Edge Detection as Finding the Minimum Cost Path in a Graph

Article · January 1998

DOI: 10.1007/978-3-7091-6487-7_4

Jean-Marc Salotti
Bordeaux Institut National Polytechnique

Special Issue on Graph Based Representations in Pattern Recognition, J.M. Jolion and W.G. Kropatsch (eds.), Springer Wien NewYork, Computing, vol. 12, p. 33-41, 1998.

Edge detection as finding the minimum cost path in a graph

Marc Salotti

Adr.: Faculté des Sciences et Techniques

Quartier Grossetti, BP 52, 20250 Corté (FRANCE)

Tel.: (33) 04 95 46 10 83; Fax: (33) 04 95 61 05 51; e-mail: salotti@univ-corse.fr

Abstract

We present new ideas to perform contour following using heuristic search strategies. On the one hand, we show that it is interesting to develop the entire graph of the search without defining the goal node: simple graph tools can be used to find closed boundaries or to determine the junctions. On the other hand, we show that a cost function with exponential curvature can favor depth first strategies, preserving the minimum cost path condition.

1 Introduction

Heuristic search strategies have been developed for many applications of artificial intelligence [9]. Martelli applied the A Algorithm to boundary detection in 1972 [6]. Other similar approaches have been proposed, most of them for the detection of boundaries in biomedical applications [5], [12] (an excellent overview is proposed by Ballard and Brown [2]). The main problem lies in the adaptability of the search. In particular, it is difficult to avoid the exploration of small undesirable paths.

We present in the next part the basic idea of heuristic search strategies. Then, we discuss the problem of the representation and we show that it is interesting to develop the entire graph of the search.. We also propose a cost function that enables depth first strategies. Finally, some results are presented to illustrate good and bad points of the method.

2 Heuristic Search Strategies

2.1 The Basic Algorithm

In 1972, Martelli showed that the problem of boundary detection can be cast to the problem of finding the minimal cost path in a weighted and directed graph, with positive costs[6]. The basic search is given by Nilsson 's algorithm [8] :

1. Expand the start node : put the successors on a list called OPEN with pointers back to the start node.
2. Remove the node Pi of minimum path cost from OPEN. If Pi is a goal node, then stop. Trace back through pointers to find the optimal path. If OPEN is empty, then fail.
3. Else expand node Pi, adding successors on OPEN with the associated path cost, or eventually updating them, with pointers back to Pi. Go to step 2.

An example is presented figure 1 and 2. It is interesting to notice that node E is not met during the search, because the cost from A to B is prohibitive. This property is important because it allows us to build dynamically the graph and to take only the best nodes into account.
The A Algorithm is an improvement of the search when the distance to the goal node can be estimated. If g is the cost of the path from the starting node to a node N, and h is an estimation of the remaining cost between N and the goal node, then f = g + h is a global cost function that can be used instead of g. The solution is guaranteed to be the minimum cost path, provided that h is inferior or at least equal to the exact distance remaining to reach the goal node.

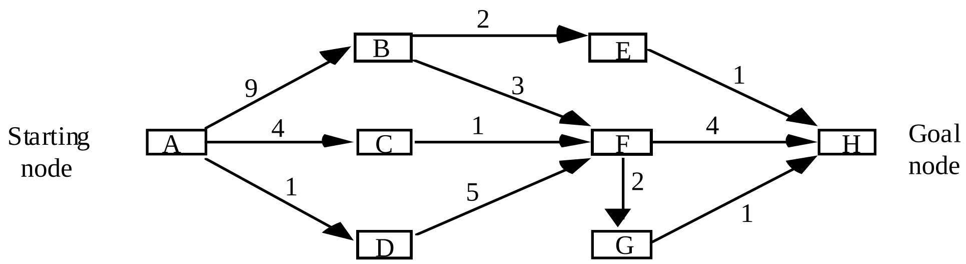

Figure 1: Example of weighted graph.

Step	1	2	3	4	5	6
Open nodes	(A 0)	(B 9)(C 4) (D 1)	(B 9) (C 4) (F 6)	(B 9) (F 5)!!	(B 9) (G 7) (H 9)	(B 9) (H 8)!!
Choice	A	D	C	F	G	H
Best path	A	AD	AC	ACF	ACFG	**ACFGH**

Figure 2: Evolution of the search applied to the graph presented figure 1.

2.2 Representations

Different representations can be chosen: for instance, Martelli proposed to map the nodes to the frontier between two pixels [7]. For convenient reasons, in particular the fact that the result can be displayed as an edge map, we prefer to match directly the nodes with the pixels of the image:
- The starting node can be selected among pixels with high gradient values.
- The successors of a node can be determined by systematically selecting the only three pixel-nodes that allow a curvature of the contour smaller than 45°.
- A critical point is the definition of the ending node. If closed boundaries have to be found, then it is possible to set the starting node as ending node. In this case, if contextual information exists about the shape of objects, then h can be used to constrain the search using the A Algorithm. When finding closed boundaries is not imposed, it could be an interesting idea to develop all the paths until the open list falls to empty. If simple restrictions are made to avoid visiting all pixels (for instance, a node is created only if the local gradient value is superior to a threshold Gmin), the result is a graph of connected edge pixels. Such a graph is a powerful representation of edges: for instance, graph tools can be used to find closed boundaries (special

arcs must be added when mark pixels are encountered) or to determine junctions. Moreover, the entire image can be scanned and several graphs can be built from different seeds, pixels-nodes being marked after each search. In this case, an edge map is available and there is a graph for each connected set of edge pixels. In terms of computing time, it is not so bad, providing that an image of the pointers to the nodes is dynamically built to speed up the search in the open list.

2.3 The local cost function

Let C(n1,n2) be the cost of the arc between two nodes n1, n2, and let h(n) be the estimated cost to reach the goal node from a node n.
Then, the global cost f(n) of the path going through a node n is given by (1) :

$$f(n) = \left(\sum_{path} C(n_i,n_{i+1})\right) + h(n) \qquad (1)$$

Martelli suggested to set h(n) to 0 and to use as local cost an expression roughly equivalent to the following [7] :

$$C(n1,n2) = M - \text{gradient}(n2), \text{ where } M = \max_x \{\text{gradient}(x)\} \qquad (2)$$

The problem of the heuristic search applied to edge detection is that the graph is dynamically built and it may become very large if using heuristics like expression (2): long paths are quickly expensive and small undesirable paths are explored. Several ideas have been proposed to solve the problem. The selection of the next node in the OPEN list can be made with a depth-first strategy or using a rating function [3], [10], [12]. Lester and *al* suggest to take the maximum cost arc of the path instead of the sum. The advantage is that the cost of the path does not grow continuously with depth, so that good paths can be followed for a long time [5]. Since the cost of the path necessary increases with depth if the costs are positive, Ashkar and Modestino proposed a cost function that takes negative values if the arc has a good evaluation [1]. However, if some interesting ideas have been proposed, the property of finding the path with minimum cost is not always guaranteed.

3. Propositions for a suitable cost function

3.1 Exponential curvature

What should be the shape of the local cost function C ? Let us assume for the moment that the gradient value is the only parameter of the function and that no information is available on the objects to be found. The main idea of the search is that a depth first strategy should be used as long as the gradient value is high. It means that the cost function should be small enough to make the cost of the entire path smaller than a single cost corresponding to a lower gradient value. Mathematically, it can be expressed as follows :
Let S be the current path $\{n_1,n_2 \dots n_k\}$ to reach n_k from the starting node,

$$\forall\, n \notin S,\ \forall\, i \in \{1..k\},\ Grad(n) < Grad(n_i) \dashrightarrow \left(\sum_{j=1}^{k-1} C(n_j,n_{j+1})\right) < C(n_i,n) \qquad (3)$$

Since only positive costs are to be used, this expression is always true if and only if C is set to 0 for gradient values superior to grad(n) and to a constant different from 0 otherwise. However, it is important to notice that there is a sum of only k terms. k is simply the depth of the path and in fact the length of the contour. From a practical standpoint, the length of any contour rarely stands 100 or 200 pixels. In this case, it is possible to choose a function that quickly decreases, so that condition (3) is roughly respected. We propose to use a function with exponential curvature, as illustrated figure 3. Let n1 and n2 be two connected pixels, let x be the gradient value on pixel n2 and Gmin a threshold to reject small gradient values, then :

If $x \leq Gmin$, $C(n1,n2) = \infty$ (or the arc is simply not created).

If $x > Gmin$, $C(n1,n2) = -1+e^{c/(x-Gmin)^2}$ (c is a constant >0 used for regularization)

Figure 3: A suitable cost function.

For instance, with Gmin set to 10, and c set to 1, a path of 40 pixels with all gradient values equal to 200 has a smaller cost than a path of 2 pixels with gradient values equal to 50. The search is in fact almost depth first. What is the role of constant c ? Its role is to determine the exact shape of the cost function, so that a depth first strategy is more or less favored. When c tends to 0, the cost function tends to a step function and condition (3) tends to be respected with higher probability. Moreover, when c tends to 0, because of the curvature, the maximum arc cost of a path can be taken as a rough approximation of the cost of the entire path. In fact, this approach is similar to the one of Lester et al, except that ours guarantees the minimum cost of the path [5]. In a previous work, we proposed a slightly different function with Gaussian curvature [11]. The function presented here is more appropriate because it is expected to tend to infinity when the gradient tends to Gmin.

3.2 Generalization

Since an exponential curvature is suitable to favor a depth first strategy, a generalization can be made for an arbitrary number of features.

$$C(n1,n2) = \Sigma_k \alpha_k . e^{1/x_k^2} \qquad (3)$$

For k features, x_k is an expression of feature k, and α_k determines the associated weight, such that $\Sigma_k \alpha_k = 1$. Examples of interesting features can be found in previous works [2], [11].

4 Results and conclusion

Before applying our method, a gradient map is obtained with Deriche operator. We propose to end the search when the Open list falls to empty and to use the function C previously described. The results are displayed below: Image "res1" shows the pixel-nodes of a single graph (the seed was a pixel with high gradient value in the middle of the image). Image "res2" shows the pixel-nodes after removing branches that are not part of cycles. Edges are rather thin because each node has only one predecessor (the one that makes the minimum cost path) and no cycle can be found with pixels located in the border of the edge. Image "res3" is a zoom of "res1": the pixels of the graph are marked but only the darkest ones belong to cycles and the junctions are correctly located.

Our conclusion is that graphs and heuristic search strategies provide rich and powerful tools for edge detection and representation. We are intended to carry on working on this technique and to present more interesting results in a near future.

References

[1] Ashkar G.P., Modestino J.W. "The contour extraction problem with biomedical applications", *CGIP*, vol. 7, 331-355, 1978.

[2] Ballard D. H., Brown C. "Computer Vision", *Prentice Hall Inc*., Englewood Cliffs, New Jersey, USA, 131-137, 1982.

[3] Ballard D.H., Slansky J. "A ladder-structured decision tree for recognizing tumors in chest radiographs", *IEEE Trans. Computers*, vol. 25, 503-513, 1976.

[4] Deriche R. "Using Canny's criteria to derive a recursively implemented optimal edge detector", *International Journal of Computer Vision*, vol. 1, 167-187, 1987.

[5] Lester J. M., Williams H.A., Weintraub B.A., Brenner J.F., "Two graph searching techniques for boundary finding in white blood cell images", *Comp. in Biol. and Med*., vol. 8, 293-308, 1978.

[6] Martelli A. "Edge detection using heuristic search methods", CGIP, vol. 1, n°2, 169-182, 1972.

[7] Martelli A. "An application of heuristic search methods to edge and contour detection", *Comm. of the ACM*, vol. 19, 73-83, 1976.

[8] Nilsson N.J. "Principles of artificial intelligence", Palo Alto, CA:Tioga, 1980.

[9] Pearl J. "Heuristics: intelligent search strategies for computer problem solving*", Addisson-Wesley*, 1984.

[10] Persoon E. "A new edge detection algorithm and its applications in picture processing", *CGIP*, vol. 5, n°4, 425-446, 1976.

[11] Salotti M., Hatimi M. "A new heuristic search for boundary detection", *Lecture Notes in Comp.Sci. 974*, C. Braccini, L. de Floriani, G. Vernazza Eds, 381-386, septembre 1995.

Special Issue on Graph Based Representations in Pattern Recognition, J.M. Jolion and W.G. Kropatsch (eds.), Springer Wien NewYork, Computing, vol. 12, p. 33-41, 1998.

[12] Weschler H., Sklansky J. "Finding the rib cage in chest radiographs", *Pattern Recognition*, vol. 9, 21-30, 1977.