FULL PAPER

Region with velocity constraints: map information and its usage for safe motion planning of a mobile robot in a public environment

Tae Hyon Kim, Kazuyuki Kon and Fumitoshi Matsuno
Kyoto University, Kyoto, Kyoto, Japan

ABSTRACT
Recently, various autonomous mobile robots have been developed for practical use. To support the coexistence of robots and humans in real environments, we propose a concept named ‘Region with Velocity Constraints (RVC),’ which is set around hazardous areas. RVCs are regions where the velocities of the robot are constrained to predefined values. Inside the RVCs, the robot has to reduce its translational velocity to avoid predicted hazards such as collisions with obstacles, and to reduce its rotational velocity to prevent undesirable motions such as sharp turns. We also propose a motion planning method for navigating the mobile robot in an environment with RVCs based on the Navigation Function and Global Dynamic Window Approach. Our method generates a trajectory satisfying both translational and rotational velocity constraints to be compatible with the surroundings. Moreover, to demonstrate the validity of our method, we performed numerical simulations and experiments.

ARTICLE HISTORY
Received 26 November 2014
Revised 31 August 2015
Accepted 31 January 2015

KEYWORDS
Motion planning; hazard region; velocity constraints; autonomous mobile robot

1. Introduction

Autonomous mobile robots have been developed for a number of applications such as automated cleaning [1] and automated order fulfillments.[2] Several other studies have also tried to address more fundamental functions such as simultaneous localization and mapping,[3–5] motion planning [6–8] and obstacle avoidance.[9–11]

Compared with indoor environments, which have been the focus of research for decades (e.g. [12,13]), robots in real-life outdoor environments have to operate in the presence of unattended surroundings and obstacles such as humans, bicycles, and cars. A key issue pertinent to such environments is to realize the coexistence of robots and their surroundings. Research addressing motion strategies for robots facing real-life environments can be found under the name Human–Robot Interaction (HRI).[14,15] For example,[14] tried to realize social interaction between humans and robots, and [15] aimed to use robots to communicate with people and guide interaction with them. Furthermore, from the viewpoint of motion planning, collision avoidance methods taking into account personal space [16] have been proposed to handle psychological interactions with humans.[17–20]

Despite extensive research into the applications of autonomous mobile robots in real environments, there persist several unsolved problems such as environmental constraints and the impact on humans. This study focuses on two of these problems: (i) avoiding collisions and (ii) predicted mobility, which is quite natural and does not cause inconvenience to surrounding humans. The first problem arises because of the potential presence of various hazardous regions in the real environment such as steps, ponds, slopes, narrow roads, and rough roads. Moreover, because the real environment is dynamic and can have many unpredictable obstacles such as humans and bicycles, the risk of collisions with such obstacles is high. Hence, it is difficult to judge from the equipped sensory data in the real environment whether the robot is safe to move at a certain speed or whether it needs to decelerate. Therefore, autonomous mobile robots should constrain their translational velocity for the safe avoidance of collisions.

The second problem is related to the presence of humans in real environments. Humans can walk and move around autonomous robots, and their walking is by various factors that exist around them.[21] They also have a sense of personal space and feel discomfort when their space is infringed by unknown other people.[22] Hence, an autonomous mobile robot should not impede walking humans or invade their personal space through unpredictable movements such as taking a sharp turn.

To solve the above-mentioned problems, we propose a concept named Region with Velocity Constraint (RVC)
that imposes constraints on both translational and rotational velocity. The RVC is a region where the translational and rotational velocities of robots are forced to satisfy pre-determined admissible values. RVCs are set around hazardous regions where the robot has a high probability of a movement with a sharp turn or a collision with obstacles. Moreover, RVCs can also be set as regions where the robot cannot pass through. If the robot plans trajectories according to the registered translational velocity of the RVCs, the risk of collision within RVCs can be reduced. In addition, through the registered rotational velocity constraint of RVCs, the generated trajectories can eliminate sharp turns while inside the RVCs.

We also propose a motion planning method to navigate a mobile robot in an environment including RVCs, in which both the translational and rotational velocity constraints are clearly satisfied. Note that the concept of RVC was originally proposed in our previous work.[23] This study generalizes the concept so that the rotational velocity constraints are also considered to prevent undesirable motions such as sharp turns. Furthermore, we evaluate the method through numerical simulation and validate the result experimentally.

The paper is organized as follows: Section 2 reviews the related work and highlights the contribution of this paper. Section 3 describes in brief the concept of RVC and clarifies the control problem of navigating a robot in real environment. Sections 4 and 5 delineate the motion generation method. Section 6 shows numerical examples to validate the proposed method. After implementing the method on our mobile robot, we present the results in Section 7. Section 8, the final section, contains the conclusion and lays out the direction of our future research.

2. Related work and contribution

In this section, we provide an overview and compare related work, and clarify the contribution of this study.

2.1. Related work

The concept of velocity constraints, similar to RVC, that improves the safety of an autonomous robot, can be found in the literature.[24,25] Karunamchi and Iagnemma [24,25] proposed a method that constrains the robot velocity based on the classification of the environment to deal with multiple environments. This method defines hard and soft hazards for the off-road navigation of a mobile robot: Hard hazards are interpreted as problematic obstacles such as ponds, while soft hazards are environmental conditions that impose additional differential constraints to the vehicle beyond the non-holonomic constraints arising from its dynamics. Because soft hazards limit the maximum speed for precise control, the vehicular motion in the soft hazard region becomes safe by satisfying the constraints in a proprioceptive way. However, no explicit motion planning is mentioned.

Several possible candidates exist to realize a motion planner in an environment including RVCs, including those based on the Model Predictive Control (MPC), Rapidly-Exploring Random Tree (RRT), and Global Dynamic Window Approach (GDWA), which are capable of considering state constraints such as velocity constraints. MPC determines the control input on-line by solving a finite horizon open-loop control optimization problem (see e.g. [26,27]). A feature of MPC-based approaches is that they can take into account various inequality constraints in the optimal control problems. Indeed, there are several studies that take into account constraints such as dynamic constraints in the area of the trajectory generation problem of a vehicular robot, including velocity constraints on the vehicle,[28,29] the obstacle avoidance problem,[30,31] and the collision avoidance problem.[32,33] Unlike the velocity constraint of the vehicle itself, to handle the velocity constraints in RVCs, which change according to the robot position, constraints switching is needed. It might be possible to handle such switching constraints by formulating them as Mixed Integer Programming, which represents the switching of the velocity constraints as an inequality constraint including binary variables.[34] Another possible way is to solve the nonlinear optimization problem [35] directly. Either way, it is difficult to apply the MPC scheme to our problem because of its heavy computational load.

Conversely, RRT and GDWA are well-known sampling-based motion planning methods. RRT is a famous path planner in higher dimensional space with lower cost.[36] It builds trees of nodes and edges that correspond to states and small amplitude motions, respectively, and is capable of incorporating kinodynamic constraints. There are several variants of RRT (e.g. [37–40]). For example, Environments-Guided RRT [37] increases the efficiency of path search, especially in the presence of narrow passages. RRT* [38–40] ensures asymptotic optimality and guarantees almost-sure convergence to a globally optimal solution without sacrificing computational efficiency. Karaman [39] adapts an optimization of the non-holonomic system with RRT*, and RRTX [40] improves RRT* to accommodate the dynamic environment and obstacles. GDWA [41] is a real-time trajectory planning method based on a Navigation Function (NF) [42] and a Dynamic Window approach.[43] It is derived directly from the dynamics of the robot, and is specially designed to deal with the constraints imposed by the limited velocities and accelerations of the robot. It
consists of two main components, first generating a valid search space, and second selecting an optimal solution in the search space. The search space is restricted to safe trajectories that can be reached within a short-time interval and are free from collisions. The trajectories are evaluated through NF, which is a local-minima free potential function, and an optimal trajectory is selected. Moreover, it guarantees obstacle avoidance and convergence to the target state.[44]

If we apply the above-mentioned sampling-based motion planning methods to an environment including RVCs, the following two problems arise: path search inefficiency and constraint violation during region transition. Because the maximum velocity of the robot varies in environments that include RVCs, path search inefficiency occurs. Moreover, because the velocity constraint changes according to the robot position, constraint switching is needed as mentioned above. This might cause constraint violation during region transition. To accommodate these problems, we decided to apply GDWA to our problem, because its NF is able to handle RVCs in the path planning step. Furthermore, we modified the trajectory generation step to ensure the satisfaction of both translational and rotational velocity constraints, even when the region transitions, by considering the posture of the robot.

2.2. Contribution

To summarize, the main contribution of this work is that it proposes a motion planning method that consists of:

- a NF taking into account the slow-downs required in RVCs,
- a trajectory generation mechanism adapted to RVCs.

Apart from the above, this research partially contributes to the following two points as well:

- Because our proposed method plans trajectories in real-time considering the posture of the robot, it does not slow down conservatively around RVCs. (The motion planning of [23] is particularly conservative around obstacles.)
- The concept of RVC is generalized so that it can handle the translational and rotational velocity constraints independently.

3. Problem formulation

In this section, we define a RVC in Section 3.1. Then, we formulate the navigation problem in an environment with RVCs in Section 3.2.

3.1. Definition of RVC

As mentioned in Section 1, an RVC is a region where the translational and rotational velocities are constrained to predefined velocities considering safety and the impact of people, and are set around hazardous regions. Definitions of the RVC are described as follows:

- **Definition 1**: An RVC is a region where the maximum translational and rotational velocities of a robot are restricted to predefined admissible values.
- **Definition 2**: RVCs are set around possible hazardous regions.
- **Definition 3**: The maximum translational and rotational velocities of RVCs are time-variant.
- **Definition 4**: An RVC is the kind of information included in a map. Overlap between RVCs is unacceptable, except in cases in which the RVC is fully included within another RVC.
- **Definition 5**: An RVC is capable of optionally including information about hazardous regions.

In Definitions 1 and 2, the RVC’s translational velocity constraint is set to avoid collisions or to stop easily, and the RVC’s rotational velocity constraint is set to eliminate sharp turns. In Definition 3, the time-varying velocity constraints are acceptable if the derivatives of the velocity constraints satisfy the acceleration constraints. For the sake of simplicity, time-varying RVCs are not considered in this paper. Moreover, to demonstrate the validity of the proposed method, information about RVCs is given in advance.

For examples, as depicted in Figure 1, the steps and of the area near the pond, where the robot is not allowed to enter, are set as RVCs. Rough roads and slopes are also set as RVCs to prevent risks caused by high-speed movements. Furthermore, crossroads, crowded streets, narrow streets, and the area around a gateway are set as RVCs, considering the impact of people and collision avoidance. Finally, there are many factors to determine appropriate constrained velocities of RVC. For example, the appropriate constrained velocities of a crowded region will be designed considering a space to transit, or a collision probability with predicted trajectories and obstacles. However, since how to determine the constrained velocities is not our target, this paper does not mention about that.

3.2. Navigation problems in an environment with RVCs

We considered a skid-steering wheeled mobile robot as the control target whose kinodynamic model is described as

\[ \dot{x}(t) = v(t) \cos \theta(t), \quad \dot{y}(t) = v(t) \sin \theta(t), \]
\[ \dot{\theta}(t) = \omega(t), \quad \dot{v}(t) = F/m, \quad \dot{\omega}(t) = \Gamma/J \]  
(1)
Figure 1. Examples of regions with velocity constraints.

Figure 2. Representation of the map used in motion planning: the environment shown in (a) is converted into the grid map shown in (b). In the case of (a), there might be a high risk of collision or slip, if the robot moves on the sandy road around the bench (Region 1) or grass (Region 2) at high speed. Therefore, these regions are set as RVCs with velocity constrained to half of the maximum translational velocity, and are also stored in the grid map as shown in (b). (c) shows the total NF value in the grid map, and (d) briefly illustrates the calculation procedure of our proposed NF.

where $v(t) \in [0, v_{\text{max}}]$, $\omega(t) \in [-\omega_{\text{max}}, \omega_{\text{max}}]$ are the translational and rotational velocities of the robot respectively, and $z(t) := (x(t), y(t))$ and $\theta(t)$ denote the measurable coordinates with respect to the global frame. $F$ and $\Gamma$ are the force in the movement direction and the moment relating the point-mass robot to the global frame, and $m$ and $J$ are the mass and moment of inertia of the robot, respectively. The maximum value of the translational velocity, rotational velocity, translational acceleration, and rotational acceleration of the robot are defined by $v_{\text{max}} > 0$, $\omega_{\text{max}} > 0$, $a_{v_{\text{max}}} > 0$, and $a_{\omega_{\text{max}}} > 0$, respectively. Although the above-mentioned concept of RVC can be applied to both holonomic and non-holonomic mobile robots, we only focus on a non-holonomic mobile robot for simplicity of exposition.

In this study, we address the control problem of navigating the robot from the start position to the goal position in an environment including RVCs as depicted in Figure 1. We assume that $N_{\text{RVC}}$ RVCs exist in the environment, and define the jth RVC as $R_{vj}$ ($j = 1, 2, \ldots, N_{\text{RVC}}$).
Figure 3. Example of path planning: (a) shows an example of the standard NF in the environment without the RVC, whereas (b) shows an example of the proposed method in the environment including Region 1 with constrained velocity $v_{\text{max}} = 0.25v_{\text{max}}$ and Region 2 with constrained velocity $v_{\text{max}} = 0.5v_{\text{max}}$. The value in each cell shows the stored value (i.e. $M_{ij}$ in (a) and $N_{ij}$ in (b)). The arrow in each cell shows the gradient of the NF, and the dashed line shows the shortest path obtained by the NF.

Figure 4. Conceptual diagram of regions with velocity constraints: This figure illustrates the admissible velocity space in the environment including the RVCs and the obstacles. (a) illustrates the admissible and inadmissible velocity space of the translational velocity, and (b) illustrates the admissible and inadmissible velocity space of the rotational velocity. The left object in this figure (a) and (b) illustrates the $j$th RVC ($R_{vcj}$), and the right one illustrates the $k$th obstacle ($O_{adk}$). The regions surrounded by $L_{\text{max}}$ on the $xy$ plane shows the $j$th AD region ($R_{vcj}$) and the $k$th obstacle region ($O_{adk}$). Because the obstacles are also represented as the RVCs with the velocity constraints $0$ (m/s) and $0$ (rad/s), the AD regions of the obstacles become much larger than those of the RVCs ($v_{\text{max}} > 0$, $\omega_{\text{max}} > 0$). The shaded areas of all objects show the AD regions defined in (6), and indicate that the constrained velocity will be violated, even though the robot decelerates with maximum deceleration.

Let $v_{\text{max}}$ ($v_{\text{max}} \leq v_{\text{max}}$) and $\omega_{\text{max}}$ ($\omega_{\text{max}} \leq \omega_{\text{max}}$) be the admissible translational and rotational velocity of the $j$th RVC, respectively. Specifically, we define obstacles and no trespassing areas such as the steps and around the pond, where $v_{\text{max}} = 0$ and $\omega_{\text{max}} = 0$ should be imposed, as the $k$th obstacle region, and define as $O_{bsk}$ ($k = 1, 2, \ldots, N_{\text{obs}} \leq N_{\text{rvc}}$).

To summarize, the control objective is to achieve Equation (2) while satisfying the constraints in (3).

\[ \lim_{t \to \infty} z(t) = z_{\text{Goal}}, \quad z(t) \in \mathbb{R}^2 \setminus O_{bsk} \quad (k = 1, 2, \ldots, N_{\text{obs}}) \]

\[ \| \dot{z}(t) \|_2 \leq v_{\text{max}} \quad \text{and} \quad |\dot{\theta}(t)| \leq \omega_{\text{max}}, \]

if $z(t) \in R_{vcj}$ \quad ($j = 1, 2, \ldots, N_{\text{rvc}}$) \quad (3)

In this paper, we propose a new motion planning method that achieves the above control objectives. The proposed motion planning method can be divided into two steps: the first step is the path planning method presented in Section 4; and the second step is the trajectory generation method presented in Section 5.

4. Path planning in an environment with RVCs

The path planning is based on NF,[42] which is a grid-based local minima-free artificial potential function...
computed using a wave-propagation technique starting at the reference position.

The outline to generate the NF is as follows. First, construct the grid map, representing free-space as ‘0’ and obstacles as ‘1’ (e.g. from Figure 2(a) to (b)). We define a cell as \((i, j)\) based on its row and column indices, and define the goal cell including the reference position as \((i_g, j_g)\). Next, calculate the distance between the centroid of the cell \(z_{ij} := (x_{ij}, y_{ij})\) and the centroid of the goal cell \(z_{ig} := (x_{ig}, y_{ig})\). This distance is stored in each cell \((i, j)\). In general, the distance between the cells is calculated by the Manhattan Distance \(M_{ij}\) as follows:

\[
M_{ij} := \begin{cases} 
\infty & \text{if } (i, j) \text{ is an obstacle cell} \\
M_{ij}’ + \|z_{ij} - z_{ij}’\|_\infty & \text{otherwise,} 
\end{cases}
\]

where a cell \((i’, j’)\) denotes one of the four neighboring cells \((i, j)\) (i.e. one of \((i - 1, j), (i + 1, j), (i, j - 1)\) and \((i, j + 1)\), storing the minimum value). Note that if the cell \((i, j)\) is an obstacle cell, a value much larger than the possible values of \(M_{ij}\) is stored. The map with the distance in (4) is called the NF (e.g. Figure 2(c)). The shortest path to the reference position can be obtained by only moving toward a cell that has a smaller value.

However, it is not reasonable to use the standard NF for a navigation problem including RVCs, because only the distances between cells are used to evaluate the path. To handle the velocity constraint of each RVC in the path planning, we modify the NF so that the stored value in each cell represents the time required to move (i.e. the information based on the admissible velocity). That is, we change the criteria \(M_{ij}\) for evaluating paths to \(N_{ij}\), the time required to move from cell \((i, j)\) to cell \((i_g, j_g)\).

Similarly, we outline how to generate the proposed NF as follows. First, store the information about the velocity constraints of the corresponding RVC to each cell. We define the stored translational velocity constraint in cell \((i, j)\) as \(v_{ij_{\text{max}}}\), and in cell \((i’, j’)\) as \(v_{ij’_{\text{max}}}\). As depicted in the left side of Figure 2(d), we define the time required to move from \(z_{ij}\) to \(z_{ij’}\), in the case that both cell \((i, j)\) and \((i’, j’)\) are free-space cells, as the unit time (i.e. \(\|z_{ij} - z_{ij’}\|/v_{\text{max}} := 1\)). The value \(N_{ij} = N_{ij’} + 1\) is stored in cell \((i, j)\). If either cell is an RVC, the required time to move is calculated by \(\frac{1}{2} v_{\text{max}}\). For example, as shown the right side of Figure 2(d), if both cells are an RVC of the same translational velocity constraint of \(v_{ij_{\text{max}}} = 0.5v_{\text{max}}\), the required time to move in each cell is \(\frac{1}{2} v_{ij_{\text{max}}}/v_{\text{max}} = 1\). Therefore, the total required time of two RVC cells is \(N_{ij} = 2\). In the same manner, the total required time between a free-cell and an RVC cell is \(N_{ij} = 0.5v_{\text{max}}\) is \(N_{ij} = 1.5\). Finally, the shortest path to the goal can be obtained by moving toward a smaller value. To summarize, the time \(N_{ij}\) required to move between cell \((i, j)\) and cell \((i_g, j_g)\) can be written as follows:

\[
N_{ij} := \begin{cases} 
\infty & \text{: obstacle cell} \\
N_{ij’} + \frac{|z_{ij} - z_{ij’}|}{2v_{\text{max}}} + \frac{|z_{ij} - z_{ij’}|}{2v_{ij’_{\text{max}}}} & \text{: otherwise:} 
\end{cases}
\]

As described in the Equation (5), the path planning takes into account only translational velocity constraints for simplicity.

Figure 3 shows an example of path planning. Compared with the path planned by the standard method and the proposed method, we see that the path in Figure 3(b) goes through Region 2, whereas the one in Figure 3(a) goes through Region 1. Therefore, RVCs are correctly taken into account during path planning. Note that the proposed method can plan a path converging to the reference position without getting stuck in local minima because the procedure is the same as the standard one.

5. Trajectory generation in an environment with RVCs

The trajectory generation is based on the method called the GDWA, which is a trajectory generation method for obstacle avoidance.[41,44] First, we briefly introduce GDWA in Section 5.1, then propose the trajectory generation method in Section 5.2.

5.1. Outline of the standard Dynamic Window Approach

The GDWA uses the NF in (4) to avoid local minima. The outline of the standard GDWA can be summarized as follows. First, make \(k\) tuples \((a_v, a_\omega)\) of translational acceleration \(|a_v| \leq a_{v_{\text{max}}}\) and rotational acceleration \(|a_\omega| \leq a_{\omega_{\text{max}}}\) discretely. Then, select one of the tuples, and predict the translational velocity \(\nu(t + \Delta t)\) and rotational velocity \(\omega(t + \Delta t)\) at \(t + \Delta t\) (s) based on the selected tuple \((a_v, a_\omega)\). The future position \(z(t + \Delta t)\) and posture \(\theta(t + \Delta t)\) of the robot is then predicted based on the kinodynamics in (1). By iterating the above procedure for each tuple, the GDWA predicts the trajectories over the evaluation time \(T(s)(\geq \Delta t)\). The predicted velocities may violate the constraints, \(\nu \in [0, v_{\text{max}}]\) and \(\omega \in [-\omega_{\text{max}}, \omega_{\text{max}}]\), because \(a_v\) and \(a_\omega\) are fixed over the time interval \(T(s)\). If this is the case, change the acceleration in tuple \((a_v, a_\omega)\) to \(0\) m/s² or \(0\) rad/s² so that the predicted velocity satisfies the above constraint, and predict the trajectory again using the modified tuple. The predicted trajectories are evaluated based on the cost function \(\Omega\).
Algorithm 1 Proposed motion planning method in an environment with RVCs

1: Calculate NF with the Equation (5). Set $t = 0$.
2: while until a trajectory reaches the goal position: $z(t) \notin (x_{\text{goal}}, y_{\text{goal}})$ do
3: \hspace{1em} for $n$ \{n: total number of candidates of suboptimal trajectories\} do
4: \hspace{2em} for $k$ \{k: total number of parameter tuples $(a_v, a_\omega)$ discretely\} do
5: \hspace{3em} for $i = 1$ to (Evaluation time $T$ / Sampling time $\Delta t$) do
6: \hspace{4em} Predict translational and rotational velocities $(v(t + i\Delta t), \omega(t + i\Delta t))$ with $k$th tuple $(a_v, a_\omega)$.
7: \hspace{4em} Predict a position $z(t + i\Delta t)$ using the kinodynamic model (1) with the predicted velocities.
8: \hspace{3.5em} Algorithm 2 Consider_RVC $(z(t + (i-1)\Delta t), z(t + i\Delta t)$ and $(v(t + i\Delta t), \omega(t + i\Delta t)))$
9: \hspace{4em} end for
10: \hspace{3em} Evaluate the predicted trajectory using a cost function $\Omega$.
11: \hspace{2em} end for
12: end for
13: Determine candidates of suboptimal trajectory based on each cost.
14: Update the time as $t = t + T$. 
15: end while

Finally, one of the trajectories maximizing $\Omega$ is selected as an optimal one. Note that because a trajectory of only a short-time interval $T(s)$ is generated, the GDWA obtains a trajectory from the initial point to the goal point by iterating the above procedure until it converges to the reference position.

\textbf{5.2. Trajectory generation considering RVCs}

Next, we propose a trajectory generation method considering RVCs. In contrast to the standard GDWA, it is necessary to consider the velocity constraints of each RVC to generate a trajectory in an environment with RVCs. To handle velocity constraints, we extended the standard GDWA as is summarized in Algorithm 1. The contribution of this study is found in line 8 (i.e. Algorithm 2) of Algorithm 1, while the other lines are almost the same as those in the standard GDWA. Let \textit{Acceleration/Deceleration(AD) region} be a region where the robot has to decelerate enough outside of the RVCs so as to satisfy the velocity constraints.\cite{23} Algorithm 2 comprises the following three steps.

\textbf{Step 1: Preliminary check for constraint violations}

1. if $z(t + \Delta t) \in R_{vcj}$ then
2. \hspace{1em} Set a tuple of admissible velocities $(v_{\max}, \omega_{\max})$ of $R_{vcj}$.
3. \hspace{1em} Setting minimum velocities $(r_j, (v, \omega) and (v_{\max}, \omega_{\max}))$
4. \hspace{1em} end if
5. else if $z(t + \Delta t) \in R_{adj}$ then
6. Compute the path length $\ell_j(t + \Delta t)$ along the curve and $(V(\ell_j(t + \Delta t)), W(\ell_j(t + \Delta t)))$ (8).
7. \hspace{1em} Setting minimum velocities $(r_j, (v, \omega)$ and $(V(\ell_j(t + \Delta t)), W(\ell_j(t + \Delta t)))$
8. \hspace{1em} end if
9. end if
10: end for
11: if $v(t + \Delta t) > v$ or $\omega(t + \Delta t) > \omega$ then
12: \hspace{1em} Recompute $z(t + \Delta t)$ and $(v(t + \Delta t), \omega(t + \Delta t)) in (9)$ so as to maintain $r(t + \Delta t)$.
13: end if
14: return altered position $z(t + \Delta t)$ and velocities $(v(t + \Delta t), \omega(t + \Delta t))$

In the following, we explain the proposed method according to the above three steps, and then explain the details of the algorithm.

\textbf{5.2.1. Step 1: Preliminary check for constraint violations}

In Step 1, if the predicted position is inside the RVC, it is obvious that the predicted position has to satisfy the velocity constraints of the RVC. However, if the predicted position is outside the RVC, we introduce an AD region so as to satisfy the velocity constraints. As depicted in Figure 4, the $j$th AD region ($R_{adj}$) is designed with a translational velocity constraint by considering the relative distance, and is the area surrounding the $j$th RVC by
Algorithm 3 Setting minimum velocities (Curvature radius $r_j$, Minimum velocities $(v, \omega)$ and constrained velocities $(v_{\text{constraint}}, \omega_{\text{constraint}})$)

1: if $v > v_{\text{constraint}}$ and Case I then
2: Calculate other tuple velocity $\omega_{\text{new}} = v_{\text{constraint}}/r_j$
3: $(v, \omega) \leftarrow (v_{\text{constraint}}, \omega_{\text{new}})$.
4: else if $\omega > \omega_{\text{constraint}}$ and Case II then
5: Calculate other tuple velocity $v_{\text{new}} = \omega_{\text{constraint}}r_j$
6: $(v, \omega) \leftarrow (v_{\text{new}}, \omega_{\text{constraint}})$.
7: end if
8: return $(v, \omega)$

$L_j(v_{\text{max}})$ defined as

$$L_j(v(t + \Delta t)) := \frac{1}{2a_{\text{max}}} \left(v^2(t + \Delta t) - v_{j_{\text{max}}}^2\right).$$

Specifically, $L_{\text{max}} := L_j(v_{\text{max}})$ denotes the minimum distance at $v = v_{\text{max}}$, and the region surrounded by $L_{\text{max}}$ is defined as the $j$th AD region (Rad$_j$). The AD region indicates the maximum region where it is possible for a predicted trajectory to violate the velocity constraints. Thus, if the trajectory is outside of the AD region, then the trajectory never violates the velocity constraints. However, since $L_{\text{max}}$ is the sufficient condition, it is impossible to check for a constraint violation inside the AD region. Therefore, we derive the exact admissible velocities depending on the relative distance between the robot and the RVC to check whether or not the trajectory will violate the velocity constraints.

5.2.2. Step 2: Admissible velocities for an AD region

To derive the exact admissible velocities, first we define the relative distance between the robot and the RVC by introducing the curve generated using predicted velocities. More precisely, if $\omega(t + \Delta t) \neq 0$ as shown in Figure 5, a curve line generated by extending along the predicted trajectory under the assumption that the robot keeps the predicted velocities $v(t + \Delta t)$ and $\omega(t + \Delta t)$ within the $j$th RVC. The path length along the curve is defined as $\ell_j(t + \Delta t)$, and is used as the minimum distance to the $j$th RVC. The curvature radius $r(t + \Delta t)$ of the trajectory and the curvature center $O_c := (x_c(t + \Delta t), y_c(t + \Delta t))$, which are used to extend along the predicted trajectory, are defined as follows:

$$r(t + \Delta t) = v(t + \Delta t)/|\omega(t + \Delta t)|,$$

$$x_c(t + \Delta t) = x(t) - r(t + \Delta t) \sin(\theta(t + \Delta t)),$$

$$y_c(t + \Delta t) = y(t) + r(t + \Delta t) \cos(\theta(t + \Delta t)),$$

where $x(t + \Delta t) := (x(t + \Delta t), y(t + \Delta t))$ and $\theta(t + \Delta t)$ are the predicted position and posture, and $v(t + \Delta t)$ and $\omega(t + \Delta t)$ are the predicted translational and rotational velocities.

In the case of $\omega(t + \Delta t) \neq 0$, the path length can be derived according to the procedure depicted in Figure 5. Suppose the robot predicts a trajectory at point $A$ at $t = t_0$, and point $B$ is a predicted point with the predicted velocities at $t = t_0 + \Delta t$. Further, suppose that point $B$ is inside the AD region of the $j$th RVC, and the robot calculates the path length $\ell_j(t + \Delta t)$. In this case, the path length is calculated by the Equation (7) and the intersection $C$ on the boundary line of the RVC. According to the derived path length, a tuple of admissible velocities can be derived as

$$V(\ell_j(t + \Delta t)) := \frac{\sqrt{v_{j_{\text{max}}}^2 + 2\ell_j(t + \Delta t)a_{\text{max}}}}{a_{\text{max}}},$$

$$W(\ell_j(t + \Delta t)) := W(\ell_j(t + \Delta t))/r(t + \Delta t).$$

$$V(\ell_j(t + \Delta t)) := \frac{\sqrt{v_{j_{\text{max}}}^2 + 2\ell_j(t + \Delta t)a_{\text{max}}}}{a_{\text{max}}},$$

$$W(\ell_j(t + \Delta t)) := W(\ell_j(t + \Delta t))/r(t + \Delta t).$$

where $V(\ell_j(t + \Delta t))$ and $W(\ell_j(t + \Delta t))$ are the admissible translational and rotational velocities, and $r_j := v_{j_{\text{max}}}/|\omega_{j_{\text{max}}}|$ is the reference curvature radius calculated by the maximum velocities of the $j$th RVC.
Using the admissible velocities (8), it is possible to judge whether a trajectory requires deceleration, even inside the AD region. Note that if the jth RVC is an obstacle, then $r_j := v_{\text{max}}/ | \omega_{\text{max}} |$.

The above procedure can be derived for the case where $v = 0$ and $\omega = 0$ in the same manner. In the case of $v = 0$, since the motion of the robot becomes a pivot turn, $V(\ell_j(\tau + \Delta t)) = v_{\text{max}}$ and $W(\ell_j(\tau + \Delta t)) = \omega_{\text{max}}$ are available. Similarly, in the case of $\omega = 0$, since a trajectory extended along velocities becomes a line, the path length is the straight-line distance from the robot to the RVC. With a straight-line distance, the admissible velocities of these cases are $V(\ell_j(\tau + \Delta t))$. The judgment condition of these cases is the translational velocity $V(\ell_j(\tau + \Delta t))$ of Case I in (8).

### 5.2.3. Step 3: Replanning of minimum admissible velocities and trajectories

The admissible velocity for the predicted position with each RVC and AD region can be derived according to Step 1 and Step 2. By repeating Step 1 and Step 2 for every RVC and AD region and choosing the minimum velocities, the minimum admissible velocities ($v, \omega$) are calculated. Comparing ($v, \omega$) with the predicted velocities ($v(\tau + \Delta t), \omega(\tau + \Delta t)$), if either constraint is violated, then the robot calculates a new acceleration and new predicted velocities defined as

$$
\begin{align*}
\text{Case I} & \ (r(\tau + \Delta t) \geq r_j) \text{ or } \omega(\tau + \Delta t) = 0 : \\
& \ a_{v_{\text{new}}} := (v - v(\tau))/\Delta t - a_{v_{\text{max}}} \\
& \ v(\tau + \Delta t) := v(\tau) + a_{v_{\text{new}}} (\tau + \Delta t) \Delta t \\
& \ \omega(\tau + \Delta t) := \text{sgn}(\omega) r(\tau + \Delta t) \\
& \ a_{\omega_{\text{new}}} := (\omega(\tau + \Delta t) - \omega(\tau))/\Delta t \\
\text{Case II} & \ (r(\tau + \Delta t) < r_j) : \\
& \ a_{v_{\text{new}}} := (\omega - \omega(\tau))/\Delta t \\
& \ \omega(\tau + \Delta t) := \omega(\tau + \Delta t) + a_{\omega_{\text{new}}} (\tau + \Delta t) \Delta t \\
& \ v(\tau + \Delta t) := \omega(\tau + \Delta t) r(\tau + \Delta t) \\
& \ a_{v_{\text{new}}} := (v(\tau + \Delta t) - v(\tau))/\Delta t
\end{align*}
$$

(9)

where $\text{sgn}(\omega)$ is a signum function. As defined in Equation (9), the admissible velocities are used to obtain new accelerations and velocities while maintaining the curvature radius. Then, the admissible predicted trajectory is also recomputed. In the case depicted in Figure 5: if the robot is at point $D$, then the path length will become longer than the path length at point $B$. As a result, the robot can enter the RVC while clearly satisfying its velocity constraints.

### 5.2.4. Overall algorithm

Finally, we explain the details of the proposed method according to Algorithm 2. A tuple of the minimum admissible velocities ($v, \omega$) is set to the maximum velocities, and the curvature radius is calculated via Equation (7) (line 1). Then, the predicted position is judged in terms of whether it belongs to either an RVC or an AD region. If the predicted position is in an RVC, the tuple of the minimum admissible velocities is updated according to Algorithm 3 (line 5). However, if the predicted position is inside an AD region, a path length $\ell_j(\tau + \Delta t)$ is computed. After that, a tuple of admissible velocities for the AD region is also computed using Equation (8) (line 7). Then, the minimum admissible velocities are updated according to Algorithm 3 (line 8). The above procedure is repeated for every RVC and AD region, and the minimum admissible velocities are derived (line 2–10). If the predicted velocities violate any of the velocities in the tuple of the minimum admissible velocities, a new acceleration is computed through Equation (9). After this, the predicted velocities and position are recomputed (line 12). Finally, the altered position $z(\tau + \Delta t)$ and velocities $(v(\tau + \Delta t), \omega(\tau + \Delta t))$ are returned (line 14). Note that the objective function in our method is $\Omega := NF$, which is a function that indicates how much motion is expected to reduce the value of NF during the next sampling time.

### 6. Numerical examples

To evaluate the effectiveness of the proposed method, we carried out some numerical simulations. First, we carried out simulations to demonstrate the basic properties of our method with the illustrative examples in Section 6.1. Then, we also carried out simulations with randomly generated maps to demonstrate the feasibility and time cost of the proposed method in Section 6.2. Note that values for the parameters used in all numerical examples are listed in Table 1, and the tuples $(a_v, a_\omega)$ are made from equably discretized accelerations. The algorithm was implemented using Matlab R2012b on a personal computer (Windows 7 64 bit, CPU: Core i7 2.8 GHz and RAM 16 GB).

| Table 1. Parameter settings of whole simulations. |
|-----------------------------------------------|
| **Notation** | **Value (unit)** |
| Size of the cell using NF | 0.5 m x 0.5 m |
| Number of candidate: $n$ | 30 counts |
| Number of acc. discretely | $a_v$: 5 counts, $a_\omega$: 5 counts |
| Count of the tuple | Total is 25 counts |
| Max translational vel.: $v_{\text{max}}$ | 1.0 m/s |
| Max rotational vel.: $\omega_{\text{max}}$ | 0.8 rad/s |
| Max translational acc.: $a_{v_{\text{max}}}$ | 1.0 m/s$^2$ |
| Max rotational acc.: $a_{\omega_{\text{max}}}$ | 0.8 rad/s$^2$ |
| Evaluation time: $T$ | 0.8 s |
| Sampling time: $\Delta t$ | 0.2 s |

ADVANCED ROBOTICS 643
### Table 2. Configuration of the velocity constraints used in simulations.

| Case | $v_{1\text{max}}$ [m/s] | $\omega_{1\text{max}}$ [rad/s] | $v_{2\text{max}}$ [m/s] | $\omega_{2\text{max}}$ [rad/s] |
|------|----------------|----------------|----------------|----------------|
| 1    | 0.5 (50% of Max.) | 0.8 (Max.)     | 0.5 (50% of Max.) | 0.8 (Max.)     |
| 2    | 0.5 (50% of Max.) | 0.8 (Max.)     | 0.8 (80% of Max.) | 0.8 (Max.)     |
| 3    | 0.5 (50% of Max.) | 0.4 (50% of Max.) | 0.5 (80% of Max.) | 0.4 (50% of Max.) |
| 4    | 0.5 (50% of Max.) | 0.4 (50% of Max.) | 0.8 (80% of Max.) | 0.64 (80% of Max.) |

### 6.1. Numerical examples with the illustrative map

First, we show four numerical examples with different RVC configurations as described in Table 2. The map used in the simulations has a size of 2.5 m × 4.5 m (i.e., 5 cell × 9 cell) including two known obstacles and two RVCs as shown in Figure 6(a).

Figures 6–9 show the simulation results of cases 1–4, where panels (a) and (b) in each figure show the generated trajectory and the time response of the admissible and actual velocities of the selected trajectory, respectively. In panel (a) of Figures 6–9, the solid lines show the predicted trajectory candidates, and the broken lines show an optimal trajectory selected based on the cost...
of the objective function. In panel (b) of Figures 6–9, the solid lines show the actual velocity of the selected trajectory, and the broken lines show each admissible translational and rotational velocity at the predicted position.

Throughout the simulations, we found that the obtained trajectory converges to the reference position even when taking into account the RVCs and when the velocity constraints are satisfied. We can also see that the obtained trajectories are different from each other because of the velocity constraints of the RVCs. Furthermore, since the proposed method considers the posture of the robot, the generated trajectory passes near obstacles without collision, as can be seen from Figures 6–9.

Comparing cases 1 and 2, which have RVCs with constraints only on the translational velocity, we see that the time required to reach the goal position in case 2 is shorter than that in case 1. It is obvious from the settings that the translational velocities of the 2nd RVC of case 1 and case 2 are constrained to 50% and 80% of each maximum value, respectively. However, as can be seen from Figure 6, a sharp turn motion occurs in the 1st RVC between 2 and 4 s. Although sharp turn motions should be prevented, motion planning without a rotational velocity...
Figures 8 and 9 have no sharp turns, whereas those in
Figure 11. The developed autonomous mobile robot.

constraint might still cause a trajectory with a sharp turn in the RVCs. Moreover, comparing cases 3 and 4, which have two RVCs with both translational velocity and rotational velocity constraints, the selected trajectories in Figures 8 and 9 have no sharp turns, whereas those in cases 1 and 2 have sharp turns. To summarize, we found that our method works well and that the rotational velocity constraints contribute to preventing sharp turn motion.

Table 3. Parameters used in the experiments.

| Notation                      | Value (unit)  |
|-------------------------------|---------------|
| Number of horizon: $h$        | 3 counts      |
| Size of the cell using NF     | 1 m × 1 m     |
| Number of candidates: $n$     | 30 counts     |
| Number of acc. discretely     | 5 counts      |
| Count of the tuple            | 5 counts      |
| Max translational vel.: $v_{\text{max}}$ | 1.0 m/s    |
| Max rotational vel.: $\omega_{\text{max}}$ | 0.8 rad/s    |
| Max translational acc.: $a_{\text{max}}$ | 1.0 m/s²    |
| Max rotational acc.: $\alpha_{\text{max}}$ | 0.8 rad/s²   |
| Evaluation time: $T$          | 0.8 s        |
| Sampling period: $\Delta t$   | 0.2 s        |
| Control period: $\Delta t_c$  | 0.1 s        |
| Range of Laser range scanner  | 15 m         |

Table 4. Velocity constraints in the indoor experiments.

| Case | Translational velocity constraint (m/s) | Rotational velocity constraint (rad/s) |
|------|----------------------------------------|---------------------------------------|
| 1    | 1.0 (Max.)                             | 0.8 (Max.)                            |
| 2    | 0.8 (80% of Max.)                      | 0.8 (Max.)                            |
| 3    | 0.8 (80% of Max.)                      | 0.4 (50% of Max.)                     |

Figure 10. Examples of the simulation results by the proposed method with randomly generated maps.
Figure 12. The environment for the indoor experiments.

Figure 13. The trajectories of the robot in each case (indoor).

Figure 14. The reference path in the outdoor experiments and the configuration of the RVC.

6.2. Numerical examples with randomly generated maps

To demonstrate the feasibility and computational time of the proposed method from a statistical perspective, simulations were performed consisting of 200 maps, each having a different configuration of 100 obstacles and 100 RVCs. The start position was located at the left bottom side of the map and the goal position was located at the top right side. The translational and rotational velocity constraints of each RVC were each set as 50% of the maximum values.

Figure 10 shows examples of the simulation results using the proposed method. As a result, trajectories could reach the reference position without getting trapped into local minima in all cases. Moreover, the velocity constraints are also satisfied clearly, but we omit the details of these owing to space limitations. In the case of conventional motion planning methods in related work, which select a distant path from the obstacle region for optimal and safe motion planning, the generated trajectory tends to be longer and with a short radius of curvature. In contrast to such methods, the trajectory using the proposed method is smooth, as shown in Figure 10. In other words, because the proposed motion planning method generates trajectories considering the posture of the robot to enter the RVC, the method selects a trajectory without changing the curvature radius as much as possible. Furthermore, the computation time during one evaluation
time $T$ was about 0.01 s, and the total computation time was about 220 s.

7. Experiments

Experiments were conducted to demonstrate the effectiveness of the proposed motion planning method. Figure 11 is a photo of the developed autonomous mobile robot. This robot was developed based on RMP200 (Segway inc.) and is equipped with a GPS, a fiber optic gyro sensor, encoders, and three laser range scanners. In the experiments, only one laser range scanner (UTM-30LX, HOKUYO AUTOMATIC CO., LTD.) located at the bottom was used for obstacle avoidance, and GPS was not available because of the satellite conditions. The robot is capable of traveling about 500 m according to gyroodometry [45] using the installed fiber optic gyro sensor (HOFG-OLC, HITACHI Cable Ltd.) with a position error of less than 2 m (accurate enough for the experiments in this study). [46] To apply the proposed motion planning method, we extended the algorithm so as to allow unknown obstacle avoidance to be solved in real-time (Appendix 1). Values of the parameters used in all experiments are listed in Table 3, and the algorithm described in Appendix 1 was implemented with OpenRTM-aist [47] on a personal computer (Ubuntu 12.10, CPU: Core i7 2.4GHz and RAM 8GB).

7.1. Experiments in the indoor environment

To evaluate the proposed method with an extension for real-time operation and unknown obstacle avoidance, we carried out three kinds of experiments including one RVC with different configurations as in Table 4 and two unknown obstacles. The experimental environment was a straight line hallway whose length was almost 20 m from the start point to the goal point. One RVC was set in the middle of the environment as shown in Figure 12(a) and two unknown obstacles were located in the RVC as in Figure 12(b).

Figure 13 shows the experimental results of cases 1, 2, and 3. As can be seen from Figure 13, the robot avoided collisions with the two unknown obstacles in all cases, and could reach the goal. Furthermore, the translational and rotational velocity constraints were satisfied in all cases.

Focusing on case 2, the robot tried to avoid the obstacle by making a short curve around 11–13 m. In other words, the robot had no choice but to turn sharply for collision avoidance because of the constraints on the translational velocity in case 2. However, in cases 1 and 3, the robot prevents sharp turns during motion planning, and the trajectories became a similar pattern. Therefore, as mentioned in Section 6, if the translational velocity is constrained, then the rotational velocity should also be constrained at the same time to eliminate the predicted trajectory of the sharp turn.

7.2. Experiments in the outdoor environment

The experiments were carried out around the C3 building in Kyoto University as shown in Figure 14. Figure 14(a) shows the satellite photo of the experimental environment and the red arrows show the reference path of the robot (almost 500 m). We set three RVCs in the environment as shown in Figure 14(b), because the first and third RVCs were the crossroads, and the second one was the corner where the robot might collide with unknown obstacles appearing suddenly. The translational and rotational velocity constraints in each RVC were set as 50% of the maximum values. The start and goal points were set at $(x_{\text{start}}, y_{\text{start}}) = (0\, \text{m}, 0\, \text{m})$ and $(x_{\text{goal}}, y_{\text{goal}}) = (52\, \text{m}, 7\, \text{m})$, respectively.

The experiments were performed seven times, and Figures 15 and 16 show the experimental results. Figure 15 shows the trajectory estimated by gyroodometry, and Figure 16 shows the time response of the translational and rotational velocities. In Figure 15, the solid line shows the
trajectory of the robot estimated by gyroodometry, the x marks indicate waypoints, the three colored squares are the RVCs, and the square of the final waypoint is the goal region. In Figure 16, the solid lines show the selected velocities, and the dashed lines show the admissible velocities.

From Figure 15, we see that the trajectory was along whole waypoints, and reached the final goal. Despite some unknown obstacles existing in the environment during the experiments, the resulting trajectory was not affected. In all the experiments, the translational and rotational velocity constraints were satisfied as shown in Figure 16.

8. Conclusions

We have proposed the concept of named regions with velocity constraints, which aims to realize safe autonomous locomotion in a public environment, including in the presence of people. The proposed RVC constrains the translational and rotational velocities of a robot; the translational velocity constraint is to achieve safe and reliable collision avoidance, and the rotational velocity constraint is to eliminate sharp turns, considering the impact of humans.

We have also proposed a motion planning method for a robot in an environment with RVCs, in which the translational and rotational velocities of the robot are constrained to predefined values while inside RVCs. One of the advantages of the proposed method is that it reduces the sharp turn motions that affect humans around the robot by introducing a curvature condition. We have also verified the proposed method via some numerical examples. Furthermore, we applied the algorithm with extensions for unknown obstacle avoidance and real-time operation to a real mobile robot, and showed the effectiveness of our method in experiments. From the results, we confirmed that our method works well even with unknown obstacles.

In future work, we would like to derive appropriate constraints depending on the environment automatically. Furthermore, we would like to realize the motion planner based on RRT* [48] which introduces gradient-based optimization to increase the global optimality, and increase computational efficiency.

Disclosure statement

No potential conflict of interest was reported by the authors.
Notes on contributors

Tae Hyon Kim received BS degree from the Department of Mechanical Engineering, Waseda University, Tokyo, Japan in 2006 and ME from the Department of Mechanical Engineering and Science, Kyoto University, Kyoto, Japan in 2012. He is currently pursuing the PhD degree with the Department at Mechanical Engineering and Science, Kyoto University, Kyoto, Japan. He was a visiting researcher of Robotics Institute of Carnegie Mellon University for four months under the support of JSPS Institutional Program for Young Researcher Overseas Visits in 2012. His current research interests are a motion planning, a human-robot interaction, a machine learning, and an image processing.

Kazuyuki Kon received his BS and MS in Engineering from University of Electro-Communications, Japan, in 2005 and 2007, and PhD degree in Engineering from Kyoto University, Japan, in 2010, respectively. From 2009 to 2011, he was as a Research Fellow of Japan Society for the Promotion of Science. From 2011 to 2014 he was an Assistant Professor of Kyoto University, Japan. Currently he is currently with NEC Corporation, Tokyo. His research interests include autonomous mobile robots and formation control. He is a member of the IEEE, SICE and RSJ.

Fumitoshi Matsuno (M’94) received a PhD (Dr Eng) from Osaka University, Osaka, Japan, in 1986. In 1986, he joined the Department of Control Engineering, Osaka University. He became a lecturer and associate professor in 1991 and 1992, respectively, in the Department of Systems Engineering, Kobe University. In 1996, he joined the Department of Computational Intelligence and Systems Science, Interdisciplinary Graduate School of Engineering Sciences, Kyushu University as an associate professor. In 2003, he became a professor with the Department of Mechanical Engineering and Intelligent Systems, University of Electro-Communications. Since 2009, he has been the associate professor with the Department of Mechanical Engineering and Science, Kyoto University, Kyoto, Japan. He is also the Vice-President of NPO International Rescue Systems Institute (IRS). His current research interests include robotics, control of distributed parameter systems and nonlinear systems, rescue support systems in fires and disasters, and swarm intelligence. He has received many awards, including the Outstanding Paper Award in 2001 and 2006, the Takeda Memorial Prize in 2001 from the Society of Instrument and Control Engineers (SICE), and the Outstanding Paper Award in 2013 from the Information Processing Society of Japan. He is a fellow member of SICE, the Japan Society of Mechanical Engineers, the Robotics Society of Japan, and a member of IEEE, and the Institute of Systems, Control and Information Engineers, among other organizations. He served as the co-chair of the IEEE Robotics and Automation Society Technical Committee on Safety, Security, and Rescue Robotics; the chair of the Steering Committee of the SICE Annual Conference; and is an editor for the Journal of Intelligent and Robotic Systems, an associate editor for Advanced Robotics and International Journal of Control, Automation, and Systems among others, and an editorial board member with the Conference Editorial Board of the IEEE Control Systems Society.

References

[1] Roomba of iRobot co. Available from: http://www.irobot.com/us/learn/home/roomba.
[2] Wurman PR, D’Andrea R, Mountz M. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. AI Mag. 2008;29:9–20.
[3] Alberto Yukinobu H, Wolf D. Terrain mapping and classification using support vector machines. In: Proceeding of IEEE Latin American Robotics Symposium. 6th Latin American; 2009 Oct; Valparaiso, Chile. p. 1–6.
[4] Milford M, Wyeth G. Hybrid robot control and SLAM for persistent navigation and mapping. Rob. Auton. Syst. 2010;58:1096–1104.
[5] Kleiner A, Dornhege C. Mapping for the support of first responders in critical domains. J. Intell. Rob. Syst. 2011;64:7–31.
[6] Steven L. Planning algorithm. Cambridge: Cambridge University Press; 2006.
[7] Aoude GS, Luders BD, Joseph JM, et al. Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. Auton. Robots. 2013;35:51–76.
[8] Howard TM, Pivtoraiko M, Knepper RA, et al. Model-predictive motion planning. IEEE Rob. Autom. Mag. 2014;21:64–73.
[9] Masehian E, Katebi Y. Robot motion planning in dynamic environments with moving obstacles and target. Int. J. Mech. Syst. Sci. Eng. 2007;1:20–25.
[10] Belkhouche F. Reactive path planning in a dynamic environment. IEEE Trans. Rob. 2009;25:902–911.
[11] Rufli M, Alonso-Mora J, Siegwart R. Reciprocal collision avoidance with motion continuum constraints. IEEE Trans. Rob. 2013;29:899–912.
[12] Fraichard T, Asama H. Inevitable collision states: a step towards safer robots? Adv. Rob. 2004;18:1001–1024.
[13] Rufli M, Ferguson D, Siegwart R. Smooth path planning in constrained environment. In: Proceeding of 2009 IEEE International Conference on Robotics and Automation; 2009 May; Kobe, Japan. p. 3780–3785.
[14] Calderon CAA, Zhou C, Mohan RE. Development of an autonomous service robot for social interactions. In: Proceeding of the IEEE International Conference Information, Communications and Signal Processing; 2011 Dec; Singapore. p. 1–6.
[15] Morales Y, Carballo A, Takeuchi E, et al. Autonomous robot navigation in outdoor cluttered pedestrian walkways. J. Field Rob. 2009;26:609–635.
[16] Hall E. The hidden dimension. New York, NY: Anchor Books; 1996.
[17] Ohki T, Nagatani K, Yoshida K. Collision avoidance method for mobile robot considering motion and personal space of evacuees. In: Proceeding of the IEEE International Conference Intelligent Robots and Systems; 2010 Oct; Taipei, Taiwan. p. 1819–1824.
[18] Scandolo L, Fraichard T. An anthropomorphic navigation scheme for dynamic scenarios. In: Proceeding of the IEEE International Conference Robotics and Automation; 2011 May; Shanghai, China. p. 809–814.
[19] Noguchi H, Yamada T, Mori T, et al. Mobile robot path planning using human prediction model based on massive trajectories. In: Proceeding of the IEEE
Karumanchi S, Iagnemma K. Reactive control in environments with hard and soft hazards. In: Proceeding of the IEEE International Conference Intelligent Robot and Systems; 2010 Oct; Taipei, Taiwan. p. 3269–3274.

Karaman S, Frazzoli E. Sampling-based optimal motion planning for non-holonomic dynamical systems. In: Proceeding of the IEEE International Conference Robotics and Automation; 2013 May; Karlsruhe, Germany. p. 5041–5047.

Otto M, Frazzoli E. RRT*: real-time motion planning/replanning for environments with unpredictable obstacles. In: Algorithmic foundations of Robotics XI. Springer International Publishing, Cham, Switzerland; 2015. p. 461–478.

Brock O, Khatib O. High-speed navigation using the global dynamic window approach. In: Proceeding of the IEEE International Conference on Robotics and Automation; 1999 May; Detroit, MI, USA. p. 341–346.

Latombe J. Robot motion planning. Dordrecht: Kluwer Academic; 1991.

Fox D, Burgard W, Thrun S. The dynamic window approach to collision avoidance. IEEE J. Rob. Autom. Mag. 1997;4:23–33.

Ögren P, Leonard N. A convergent dynamic window approach to obstacle avoidance. IEEE J. Rob. Autom. Mag. 1997;4:23–33.

Stoneman S, Lampariello R. Embedding nonlinear optimization in RRT* for optimal kinodynamic planning. In: IEEE Conference on Decision and Control, 2014 IEEE 53rd Annual Conference; 2014 Dec; Los Angeles, CA, USA. p. 3737–3744.