Geometric Motion Planning: Finding Intersections

Sándor P. Fekete* Henning Hasemann*† Tom Kamphans*‡ Christiane Schmidt*

Abstract

We investigate a new model for mobile agents: Motion planning with geometric primitives, similar to ruler-and-circle constructions in classical geometry. In this first paper on this subject, we consider finding intersection points between two geometric objects using mobile agents that move on these objects. This amounts to finding the rendezvous point of the two agents.

1 Introduction

Algorithms for planning motions for mobile agents have been studied thoroughly in the past decades, both in computational geometry and in the robotics community. Such an algorithm relies on a set of primitives for motion, sensors, and communication that can be performed directly by the agent. Usually, the set of motion primitives models the agent's basic capabilities, most commonly are the primitives *move* and *turn*. As these basic motions are prone to error, it is worth investigating other models (i.e., set of motion primitives).

In this work, we introduce a new model for mobile agents. We assume that the agents are capable of performing geometric primitives, such as *move towards another agent*, *move on a circle*, *follow the ray that is defined by agent a and agent b*. That is, the agents are able to perform constructions similar to ruler-and-circle constructions in classical geometry (without marking trajectories). This model raises two questions: *What can we achieve using this model?* and *How can we implement such geometric primitives given, for example, a real robot?* In this first work on this subject, we focus on one particular aspect of the first question: determining the intersection point of two agents following a trajectory that is modeled by a simple geometric object.

Related Work

Search and rendezvous problems have been considered in many settings (e.g., [1, 5, 2]). Other models for mobile agents have been presented by defining minimal capabilities and investigating solvable tasks and hierarchy relations among different models [8, 4, 7]. Although our search space is a torus in some cases, we do not consider rendezvous on a torus as in [6].

Model

In general, we are given a swarm of non–point-shaped agents. Each agent has perfect communication capabilities, but only limited vision and restricted motion abilities. Motion is limited to a set of primitives that resemble moving on geometric objects.

In our case, we consider two agents with the motion primitives: *Move distance d on a circle* and *Move distance d on a given curve*.

2 Finding Intersections

We are interested in tasks that can be accomplished using our model. A useful building block for modeling complex tasks is to determine the intersection point between two geometric object. This amounts to finding rendezvous strategies for the agents moving on these objects.

Let C_1 and C_2 be two curves in the plane of lengths ℓ_1 and ℓ_2, respectively; A_1 and A_2 be the agents following C_1 and C_2, respectively. Note that we also allow open curves of unbounded length. As the agents are not point shaped, we can safely assume that the agent's minimum travel distance is its diameter (with one exception: In the case of closed curves when the agent returns to its start position after driving around the whole curve, we allow a move with smaller step width). Thus, we have a discrete search space. Transforming the possible positions of the agents into the plane (projecting the positions of A_1 along the X-axis, those of A_2 along the Y-axis) yields an integer grid. The initial position of the agents is defined as $S := (0, 0)$. The topology of the grid depends on whether the curves are closed or not: Two open curves define a disk, one open and one closed curve define a cylinder, and two closed curves define a torus. Now, finding the intersection point of C_1 and C_2 amounts to finding the (first) point $T := (x, y)$ on this grid where the agents meet.

Baeza-Yates et al. [3] considered searching on an infinite integer grid. They showed that any online strategy for finding a point within distance at most k (in L_1-metric) requires at least $2k^2 + O(k)$ steps and presented some strategies that achieve this bound. We use the strategy NSESWSNWN, see Figure 1(ii), for

*Algorithms Group, Braunschweig Institute of Technology, Germany. http://www.ibr.cs.tu-bs.de/alg

†Supp. by 7th Framework Progr. contr. 258885 (SPITFIRE)

‡Supp. by 7th Framework Progr. contr. 215270 (FRONTS)

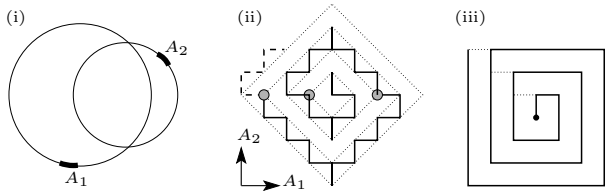

Figure 1: (i) Rendezvous space, (ii)/(iii) Search space. Strategies for (ii) L_1 (NSESWSNWN), (iii) L_∞.

our search. The strategy proceeds by successively visiting all points within distance k, which lie on a *diamond* around the origin; it requires $2k^2+5k+2$ steps. Note that after visiting every point within distance $k-1$, the strategy needs just $4k+3$ additional steps to visit every point within distance k. Although there is a slightly better strategy, we prefer NSESWSNWN because it produces a symmetric search path.

2.1 One-Dimensional Agents

In this section, we consider agents that move on a given curve. For now, the agents are one dimensional; that is, they extend only along the curve. Imagine a snake-like robot following a trail. Further, the agents do *not* move simultaneously.

2.1.1 Closed Curves of Equal Length

Theorem 1 *Let C_1 and C_2 be two curves of length ℓ. Any algorithm that finds the intersection at distance at most k needs at least*
(i) $2k^2+2k-4$ steps (for $k<n=\lceil \ell/2 \rceil$),
(ii) $2n^2+4zn+2n-2z^2-2z-4$ steps if $n<k, k=n+z$.

Proof. Any algorithm must visit every point at distance $\leq k$. Visiting m points takes $m-1$ steps. For $k=0$ we have one point. Each diamond of size k has $4k$ points; thus, we have $1+\sum_{i=1}^{k} 4i = 2k^2+2k+1$ points for $k \leq n$. Beyond n, the number of points per diamond shrinks. Let $z=k-n$, then we have $4(n-z)$ points per diamond, which yields a total of $2n^2+2n+1+\sum_{i=1}^{z} 4(n-i) = 2n^2+4zn+2n-2z^2-2z+1$ points. □

Theorem 2 *Let C_1 and C_2 be two intersecting, closed curves of length ℓ and let A_1 and A_2 be two mobile agents moving on C_1 and C_2, respectively. Let k be the distance to the (closest) rendezvous point, T, of A_1 and A_2. For finding T, the agents need at most*
(i) $2k^2+5k+2$ steps if $k \leq n$,
(ii) $2n^2+4zn+7n-2z^2-3z+2$ steps if $n<k, k=n+z$.

Proof. We use the following rendezvous strategy: We use the NSESWSNWN strategy to explore the search space, until we meet a point that was already explored before. This happens when two tips of the diamond touch each other, because the search space is a torus.

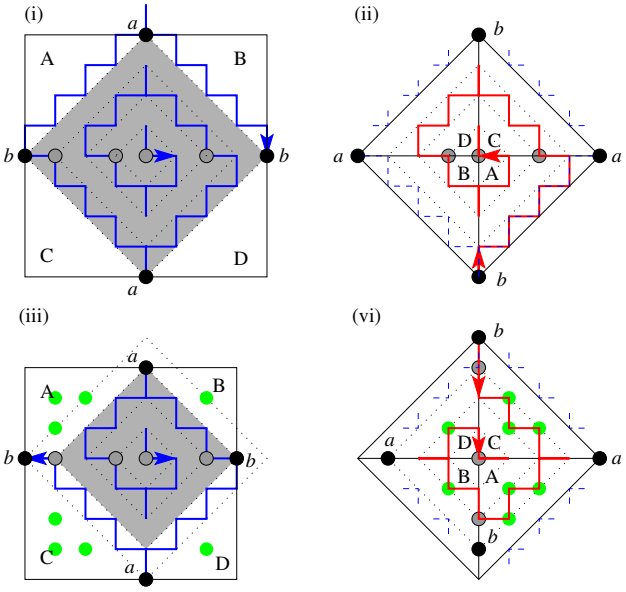

Figure 2: (i) and (iii) searching a diamond with NSESWSNWN until b is visited twice. (ii) and (iv) exploring the remaining diamond. (i) and (ii) show the case for even n, (iii) and (iv) for odd n. The gray dots show the end of one round.

Now we searched a diamond-shaped area and are left with four unsearched triangles, see Figure 2(i). As the search space is a torus, these triangles are connected and form, in turn, another diamond if seen from a different viewpoint, see Figure 2(ii). Thus, we search this diamond with a similar strategy, but starting with the outer layer and proceeding towards the origin of the diamond. Let n be the round when we switch the strategy and k be the current round (that is, we want to explore every point within distance k). When switching the strategies, we need $2n$ additional steps to move from b to the start of the next round if n is even, one step if n is odd. Let $S(k)$ be the total number of steps after finishing round k.
For $k \leq n$ we have $S(k) = 2k^2+5k+2$.
For $n<k<2n$, $k=n+z$, we have $4(n-1-z)+3$ additional steps per layer. Altogether, we have
$S(k) \leq 2k^2+5k+2+2n+\sum_{i=1}^{z} 4\,(n-i-1)+3$
$= 2n^2+4zn+7n-2z^2-3z+2$
□

2.1.2 Closed Curves of Different Length

If the curves have different sizes, the search space is no longer a square projected onto a torus. This means that the expanding diamonds begin to overlap in one dimension, while the other dimension is not yet explored in its total width, see Figure 3.

Let point a be the point where the expanding diamonds touch and let n be the layer in which we reach a. Let m be the layer where b is met.

Theorem 3 *Let C_1 and C_2 be two curves of length ℓ_1 and ℓ_2 ($\ell_1 < \ell_2$), respectively, $n = \lceil \ell_1/2 \rceil, m = \lceil \ell_2/2 \rceil$. Any algorithm that finds the intersection at distance at most k needs at least*
(i) $2k^2 + 2k - 4$ steps if $k \leq n$,
(ii) $2n^2+2n+z'(4n+2)-4$ steps if $n < k = n+z' \leq m$,
(iii) $4mn-2n^2+4nz-2z^2-2z+2m-4$ steps if $k = m+z$.

Proof. Part (i) is the same as in Theo. 1. Beyond the n-diamond, we have $z' = k - n$ $\langle\,\rangle$-shaped layers with $4n + 2$ points each. For (iii), we have the points from Case (ii) with $z' = (m-n)$ and add $\sum_{i=1}^{z} 4(n-i)$. □

Theorem 4 *Let C_1 and C_2 be two intersecting, closed curves of length ℓ_1 and ℓ_2, respectively. Let A_1 and A_2 be two mobile agents moving on C_1 and C_2, respectively. Let k be the distance to the (closest) rendezvous point, T, of A_1 and A_2. For finding T, the agents need at most*
(i) $2k^2 + 5k + 2$ steps if $k \leq n$,
(ii) $6n^2 + 7n + 2^j(n + 3) + 4nz' + 2j - 2$ steps if $n < k = n + z', 2^{j-1} < z' \leq 2^j$,
(iii) $5mn + n^2 + 4nz - 2z^2 - 2z + 4n + 3m + +2\log(m - n) - 2$ if $k = m + z$.

Proof. As above, we start with the NSESWSNWN strategy, until we complete layer n and are located at point b'. Now we switch the strategy. First, we return to a. Then we alternatingly search the areas left and right to the explored diamond, using a stairway-like search pattern, as shown in Figure 3. To keep the relocation costs low, we use the *doubling paradigm*; that is, we double the exploration depth each time we switch sides. We continue until we reach point b, from where we use the inverse NSESWSNWN strategy, as described in the preceding section.

To visit every point within distance k, $n < k < m$, $k = n+z$, we choose j such that $2^{j-1} < z \leq 2^j$. Up to b', we need $2k^2+5k+2$ steps as above. The move from b' to a costs $2n$. Then we explore up to a depth of 2^j using doubling. For doubling step j, we move from a to the point at which we ended after doubling step $j-2$ (plus one step to reach the position for the new stairway), incurring a cost of $2^{j-2}+1$. Then we move on the stairways. Note that each stairway covers two layers; thus, we explore $\frac{1}{2}(2^j - 2^{j-2})$ stairways with $4n$ steps each. Afterwards, we use 2^j+1 steps to return to a. Note that we do not need additional steps between two stairways. Thus, we have for $n < k \leq m$:
$S(k) = 2n^2 + 5n + 2 + 2n + 4n + 4$ for $j = 1$
$S(k) = 2n^2 + 5n + 2 + 2n + 12n + 10$ for $j = 2$
and for $j > 2$:
$S(k) \leq 2n^2 + 5n + 2 + 2n$ (from (i) and move to a)
$+4+6+\sum_{i=3}^{j}(2^i+2^{i-2}+2)+2^{j-1}+1$ (relocation costs)
$+4n\left(1 + 2 + \sum_{i=3}^{j}(2^{i-1} - 2^{i-3})\right) + 4n(k-2^{j-1})$ (stairways)
$= 6n^2 + 7n + 2^j(n + 3) + 4nz' + 2j - 2$

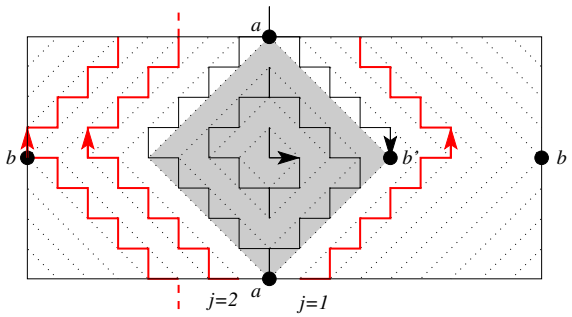

Figure 3: NSESWSNWN and stairways.

After visiting the stairways with the doubling technique, we use an inverse NSESWSNWN strategy as in Case (ii) of Theorem 2. For convenience, assume $m = n+2^j$ and let $k = m+z$. We add $\sum_{i=1}^{z} 4(n-i)-1$ to the result of the preceding case and get $S(k) \leq -2z^2+4nz-4z+6n^2+7n+2^j(n+3)+4nz'+2j-2$. With $2^j = m - n$ we get $S(k) \leq 5mn + n^2 + 4nz - 2z^2 - 2z + 4n + 3m + +2\log(m - n) - 2$. □

2.2 One-Dimensional Agents Moving Simultaneously

In the preceding sections, we assumed that the agents move alternatingly. How do the agents proceed, if they are allowed to move simultaneously? The search space remains the same. If the agents move alternatingly, all points of equal distance to the start lie on a diamond (i.e., a scaled copy of the L_1-unit circle). In this case, points of equal distance form a square (i.e., a scaled copy of the L_∞-unit circle). We consider the case of two curves of equal length.

Theorem 5 *Even if the agents are allowed to move simultaneously, there is an optimal strategy in which the agents move alternatingly.*

Proof. We show that an optimal strategy moves on a rectangular spiral-like search pattern, as shown in Figure 1(iii). Let the target be located at some unknown finite distance, k, from the start point. If an upper bound k' is known to the agent, visiting points in distance $k'+1$ (before every point of distance $\leq k'$ is visited) does not make sense to the agent, because these visits prolong the search path to the unvisited points within distance k'. Thus, if the agent knows no such upper bound, it has to cover each layer of points of same distance i completely before it proceeds to the next layer with points of distance $i+1$. As connecting two successive layers costs one step, the squared spiral is optimal. An axis-parallel move in the search space corresponds to a move of a single agent in the rendezvous space. Thus, the agents move only alternatingly, even if they are allowed to move simultaneously. □

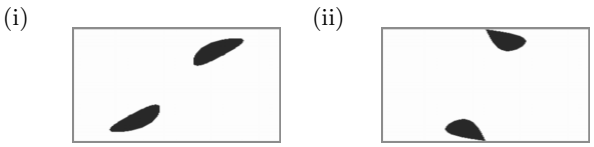

Figure 4: The search space for $R = 1$ and $r = 4$ (i) $d = 3$ (ii) $d = 6$, corner case $d \geq 2r - 2R$, i.e. the rendezvous area is connected.

2.3 Two-Dimensional Agents

Consider two mobile agents shaped as disks of radius R, each moving on one of two intersecting circles of radius r, $r \geq R$, whose centers have distance $d \leq 2r$. We identify the position of the agent on the first circle with the angle α between the circles center and the line between the two circles' centers. Analogously, we express the position of the other agent as an angle β and call the circles of the agents C_α and C_β, respectively. Again, the rendezvous search space is a torus.

Note that, in contrast to the preceding section, modeling the search space as a grid is not sufficient anymore, because now there is an infinite number of rendezvous points, see Figure 4. We observe, however, that the set of rendezvous points for this scenario consists of no more than two simply connected components, one for each intersection point. For $d \geq 2r-2R$ or $d \leq 2R$, we have only one simply connected component. In the rest of this section, we will assume that none of these special cases occurs.

Searching for a point on a torus is quite involved; thus, we want to find a convex region of a certain size. This allows us either to inspect a finite set of points on a grid or to move on an Archimedean spiral.

Lemma 6 *In the search space, there is a square of size at least $2R \times 2R$ such that all points inside the square are rendezvous points.*

Proof. First, observe the line, M, through the centers of C_α and C_β. For reasons of symmetry, in the following we consider only one side of M. Construct the line segment, L, of length $2R$ that is parallel to M and whose end points lie on C_α and C_β, see Figure 5. Let p_α and p_β be the endpoints of L lying on C_α and C_β, respectively. Let q_α and q_β be the intersection points of C_α and C_β and the lines perpendicular to L through p_β and p_α, respectively. Note that $|\overline{p_\alpha q_\beta}| = |\overline{q_\alpha p_\beta}|$. We consider two cases:

Case 1: $|\overline{p_\alpha q_\beta}| \leq 2R$. All pairs of points with α on the arc from p_α to q_α and β on the arc from p_β to q_β define a rendezvous point.

Case 2: $|\overline{p_\alpha q_\beta}| > 2R$. We parallely move both $\overline{p_\alpha q_\beta}$ and $\overline{q_\alpha p_\alpha}$ towards the intersection point, such that $|\overline{p'_\alpha q'_\beta}| = |\overline{q'_\alpha p'_\beta}| = 2R$ holds. Now, all pairs of points with α on the arc from p'_α to q'_α and β on the arc from p'_β to q'_β define a rendezvous point.

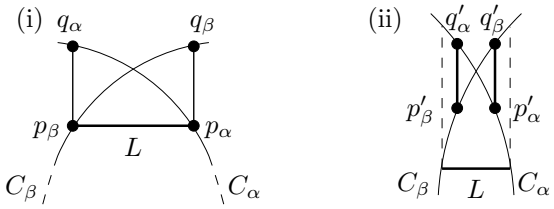

Figure 5: Construction of $p/q_{\alpha/\beta}$. The bold lines have length $2R$. (i) Case 1, (ii) Case 2.

In both cases, the arcs for α and β have length greater than or equal to $2R$. Thus, there is a $2R \times 2R$ square of rendezvous points in the search space. □

2.4 Other Topologies

If one of the curves is open/infinite, the search space is no longer a torus. In these cases, we search the rendezvous point with strategies combining spiral searches and doubling. In the case of finite, open curves it may happen that one end of the search space is fully explored, while other parts are unexplored. In this case, we use a meandering search path. We leave the details to the full version of this paper.

Acknowledgements

We thank Friedhelm Meyer auf der Heide, Bastian Degener, and Barbara Kempkes for helpful discussions.

References

[1] S. Alpern and S. Gal. *The Theory of Search Games and Rendezvous*. Kluwer Academic Publications, 2003.

[2] E. J. Anderson and S. P. Fekete. Two-dimensional rendezvous search. *Oper. Res.*, 49:107–118, 2001.

[3] R. Baeza-Yates, J. Culberson, and G. Rawlins. Searching in the plane. *Inform. Comput.*, 106:234–252, 1993.

[4] J. Brunner, M. Mihalak, S. Suri, E. Vicari, and P. Widmayer. Simple robots in polygonal environments: A hierarchy. In *Proc. Algosensors*, 2008.

[5] R. Fleischer, T. Kamphans, R. Klein, E. Langetepe, and G. Trippen. Competitive online approximation of the optimal search ratio. *Siam J. Comput.*, pages 881–898, 2008.

[6] E. Kranakis, D. Krizanc, and E. Markou. Mobile agent rendezvous in a synchronous torus. In *LATIN*, pages 653–664, 2006.

[7] J. M. O'Kane and S. M. LaValle. Dominance and equivalence for sensor-based agents. In *Proc. 22nd National Conf. Artif. Intell.*, pages 1655–1658, 2007.

[8] S. Suri, E. Vicari, and P. Widmayer. Simple robots with minimal sensing: From local visibility to global geometry. *Int. J. Robot. Res.*, 27:1055–1067, 2008.