Proceedings of the 45th IEEE Conference on Decision & Control
Manchester Grand Hyatt Hotel
San Diego, CA, USA, December 13-15, 2006

Distributed algorithms for polygonal approximation of convex contours

Sara Susca Sonia Martínez Francesco Bullo

Abstract— **We propose algorithms that compute polygon approximations for convex contours. This geometric problem is relevant in interpolation theory, data compression, and has potential applications in robotic sensor networks. The algorithms are based on simple feedback ideas, on limited nearest-neighbor information, and amount to gradient descent laws for appropriate cost functions. The approximations are based on intuitive performance metrics, such as the area of the inner, outer, and "outer minus inner" approximating polygons.**

I. Introduction

In applications such as monitoring of environmental processes it is important to be able to approximate the contour of the region of interest. For some specific monitoring tasks such as the containment of a region of interest (e.g., an oil spill) or the specification of an initial condition for the prediction of certain environmental phenomenon it is meaningful to obtain contour approximations that resemble as much as possible the region to be determined. Finding efficient or optimal approximating polygons is also relevant in other applications like solving interpolation problems or data compression. It is useful in fact to be able to represent a contour for which no concise mathematical expression is known by only using a few points. It turns out that constructing an optimal polygonal approximation of a contour has been a research subject for mathematicians and engineers across the last three centuries. Still interesting problems continue to remain unsolved especially for the general setting of non-convex bodies. Arguably, the extension of polygonal approximation to non-convex and time-varying contours will provide a novel challenging problem in boundary estimation, tracking, and surveillance.

In this paper we investigate distributed algorithms enabling a robotic sensor network to generate an approximating polygon for any given convex planar contour. As a key modeling assumption, the nodes of the sensor network are the vertices of the approximating polygon. We require that the approximating polygon minimizes a certain meaningful metric. Boundary estimation and tracking is also a relevant problem in computer vision; the so-called "snake algorithms" were introduced in the seminal paper [1]. Some references on the boundary estimation problem for robotic sensor networks include [2], [3], [4], [5]. A different and interesting application of boundary estimation and tracking is presented in [6]; here a feedback law is proposed to steer the tip of an atomic force microscope so that the imaging time is drastically reduced.

As pointed out by the authors in [7], in the XIX century it was known how to geometrically characterize the polygon, enclosed into a given convex body, that minimizes the area difference between itself and the enclosing convex body. On the other hand, the geometric characterization of a polygon, enclosing a given convex body, that again minimizes the difference of the areas is more complex and less intuitive; to the best of our knowledge, the earliest reference on this matter appeared only in 1949 by Trost, see [8]. Sometime in the XX century it was also proved that for a planar body the approximation error, for various useful metrics, goes to zero as C/n^2, where n is the number of vertices of the interpolating polygon. For example, in 1975 McClure and Vitale [9] give sharp estimate for the constant C using support functions. They also suggest two different methods to construct asymptotically efficient approximating polygon, even though both approaches are not suited to a distributed implementation. For a more detailed list of references we refer to the beautiful surveys [10] and [11]. Finally, a recent reference related to our work is [12].

Given n points on a convex contour, it is possible and natural to define an enclosed (i.e., inscribed) polygon and an enclosing (i.e., circumscribed) polygon to the contour. Here the faces of the enclosing polygon are subsets of the tangent lines to the convex contour. We adopt three geometrically-motivated error metrics that the approximating polygon can minimize. They are described as follows. The first two metrics we considered are the difference between the area enclosed in the contour and the following areas: the inner polygon area and the outer polygon area. The third metric is the sum of the previous two metrics. We derive the expressions, two of which are novel contributions of this paper, of the error metrics as functions of the vertex positions of the approximating polygon. We propose three feedback laws to dynamically construct the optimal approximating polygon using gradient descent. These feedback laws rely only on local information about the contour and about the immediate neighboring vertices. We analyze the dynamical system behavior of these feedback laws and present simulation results. Even though the algorithms are designed for smooth convex contours they can be extended to non-smooth convex contours. We also present discrete-time feedback laws that allows the nodes to reach locally optimal configurations for two of the metrics introduced.

The paper is organized as follows. In Section II we define some notation and the three performance metrics used through the paper. In Sections III and IV we present

This material is based upon work supported in part by ONR YIP Award N00014-03-1-0512 and NSF SENSORS Award IIS-0330008.

Sara Susca and Francesco Bullo are with the Center for Control, Dynamical Systems and Computation, University of California at Santa Barbara, sara@ece.ucsb.edu, bullo@engineering.ucsb.edu

Sonia Martínez is with the Mechanical and Aerospace Engineering Department, University of California at San Diego, soniamd@ucsd.edu

1-4244-0171-2/06/$20.00 ©2006 IEEE.

the continuous time gradient descent algorithms and their respective discrete time versions to construct the best inner and outer approximating polygon, while in Section V we present an algorithm to construct the polygon minimizing the outer minus inner area.

II. Notation and problem statement

Let $Q \subseteq \mathbb{R}^2$ be a bounded, convex body and ∂Q its boundary. Let $\mathbb{T} \subseteq \mathbb{R}^2$ denote the unit circle. We parametrize ∂Q by a map $\gamma\colon \mathbb{T} \to \partial Q$, and represent its signed curvature by $\kappa\colon \mathbb{T} \to \mathbb{R}$. We assume that κ remains positive as we traverse the curve γ in a counter-clockwise manner corresponding with the parameter s increasing. We alternatively use the notation $\mathbf{t}(s) \equiv \gamma'(s)$, $\forall s \in \mathbb{T}$, and define $\mathbf{n}\colon \mathbb{T} \to \mathbb{R}^2$ so that $\mathbf{n}(s)$ is the unit outward normal vector at $\gamma(s) \in \partial Q$ for all $s \in \mathbb{T}$. With a slight abuse of notation, we sometimes refer to the particular tangent and normal vectors at a point $p_i \in \partial Q$ as $\mathbf{t}_i$ and $\mathbf{n}_i$, $i \in \{1, \dots, N\}$.

Let $p_1, \dots, p_N$ be the positions of N agents constrained to be on ∂Q and let $\mathcal{P}(\mathbb{R}^2)$ denote the parts of $\mathbb{R}^2$; i.e. the collection of all subsets of $\mathbb{R}^2$. Since Q is a convex set, the set-valued map $P_I\colon (\partial Q)^N \to \mathcal{P}(\mathbb{R}^2)$, that assigns to a tuple $(z_1, \dots, z_n) \in (\partial Q)^N$ the polygon generated by the vertices $\{z_1, \dots, z_n\}$, satisfies $\operatorname{Im} P_I \subseteq \mathcal{P}(Q)$. In other words, $P_I(z_1, \dots, z_N)$ is the convex hull of the set $\{z_1, \dots, z_N\}$.

Let $\mathcal{H}$ denote the set-valued map $\mathcal{H}\colon \partial Q \to \mathcal{P}(\mathbb{R}^2)$ such that for any $p \in \partial Q$, with $p = \gamma(s)$, $\mathbf{n} = \mathbf{n}(s)$, for some $s \in \mathbb{T}$, we have $\mathcal{H}(p) = \{z \in \mathbb{R}^2 \mid (p - z) \cdot \mathbf{n} \leq 0\}$. That is, $\mathcal{H}(p)$ is the half-plane containing Q and with boundary given by the line passing through p and tangent to ∂Q, as shown in Figure 1. Now we can define the function $P_E\colon (\partial Q)^N \to \mathcal{P}(\mathbb{R}^2)$ as follows $P_E(z_1, \dots, z_N) = \mathcal{H}(z_1) \cap \dots \cap \mathcal{H}(z_N)$.

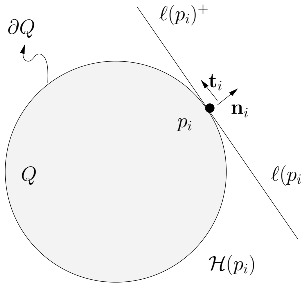
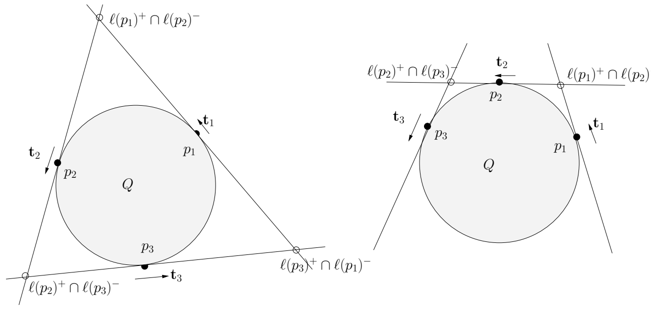


Fig. 1. $\mathcal{H}(p_i)$ and its boundary $\ell(p_i) = \ell(p_i)^+ \cup \ell(p_i)^-$.

The intersection of half-planes defines a convex region of the space $\mathbb{R}^2$ containing Q, but not in general a polygon. To generate a polygon some constraints on the half-planes have to be imposed. Let $\ell(p_i) = \partial\mathcal{H}(p_i)$ be the line that passes through p_i and is the boundary of $\mathcal{H}(p_i)$. Let us denote by $\ell(p_i)^+ = \{z \in \mathbb{R}^2 | z = p_i + \lambda \mathbf{t}_i\,,\ \lambda \geq 0\}$ and $\ell(p_i)^- = \{z \in \mathbb{R}^2 | z = p_i + \lambda \mathbf{t}_i\,,\ \lambda \leq 0\}$, then $\ell(p_i) = \ell(p_i)^+ \cup \ell(p_i)^-$. If the intersections $\ell(p_i)^+ \cap \ell(p_{i+1})^- \neq \emptyset$ for any consecutive nodes, then $P_E(p_1 \dots, p_N)$ defines an *exterior polygon*, as shown in Figure 2, whose edges lie in $\ell(p_i)$, $i \in \{1, \dots, N\}$, and contains both Q and $P_I(p_1, \dots, p_N)$.



$\ell(p_1)^+ \cap \ell(p_2)^-$
$\mathbf{t}_1$
p_1
$\mathbf{t}_2$
p_2
Q
p_3
$\mathbf{t}_3$
$\ell(p_2)^+ \cap \ell(p_3)^-$
$\ell(p_3)^+ \cap \ell(p_1)^-$
$\ell(p_2)^+ \cap \ell(p_3)^-$
$\mathbf{t}_2$
$\ell(p_1)^+ \cap \ell(p_2)^-$
p_2
$\mathbf{t}_3$
p_3
Q
p_1
$\mathbf{t}_1$

Fig. 2. From the left to the right: three points defining an outer polygon, three points not defining an outer polygon.

We quantify the approximation error of Q through different measures of area that we specify in the following. The *inner set approximation error metric* is defined as $\mathrm{E}_I(Q, P) = \text{Area}(Q \setminus P)$, where $P \subset Q$. Equivalently, the *outer set approximating error metric* is defined as $\mathrm{E}_O(Q, P) = \text{Area}(P \setminus Q)$ where $Q \subseteq P$. Now, given sets $P_1 \subseteq Q \subseteq P_2$ we can define the *symmetric difference error metric* as $\mathrm{E}_S(Q, P_1, P_2) = P_1 \Delta P_2 = (P_2 \setminus P_1) \cup (P_1 \setminus P_2)$.

III. Inner-polygon approximation algorithms

In the following sections we present distributed descent algorithms for the approximation of smooth convex bodies. The algorithms of this section are based on the interpolation error E_I.

In order to find a characterization of the configurations $\{p_1, \dots, p_N\} \subseteq \partial Q$ which minimize the inner set approximation error metric, observe that:

$$\mathrm{E}_I(Q, P_I(p_1, \dots, p_N)) = \text{Area}(Q) - \text{Area}(P_I(p_1, \dots, p_N))\,.$$

Assume that the set of points $\{p_1, \dots, p_N\}$ is ordered in a counter-clockwise direction.[1] Then, an expression for the $\text{Area}(P_I(p_1, \dots, p_N))$ can be obtained as

$$\text{Area}(\mathcal{P}_I(p_1, \dots, p_N)) = \frac{1}{2} \sum_{k=1}^{N} (x_k y_{k+1} - x_{k+1} y_k)\,, \quad (1)$$

where $p_k = (x_k, y_k)$ are the coordinates of the k^{th} point. The dynamical system defined as the gradient descent of E_I

$$\dot{p_i} = -\frac{\partial\, \mathrm{E}_I}{\partial p_i}\,, \qquad i \in \{1, \dots, N\}\,,$$

guarantees that the p_i converge to the set of critical points of E_I. This dynamical system can be rewritten as follows:

$$\begin{aligned} \dot{p_i} &= \text{proj}_{\,\partial Q} \left(\frac{\partial \text{Area}(P_I(p_1, \dots, p_N))}{\partial p_i} \right) \\ &= \left(\mathbf{t}_i \cdot \frac{\partial \text{Area}(P_I(p_1, \dots, p_N))}{\partial p_i} \right) \mathbf{t}_i\,, \end{aligned} \quad (2)$$

[1] In what follows we use the identification $0 \equiv N$ and $N + 1 \equiv 1$ for the indices $i \in \{1, \dots, N\}$.

where $\mathrm{proj}_{\partial Q}$ means the projection on the vector $\mathbf{t}_i$ tangent to the contour ∂Q at p_i. Substituting (1) in (2), we obtain

$$\begin{aligned}\dot{p}_i &= \mathrm{proj}_{\partial Q}\left(\left[\begin{array}{c}\frac{1}{2}(y_{i+1}-y_{i-1})\\ \frac{1}{2}(x_{i-1}-x_{i+1})\end{array}\right]\right)\\ &= \left(\frac{1}{2}\mathbf{t}_i^T\left[\begin{array}{c}y_{i+1}-y_{i-1}\\ x_{i-1}-x_{i+1}\end{array}\right]\right)\mathbf{t}_i,\quad i\in\{1,\dots,N\}.\end{aligned}\tag{3}$$

As it can be seen, $\dot{p}_i$ depends on p_{i-1}, p_{i+1}, and $\mathbf{t}_i$, $i \in \{1,\dots,N\}$. This requires that every agent has knowledge of the positions of its immediate clockwise and counter-clockwise neighbors and of the gradient of the contour at its position. Equation (2) (and hence (3)) describes the gradient flow of the area of the approximating polygon and it guarantees that the agents positions converge to a set of critical points so that

$$\mathbf{t}_i \cdot \left[\begin{array}{c}y_{i+1}-y_{i-1}\\ x_{i-1}-x_{i+1}\end{array}\right] = 0\,,\qquad \forall\, i\in\{1,\dots,N\}\,,\tag{4}$$

or, equivalently, $\mathbf{t}_i$ is parallel to $(p_{i+1}-p_{i-1})$, for $i \in \{1,\dots,N\}$. Unfortunately we can not say that every critical point is an extremum. Consider the situation where Q has the shape of an equilateral triangle with smoothed-out corners (see Figure 3). Despite the configuration shown in the figure satisfies condition (4), it is not a local minimum. As the figure shows, this is a saddle-point configuration, since we can grow or diminish the error by moving the nodes in appropriate ways.

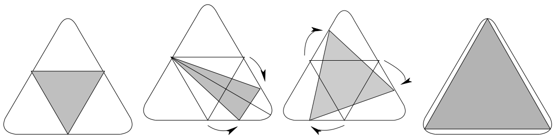


Fig. 3. From left to right: saddle point configuration, configuration that increases the error E_I, configuration that decreases the error E_I, configuration corresponding to a minimum error configuration.

The characterization (4) of critical points was already obtained in the XIX century according to [7]. The paper [7] additionally shows how the critical-point configurations satisfy the condition that points remain closer in regions of higher mean curvature, which is a desirable condition for shape representation. It is believed [10] that as the number of nodes increases, the type of configurations that satisfy (4) correspond only to global error minima.

Simulations inner-polygon approximation algorithm: Figure 4 shows the results of the implementation of the inner-polygon approximation algorithm. The eleven nodes are on the contour described by, for $\theta \in [0,1)$:

$$\gamma(\theta) = (2.1+\sin(2\pi\theta))\begin{pmatrix}\cos(2\pi\theta)\\ \sin(2\pi\theta)\end{pmatrix}.\tag{5}$$

A. Discrete-time inner-polygon approximation algorithms

Here we present two discrete-time versions of the algorithm of the previous section. The first algorithm exploits an a priori labelling of nodes and requires that each agent has available the position of its closest two neighbors along the contour. The second algorithm does not require an a-priori labelling of nodes but each agent is required to have some knowledge about a subarc of the contour.

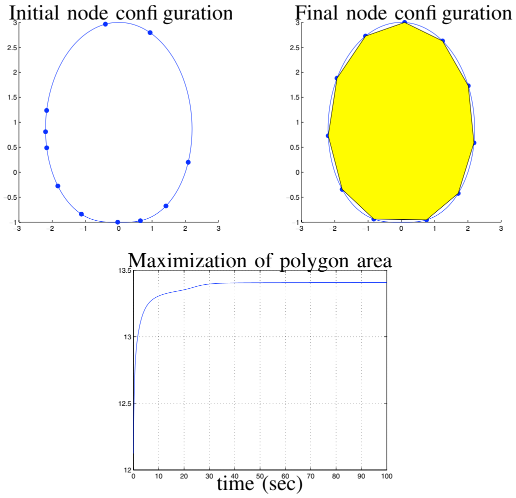


Fig. 4. Implementation of inner-polygon approximation algorithm

Algorithm 1. Assume each node p_i has knowledge about its own label number $i \in \{1,\dots,N\}$. At each discrete time instant that we index by $k \in \mathbb{N}$ we define:

$$p_i(k+1) = \begin{cases} q(p_{i-1}(k), p_{i+1}(k))\,, & \text{if } i \equiv k \mod N,\\ p_i(k), & \text{if } i \not\equiv k \mod N\,,\end{cases}\tag{6}$$

where $q(p_{i-1}(k), p_{i+1}(k))$ is the closest point to $p_i(k)$ on the arc in ∂Q from $p_{i-1}(k)$ to $p_{i+1}(k)$ such that its tangent at ∂Q is parallel to the line $\overline{p_{i-1}(k)p_{i+1}(k)}$.

Proposition 3.1: The dynamical system (6) is a descent algorithm for E_I and convergent to the set of critical points of E_I.

Proof: Let P_k be the area of the polygon generated by $p_1(k),\dots,p_N(k)$ and let i be congruent $\mod N$ with k. We have that $P_k = T_k + \bar{P}_k$, where T_k is the area of the triangle generated by $p_{i-1}(k)$, $p_i(k)$ and $p_{i+1}(k)$, and $\bar{P}_k$ is the area of the polygon generated by the complementary set of nodes. Since ∂Q is convex, it is easy to see that $T_k \leq \bar{T}_{k+1}$, where $\bar{T}_{k+1}$ is the area of the new triangle with vertices $p_{i-1}(k), p_i(k+1), p_{i+1}(k)$. In this way, one can obtain:

$$P_k = T_k + \bar{P}_k \leq \bar{T}_{k+1} + \bar{P}_k = P_{k+1}\,,$$

the area of the inner polygon is maximized and the error E_I is minimized. Clearly, only when a set of critical points is reached, the algorithm leaves the nodes stationary. ■

Remark 3.2: It is easy to envision extensions of Algorithm 1 to a setting where alternating but spanning sets of independent nodes alternate their motion. •

Algorithm 2. The following is an algorithm that does not require a labelling of agents, but requires knowledge about part of the contour. For each $k \in \mathbb{N}$ define:

$$p_i(k+1) = \begin{cases} q(p_{i-1}(k), p_{i+1}(k))\,, & p_i(k) \notin \mathcal{V}_i,\\ p_i(k), & \text{otherwise},\end{cases}\tag{7}$$

for $i \in \{1, \dots, N\}$, and where $\mathcal{V}_i$ is given by the union of certain arcs on ∂Q as we describe next. Let (z_1, z_2) denote the arc on ∂Q going from z_1 to z_2 in a counter-clockwise direction, for any $z_1,\ z_2 \in \partial Q$. Given $i \in \{1, \dots, N\}$, consider the arc (p_{i-2}, p_{i+2}) which contains the points $q(q(p_i, p_{i+2}), p_{i-1})$, $q(q(p_{i-2}, p_i), q(p_{i+2}, p_i))$ and $q(q(p_{i-2}, p_i), p_{i+1})$. Going from p_{i-2} towards p_{i+2} we can define arcs with $q(q(p_{i-2}, p_i), p_{i+1})$ and $q(p_{i-1}(k), p_{i+1}(k))$, with $q(q(p_i, p_{i+2}), p_{i-1})$ and $q(p_{i-1}(k), p_{i+1}(k))$, and with $q(q(p_{i-2}, p_i), q(p_{i+2}, p_i))$ and $q(p_{i-1}(k), p_{i+1}(k))$ as extremes respectively. With a slight abuse of notation, let us denote these arcs by $(q(q(p_{i-2}, p_i), p_{i+1}), q(p_{i-1}(k), p_{i+1}(k)))$, $(q(q(p_i, p_{i+2}), p_{i-1}), q(p_{i-1}(k), p_{i+1}(k)))$ and $(q(q(p_{i-2}, p_i), q(p_{i+2}, p_i)), q(p_{i-1}(k), p_{i+1}(k)))$. The set $\mathcal{V}_i$ is the union of these three arcs along the contour. Because of convexity, a node p_i can detect if it belongs to any of the above defined arcs by knowing the arc (p_{i-2}, p_{i+2}). Basically, the statement $p_i \notin \mathcal{V}_i$ is equivalent to the following statement: moving p_i towards the positions $q(p_{i-1}(k), p_{i+1}(k))$, $q(q(p_{i-2}, p_i), q(p_{i+2}, p_i))$, $q(q(p_i, p_{i+2}), p_{i-1})$ and $q(q(p_i, p_{i+2}), p_{i-1})$ requires that p_i moves in the same counter-clockwise or clockwise direction. It can be checked that $\mathcal{V}_i$ is in fact a connected arc along the contour ∂Q.

Here is our main analysis result in this section. We omit the proof for space reasons and we refer to a forthcoming technical report.

Theorem 3.3: The dynamical system (7) is a descent algorithm for E_I.

Remark 3.4: Stationary configurations of (7) are not necessarily critical points of E_I. A node p_i might become stuck at a position such that $p_i \in \mathcal{V}_i$ and $\mathbf{t}_i$ is not parallel to $\overline{p_{i-1}p_{i+1}}$. The reason for this is that either p_{i-1} or p_{i+1} are themselves stationary. A set of nodes could be "unlocked" by running a leader-election algorithm between neighbors and giving priority of motion to the consensual leader. This operation respects the descent nature of the algorithm and guarantees that we reach a desired critical configuration. •

IV. Outer-polygon approximation algorithms

Following [8], one can obtain a geometric characterization of the configurations $p_1, \dots, p_N$ in ∂Q that provide an optimal outer polygon approximation that minimizes E_O, when Q is strictly convex. This characterization is established through the corresponding unit tangent vectors $\mathbf{t}_1, \dots, \mathbf{t}_N$, the angles $\xi_i = \measuredangle(\mathbf{t}_i, \mathbf{t}_{i+1})$, $i \in \{1, \dots, N\}$ (measured in counter-clockwise order), and assumes that:

(i) $\ell(p_{i-1})^+ \cap \ell(p_{i+1})^- \neq \emptyset$,

(ii) the tangent at p_i forms a triangle, as shown in Figure 5.

We briefly summarize the result in the following.

Theorem 4.1: ([8]) Under conditions (i) and (ii), define the triangles $\mathcal{T}_i \equiv A_i B_i C_i$ whose vertices are given by the intersections $B_i = \ell(p_{i+1})^- \cap \ell(p_i)^+$, $C_i = \ell(p_{i+1})^- \cap \ell(p_{i-1})^+$ and $A_i = \ell(p_i)^- \cap \ell(p_{i-1})^+$. Then, the following formula holds:

$$\frac{\partial}{\partial p_i}\, \mathrm{E}_O(Q, P_E(p_1, \dots, p_N)) = -\frac{\partial \mathrm{Area}(\mathcal{T}_i)}{\partial p_i}\,. \qquad (8)$$

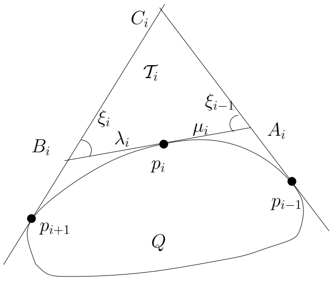


Fig. 5. Assumptions (i) and (ii) on every three nodes, p_{i-1}, p_i and p_{i+1} for formula (8) to be applicable.

Let $\overline{p_i\, A_i}$ (resp. $\overline{B_i\, p_i}$) denote the length of the segment defined by p_i and A_i (resp. by p_i and B_i). An expression for the above partial derivative is given by:

$$-\frac{\partial \mathrm{Area}(\mathcal{T}_i)}{\partial \xi_i}\frac{\partial \xi_i}{\partial p_i} = -\frac{\left(\overline{p_i\, A_i} - \overline{B_i\, p_i}\right)\overline{A_i B_i}}{2 \sin \xi_i \sin \xi_{i-1}}\frac{\partial \xi_i}{\partial p_i}\,. \qquad (9)$$

That is, the critical configuration for p_i that falls under assumptions (i) and (ii) must satisfy $\overline{B_i\, p_i} = \overline{p_i\, A_i}$. Alternatively, we have the expression:

$$\overline{B_i p_i} = \frac{(p_{i+1} - p_i) \cdot \mathbf{n}_{i+1}}{\mathbf{t}_i \cdot \mathbf{n}_{i+1}} = \frac{(p_i - p_{i-1}) \cdot \mathbf{n}_{i-1}}{\mathbf{t}_i \cdot \mathbf{n}_{i-1}} = \overline{p_i A_i}\,.$$

In the following, we make use of Theorem 4.1 to design a control law that asymptotically leads the nodes to a local minimum configuration for E_O. Unlike [8], we handle the cases where the $P_E(p_1, \dots, p_N)$ is not necessarily bounded and where Assumptions (i) and (ii) are not necessarily met.

Lemma 4.2: Let $p_1 = \gamma(s_1)$, $p_2 = \gamma(s_2) \in \partial Q$, with $s_1 < s_2$. The angle $\xi = \measuredangle(\mathbf{t}_1, \mathbf{t}_2)$ can be obtained as:

$$\xi_2 = \mathrm{atan2}(\mathbf{t}_2) - \mathrm{atan2}(\mathbf{t}_1) = \mathrm{atan2}(\mathbf{n}_2) - \mathrm{atan2}(\mathbf{n}_1)\,,$$

where the function $\mathrm{atan2}\colon \mathbb{R}^2 \to \mathbb{R}$ is defined by $\mathrm{atan2}(\mathbf{v}) = \measuredangle((1, 0), \mathbf{v})$, for $\mathbf{v} \in \mathbb{R}^2$.

Let $\mathrm{Q_I}$, $\mathrm{Q_{II}}$, $\mathrm{Q_{III}}$ and $\mathrm{Q_{IV}}$ denote the four quadrants in $\mathbb{R}^2$. Define the sets $S_1 = (\mathrm{Q_I} \cup \mathrm{Q_{II}}) \setminus (\partial(\mathrm{Q_I} \cup \mathrm{Q_{II}}))$ and $S_2 = \mathrm{Q_{III}} \cup \mathrm{Q_{IV}}$. Observe that for $\mathbf{v} = (v^1, v^2) \in \mathbb{R}^2 \setminus \{\mathbf{0}\}$, we can write:

$$\mathrm{atan2}(\mathbf{v}) = \begin{cases} \arctan\left(\frac{v^2}{v^1}\right), & \text{if } \mathbf{v} \in S_1, \\ \arctan\left(\frac{v^2}{v^1}\right) + \pi, & \text{if } \mathbf{v} \in S_2. \end{cases}$$

Therefore we can define $\xi_i = \measuredangle(\mathbf{t}_i, \mathbf{t}_{i+1})$ as the function $\xi_i\colon \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}$ such that:

$$\xi_i(\mathbf{t}_i, \mathbf{t}_{i+1}) = \begin{cases} \arctan\left(\frac{t^2_{i+1}}{t^1_{i+1}}\right) - \arctan\left(\frac{t^2_i}{t^1_i}\right), \\ \qquad \text{if } \mathbf{t}_i, \mathbf{t}_{i+1} \in S_1 \text{ or } \mathbf{t}_i, \mathbf{t}_{i+1} \in S_2\,, \\ \\ \arctan\left(\frac{t^2_{i+1}}{t^1_{i+1}}\right) - \arctan\left(\frac{t^2_i}{t^1_i}\right) + \pi, \\ \qquad \text{if } \mathbf{t}_{i+1} \in S_1,\ \mathbf{t}_i \in S_2 \text{ or vice-versa.} \end{cases}$$

The function $\xi_i(\mathbf{t}_i, \mathbf{t}_{i+1})$ is discontinuous in the regions:

$$D_1 = \{(\mathbf{t}_i, \mathbf{t}_{i+1}) \in \mathbb{R}^2 \times \mathbb{R}^2 \mid \mathbf{t}_i \in S_1,\ \mathbf{t}_{i+1} \in \partial(\mathrm{Q_I} \cup \mathrm{Q_{II}})\}\,,$$
$$D_2 = \{(\mathbf{t}_i, \mathbf{t}_{i+1}) \in \mathbb{R}^2 \times \mathbb{R}^2 \mid \mathbf{t}_i \in \partial(\mathrm{Q_I} \cup \mathrm{Q_{II}}),\ \mathbf{t}_{i+1} \in S_2\}\,.$$

Since $\xi_i(\mathbf{t}_i, \mathbf{t}_{i+1})$ is discontinuous, its gradient is not well defined everywhere. However, the gradient admits a continuous extension to $\mathbb{R}^2 \times \mathbb{R}^2$:

$$\frac{\partial \xi_i}{\partial \mathbf{t}_i}(\mathbf{t}_i, \mathbf{t}_{i+1}) = (t_i^2, -t_i^1)\,, \qquad \mathbf{t}_1, \mathbf{t}_2 \in \mathbb{R}^2\,.$$

Let us use this information to define our control law. Denote by $\overline{\mathbb{R}} \equiv [-\infty, +\infty]$ and define the following values for μ_i, $\lambda_i \in \overline{\mathbb{R}}$, $i \in \{1, \dots, N\}$:

$$\mu_i = \begin{cases} \dfrac{(p_i - p_{i-1}) \cdot \mathbf{n}_{i-1}}{\mathbf{t}_i \cdot \mathbf{n}_{i-1}}\,, & \mathbf{t}_i \cdot \mathbf{n}_{i-1} \neq 0, \\ +\infty\,, & \text{otherwise}\,, \end{cases}$$

$$\lambda_i = \begin{cases} \dfrac{(p_{i+1} - p_i) \cdot \mathbf{n}_{i+1}}{\mathbf{t}_i \cdot \mathbf{n}_{i+1}}\,, & \mathbf{t}_i \cdot \mathbf{n}_{i+1} \neq 0, \\ +\infty\,, & \text{otherwise}\,. \end{cases}$$

The distances μ_i and λ_i are graphically shown in Figure 5. Observe that because Q is strictly convex, μ_i and λ_i can not be both $+\infty$. Now, by means of λ_i and μ_i, we define the dynamical system:

$$\dot{p}_i = -\operatorname{sat}_v(\mu_i - \lambda_i)\,\mathbf{t}_i\,, \qquad i \in \{1, \dots, N\}, \tag{10}$$

where the function $\operatorname{sat}_v \colon \overline{\mathbb{R}} \to \mathbb{R}$, defined for some positive saturation value $v \in (0, +\infty)$, is given by:

$$\operatorname{sat}_v(x) = \begin{cases} x\,, & |x| \leq v \\ \frac{x}{|x|}\, v\,, & |x| \geq v\,. \end{cases}$$

We use the convention $|\pm\infty| = +\infty$, and the usual operations in $\overline{\mathbb{R}}$.

Theorem 4.3: Let the number of nodes N be $N \geq 3$. The control law (10) decreases monotonically $\mathrm{E}_O(Q, P_E(p_1, \dots, p_N))$. A critical point $(p_1^*, \dots, p_n^*)$ satisfies $\lambda_i^* = \mu_i^*$ for all $i \in \{1, \dots, N\}$.

Again, we omit the proof of this result for space reasons and we refer to a forthcoming technical report.

Simulations of outer-polygon approximation algorithm: Figure 6 shows the result of the implementation of the outer-polygon approximation algorithm. The eleven nodes are on the contour described by (5).

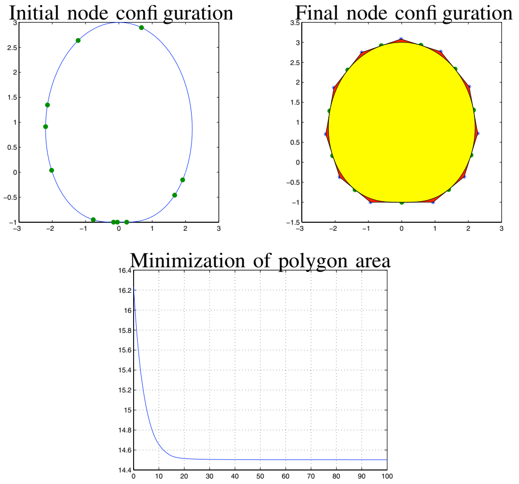


Fig. 6. Implementation of outer-polygon approximation algorithm.

A. Discrete-time outer-polygon approximation algorithms

It is easy to prove that an algorithm analogous to Algorithm 1 in the previous section guarantees convergence to a local extremum of E_O. We state the analogous results here omitting the corresponding proof.

Algorithm 3. Assume each node p_i has knowledge about its own label number $i \in \{1, \dots, N\}$. At each discrete time instant that we index by $k \in \mathbb{N}$ we define:

$$p_i(k+1) = \begin{cases} \bar{q}(p_{i-1}(k), p_{i+1}(k))\,, & \text{if } i \equiv k \mod N, \\ p_i(k), & \text{if } i \not\equiv k \mod N\,, \end{cases}$$

where $\bar{q}(p_{i-1}(k), p_{i+1}(k))$ is the closest point to $p_i(k)$ on the arc in ∂Q from $p_{i-1}(k)$ to $p_{i+1}(k)$ such that its tangent at ∂Q satisfies $\lambda_i = \mu_i$.

V. Outer minus inner polygon approximation algorithms

An alternative cost function that quantifies the approximation of the boundary of a convex body Q, is provided by the measure $E_S(\mathcal{P}_E(p_1, \dots, p_N), \mathcal{P}_I(p_1, \dots, p_N))$. Here we establish new computations of $\frac{\partial E^S}{\partial p_i}$, $i \in \{1, \dots, N\}$, when the outer polygon is well defined. This will lead to a new type of gradient decent algorithm.

Lemma 5.1: Let p_i, $p_{i+1} \in \partial Q$. Assume that $\mathbf{t}_{i+1} \times \mathbf{t}_i \neq \mathbf{0}$. Then, the area A of the triangle formed by the lines passing through $p_{i+1} - p_i$, $\ell(p_i)$ and $\ell(p_{i+1})$ can be expressed as:

$$A = \frac{1}{2} \frac{(\mathbf{n}_i \cdot (p_i - p_{i+1}))(\mathbf{n}_{i+1} \cdot (p_i - p_{i+1}))}{(\mathbf{n}_i \times \mathbf{n}_{i+1}) \cdot \mathbf{e}_3} \tag{11}$$

where $\mathbf{e}_3 = (0, 0, 1)$ and $\mathbf{n}_i \times \mathbf{n}_{i+1}$ is interpreted as a vector in $\mathbb{R}^3$.

Let us denote by $A(p_i, p_{i+1}, \mathbf{n}_i, \mathbf{n}_{i+1}) \equiv A_i$ the area (11) corresponding to p_i, p_{i+1}. We can write our cost function as

$$E_S(P_E(p_1, \dots, p_N), P_I(p_1, \dots, p_N)) = \sum_{i=1}^{N} A(p_i, p_{i+1}, \mathbf{n}_i, \mathbf{n}_{i+1}).$$

If we consider p_i, $\mathbf{n}_i$, as functions depending on the parameter $s \in [0, 1]$, we have that

$$\frac{\partial E_S(P_E(p_1, \ldots, p_N), P_I(p_1, \ldots, p_N))}{\partial s} = \frac{\partial A_i}{\partial s} + \frac{\partial A_{i-1}}{\partial s} ,$$

where we have used the shorthand notation A_i, $i \in \{1, \ldots, n\}$. Now for example, we can develop the expression for $\frac{\partial A_{i-1}}{\partial s}$, as follows:

$$\frac{\partial A_{i-1}}{\partial s} = \frac{\partial A_{i-1}}{\partial p_i}\frac{\partial p_i}{\partial s} + \frac{\partial A_{i-1}}{\partial \mathbf{n}_i}\frac{\partial \mathbf{n}_i}{\partial s} .$$

On the other hand, the Frenet-Serret equations imply

$$\frac{\mathrm{d}\,\mathbf{t}}{\mathrm{d}\,s}_{|s_0} = \kappa(s_0)\mathbf{n}(s_0) , \quad \frac{\mathrm{d}\,\mathbf{n}}{\mathrm{d}\,s}_{|s_0} = \kappa(s_0)\mathbf{t}(s_0) ,$$

where $\mathbf{t}$ and $\mathbf{n}$ are tangent and normal vectors such that $\mathbf{t} \times \mathbf{n}$ points towards the reader. Therefore, the expression in the partial derivative of A_{i-1} admits the following rewriting:

$$\frac{\partial A_i}{\partial s} = \left[\frac{\partial A_i}{\partial p_i} - \kappa(s)\frac{\partial A_i}{\partial \mathbf{n}_i}\right]\mathbf{t}_i .$$

In the following we include the explicit expressions for $\dfrac{\partial A_{i-1}}{\partial p_i}$ and $\dfrac{\partial A_{i-1}}{\partial \mathbf{n}_i}$. Let us use the notations:

$$\frac{\partial A_{i-1}}{\partial p_i} = \left[\frac{\partial A_{i-1}}{\partial x_i}, \frac{\partial A_{i-1}}{\partial y_i}\right] , \quad \frac{\partial A_{i-1}}{\partial \mathbf{n}_i} = \left[\frac{\partial A_{i-1}}{\partial n_i^1}, \frac{\partial A_{i-1}}{\partial n_i^2}\right] .$$

Then, one can compute that:

$$\begin{aligned}
\frac{\partial A_{i-1}}{\partial x_i} &= \frac{(p_i - p_{i-1}) \cdot (\, 2n_{i-1}^1 n_i^1 ,\, \mathbf{n}_{i-1} \times \mathbf{n}_i^+ \,)}{2(\mathbf{n}_{i-1} \times \mathbf{n}_i^+)} , \\
\frac{\partial A_{i-1}}{\partial y_i} &= \frac{(p_i - p_{i-1})(\, \mathbf{n}_{i-1} \times \mathbf{n}_i^+ ,\, 2n_{i-1}^1 n_i^1 \,)}{2(\mathbf{n}_{i-1} \times \mathbf{n}_i^+)} , \\
\frac{\partial A_{i-1}}{\partial n_i^1} &= \frac{n_i^2(\mathbf{n}_{i-1} \cdot (p_i - p_{i-1})^2)}{2(\mathbf{n}_{i-1} \times \mathbf{n}_i^+)^2} , \\
\frac{\partial A_{i-1}}{\partial n_i^2} &= \frac{n_i^1(\mathbf{n}_{i-1} \cdot (p_i - p_{i-1})^2)}{2(\mathbf{n}_{i-1} \times \mathbf{n}_i^+)} ,
\end{aligned}$$

where $\mathbf{n}_{i-1} \times \mathbf{n}_i^+ = n_{i-1}^1 n_i^2 + n_i^1 n_{i-1}^2$. To summarize, the gradient control law for each node is:

$$-\dot{p}_i = \frac{\partial A_{i-1}}{\partial p_i} + \frac{\partial A_i}{\partial p_i} - k(s_i)\left[\frac{\partial A_{i-1}}{\partial \mathbf{n}_i} + \frac{\partial A_i}{\partial \mathbf{n}_i}\right] .$$

Simulations for the inner/outer polygon approximation: Figure 7 show the results of inner/outer polygon approximation. The eleven nodes are on the contour described by (5).

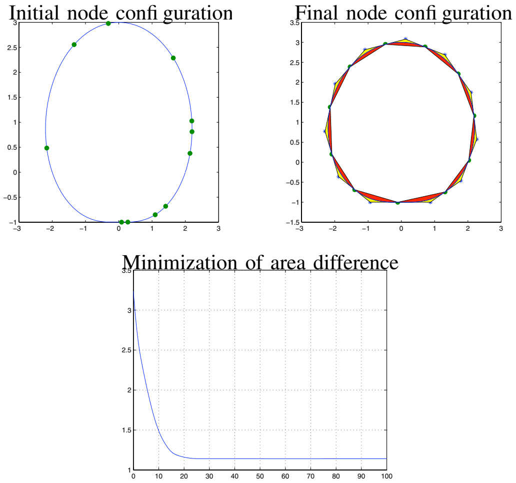


Fig. 7. Implementation of inner/outer-polygon approximation algorithm

VI. Conclusions

We have discussed various geometric optimization problems and corresponding gradient flows. Future works will focus on nonsmooth contours such as polygons, non-convex sets, and more general algorithms for optimal interpolation of boundaries.

References

[1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321–331, 1987.

[2] A. L. Bertozzi, M. Kemp, and D. Marthaler, "Determining environmental boundaries: asynchronous communication and physical scales," in *Cooperative Control. (Proceedings of the 2003 Block Island Workshop on Cooperative Control)* (V. Kumar, N. E. Leonard, and A. S. Morse, eds.), vol. 309 of *Lecture Notes in Control and Information Sciences*, pp. 25–42, New York: Springer Verlag, 2004.

[3] J. Clark and R. Fierro, "Cooperative hybrid control of robotic sensors for perimeter detection and tracking," in *American Control Conference*, (Portland, OR), pp. 3500–3505, June 2005.

[4] D. W. Casbeer, D. B. Kingston, R. W. Beard, T. W. Mclain, S.-M. Li, and R. Mehra, "Cooperative forest fire surveillance using a team of small unmanned air vehicles," *International Journal of Systems Sciences*, 2005. To appear.

[5] F. Zhang and N. E. Leonard, "Generating contour plots using multiple sensor platforms," in *IEEE Swarm Intelligence Symposium*, (Pasadena, CA), pp. 309–316, June 2005.

[6] S. B. Anderson, "An algorithm for boundary tracking in AFM," in *American Control Conference*, (Minneapolis, MN), pp. 508–513, June 2006.

[7] H. H. Johnson and A. Vogt, "A geometric method for approximating convex arcs," *SIAM Journal on Applied Mathematics*, vol. 38, no. 2, pp. 317–325, 1980.

[8] E. Trost, "Über eine Extremalaufgabe," *Nieuw Archief voor Wiskunde*, vol. 2, pp. 1–3, 1949.

[9] D. E. McLure and R. A. Vitale, "Polygonal approximation of plane convex bodies," *Journal of Mathematical Analysis and Applications*, vol. 51, no. 2, pp. 326–358, 1975.

[10] P. M. Gruber, "Approximation of convex bodies," in *Convexity and its Applications* (P. M. Gruber and J. M. Willis, eds.), pp. 131–162, Birkhäuser Verlag, 1983.

[11] P. M. Gruber, "Aspect of approximation of convex bodies," in *Handbook of Convex Geometry* (P. M. Gruber and J. M. Willis, eds.), vol. A, pp. 319–345, Oxford, UK: Elsevier, 1993.

[12] L. Chen, "Mesh smoothing schemes based on optimal Delaunay triangulations," in *13th International Meshing Roundtable*, (Sandia National Laboratories), pp. 109–120, 2004.