Color Adjacency Modeling for Improved Image and Video Segmentation

Brian L. Price, Bryan S. Morse
Dept. of Computer Science
Brigham Young University
Provo, Utah, USA
(bprice,morse)@cs.byu.edu

Scott Cohen
Advanced Technology Labs
Adobe Systems
San Jose, California, USA
scohen@adobe.com

Abstract—Color models are often used for representing object appearance for foreground segmentation applications. The relationships between colors can be just as useful for object selection. In this paper, we present a method of modeling color adjacency relationships. By using color adjacency models, the importance of an edge in a given application can be determined and scaled accordingly. We apply our model to foreground segmentation of similar images and video. We show that given one previously-segmented image, we can greatly reduce the error when automatically segmenting other images by using our color adjacency model to weight the likelihood that an edge is part of the desired object boundary.

Keywords—Image segmentation, Image color analysis, Image edge analysis

I. INTRODUCTION

Object selection is an important task for many image and video editing applications. A common component in many foreground segmentation algorithms is color, which is used to generate descriptions of the object of interest and the background. There are many ways to implement color models, such as histograms [1], Gaussian mixture models [2], clustering [3], and so on.

While often overlooked, the relationships between colors in an image can also be used for segmentation when boundary characteristics are known. While object boundaries are usually placed on edges in an image, in most images many edges do not correspond to the desired boundary. Such edges produce attractive locations for a segmentation algorithm to place the object boundary incorrectly. A color adjacency model can be used to help distinguish between desirable and undesirable edges for segmentation boundaries.

For example, assume we want to select the frog toy in Fig. 1(a). There are many gradients that do not belong to the object boundary, such as the lines in the background pattern (Fig. 1(b)). Color transitions such as these lines, the edge between the frog’s head and the blue attachment, or the frog’s green body and orange underbelly, create strong edges that may be a desirable object boundary for a given segmentation algorithm. However, since these color transitions occur only within the object or background and never across the object boundary, these edges could be ignored. By modeling not only the colors themselves, but also the color transitions, we may be able to identify such edges and not consider them as potential object boundaries (Fig. 1(c)).

We introduce a method for modeling color adjacency relationships in order to improve the performance of object selection tasks, including the following contributions. First, we present our color adjacency model. Second, we provide several enhancements to the model, specifically allowing for global or local color information and providing means of decontaminating mixed pixels at object boundaries. Finally, we incorporate our model into graph-cut segmentation algorithms and demonstrate improved performance in video and similar-image segmentation tasks. By similar images, we mean images of the same background scene and foreground object but with possible differences due to changes such as the camera position and angle, pose of the object, or small variations in the background scene. This can occur in multiple shots of the same scene or in successive frames of video.

II. RELATED WORK

While many segmentation algorithms use color models [2], [3], especially when propagating segmentation information to similar images or successive video frames, few have incorporated color adjacency relationships. Recently, [4] used a form of color adjacency modeling to help transfer information from a segmented image to similar images by modifying local edge weights. However, the probability of a transition is not modeled directly but instead derived from foreground/background color probabilities. If two colors both appear in an object, the probability of
METHODS

Combining the color components of the image to cause the significance of the locality to q are the red, green, and blue channels of the color at pixel p respectively. Other color representations such as LUV may be used.

To model the color adjacencies, we estimate likelihoods for data points \( t_{pq} \) using a kernel density estimation computed by the Fast Gauss Transform [13]. This method uses a sparse representation and does not require a large amount of storage or coarse quantization.

B. Global vs. Local Information

This formulation gives a model that is independent of the location of the pixels in the image, which may be desirable for applications such as transferring knowledge from one image to another when object placements are not aligned. However, locality may be desirable in certain applications such as video segmentation where the amount of motion is small between frames.

To include locality, spatial position can simply be added to the adjacency tuple. For example, given a two-dimensional image with an RGB color representation, \( t_{pq} = (r_p, g_p, b_p, r_q, g_q, b_q, x, y) \), where \( x \) and \( y \) are the location of the adjacency relationship.

To adjust the locality of the model, the relative range of the location components to the color components of \( t_{pq} \) may be altered. We normalize the color values to the range \([0, 1]\), then scale the image width or height (whichever is greater) to the range \([0, \omega]\), with the other dimension scaled accordingly. Smaller values of the parameter \( \omega \) cause the model to become less sensitive to spatial locality, while larger values of \( \omega \) cause the significance of the locality to increase.

C. Color Decontamination

One difficulty in computing color adjacencies for adjacent pixels is that edges are rarely hard boundaries. Because of partial-pixel effects, focus or motion blur, or semi-transparent edges, object boundaries often extend over many pixels. While in some cases we may want to model these mixed pixels, in other cases we would like to use only the colors from the actual objects and not the mixed colors.

To account for this, we apply a color decontamination step by assuming that the colors at adjacent pixels are linear combinations of the color adjacency model to become less sensitive to spatial locality, while by using color decontamination. The model is applied to a new problem set, that of selecting objects in similar images. We also validate the effectiveness of the model on both similar image and video segmentation tasks.

III. METHODS

A. General Color Adjacency Modeling

To model the co-occurrence of adjacent colors in images, we represent each color adjacency combination as an adjacency tuple \( t \) combining the color components of the adjacent colors. For example, for two neighboring pixels \( p \) and \( q \), a tuple \( t_{pq} = (r_p, g_p, b_p, r_q, g_q, b_q) \) may be created, where \( r_p, g_p, b_p \) are the red, green, and blue channels of the color at pixel \( p \) respectively. Other color representations such as LUV may be used.

To model the color adjacencies, we estimate likelihoods for data points \( t_{pq} \) using a kernel density estimation computed by the Fast Gauss Transform [13]. This method uses a sparse representation and does not require a large amount of storage or coarse quantization.
all pixels in the test images. In these cases it is not known whether the adjacent pixels lie along an object edge or not, but not decontaminating these pixels can lead to problems in the adjacency model. For example, consider a simple case of an edge from white to black, which is blurred somewhat to create midrange grayscale values along the edge. If we decontaminate these pixels while training our model, we will model a black-to-white transition. If we do not decontaminate when evaluating new pixels, we will see midrange grayscale transitions that are not represented in the trained model. In order for the decontamination to work best, we must assume that color adjacencies in the test images are along the boundary and attempt to decontaminate them.

In order to decontaminate these unknown transitions, we choose candidate decontaminated colors by sampling pixels near the adjacent pixels in the same row or column as illustrated in Fig. 3(a). It is assumed that one of the decontaminated colors belongs to one of the pixels in one group (indicated by the green brackets), and the other color in the other group. The colors \( u_0 \) and \( u_1 \) of the adjacent pixels are thereby assumed to be a linear blend of the two (candidate) decontaminated colors \( v_0 \) and \( v_1 \). The blending coefficient \( \alpha \) at the pixel with color \( u_0 \) is given by

\[
\alpha(u_0, v_0, v_1) = \frac{(u_0 - v_0) \cdot (v_1 - v_0)}{||v_1 - v_0||^2}
\]

(1)

The distance \( d \) from \( u_0 \) to the closest point on the line between \( v_0 \) and \( v_1 \) can then be computed by

\[
d(u_0, v_0, v_1) = \frac{||(u_0 - v_0) \times (u_0 - v_1)||}{||v_1 - v_0||}
\]

(2)

If the closest point is outside the line segment between \( v_0 \) and \( v_1 \), we set \( d(u_0, v_0, v_1) = \infty \). The values of \( \alpha \) and \( d \) are shown graphically in Fig. 3(b) as they relate to the colors \( u_0 \) and \( u_1 \) at adjacent pixels and the candidate decontaminated colors \( v_0 \) and \( v_1 \).

Two candidates \( v_0 \) and \( v_1 \) are chosen as the decontaminated colors for adjacent pixels with colors \( u_0 \) and \( u_1 \) so that they minimize the cost function

\[
\Gamma(u_0, u_1, v_0, v_1) = \frac{d(u_0, v_0, v_1) + d(u_1, v_0, v_1) + 1}{||v_0 - v_1||^2 + 1}
\]

(3)

on condition that the difference between \( u_0 \) and \( u_1 \) is sufficiently high (we use \( ||u_0 - u_1|| > 10 \) for colors \( u \in [0, 255] \)) and that either \( 0.25 \leq \alpha(u_0, v_0, v_1) \leq 0.75 \) or \( 0.25 \leq \alpha(u_1, v_0, v_1) \leq 0.75 \). The cost function in Eq. 3 is designed to be small when the distance from the adjacent pixel colors to the line between the candidate decontaminated colors is low, ensuring that the adjacent pixel colors are well represented as linear blends of the candidate colors. It is also small when the distance between the candidate colors is large, which helps ensure that the decontaminated colors are not linear blends themselves.

D. Use in Object Selection

To use our color adjacency model to assist object selection, we modulate the gradient magnitudes by multiplying them by

\[
s(x) = \begin{cases} 
(1 + \beta |E(x) - L(x)|)^2 & L(x) > L_I(x) \\
(1 + \beta |L_I(x) - E(x)|)^{-2} & \text{otherwise}
\end{cases}
\]

(4)

where \( \beta > 0 \) adjusts the effect of the color adjacency model and \( L_E(x) \) is the posterior probability (assuming equal priors) that the pixel transition \( x \) belongs to the object boundary (denoted by \( E \)) as given by

\[
L_E(x) = \frac{P(x|E)}{P(x|E) + P(x|\overline{E})}
\]

(5)

where \( I \) denotes the interior of the background or object, and \( P \) is the probability as computed by our color adjacency model. The effect of Eq. 4 is that when \( L_E(x) = L_I(x) \), the edge strength is left unchanged. As \( L_E(x) \) increases, the reweighting factor approaches \((1 + \beta)^2\), and the local edge weight increases. As \( L_E(x) \) decreases, the reweighting factor approaches \((1 + \beta)^{-2} \) which decreases the edge weight.

Fig. 4 visualizes the effect that the probability difference \( L_E(x) - L_I(x) \) has on an image. Red indicates that the probability of the transition being a desired edge is large, and white indicates that the probability of the transition being in the interior is large. More precisely, the range \([0, 1]\) for the difference maps to shades of red, and the range \([-1, 0]\) maps to shades of grey (white). Black indicates that both possibilities are equally likely. Notice how many of the edges in the image that do not correspond to the object boundary but would be attractive places to place a segmentation boundary based on the strength of the gradient have been suppressed. For example, the diagonal lines in the frog toy image have been largely eliminated. Conversely, many edges along the desired boundary have been strengthened, such as along the boundary of the cat.

IV. RESULTS

We show the effectiveness of our method by comparing segmentation results with and without the color adjacency model on sets of similar images and on video frames. We transfer the selection from a manually-segmented image to similar images or subsequent video frames within a common graph-cut framework [3]. We compare this selection to
one created with [3] but with edge weights modified by modulating the gradient magnitudes according to Eq. 4 with and without decontamination. Note that we use the graph-cut framework for our segmentation results here since it is currently the most popular and arguably best interactive segmentation framework. While we use [3], Eq. 4 should apply to any graph-cut or similar method. The ground truth for our data was generated manually using the Quick Selection tool in Adobe Photoshop, and the number of mislabeled pixels is reported as the error. Any pixels within two pixels of the boundary are not included in the error measurement.

Table I shows the results for transferring a selection from one image to five related images, and from one video frame to the next five frames. For similar images, because the size, position, and orientation of the objects may differ greatly between images, we de-emphasize the local component in our model (ω = 0.05). For each example, the error is reduced greatly when including the color adjacency model. For the video examples, we enforce locality more by using ω = 1.5. Segmenting video using our method clearly improves performance.

Note that while for this experiment we use the same model for each frame in a video sequence, in long videos where the object or background change dramatically best performance is achieved by recomputing the color adjacency model at each frame. This is how the color adjacency model (computed in subsecond time) was applied in [12], which gives segmentation results using our model for sequences of more than 350 frames in length.

Results with and without color adjacency modeling are shown in Fig. 5. When using global color models alone, there are many disjoint pieces and holes in the result.

![Figure 4. Edge reweighting. Red indicates likely desired edges, white likely interior edges, and black equally likely edges.](image)

![Table I](image)

<table>
<thead>
<tr>
<th>Images</th>
<th>Standard</th>
<th>With Adjacency</th>
<th>Decontaminated Adjacency</th>
<th>% Reduced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cat</td>
<td>7.5</td>
<td>2.4</td>
<td>0.5</td>
<td>93%</td>
</tr>
<tr>
<td>Orca</td>
<td>29.7</td>
<td>17.3</td>
<td>16.1</td>
<td>46%</td>
</tr>
<tr>
<td>Frog</td>
<td>10.4</td>
<td>2.8</td>
<td>2.4</td>
<td>77%</td>
</tr>
<tr>
<td>Video</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>6.2</td>
<td>0.9</td>
<td>0.5</td>
<td>91%</td>
</tr>
<tr>
<td>54</td>
<td>3.4</td>
<td>0.4</td>
<td>0.4</td>
<td>88%</td>
</tr>
<tr>
<td>Cat</td>
<td>9.7</td>
<td>0.4</td>
<td>0.3</td>
<td>97%</td>
</tr>
<tr>
<td>Football</td>
<td>139.2</td>
<td>87.2</td>
<td>87.2</td>
<td>37%</td>
</tr>
<tr>
<td>Man</td>
<td>58.7</td>
<td>30.1</td>
<td>29.8</td>
<td>49%</td>
</tr>
<tr>
<td>Ballet</td>
<td>16.2</td>
<td>1.2</td>
<td>1.0</td>
<td>94%</td>
</tr>
</tbody>
</table>

![Figure 5. Example segmentation results.](image)
Reweighting the edges based on color adjacency greatly improves the results of the segmentation. Fig. 6 shows the results for all five of the frog images.

V. CONCLUSION

We have presented a method of improving object selection in cases where the boundary information is known, such as in video or sets of similar images, by the application of a color adjacency relationship model. By increasing the strength of edges similar to the desired object boundary and weakening other edges using a color adjacency model, segmentation algorithms can more easily isolate the correct object boundary.

REFERENCES


