag{15}$$

where F_{dt} is computed using the following formula: $uF_{dx} + vF_{dy} + F_{dt} = 0$;

5) edge intensity map using the pB edge detector [25].

A grid search is performed on the training set in order to obtain good training performance. Since the number of negative example far exceeds the number of positive examples, a random resampling procedure is employed for the negative examples to produce the training set. Each feature is normalized to be in the range [0, 1]. Training time is on the order of 2 h for all 30 sequences in the CMU data set using an Intel Core i7 965 processor with 12 GB of random access memory.

C. Performance And Occlusion Types

As the runtime of our proposed algorithm is linear with respect to the number of experts, it is important to determine what effect the expert count has on performance. To test this, the F-score of the *chair* sequence from the CMU data set was examined with a variable size of the expert set. Results are displayed in Fig. 11. As N is increased, additional occlusion types are enabled in the order $\{\Gamma(\theta, 0), \Gamma(\theta, 1)\}$ for each angle θ. Angles are added in the following order: $\theta = \{(\pi/2), 0, (\pi/4), (3\pi/4), (\pi/8), (5\pi/8)\}$. It is observed that the marginal performance increase above $N = 9$ is diminishing, thus, the selection of $N = 9$ for all the experiments conducted in the proposed work.

D. Occlusion Boundary Classification

In addition to occlusion boundary detection, the proposed algorithm can provide classification results based on the occlusion types specified. An example of this is demonstrated in Fig. 12 for the *rocking horse* sequence of the CMU data set. Further

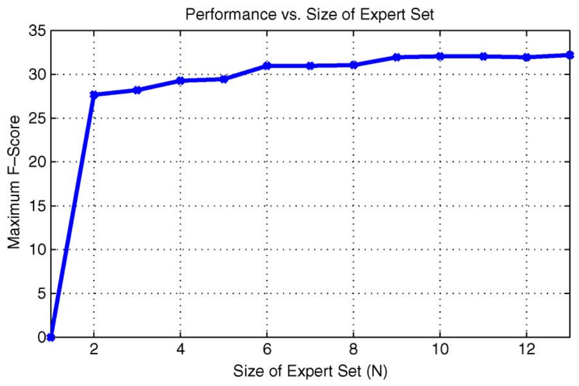

Fig. 11. Proposed algorithm performance as a function of expert set size for the *chair* sequence. We have selected a set of $N = 9$ experts, as selecting further occlusion types yields diminishing performance returns.

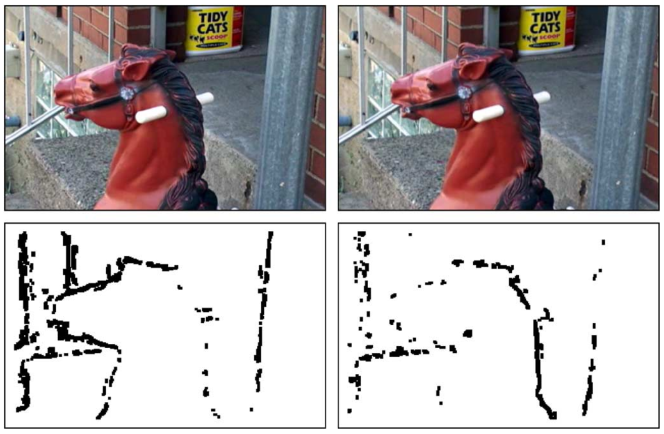

Fig. 12. Occlusion boundary classification into two classes. (Left) Uncovering. (Right) Covering.

classification can be performed based on the angle of the occlusion type, if desired. This technique is well suited for any application that will treat covering and uncovering occlusion types separately.

VI. Conclusion

In this paper, an efficient online learning approach to occlusion boundary detection has been presented. This method boasts a runtime linear with respect to the number of tracked particles and number of experts. In addition, the proposed algorithm does not require training, making it much simpler to use than competing methods and much more suitable for novel video sequences when training data is unavailable. Despite the lack of training, the algorithm has demonstrated excellent performance both on the CMU occlusion data set and on a synthetic video sequence. We outperform previous approaches that do not require a training stage while approaching the performance of a fully trained classifier. The efficiency of the proposed algorithm makes it well suited as an off-the-shelf implementation, which can be used on its own, or as a preprocessing step for other video processing tasks such as disparity estimation, motion vector refinement, and frame rate up-conversion.

Future work for this research may include an improved algorithm that detects object scale in addition to occlusion boundaries. In addition, NormalHedge [26] may be used to remove learning rate parameter η and to allow a larger set of experts.

Acknowledgment

The authors would like to thank Dr. Sargin for the assistance in providing comparison data to our method using a trained support vector machine classifier based on his thesis work. Without his patience and guidance, the comparison with training-based methods would not have been possible.

References

[1] P.-S. Toh and A. Forrest, "Occlusion detection in early vision," in *Proc. Int. Conf. Comput. Vis.*, Dec. 1990, pp. 126–132.
[2] K. Nakayama and S. Shimojo, "Da Vinci stereopsis: Depth and subjective occluding contours from unpaired image points," *Vis. Res.*, vol. 30, no. 11, pp. 1811–1825, 1990.
[3] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.
[4] M. J. Black and P. Anandan, "The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields," *Comput. Vis. Image Underst.*, vol. 63, no. 1, pp. 75–104, Jan. 1996.
[5] Y. Fu, A. Erdem, and A. Tekalp, "Tracking visible boundary of objects using occlusion adaptive motion snake," *IEEE Trans. Image Process.*, vol. 9, no. 12, pp. 2051–2060, Dec. 2000.
[6] P. Aguiar and J. Moura, "Figure-ground segmentation from occlusion," *IEEE Trans. Image Process.*, vol. 14, no. 8, pp. 1109–1124, Aug. 2005.
[7] A. Ogale, C. Fermuller, and Y. Aloimonos, "Motion segmentation using occlusions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, pp. 988–992, Jun. 2005.
[8] D. Feldman and D. Weinshall, "Motion segmentation and depth ordering using an occlusion detector," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 7, pp. 1171–1185, Jul. 2008.
[9] C. Zitnick and T. Kanade, "A cooperative algorithm for stereo matching and occlusion detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 7, pp. 675–684, Jul. 2000.
[10] O. Williams, M. Isard, and J. MacCormick, "Estimating disparity and occlusions in stereo video sequences," in *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, Jun. 2005, vol. 2, pp. 250–257.
[11] Z.-J. Zhu, Y.-E. Wang, G.-Y. Jiang, and Q.-W. Zhang, "Novel scheme for disparity estimation and occlusion detection based on variable line-segment primitive," in *Proc. Int. Conf. Sig. Process.*, Nov. 2006, vol. 2, pp. 1–4.
[12] J. Ullmann, "Analysis of 2-D occlusion by subtracting out," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 14, no. 4, pp. 485–489, Apr. 1992.
[13] R. Depommier and E. Dubois, "Motion estimation with detection of occlusion areas," in *Proc. IEEE Conf. Acoust., Speech, Signal Process.*, Mar. 1992, vol. 3, pp. 269–272.
[14] M. J. Black and D. J. Fleet, "Probabilistic detection and tracking of motion boundaries," *Int. J. Comput. Vis.*, vol. 38, no. 3, pp. 231–245, Jul./Aug. 2000.
[15] V. Kolmogorov and R. Zabih, "Computing visual correspondence with occlusions using graph cuts," in *Proc. Int. Conf. Comput. Vis.*, 2001, pp. 508–515.
[16] N. Apostoloff and A. Fitzgibbon, "Learning spatiotemporal T-junctions for occlusion detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2005, vol. 2, pp. 553–559.
[17] S. Ince and J. Konrad, "Geometry-based estimation of occlusions from video frame pairs," in *Proc. IEEE Int. Conf. Accoust. Speech Signal Process.*, 2005, vol. 2, pp. ii/933–ii/936.
[18] A. Stein and M. Hebert, "Combining local appearance and motion cues for occlusion boundary detection," in *Proc. Brit. Mach. Vision Conf.*, Sep. 2007, pp. 1–10.
[19] A. Stein, "Occlusion boundaries: Low-level detection to high-level reasoning," Ph.D. dissertation, Robotics Inst., Carnegie Mellon Univ., Pittsburgh, PA, May 2008.
[20] M. Sargin, L. Bertelli, B. Manjunath, and K. Rose, "Probabilistic occlusion boundary detection on spatio-temporal lattices," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2009, pp. 560–567.

[21] P. Sand and S. Teller, “Particle video: Long-range motion estimation using point trajectories,” in *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, 2006, vol. 2, pp. 2195–2202.

[22] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth, “Using and combining predictors that specialize,” in *Proc. 29th ACM Symp. Theory Comput.*, New York, 1997, pp. 334–343.

[23] Vision texture homepage [Online]. Available: http://vismod.media.mit.edu/vismod/imagery/VisionTexture/vistex.html

[24] C. C. Chang and C. J. Lin, LIBSVM: A library for support vector machines 2001.

[25] D. Martin, C. Fowlkes, and J. Malik, “Learning to detect natural image boundaries using local brightness, color, and texture cues,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 5, pp. 530–549, May 2004.

[26] K. Chaudhuri, Y. Freund, and D. Hsu, “A parameter-free hedging algorithm,” Compt. Research Repository pp. 1–9, 2009, abs/0903.2851.

Natan Jacobson (S’10) was born in Tucson, AZ, in 1984. He received the B.S. degree in electrical engineering with a minor in mathematics from Arizona State University, Tempe, in 2006 and the M.S. degree in electrical and computer engineering in 2008 from the University of California at San Diego, La Jolla, where he is currently working toward the Ph.D. degree in image/video processing.

He has worked with NXP Semiconductor and Sharp Laboratories in the field of video processing. His current research interests include signal and image processing, human vision, and machine learning.

Yoav Freund is a Professor of computer science and engineering with the University of California at San Diego, La Jolla. He is an internationally known Researcher in the field of machine learning, which is a field that bridges computer science and statistics. His work is in the area of machine learning and computational statistics and their applications.

Dr. Freund is best known for his joint work with Dr. Schapire on the Adaboost algorithm. For this work, they were awarded the 2003 Godel prize in Theoretical Computer Science and the Kanellakis Prize in 2004.

Truong Q. Nguyen (F’05) is currently a Professor with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla. He is the coauthor (with Prof. Strang) of a popular textbook “Wavelets and Filter Banks” (Wellesley-Cambridge Press, 1997) and the author of several MATLAB-based toolboxes on image compression, electrocardiogram compression, and filter bank design. He has over 300 publications. His research interests are video processing algorithms and their efficient implementation.

Prof. Nguyen is currently the Series Editor (Digital Signal Processing) for Academic Press. He served as an Associate Editor for the IEEE TRANSACTIONS ON SIGNAL PROCESSING from 1994 to 1996, the Signal Processing Letters from 2001 to 2003, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS from 1996 to 1997 and from 2001 to 2004, and the IEEE TRANSACTIONS ON IMAGE PROCESSING from 2004 to 2005. He was the recipient of the IEEE TRANSACTIONS ON SIGNAL PROCESSING Paper Award (Image and Multidimensional Processing area) for the paper he cowrote with Prof. Vaidyanathan on linear-phase perfect-reconstruction filter banks (1992). He was the recipient of the National Science Foundation Career Award in 1995.