This appears in: CVPR Workshop on Perceptual Organization in Computer Vision, 2006

Globally Optimal Interactive Boundary Extraction Using Markov Chain Modeling

Christina Pavlopoulou Avi Kak
School of Electrical and Computer Engineering, Purdue University
{pavlo,kak}@ecn.purdue.edu

Abstract

We present a novel boundary-based (discontinuity tracking) hierarchical statistical criterion to address the interactive contour extraction problem. Our criterion relies on a Markov Chain representation of the boundary and can be efficiently optimized using Dijkstra's algorithm for solving the shortest paths problem. Unlike other criteria optimized with Dijkstra's algorithm, ours is capable of extracting geometrically complex boundaries even when the features incorporated in the objective function are based only on user markings on a small part of the image. The critical quantity in our criterion that yields the above-mentioned results is a normalization factor that boosts the probability of a particular boundary segment based on the candidate boundary segments in its vicinity. Although similar in spirit to the technique of non-maximum suppression routinely employed in edge detection, our method boosts gradually the probability of a particular segment given its surroundings using windows of increasing size in a hierarchical fashion.

Keywords: *interactive segmentation, shortest paths, Dijkstra's algorithm, Markov Chain modeling, hierarchical modeling*

1. Introduction

Segmentation is an important computer vision problem. However in the 2D case, it is ill-posed unless previously constructed models of the object to segment are used or the help of the user is enlisted. Although interactive segmentation has been the focus of many research papers in the past, it is still an unsolved problem. In this paper, we will focus on interactive contour extraction using boundary-based or discontinuity-based modeling. The goal of such an approach is the accurate and reliable boundary extraction of an object with the *least* amount of human input.

Typical boundary-based objective criteria for the particular problem are energy-based, which can also be interpreted from a Bayesian perspective. One of the earliest examples of such a criterion is Snakes [12]. The advantage of such criteria is that they can be globally optimized. If the input consists of an approximate delineation of the structure to extract, then the global optimum can be found using dynamic programming [1, 11]. If on the other hand, the user input consists of points on the boundary to be segmented, then the optimum can be found more efficiently using Dijkstra's algorithm, *when the weights of the graph constructed are positive*, as for example in [6, 9, 13, 15].

As we shall explain in more detail in the following sections, the definition of the weights is critical for the "effectiveness" of such a method, i.e. for its ability to extract geometrically complex boundaries with a small amount of human input. In particular the sign of the weights affects not only the choice of the algorithm but also the form that the optimization criterion should have. The various objective functions proposed so far typically do not result in all positive or all negative weights. However, their optimization using Dijkstra's algorithm (and in most cases dynamic programming) dictates their transformation to either positive or negative. This transformation is oftentimes achieved by adding a constant to the objective criterion. Although, this allows for the use of Dijkstra's algorithm, the optima of the function are not preserved. Such transformation is equivalent to the addition of an extra smoothing term which will in general prevent the method from extracting geometrically complex boundaries with small amount of human input.

In this paper we propose a statistical criterion that can be optimized globally using Dijkstra's algorithm for solving the single source shortest paths problem. Our criterion models the boundary to be extracted with a Markov Chain. The derivations of the transition probabilities lead to a normalization factor that increases the probability of a boundary segment when it is "lo-

cally best". This is reminiscent of the technique of non-maximum suppression routinely used in edge detection. Our criterion can be optimally optimized using Dijkstra's algorithm. We also present a hierarchical extension, that solves instability issues associated with the calculations of the probabilities and incorporates higher order dependencies between the points of the contour.

The rest of the paper is organized as follows. In the following section we briefly review related work. We continue with a short presentation of the shortest paths optimization and a discussion about the weights and the transformation of the solutions. Subsequently, we introduce our criterion and we theoretically study the effects of the parameters it involves. To this end we introduce the notion of *contrast* between two paths, which is related to how effective a weighting scheme is in extracting complex boundaries. We then present the hierarchical extension of our criterion, followed by experimental results and conclusions.

2. Previous Work

Broadly speaking, segmentation methods can be divided into two categories: region-based and boundary-based. Region-based methods are more suitable for extracting multiple structures from an image and the criteria used typically need to be evaluated on the entirety of the pixels in the interior and/or exterior of the object, in order to determine the boundary. Interactive region-based methods include for example the work reported in [4, 2, 18].

Our method falls in the second category; it is a boundary-based method. These methods are more suitable for extracting the boundary of a single object in an image and the criteria involved typically are evaluated in the vicinity of possible boundaries. Boundary-based methods are the majority of the interactive methods that have appeared so far in the literature. They can be categorized based on the type of human input allowed since it largely determines the optimization method that can be used.

One type of human input is to indicate, usually with a closed contour, where the boundary to be segmented lies. In this scenario, the objective function is usually optimized locally using gradient descent [12], using level-sets propagation [19] and globally using dynamic programming [1, 11]. Recently, graph cuts have been used to optimize a similar criterion in the work reported in [20]. In this work the optimization is carried out in the vicinity of the boundary and thus it can get stuck to local minima.

When the human input allowed consists of points on the boundary of the object to be segmented, shortest paths are a desirable alternative for global optimization. Examples that employ this optimization method include [6, 9, 13, 15]. The work in [6] uses a level-set optimization method similar in spirit to Dijkstra's algorithm. However, heuristics are used to enforce closure constraints. The work in [9, 13] uses Dijkstra's algorithm. However, their criteria are not statistical and they do not provide the hierarchical modeling scheme we do. More recently, the work in [15] also employs a normalization mechanism for defining probabilities for what constitutes a good candidate for inclusion in the final boundary. Although their criterion is based on probabilities, the normalization factor is not derived mathematically.

Additionally, the work in [16] finds the optimal boundary in a piecewise manner and in [14] the authors develop a confidence measure for piecewise boundary extraction. This has the same flavor as our work on hierarchical modeling but the two differ in the underlying mathematical rationale. Recently, in [5] flow network algorithms have been used for boundary based segmentation. Typically, such algorithms tend to be more computationally intensive and are not easily amenable to the incorporation of topological constraints such as the requirement that the extracted region be a single connected entity. Finally, there is a line of work [8] that tries to perform interactively higher level operations. The scope of this work is different from ours since we assume a low-level image representation.

3. Shortest Paths Optimization

Let $G = (V, E)$ be a weighted, directed or undirected graph, where V is the set of vertices and E is the set of edges, and let w be the weight function mapping edges to real values. The weight of a path P, denoted $w(P)$ is the sum of the weights of its constituent edges.

The single-source shortest paths problem seeks to find the paths of minimum cost from a given source node to all the other nodes in a weighted, directed or undirected graph. The problem becomes ill-defined for graphs with negative cycles, since these have $-\infty$ cost. Assuming positive weights, the well-known Dijkstra's algorithm [7] solves the problem in low-order polynomial time $O(V \log V + E)$. For negative weights and non-negative cycles the problem can be solved in $O(VE)$ using the Bellman-Ford algorithm [7]. Whereas in general the optima of the objective function are invariant under linear transformations of the function this is not the case with the shortest paths criteria. Specifically, we have the following observations. The proofs are omitted for reasons of limited space.

Observation 1. *The optimum paths found by a short-*

est paths algorithm are invariant with respect to multiplication of a positive constant with the weights of the graph.

Observation 2. *The optimum paths found by a shortest paths algorithm are not preserved with respect to addition of a constant to the weights of the graph.*

The objective functions that can be optimized using a shortest paths algorithm are summations of functions expressing the goodness of the path *locally*. These functions include well-known and popular criteria like active contour and Bayesian criteria. The best contour according to a Bayesian formulation is given by minimizing the following function (assuming second order dependencies):

$$\log p(C|D) \propto \sum_i \log p(D_{i+1}, D_i, D_{i-1}|C_i, C_{i-1}) + \sum_i \log p(C_{i+1}, C_i, C_{i-1}) \quad (1)$$

where C is the set of pixels on the entire contour and D the corresponding set of observations. In the equation above, C_i denotes a point on the discretized contour C and D_i denotes the observations associated with C_i. The first part of Equation 1, the posterior, expresses our belief about what features should characterize the boundary. The second part of Equation 1, the prior, expresses smoothness constraints. Similar interpretations hold for the active contour functionals.

Note that the weights derived from such criteria are not all necessarily positive. For example, if in Equation 1 the likelihoods are expressed using continuous densities, then we could have both negative and positive weights. Normal densities is a usual choice and in this case feature values close to the mean, that is, highly likely boundary segments, result in negative values.

However, maximizing a function that is piecewise negative or positive cannot be achieved with Dijkstra's algorithm. Provided there are no negatively weighted cycles, more computationally intensive procedures like the Bellman-Ford algorithm have to be employed. Unfortunately, negatively weighted cycles will generally exist. Our graph will typically have negatively weighted cycles especially in parts of the image rich in features. This results in self-intersections in the contour produced, a phenomenon that plagues many active contour methods. To avoid these problems, researchers typically convert the weights resulting from the above function to positive by adding a constant to the criterion. This alters the optima of the function by Observation 2 stated previously. The result of this assumption is that smoother curves will be preferred over the ones dictated by the original criterion. No matter how we change the relative importance between the posterior and the prior, we will never be able to capture certain boundaries with a small amount of human input. For example, consider the boundary of Figure 1 and assume that all the weights are equal to a positive value. If we globally optimize the resulting function, we will *always* get as the best solution boundary ADEFG assuming the user clicks at a point on segment AG. Additional input will be needed to capture boundary ABCEFG. Put simply, for geometrically complex boundaries a lot of input will be sometimes required even for well-defined boundaries.

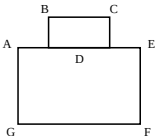


Figure 1. Assuming equal strength of the features on the edges of the above shape, the optimal solution assuming positive weights is always ADEFG and never ABCEFG no matter what value the prior obtains.

Our observation above does not mean that there are no positive weights that can capture complex shapes with small amount of interaction. After all there is a bijection between the reals and the real interval [0,1]. It means that the traditional criteria fail to do so given the additional constraints of efficient optimization and non-intersection of the boundary extracted.

4. Markov Chains for Boundary Extraction

In this section, we present our statistical criterion. Similar in spirit to the Bayesian criterion of Equation 1, the best contour connecting two user-supplied points s and d is the one that maximizes the probability:

$$\mathbf{C}^* = \arg\max_{\mathbf{C}} P(s \xrightarrow{\mathbf{C}} d|D) \quad (2)$$

In traditional Bayesian modeling of images, Bayes rule is used as a first step and subsequently independence assumptions are made in order to eventually obtain Equation 1. The stochastic quantity that one seeks to estimate is the label of the pixel, that is, whether the pixel is on the desired boundary or not. Our formulation follows a different decomposition of 2 in which the *stochastic quantity to be estimated is the location of a boundary point*, that is, the (x, y) coordinates of the image pixel it passes through.

Let $\mathbf{C} = (C_1 \dots C_n)$, where $C_1 = s$ and $C_n = d$, and assume first order Markov dependencies. Then we have:

$$P(C_1, \cdots, C_n|D) = \prod_{i=2}^{n} P(C_i|C_{i-1}, D) \quad (3)$$

where we have assumed that $P(C_1|D) = 1$. To calculate the conditional probabilities, we model the object boundary as a Markov Chain of states, where each image pixel corresponds to a state. Neighboring states, that is, states for which there is a transition, correspond to neighboring pixels. We assume that the pixels are connected with their 8 neighbors. A high transition probability between two states implies that if one state is part of the desired boundary, then the other state is very likely to belong to the boundary too. Thus, we have:

$$P(C_i|C_{i-1}, D]) = \frac{P(C_i, C_{i-1}|D)}{P(C_{i-1}|D)} \quad (4)$$

$$P(C_{i-1}|D) = \sum_{C_i \in \mathcal{N}(C_{i-1})} P(C_i, C_{i-1}|D) \quad (5)$$

The denominator of the transition probability is computed as a marginal probability. The summation involved in that normalization factor is calculated over all neighboring states of state C_{i-1}. The transition probabilities computed this way form a row stochastic matrix, that is, $\sum_{C_i} P(C_i|C_{i-1}, D) = 1$. Note also that the Equations 4 defining the transition probabilities are not affected if instead of a regular probability function we use a non-normalized function.

$P(C_i, C_{i-1}|D)$ denotes the probability that the path consisting of pixels (or states) $\{C_i, C_{i-1}\}$ is part of the desired boundary given the observations. Let D_{i-1} be the observations associated with the boundary segment $\{C_i, C_{i-1}\}$. Using Bayes rule we have:

$$P(C_i, C_{i-1}|D_{i-1}) = \frac{P(D_{i-1}|C_i, C_{i-1})\ P(C_i, C_{i-1})}{P(C_{i-1})} \quad (6)$$

When substituting Equation 6 in Equations 4 the denominator $P(D_{i-1})$ cancels out. We only need to compute $P(D_{i-1}|C_i, C_{i-1})$ and $P(C_i, C_{i-1})$. $P(D_{i-1}|C_i, C_{i-1})$ is the posterior probability of the observations given the boundary. We will compute it as a combination of grayscale and edge features. $P(C_i, C_{i-1})$ expresses our a priori belief about $\{C_i, C_{i-1}\}$ being part of the desired boundary. This probability is defined so that smoother boundaries are encouraged for inclusion in the final solution. It is the quantity responsible for bridging gaps in a boundary.

To optimize Equation 2 we can take the log and reduce the problem to a shortest paths problem. Thus, we need to optimize:

$$\mathbf{C}^* = \arg\min_{\mathbf{C}} \sum_{\{C_i, C_{i-1}\} \in \mathbf{C}} \{\log P(D_{i-1}|C_i, C_{i-1}) + \log P(C_i, C_{i-1}) - \log P(C_{i-1}|D)\} \quad (7)$$

The arguments in the summation are the transition probabilities and their values are between 0 and 1. Since the above equation is a maximization of a summation of negative terms, we can optimize it using Dijkstra's algorithm.

In the above equation, the term $\log P(C_{i-1}|D)$ which is the normalization factor of the transition probabilities, is a negative term subtracted from a negative quantity. The effect of this subtraction is that the locally best set of pixels are favored. This is reminiscent of the technique of non-maximum suppression of edge detection [17]. However, non-maximum suppression is an ad-hoc approach that boosts the graylevel at a locally maximum pixel in proportion to its difference from the gray levels at all the pixels in its immediate neighborhood.

4.1. Grayscale Information

We will use grayscale information to calculate $P(D_{i-1}|C_i, C_{i-1})$. To this end, we assume that our set of observations D_{i-1} consists of grayscale observations. We believe that a reliable calculation of probabilities in this case requires second order Markov dependencies. Second order dependencies are a straightforward extension of the equations described earlier. We basically need to calculate $P(D_{i-1}|C_i, C_{i-1}, C_{i-2})$. In general, D_i consists of the gray levels in a 3×3 window centered at the pixel corresponding to C_i.

We use the observation that the boundary points divide the neighboring pixels into two classes: inside and outside the desired boundary. We therefore have $P(D_{i-1}|C_i, C_{i-1}, C_{i-2}) = P(D^L_{i-1}|M_L)\ P(D^R_{i-1}|M_R)$. where $D^L_{i-1} \cup D^R_{i-1} = D_{i-1}$ and M_L and M_R are models describing the two sides of the boundary. Furthermore, we assume that the observations are independent given their class label. To handle the second-order dependencies induced by the grayscale observations, we have provided a simple extension to Dijkstra's algorithm. We omit the details here.

4.2. Approximation for log of Summation of Exponentials

One must be careful with the computation of the normalization factor of Equation 5. Since it is the log of a summation of exponentials of typically very small numbers, it approaches $-\infty$ very fast making the calculation of the transition probabilities very unstable. To handle this case we proceed as follows:

$$\sum_{x_i} e^{-x_i} = e^{-x_m} \left(1 + \sum_{x_i \neq x_m} e^{-(x_i - x_m)}\right) = e^{-x_m}\ (1+S)$$

where x_m is the minimum of all x_i's. Taking the log of the above expression, we have $-x_m + \log(1+S)$. For $|S| < 0.1$, we can use the approximation $\log(1+S) \simeq S$.

5. Contrast of Weighted Paths

In this section we will develop the notion of contrast which is important in studying the effectiveness of different weighting schemes. We also analyze the effect that the parameters of our statistical model have on the contrast of two paths.

Definition 1. *Let X, Y, Z be three distinct nodes of a weighted graph with positive weights, such that the paths P and Q that connect X to Y and X to Z consist of an equal number of edges. Then, the* contrast *between P and Q is the absolute value of the difference of their costs, i.e.*

$$\mathcal{C}(P,Q) = |w(P) - w(Q)| \tag{8}$$

In the above definition, we defined the contrast on paths having equal number of edges. This restriction is important; since the graph has positive weights, the contrast between any two paths can increase or decrease arbitrarily. Such a quantity would not give us any information about the effectiveness of the weights. Clearly, the more contrast we can achieve between any two paths consisting of an equal number of edges, the more complex boundaries we can capture.

Recall from Equations 4,5, and 7 that the transition probabilities are given as the difference of two logarithmic factors. The first is a summation of logarithmic quantities and the second is the log of a summation of exponentials. The number of quantities in the summations is the key parameter for increasing the contrast between two paths. This parameter corresponds to the number of observations associated with a boundary part. In what follows, we will show that (a) the normalization of Equation 5 increases the contrast, and that (b) increase in the number of observations used to compute the posterior $P(D_i|C_i)$ results in increase of contrast.

To this end, let $G(V,E)$ be a weighted directed graph and let w_{ij} denote the weight of the edge connecting node i to node j. Let the weight w_{ij} be of the form $w_{ij} = \sum_k \log c_{ijk}$, where $c_{ijk} \in (0,1)$. Let now $G^N(V,E)$ be a weighted graph, the weights of which are defined as $w_{ij}^N = w_{ij} - \log \sum_j \exp\{w_{ij}\}$. That is, the graph G^N has the same structure as graph G and its weights are normalized based on the weights of the neighbors using Equations 4 and 5. We will call G^N the *normalized graph* of G. Then, it is easy to see that the following hold. The proofs are omitted for lack of space.

Proposition 1. *The contrast of the paths of the normalized graph is greater than or equal to the contrast of the paths of the original graph, i.e. $\mathcal{C}^N(P,Q) \geq \mathcal{C}(P,Q)$, where P and Q are paths consisting of equal number of edges, emanating from a common node X and connecting two distinct nodes Y and Z.*

(a) (b) (c)

Figure 2. Effect of the number k of observations used to calculate the posterior. For the specific example only grayscale information based on the human markings was used. (a) $k = 6$ (b) $k = 12$ (c) Two-level hierarchical approach with $k = 6$.

The above proposition establishes that the contrast between paths when using the Markov Chain criterion to define the weights will in general be larger than that when a Bayesian criterion similar to that of Equation 1 is used.

Proposition 2. *The contrast increases as the number k of the terms of the summations of the edge weights increases.*

The above proposition shows that by increasing the number of observations associated with each boundary point we can enhance the contrast of the weights of the paths and thus extract geometrically complex boundaries. Figures 2(a-b) show the effect of this parameter; they display the largest part of the boundary that could be extracted given initial human markings. Note how using more observations for the calculation of the posteriors enables the method to extract larger part of the boundary (Figure 2(b)).

Although theoretically, increasing the number of observations should enable such a method to to cope with complex objects, this is not a practically viable solution. The computations of the transition probabilities for very small numbers become unstable mainly because of the normalization factor of Equation 5. This results in the extraction of noisy contours or contours that do not follow closely the object boundary, as can be seen in Figure 2. Our solution to the problem is a hierarchical statistical formulation which we present in the following section.

6. Hierarchical Modeling

Hierarchies are a popular representation scheme in computer vision. Broadly speaking the goal is to represent a domain at different levels of abstraction for the purpose of computational efficiency, stability of results as well as incorporation of higher order dependencies in Markov models. They are either built in a top-down or

bottom-up fashion. In our case we need a hierarchical modeling scheme primarily to enhance the contrast of weighted paths in case we cannot sufficiently discriminate in a statistical sense between the two sides of the boundary.

For the particular problem we can either build hierarchies in the *image domain* or the *contour domain*. A hierarchy in the image domain would result in a multiscale approach where each pixel of a coarser scale corresponds to several pixels (usually 4) of the finer scale. This is a common technique when using Markov Random fields for image segmentation [3]. For our modeling, this type of hierarchy would be useful mainly because local variations of the image would be smoothed out as well as for computational efficiency. However we would run into two significant problems. First, the optimization would be performed in a top-down fashion, that is, the boundary finding would proceed from coarse to fine levels. This strategy is difficult to implement for geometrically complex boundaries especially for hierarchies consisting of a large number of levels. Second, the contrast of the paths cannot be enhanced significantly. To see why this is the case consider the simple scenario of a two-level hierarchy consisting of a coarse level and a fine level. The fine level corresponds to the original image, and each pixel of the coarse level corresponds to 4 pixels of the fine level. Assume also that the grayscale or color of the interior and the exterior of the object are normally distributed. Then, the weights of the graph at the coarse scale are equal to the weights of the graph at the fine scale divided by two. From Observation 1 the optima of the objective functions at the coarse and the fine level will be the same.

For these reasons, we have developed a statistical hierarchical model for the contour domain. Our hierarchical model is based on ideas of the Hierarchical Hidden Markov Model (HHMM) presented in [10]. Figures 3 and 4 illustrate our model. The model consists of a set of states C_i^l. The superscript l denotes the level of the hierarchy the state belongs to. $l = 0$ denotes the finest level and $l = d$ denotes the coarsest level. At the finest level, C_i^0 corresponds to each image pixel. Each state C_i^l for $l > 0$ consists of a set of states C_i^{l-1} of level $i-1$. In a hierarchical Markov Chain model, there are two kinds of transition probabilities. There are *vertical* transition probabilities between states of different abstraction levels and there are *horizontal* transition probabilities between states of the same level. The vertical probabilities are independent of the horizontal probabilities and they individually sum up to 1. In our case, each state of level $l > 0$ is connected with a single state of level $l-1$ and thus the vertical transition probabilities are equal to 1. Figure 3 shows a contour and its two-level hierarchical representation. The dashed arrows connecting the states from the fine to the coarse level are not assigned a probability; they merely show the flow between the levels of the hierarchy. Figure 4 shows the hierarchical contour model embedded in the image (the dashed arrows are omitted).

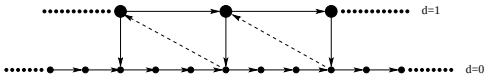


Figure 3. Two-level hierarchical modeling of a contour.

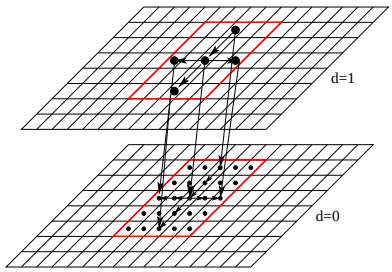


Figure 4. Hierarchical modeling of image in the contour domain.

The horizontal transition probabilities between the states of levels $l > 0$ are calculated in a manner similar to Equations 4 and 5 in Section 4. More specifically, we have:

$$P(C_i^l | C_{i-1}^l, D]) = \frac{P(C_i^l, C_{i-1}^l | D)}{P(C_{i-1}^l | D)} \tag{9}$$

$$P(C_{i-1}^l | D) = \sum_{C_i^l \in \mathcal{N}(C_{i-1}^l)} P(C_i^l, C_{i-1}^l | D) \tag{10}$$

where $\mathcal{N}(C_{i-1}^l)$ denotes the neighbors of C_{i-1}^l. Note that the neighbors of C_{i-1}^l for $l > 0$ are not adjacent in the 4- or 8-neighborhood sense. The objective function we eventually optimize consists of a summation of log transition probabilities between states at the coarsest level d, i.e. we maximize the quantity:

$$\sum_{\{C_i^d, C_{i-1}^d\} \in \mathbf{C}} \Big\{ \log P(D_{i-1} | C_i^d, C_{i-1}^d) + \log P(C_i^d, C_{i-1}^d) - \log P(C_{i-1}^d) \Big\} \tag{11}$$

Observe that the term $P(C_i^d, C_{i-1}^d)$ is the smoothing term between the states at the coarsest level. This term enables us to encode higher order prior geometric information in our objective function. The above criterion can be optimized using Dijkstra's algorithm at the coarsest level. Note that the graph at that level is "sparse" since each state corresponds to a number of states of the finer level.

There is of course the issue of how we group states at level $l-1$ to create states for level l as well as how

we compute the term $P(C_i^l, C_{i-1}^l|D)$ required for Equation 9. Recall that C_i^l, C_{i-1}^l are sets of states and thus this term can be calculate recursively from the finer levels. At the finest level $l = 0$ this term can be calculated from Equations 4 and 5. To find what states of level $l-1$ correspond to each state of level l we perform Dijkstra's algorithm in local windows at level $l-1$. At a given pixel X, we use Dijkstra's algorithm in a local window to identify the best paths that connect X with the pixels on the four sides of the window. We select a prespecified number of these paths with the additional constraint that the distance between their endpoints is greater than a prespecified threshold.

It is important to observe that the transitions of the hierarchical model will further discount the best path locally given the probabilities of the neighboring paths. Since we employ larger windows, this operation will not be so sensitive to noise as in the finest level. Furthermore, more complex boundaries can be captured and the method becomes less sensitive to the initial user input.

7. Results

Figures 5 and 6 show some of our results. **Only color information was used for the color images and grayscale for the grayscale images and no discontinuity information was used like gradient intensity.** The model for this type of information was built **based on the initial user markings only**. Specifically, the user initially clicked on two closely located points on the boundary of the object to be segmented such that the straight line that connected these two points roughly coincided with the object boundary. Small regions (windows of size 10×10) on the two sides of the line connecting the user points were used to estimate the parameters of two normal distributions. These distributions were used for the calculation of the transition probabilities. We did not employ any further statistical information based on the user's subsequent inputs.

As can be seen from Figure 5(d) this information is far from complete. This figure shows the Bayesian classification of the image pixels to two categories (object or background) using the models computed from the user markings in a very local area of the image. There are several parts of the object boundary where the pixels have not been classified correctly. Yet our criterion is able to extract the correct boundary. Figure 5(a) shows the results obtained using the fine level only. The user provided input 3 times. Figures 5(b-c) show the results obtained with the hierarchical extension of our method using 2 levels and a window for the fine level of size 11×11. The difference between the two results is the smoothing parameters used. Both cases required two user interactions.

Figure 6 show some results on additional images, where a single input was used for the boundary displayed. The same parameters (same with those of Figure 5(c)) were used for all these examples. From our experimentation we found that our method is not particularly sensitive to the values of the parameters. It is more sensitive to the initial model constructed since for these experiments it was the only feature used. In general it was preferable for the user to click on statistically well-separated regions. This gave more consistent results, since the initial model was more reliably estimated. Furthermore, the two-level hierarchical approach was more insensitive to the initial user markings than the one-level approach.

For the specific images the Bayesian criterion of Equation 1 required at least one more input than the Markov Chain criterion. In was also interesting to see that for the flower of Figure 5 it failed. The background of the flower transitions from green (leaf) to dark blue (water). The initial human input provided information for the green part only and thus the Bayesian criterion could not find the boundary of the flower on the blue part of the background without an excessive amount of human input.

8. Conclusions

We presented a statistical criterion capable of extracting geometrically complex boundaries with small amount of human input, even when the probabilities involved are computed based on incomplete information provided by the user. Our method relies on a Markov chain representation of the boundary and the transition probabilities result in a normalization factor that favors the locally best boundary segment. A hierarchical extension increases the robustness of our method as well as its ability to extract complex boundaries.

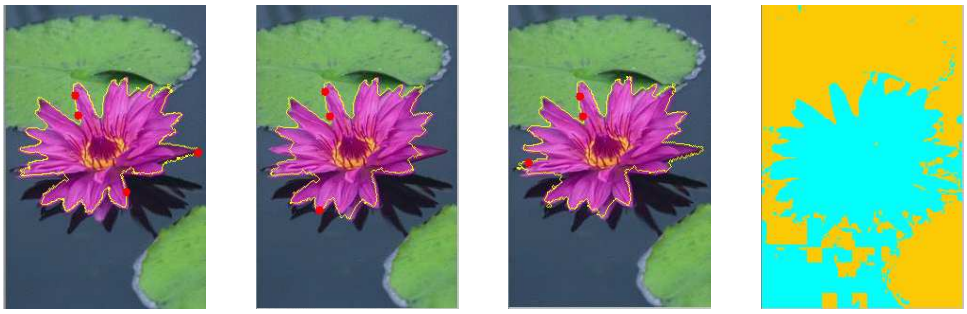
References

[1] A. Amini, T. Weymouth, and R. Jain. Using Dynamic Programming for Solving Variational Problems in Vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–867, 1990. 1, 2

[2] A. Blake, C. Rother, M. Brown, P. Perez, and P. Torr. Interactive Image Segmentation Using an Adaptive GMMRF Model. In *Proc. European Conference of Computer Vision*. LNCS, 2004. 2

[3] C. Bouman and M. Shapiro. A Multiscale Random Field Model for Bayesian Image Segmentation. *Transactions on Image Processing*, 3(2):162–177, 1994. 6

[4] Y. Boykov and M. Jolly. Interactive Graph Cuts for Optimal Boundary and Region Segmentation of ob-



(a) (b) (c) (d)

Figure 5. (a) Fine level was used only; 3 inputs were required by the user. (b-c) Two-level hierarchy was used; 2 inputs were required. The results were obtained with different smoothing parameters. (d) Bayesian classification of the object using the model constructed based on the initial user markings.

(a) (b) (c)

Figure 6. Results obtained using a two-level hierarchy. All examples required a single user input.

jects in N-D images. In *Proc. Int'l Conference on Computer Vision*. IEEE, 2001. 2

[5] Y. Boykov and V. Kolmogorov. Computing Geodesics and Minimal Surfaces via Graph Cuts. In *Proc. Computer Vision and Pattern Recognition*. IEEE, 1996. 2

[6] L. Cohen and R. Kimmel. Global minimum for active contour models: A minimal path approach. In *Proc. Computer Vision and Pattern Recognition*. IEEE, 1996. 1, 2

[7] T. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms*. McGraw Hill, 1990. 2

[8] J. Elder and R. Goldberg. Image Editing in the Contour Domain. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3), 2001. 2

[9] A. Falcao, J. Udupa, S. Samarasekera, and S. Sharma. User-Steered Image Segmentation Paradigms: Live Wire and Live Lane. *Graphical Models and Image Processing*, 60:233–260, 1998. 1, 2

[10] S. Fine, Y. Singer, and N. Tishby. The Hierarchical Hidden Markov Model: Analysis and Applications. *Machine Learning*, 32(1):41–62, 1998. 6

[11] D. Geiger, A. Gupta, L. Costa, and J. Vlotzos. Dynamic Programming for Detecting, Tracking, and Matching Deformable Contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(3):294–302, 1995. 1, 2

[12] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. In *Int'l Conference on Computer Vision*. IEEE, 1987. 1, 2

[13] E. Mortensen and W. Barrett. Interactive Segmentation with Intelligent Scissors. *Graphical Models and Image Processing*, 60(5), 1998. 1, 2

[14] E. Mortensen and W. Barrett. A Confidence Measure for Boundary Detection and Object Selection. In *Proc. Computer Vision and Pattern Recognition*. IEEE, 2001. 2

[15] E. Mortensen and J. Jia. A Bayesian Network Framework for RealTime Object Selection. In *Proc. Workshop on Perceptual Organization in Computer Vision*. IEEE, 2004. 1, 2

[16] W. Neuenschwander, P. Fua, L. Iverson, G. Szekely, and O. Kubler. Ziplock Snakes. *Int'l Journal of Computer Vision*, 25(3):191–201, 1997. 2

[17] A. Rosenfeld and A. C. Kak. *Digital Picture Processing*. Academic Press, 1982. 4

[18] C. Rother, V. Kolmogorov, and A. Blake. GrabCut - Interactive Foreground Extraction using Iterated Graph Cuts. In *Transactions on Graphics (SIGGRAPH)*. ACM, 2004. 2

[19] J. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge monographs on Applied and Computational Mathematics, 1999. 2

[20] N. Xu, R. Bansal, and N. Ahuja. Object Segmentation Using Graph Cuts Based Active Contours. In *Proc. Computer Vision and Pattern Recognition*. IEEE, 2003. 2