Path Planning for Swarms by Combining Probabilistic Roadmaps and Potential Fields

Alex Wallar[1] and Erion Plaku[2]

[1] School of Computer Science, University of St Andrews,
St Andrews, Fife KY16 9AJ, Scotland, United Kingdom
[2] Dept. of Electrical Engineering and Computer Science
Catholic University of America, Washington DC 20064 USA

Abstract. This paper combines probabilistic roadmaps with potential fields in order to enable a robotic swarm to effectively move to a desired destination while avoiding collisions with obstacles and each other. Potential fields provide the robots with local, reactive, behaviors that seek to keep the swarm moving in cohesion and away from the obstacles. The probabilistic roadmap provides global path planning which guides the swarm through a series of intermediate goals in order to effectively reach the desired destination. Random walks in combination with adjustments to the potential fields and intermediate goals are used to help stuck robots escape local minima. Experimental results provide promising validation on the efficiency and scalability of the proposed approach. Source code is made publicly available.

1 Introduction

Swarm robotics seeks to enable a large number of robots to accomplish complex tasks via simple interactions with one another and the environment [1]. Swarm robotics draws inspiration from social insects, such as ants and bees, where intelligent group behaviors emerge from simple interactions among the individuals. Such framework provides a level of scalability and robustness that is difficult to achieve with centralized approaches. As research progresses, applications of swarm robotics are emerging in exploration, mapping, monitoring, inspection, and search-and-rescue missions, as surveyed in [2, 3].

In many of these applications, the swarm needs to move in cohesion to a goal destination while avoiding collisions with obstacles and among the robots in the swarm. This path-planning problem presents significant challenges. As path planning is PSPACE-complete, exact algorithms, which always find a solution if it exists and report no solution otherwise, are difficult to implement and are limited in practicality to low-dimensional systems due to the exponential dependency on the problem dimension [4–6]. As a result, research has focused on alternative approaches that do not determine the existence of a solution but seek instead to achieve efficiency and scalability in increasingly complex settings.

A common approach in path planning for robotic swarms is to impose artificial potential functions (APFs) that seek to push the swarm away from the

obstacles and toward the goal [7–10]. APFs provide scalability as the addition of new robots to the swarm generally requires only computation of repulsive forces from obstacles, attractive forces from the goal, and forces resulting from the limited interactions with neighboring robots. However, an inherent challenge with APFs is the tendency to get stuck in local minima especially when planning paths for large swarms moving in cluttered environments and through narrow passages. APFs that avoid local minima exist in limited cases, e.g., a point robot in a generalized sphere world [11], but not for general path planning for swarms.

Alternative approaches aim to avoid local minima by relying on probabilistic roadmaps (PRMs) [12]. PRMs capture the connectivity of the configuration space via a roadmap obtained by sampling collision-free configurations and connecting neighboring configurations with collision-free paths. A configuration generally specifies the placement of each robot in the swarm and a path between two configurations is typically obtained by interpolation. A collision-free path from an initial to a goal configuration is then obtained by performing graph search on the roadmap. Starting with the original PRM [12] and continuing over the years, PRMs have had great success in solving challenging problems [13–18]. The work in [19, 20] use APFs to increase PRM sampling near obstacles to facilitate connections through narrow passages. The work in [21–23] uses PRM to enable robotic swarms achieve different behaviors such as homing, coverage, goal searching, and shepherding. The roadmap, however, is built over the high-dimensional configuration space, which considerably increases the computational cost. The work in [24] uses multi-level PRMs in combination with Bezier curves to guide a multi-robot system to a desired destination while maintaining a specific formation. The approach has been applied only to a small number of robots and requires considerable precomputation to build the multi-level PRMs.

Even though significant progress has been made, scalability still remains problematic in PRMs [25, 26]. As the number of robots increases, it becomes difficult to sample collision-free configurations and generate collision-free paths that connect neighboring configurations. Moreover, significantly larger roadmaps are needed to capture the connectivity of the configuration space. As a result, efficiency of PRMs starts deteriorating as the number of robots is increased. These issues become even more problematic when considering robotic swarms since PRMs do not have a mechanism to ensure that the robots move in cohesion as a swarm rather than as individual entities.

To improve the efficiency of path planning for robotic swarms, this paper develops a novel approach, named `CRoPS` (`C`ombined `Ro`admaps and `P`otentials for `S`warms), which combines APFs with PRMs. While `CRoPS` draws from PRM the underlying idea of using a roadmap, it does not suffer from scalability issues as the roadmap is constructed over the two-dimensional workspace instead of the high-dimensional configuration space of the swarm. Moreover, `CRoPS` does not use the roadmap to plan the entire path of the swarm, but rather to generate a series of intermediate goals that serve as attractive potentials to guide the swarm toward the desired destination. `CRoPS` then relies on APFs to enable the robots move in cohesion as a swarm from one intermediate goal to the other

while avoiding collisions. This combination of PRMs with APFs is crucial to the efficiency and scalability of `CRoPS`. Experimental results in simulation with increasingly large swarms moving in complex environments containing numerous obstacles and narrow passages provide promising validation.

2 Method

The swarm motions are governed by the following criteria:

1. There is long range attraction to intermediate goals and final destination.
2. Robots are repulsed from obstacles.
3. Robots move as a swarm while keeping some separation from one another.
4. A robot's heading is influenced by the headings of its neighbors.

To guide the swarm to the final destination, global path planning based on PRMs is used to determine suitable intermediate goals. The roadmap is constructed by sampling collision-free points and connecting neighboring points with collision-free edges to obtain a graph that captures the connectivity of the environment. Roadmap vertices and edges are associated with weights that estimate the feasibility of the swarm to pass through their surrounding areas. The shortest path in the roadmap to the final destination is used to provide a series of intermediate goals that the swarm can follow to effectively reach the final destination. An attractive potential, denoted as $PF_{igoal}(b)$, is added between each robot b and its current intermediate goal. When the robot reaches the current intermediate goal, the next point in the shortest path is set as the new intermediate goal.

To avoid collisions with obstacles, `CRoPS` creates a strong repulsive potential field, denoted as $PF_{obst}(b)$, which pushes each robot b away from the obstacles. The repulsive potential increases quadratically with respect to the inverse of the distance from the robot to the obstacles. Such rapid increase prevents the robots from getting too close to the obstacles.

To maintain separation among the robots in the swarm, each robot is pushed away from its neighbors. A weak sigmoidal repulsive potential, denoted as $PF_{sep}(b)$, is employed rather than a strong quadratic repulsive potential in order to push b away from its neighbors but not so strongly as to separate it from the swarm.

To move as a swarm, a robot's heading is influenced by the headings of its neighbors, defined as $PF_{heading}(b)$. To promote effective movements, preference is given to neighbors that are not stuck and are neither too close nor too far. The headings of the neighbors that are chosen are averaged and combined with the other potential fields to determine the new heading and position of each robot.

The different potential fields are superimposed to obtain the overall force vector applied to each robot b. In this way, the potential field on the robot b exerts a strong repulsive potential away from the obstacles while attracting it to the current intermediate goal, maintaining a separation distance from the other robots, and adjusting the heading so that the robots move as a swarm toward the final destination. If a robot gets stuck in local minima, random walks in combination with adjustments to the potential fields and intermediate goals are

Algorithm 1 Pseudocode for `CRoPS`

$RM = (V, E) \leftarrow$ ConstructRoadmap(), where
    $V \leftarrow$ sample numerous random points, discard those that are in collision
    $E \leftarrow$ connect neighboring points in V, discard those edges that are in collision
    $w(q_i) \leftarrow$ assign weight to each vertex $q_i \in V$
    $w(q_i, q_j) \leftarrow$ assign weight to each edge $(q_i, q_j) \in E$
$\zeta = [q_k]_{k=1}^{m} \leftarrow$ IntermediateGoals(RM) ♢ *obtained from shortest path*
$igoal(b) \leftarrow \zeta(1)$ ♢ *set current intermediate goal for each robot*
while solved() = `false` **do**
  for $b \in Robots$ with ReachedIntermediateGoal(b) = `true` **do**
    $igoal(b) \leftarrow$ NextIntermediateGoal(ζ)
  for $b \in Robots$ **do**
    $PF(b) \leftarrow$ superimpose($PF_{obst}(b), PF_{sep}(b), PF_{igoal}(b), PF_{heading}(b), PF_{escape}(b)$)
    $NewHeading(b) \leftarrow w_1 heading(b) + w_2 PF(b)$
  for $b \in Robots$ **do**
    $heading(b) \leftarrow NewHeading(b)$; $pos(b) \leftarrow pos(b) + heading(b)$

used to help it escape. Pseudocode for the overall approach is given in Algo. 1. Details of the main steps follow. Source code and detailed documentation including all parameter values are publicly available [27]. Parameter values are generally determined empirically.

2.1 Roadmap Construction

`CRoPS` constructs a roadmap in order to effectively guide the swarm toward the goal. Since the swarm could have many robots, the roadmap is not constructed over the high-dimensional configuration space, as it is often the case in PRM approaches, but is instead constructed over the low-dimensional workspace where the swarm moves. As explained in this section, `CRoPS` uses the roadmap to find intermediate areas in which the swarm can move to effectively reach the goal.

Roadmap Vertices: The roadmap is constructed by first sampling a large number of points uniformly at random inside the workspace boundaries and then discarding all the points that are in collision or too close to an obstacle. A parameter, d_{clear}, determines the minimum acceptable distance from a sampled point to the nearest obstacle. The remaining points, which are all at least d_{clear} units away from the obstacles, are added as vertices to the roadmap graph $RM = (V, E)$. A roadmap vertex q_i and the clearance d_{clear} conceptually define a clearance area as a disk centered at q_i with radius d_{clear}, denoted as $area(q_i)$. In order to bias the swarm movements toward less cluttered areas, each roadmap vertex q_i is associated with a weight $w(q_i)$ which estimates how feasible it is for the swarm to travel through $area(q_i)$. More specifically, the weight is defined as

$$w(q_i) = (\sum_{o \in Obstacles} dist(q_i, o))^3,$$

where $dist(q_i, o)$ denotes the minimum distance from q_i to the obstacle o. In this way, small weights indicate the presence of obstacles nearby, which may make it more difficult for the swarm to pass through. As explained later in the section, `CRoPS` gives preferences to roadmap vertices associated with high weights which are indicative of areas with high clearance. Note that other definitions for the weight function are possible. The particular function used in this paper worked well for the experiments as it captures the desired properties of biasing the swarm movements towards less cluttered areas.

Roadmap Edges: After generating roadmap vertices, `CRoPS` connects each roadmap vertex to its k nearest neighbors. Edges that are in collision are discarded. The weight of an edge connecting q_i to q_j is defined as

$$w(q_i, q_j) = ||q_i, q_j||_2 / \min(w(q_i), w(q_j)),$$

where $||q_i, q_j||_2$ is the Euclidean distance from q_i to q_j. Note that $w(q_i, q_j)$ is small when q_i and q_j are close to each other and away from obstacles.

Intermediate Goals along Shortest Roadmap Path: Dijkstra's shortest-path algorithm is used to compute the shortest path ζ in the roadmap to the final destination, where the weight of a roadmap edge (q_i, q_j) is defined by $w(q_i, q_j)$ as described above. Each vertex $q_k \in \zeta$ defines an intermediate goal for the swarm. More specifically, `CRoPS` seeks to move the swarm to the goal by passing through the areas $area(q_k)$ as defined by the vertices q_k along the shortest path ζ. Note that the dependency of the edge weights on vertex weights ensures that the shortest path in the roadmap does not come too close to the obstacles, which could lead the swarm to often get stuck in local minima.

2.2 Potential Fields

Repulsion from Obstacles: An imperative objective for the swarm is to always avoid collisions with obstacles. For this reason, a repulsive potential is defined that pushes the robots away from the obstacles. More specifically, the repulsive potential between a robot b and an obstacle o is defined as

$$P_{obst}(b, o) = \frac{1}{(dist(pos(b), o) - radius(b))^2},$$

where $pos(b)$ and $radius(b)$ denote the position and radius of the robot b, respectively. Note that the important aspect of this repulsive function is that its value increases rapidly as the robot approaches an obstacle. This ensures that the robot would be pushed away and never collide with an obstacle.

In order to limit the influence of the obstacles that are far away, the repulsion is computed only from those obstacles that are within a certain distance Δ_{obst} from the robot. The potential field imposed by the obstacles is then defined as

$$PF_{obst}(b) = \sum_{\substack{o \in Obstacles \\ dist(b,o) \leq \Delta_{obst}}} (pos(b) - ClosestPoint(o, pos(b)))P_{obst}(b, o),$$

where $ClosestPoint(pos(b), o)$ denotes the closest point on the obstacle o to $pos(b)$.

Repulsion from other Robots: As the swarm moves, the robots need to avoid coming too close to each other as it could lead to collisions. At the same time, the robots should not be far away from each other in order to move as a swarm. To achieve these objectives, `CRoPS` uses a weak repulsive sigmoid function

$$P_{sep}(b_i, b_j) = \frac{1}{1 + \exp(\delta_{sep}\,||pos(b_i), pos(b_j)||_2)},$$

where δ_{sep} is a scaling constant. In order to limit the influence of the robots that are far away, similar to the potential field for obstacles, the repulsion is computed only from those robots that are within a certain distance Δ_{sep}. The potential field imposed on the robot b by the other robots is then defined as

$$PF_{sep}(b) = \sum_{\substack{b_i \in Robots - \{b\} \\ dist(b,b_i) \leq \Delta_{sep}}} (pos(b) - pos(b_i)) P_{sep}(b, b_i).$$

In this way, the robots travel close together but are pushed away when the come too close to one another.

Attraction to the Current Intermediate Goal: As discussed, `CRoPS` uses the shortest path ζ in the roadmap to set the intermediate goals for the swarm. Let $igoal(b)$ denote the current intermediate goal of the robot b. Note that $igoal(b)$ is associated with some point q_k in ζ. An attractive potential field that pulls the robot b towards $igoal(b)$ is then defined as

$$PF_{igoal}(b) = \frac{igoal(b) - pos(b)}{1 + \exp(\delta_{igoal}\,||pos(b), igoal(b)||_2)},$$

where δ_{igoal} is a scaling constant. Although other definitions are possible, the sigmoid function allows the robots to reach the goal without getting too greedy, which could lead to getting stuck in local minima. When b reaches $igoal(b)$, the next point q_{k+1} in the shortest path ζ is set as the new intermediate goal for b.

Influence of Neighbors on Heading: In order to make the robots move as a swarm, the heading of a robot b is also influenced by the headings of neighboring robots. In order to select suitable neighbors, robots that are stuck are excluded from consideration. Moreover, preference is given to those neighbors that are neither too far nor too close from b. This is achieved by using a Gaussian function $\gamma(b, b_i)$ with mean μ and standard deviation σ, i.e.,

$$\gamma(b, b_i) = \exp\left(\frac{-(||pos(b), pos(b_i)||_2 - \mu)^2}{2\sigma^2}\right),$$

and selecting as $Neighs(b)$ the k closest nonstuck robots according to $\gamma(b, b_i)$. The potential field imposed on the robot b by the headings of the neighboring robots is then defined as

$$PF_{heading}(b) = \sum_{b_i \in Neighs(b)} heading(b_i).$$

Escaping Local Minima: Each robot b keeps track of its past positions in order to determine if it is stuck in local minima. More specifically, a robot b is considered stuck if it has moved very little during the last ℓ time steps, i.e.,

$$stuck(b) = \begin{cases} 1, & \text{if } ||pos(b) - prev_\ell(b)||_2 < \Delta_{stuck} \\ 0, & \text{otherwise,} \end{cases}$$

where $prev_\ell(b)$ denotes the position of the robot ℓ steps in the past, and Δ_{stuck} is a threshold constant.

If a robot b is determined to be stuck, then a random vector is added to the mix of the potential fields, i.e.,

$$PF_{escape}(b) = stuck(b)(r_x, r_y),$$

where r_x, r_y constitute a random direction. Note that $PF_{escape}(b) = (0, 0)$ if the robot is not stuck. By performing a random walk for several steps, the robot increases the likelihood of escaping local minima.

In addition, when $stuck(b) = \texttt{true}$, a different mean μ_{stuck} and a different standard deviation σ_{stuck} are used to compute $PF_{heading}(b)$. The new mean and standard deviation have smaller values in order to select more nearby neighbors (recall that only nonstuck neighbors are considered for the selection.) This allows the robot to select different neighbors to influence its heading, since the original neighbors could have contributed to the robot being stuck in the local minima.

To further increase the likelihood of escaping local minima, `CRoPS` also adjusts the intermediate goals of a stuck robot. In particular, if q_k is the current intermediate goal, then `CRoPS` changes it to q_{k-1} if the robot is still in a local minima after a few iterations. If the robot is still unable to escape the local minima, the intermediate goal is set to q_{k+1}. By switching from the current to a past or to a future intermediate goal, the robot is given further flexibility which facilitates escaping local minima.

Superimposition of Potential Fields: The different potential fields are superimposed to obtain the overall force vector applied to the robot b:

$$PF(b) = \frac{\sum_{\phi \in fields} (||PF_\phi(b)||_2 PF_\phi(b))}{\sum_{\phi \in fields} ||PF_\phi(b)||_2},$$

where $fields = \{obst, sep, igoal, heading, escape\}$. The heading and the position of the robot b are then updated as

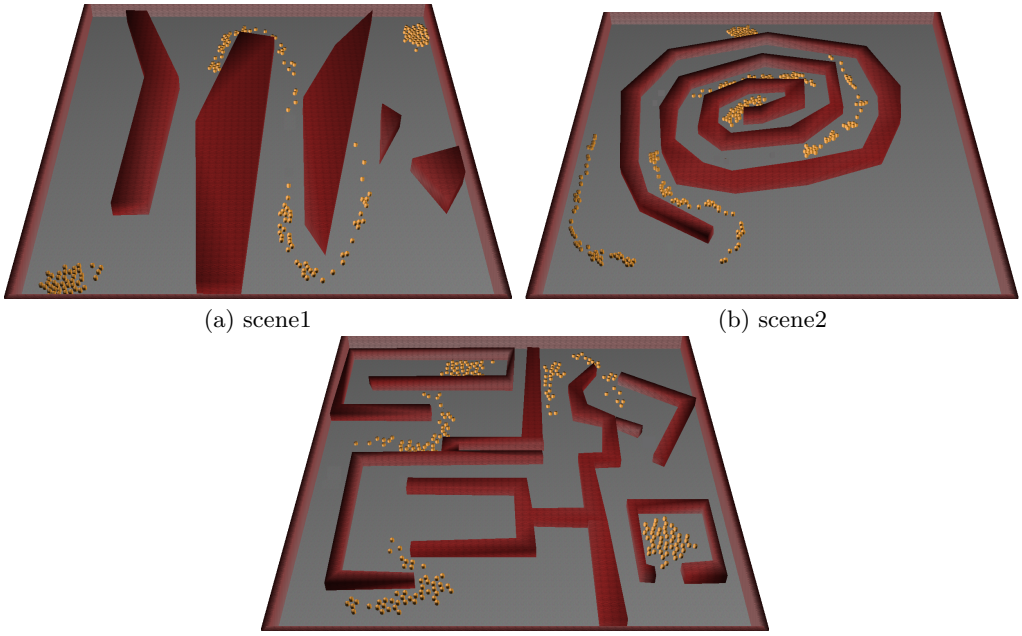
$$heading(b) \leftarrow w heading(b) + PF(b)$$
$$pos(b) \leftarrow pos(b) + heading(b),$$

where w is an adjustment constant.

The calculation of $PF(b)$ ensures that the subfield that has the highest potential during the current iteration will have the highest influence when the heading is calculated. The overall potential has a sigmoidal attraction to the immediate goal and sigmoidal repulsion from the robots as well as an inverse distance squared repulsion from obstacles. In this way, the potential field on the robot b exerts a strong repulsive potential away from the obstacles while attracting it to the current intermediate goal, maintaining a separation distance from the other robots, and adjusting the heading so that the robot b moves in a direction similar to its neighbors. Escape strategies are also applied in order for the robot to avoid getting stuck in local minima.

3 Experiments and Results

Experiments are conducted in simulation using different scenes and an increasing number of robots to test the efficiency and scalability of the approach. Fig. 1 provides an illustration of the scenes. These scenes provide challenging test cases as the swarm has to avoid numerous obstacles and pass through multiple narrow passages in order to reach the final destination.



(a) scene1 (b) scene2

(c) scene3

Fig. 1. Scenes used in the experiments. Each figure also shows intermediate swarm configurations along the path from the initial to the goal.

3.1 Measuring Performance

A problem instance is defined by a scene and the number of robots. Due to the probabilistic nature of the roadmap, performance on a particular problem instance is based on twenty different runs. Results report the average time for all the robots to reach the final destination. Results also report the average distance among all the robot pairs. More specifically, the average distance for a problem instance is measured by adding all the pairwise distances at every time step for all the runs to a vector and then diving by the size of the vector. Finally, the average distance is scaled by the robot diameter. As an example, a scaled distance of 5.14 indicates that the swarm is maintaining an average separation distance of roughly 5 robots. Small values (close to 1) indicate that the robots are too close to one another and large values indicate that the robots are separating. Standard deviations are shown for both time and scaled distance results.

Experiments are conducted on an Intel Core i3 machine (CPU: 2.40GHz, RAM: 4GB) using Ubuntu 13.04. Code is written in Python 2.7.3. Code is publicly available at [27].

3.2 Results

Fig. 2(a) provides a summary of the results on the average time for all the robots to reach the final destination. These results indicate that `CRoPS` is capable of effectively planning motions for large swarms moving through complicated environments. Fig. 2(b) takes a closer look at these results by showing how the average time scales as a function of the number of robots in the swarm. As the results indicate, the time grows only linearly. Such results provide promising validation on the scalability of `CRoPS`.

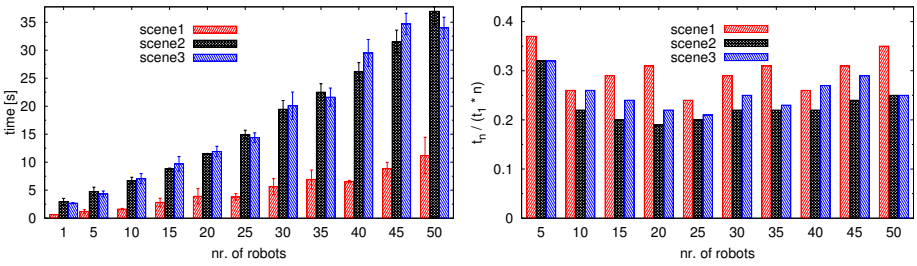


Fig. 2. (a) Results on the average time for all the robots to reach the final destination as a function of the number of robots. Bars indicate one standard deviation. (b) Scaled results, where t_n denotes the average time for all n robots to reach the final destination.

The efficiency of `CRoPS` derives from the combination of global path planning via probabilistic roadmaps with local path planning via potential fields. To test this further, `CRoPS` was run without the probabilistic roadmap. In this scenario, the robots would be guided by the potential fields and only be attracted to

the final destination but not to any intermediate goals. Without probabilistic roadmaps, however, the approach timed out and failed to send any robot to the final destination. These experiments indicate the importance of combining probabilistic roadmaps with potential fields when planning motions for large swarms in complicated environments.

Fig. 3 shows the average time difference between the first and the last robot to reach the final destination. The plot shows that the robots reach the final destination nearly at the same time even as the number of robots is increased. These results indicate that the robots remain together and move as a swarm.

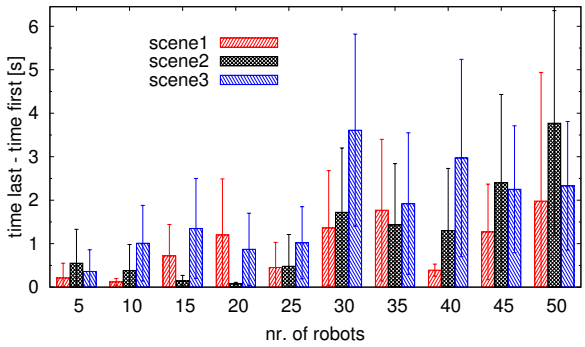


Fig. 3. Results on the average time difference between the first and the last robot to reach the final destination as a function of the number of robots. Bars indicate one standard deviation.

Fig. 4 shows the average scaled distance among all robot pairs (see Section 3.1). The scaled distance provides an indication of how close the robots are to one another. It is desirable that the robots are neither too close (as it could cause collisions or getting stuck in local minima) nor too far from each other (as it could cause some robots to get separated from the swarm). The results indicate that the robots maintain a desirable separation distance. Moreover, the separation distance changes very little even as the number of robots is increased.

4 Discussion

The proposed approach, `CRoPS`, combined probabilistic roadmaps with APFs in order to enable a swarm of robots to effectively move to a desired destination while avoiding collisions with obstacles and each other. The probabilistic roadmap provides global path planning to determine appropriate intermediate goals for the swarm. The potential fields provide local planning to enable the robots move together as a swarm towards the goal while avoiding collisions.

The combination of probabilistic roadmaps with APFs opens up several venues for future research. One research direction is to improve the interplay

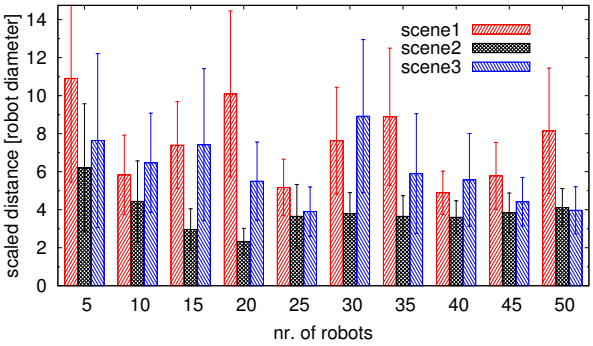


Fig. 4. Results on the average scaled distance among all robot pairs as a function of the number of robots. Scaling is done with respect to the robot diameter. Bars indicate one standard deviation.

between the roadmap planning and APFs in order to more effectively move the swarm to the desired destination. Another research direction is to accommodate moving obstacles. As the motion direction and velocity of the moving obstacles might not be known in advance, it will be important to be able to predict such motions and take them into account during planning.

References

1. Reynolds, C.W.: Flocks, herds and schools: A distributed behavioral model. In: ACM SIGGRAPH Computer Graphics. Volume 21. (1987) 25–34
2. Şahin, E.: Swarm robotics: from sources of inspiration to domains of application. In: International Conference on Swarm Robotics. (2004) 10–20
3. Brambilla, M., Ferrante, E., Birattari, M., Dorigo, M.: Swarm robotics: a review from the swarm engineering perspective. Swarm Intelligence (2012) 1–41
4. Reif, J.: Complexity of the mover's problem and generalizations. In: IEEE Symposium on Foundations of Computer Science. (1979) 421–427
5. Canny, J.: The Complexity of Robot Motion Planning. MIT Press, Cambridge, MA (1988)
6. Schwartz, J.T., Sharir, M.: A survey of motion planning and related geometric algorithms. Artificial Intelligence **37** (1988) 157 – 169
7. Khatib, O.: Real time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics Research **5**(1) (1986) 90–99
8. Reif, J.H., Wang, H.: Social potential fields: A distributed behavioral control for autonomous robots. Robotics and Autonomous Systems **27**(3) (1999) 171–194
9. Spears, W.M., Spears, D.F.: Physicomimetics: Physics-based swarm intelligence. Springer (2012)
10. Tanner, H.G., Kumar, A.: Formation stabilization of multiple agents using decentralized navigation functions. In: Robotics: Science and Systems. (2005) 49–56
11. Rimon, E., Koditschek, D.: Exact robot navigation using artificial potential functions. IEEE Tr. on Rob. and Autom. **8** (1992) 501–518

12. Kavraki, L.E., Švestka, P., Latombe, J.C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Transactions on Robotics and Automation **12**(4) (1996) 566–580
13. Denny, J., Amato, N.M.: Toggle PRM: A coordinated mapping of C-free and C-obstacle in arbitrary dimension. Volume 86 of Springer Tracts in Advanced Robotics. (2013) 297–312
14. Yeh, H.Y., Thomas, S., Eppstein, D., Amato, N.M.: UOBPRM: A uniformly distributed obstacle-based PRM. In: IEEE/RSJ International Conference on Intelligent Robots and Systems. (2012) 2655–2662
15. Plaku, E., Bekris, K.E., Chen, B.Y., Ladd, A.M., Kavraki, L.E.: Sampling-based roadmap of trees for parallel motion planning. IEEE Transactions on Robotics **21**(4) (2005) 587–608
16. Sun, Z., Hsu, D., Jiang, T., Kurniawati, H., Reif, J.: Narrow passage sampling for probabilistic roadmap planners. IEEE Transactions on Robotics **21**(6) (2005) 1105–1115
17. Boor, V., Overmars, M.H., van der Stappen, A.F.: The Gaussian sampling strategy for probabilistic roadmap planners. In: IEEE International Conference on Robotics and Automation. (1999) 1018–1023
18. Pan, J., Chitta, S., Manocha, D.: Faster sample-based motion planning using instance-based learning. Volume 68 of Springer Tracts in Advanced Robotics. (2013) 381–396
19. Aarno, D., Kragic, D., Christensen, H.I.: Artificial potential biased probabilistic roadmap method. In: IEEE International Conference on Robotics and Automation. (2004) 461–466
20. Katz, R., Hutchinson, S.: Efficiently biasing prms with passage potentials. In: IEEE International Conference on Robotics and Automation. (2006) 889–894
21. Bayazıt, O.B., Lien, J.M., Amato, N.M.: Swarming behavior using probabilistic roadmap techniques. In: Swarm Robotics. Springer (2005) 112–125
22. Bayazıt, O.B., Lien, J.M., Amato, N.M.: Better group behaviors using rule-based roadmaps. In: International Workshop on Algorithmic Foundations of Robotics. (2004) 95–112
23. Harrison, J.F., Vo, C., Lien, J.M.: Scalable and robust shepherding via deformable shapes. In: Motion in Games. Springer (2010) 218–229
24. Krontiris, A., Louis, S., Bekris, K.E.: Multi-level formation roadmaps for collision-free dynamic shape changes with non-holonomic teams. In: IEEE International Conference on Robotics and Automation. (2012)
25. Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L.E., Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press (2005)
26. LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge, MA (2006)
27. Wallar, A., Plaku, E.: Source code for CRoPS: Combined roadmaps and potentials for swarm path planning (2013) http://aw204.host.cs.st-andrews.ac.uk/CRoPS/.