# Modified Spline-Based Navigation: Guaranteed Safety for Obstacle Avoidance

Roman Lavrenov[1(✉)], Fumitoshi Matsuno[2], and Evgeni Magid[1]

[1] Kazan Federal University, Kazan 420008, Russian Federation
{lavrenov,magid}@it.kfu.ru
[2] Kyoto University, Kyoto 615-8540, Japan
matsuno@me.kyoto-u.ac.jp

**Abstract.** Successful interactive collaboration with a human demands mobile robots to have an advanced level of autonomy, which basic requirements include social interaction, real time path planning and navigation in dynamic environment. For mobile robot path planning, potential function based methods provide classical yet powerful solutions. They are characterized with reactive local obstacle avoidance and implementation simplicity, but suffer from navigation function local minima. In this paper we propose a modification of our original spline-based path planning algorithm, which consists of two levels of planning. At the first level, Voronoi-based approach provides a number sub-optimal paths in different homotopic groups. At the second, these paths are optimized in an iterative manner with regard to selected criteria weights. A new safety criterion is integrated into both levels of path planning to guarantee path safety, while further optimization of a safe path relatively to other criteria is secondary. The modified algorithm was implemented in Matlab environment and demonstrated significant advantages over the original algorithm.

**Keywords:** Path planning · Safety · Potential field · Voronoi graph

## 1 Introduction

Contemporary robotic applications target for human replacement in diverse scenarios that spread from social-oriented human-robot interaction [11] and collaboration [13] to automatic swarm control [12] and urban search and rescue in hostile environments [9]. All such applications demand indoor and outdoor autonomous path planning and navigation abilities with simultaneous localization and mapping (SLAM) [2], collaboration with other robots [10] and other functionality.

Path planning distinguishes global and local approaches. While the later operates in a completely unknown environment and robots make immediate decisions that are based on locally available information only, the global approach can access complete knowledge about environment, i.e., robot shape, initial and


A. Ronzhin et al. (Eds.): ICR 2017, LNAI 10459, pp. 123–133, 2017.
DOI: 10.1007/978-3-319-66471-2_14

goal postures, and a set of environment obstacles are known in advance. Within a global approach model, potential field based methods [14], [4] could provide a globally defined potential function such that a goal position is represented as an attractive pole and obstacles are represented with repulsive surfaces. Next, a robot follows the potential function gradient toward its minimum [1]. Two main problems of potential field methods are oscillations in narrow passages between obstacles and a failure to select a good global potential function, which in turn results in local minima issues.

Our previous research had proposed a global approach path planning spline-based algorithm for a car-like mobile robot [7]. It uses potential field for obstacle avoidance and provides a locally sub-optimal path with regard to path length, smoothness and safety optimization criteria. In this paper we propose a modification of our original algorithm, which targets to improve robot safety with regard to environment obstacles. The path planning is performed in two stages: first, Voronoi-based approach provides a number sub-optimal paths in different homotopic groups; next, one of these paths is optimized in an iterative manner with regard to selected criteria weights. The algorithm integrates safety criteria into both levels of path planning in order to guarantee path safety, while further optimization of a safe path relatively to other path evaluation criteria is secondary. The new algorithm was implemented in Matlab environment and demonstrated significant advantages over the original algorithm.

The rest of the paper is organized as follows. Section 2 briefly describes our previous research. Section 3 presents a new criterion that improves the algorithm performance with regard to robot safety, and our modified spline-based algorithm, which successfully overcomes the weaknesses of the initial approach and uses advantages of the new optimization criterion. Section 4 compares the results of the original and the new algorithm, demonstrating the superiority of the later. Finally, we conclude in Sect. 5.

## 2 Original Spline-Based Navigation Algorithm Drawbacks

In our previous research [7] we had proposed a spline-based method that navigates an omnidirectional circle-shape robot in a 2D configuration space with known a-priori static obstacles. Each obstacle is approximated with a finite set of intersecting circles of different size. Given a start and a target positions, the robot searches for a collision-free path being guided by a cost function with pre-determined weights of each criteria.

Collision avoidance is managed with a repulsive potential function with a high value inside of a circular obstacle and a small value in free space. High values of the function push points of a path outside obstacles to minimize path cost during optimization. The potential field begins to decrease drastically on obstacle boundary and wanes rapidly with distance while moving away from the boundary. A contribution of a single circular obstacle repulsive potential at robot position $q(t) = (x(t), y(t))$ in time $t$ is described with the following equation:

$$U_{rep}(q) = 1 + \tanh(\alpha(\rho - \sqrt{(x(t) - x)^2 + (y(t) - y)^2})) \quad (1)$$

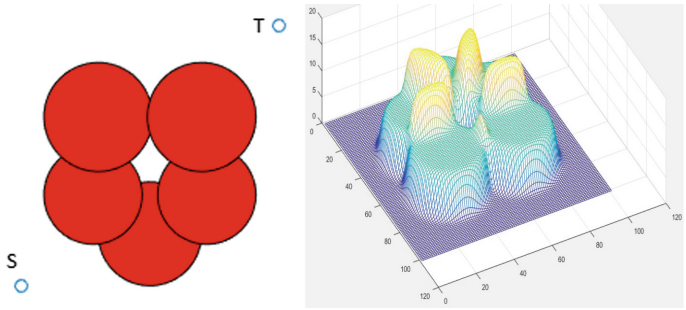


**Fig. 1.** Example of environment with five obstacles and start/target points (left) and repulsive potential function of Eq. 1 for $\alpha = 0.5$ (right) that corresponds to the obstacles

where $\rho$ is the radius of the obstacle with centre $(x, y)$ and $\alpha$ is an empirically defined parameter that is responsible for pushing a path outside of an obstacle. Figure 1 (right) demonstrates the example with $\alpha = 0.5$ for a single obstacle that is formed by five intersecting circles (Fig. 1, left), and the potential function has undesirable peaks at the circle intersections. Next, influence of all $N$ obstacles of the environment are accumulated into *topology* $T(q)$ parametric function that is defined within $[0, 1]$:

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

where $\delta l(t)$ is a segment length. Smoothness property of the path is considered through *roughness* $R(q)$ function and is also integrated along the path:

$$R(q) = \sqrt{\int_{t=0}^{1} (x''(t))^2 + (y''(t))^2 dt} \tag{3}$$

And the path length $L(q)$ sums up the lengths of all path segments:

$$L(q) = \int_{t=0}^{1} \delta l(t) \cdot dt \tag{4}$$

The final path cost function sums up the three components with empirically predefined weights $\gamma_{i=1\ldots3}$:

$$F(q) = \gamma_1 T(q) + \gamma_2 R(q) + \gamma_3 L(q) \tag{5}$$

For example, obstacle penalty influence component is defined as $\gamma_1 = \frac{\beta}{2}$, where $\beta$ ranges over an array that correlates with array of $\alpha$ parameters from Eq. 1 [6].

The original algorithm works iteratively, starting from a straight line (between $S$ and $T$ points) initial path. This line forms a first spline that utilizes three points: $S$, $T$ and an equidistant point in between. Equation 5 sets the

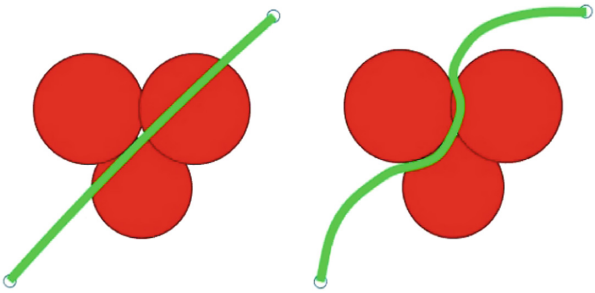

**Fig. 2.** Compound obstacle: first iteration (left) and final path after 18 iterations (right)

path cost, which is further optimized with Nelder-Mead Simplex Method [5] to minimize the total cost. A resulting *better* path serves as an initial guess for the next iteration. The optimization deals only with the points which define the spline, while path evaluation accounts for all points of the path. The spline is rebuilt at each iteration using information from a previous stage, increasing the number of spline's points by one. The algorithm terminates when number of iterations overflow a user-defined limit or if a new iteration fails to improve a previous one.

The original method is successful for simple obstacles that could be approximated with a single circle, while for compound obstacles a good selection of initial path becomes essential. Compound obstacles that consist of intersecting circles introduce potential field local maxima at each intersection (e.g., Fig. 1, right), which may trap a path in cost function local minimum as a next iteration spline can not overcome local maxima due to a local nature of the optimization process. Figure 2 demonstrates an example with one compound obstacle; three intersecting circles produce local repulsive potentials that sum up in a such way that an optimization succeeds to avoid local maxima, but stops after 18 iterations and reports a failure of providing a collision free path (Fig. 2, right).

## 3 Voronoi Graph Based Solution with Integrated Safety

We propose a modification of our original spline-based path planning algorithm for a mobile robot. The new algorithm consists of two levels of path planning. At the first level, Voronoi-based approach provides a number sub-optimal paths in different homotopic groups. At the second level, one of these paths is optimized in an iterative manner with regard to selected criteria weights. The algorithm integrates safety criteria into both levels of path planning in order to guarantee path safety, while further optimization of a safe path relatively to other path evaluation criteria is secondary.

### 3.1 Minimal Distance from Obstacles Criterion

First, we introduce an additional path quality evaluation criterion - minimal distance from obstacles. While navigating in obstacle populated environments, a robot should maximize its distance from the obstacles, and this feature is integrated into T(q) component of our cost function. However, in hostile environments, there may be an explicit requirement to stay at least at some minimal distance from obstacle boundaries - these may include semi-structured debris of urban 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 Eq. 5 should be forbidden due to violation of minimal distance requirement, and a different path should be selected. A minimal distance of a robot from all obstacles of environment is calculated in each configuration $q(t)$ along the parametrically defined path as follows:

$$m(C) = \min_{\forall t \in [0,1]} dist_c(q(t)) \tag{6}$$

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

$$dist_c(q(t)) = \min_{\forall c \in C} \sqrt{(x(t) - x(c))^2 + (y(t) - y(c))^2} - r(c) \tag{7}$$

Here $C$ is a set of all circular obstacles $c$ with the centre at $(x(c), y(c))$ and radius $r(c)$; further, these elementary circular obstacles may intersect to form compound obstacles. Also the user is required to specify a particular minimally acceptable distance to the boundaries of any obstacle $d_m$, and them optimization criterion D(q) is defined as follows:

$$D(q) = \omega^{d_m - m(C)} \tag{8}$$

where $\omega$ 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 $d_m$, but 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. We combine all four criteria within the total cost function for the optimization procedure as follows:

$$F(q) = \gamma_1 T(q) + \gamma_2 R(q) + \gamma_3 L(q) + \gamma_4 D(q) \tag{9}$$

where $\gamma_4$ is the weight for minimum distance criteria influence. Figure 3 demonstrates the criterion influence on the path: while with $\gamma_4 = 0$ it does not contribute to the total cost (left sub-figure), with $\gamma_4 = 1$, $d_m = 3$ and $\omega = 20$ the path avoids touching the obstacles (right sub-figure); other parameters are defined as $\gamma_1 = 1$, $\gamma_2 = 1$, and $\gamma_3 = 0.5$ for the both cases.

### 3.2 Selecting a Path with Voronoi Graph

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

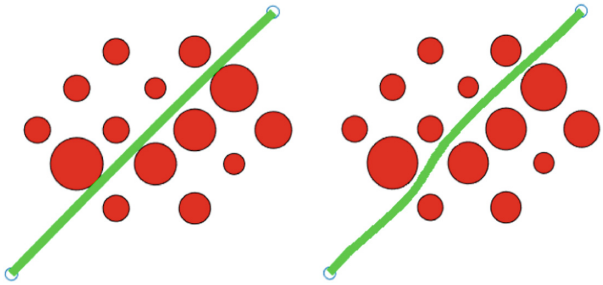

**Fig. 3.** The path moves away from obstacles when we use the new criterion

approach [15]. This helps to avoid the drawbacks of original algorithm, which is caused by a poor selection of an initial spline. The previously mentioned first level of path planning includes three stages: preparing environment, constructing Voronoi graph $VG$ that spans the environment, and selecting a number of sub-optimal paths within $VG$ with regard to a single evaluation criterion from Eq. 9.

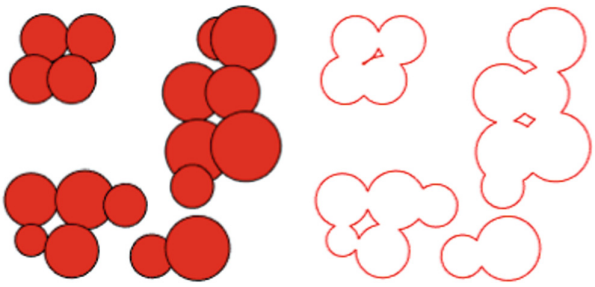

**Fig. 4.** Compound obstacles (left) and their external contours (right)

To prepare the environment, two steps are performed. First step groups intersecting circles $c \in C$ together in order to form a single compound obstacle $O_1$ through iterative growth. It continues to form such obstacles $O_i, i = 1 \ldots k$, where $k$ is a number of compound obstacle within the environment, until all circles of the environment will be assigned to a particular compound obstacle. For example, there are four obstacles that are formed by groups of circles in Fig. 4 and three in Fig. 5. The second step shapes outer boundaries of each compound obstacle $O_i$ of set $Obst = \{O_1, O_2, \ldots, O_k\}$. For example, this successfully removes three tiny internal contours inside compound obstacles in Fig. 4.

Next, Voronoi graph $VG$ is constructed based on a classical brushfire approach [3]. Figure 5 (the middle image) demonstrates Voronoi graph $VG$ construction example for the environment in Fig. 5(left). Thick blue lines are compound obstacles borders and thin blue lines depict emerging outwards and inwards rays. The rays intersection points are equidistant to nearest obstacles, and all together

form $VG$, which is depicted with a thin red line in Fig. 5 (middle and right). At this stage we apply safe value $d_m$ in order to remove from $VG$ all edges that already do violate the safety requirements. Such example is demonstrated in Fig. 7 and will be explained in more details in the next section.

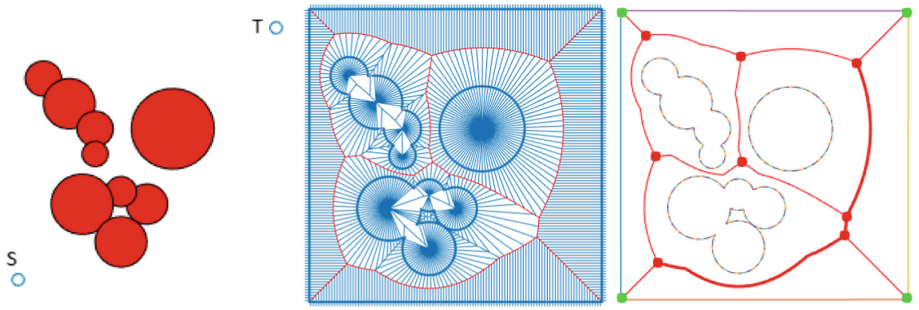


**Fig. 5.** Environment with obstacles (left), Voronoi graph building procedure (middle), the obtained Voronoi graph and the path within the graph (right) (Color figure online)

Finally, upon obtaining Voronoi graph $VG$, we finalize it by adding start $S$ and target $T$ nodes and perform a search of several sub-optimal paths within $VG$ with Dijkstra algorithm; they are depicted with thick red lines in Fig. 5 (right). Next, a set of spanning points is extracted from this spline candidate and these points are utilized as via points for initial spline of the spline-based method [7]. As the path optimization with regard to Eq. 9 is performed only locally, the influence of additional parameter is also local. At the second level of path planning, a selected at the first level path is optimized in an iterative manner with a help of Eq. 9. More technical details about graph construction and spanning points selection could be found in [8].

## 4 Simulation Results

In order to verify our approach the new algorithm was implemented in Matlab environment and an exhaustive set of simulations was run. Particular attention was paid to the cases where the original algorithm fails [7]. The cost function of Eq. 9 was applied with empirical parameter selection $\gamma_1 = 1,\ \gamma_2 = 1,\ \gamma_3 = 0.5,\ \gamma_4 = 1,\ d_m = 3$ and $\omega = 20$. 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.

Figure 6 demonstrates environment, where the original spline-based algorithm had failed. At the first level, Voronoi graph $VG$ provides us with a safe path without obstacle collisions. It serves as an initial path at the second level, which ensures a final path calculation with the modified spline-based algorithm

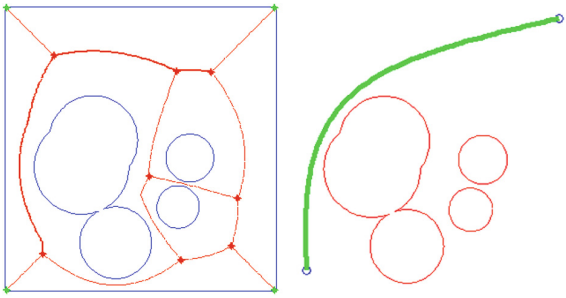

**Fig. 6.** Voronoi graph path (left) and corresponding spline-based optimal path (right)

within a significantly smaller number of iterations. Even though potentially significant time complexity of $VG$ construction was not considered as an issue due to its off-line construction, the simulations empirically demonstrated that $VG$ calculations take acceptably small amount of time for simple cases, while more simulations in complicated large-size environments are scheduled for the future.

For example, for the environment in Fig. 6 the Voronoi-based initial path calculation took only 2 s. The total running time of the new algorithm decreased in three times in average with regard to the original algorithm. This way, the final path in Fig. 6 was calculated in just 2 iterations within 2.5 min in Matlab, while the original spline-based algorithm had spent 18 iterations and 38 min to report its failure to provide a collision free path. Similarly, the original algorithm required 9 iterations and 15 min to provide a good path within Fig. 3 environment, while the new algorithm required 4 iterations and 5 min. In Fig. 4 the original algorithm failed to find a path, while the new algorithm successfully completed the task within 3 iteration and 2 min.

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. In our simulated experiments we decided to limit the algorithm to no more than 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. For each selected homotopy we verify minimal distance value $d_m$ within a corresponding equidistant to obstacles edges and reject homotopies that pass in dangerous proximity of obstacles. The calculation of all suitable homotopies takes 1 to 3 min in average, depending on environment complexity. In Fig. 7 for $d_m = 3$, safe VD edges are depicted with green and forbidden edges in red colour (Fig. 7, left). Thus, only the homotopies that contain green edges could be considered for safe path candidates. After the first level of planning provided a candidate path, second level optimal path (Fig. 7, right) calculation took just 3 iteration and 2 min.

Figure 8 demonstrates an example where original spline-based method spends 32 min and 18 iterations before reporting a failure. The new algorithm calculates

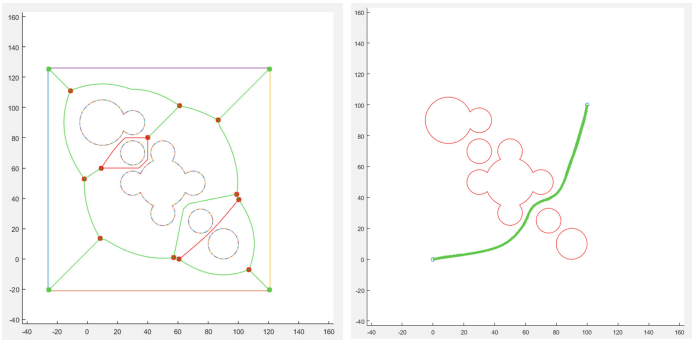

**Fig. 7.** Voronoi graph with dangerous edges (left) and a resulting safe path (right) (Color figure online)

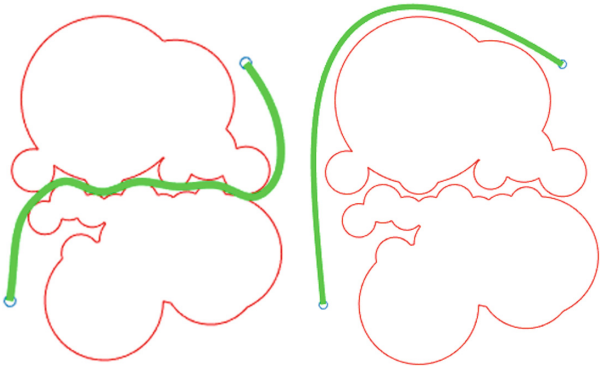

**Fig. 8.** Path with (right) and without minimal distance criterion (left)

5 homotopies within 3 min (Fig. 8 demonstrates paths in 2 different homotopy classes) and bases its selection on safety criterion (Fig. 8, right).

## 5 Conclusions

In this paper we presented a modification of our original spline-based path planning algorithm, which consists of two levels of planning. At the first level, Voronoi-based approach provides a number sub-optimal paths in different homotopic groups. At the second, these paths are optimized in an iterative manner with regard to selected criteria weights. A new safety criterion is integrated into both levels of path planning to guarantee path safety, while further optimization of a safe path relatively to other criteria is secondary. The modified algorithm was implemented in Matlab environment and demonstrated significant advantages over the original algorithm, including guaranteed path acquisition, successful avoidance of local minima problem, increased speed and guaranteed path safety in static planar environment. As a part of our future work, the algorithm

will be tested in large-size environments in order to verify the acceptability of the Voronoi graph construction time at the first level of path planning. We also consider extending the path cost function with additional optimization criteria and perform exhaustive testing of criteria weight selection.

**Acknowledgments.** 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. Andrews, J.R., Hogan, N.: Impedance control as a framework for implementing obstacle avoidance in a manipulator. M. I. T., Department of Mechanical Engineering (1983)
2. Buyval, A., Afanasyev, I., Magid, E.: Comparative analysis of ROS-based monocular SLAM methods for indoor navigation. In: Proceedings of SPIE, 9th International Conference on Machine Vision, vol. 10341, pp. 103411K–103411K-6 (2016)
3. Choset, H.M.: Principles of Robot Motion: Theory, Algorithms, and Implementation. MIT Press, Cambridge (2005)
4. Khatib, O., Siciliano, B.: Springer Handbook of Robotics. Springer, Heidelberg (2016)
5. Lagarias, J.C., Reeds, J.A., Wright, M.H., Wright, P.E.: Convergence properties of the Nelder-Mead simplex method in low dimensions. SIAM J. Optim. **9**(1), 112–147 (1998)
6. Magid, E.: Sensor-based robot navigation. Technion - Israel Institute of Technology (2006)
7. Magid, E., Keren, D., Rivlin, E., Yavneh, I.: Spline-based robot navigation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2296–2301 (2006)
8. Magid, E., Lavrenov, R., Khasianov, A.: Modified spline-based path planning for autonomous ground vehicle. In: Proceedings of International Conference on Informatics in Control, Automation and Robotics (to appear)
9. Magid, E., Tsubouchi, T., Koyanagi, E., Yoshida, T.: Building a search tree for a pilot system of a rescue search robot in a discretized random step environment. J. Robot. Mechatron. **23**(4), 567–581 (2011)
10. Panov, A.I., Yakovlev, K.: Behavior and path planning for the coalition of cognitive robots in smart relocation tasks. In: Kim, J.-H., Karray, F., Jo, J., Sincak, P., Myung, H. (eds.) Robot Intelligence Technology and Applications 4. AISC, vol. 447, pp. 3–20. Springer, Cham (2017). doi:10.1007/978-3-319-31293-4_1
11. Pipe, A.G., Dailami, F., Melhuish, C.: Crucial challenges and groundbreaking opportunities for advanced HRI. In: IEEE/SICE International Symposium on System Integration, pp. 12–15 (2014)
12. Ronzhin, A., Vatamaniuk, I., Pavluk, N.: Automatic control of robotic swarm during convex shape generation. In: International Conference and Exposition on Electrical and Power Engineering, pp. 675–680 (2016)
13. Rosenfeld, A., Agmon, N., Maksimov, O., Azaria, A., Kraus, S.: Intelligent agent supporting human-multi-robot team collaboration. In: International Conference on Artificial Intelligence, pp. 1902–1908 (2015)

14. Tang, L., Dian, S., Gu, G., Zhou, K., Wang, S., Feng, X.: A novel potential field method for obstacle avoidance and path planning of mobile robot. In: IEEE International Conference on Computer Science and Information Technology, vol. 9, pp. 633–637 (2010)
15. Toth, C.D., O'Rourke, J., Goodman, J.E.: Handbook of Discrete and Computational Geometry. CRC Press, Boca Raton (2004)