Multi-Path Planning for Autonomous Navigation of Multiple Robots in a Shared Workspace with Humans

Lan Anh Trinh, Mikael Ekström and Baran Cürüklü

***Abstract*— Path finding for multiple robots is one of most important problems in robotics when to find a way to move robots from their starting positions to reach their respective goals without collisions. However, in the case of a complex environment with the presence of humans and other unpredictable moving objects, fixing a single path to the goal may lead to a situation where there are a lot of obstacles on the planned path and the robots may fail to realise the moving plan. To address this issue, a new approach of using multiple path planning where each robot has different options to choose its path to the goal is introduced in this paper. The information about planned moving paths are shared among the robots in the working domain, combined with obstacle avoidance constraints in local ranges, and formulated as an optimisation problem. Solution of the problem leads to the optimal moving plans of robots. The effectiveness of the proposed approach is demonstrated by experimental results.**

***Keywords*— multiple path planning; multiple robot; autonomous navigation; obstacle avoidance**

I. INTRODUCTION

Path planning and obstacle avoidance are important components of robotic navigation. Advances in key technologies, in combination with public acceptance, have opened the way towards allowing several autonomous robots coexist with humans in unstructured environments. This assumes autonomous navigation. One of the challenges in this regard is handling navigation failures of multiple-robots when they are operating together in a shared working space, which is also complex and cluttered. To avoid complete failures, the robots should have recovering mechanisms so that they are able to come back to their normal activities. In this context, the complete failures happen when the robots stop working and cannot finish their navigation tasks. The common procedure assumes that the global path planning searches for a path, from a start to a goal, through an empty space within a map of static obstacles. The local obstacle avoidance drives the robot to follow the planned global path while taking into account possible collisions with other robots and dynamic obstacles.

Relying on a single and a fixed global plan could lead to a deadlock, or livelock situation where a robot take a very long time to reach its goal, or will not even be able to do so. This could happen in the case of multiple robots moving in a narrow area, with respect to the size of the robots e.g., 2x2 of a robots diameter. Since the local navigation to avoid obstacles only takes into account the collision with other robots within a close range, a robot must turn back to the configured path to be able to reach its goal. However, if two robots are routed through a very narrow area, like a corridor, and enter it through two opposite sites, the robots may face a situation where they repeat the same moving trajectories within that area again and again without ever finding the path to the goal.

Factory workshops, and other industrial spaces (also outdoor), have emerged as important cases for navigation of multiple robots. In this case the robots transport objects between different stations, thus shared the same space with humans. In the most common setup, automated guided vehicles (AGVs) are deployed. They are configured with a predefined moving path. Due to safety reasons, the operation of an AGV is terminated when a human enters the working zones or crosses the moving trajectory of a vehicle. Yet, replacing these systems with robots that do not follow a predefined path, and have autonomous path planning, have the potential of allowing more flexible solutions, in presence of humans and other moving objects also in unfamiliar environments. During the process of controlling the paths of multiple robots, by sharing location information to each other, robots are able to avoid collisions by moving toward any open area without interfering with trajectories of others. However, the presence of humans introduces uncertainty, since robots are not able to know the intended movement of the humans. To deal with such uncertainty, the footprint i.e., the representation of human obstacles are enlarged with respect to the probability of uncertainty, or to the safety level, to prevent the collision of human with robot. A big footprint combined with a unpredictable trajectory of a human could increase the chances of blocking all feasible moves of the robot to realise the defined global moving plan.

From the described scenarios above, it is evident that relying on a single path planning could lead to a navigation failure when there is no feasible way to implement the path due to the obstacle avoidance function. Therefore it is important for a robot to have alternative paths to reach its goal and the robot must be able to proactively switch among solutions whenever necessary to prevent deadlock situations.

In this paper, a new navigation system with multi-path planning is introduced. Each robot is able to frequently establish multiple paths from its current position to reach the defined goal. All robots in the working domain share their sets of possible planned paths to each other via a communication channel. Consequently, an optimisation problem is formulated to find the next move of the robots with respect to the constraints, which are to ensure no interference between

*School of Innovation, Design, and Technology, Mälardalen University, Västerås, Sweden `anh.lan@mdh.se`, `mikael.ekstrom@mdh.se` and `baran.curuklu@mdh.se`

the planned paths among different robots and no collisions between robots and other moving obstacles. In overall, the proposed multi-path planning algorithm presents an effective mechanism for fault tolerance to recover the robot activities to handle up to some levels of failures of the navigation system.

The rest of the paper is organised as follows. Section 2 presents related works. Section 3 describes the methodology of the proposed approaches. Section 4 provides experimental results for evaluation. Finally, Section 5 concludes the paper with discussion.

II. RELATED WORKS

A. *Multiple Path Planning*

The multi-agent path finding (MAPF) problem has been introduced to find collision-free paths for multiple robotic agents from starting positions to their goals. A MAPF algorithm considers multiple paths for each agent and searches for a path to optimise a criteria function like a minimum total traveling distance. However, a MAPF algorithm is mainly suitable for well-defined environment without unpredictable obstacles.

For the graph-based solutions, the robots move on a connected graph from a vertex to its neighbors in one search iteration to reach their goals. A conflict happens when two robots are to occupy a single vertex at the same time. Thus, the main aim of solving the MAPF problem is to find a set of paths passing through non-conflict vertices on the defined graph. To limit unnecessary search, an extra cost function, namely sum-of-cost, like the total maximum time for all robots to reach their goals (or the cost of the paths) is introduced as an optimal condition for the search. Since the problem is non-deterministic polynomial-time (NP) hard [1], numerous approaches have chosen to seek for a close optimal solution to reduce processing time. The A*-based search uses a heuristic function to find an optimal solution among all combinations of assigning k-agent into the graph. To deal with the exponential growth of the state-space with respect to the number of robotic agents, different methods have been applied. For instance, independence detection (ID) method by Standley [5] focused on single robot and only considered a group of multiple robots jointly when necessary.

Alternative to A*, the increasing cost tree search (ICTS) [2] proposed two-layers including high-level and low-level searching where the lower is used as a goal test of the higher. Another solution different from A*, conflict based search (CBS) method is introduced by Sharon et al. [3]. In CBS method, agents are constrained by a triple of parameters including the agent, occupying vertex, and time step. It means that the agent at the particular time step is refused to occupy an occupied vertex. The path is found only if all agent's constraints are satisfied. The searching is completed when the paths for every agents are resolved.

Beside the above solutions, there have been suboptimal solutions for the MAPF problem. For instance, hierarchical cooperative A* (HCA*) approach [4] introduced a reservation table which is used to store the path assigned into an agent. The other agents will, according to their priority, search for paths not registered in the reservation table and, after the paths are found, update the table accordingly. In an improved version of HCA* like Windowed-HCA* (WHCA*) [4], the reservation table is only applied for a limited time slot, i.e. window, when the other agents are rejected to reserve to the table. In overall, the heuristic search A* and its variants are still costly computational solutions.

There have been researches developed to reduce the running time of the search-based algorithms with rule-based algorithms. Specific rules are defined for the movement of the agents to reduce searching time. Yet, the resulted paths from the rule-based algorithms are not always optimal. Alternatively, in the work of Yu and Lavalle [6], the path planning problem for multiple agents is modeled as a network flow and the collision-free paths are found by the integer linear programming (ILP) solver.

Most of the presented solutions for the MAPF problem are based on an assumption of a working environment without the presence of humans. It is due to that the mathematics model of those works are not defined to cover both obstacle avoidance and multiple-path planning into one combined framework. As a result, the operation of robot will be terminated as a human enters the safety regions of robots, making the solution limited to specific applications like robotics warehouse system. In the presented work, a new method of multiple path planning is proposed to consider both the human as well as other uncontrolled moving objects as factors into the path planning problem. This helps to enhance autonomous functions of robot navigation by allowing more flexibility of robots to continue working even with the presence of other robots in an unfamiliar environment.

B. *Collision Avoidance*

A field-based approach is one way to perform obstacle avoidance. In general, the field consists of a repulsive field to push the agent away from the obstacles, and an attractive field to pull the agent towards the goal. For instance, Ok et al. [7] proposed a method with an uncertainty field which is build from Voronoi diagram from the start to the goal to create the attractive field to drive the robot to the goal and the repulsive field from the robot to the obstacles. The main issue with using this method is that the repulsive field may push the agent to reach other obstacles or statures with the attractive field. Due to this problem, the robot may be trapped into a local optimum or loose its way toward the goal.

Controlling the speed and directions of a robot is also another way to provide the robot a collision free path. Owen and Montano [9] defined velocity obstacle (VO) to estimate the arrival time of moving objects to a region of collision. The acceptable velocity is the one that helps the robot to avoid collision regions. Damas and Santos-Victor [10] developed a map of forbidden velocity zones which is constructed as a limit on the velocity of the robot to avoid collision with obstacles. When the robot enters into the forbidden zones, it may adjust its speed to avoid the obstacles. In the work of Berg et al. [8], the reciprocal velocity obstacle (RVO) is introduced. In this method, the interaction of robots is modelled in both distributed and an optimal pairwise while the other agents are assumed to continue moving with the current speed in a straight line trajectory and a function of relative velocity may be used to predict further collision. The extensions of this method are developed by Wilkie et al. [12] which is generalised for nonholonomic robots, and are improved by Berg et al. [14] by introducing the optimal reciprocal collision-avoidance (OCRA) to prevent the problem of reciprocal dances and casts. Additionally, Berg et al. [11] integrated the acceleration while Lee et al. [13] defined the footprint of the robot as an ellipse for obstacle avoidance. Usually, to follow the global path, the preferred velocity is defined. Yet, the presence of multiple obstacles, especially non-static obstacles, usually leads to the case where no optimal velocity is found for the next moving steps, which may lead to a deadlock situation.

III. MULTI-PATH PLANNING WITH OBSTACLE AVOIDANCE

A. Preliminaries

In this paper, a vector is presented in bold $\mathbf{x}$, matrix in capital and bold $\mathbf{X}$, and a set in mathcal $\mathcal{N}$. All robotic agents and dynamic obstacles move on a free space on a 2D-plane. Assume that there are n robotic agents in the working space, denoted by $\mathcal{A} = \{i | i \in 1, 2, ..., n\}$. The position of each robotic agent i at time t is presented by a function $\mathbf{a}_i(t) \in \mathbb{R}^2$ with the correspondent velocity $\mathbf{v}_i(t) = \dot{\mathbf{a}}_i(t)$. Correspondingly, let $\mathcal{O}_i = \{j | j \in 1, 2, ..., n_i\}$ be a set of moving obstacles detected by the agent i with position $\mathbf{o}_i^j(t)$ and velocity $\mathbf{vo}_i^j(t) = \dot{\mathbf{o}}_i^j(t)$. The footprint of a robot i is modelled by a closed disk with the radius r_i. For simplicity, every function $\mathbf{x}(t)$ by time t has an equivalent representation of $\mathbf{x}$ in short. To check for the collisions among robots and moving obstacles in a local range, the concept of velocity obstacle is utilised. The velocity of a moving robot is considered to be a straight-line constant velocity within short time, leading to the position is updated by the equation

$$\mathbf{a}(t) = \mathbf{a}(t_0) + (t - t_0)\mathbf{v}, t \geq t_0, \tag{1}$$

where t_0 and $\mathbf{a}(t_0)$ are the current time and position of the robot at that time respectively. Given two robots with position $\mathbf{a}_A$ and $\mathbf{a}_B$, a set of relative reference velocities $\mathbf{v}_{AB} = \mathbf{v}_A - \mathbf{v}_B$ leading to the collision within time τ is given by,

$$\mathcal{VO}_{AB}^{\tau} = \{\mathbf{v}_{AB} | \forall t \in [0, \tau], \|\mathbf{a}_A - \mathbf{a}_B + t\mathbf{v}_{AB}\| \leq r_A + r_B\}. \tag{2}$$

This velocity obstacle $\mathcal{VO}_{AB}^{\tau}$ is visualised by a truncated cone and approximated by the non-convex space formulated by three half planes (Fig. 1). Once the velocities and positions of moving obstacles are detected by a robot, the pairwise collisions between a robot and an obstacle is modelled and checked by the similar way. Yet, the moving obstacles may be seen by one one robot but not others, therefore, the collision checking between robot-moving obstacle is only available within local regions.

Meanwhile, the global map of a set of static obstacles $\mathcal{M}$ is presented by a binary image. To take into account the size of robot to avoid collisions with obstacles, the global map image is usually dilated by the radius of the robots' footprint. Let $\mathcal{M}(r)$ be a map with dilated obstacles of the radius r. The radius is usually set by the maximum radius r_{max} among different robots' footprint.

B. Problem Formulation

From the current position $\mathbf{a}_i(t)$ of the robot i, there are available p_i paths to its goal. Those paths can be established by running a random-related algorithm, e.g. rapidly exploring random tree (RRT), several (multiple) trials or by using different path finding algorithms, or even by using manual inputs from users. In this work, the any-angle searching with Theta* [15] is utilised to generate paths by using sequential inserting a set of found paths (the thickness of the path is dilated by the radius of the robot) into an obstacle map. By this way, the next found path will not overlap with the previous one. The any-angle searching Theta* is chosen instead of using A* or Djikstra's algorithms because Theta* is able to provide the optimal path with few turns and in the form of a set of line segments that reduces the changes in orientations to save energy by maintaining a constant moving speed and orientation. Also, by this way, it is convenient to find the intersections of two paths and define constraints for potential collision areas.

There is a preferred velocity of a robot defined on each path in such a way that the velocity remains constant along the path and smoothly decreases when the robot approaches to its goal. Let $\mathcal{V}_i = \{k | k \in 1, 2, ..., p_i\}$ be a set of the available paths for robot i, p_i be the number of paths, $\mathcal{P}_i = \{\bar{\mathbf{v}}_i^1, \bar{\mathbf{v}}_i^2, ..., \bar{\mathbf{v}}_i^{p_i}\}$ be a set of preferred velocities on each path. The control velocity $\mathbf{v}_i$ to determine the next move of the robot is set to be close to one of the preferred velocities as only one path is chosen among $\mathcal{V}_i$. Let $\mathbf{z}_i = [z_i^1, z_i^2, ..., z_i^{p_i}]^T$ be the binary vector to select the path, $z_i^k \in \{0, 1\}$. The optimisation cost function $C_i(\mathbf{v}_i, \mathbf{z}_i)$ is defined as follows:

$$\begin{aligned} C_i(\mathbf{v}_i, \mathbf{z}_i) &= \|\mathbf{v}_i - \sum_{k=1}^{p_i} \bar{w}_i^k z_i^k \bar{\mathbf{v}}_i^k\|^2 + \sum_{k=1}^{p_i} z_i^k \bar{s}_i^k \\ \text{s. t.} \quad & \sum_{k=1}^{p_i} z_i^k = 1 \end{aligned} \tag{3}$$

where $\bar{w}_i^k$ and $\bar{s}_i^k$ are the weights and the travelled lengths of the path. Without further constraints, minimising $C_i(\mathbf{v}_i, \mathbf{z}_i)$ leads to the selection of the shortest path among candidates.

The joint optimisation function for all robots to find their optimal paths and velocities is expressed by,

$$\begin{aligned} C(\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n, \mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_n) &= \sum_{i=1}^{n} C_i(\mathbf{v}_i, \mathbf{z}_i) \\ \text{s. t.} \quad \sum_{k=1}^{p_i} z_i^k &= 1, \forall i \in [1, n]. \end{aligned} \tag{4}$$

Along with the cost function, a set of constraints are defined to find collision-free paths for robots considering that

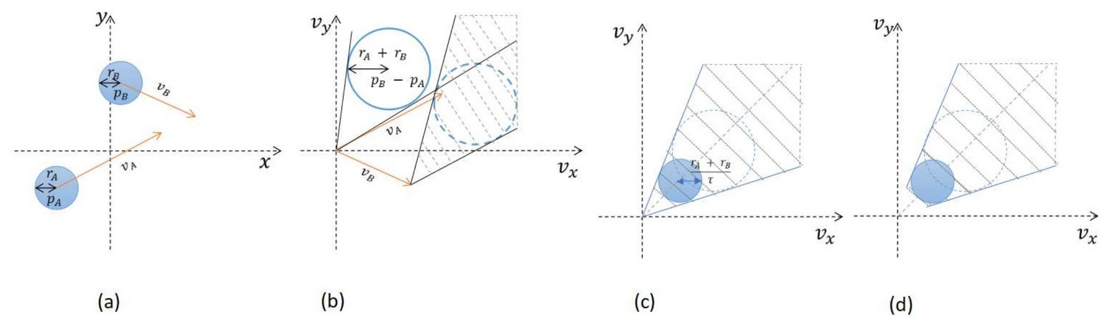

Fig. 1. Formulation of velocity obstacles. (a) The workspace configuration of the two robots R_A and R_B with their velocities v_B and v_B respectively. (b) The translation into velocity space and the resulting VO for robot R_A. (c) The VO of an obstacle is truncated at $\tau = 2$. (d) The approximating of truncated VO.

they have different options to chose their paths and also they need to avoid any dynamic obstacles on their moving ways.

1) Multi-path conflict-free constraints: Assume that two robots A and B are configured with multiple paths to goals (Fig. 2). However, some specific combinations of the path for the two robots could lead to a potential collision or deadlock. For instance, if robot A and B selects the path *p_2A* and *p_2B* for their moving plans, there exists an overlapping segment of the paths where the two robots may meet with each other. In the case the overlapping segment is bounded by a narrow area surrounding with static obstacles, there is a high possibility that the two robots are getting stuck inside the region. A set of constraints are used to penalise such combinations,

$$\begin{aligned} \mathcal{CF} = \{z_i^k + z_j^l \leq 1 | \forall i, j \in \mathcal{A}, k \in \mathcal{V}_i, l \in \mathcal{V}_j, \\ \text{if there is a potential conflict when robot } i \\ \text{chooses the path } k \text{, and } j \text{ chooses } l\}. \end{aligned} \tag{5}$$

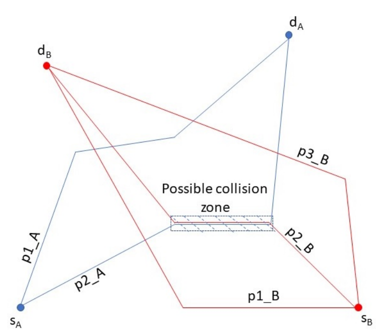

Fig. 2. An example of creating multiple paths for two moving robots with a possible collision zone.

2) Moving obstacle avoidance constraints: According to the definition of VO (Section III-A), the collision between robot A and another robot or a moving obstacle B is avoided if $\mathbf{v}_A - \mathbf{v}_B \notin \mathcal{VO}^{\tau}_{AB}$. This non-convex constraint $R^2 \setminus VO^{\tau}_{AB}$ is approximated by three linear constraints $\mathbf{n}^l_{AB}.\mathbf{v}_{AB} \leq b^l_{AB}$, with $l \in 1, 2, 3$

$$\begin{aligned} \begin{bmatrix} \cos(\alpha+\beta) \\ \sin(\alpha+\beta) \end{bmatrix} \mathbf{v}_{AB} \leq 0, \\ \begin{bmatrix} \cos(\alpha-\beta) \\ \sin(\alpha-\beta) \end{bmatrix} \mathbf{v}_{AB} \leq 0, \\ -\frac{\mathbf{p}_{AB}}{p_{AB}}.\mathbf{v}_{AB} \leq \frac{p_{AB} - \bar{r}_{A+B}}{\tau}, \end{aligned} \tag{6}$$

in which $\mathbf{p}_{AB} = \mathbf{a}_A - \mathbf{a}_B$, $\bar{r}_{A+B} = r_A + r_B$, $p_{AB} = \|\mathbf{p}_{AB}\|$, $\alpha = \arctan 2(-\mathbf{p}_{AB})$, and $\beta = \arccos(\bar{r}_{A+B}/p_{AB})$. The first and second constraints are to realize the right and left side of avoidance. The last constraint makes sure that there are no collision up to up to $\tilde{t} = \tau$.

One of the approaches to add this non-convex constraint into the optimisation problem introduces extra binary variables to select one of these linear constraints to apply. However, the number of binary variables increases rapidly with respect to the number of agents. To avoid doing so, in this work (also similar to approaches used in [20]) only one of the three linear constraints is applied where the constraint l is selected based on the current velocity of robots at the current time t_{curr},

$$l^* = \arg\min_l \; \mathbf{n}^l_{AB}.(\mathbf{v}_A(t_{curr}) - \mathbf{v}_B(t_{curr})) - b^l_{AB}. \tag{7}$$

3) Static obstacle avoidance: Since the static obstacles can be treated as moving obstacles with zero velocities, the static obstacle avoidance can be addressed using VO as presented in Section III-B.2. However, as the combined VO areas are proportional to the size and the number of obstacles in the global map, adding many static obstacle avoidance constraints may lead to deadlock situations where the optimisation of the problem will not be able to find a feasible velocity. Therefore, in this work, the static obstacle avoidance is handled with dynamic window approach (DWA), that is described further in Section IV-A.

Finally, the overall optimisation problem is formulated, in which the optimal control velocities and selected global paths $[\mathbf{v}^*_{1:n}, \mathbf{z}^*_{1:n}] = [\mathbf{v}^*_1, \mathbf{v}^*_2, ..., \mathbf{v}^*_n, \mathbf{z}^*_1, \mathbf{z}^*_2, ..., \mathbf{z}^*_n]$ are estimated in a joint manner,

$$
\begin{aligned}
[\mathbf{v}^*_{1:n}, \mathbf{z}^*_{1:n}] &= \underset{[\mathbf{v}_{1:n}, \mathbf{z}_{1:n}]}{\arg\min}\, C(\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n, \mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_n) \\
\text{s. t.} \quad & \sum_{k=1}^{p_i} z_i^k = 1, \forall i \in [1, n] \\
& \mathcal{CF} \quad \text{(Constraint 1)} \\
& \mathbf{v}_i, \mathbf{v}_j \notin \mathcal{VO}^{\tau}_{ij}, \forall i, j \in \mathcal{A} \quad \text{(Constraint 2)} \\
& \mathbf{v}_i \notin \mathcal{VO}^{\tau}_{io_j}, \forall i \in \mathcal{A}, j \in \mathcal{O}_i.
\end{aligned} \tag{8}
$$

IV. EXPERIMENTS

A. ROS-based Implementation

The overall system presented in this work is implemented with a well known platform for robots, robot operating system (ROS) [16], with the specific version of Kinetic Kame installed on Ubuntu 16.04. The evaluation is performed with the comprehensive Gazebo simulator and robotic mowers developed based on Husqvarna research platform (HRP) [17]. Those simulated robots are mounted with an extra RGB camera, a depth sensor, and a LIDAR laser scanner. As the system is operated in a centralised manner, a ROS node is designed as a server to collect information about planned paths, position updates, and velocities of all robots as well as moving obstacles detected by the robots in the working domains. The optimal velocities are computed at the ROS center node and are sent back to the robots to control their movements. Human objects are modelled as actors in Gazebo simulator with either a repeated predefined trajectory or a random trajectory. The overall optimisation formulation for the whole system is a mixed quadratic integer programming (MQIP) problem, and the solution for the problem is calculated by IBM CPLEX solver.

To apply the estimated optimal velocities to control the movements of robots, the dynamic window approach (DWA) [18] is used. The DWA is a commonly sampling-based approach that allows to generate a set of possible trajectories of a robot in a short time slot based on feasible velocities and limited accelerations. The trajectories are scored with regards to the distance between the robot and obstacles, the distance to reach the goal, or the deviation from the global path. The trajectories leading to collisions with static obstacles on the map are removed from consideration. In this work, to utilise DWA to realise the estimated controlled velocities for the robots, the averaged velocity on each trajectory is approximated by dividing the distance between the start and end of the trajectory with the traveling time. The DWA scoring function therefore aims to find the trajectory that minimizes the differences between the trajectory's velocity with the targeted velocity. In this way, both static as well as dynamic objects are considered when optimizing the possible path for each robot.

B. Two Robots Crossing Narrow Corridor Scenario

A typical scenario of two robots crossing a narrow corridor is evaluated in this section to demonstrate how multi-path planning is used to address the congestion problem. Two HRP robots are located at two different sides of the corridor. By applying multiple path planning, each robot has found a set of two possible paths from its starting position to its goal (Fig. 3). The two red paths are the paths found by the robot 1, and the green ones are for the other. The yellow paths are the actual trajectories of the two robots after they have received optimal velocity control from the center ROS node. If the first robot chooses to navigate through the corridor, the other one will choose the other path in order to avoid congestion in the corridor although this path is longer than its optimal route.

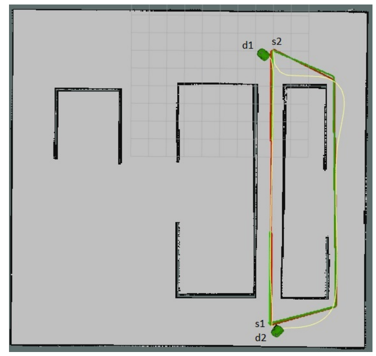

Fig. 3. Multiple planned paths of two moving robots to reduce the risks of collisions.

C. Scenario with Robots/Humans Together

A simulated working space with several narrow corridors as depicted in Fig. 4 is used to further evaluate the system on congestion and obstacle avoidance. Three HRP robots are fixed with starting points but randomly assigned goals. In a half of experiments, two robots are arranged on the two sides (top and bottom) of the map (Fig. 4) and move from one side to the other. Two human actors are added with predefined moving trajectories. The experiments has been repeated 10 times and the proposed algorithm is compared with DWA and DWA+VOs [19]. A collision happens if the distance between robot/robot or robot/human is less than 0.5 meters regarding the size of robots. The results (Table I) show that the proposed algorithm is superior to previous works with respect to ability to avoid collisions/dead-locks. Outside the narrow corridors, the optimal control helps them to avoid collisions with humans (The minimum distance between robots and humans in all experiments is 0.51 meters when robots are controlled by the proposed navigation algorithm). This is shown by the changing directions on the moving trajectories of different robots (Fig. 5). It is noted that the human actors are only visualised on the Gazebo simulator.

V. CONCLUSIONS

This paper presents a novel multiple path planning approach, which can deal with an uncertain and dynamic environment containing non-static objects such as humans and robots. The presented method introduces effective means of a global planner for avoiding deadlock situations to overcome the risk of congestion when multiple robots are navigated through,

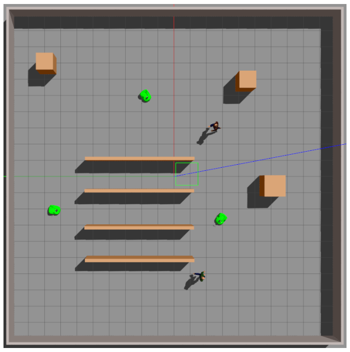

Fig. 4. The simulated working space with three robots and humans.

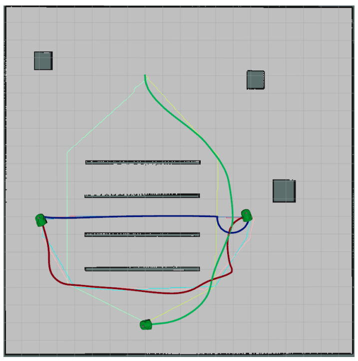

Fig. 5. Moving trajectories of robots. The thin trajectories are the two planned path while the bold ones are the actual moving trajectories of robots.

relative to the robots, a narrow area. The combination of VO-based method and common DWA planner allows robots to avoid collisions with moving obstacles. In overall, the velocities of the robots are transferred into an optimisation problem to improve the performance of controlling the robots' movements. In addition, the ROS based communication channel allows the robots to negotiate between the different possibilities to have collision-free path solutions, and also to allow continuous updates of the positions of each obstacle in the environment used in calculating new possible paths. The evaluations in the Gazebo simulator has proved that the proposed approach with multiple path planning is effective, safe and promising for an autonomous robot team. In the future, the decentralised movement control will be investigated to reduce the dependence of the planning algorithms on communication infrastructure. Also the method can be improved by applying a delay to be lower the energy consumption of robots: The robot may, instead of choosing the longer path, wait for others to follow the shortest path to reach its goal. Finally, an extensive evaluation with real robots will be planned.

ACKNOWLEDGMENT

The research leading to the presented results has been undertaken within the research profile DPAC - Dependable Platform for Autonomous Systems and Control, funded by the Swedish Knowledge Foundation.

TABLE I

MINIMUM DISTANCE AMONG ROBOTS (IN METERS), NUMBER OF COLLISIONS, AND NUMBER OF DEAD-LOCKS OVER 10 TRIALS.

	Distance	Collisions	Dead-locks
Multi-path planning	0.56	0	0
DWA	0.21	3	5
DWA + VOs [19]	0.57	0	3

REFERENCES

[1] J.Yu, and S.-M. LaValle, Structure and intractability of optimal multi-robot path planning on graphs, Proceedings of the 27th AAAI Conference on Artificial Intelligence, pp. 1443-1449, 2013.

[2] G. Sharon, R. Stern, M. Goldenberg, and A. Felner, The increasing cost tree search for optimal multi-agent pathfinding, Journal of Artificial Intelligence", vol. 195, pp. 470-495, 2013.

[3] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, Conflict based search for optimal multi-agent pathfinding, Journal of Artificial Intelligence, vol. 219, pp. 40-66, 2015.

[4] D. Silver, Cooperative pathfinding, Proceeding of the First AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, pp. 117-122, 2005.

[5] T. S. Standley, Finding optimal solutions to cooperative pathfinding problems, Proceedings of the 24th AAAI Conference on Artificial Intelligence, pp. 173-178, 2010.

[6] J. Yu, and S. M. LaValle, Planning optimal paths for multiple robots on graphs, In IEEE International Conference on Robotics and Automation (ICRA), pp. 3612-3617, 2013.

[7] K. Ok, S. Ansari, B. Gallagher, W. Sica, F. Dellaert, and M. Stilman, Path Planning with Uncertainty: Voronoi Uncertainty Fields, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 4581-4586, 2013.

[8] J. V. D. Berg, M. Lin, and D. Manocha, Reciprocal Velocity Obstacles for Real-Time Multi-Agent Navigation, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1928-1935, 2008.

[9] E. Owen, and L. Montano", A Robocentric Motion Planner for Dynamic Environments Using the Velocity Space, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4368-4374, 2006.

[10] B. Damas, and J. Santos-Victor, Avoiding Moving Obstacles: the Forbidden Velocity Map, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4393-4398, 2009.

[11] J. V. D. Berg, J. Snape, S. J. Guy, and D. Manocha, Reciprocal Collision Avoidance with Acceleration-Velocity Obstacles, Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 3475-3482, 2011.

[12] D. Wilkie, J. V. D. Berg, and D. Manocha", Generalized Velocity Obstacles, Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 5573-5578, 2009.

[13] B. H. Lee, J. D. Jein, and J. H. Oh", Velocity obstacles based local collision avoidance for holonomic elliptic robot, Journal of Autonomous Robots, vol. 41, pp. 1347-1363, 2017.

[14] J. V. D. Berg, S.J. Guy, M. Lin, and D. Manocha, Reciprocal n-body collision avoidance, International Symposium on Robotics Research, pp.3-19, 2009.

[15] A. Nash, K. Danial, S. Koening, and A. Felner, Theta*: Any angle path planning on grids, Journal of Intelligent Robot System, vol. 39, pp. 533-579, 2010.

[16] M. Quigley et al., ROS: An open-source robot operating system. In Proceedings of the open-source software workshop, 2009.

[17] https://github.com/HusqvarnaResearch/hrp.

[18] D. Fox, W. Burgard, and S. Thrun, The dynamic window approach to collision avoidance, IEEE Robotics and Automation Magazine, pp. 23-33, 1997.

[19] D. Claes, and K. Tuyls, Multi robot collision avoidance in a shared workspace, Autonomous Robots, Springer, 2018.

[20] J. A-Mora, P. Beardsley, and R. Siegwart, Cooperative Collision Avoidance for Nonholonomic Robots, IEEE Transactions on Robotics, vol. 34, issue 2, pp. 404-420, April 2018.