Continuous Geometric Algorithms for Robot Swarms with Multiple Leaders

Maximilian Ernestus* Sándor Fekete* Michael Hemmer* Dominik Krupke*

Abstract

We consider the problem of building a dynamic and robust network between mobile terminals with the help of a large swarm of robots in the continuous Euclidean plane. Individually, the robots have limited capabilities, both in terms of global information and computation. We propose a set of local *continuous* algorithms that together produce a generalization of a Euclidean Steiner tree. At any stage, the resulting overall shape achieves a good compromise between local thickness, global connectivity, and flexibility to further continuous motion of the terminals.

1 Introduction

Robot navigation is one of the classical application areas for computational geometry. How can we gather the geometric information that is necessary for orienting ourselves in a known or unknown environment? How can we carry out geometric computations efficiently, and how can we optimize specific objectives? Without a doubt, the close interaction between theory and practice for these challenges has motivated major progress, both in robotics and in computational geometry. Even without a specific focus on robotics, a relatively new area of algorithmics has arisen from considering not just a single active agent, but a whole group or even swarm. Swarm robotics combines classical robotics with distributed algorithms and many aspects of wireless sensor networks.

Traditionally, computational geometry has focused on discrete algorithms. In this paper, we demonstrate that a more continuous (not event-based) approach is able to lead to interesting and non-trivial geometric algorithms. In particular, we consider a large swarm of mobile robots with very simple individual capabilities. Motion is continuous, as is interaction and response between different robots.

The challenge is to combine two fundamentally opposite objectives: How can we develop local self-stabilizing mechanisms that allow the swarm to stay locally well connected, even when being pulled apart by several distant and mobile sites?

*Department of Computer Science, TU Braunschweig, Germany. maximilian@ernestus.de, s.fekete@tu-bs.de, mhsaar@gmail.com, d.krupke@tu-bs.de

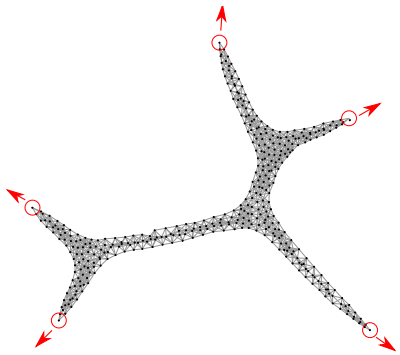


Figure 1: A robust robot swarm emulating a Steiner tree between five diverging leader robots.

Related Work. Even in a centralized and static setting with full information, we have to deal with the well-known NP-hard problem of finding a good Steiner tree [3]. There is a large body of work on geometric swarm behavior; for lack of space, we only mention Chazelle [1] for flocking behavior, and Fekete et al. [2, 5] for geometric algorithms for static sensor networks. As far as we know, only Hamann and Wörn [4] have explicitly considered the construction of Steiner Trees by a robot swarm. For static terminals, they start with an exploratory network; as soon as all terminals are connected, only best paths are kept and locally optimized. More specific references are given in Section 3.1, where they are used as building blocks.

This Paper. We propose a set of local, self-stabilizing algorithms that maintain a dynamic and robust network between leader robots. The algorithms ensure that the swarm adopts the directions of multiple leaders, while preserving a uniform thickness along the edges of the Steiner tree. We demonstrate the usefulness of this approach by simulations with a swarm of 400 robots, five leaders and various failure rates.

2 Preliminaries

For a finite set of robots $\mathcal{R}$ with an externally controlled subset of leader robots $\mathcal{L} \subsetneq \mathcal{R}, |\mathcal{L}| \ll |\mathcal{R}|$, we want the remaining robots $\mathcal{R} \setminus \mathcal{L}$ to maintain a dynamic and robust network that keeps the swarm connected, even in the presence of random robot failures and arbitrary leader movements. Thus, the over-

all shape of the swarm should form a "thick" Steiner tree among the leaders with the robots $\mathcal{R} \setminus \mathcal{L}$ evenly distributed along the edges, as shown in Figure 1.

Robots have the shape of circles; two of them are connected when within a maximum distance and with an unobstructed line of sight. Robots know the relative positions and orientations of their neighbors and can communicate asynchronously. Each robot has a unique ID; leader IDs are known by all others. Robot's translations and rotations are limited in velocity and acceleration. Communication is possible by broadcasting to immediate neighbors.

The perception of all robots is local; however, due to the known position and orientation difference, each robot can transform vectors of its neighbors to its own coordinate system. We avoid multi-hop transformations to keep errors small.

3 Algorithm

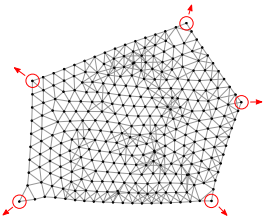
The proposed approach consists of a set of local self-stabilizing mechanisms that either detect a condition or induce a force. The weighted sum of the induced forces determines the robot motion; input for the local mechanisms of the local state and environment of the robot, output is a value for current robot motion. In principle, these mechanisms are continuous. (Our implementation described later updates at 60 Hz.)

We first discuss the base behavior of the robots in Section 3.1, inducing an almost convex swarm shape. This is subsequently improved by leader forces, stability improvement and thickness contraction.

3.1 Base Behavior

Our base behavior consists of three components that result in a swarm shape of a droplet. (i) The *flocking algorithm* of Olfati-Saber [8] considers regular distribution and movement consensus. The algorithm is a stateless equation based on potential fields and is proven to converge. It uses three rules: Attraction to neighbors, repulsion from too close neighbors, and adaption to the velocity of neighbors. We slightly modified the algorithm for better response to additional forces. (ii) An extended version of the *boundary detection* algorithm of McLurkin and Demaine [7], which determines if a robot lies on the boundary and also identifies small holes[1] by using the average angle. (iii) The *boundary tension* of Lee and McLurkin [6], which straightens and minimizes the boundary of the swarm. This is done by simply pushing boundary robots to the middle of its two boundary neighbors.

[1]The method theoretically allows the robots to distinguish exterior and interior boundaries and determine their size, but the limited precision and the convergence time limit this usage.



However, the base behavior without any other forces results in at most convex shapes before losing connectivity. The figure to the right depicts a situation in which the swarm is just about to lose connectivity. For stronger control and more variable shapes, leader forces are introduced.

3.2 Leader Forces

A single leader constitutes the simplest form of swarm control. In this case the swarm motion is determined by the leader's velocity. With multiple (possibly antagonistic) leaders, the swarm is not just steered, but may be stretched to the limit until connectivity is lost. Therefore, each robot needs to find an appropriate balance between the influence of different leaders. See top of Figure 2 for an illustration.

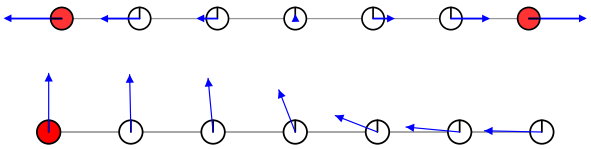


Figure 2: (Top) A one-dimensional scenario with two leaders (red) moving in opposite directions. (Bottom) With increasing distance to the leader, the effect shifts from velocity matching to leader pursuit.

There are two ways of following a leader: either by matching its velocity or by moving towards it. Velocity matching preserves the overall shape of the swarm, but fails with multiple leaders. In addition, there are accumulated losses in accuracy with each hop because the velocity information needs to be passed between robots with noisy sensors. On the other hand, moving towards the leader causes a deformation of the swarm and can also be used to control its shape when multiple leaders are used. However, regions close to the leaders suffer from "compression". We therefore combine both methods by a smooth transition between velocity matching close to the leaders and leader pursuit when further away; see bottom of Figure 2.

In order to achieve the combination of movement *with* the leader and *towards* the leader, three public variables are used for each leader. The **leader distance** is the minimum hop count to the leader. Let $\text{pred}(r)$ be the predecessor in a minimum-hop tree to the leader, which can be the leader itself. The **leader velocity** is the one of $\text{pred}(r)$ for a non-leader, and the robot's own velocity for the leader. The **leader direction** is a normalized direction vector calculated incrementally from the direction to $\text{pred}(r)$ as follows: Each robot takes the *leader direction* of its $\text{pred}(r)$ and merges it with the normalized direction

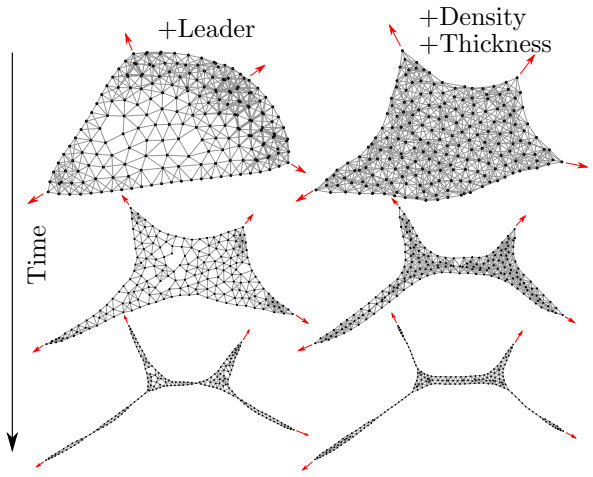


Figure 3: (Left) The basic swarm with leader forces added. (Right) Swarm with stability improvement. Lower swarms are scaled down for better visibility.

to pred(r). If pred(r) is the leader, only the normalized direction to it is used. For computing the leader force, the *leader direction* is scaled to the length of the *leader velocity* and then combined with a *leader distance*-sensitive weighting.

For $\ell \in \mathcal{L}$, let $c_\ell : \mathcal{R} \to \mathbb{R}^2$ be the force on a specific robot and let $d_\ell : \mathcal{R} \to \mathbb{N}$ be its distance to ℓ. The leader forces on robot r are combined as follows:

$$\sum_{\ell \in \mathcal{L}} c_\ell(r) \frac{d_\ell(r)^{-1}}{\sum_{\ell' \in \mathcal{L}} d_{\ell'}(r)^{-1}}.$$

Additionally we provide leaders with too few neighbors with an attraction force, so they do not lose connection to the swarm. This attraction spreads over some distance, but decreases exponentially.

3.3 Stability Improvement

Near Steiner points, connections along concave swarm boundaries may be stretched by boundary forces. When the involved edges approach the upper bound for communication, connections may be disrupted, to the point where the swarm loses connectivity. By adding a thickness-dependent compression force, we reduce neighbor distances without influencing the Steiner-tree shape of the swarm; in effect, this works similar to compression stockings. In the following, we give a heuristic for thickness computation and compression. In order to let the flocking algorithm handle this compression without destroying the regular distribution, we sketch a density distribution heuristic later in this Section. A comparison of a swarm with and without the stability improvement can be seen in Figure 3.

Thickness Contraction. We define the local thickness at a robot as the radius of the largest hop circle

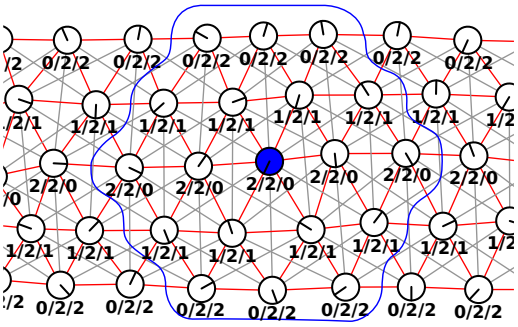


Figure 4: Thickness determination ($b(r)/t(r)/h(r)$) for a limb part. The red edges fulfill the Gabriel graph condition. A largest *hop circle* is marked in blue.

containing it. A hop circle of radius h with robot c as circle center is the set of all robots with a hop count $\leq h$ to c; only robots with distance equal to h may be on the boundary. An example is highlighted in blue in Figure 4.

The relationship between geometric thickness and boundary hop distance may be distorted by long connections that skip over robots. This can be avoided by only considering edges that fulfill the edge condition of the Gabriel graph, meaning that no robot is allowed to be closer to the midpoint of an edge than the robots connected by it. We denote the corresponding reduced neighborhood of a robot r as N'_r.

The following method is a simplified implementation of the thickness metric above, which performed well enough in simulation. It gets by with only three public variables; all circles with its center within a larger circle are ignored.

For this heuristic evaluation of the thickness $t(r)$ of a robot r, we need the hop distance $b(r)$ from the boundary and the circle center distance $h(r)$. Computing the hop distance to the boundary for each robot can easily be achieved by setting $b(r)$ to 0 for all robots on the boundary, while all others take the minimum of their neighbors plus one, as follows

$$b(r) = \begin{cases} 0 & r \text{ on boundary} \\ \min\{b(n)+1 \mid n \in N'_r\} & \text{else} \end{cases}$$

Small holes, that occur frequently but also vanish quickly, are excluded from the boundary, otherwise the value can become too instable. The thickness $t(r)$ is determined as the maximum $b(r)$ within some range $h(r)$, as follows.

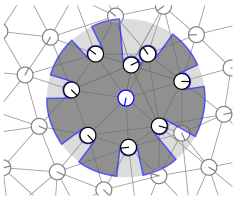
$$t(r) := \max\{\{b(r)\} \cup \{t(n) \mid n \in N'_r \wedge t(n) + \lambda \geq h(n)\}\},$$

where $\lambda \in \mathbb{N}$ is a small constant (e.g. $\lambda = 2$) that tackles the problem of irregular boundaries. If r is a circle center ($t(r) = b(r)$), then the circle center distance $h(r)$ is 0. Otherwise,

$$h(r) := \min\{h(n)+1 \mid n \in N'_r \wedge t(n) = t(r)\}$$

Based on this thickness $t(r)$, the described compression force grows linearly with this $t(r)$. It acts only on robots of large boundaries, so that small holes are not prevented from closing.

Density. The local density of a robot refers to the number of neighbors in relation to its observable area. By introducing an attraction to low and repulsion from high local density neighbors, the overall swarm density is maintained at a specific homogeneous level. It is determined by dividing the number of neighbors by the roughly calculated observable area as depicted in the figure to the right. In order to avoid lumps, robots in collision range are weighted higher. For robots on the boundary the computation is a bit more involved. However, further details are omitted due to limited space.



Let $\rho(r')$ be the averaged local density of robot r', ϱ the optimal density, and N_r the neighbors of r, then the density distribution force for a robot r is given by

$$\sum_{n \in N_r} \overline{p}_r(n) * \phi(\rho(n) - \varrho) \text{ with } \phi(x) = x^3/|x|,$$

where $\overline{p}_r : \mathcal{R} \to \mathbb{R}^2$ is the direction from robot r to a neighbor with the length of the distance for $\rho(n) \leq \varrho$ and of range minus distance else. We do not apply this force to robots on the boundary.

4 Experiments

We validated our approach by conducting experiments with a set of five leaders stretching out a swarm of 400 robots until it disconnects. The performance is measured by the length of the minimal Steiner tree on disconnection (calculated by the Geosteiner software [9]), divided by the theoretically maximal possible length estimated by $|\mathcal{R}'| * \text{range}$, where $\mathcal{R}'$ are the robots that did not fail yet. This would correspond to an optimal but extremely fragile Steiner tree in which *any* node failure disconnects the swarm. Thus, the best possible value of 1 is completely elusive.

For comparison we tested three configurations: Base—only the base behavior as discussed in Section 3.1; Lead—the basic behavior enriched by leader forces as discussed in Section 3.2; All—the final configuration that also incorporates Density and Thickness Contraction as presented in Section 3.3.

The experiments were carried out with 60 iterations per simulated second, a robot diameter of 10 cm and a range of 1.2 m. The maximal robot velocity was set to $1\,\mathrm{m\,s^{-1}}$, but the leaders only moved by at most $0.25\,\mathrm{m\,s^{-1}}$ in order to give the swarm robots the opportunity to react. These parameters are chosen arbitrary but are oriented to the R-One swarm robots of the Rice University.

Failure rate	Base	Lead	All
0	.07 .08 .09	.25 .30 .34	.28 .32 .35
$5 \cdot 10^{-6}$	.07 .08 .09	.25 .28 .32	.26 .29 .33
10^{-5}	.06 .08 .09	.23 .28 .31	.26 .30 .33
$2 \cdot 10^{-5}$	.07 .08 .09	.22 .25 .29	.26 .30 .33

Table 1: Relative Steiner tree sizes reached by first, second, and third quartiles. The failure rate is the probability of each robot to die in each step of the simulation.

For each configuration there were 100 random trials on four different failure rates. The results in Table 1 show that leader forces already produce decent swarm behavior, with survivability four times higher than for the base forces. Without robot losses, it reaches around 30% of the length of the hypothetical optimum. With robot failures, the performance gets weaker with increasing failure probability. The variant with additional stability improvement is slightly better without failures, but is clearly more robust against robot losses.

References

[1] B. Chazelle. The convergence of bird flocking. *J. ACM*, 61(4):21, 2014.

[2] S. P. Fekete and A. Kröller. Geometry-based reasoning for a large sensor network. In *Proc. SoCG*, pages 475–476, 2006.

[3] M. R. Garey, R. L. Graham, and D. S. Johnson. The complexity of computing steiner minimal trees. *SIAM J. Appl. Math.*, 32(4):835–859, 1977.

[4] H. Hamann and H. Wörn. Aggregating robots compute: An adaptive heuristic for the euclidean steiner tree problem. In *From Animals to Animats 10*, pages 447–456. Springer, 2008.

[5] A. Kröller, S. P. Fekete, D. Pfisterer, and S. Fischer. Deterministic boundary recognition and topology extraction for large sensor networks. In *Proc. SODA*, pages 1000–1009, 2006.

[6] S. K. Lee and J. McLurkin. Distributed cohesive configuration control for swarm robots with boundary information and network sensing. In *Proc. IROS*, pages 1161–1167. IEEE, 2014.

[7] J. McLurkin and E. D. Demaine. A distributed boundary detection algorithm for multi-robot systems. *Proc. IROS*, pages 4791–4798, 2009.

[8] R. Olfati-Saber. Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Trans. Automat. Contr.*, 51:401–420, 2006.

[9] D. Warme, P. Winter, and M. Zachariasen. Geosteiner 3.1. *Department of Computer Science, University of Copenhagen (DIKU)*, 2001.