Towards heterogeneous robot team path planning: acquisition of multiple routes with a modified spline-based algorithm

Roman Lavrenov* and Evgeni Magid

Intelligent Robotics Department, Higher Institute for Information Technology and Information Systems (ITIS), Kazan Federal University, 35 Kremlyovskaya street, Kazan, 420008, Russian Federation

Abstract. Our research focuses on operation of a heterogeneous robotic group that carries out point-to-point navigation in GPS-denied dynamic environment, applying a combined local and global planning approach. In this paper, we introduce a homotopy-based high-level planner, which uses a modified spline-based path-planning algorithm. The algorithm utilizes Voronoi graph for global planning and a set of optimization criteria for local improvements of selected paths. The simulation was implemented in Matlab environment.

1 Introduction

Currently, one of the actual problems of robotics deals with interaction between the unmanned ground robots (UGV) and unmanned aerial vehicles (UAVs). A heterogeneous group of robots, which work as a team, can significantly improve information acquisition about environment and solve various tasks of transportation logistics, reconnaissance, and search-and-rescue due to different sensory system features of the robots and different operational space [1-3].

Our research is targeted to investigate collaboration aspects between a heterogonous group comprising a UGV and a number of small-size UAVs, focusing on operation in uncertain and dangerous environments [4]. The latter could be completely unknown, include dynamic scene or objects, or represented by an outdated and imprecise map. To facilitate a reliable autonomous operation, robots will collaboratively perceive the surrounding environment and infer their own states, and plan high-quality actions and paths while taking different sources of uncertainty into account. In this research, we investigate a multi-robot collaborative framework, with the aim to advance the state of the art in multi-robot autonomy. Our objective is to develop collaboration strategies between robots equipped with various sensors such that these can autonomously operate given partial and possibly uncertain information regarding the environment. To that end, we will build upon recent progress in simultaneous localization and mapping (SLAM, [5]) and belief space planning [6], and explore multi-robot belief space planning approaches that take into account the uncertainty in the environment and are expected to facilitate efficient collaboration between robots. In particular, such a framework will allow UAVs and the

UGV to devise proper motion to improve localization and mapping even in lack of GPS signal.

2 Path planning approach

Path planning is the act of finding a path from some starting configuration S to goal configuration G. The objective of a path planner is to provide a continuous path from S to G, which will avoid all obstacles and minimizes some positive cost metric [7]. Classical path planning methods usually involve a global or a local path planner. A global path planner exploits a precise map of the environment, which is available to the planner a priori, and the path planning process generates a hazard-free path in the off-line mode; its main concern is to provide a computationally effective scheme for the particular cost metric (e.g., [8]). Global planning allows optimizing the solution with regard to various user-defined criteria in order to obtain a path, which is close to the optimal or at least sub-optimal path. Thus, the algorithm performance requirements are mainly dictated by such qualities as ability to generate a collision free path (soundness), time and space performance (complexity), and a guaranteed ability to discover a path if it does exist (completeness). However, in realistic settings, a robot cannot rely on complete a priori knowledge of the environment, but has to continuously collect sensory data from its potentially dynamic environment and plan accordingly. The two main sensor based approaches use either global or local path planning. On the opposite, a local path planner is a purely reactive on-line planner that collects sensory information and acts based on its analysis (e.g., [9]).

Each of these two classical approaches has its own strong and weak points and, while in the past robotics researchers were mainly developing them separately,

* Corresponding author: magid@it.kfu.ru

a more effective approach would combine them together in order to strengthen their advantages and compensate for the pitfalls (e.g., [10]). Moreover, the recent appearance of affordable swarm-type UAV and UGV robots on the market made it possible to develop a more reliable global (e.g., [11]) and local path planning (e.g., [12]) approaches by efficient use of multiple agents working toward a common goal.

In uncertain environments of urban search and rescue (USAR) or surveillance scenarios pure global path planning is of a little use because previously detailed maps become outdated and imprecise due to dynamic changes, while precise positioning becomes impossible due to the lack of GPS signal. However, often it is the case that some global data about the environment is still available: plans of the buildings, a bird-eye view map of the scene (e.g., obtained by a fleet of UGVs over the site) or abstract sketches by survivors. In such circumstances, the team of UAVs and UGVs could re-build the outdated map dynamically while moving toward the specified target and initially use this outdated map.

We are developing Homotopy based high level planner [13] that utilizes helpful global data of an outdated map and select pareto optimal trajectory classes based on user-tuned minimization criteria. Further, the selected class trajectory will be applied by the pilot system and actively combined with sensor based local path planning. The robotic team will create a detailed map and further update it on the fly as the team moves toward its target. The global re-planning will take place in real time taking into account dynamic changes of the environment and objective function, which contains various optimization criteria such as path length, safety, stealth, energy efficiency, path curvature, etc. combined together and allowing dynamic selection of coefficients for each function criterion. Some of such criteria are discussed in more details in the next section. To construct a global road map we apply Hierarchical Generalized Voronoi Graph method [14]. Next, the map is updated locally in real-time in order to introduce newly discovered obstacles [15] utilizing UAVs and UGV updates.

One of popular approaches for dynamical solution of a collision avoidance problem is a potential field approach [16]. It is based on an artificial potential field concept with an attractive pole at a target and obstacles that are represented with repulsive surfaces, while a robot follows the potential function gradient toward its minimum. The major advantages of potential field methods are simplicity and a capability of applying them reactively for obstacles avoidance. Its typical drawbacks include oscillations for certain obstacle setup and presence of local minima that attracts the robot and then keeps it captured inside unless the robot succeeds to detect the local nature of the minima and to execute some special escape procedure.

In [8] we had proposed an iterative path planning algorithm for a car-like mobile robot that simultaneously optimizes (collision free) path length

and smoothness. Moreover, while typical path planning algorithms apply path-planning smoothing in last stages only [17], our algorithm, which we briefly overview in Section IV, considered smoothness criterion from its first iterations. In order to improve the original algorithm performance, to add flexibility for path optimization and a possibility for simultaneous exploration with a team of UAVs that would afford a fast re-planning for UGV path in a case the initially off-line selected path becomes unavailable, we integrated Voronoi Diagram approach into our algorithm.

3 Cost function criteria selection

This section presents a number of valuable criteria for path optimization. Some particular criteria that are selected for our algorithm are introduced in more details through the prism of the spline-based navigation algorithm [8]. This algorithm navigates an omni-directional point robot in a planar known 2D configuration space environment populated with static obstacles. Each obstacle is presented with a finite set of intersecting circles of different sizes, assuming that any arbitrary obstacle could be well approximated with such set. Other criteria, which are not integrated into our path-planning algorithm, are only briefly overviewed in this section due to the lack of space.

Collision avoidance, which we further refer as path topology function is a basic requirement for any path to generate a collision free path. With potential function based approach, to provide a collision free path, a repulsive potential function should have a high value inside an obstacle (and on its boundary) and a small value in free space. Its high value in an obstacle center pushes a path outside in order to minimize path cost during local optimization procedure. The potential field begins to decrease drastically on obstacle's boundary, keeps decreasing with distance from the boundary and becomes zero in free space rather fast.

Assuming the robot's position at $q(t)=(x(t), y(t))$ at time t , a contribution of a single circle (which is a part of an obstacle) repulsive potential to a global potential function is defined with the equation:

$$U_{rep}(q) = 1 + \tanh(\alpha(\rho - R_{inf})) ,$$

$$R_{inf} = \sqrt{(x(t) - x_c)^2 + (y(t) - y_c)^2} , \quad (1)$$

where ρ is the radius of the obstacle with the center at (x_c, y_c) and α is an empirically defined parameter that is responsible for pushing a path outside of an obstacle. Figure 1 demonstrates the examples of two different selections of α parameter for the environment with two obstacles, shown in Figure 4. Figure 2 demonstrates the examples of $\alpha = 0.5$ and $\alpha = 0.2$ for the environment in Figure 5 with two complicated concave obstacles that are formed by finite sets of intersecting circles.

Fig. 1. Repulsive potential function of eq.1 for $\alpha = 0.5$ (left) and $\alpha = 0.2$ (right) that corresponds to the obstacle in Fig.4.

Fig. 2. Repulsive potential function of eq.1 for $\alpha = 0.5$ (left) and $\alpha = 0.2$ (right) that corresponds to the obstacles in Fig.5.

Topology function $T(q)$ takes into account all N obstacles of the environment and their influence on the robot along the whole path, which is defined as a parametric function within $[0,1]$:

$$T(q) = \sum_{j=0}^{N-1} \int_{t=0}^1 U^j_{rep}(q) \cdot \delta l(t) \cdot dt, \quad (2)$$

where $\delta l(t)$ is simply a length of a short segment:

$$\delta l(t) = \sqrt{(x'(t))^2 + (y'(t))^2}. \quad (3)$$

Path smoothness, which we further refer as path roughness function is responsible for absence of abrupt changes in the trajectory. It is particularly important for wheeled robots, and allows to maximize robot speed along the path and to avoid extreme accelerations and decelerations while following the trajectory. Roughness function $R(q)$ is also integrated along the path:

$$R(q) = \sqrt{\int_{t=0}^1 (x''(t))^2 + (y''(t))^2 dt}. \quad (4)$$

Path length is a rather typical criterion for all path-planning algorithms. It simply sums up the lengths of all path segments:

$$L(q) = \int_{t=0}^1 \delta l(t) \cdot dt. \quad (5)$$

Safety or minimal distance from obstacles criterion is responsible for preserving some minimal distance from obstacle boundaries [18]. While this feature is partially integrated into topology function, in hostile environments, there may be an explicit requirement to stay at least at some minimal distance from obstacle boundaries - these may happen in semi-structured debris of USAR scene, which risk to collapse further, or a danger of short-range emissions from the obstacles, etc. In such cases, the use of some paths that may be optimal with regard to other criteria should be completely forbidden due to violation of minimal distance requirement, and a different path should be selected. A minimal distance from all obstacles of environment is calculated in each robot

configuration $q(t)$ along the parametrically defined path:

$$m(C) = \min_{\forall t \in [0,1]} d_c(q(t)), \quad (5)$$

where $d_c(q(t))$ is a minimal distance from obstacles in configuration $q(t)$:

$$d_c(q(t)) = \min_{\forall c \in C} \sqrt{(x(t) - x_c)^2 + (y(t) - y_c)^2} - r_c, \quad (6)$$

where C is a set of all circular obstacles c with the center at (x_c, y_c) and radius r_c ; further, these elementary circular obstacles may intersect to form compound obstacles. The user is required to specify a particular minimally acceptable distance to the boundaries of any obstacle d_m , and then optimization criterion $D(q)$ is defined as follows:

$$D(d) = \omega^{d_m - m(C)}, \quad (7)$$

where ω is a sufficiently large empirically defined value that prevents a path from approaching obstacles beyond the permitted limit. Value of this function is one when minimal distance to obstacles $m(C)$ is equal to safe value d_m , while it gains large positive value when the limit is violated, and diminishes to a small positive value when the robot keeps safe distance from an obstacle.

Maximal distance from obstacles criterion is an opposite for safety criterion and it is responsible for preserving some maximally allowed clearance of a path with regard to obstacles. The situation of keeping obstacles at particular distance may become necessary when visual localization requires the obstacles that, for example, serve as environment features for SLAM to be within a limited sensor range.

Start and target point visibility time refers to the path length where the robot keeps the start point (or target point respectively) within its direct line of sight without any obstacle occlusions [19]. These two criteria are important if a robot needs to maximize time of a direct visual or radio contact with a monitoring device or a router, which supports path planning, localization or any other functionality of the robot. The criterion considers a ratio of visible and invisible from start S (or target T) points segments of a path:

$$I_S = 1 - \lim_{\delta t, \delta l(t) \rightarrow 0} \frac{\sum_{t=0}^u \text{dist}(A(t), A(t + \delta t))}{\int_{t=0}^1 \delta l(t) dt} \quad (8)$$

$$\forall t \in [0, u + \delta t] : [A(t), S] \cap \left(\bigcup_{j=1}^N \text{Obs}_j \right) = \emptyset.$$

Equation (10) defines start point S visibility time and the equation for target point T is similar.

Path passing time determines importance of passing a route in a fastest way. The value correlates with trajectory length, smoothness, and a number of other criteria.

If there are critical or danger points within a map [20] or, on the contrary, attraction points that always should be kept at a certain distance (e.g. for

communication routers), then we might set several additional parameters:

Maximum distance to critical points, which is necessary to be preserved in order to avoid losing contact between a robot and a monitoring device, e.g., while inspecting a map or following a path via control points that should stay within sensors range.

Safe path length rate, which is defined as a percentage of a sheltered from hazards path segments (i.e., their length). For example, if there is a radiation source or another source of danger in a critical point on a map, then a mobile robot should follow the safest way (e.g., behind obstacles for protection).

Minimum permitted distance to danger points is the minimal distance to a particular dangerous object or location, which a robot should always maintain.

4 Spline-based planning with potential field approach

Given a complete information about the environment, start S and target T positions of a robot, the robot searches for a collision-free path, which is guided by a pre-determined cost function. The original algorithm of spline-based path planning works iteratively [8]. An initial path is suggested as a straight line between S and T points, which defines a first spline with three points: S , T and an equidistant point that lies on the straight line between S and T . Equation 9 sets a path cost, and the path is further optimized with Nelder-Mead Simplex Method to minimize the cost:

$$F(q) = \gamma_1 T(q) + \gamma_2 R(q) + \gamma_3 L(q) \quad (9)$$

Fig. 3. Simple convex obstacles: the initial state (left), the final path after 6 iterations (right).

A resulting better path serves as an initial guess for the next iteration. The optimization procedure operates only with those points of the path, which define its spline, while path evaluation accounts for all points of the path. The path-defining spline is rebuilt at each iteration using information from a previous stage, increasing the number of spline's points by one and adjusting parameters of the target cost function. The algorithm terminates if iteration count exceeds a user-defined limit or a new iteration fails to improve a previous one. Figure 3 demonstrates an example of a successful execution of the original algorithm with cost function from eq. 9. The initial path was obtained in six iterations that took 9 minutes

and succeeded to reduce total cost of the path, while further increase of via points number did not provide any improvement. Further details of the algorithm could be found in [8].

The original spline-based method succeeds to obtain a collision free smooth path for any environment under assumption that each obstacle is approximated with a single circle, there are no circle intersections and some minimal distance between the obstacles is preserved. In this case, since the potential function diminishes rapidly as we move away from the obstacle boundary, we could neglect the probability of getting stuck in a local minima. When a number of circles intersect to form a compound concave obstacle, intersections introduce potential field local maxima. Moreover, if in such settings an initial spline passes through intersection of several obstacles, the cost function $F(q)$ priority is to push the spline out of the intersection area. Upon successful pushing out, a further solution often sticks in a local minima and a next iteration spline cannot overcome it due to a local nature of the optimization.

Fig. 4. Simple concave obstacle: the first iteration path (left) and the final path after 18 iterations (right).

Fig. 5. Two complicated concave obstacles: the first iteration path (left) and the final path after 37 iterations (right).

While in some cases the original algorithm succeeds to negotiate with concave obstacles (that are formed with a set of intersecting circles) as well, it is not a generic case. Figure 4 presents an example of two obstacles (the corresponding potential field is in Figure 1). The algorithm stops after 18 iterations as no further improvements are possible - while a spline succeeds to optimize the path locally so that it avoids a local maximum, it is still stuck at a local minimum and the resulting path could not be used for navigation due to obstacle collision. Figure 5 demonstrates an example with two complicated concave obstacles and a narrow pass in between. Due to multiple intersecting circles, which generate their own local repulsive

potentials, the corresponding global field conceals the pass (Figure 2) and permits only its local optimization. The local optimization successfully avoids local maxima, but even after 37 iterations that continued for 44 minutes, the final path is still occluded with obstacles and thus useless for navigation.

5 Voronoi diagram based solution for multi-robot exploration

We recognize that the local nature of the optimization procedure creates a strong dependence of algorithm success or failure on initial spline. In order to provide a good initial spline that could be further improved locally with regard to user selection of the cost weights, we apply Voronoi Diagram approach. Voronoi graph VG is constructed (Fig. 6, top left) based on a classical brushfire approach [14] and then VG nodes and edges, which are a part of some selected path from S to T, help to define an initial spline. Voronoi graph helps to avoid the spline seizing at local minima.

Fig. 6. Voronoi graph VG construction (top left): blue circles denote obstacles, blue lines are emitted at construction stage by obstacles rays, red lines form VG. Three other sub-figures demonstrate examples of path calculation within different homotopies: grey figures are obstacles, green lines are edges of VD, red dots are nodes of VG, red lines are the selected paths in different homotopies.

Voronoi graph VG contains multiple homotopy class paths and their variety depends on map complexity: the more distinct obstacles appear within the map, the larger is the amount of homotopy classes. For example, Figure 6 demonstrates three different paths that belong to different homotopy classes. Having multiple options for initial spline selection within various homotopies is of special interest for USAR: for example, USAR mission control centre has a city plan before an earthquake and thus could create a Voronoi graph VG offline for further use by a rescue robot. It is worth mentioning that off-line stage

does not limit us with valuable on-board resources, as it could be performed far in advance using high performance computational clusters. VG contains a large number of paths, one for each homotopy class. This feature provides a possibility for a simultaneous exploration of a number of prospective paths with a team of UAVs and affords discovering blocked passages that were previously available for a main rescue UGV. In turn, such approach permits a fast dynamic re-planning for UGV path within a different homotopy class in a case that the initially off-line selected path becomes unavailable.

We emphasize that it is not feasible to use all possible homotopies and most of them would obviously offer non-optimal paths. Thus, in our simulated experiments, we limit the algorithm to 5-7 homotopies. Next, these selected homotopies served as initial spline for a new smart spline-based method. We have tested this strategy with a number of environments.

6 Simulations and results

In order to verify our approach, the new algorithm was implemented in Matlab environment and an exhaustive set of simulations was performed. Particular attention was paid to the cases where the original algorithm fails. As a cost function we used the following equation:

$$F(q) = \gamma_1 T(q) + \gamma_2 R(q) + \gamma_3 L(q) + \gamma_4 D(d), \quad (10)$$

where the empirical parameter selection was: $\gamma_1=1$, $\gamma_2=1$, $\gamma_3=0.5$, $\gamma_4=1$, $d_m=3$ and $\omega=20$ (in eq. 7). We included additional safety criteria into original algorithm's cost function (eq. 9). The algorithm succeeded to provide collision-free paths in all cases, which was a natural consequence of applying initial Voronoi-based path as an input for iterative optimization algorithm.

We compute the cost function value (eq. 10) for each homotopy. As a result, optimization takes place in several homotopies simultaneously. For example, in Figure 7 the resulting path in the left image is more safe than the path in the right image. Since path optimization is performed only locally, the effect of the additional parameter(s) is also local. The path is constrained to remain within the same homotopy group even in the case when it is possible to bypass any obstacle from the other side.

Figure 7 environment was selected to demonstrate the advantage of the new algorithm with regard to the original algorithm. In such environment the original spline-based method failed to find an existing path within the map - the algorithm had spent 44 minutes and completed 18 iterations before reporting a failure. On the contrary, our new smart spline-based method succeeded to provide high quality paths for both initial spline selections from two different homotopies.

Fig. 7. Path planning results with a new spline-based algorithm: two corresponding optimal for a particular homotopy paths.

In the first homotopy example (Fig. 7, left), Voronoi graph VG calculation took only 5 seconds, and another 8 minutes were spent for two iterations of local optimization of the path with regard to user-defined criteria. In the second case, the same VG was utilized and a further optimization spent 21.5 minutes in order to complete the process within five iterations (Fig. 7, right).

7 Conclusions

Our research focuses on effective operation of a heterogeneous robotic group that could carry out point-to-point navigation in dynamic environments, applying a combination of local and global planning approaches. This paper presents a homotopy-based high-level planner that uses a modified spline-based iterative algorithm, which constructs Voronoi graph off-line and runs local path optimization with a potential function approach. Voronoi graph supports a selection of an initial spline, which is further optimized with regard to path length, curvature and safety criteria. Furthermore, Voronoi graph provides an opportunity to perform simultaneous exploration of several prospective paths with a team of UAVs varying paths from different homotopy classes. This way UAVs discover blocked passages that were previously available for a main UGV, and a fast dynamic re-planning of the UGV path within a different homotopy class could be performed.

The algorithm was implemented in Matlab environment and its results were explicitly compared to our original algorithm. The new approach requires less optimization iterations than the original algorithm due to a smart selection of an initial spline. While the original algorithm fails to find an existing path in complicated environments with concave obstacles, its smart version was successful in all simulated tests.

This work was partially supported by the Russian Foundation for Basic Research (RFBR) and Ministry of Science Technology & Space State of Israel (joint project ID 15-57-06010). Part of the work was performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

References

1. A. Rosenfeld, N. Agmon, O. Maksimov, A. Azaria, S. Kraus, *24th Int. Conf. on Artificial Intelligence*, 1902-1908 (2015)
2. A. Ronzhin, I. Vatamaniuk, N. Pavluk, *Int. Conf. and Exposition on Electrical and Power Engineering*, 675-680 (2016)
3. A.I. Panov, K. Yakovlev, *Robot Intelligence Technology and Applications*, 4, 3-20 (2017)
4. E. Magid, T. Tsubouchi, E. Koyanagi, and T. Yoshida, *Journal of Robotics and Mechatronics*, **23**(4), 567 (2011)
5. A. Buyval, I. Afanasyev, and E. Magid. *9th Int. Conf. on Machine Vision* (2016).
6. V. Indelman, L. Carlone, F. Dellaert, *The International Journal of Robotics Research* (2015)
7. A. Stentz, *Optimal and efficient path planning for unknown and dynamic environments* (1993)
8. E. Magid, D. Keren, E. Rivlin, I. Yavneh, *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2296-2301 (2006)
9. E. Magid, E. Rivlin, *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* (2004)
10. H. Zhang, J. Butzke, M. Likhachev, *IEEE Int. Conf. on Robotics and Automation* (2012)
11. M. Whitty, J. Guivant, *IEEE Int. Conf. on Robotics and Automation* (2011)
12. L. Rodriguez, J. A. Cobano, A. Ollero, *XV Workshop of physical agents: book of proceedings* (2014)
13. M. Kuderer, Ch. Sprunk, H. Kretzschmar, W. Burgard, *IEEE Int. Conf. on Robotics and Automation* (2014)
14. H. M. Choset, *Principles of robot motion: theory, algorithms, and implementation*. (MIT press, 2005)
15. B. Lau, and C. Sprunk, W. Burgard, *IEEE Int. Conf. on Intelligent Robots and Systems* (2010)
16. L. Tang, S. Dian, G. Gu, K. Zhou, S. Wang, X. Feng. *3rd IEEE Int. Conf. on Computer Science and Information Technology*, 9, 633-637 (2010)
17. S. Fleury, P. Soueres, J.-P. Laumond, R. Chatila, *IEEE Transactions on robotics and automation*, **11**(3), 441-448 (1995)
18. R. Lavrenov, E. Magid, *2nd Int. Con. on Interactive Collaborative Robotics*, LNCS, Springer (2017, to be published)
19. E. Magid, R. Lavrenov, A. Khasianov, *Int. Conf. on Informatics in Control, Automation and Robotics*, MATEC Web of Conferences (2017, to be published)
20. E. Magid, R. Lavrenov, I. Afanasyev, *Int. Conf. on Mechanical, System and Control Engineering*, MATEC Web of Conferences (2017, to be published)