Fast Dichotomic Multiple Search Algorithm for Shortest Cirular Path

Martin de La Gorce and Nikos Paragios
MAS - Ecole Centrale de Paris
Grande Voie des Vignes, 92295 Châtenay-Malabry, France
http://vision.mas.ecp.fr
{martin.delagorce,nikos.paragios}@ecp.fr

Abstract

Circular shortest path algorithms for polar objects segmentation have been proposed in [1, 11, 10] to address discrete case and extended in [2] to the continuous domain for closed global optimal geodesic calculation. The best method up to date relies on a branch and bound approach and runs in $O(u^{1.6}v)$ on average while $O(u^2v)$ in worst case for a $u \times v$ discrete trellis warped in the direction of v. We propose an new algorithm called dichotomic multiple search (DMS) which finds the global minimum with a $O(ulog_2(u)v)$ worst case scenario complexity. Our algorithm relies on the fact that two minimal paths never cross more than once. This allows to sequentially partition the trellis in a dichotomic manner. Each computed circular minimal path with chosen starting point allows cutting the trellis into two sub trellis. The algorithm is then recursively applied on each sub trellis. Application to object segmentation is presented.

1 Introduction

Shortest Path computation is a common optimization problem which appears in a variety of domains and has emerged to be among the most prominent techniques in computer vision. In the context of image segmentation, shortest path algorithms such as the Dijkstra algorithm [6] or its continuous adaptation, the fast-marching [5], have been used successfully to geodesic active contour computation. Using a regular grid of the image size and a cost (or metric) inversely proportional to the image gradient intensity these methods allow to compute optimal boundary open curves between two specified extremity points in linear time with the number of pixels. Such methods have the strength of recovering the global optimum, that is not the case of the classical variational framework used for geodesic active contours [4, 8]. Closed curves can also be obtained given one single point lying on the curve [5] using fast-marching and a saddle point detection on the distance map. However in certain application we would like to compute closed curves specifying none of the points lying on the curve but a single point which would be within the interior of the curve after its computation. In the literature addressing such an limited scenario often consists in : 1) unwrapping the image about the specified point going from cartesian to polar coordinates 2) finding a 2π periodic shortest path in the unwrapped image, i.e. a path whose both extremities matches after re-wrapping. However regular shortest path algorithms do not allow to take into account the constrain for the path to be circular. In [12] several algorithms have been proposed. The Multiple Search Algorithm simply runs independently u single source shortest path algorithm for all point in the first column. This method ensure to find the global optimum but is very slow $O(u^2v)$ for an $u \times v$ image. The other algorithms proposed in [12] (Patching algorithm, Multiple back-tracking algorithm or hybrid algorithm) are linear with the image size ($O(uv)$) but do not guarantee to find the global optimum. More recently an efficient algorithm based on a branch and bound search as been proposed in [1]. It gives the global optimum with a mean complexity of $O(u^{1.6}v)$ for random noise images but its worst case complexity is of the same order as the Multiple Search Algorithm ($O(u^2v)$). This algorithm has been extended to the continuous domain in [2] in order to reduce metric error and to compute the so called globally optimal geodesic active contour. A rather different approach to the problem has also been proposed by the same authors in [3] extending the graph preflow-push method to the continuous domain. This method can be extended to higher dimension but is rather slow compared to the branch and bound method. Finally a new improvement to circular shortest path computation has been proposed in [7] in the context of semi-automatic segmentation. The method takes advantage of the fact that to minimal path crossing at two different points should share the trajectories between these two points. A clever choice of source point allows to stop the search after two pass in the image for most of the cases if u is small compared to v, resulting in a $O(uv)$ complexity In case it fails, the multiple search algorithm is applied as a fall-back method. This allows to lower the average complexity but still share the same worst case scenario complexity. Our algorithm takes advantage of the same property of minimal path. However we further developed the idea to propose a new algorithm with a lower worst case scenario complexity $O(ulog(u)v)$ even if u isn't small compared to v. The reminder of the paper is organized as follows: In section 2 we introduce the problem and present the prior art. In section 3 we introduce our approach, while complexity is discussed in section 4. Experimental results along with the conclusion are presented in section 5.

2 Problem statement and prior work

Let first formulate the problem we aim to solve. Similarly to [11], we consider an image of c size $u \times v$ with $I(i,j)$ corresponding to the cost of traversing the pixel at row i and column j. In the object segmentation context this image could be a decreasing function of the image gradient magnitude.

Denoting $Z_u = \{0,\dots,u-1\}$, $Z_v = \{0,\dots,v-1\}$, $y^- \equiv (y-1)\mathrm{modv}$ and $y^+ \equiv (y+1)\mathrm{modv}$, we define the set of circular path as follow

$$E = \{c \in {Z_u}^v, |c(y) - c(y^+))| \leq 1 \forall y \in Z_v\}$$

$c(y)$ corresponds to the raw position of the path in the y^{th} column. One can now only consider the image space and define the cost a particular circular path as follows :

$$L(c) = \sum_{x=1}^{v} I(c(y), y)$$

While the set of shortest circular paths is defined as :

$$C_{min} = argmin_{c \in E} L(c)$$

Note that this set might have more than one element since several circular path could have the same cost.

The simplest method to compute shortest circular path in the literature is the Multiple Search Algorithm (MSA) proposed in [11]. This method consist in running u independent single source shortest path computation and run in $O(u^2v)$. Let denote c_{xy} the shortest circular path passing through the point (x,y).

$$c_{xy} \in textargmin_{c \in E, c(y)=x} L(c)$$

This path can be computed for any (x,y) using the Single Source Shortest Path algorithm (see algorithm 1) which is based on a on a dynamic programming method similar to the viterbi algorithm for hidden markov chains. We first compute the distance map from the point (x,y) by propagating distance column after column from left to right. The minimal path is obtained by backtracking from right to left. Note that we added a minor modification as we compute the distance only within an area bounded by two path c_{inf} and c_{sup}. This modification will be useful for our new algorithm. If we ignore the two bounding paths, this leads to a $O(uv)$ complexity for the single source shortest path computation, otherwise the complexity is proportional to the surface of the area of interest i.e. $\sum_{j=1}^{v} c_{sup}(j) - c_{inf}(j)$.

Algorithm 1 - SSSP$(I, x, y, c_{sup}, c_{inf})$

INPUT:
I image specifying the weights
c_{sup} and c_{inf} are the path bounding the area of interest.
OUPUT:
c_{xy}: one of the shortest circular path with fixed source point in the given area
Begin
// distance map and predecessor flag from the point(x,y) *:*
  $D(i,y) \leftarrow \infty \, \forall i \in Z_u$
  $D(x,y) \leftarrow I(x,y)$
  $\forall (i,j) \in Z_u \times Z_v \; j \neq y, \; c_{inf}(j) \leq i \leq c_{sup}(j)$:
  $P(i,j) \in textargmin_{k \in \{i, i\pm1\} \cap [c_{inf}(j), c_{sup}(j)]} D(k, j^-)$
  $D(i,j) \leftarrow I(i,j) + D(P(i,j), j^-)$
// backtracking :
  $c_{xy}(y) \leftarrow x$
  $\forall j \in Z_v \backslash \{y\}$: $c_{xy}(j) \leftarrow P(c_{xy}(j^+), j^+)$
End

The MSA simply computes c_{xy} for $y = 0$ and all $x \in Z_u$ and keep the shortest. This naturally lead to a $O(u^2v)$ complexity.

The best algorithm up to date to compute circular minimal path has been proposed by [1] and is based on a branch and bound approach. Rather than using a single point source for shortest path computation as we do in the SSSP algorithm, the method uses a whole set of point $S = \{(s_{min}, 0), \dots, (s_{max}, 0)\}$ in the first column . The shortest path starting in S and ending in $\{(s_{min}-1, v), \dots, (s_{max}+1, v)\}$ gives a lower bound of all circular paths going through S. This allows to avoid testing all position in the first column (as does MSA) while pruning some positions $(x, 0) \in S$: Suppose that the best circular path found up to a given time has the length α. In case the lower bound associated to a set of position in the first column is higher than α, we can safely prune all position in this set. The overall method works as follow : The set of point in the first column is recursively split until all position are pruned but one, which is the shortest circular path. The worst complexity of the algorithm is $O(u^2v)$ and the average complexity is $O(u^{1.6}v)$. We propose an algorithm running in $O(ulog_2(u)v)$ in worst case.

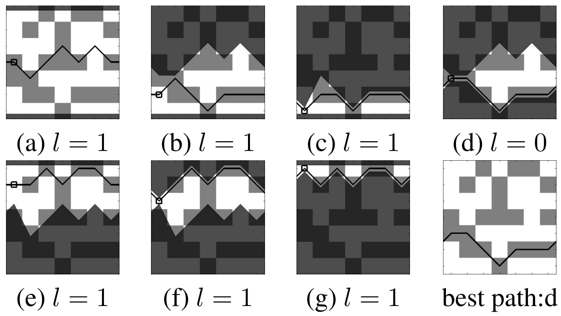

Figure 1. *Tree nodes and cost* $l = L(c_{xy})$.

3 A Dichomotic Multiple Search Algorithm

It is possible to accelerate the multiple search algorithm when taking into account certain properties of circular shortest paths. For any $(x, y) \in Z_u \times Z_v$ there is a circular minimal path $c_{min} \in C_{min}$ which stay either above or under c_{xy}, otherwise c_{xy} is itself a shortest circular path.

This means that at least one of the following conditions is verified :

1. $\exists\, c \in C_{min},\ c(j) \leq c_{xy}(j)\ \forall j \in Z_v,\ c(y) < x$
2. $\exists\, c \in C_{min},\ c(j) \geq c_{xy}(j)\ \forall j \in Z_v,\ c(y) > x$
3. $c_{xy} \in C_{min}$

This remark leads to a dichotomic algorithm called Dichotomic Multiple Search algorithm (see algorithm 2). Each time we compute a path c_{xy}, we can split the image into two areas respectively above and under c_{cy}. The minimal circular path will then be retrieved on each area independently. The main advantage is to lower the size of the area on which we compute the distance map after each single source path computation. This decrease the complexity of the algorithm to $ulog_2(u)v$ as it will be proven in the next section. The search sequence can be interpreted as a binary tree exploration where each node corresponds to the split of an area into to sub areas. Choosing carefully the source of the circular path for each node ensure a maximal tree depth of $\lceil log_2(u+1) \rceil$. In order to illustrate how he search is done we consider a 7×7 binary weight image. The sequence is illustrated in figure 1. The black square is the source point (x,y) . The black line correspond to the found shortest path c_{xy} and the darker area correspond to the exclude area in the shortest path computation. The sequence can be interpreted as a depth first binary tree exploration with depth 3: Area in a) is split into b) and e) ; b) into c) and d); e) into f) and g).

4 Time Complexity Analysis

The estimate of the worst case scenario complexity of our algorithm is quite simple to obtain. Interpreting the search as a binary tree exploration, the total computational cost is the sum of the cost of all nodes. Let us consider

---

Algorithm 2 - DMS(I, c_{sup}, c_{inf})

INPUT:
I image specifying the weights
c_{sup} and c_{inf} are the paths bounding the area of interest on which we aim to find the minimal circular path.
OUPUT:
c_{min}: one of the shortest circular path in the given area:
$E' = \{c \in E, c_{inf} \leq c \leq c_{sup}\}$
$c_{min} \in textargmin_{c \in E'} L(c)$
Begin
// *choice of the source point* (x, y) *(minimal width)*
$y \leftarrow textargmin_j(c_{sup}(j) - c_{inf}(j))$
if $(c_{sup}(y) < c_{inf}(y))$ return $\emptyset$
$x \leftarrow \lfloor \frac{1}{2}(c_{inf}(y) + c_{sup}(y)) \rfloor$
// *computation of* c_{xy} *using single source algorithm*
$c_{xy} \leftarrow SSSP(I, x, y, c_{sup}, c_{inf})$
$c_{min} \leftarrow c_{xy}$
// *shortest circular path above :*
$c \leftarrow c_{xy}, c(y) \leftarrow x + 1$
$c_t \leftarrow DMS(I, c_{sup}, c)$
if $(L(c_t) < L(c_{min}))$ then $c_{min} \leftarrow c_t$
// *shortest circular path below :*
$c \leftarrow c_{xy}, c(y) \leftarrow x - 1$
$c_t \leftarrow DMS(I, c, c_{inf})$
if $(L(c_t) < L(c_{min}))$ then $c_{min} \leftarrow c_t$
End

---

the cost of all nodes at a given depth p of the tree. At this given depth the image is split into 2^p areas. These areas overlap on their boundary and their union equals the entire image area. During the computation of all distance maps, pixels on the boundary are considered twice (once for the area above and once for the area below).

The total cost of distance map computation for the given depth is then $uv + (2^p - 1)v$. The choice of the coordinate (x, y) from which we compute the circular path c_{xy} ensures that the depth doesn't exceed $\lceil log_2(u + 1) \rceil$. Indeed our choice of source for the $SSSP$ algorithm ensure that the minimal width of the sub-areas $(min_i(c_{xy}(i) - c_{inf}(i))$ and $min_i(c_{sup}(i) - c_{xy}(i)))$ are smaller the half the minimal width of the area $(min_i(c_{sup}(i) - c_{inf}(i))$. It follows that the worst case complexity is :

$$\begin{aligned} C(u, v) &= \sum_{p=1}^{\lceil log_2(u+1) \rceil} [uv + (2^p - 1)v] \\ &= \lceil log_2(u + 1) \rceil uv + v(2u - 2 - log_2(u)) \\ &< (\lceil (log_2(u + 1) \rceil + 2)uv \end{aligned}$$

In order to estimate the average time complexity we tested our algorithm on random images. Image size ranging from 500×500 to 2500×2000 were considered with pixel values drawn independently from the uniform random distribution on $[0, 1]$. The complexity for each image size is estimated using 50 samples. The complexity is approximated as the number of additions, multiplications or comparisons. The mean complexity (curves) and standard deviation (vertical lines) for different image sizes are shown in fig 2. Note that as expected the complexity increases with u and v. Our algorithm compares favorably to the branch and bound algorithm on noise images, and is about four times faster on a 2500×1500 image. The standard deviation of our algorithm is too small to be visible on the plot for our algorithm. The variance is much bigger for the B&B method. This means that our method complexity is less sensitive to the image content. This can be of interest if we want to have a good control on the algorithm complexity.

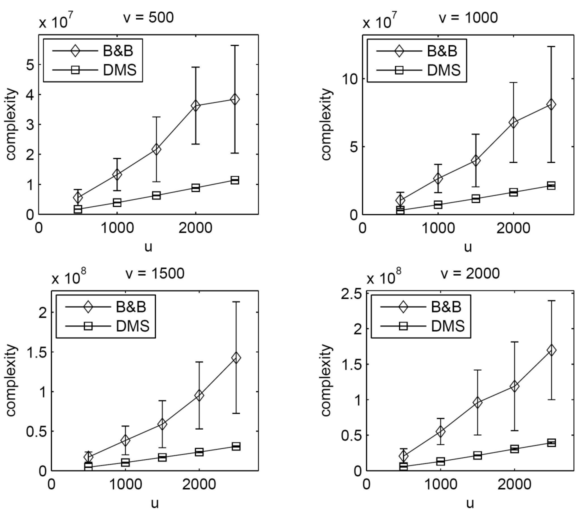

Figure 2. *Mean complexity curves and standard deviation for of B&B and DMS with varying images size*

5 Discussion

In this paper a new minimal circular path algorithm has been proposed. It ensure to find the global optimum with a worst case complexity of $O(ulog_2(u)v)$. The algorithm recursively computes the circular shortest path with specified source and split the image area into sub-areas. Circular shortest paths search can be a valuable element to image segmentation with a number of applications, like iris extraction, segmentation of anatomical and biological structures, etc. In order to provide a simple demonstration of the whole process we segment the iris on an eye's image [see fig 3]. We selected roughly the center of the pupil. The polar unwrapping U about this center point is computed. The shortest circular path is computed using cost function $1/(1 + |dU/dx|)$ which decrease as the edge intensity increase. The circular path is finally back-projected onto the original image.

The proposed algorithm outperforms the previous ones when considering worth case complexity or mean complexity with white noise images. Because it works on discrete lattice it may lead to stairs effect due to metric errors. Following the approach proposed in [2] current effort is made to extend our algorithm to the continuous domain and compute shortest path using the heuristically driven fast-marching method [9]. This would lead to a fast computation of the globally optimal geodesic active contour.

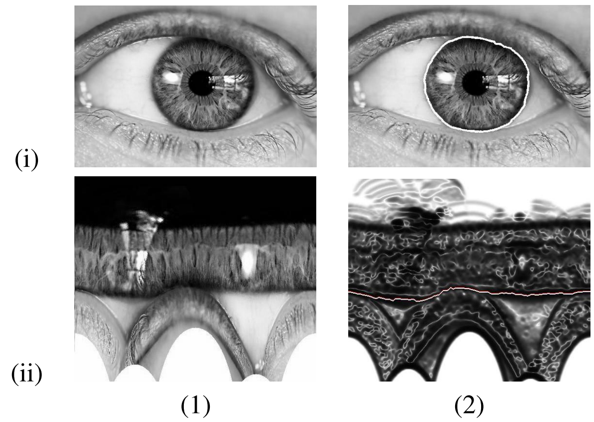

Figure 3. *Pupil segmentation: (i.1)eye image (i.2) result on original image (ii.1)unwrapped image (ii.2)cost function and minimal circular path*

References

[1] B. Appleton and C. Sun. Circular shortest paths by branch and bound. *Pat.Rec.*, 36(11):2513–2520, 2003.
[2] B. Appleton and H. Talbot. Globally optimal geodesic active contours. *JMIV*, 23(1):67–86, 2005.
[3] B. Appleton and H. Talbot. Globally minimal surfaces by continuous maximal flows. *IEEE PAMI*, 28(1):106–118, 2006.
[4] V. Caselles, R. Kimmel, and G. Sapiro. Geodesic active contours. In *ICCV*, pages 694–699, 1995.
[5] L. D. Cohen and R. Kimmel. Global minimum for active contour models: A minimal path approach. *IJCV*, 24(1):57–78, 1997.
[6] E. W. Dijkstra. A note on two problems in connexion with graphs. In *Num. Math.*, pages 269–271, 1959.
[7] D. Farin, M. Pfeffer, P. H. N. de With, and W. Effelsberg. Corrisor scissors: A semi-automatic segmentation tool employing minimum-cost circular paths. In *ICIP*, 2004.
[8] R. Goldenberg, R. Kimmel, E. Rivlin, and M. Rudzsky. Fast geodesic active contours. In *Scale-Space*, pages 34–45, 1999.
[9] G. Peyré and L. D. Cohen. Heuristically driven front propagation for geodesic paths extraction. In *VLSM*, pages 173–185, 2005.
[10] C. Sun and B. Appleton. Multiple paths extraction in images using a constrained expanded trellis. *IEEE PAMI*, 27(11):1923 1933, 2005.
[11] C. Sun and S. Pallottino. Circular shortest path on regular grids. In *ACCV*, pages 852–857, 2002.
[12] C. Sun and S. Pallottino. Circular shortest path in images. *Pat.Rec.*, 36(3):711–721, 2003.