Path Planning for Swarms in Dynamic Environments by Combining Probabilistic Roadmaps and Potential Fields

Alex Wallar Christopher Choi Erion Plaku

Abstract— **This paper presents a path-planning approach to enable a swarm of robots move to a goal region while avoiding collisions with static and dynamic obstacles. To provide scalability and account for the complexity of the interactions in the swarm, the proposed approach combines probabilistic roadmaps with potential fields. The underlying idea is to provide the swarm with a series of intermediate goals which are obtained by constructing and searching a roadmap of likely collision-free pathways. As the swarm moves from one intermediate goal to the next, it relies on potential fields to quickly react and avoid collisions with static and dynamic obstacles. Potential fields are also used to ensure that the swarm moves in cohesion. When the swarm deviates or is unable to reach the planned intermediate goals due to interferences from dynamic obstacles, the roadmap is searched again to provide alternative pathways. Experiments conducted in simulation demonstrate the efficiency and scalability of the approach.**

I. Introduction

Emerging applications of swarm robotics in exploration, monitoring, inspection, and search-and-rescue missions require the swarm to be able to move in cohesion to a goal destination while avoding collisions with obstacles. To provide scalability and account for the complexity of the interactions in the swarm, this paper builds upon prior work which combined probabilistic roadmaps with potential fields. While prior work was limited to static obstacles, the proposed approach can efficiently guide the swarm to the goal even in the presence of dynamic obstacles. The underlying idea is to provide the swarm with a series of intermediate goals which are obtained by constructing and searching a roadmap of likely collision-free pathways. As the swarm moves from one intermediate goal to the next, it relies on potential fields to quickly react and avoid collisions with static and dynamic obstacles. Potential fields are also used to ensure that the swarm moves in cohesion. When the swarm deviates or is unable to reach the planned intermediate goals due to interferences from dynamic obstacles, the roadmap is searched again to provide alternative pathways. Experiments conducted in simulation demonstrate the efficiency and scalability of the approach.

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].

A. Wallar and C. Choi are with the School of Computer Science, University of St Andrews, Fife KY16 9AJ, Scotland, UK. E. Plaku is with the Dept. of Electrical Engineering and Computer Science, Catholic University of America, Washington DC 20064 USA.

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 `dCRoPS` (`C`ombined `Ro`admaps and `P`otentials for `S`warms), which combines APFs with PRMs. While `dCRoPS` 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, `dCRoPS` 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. `dCRoPS` 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 `dCRoPS`. Experimental results in simulation with increasingly large swarms moving in complex environments containing numerous obstacles and narrow passages provide promising validation.

II. 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.

In order to guide the swarm to the final destination, a global path planner is used. A PRM is used to determine the shortest weighted path from the initial configuration to the final configuration. The roadmap for the PRM is constructed by taking by sampling collision free points and connecting collision free neighboring edges. The result of the PRM sampling is a graph that represents the connectivity of the configuration space. The vertices in the roadmap are weighted according to an estimate of how well the swarm will perform in that area of the environment. The edges are weighted according to the distance from the two vertices on a given edge. With these weights, the shortest path in the graph is computed using Dijkstra's algorithm. With the set of vertices that lie on the shortest path, we are able to give the boids a set of intermediate goals. By traversing the intermediate goals, the swarm will effectively reach the final destination. An attractive potential, denoted $PF_{igoal}(b)$, is applied for each boid to teh immediate goal. Once the robot reaches the immediate goal, the enxt subgoal in the shortest path is set as the new immediate goal.

In order to avoid collisions with obstacles, `dCRoPS` uses a strong repulsive potential field, denote das $PF_{obst}(b)$, which repulses each robot away from the obstacles. Since, `dCRoPS` is designed for dynamic obstacles, the responsiveness in the obstacle potential needs to also increase in order for the boids to react quickly to moving obstacles. The repulsive potential increases inversely with the square root of the distance to an obstacle. Using this function rather than the traditional inverse squared relationship is intended to increase the responsiveness of boids when confronted with dynamic obstacles by having a very quick change in the change of potential.

To maintain separation within the swarm, each robot is pushed away to some degree from its neighbors. There exists a weak sigmoidal repulsive potential field, deonted as $PF_{sep}(b)$ that governs the repulsion from other boids in the swarm. The reason to use a sigmoidal rather than a quadratic curve is to make sure that boid does get pushed away from other members of the swarm while keeping the swarm intact.

To move as a swarm, a boid's heading is influenced by the headings of its neighbors. The potential defining the influence is denoted $PF_{heading}(b)$. To encourage positive, swarm-like behavior, neighbors are chosen based on perfomance (i.e. if they are stuck or not) and based on their distance from b. Neighbors that are chosen are not too close or too far away and are not currently in a stuck state. This potential combined with the others described are combined and a weighted average is produced. This weighted average determines the new heading for the boid.

The different potential fields are then superimposed to obtain the overall heading for each robot. With this combination, there is a natural strong repulsion from obstacles, weak repulsion from other boids, an attraction to an immediate goal, and an influence from other boids who are performing well. The superpositioning of these fields makes it so the boids do not come too close to other boids or obstacles, move towards the immediate goal, and learn from boids who are progressing through the environment. The combination of the fields results in swarm like behavior. If the robot gets stuck in a local minima, the next few edges on the current path are weighted on the global roadmap, and the shortest path is recomputed. This local repair is used to releave swarm congestion when it is confronted by an obstructing obstacle. The creation of a new path may not always make the swarm move faster to the end goal, but it reduces the average wait time, the average time stuck, and it can guide the swarm around obstructing dynamic obstacles that are blocking critical passageways. 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.

A. Roadmap Construction

`dCRoPS` 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, `dCRoPS` uses the roadmap to find intermediate areas in which the swarm can move to effectively reach the goal.

1) 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, `dCRoPS` 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.

2) Roadmap Edges: After generating roadmap vertices, `dCRoPS` 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.

3) 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, `dCRoPS` 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.

B. Potential Fields

1) 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 $pos(b)$.

2) 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, `dCRoPS` 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.

3) Attraction to the Current Intermediate Goal: As discussed, `dCRoPS` 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 $\mathit{igoal}(b)$ is then defined as

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

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 $\mathit{igoal}(b)$, the next point q_{k+1} in the shortest path ζ is set as the new intermediate goal for b.

4) 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.,

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

and selecting as $\mathit{Neighs}(b)$ 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

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

5) 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.,

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

where $\mathit{prev}_\ell(b)$ 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.,

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

where r_x, r_y constitute a random direction. Note that $PF_{\mathit{escape}}(b) = (0, 0)$ 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 $\mathit{stuck}(b) = \texttt{true}$, a different mean μ_{stuck} and a different standard deviation σ_{stuck} are used to compute $\mathit{PF}_{\mathit{heading}}(b)$. 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, `dCRoPS` also adjusts the intermediate goals of a stuck robot. In particular, if q_k is the current intermediate goal, then `dCRoPS` 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.

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

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

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

$$\begin{aligned} \mathit{heading}(b) &\leftarrow w\mathit{heading}(b) + PF(b) \\ \mathit{pos}(b) &\leftarrow \mathit{pos}(b) + \mathit{heading}(b), \end{aligned}$$

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.

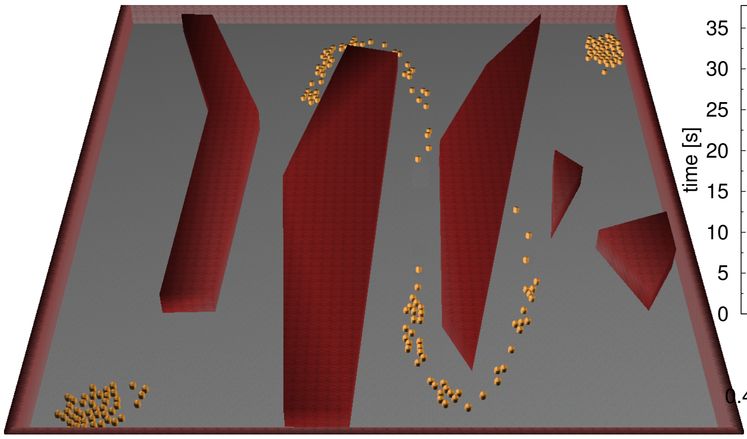
III. Experiments and Results

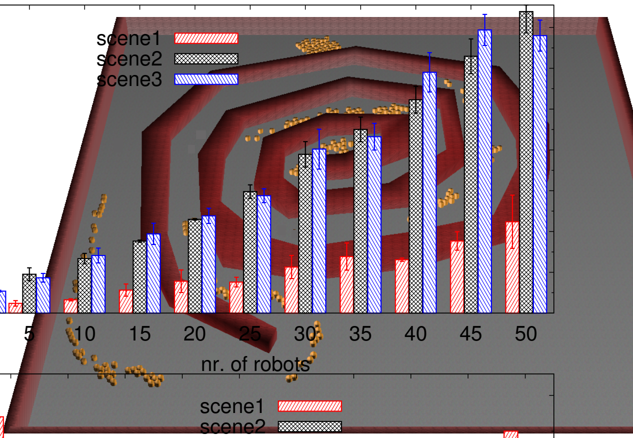
Experiments are conducted in simulation using different scenes and an increasing number of robots to test the efficiency and scalability of the approach. Fig. 1 provides an illustration of the scenes. These scenes provide challenging test cases as the swarm has to avoid numerous obstacles and pass through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

A. 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].


(a) scene1


(b) scene2

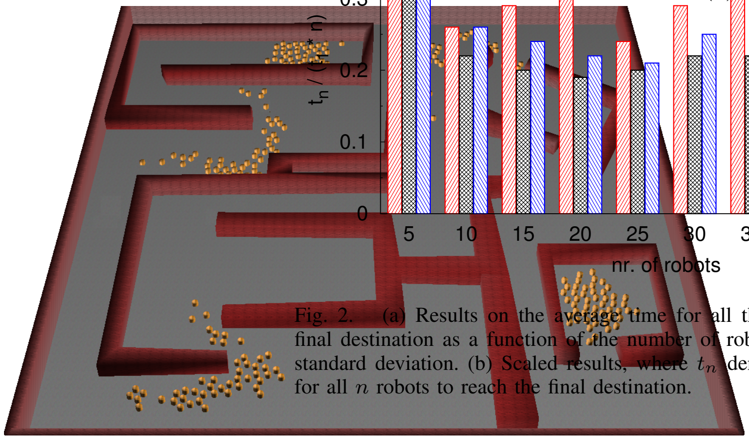
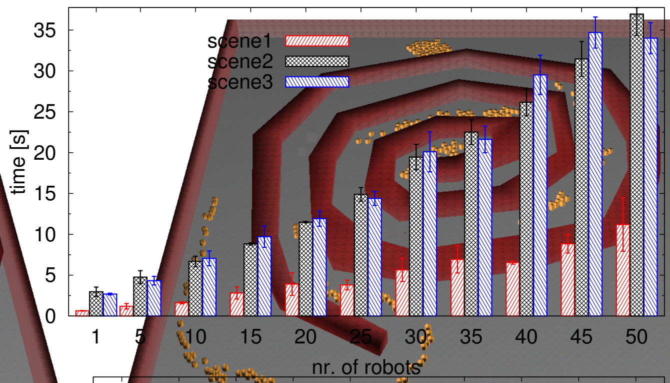

(c) scene3

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



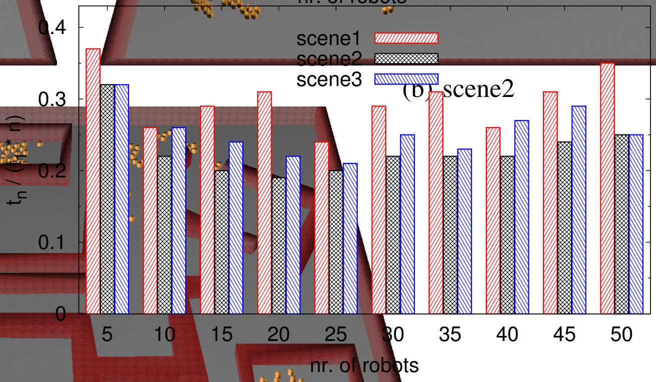


Fig. 2. (a) Results on the average time for all the robots to reach the final destination as a function of the number of robots. Bars indicate one standard deviation. (b) Scaled results, where t_n denotes the average time for all n robots to reach the final destination.

B. 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 dCRoPS 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 dCRoPS.

The efficiency of dCRoPS derives from the combination of global path planning via probabilistic roadmaps with local path planning via potential fields. To test this further, dCRoPS 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. 4 shows the average scaled distance among all robot pairs (see Section III-A). 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.

IV. Discussion

The proposed approach, dCRoPS, 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 between the roadmap

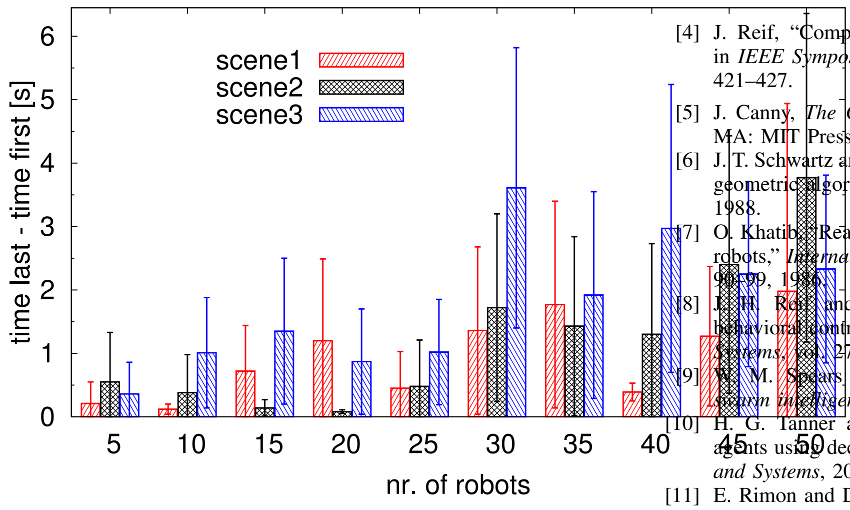


Fig. 3. Results on the average time difference between the first and the last robot to reach the final destination as a function of the number of robots. Bars indicate one standard deviation.

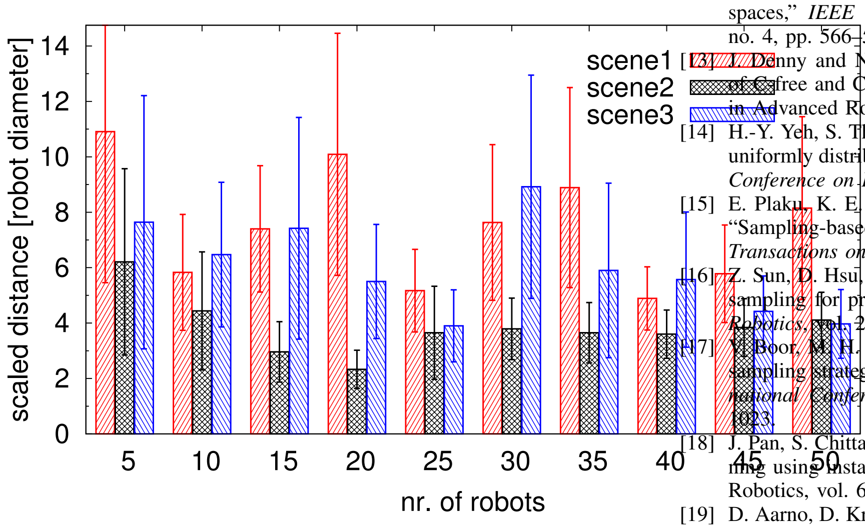


Fig. 4. Results on the average scaled distance among all robot pairs as a function of the number of robots. Scaling is done with respect to the robot diameter. Bars indicate one standard deviation.

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] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, 1987, pp. 25–34.

[2] E. Şahin, "Swarm robotics: from sources of inspiration to domains of application," in *International Conference on Swarm Robotics*, 2004, pp. 10–20.

[3] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: a review from the swarm engineering perspective," *Swarm Intelligence*, pp. 1–41, 2012.

[4] J. Reif, "Complexity of the mover's problem and generalizations," in *IEEE Symposium on Foundations of Computer Science*, 1979, pp. 421–427.

[5] J. Canny, *The Complexity of Robot Motion Planning*. Cambridge, MA: MIT Press, 1988.

[6] J. T. Schwartz and M. Sharir, "A survey of motion planning and related geometric algorithms," *Artificial Intelligence*, vol. 37, pp. 157 – 169, 1988.

[7] O. Khatib, "Real time obstacle avoidance for manipulators and mobile robots," *International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–99, 1986.

[8] J. H. Reif and H. Wang, "Social potential fields: A distributed behavioral control for autonomous robots," *Robotics and Autonomous Systems*, vol. 27, no. 3, pp. 171–194, 1999.

[9] W. M. Spears and D. F. Spears, *Physicomimetics: Physics-based swarm intelligence*. Springer, 2012.

[10] H. G. Tanner and A. Kumar, "Formation stabilization of multiple agents using decentralized navigation functions," in *Robotics: Science and Systems*, 2005, pp. 49–56.

[11] E. Rimon and D. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Tr. on Rob. and Autom.*, vol. 8, pp. 501–518, 1992.

[12] L. E. Kavraki, P. Švestka, J. C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[13] J. Denny and N. M. Amato, "Toggle PRM: A coordinated mapping of C-free and C-obstacle in arbitrary dimension," ser. Springer Tracts in Advanced Robotics, vol. 86, 2013, pp. 297–312.

[14] H.-Y. Yeh, S. Thomas, D. Eppstein, and N. M. Amato, "UOBPRM: A uniformly distributed obstacle-based PRM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 2655–2662.

[15] E. Plaku, K. E. Bekris, B. Y. Chen, A. M. Ladd, and L. E. Kavraki, "Sampling-based roadmap of trees for parallel motion planning," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 587–608, 2005.

[16] Z. Sun, D. Hsu, T. Jiang, H. Kurniawati, and J. Reif, "Narrow passage sampling for probabilistic roadmap planners," *IEEE Transactions on Robotics*, vol. 21, no. 6, pp. 1105–1115, 2005.

[17] V. Boor, M. H. Overmars, and A. F. van der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," in *IEEE International Conference on Robotics and Automation*, 1999, pp. 1018–1023.

[18] J. Pan, S. Chitta, and D. Manocha, "Faster sample-based motion planning using instance-based learning," ser. Springer Tracts in Advanced Robotics, vol. 68, 2013, pp. 381–396.

[19] D. Aarno, D. Kragic, and H. I. Christensen, "Artificial potential biased probabilistic roadmap method," in *IEEE International Conference on Robotics and Automation*, 2004, pp. 461–466.

[20] R. Katz and S. Hutchinson, "Efficiently biasing prms with passage potentials," in *IEEE International Conference on Robotics and Automation*, 2006, pp. 889–894.

[21] O. B. Bayazıt, J.-M. Lien, and N. M. Amato, "Swarming behavior using probabilistic roadmap techniques," in *Swarm Robotics*. Springer, 2005, pp. 112–125.

[22] ——, "Better group behaviors using rule-based roadmaps," in *International Workshop on Algorithmic Foundations of Robotics*, 2004, pp. 95–112.

[23] J. F. Harrison, C. Vo, and J.-M. Lien, "Scalable and robust shepherding via deformable shapes," in *Motion in Games*. Springer, 2010, pp. 218–229.

[24] A. Krontiris, S. Louis, and K. E. Bekris, "Multi-level formation roadmaps for collision-free dynamic shape changes with nonholonomic teams," in *IEEE International Conference on Robotics and Automation*, 2012.

[25] H. Choset, K. M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT Press, 2005.

[26] S. M. LaValle, *Planning Algorithms*. Cambridge, MA: Cambridge University Press, 2006.

[27] A. Wallar and E. Plaku, "Source code for CRoPS: Combined roadmaps and potentials for swarm path planning," 2013, http://faculty.cua.edu/plaku/CRoPS.zip.