Consistent Segmentation for Optical Flow Estimation

C. Lawrence Zitnick Nebojsa Jojic Sing Bing Kang
Microsoft Research
Redmond, WA
{*larryz,jojic,sbkang*}*@microsoft.com*

Abstract

In this paper, we propose a method for jointly computing optical flow and segmenting video while accounting for mixed pixels (matting). Our method is based on statistical modeling of an image pair using constraints on appearance and motion. Segments are viewed as overlapping regions with fractional (α) contributions. Bidirectional motion is estimated based on spatial coherence and similarity of segment colors. Our model is extended to video by chaining the pairwise models to produce a joint probability distribution to be maximized. To make the problem more tractable, we factorize the posterior distribution and iteratively minimize its parts. We demonstrate our method on frame interpolation.

1. Introduction

Motion estimation is inherently ill-posed, and techniques proposed for solving it range from spatial regularization [8], use of global or parametric motion models [2], to segmentation [18]. Segmentation-based approaches partition the image into regions, with each region assuming a parametric motion model (such as affine motion). Recently, color segmentation approaches have gained in popularity, both for optical flow computation [7, 14] and stereo [17, 22]. They use the reasonable assumption that similarly colored neighboring pixels have similar motions (or depths). Color discontinuities are used to delineate object boundaries and thus motion discontinuities. However, segments tend to be *statically-determined* prior to actual motion estimation.

The problem with using precomputed static color segments is the inability to recover from segmentation errors. Segmentation of a single image is typically ambiguous without the context of neighboring images. In this paper, we propose a technique for producing temporally consistent segmentations. That is, segments across neighboring images have similar shapes and colors. Using the consistent segmentations, we convert the motion estimation problem from pixel to segment matching.

We apply our results to the problem of video editing, more specifically frame interpolation. To produce higher-quality results, we also integrate matting (extraction of foreground and background colors) into our algorithm. Matting has been shown to be an effective tool for handling observed mixed color pixels [5].

2. Previous work

Early work in optical flow centered around efficient methods using image gradients and hierarchical approaches [8, 13]. Black and Anandan [2] expanded upon these methods using robust statistics to handle discontinuities in the flow field. Szeliski and Coughlin [16] use 2D splines to approximate pairwise image flow.

Layered or segmentation approaches were proposed to allow for discontinuities while being able to enforce constraints on the flow within the segments. Wang and Adelson [18] were arguably the first to develop this idea, using the affine model for flow. They proposed an iterative approach to create and remove segments, based on the pixelwise flow computed using a method similar to [2]. For segmentation, several methods use an expectation maximization approach [9, 1, 20]. These methods include using mixture models [9], minimum description length encoding for segment creation [1] and spatial coherence [20]. Different constraints for flow vectors within segments have been used, including smoothness constraints [21, 4] and parameterized motion models [3]. Unfortunately, results obtained using flow-based segmentation tend to be unpredictable at object boundaries due to the local aperture problem. One approach for joint segmentation and flow computation is described in [10], but the patch-based method is computationally expensive.

Instead of segmenting based on just flow, there are techniques that use color information [7, 14] or a combination of flow and color [11]. Color-based segmentation has also been successfully used in the context of stereo and view interpolation [17, 22].

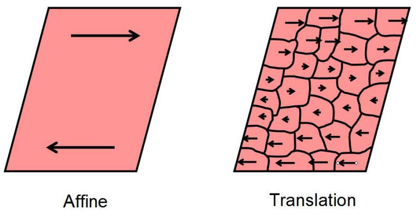

Figure 1. Affine movement within large segments can be approximated using an over-segmentation with translational movement.

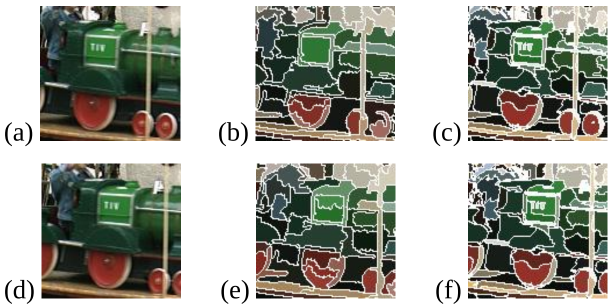

Figure 2. Consistent segmentation: (a,d) Cropped original frames, (b,e) segmentations computed independently, (c,f) consistent segmentations.

3. Our approach

Our work is motivated by the application of high-quality video editing. This requires not only good optical flow estimation, but also accurate segmentation and background/foreground separation. The degree to which an image should be segmented and the complexity of the motion model are interdependent. One can adopt a relatively coarse segmentation with affine, 3D planar, or even higher degree motion models [1, 18, 20]; the alternative is a fine segmentation with a simple translational motion. We use the latter strategy.

As shown in figure 1, an object undergoing some complex motion can be reasonably well approximated as a collection of small regions with simple motions (translation in our case). We chose this strategy to reduce the possibility of not finding object, and thus motion, boundaries. In addition, incrementally enforcing temporal consistency with smaller (and *adaptive*) regions is easier than with large regions. Temporal consistency involves ensuring similarity of segment shape and color across time as well as spatial coherence (figure 2). Since the number of segments is much smaller than the number of pixels, segment correspondence can be performed much more efficiently.

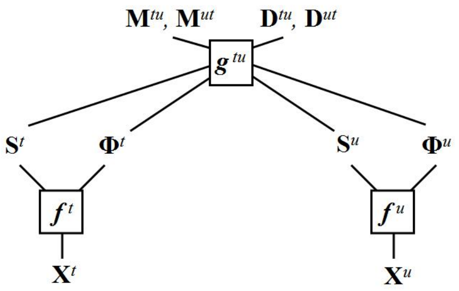

Figure 3. Factor graph of proposed approach.

We formulate the simultaneous estimation of flow, segmentation, and matting as a single generative model using appearance and motion constraints. The generative model is cast as a factor graph [12]. A factor graph defines a function as a product of it's factors, and it can be used to express a wide variety of models, such as Markov Random Fields and Bayesian networks. In Section 3.2, we describe the factor graph for an image pair, depicted in figure 3, followed by generalization to image sequences. The variable nodes are the images $\mathbf{X}_t$ and $\mathbf{X}_u$, the segmentation parameters $\mathbf{\Phi}$, $\mathbf{S}$, and the motion parameters $\mathbf{M}$, $\mathbf{D}$. Following the property of a factor graph, the joint distribution is equal to the product of the function nodes f^t, g^{tu}, and f^u, which are themselves functions of the variables.

We begin by describing our generative model for a single image expressed by the functions f^t and f^u, for images $\mathbf{X}^t$ and $\mathbf{X}^u$ respectively, in the factor graph. We assume that each image consists of a set of color segments. To handle pixels which may receive contribution from multiple segments, each pixel can be assigned to two segments with a corresponding alpha value for blending.

To make the inference problem tractable, we factorize the posterior distribution and iteratively minimize its parts using a variational approach. In practice, this translates to an algorithm that iteratively computes the segmentation given the current estimate of the motion vectors, followed by computing the motion vectors given the segmentation. This process is repeated until convergence. Alternatively, other techniques such as loopy belief propagation and sampling could be used for maximizing the function defined by the factor graph.

3.1 Overlapping segments as a generative model of a single image

We model each image as a set of segments, each with corresponding appearance and motion. Each segment k has a distribution over the pixel colors and coordinates described by parameters $\mathbf{\Phi}_k$. In our work, we used the Gaussian model described by the mean μ_k and covariance matrix $\mathbf{\Sigma}_k$, for the segment's color distribution. We also used the Gaussian model with mean $\boldsymbol{\eta}_k$ and covariance matrix $\mathbf{\Delta}_k$ to describe the spatial distribution of the segment's pixels. Therefore, the parameters describing the color and coordinate variation in the segment are given by $\mathbf{\Phi}_k = (\boldsymbol{\mu}_k, \mathbf{\Sigma}_k, \boldsymbol{\eta}_k, \mathbf{\Delta}_k)$. Other possible parameterizations of the probabilistic constraints on the segment's shape and appearance include mixtures of Gaussians, color histograms, feature mixtures, or image patch mixtures.

The segment parameters describe the extent of the within-segment pixel similarity and variability. In addition to this generalized description, the segments also have their realization in the image, which could be defined, for example, by the index map $\mathbf{S} = \{s_i | s_i \in \{1, ..., K\}\}$, where i denotes the pixel index, and $s_i = k$ indicates that the i-th pixel belongs to the k-th index. For each segment k, we also treat as *hidden* the particular realization of colors $\mathbf{c}_{i,k}$ for the segment pixels.

Treating the colors inside the segment as hidden enables us to model segment overlaps and alpha-blending of their boundaries when necessary. In particular, instead of a single index map $\{s_i\}$, we can assign each pixel i to *two* hidden segment indices s_i^1 and s_i^2 and a hidden alpha value α_i. The observed color of the pixel is expressed as $\mathbf{c}_i \approx \alpha_i \mathbf{c}_{i,s_i^1} + (1-\alpha_i)\mathbf{c}_{i,s_i^2}$. In our model, we handle boundaries between two segments only. For short, we use the notations $\mathbf{c}_i^1 = \mathbf{c}_{i,s_i^1}$ and $\mathbf{c}_i^2 = \mathbf{c}^2_{i,s_i^2}$. Note that the pixels that are not on the boundary are captured by the case $s_i^1 = s_i^2$.

This parametrization of the variability in the data corresponds to a generative model of a single image, which generates pixel colors and positions by the following hierarchical statistical process. First, hidden index pairs (s_i^1, s_i^2) are sampled from a uniform distribution. Then, two hidden pixel colors $\mathbf{c}_i^1$ and $\mathbf{c}_i^2$ are generated with the assumption that the position $\mathbf{r}_i$ of the pixel is observed. Then, the alpha value α_i is generated from a prior distribution (either uniform or the one favoring $\alpha = 0$ or $\alpha = 1$). The generative process ends by generating the observed pixel color $\mathbf{c}_i$ by a noisy alpha-blending of the two parent pixel colors $\mathbf{c}_i^1$ and $\mathbf{c}_i^2$. (Note again that non-boundary pixels would simply have the two hidden parents belonging to the same segment.)

The first set of distributions in the generative model would correspond to the priors

$$p(s_i^1) = p(s_i^2) = 1/K, \tag{1}$$

but since they are constant, these factors would have no influence on the inference in the factor graph. Hence, we can omit these whenever it leads to more compact equations.

The following pair of factors for each pixel i correspond to the conditional distribution over the two hidden pixel colors $\mathbf{c}_i^1$, $\mathbf{c}_i^2$ and position $\mathbf{r}_i$, given the segment indices s_i^1, s_i^2:

$$\begin{aligned} p(\mathbf{c}_i^1, \mathbf{r}_i | s_i^1, \boldsymbol{\phi}_{s_i^1}) &= \mathcal{N}(\mathbf{c}_i^1; \boldsymbol{\mu}_{s_i^1}, \mathbf{\Sigma}_{s_i^1}) \mathcal{N}(\mathbf{r}_i; \boldsymbol{\eta}_{s_i^1}, \mathbf{\Delta}_{s_i^1}) \\ p(\mathbf{c}_i^2, \mathbf{r}_i | s_i^2, \boldsymbol{\phi}_{s_i^2}) &= \mathcal{N}(\mathbf{c}_i^2; \boldsymbol{\mu}^c_{s_i^2}, \mathbf{\Sigma}_{s_i^2}) \mathcal{N}(\mathbf{r}_i; \boldsymbol{\eta}_{s_i^2}, \mathbf{\Delta}_{s_i^2}), \end{aligned}$$

where $\mathcal{N}(x; \mu, \mathbf{\Sigma}^2)$ is the usual normal distribution with mean μ and covariance matrix $\mathbf{\Sigma}$. The observed color of the pixel is related to the segment colors, using alpha-blending, in our last factor of the model:

$$p(\mathbf{c}_i | \mathbf{c}_i^1, \mathbf{c}_i^2, \alpha_i) = \mathcal{N}_i(\mathbf{c}_i; \alpha_i \mathbf{c}_i^1 + (1-\alpha_i)\mathbf{c}_i^2, \boldsymbol{\psi}). \tag{2}$$

If we assume that the observations defining the image $\mathbf{X}$ are the pixel colors $\mathbf{c}_i$ and coordinates $\mathbf{r}_i$ of all pixels, then the product of all factors is in fact a normalized distribution

$$\begin{aligned} &f = p(\{\mathbf{c}_i, \mathbf{c}_i^1, \mathbf{c}_i^2, \mathbf{r}_i, s_i^1, s_i^2\}_{i=1}^I) = \\ &\textstyle\prod_i p(s_i^1) p(s_i^2) p(\alpha_i) p(\mathbf{c}_i^1, \mathbf{r}_i | s_i^1) p(\mathbf{c}_i^2, \mathbf{r}_i | s_i^2) p(\mathbf{c}_i | \mathbf{c}_i^1, \mathbf{c}_i^2, \alpha_i). \end{aligned}$$

The prior on α is constructed to favor values close to one, and this is expressed through normalized factors $p(\alpha_i)$.

If we were to only segment the image, we could develop an algorithm that jointly segments the data and learns the parameters $\boldsymbol{\psi}$ and $\{\mathbf{\Phi}_k\}_{k=1}^K$, using either the exact expectation maximization (EM), or a faster variational version. However, once we introduce new constrains among segments in a *pair* of images, exact inference will become intractable and approximate algorithms will be needed.

3.2 Modeling an image pair

In this section, we consider a statistical model of a pair of images with corresponding segments. Such image pairs can be found, for example, in a video sequence, a collection of photographs of the same (possibly dynamic) scene, or even in a collection of photographs or video sequences containing similar objects. While the modeling framework we develop here is rather general, our focus in this paper is on closely matching image pairs, leading to applications in motion analysis.

When computing our motion vectors we will be making the critical assumption that the segmentations of the images are consistent. That is, if two pixels belong to the same segment in image $\mathbf{X}^u$, then their corresponding pixels in the other image, $\mathbf{X}^t$, also belong to the same segment. Figure 2 illustrates an example of a non-consistent and consistent segmentation. As a result, our model of an image pair not only incorporates the motion vectors of the segments, but also the consistent segmentation constraint.

To differentiate between the hidden variables associated with $\mathbf{X}^t$ and $\mathbf{X}^u$, we use superscripts ^{u} and ^{t}.

3.2.1 Segment mapping variables

The segment correspondence is defined by two mappings: the mapping $\mathbf{M}^{tu}$ of segments describing $\mathbf{X}^t$ to the segments describing $\mathbf{X}^u$, and the mapping $\mathbf{M}^{ut}$ (which is similarly defined). The two mappings should be mostly consistent, but given that certain pieces of scene could disappear from one image to the next (especially in dynamic scenes), and that one image may be over-segmented in certain regions, deviations from 1-1 correspondence are allowed. In addition, optimal 1-1 segment matching is NP-hard; the use of dual maps increases the robustness of the approximate inference (i.e., reduces local minima problems).

Each mapping is defined by a set of variables $\mathbf{M} = \{m_k\}_{k=1}^{K}$, one for each segment k, which point to the corresponding segment in the other image. For example, $m_k^{tu} = j$ indicates that the k-th segment of the image $\mathbf{X}^t$ corresponds to the j-th segment of $\mathbf{X}^u$. By symmetry, in this example we would expect that $m_j^{ut} = k$, which is most often the case in the experiments we report here, but the flexibility of allowing $m_{m_k^{tu}}^{ut} \neq k$ is important for modeling occlusion and disocclusion events in sequences, as well as matching photographs in which many segments may need to remain unmatched.

While multiple segments may have the same mapping, we insist that each segment k has a mapping in the other image. More specifically, $m_k^{tu} \in \{1, ..., K^u\}$, where K^u is the number of segments in the generative model of $\mathbf{X}^u$.

3.2.2 Segment displacement variables

In addition to the mapping variables, we also define bidirectional displacement field (flow) between the pixels in two images. The displacement field $\mathbf{D}^{tu} = \{d_k^{tu}\}$ is defined as a set of flow vectors for each segment k in the image $\mathbf{X}^t$. The actual dense image flow is constructed by assigning to each pixel the flow associated with its segment. Similarly, we define the flow $\mathbf{D}^{ut} = \{d_\ell^{ut}\}$. Assuming the segments are not occluded and taking the displacement to be the shift in segment centroids, we have $\boldsymbol{\eta}_k^t + d_k^{tu} = \boldsymbol{\eta}_{m_k^{tu}}^u$ and $\boldsymbol{\eta}_{m_k^{tu}}^u + d_{m_k^{tu}}^{ut} = \boldsymbol{\eta}_k^t$. However, we allow deviations from this in order to deal with partial or total occlusion of segments. The k-th segment of $\mathbf{X}^t$ can be mapped to the ℓ-th segment in $\mathbf{X}^u$ according to $\mathbf{M}^{tu}$, but some part of segment ℓ may be occluded by another object. Hence, the flow in $\mathbf{D}^{tu}$ may deviate from the shift in the segments' centroids.

Having introduced new variables, we now turn the reader's attention to the new constraints needed to properly capture correlations among variables describing a pair of images. As before, these constraints are expressed in terms of factors whose product defines the optimization criterion to be optimized using variational inference. Each factor is a function of the subset of the variables in the model.

3.2.3 Constraints on the variables of an image pair

Our model of image pairs will be broken into two parts. The first expresses our desire for segmentations across images to be consistent. The second relates the motion variables of the segmentations.

In the first part, we enforce a consistent segmentation across images. To do this, we place constraints on the segment maps $\mathbf{S}^{1,t}$, $\mathbf{S}^{2,t}$, $\mathbf{S}^{1,u}$ and $\mathbf{S}^{2,u}$. One constraint ensures a locally consistent segmentation within each image. Another constraint enforces segment shapes to be consistent across images. Shape consistency is accomplished by favoring segment assignments for which neighboring pixels belong to the same segment within an image, and to the same corresponding segment across images. These constraints are formulated as the following multiplicative factors:

$$\begin{aligned} h_i^t &= \prod_{j \in \varepsilon_i \setminus i} \left(\epsilon [s_i^{1,t} \neq s_j^{1,t}] + (1-\epsilon)[s_i^{1,t} = s_j^{1,t}] \right) \times \\ &\times \prod_{j \in \varepsilon_i \setminus i} \left(\epsilon [s_i^{2,t} \neq s_j^{1,t}] + (1-\epsilon)[s_i^{2,t} = s_j^{1,t}] \right), \end{aligned}$$

where ε_i is a small [5x5] neighborhood around pixel i, and ϵ is a value close to 0 that controls how much we favor locally consistent segmentations. In our experiments, $\epsilon = 0.01$. We can then apply the same constraint across image pairs:

$$\begin{aligned} h_i^{tu} =& \prod_{j \in \varepsilon_{r_i^t + d_i^{tu}}} \left(\epsilon [s_j^{1,u} \neq m^{tu}(s_i^{1,t})] + (1-\epsilon)[s_j^{1,u} = m^{tu}(s_i^{1,t})] \right) \times \\ &\times \prod_{j \in \varepsilon_{r_i^t + d_i^{tu}}} \left(\epsilon [s_j^{2,u} \neq m^{tu}(s_i^{1,t})] + (1-\epsilon)[s_j^{1,u} = m^{tu}(s_i^{1,t})] \right), \end{aligned}$$

where $m^{tu}(k) = m_k^{tu}$ and $\varepsilon_{r_i^t + d_i^{tu}}^u$ denotes a small neighborhood around the pixel with coordinates $r_i^t + d_i^{tu}$ in image $\mathbf{X}^u$. $r_i^t + d_i^{tu}$ denotes the coordinates to which the i-th pixel is going to move according to the displacement field $\mathbf{D}^{tu}$. In both constraints, the segment index pair s^1, s^2 receive an asymmetric treatment, with the first index being the primary one, and the second index being only influenced by the assignment of the first one in the matching neighborhood.

The second part of our model for image pairs concerns our motion variables $\mathbf{M}$ and $\mathbf{D}$. Our first factor states that the mean color of corresponding segments from image $\mathbf{X}^t$ to $\mathbf{X}^u$ should be similar:

$$z_k^{tu} = \mathcal{N}(\boldsymbol{\mu}_k^t; \boldsymbol{\mu}_{m_k^{tu}}^u, \boldsymbol{\psi}^{tu}), \tag{3}$$

with analogous factors z_ℓ^{ut} for the mapping from image $\mathbf{X}^u$ to $\mathbf{X}^t$.

As we stated before, in most cases, the motion vector for a segment can be computed by finding the shift in the corresponding segments' centroids, assuming consistent segmentation across images. To handle exceptions, such as when a segment is occluded, we will allow a segment's motion d_k^{tu} to be one of the differences between any matching segments' centroids within a neighborhood:

$$v_k^t = 1 - \prod_{j \in \varepsilon_k} \left(1 - [d_k^{tu} = \boldsymbol{\eta}_j^t - \boldsymbol{\eta}_{m_j^{tu}}^u]\right), \tag{4}$$

where ε_k is defined as the set of all segments in the neighborhood of k. More specifically, this neighborhood includes each segment j which has at least one pixel i in the neighborhood of $\boldsymbol{\eta}_k^t$ for which $s_i^t = j$. This hard constraint regularizes the mapping and displacement estimation, while still allowing some deviation of the segments' flow. This approach is similar to that taken by [17] for occluded segments.

Our final factor enforces smoothness in the computed velocities. That is, the displacement vectors d_k^{tu} should be similar for neighboring segments:

$$v_k^{t,u} = \prod_{j \in \varepsilon_k} \mathcal{N}(d_k^{tu}; d_j^{tu}, \boldsymbol{\delta}) \mathcal{N}(d_k^{tu}; -d_{m_j^{tu}}^{ut}, \boldsymbol{\delta}) \tag{5}$$

In a single factor, all the constraints between the sets of hidden variables and parameters associated with individual images can be expressed as:

$$\begin{aligned} g^{t,u} \quad = \quad & \left(\prod_{k=1}^{K^t} z_k^{tu} v_k^t v_k^{tu}\right)\left(\prod_{\ell=1}^{K^u} z_\ell^{ut} v_\ell^u v_\ell^{ut}\right)\left(\prod_{i=1}^{I^t} h_i^t h_i^{tu}\right) \times \\ & \times \left(\prod_{j=1}^{I^u} h_j^u h_j^{ut}\right), \end{aligned}$$

where K^t is the number of segments of image $\mathbf{X}^t$ and I^t is the number of pixels in the image. Similar notation is used for the parts of the other image, $\mathbf{X}^u$.

The probability distribution over all the variables describing the image the pair is then expressed as

$$P = \frac{1}{Z} f^t g^{t,u} f^u, \tag{6}$$

where Z is the normalization constant, and f^t and f^u are the generative models of individual images defined by overlapping segments as in the previous section.

3.3 Modeling a sequence of images

A straightforward extension of the image pair model is the following joint probability distribution,

$$P = \frac{1}{Z} \prod_t f^t g^{t,t+1}, \tag{7}$$

obtained by chaining the model of (6). Note, however, that the model can be made stronger by adding terms connecting distant frames as well, e.g.,

$$P = \frac{1}{Z} \prod_t f^t \prod_{\triangle t} g^{t,t+\triangle t}. \tag{8}$$

We used parameterization (7) for computational efficiency reasons.

4 Inference

In the previous sections, we introduced a number of hidden variables and parameters. We can show that the normalization constant Z is only a function of the parameters, such as the segmentation consistency parameter ϵ which controls the Markov random field on segment indices $\mathbf{S}$, the inter-image color noise covariance matrices $\boldsymbol{\psi}$, and the variance $\boldsymbol{\delta}$ for motion regularization. Experimental results have been fairly robust to different settings of ϵ; the results shown in this paper were obtained with $\epsilon = 0.01$. The covariance matrices $\boldsymbol{\psi}$ were held constant and set equal for all segments. In practice, $\boldsymbol{\psi}$ should be set to some scalar of the image noise. $\boldsymbol{\delta}$ was set conservatively to 20.0.

Inference is based on minimizing the free energy [15]

$$F = \int_{\mathcal{H}} Q \log Q - \int_{\mathcal{H}} Q \log P, \tag{9}$$

which is the lower bound on the likelihood of the data $\int_{\mathcal{H}} P$, with P given by (7). The form of the approximate posterior over the hidden variables $Q(\mathcal{H})$ is chosen so as to speed up the inference while keeping as much of the needed uncertainty and correlation among hidden variables. We found the following factorization of Q to be particularly useful:

$$\begin{aligned} Q \quad = \quad & \prod_t \prod_{i=1}^{I^t} \left(q(s_i^{1,t}, s_i^{2,t}) q(\mathbf{c}_i^{1,t}, \mathbf{c}_i^{2,t} | s_i^{1,t}, s_i^{2,t}) q(\alpha_i^t) \right) \times \\ & \times \prod_t \prod_{k=1}^{K^t} q(m_k^{t,t-1}, \mathbf{d}_k^{t,t-1}) q(m_k^{t,t+1}, \mathbf{d}_k^{t,t+1}). \end{aligned}$$

By using this form of Q, the free energy reduces to many additive terms, and each factor of Q, and each parameter of P is only influencing a small number of these additive terms. Thus minimization of F with respect to either individual distributions in Q or parameters of P can be done efficiently if all other parts of Q and P are kept fixed, i.e., we can compute the segmentation keeping the flow fixed, and vice-versa. Iterating these minimizations leads to a reduction in free energy in each step.

Some of the factors in the Q distribution have a constrained form. The distribution over hidden blended colors for each pixel $q(\mathbf{c}_i^{1,t}, \mathbf{c}_i^{2,t} | s_i^{1,t}, s_i^{2,t})$ is expressed by a Gaussian distribution, which is the form the exact posterior has, too, when α_i is given. The posterior distributions over the matting variables are expressed by Dirac functions $q(\alpha_i^t) = \delta(\alpha_i^t - \hat{\alpha}_i^t)$. The distribution $q(s_i^{1,t}, s_i^{2,t})$ is fully expressive in principle; it is a $K^t \times K^t$ table of probabilities that add up to one, but some entries in the table are forced to be zero to speed up the search for $q(s_i^{1,t}, s_i^{2,t})$ that reduces the free energy. In particular, the columns and rows corresponding to the segments which have been far away from the i-th pixel during learning, are zeroed.

Finally, the posterior over the mapping and displacement is assumed to be deterministic (or Dirac, which is in accordance with the hard constraints on the deformation field v_i^t), $q(m_k^{t,u}, d_k^{t,u}) = [m_k^{t,u} = \hat{m}_k^{t,u}]\delta(d_k^{t,u} - \hat{d}_k^{t,u})$. The update on each of these posteriors reduces to searching for the mapping $\hat{m}_k^{t,u}$, which reduces the free energy the most. The displacement $\hat{\mathbf{d}}_k^{t,u}$ is set to either the new segment displacement for this mapping, or to one of the old displacements of the segments in the neighborhood, whichever has the best color correspondence.

The free energy F is iteratively reduced with respect to the parameters of P and Q until convergence. At this point, the displacements $\hat{d}_k^{t,u}$ define the dense flow field useful for a variety of applications, such as frame interpolation, object segmentation, etc.

The image segmentation is initialized using a quad-tree approach that recursively breaks the image into smaller segments based on the variance of the color within the segment. The flow vectors are initialized to 0, from which the mappings with highest overlap are found.

5. Results

In this section, we show three results of frame interpolation. Our first example uses the first 5 even frames from the MPEG dataset of a merry-go-round scene. This is a difficult scene for several reasons: there are many small and thin independently moving objects, ordering between objects is not always preserved, and some of the vertical bars are transparent due to motion blur. To test our algorithm, we synthesize frame 3 using the matting and flow information computed using our algorithm. The results can be seen in figure 4. The flow information for most small objects, including many thin vertical bars looks reasonable, as can be seen in figure 4(d). Because of alpha blending, the final rendered result has smooth boundaries as shown by the highlighted area. The slight blurriness in the interpolated frames is mostly caused by resampling.

Figure 5 shows the results for a pair of photographs (433×500). This example demonstrates the ability of the algorithm to find correct flow vectors even when the displacements are large (> 30 pixels).

Our third example shows results of our algorithm on the popular garden sequence, seen in figure 6. Once again, the rendered interpolated view has soft boundaries due to the alpha matting. Some flow vectors have been incorrectly computed, but these are mostly limited to areas of low texture such as the blue sky (resulting in imperceptible artifacts in the interpolated views).

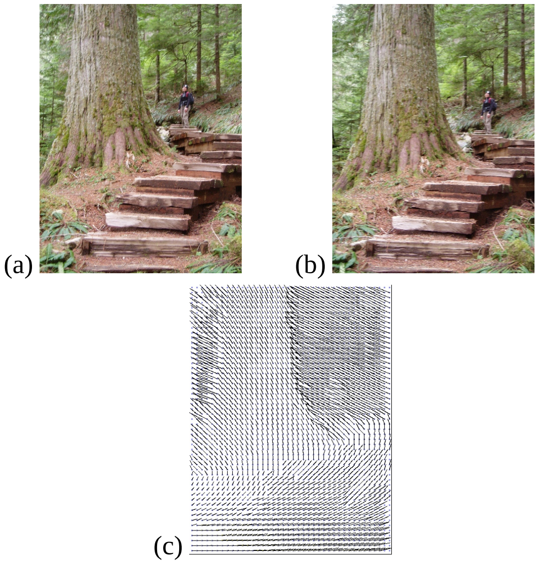

Figure 5. Forest results: Input images (top row), flow field (bottom).

Each example was run for 80 iterations, after which, segments that had inconsistent mappings where assigned flow vectors equal to the average flow vector within a neighborhood of the segment. The number of segments per image with inconsistent mappings was typically small (< 10).

6. Discussion

Our technique is generally applicable to video since it uses only color consistency and similarity in extracting flow. Results show that our decision to combine flow estimation with segmentation and matting was a reasonable one.

However, our technique fails in the presence of significant occlusions, where large groups of segments appear or disappear. This is because our technique has no explicit occlusion reasoning. In addition, since it relies on color similarity, it also fails when colors or intensities change dramatically (e.g., when the camera exposure or the lighting condition suddenly changes). To mitigate this problem, the matching criteria would have to be modified or the frames would have to be preprocessed to match their histograms.

For a video of a rigid scene, we could incorporate epipolar geometry into the formulation to constrain flow. In such a case, depth ordering would be easy, and rendering interpolated views can be performed in a back-to-front manner. However, for videos of dynamic scenes, such ordering does

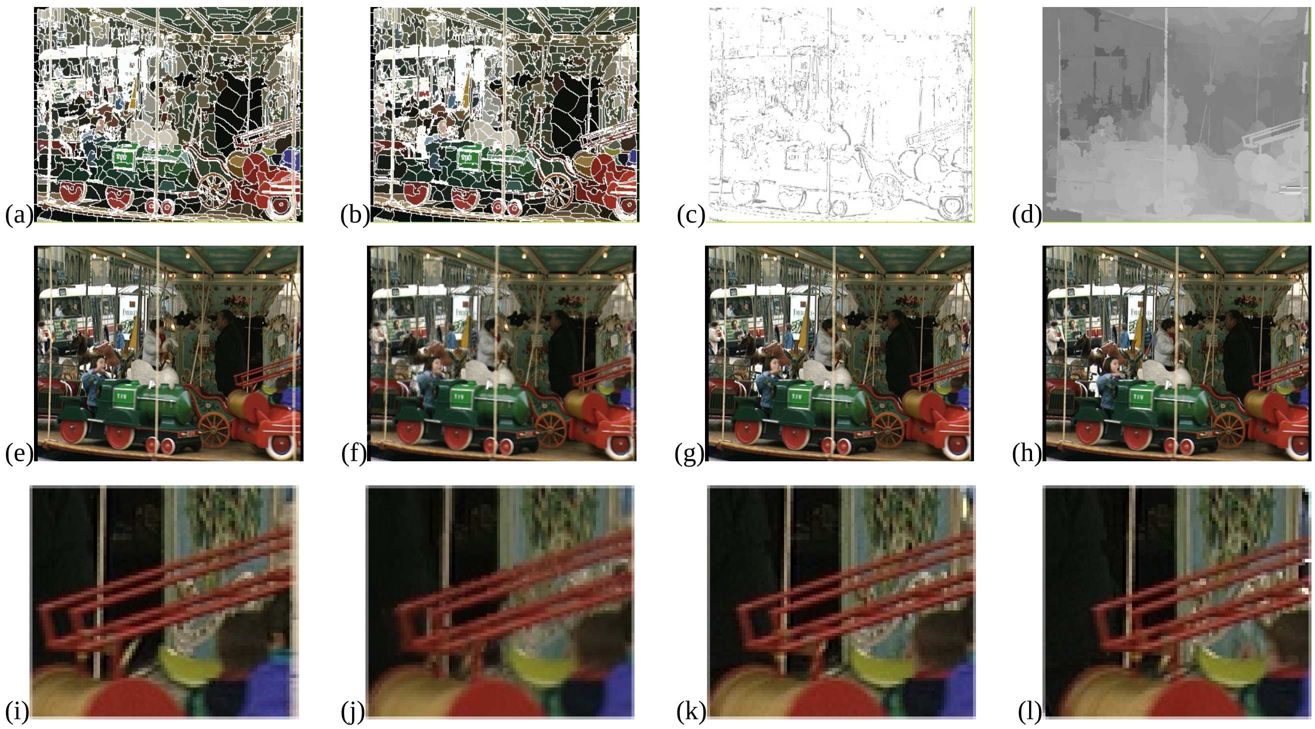

Figure 4. Fair results: (a) Frame 2 segmentation, (b) Frame 4 segmentation, (c) Frame 2 α map, (d) Frame 2 horizontal flow, (e) Frame 2, (f) Interpolated frame 3, (g) Frame 3, (h) Frame 4, (i-l) Close-up views.

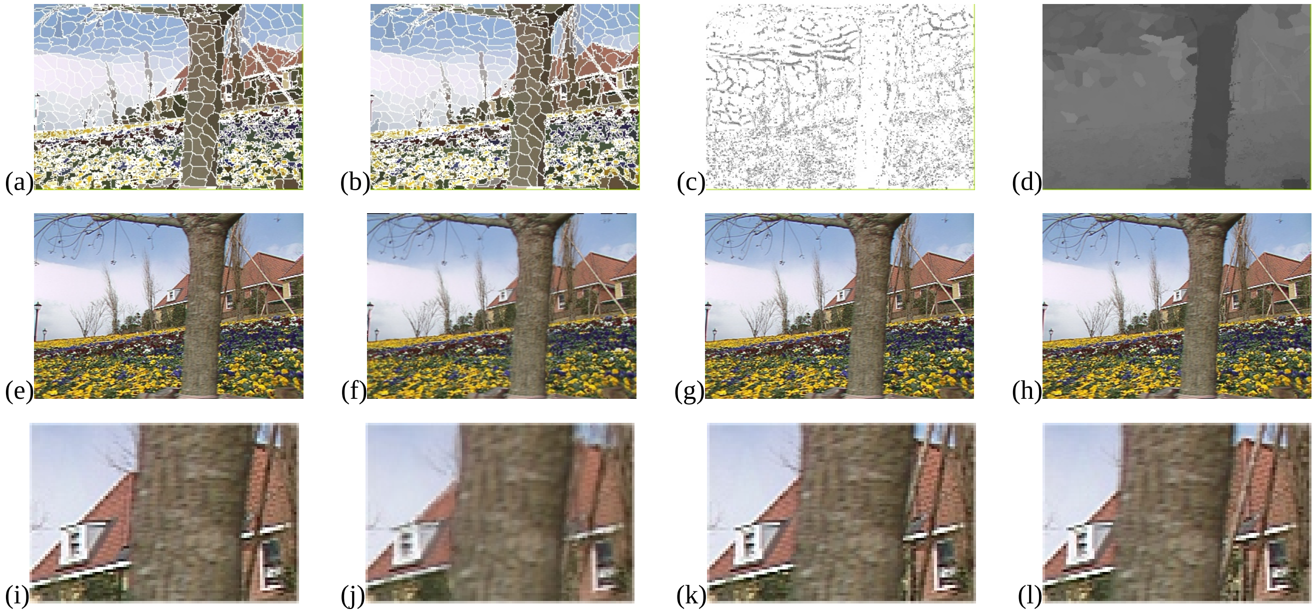

Figure 6. Garden results: (a) Frame 2 segmentation, (b) Frame 4 segmentation, (c) Frame 2 α map, (d) Frame 2 horizontal flow, (e) Frame 2, (f) Interpolated frame 3, (g) Frame 3, (h) Frame 4, (i-l) Close-up views.

not exist. We currently assume that segments with larger flows are closer to the camera.

In cases where motion is significant, holes may appear in the interpolated frame. We currently fill holes by iteratively using the average of boundary colors. A more sophisticated inpainting algorithm (e.g., [6]) can be used instead.

One future direction is to use the segment flows for grouping (based on similarity of flow). This could be used for applications such as recognition. In addition, temporally consistent segmentation would be helpful for automatic segmentation of moving objects.

7. Conclusions

Flow estimation is a particularly difficult problem due to ambiguities in textureless areas, occlusions, and mixed pixels. In our case, we use flow for the purpose of high-quality frame interpolation. As such, extracting *plausibly correct* flow is adequate. However, detecting flow discontinuities at object boundaries (which is non-trivial) is critical. We have shown that our technique is capable of accurately delineating these boundaries.

Our design decisions are based on both expediency and effectiveness. Matching segments instead of pixels is significantly more efficient without sacrificing visual quality. In addition, it reduces the ill-posed nature of flow estimation. We avoid committing to a initial fixed segmentation for flow estimation; instead, we adaptively reshape segments based on both spatial and temporal evidence.

We also deliberately factored in matting to account for mixed pixels. As results show, the extracted α distributions help to significantly reduce typical artifacts (such as haloing at object boundaries) in interpolated frames. In addition, it permits extraction of very thin objects, which would have been very difficult to recover otherwise.

References

[1] S. Ayer and H. Sawhney, “Layered representation of motion video using robust maximum-likelihood estimation of mixture models and MDL encoding,” *ICCV*, pp. 777-784, 1995.

[2] M. Black and P. Anandan, “The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields,” *CVIU*, vol. 63, no. 1, pp. 75-104, 1996.

[3] M. Black and Y. Yacoob, A. Jepson and D. Fleet, “Parameterized models of image motion,” *CVPR*, pp. 561-567 , 1997.

[4] M. Chang, A. Tekalp, and M. Sezan, “Simultaneous motion estimation and segmentation,” *IEEE Trans. on Image Processing*, vol. 6, pp. 1326-1333, 1997.

[5] Y.-Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski, “A Bayesian approach to digital matting,” *CVPR*, vol. II, pp. 264-271, 2001.

[6] A. Criminisi, P. Perez, and K. Toyama, “Region filling and object removal by exemplar-based inpainting,” *IEEE Trans. on Image Processing*, vol. 9, no. 13, pp. 1200-1212, 2004.

[7] B. Heisele, U. Krebel, and W. Ritter, “Tracking non-rigid, moving objects based on color cluster flow,” *CVPR*, pp. 253-257, 1997.

[8] B. Horn and B. Schunck, “Determining optical flow,” *Artificial Intelligence*, vol. 17, pp. 185-204, 1981.

[9] A. Jepson and M. Black, “Mixture models for optical flow computation,” *CVPR*, pp. 760-761, 1993.

[10] A. Kannan, N. Jojic, and B. Frey, “Generative model for layers of appearance and deformation,” In *AIStats* ’05, 2005.

[11] S. Khan and M. Shah, “Object based segmentation of video using color, motion and spatial information,” *CVPR*, pp. 746-751, 2001.

[12] F. Kschischang, B. Frey, and H. Loeliger, “Factor graphs and the sum-product algorithm,” *IEEE Trans. on Information Theory*, vol. 47, no. 2, pp. 498-519, 2001.

[13] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” *Proc. DARPA Image Understanding Workshop*, pp. 121-130, 1981.

[14] D. Mukherjee, Y. Deng, and S. Mitra, “Region based video coder using edge flow segmentation and hierarchical affine region matching,” *Proc. SPIE, Visual Communications and Image processing*, vol. 3309, pp. 338-49, 1998.

[15] R. Neal, and G. Hinton, “A view of the EM algorithm that justifies incremental, sparse, and other variants,” In Jordan, M.I., editor *Learning in Graphical Models*, Kluwer, Dordrecht, pp. 355-368, 1998.

[16] R. Szeliski, and J. Coughlin, “Spline-based image registration,” *IJCV*, vol. 22, no. 3, pp. 199-218, 1997.

[17] H. Tao, H. Sawhney, and R. Kumar, “A global matching framework for stereo computation,” *ICCV*, pp. 532-539, 2001.

[18] J. Wang and E. Adelson, “Representing moving images with layers,” *Proc. of IEEE Trans. on Image Processing*, vol. 3, no. 5, pp. 625-638, 1994.

[19] J. Wang, Y. Xu, H. Shum, M. Cohen, “Video tooning,” *ACM SIGGRAPH and ACM Trans. on Graphics*, pp. 574-583, 2004.

[20] Y. Weiss and E. Adelson, “A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models,” *CVPR*, pp. 321-326, 1996.

[21] Y. Weiss, “Smoothness in layers: Motion segmentation using nonparametric mixture estimation,” *CVPR*, pp. 520-527, 1997.

[22] C. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, “High-quality video view interpolation using a layered representation,” *ACM SIGGRAPH*, pp. 600-608, 2004.