KidVO: a kinodynamically consistent algorithm for online motion planning in dynamic environments

Mostafa Mahmoodi
Faculty of Industrial and Mechanical Engineering, Islamic Azad University, Qazvin Branch, Qazvin, Iran
Khalil Alipour
Department of Mechatronics Eng., Faculty of New Sciences and Technologies, University of Tehran, Tehran, Iran, and
Hadi Beik Mohammadi
Mechatronics Research Laboratory, Faculty of Electrical, Computer and IT Engineering, Islamic Azad University, Qazvin Branch, Qazvin, Iran

Abstract
Purpose – The purpose of this paper is to propose an efficient method, called kinodynamic velocity obstacle (KidVO), for motion planning of omnimobile robots considering kinematic and dynamic constraints (KDCs).
Design/methodology/approach – The suggested method improves generalized velocity obstacle (GVO) approach by a systematic selection of proper time horizon. Selection procedure of the time horizon is based on kinematical and dynamical restrictions of the robot. Toward this aim, an omnimobile robot with a general geometry is taken into account, and the admissible velocity and acceleration cones reflecting KDCs are derived, respectively. To prove the advantages of the suggested planning method, its performance is compared with GVOs, the so-called Hamilton-Jacobi-Bellman equation and the rapidly exploring random tree.
Findings – The obtained results of the presented scenarios which contain both computer and real-world experiments for complicated crowded environments indicate the merits of the suggested methodology in terms of its near-optimal behavior, successful obstacle avoidance both in static and dynamic environments and reaching to the goal pose.
Originality/value – This paper proposes a novel method for online motion planning of omnimobile robots in dynamic environments while considering the real capabilities of the robot.

Keywords Mobile robots, Dynamic environment, Kinematic constraints, Obstacle avoidance

Paper type Research paper

1. Introduction

A fundamental ability of an autonomous mobile robot is to plan a collision-free path from a start to a goal position among a collection of stationary/moving obstacles. As a result, any effective algorithm enabling the robot to receive the user commands in high level, and produce the robot motion is welcome. Some researchers have tried to propose motion/path planning algorithms during past decades. However, a large and growing body of relevant literature has well focused on immobile obstacles (Canny and Lin, 1993; Kimmel *et al.*, 1998; Hsu *et al.*, 2002; Zucker *et al.*, 2007; Berg and Overmars, 2007; Petti and Fraichard, 2005; Aivar *et al.*, 2008; Fulgenzi *et al.*, 2009; Xu *et al.*, 2010; Cherubini *et al.*, 2013), in which more emphasis has been placed on global planning. Global or complete path planning can be performed to enable navigation beyond local regions. Some of these planners such as (Kimmel *et al.*, 1998; Canny and Lin, 1993; Hsu *et al.*, 2002; Zucker *et al.*, 2007; Berg and Overmars, 2007) have been developed for planning in dynamic environments and are applicable to omnimobile robots.

In general case, a path planning system needs to collect data from surroundings to generate motion commands. On the other hand, some of these studies such as Kant and Zucker (1986), Berg and Overmars (2007) and Wang and Qi (2001) solve the planning problem in two steps: first, computing a feasible path and second, adding the time dimension to the robot's configuration space along the path and generating a velocity profile for the robot to safely accomplish the desired tasks. As these global planners lack enough robustness due to environment uncertainties, these cannot be used for situations of dynamically changing environments. Moreover, due to the inherent complexity involved in motion planning problem, computing a complete motion to the goal may not be achievable at once. One way to compute a full path from a start to a goal point is using rapidly exploring random tree

Industrial Robot: An International Journal
43/1 (2016) 33–47
© Emerald Group Publishing Limited [ISSN 0143-991X]
[DOI 10.1108/IR-05-2015-0096]

Received 19 May 2015
Revised 25 August 2015
4 September 2015
Accepted 7 September 2015

(RRT) (Kuffner and LaValle, 2000), which is designed to efficiently search through high-dimensional space. Although the RRT algorithm swiftly computes candidate feasible solutions, it is not suitable to plan motion in crowded dynamically changing situations. In addition to RRT, other methods exist to compute complete path to the goal. Shortest path algorithms such as A* (Nilsson, 1982) and its derivatives, e.g. D*-lite (Koenig and Likhachev, 2002), guarantee path completion to the goal. These approaches do not take into account how fast the robot will get to a desired location.

Other than global planners, local or reactive planners can be used to generate or modify the robot trajectories to reach a goal (Berg and Overmars, 2005; Vannoy and Xiao, 2008; Toit and Burdick, 2012; Jing, 2005; Kuwata *et al.*, 2009). These planners are well-suited alternatives to global planners. Some studies such as Fox *et al.*, (1997), Brock and Khatib (1999) and Minguez and Montano (2000) are examples of reactive planners, which are developed for slow dynamic systems. So, these approaches have poor performance in predicting future states of collisions. Furthermore, most of these planners do not address the issue of safety due to limited ability to avoid the inevitable collision states (ICS). Some of the eminent works in the reactive planning are based on velocity space technique (Fiorini and Shiller, 1998; Fox *et al.*, 1997; Simmons, 1996; Wilkie *et al.*, 2009). This technique addresses KDCs by modeling elements such as robot kinematics, actuator configuration and the robot's current and obtainable velocities. A well-known example of this technique is the notion of the so-called *Velocity Obstacles* (VO), which has been proposed by Fiorini and Shiller (1998). VO is the set of all robot velocities, resulting in collision with an obstacle moving by a prescribed velocity. Another velocity-based obstacle avoidance algorithm is called generalized velocity obstacle (GVO) (Wilkie *et al.*, 2009), which is in fact an extension of the VO concept. It offers a method to identify velocities for the robot that will allow avoiding collision with a moving obstacle at some future time. The main advantage of the GVO notion in comparison to the VO is its simple and practical notations to describe the set of insecure velocities, upon considering kinematic constraints of the robot. In addition, some other reactive planning approaches are available, e.g. artificial potential field (APF) (Khatib, 1985) and vector field histogram (VFH) (Borenstein and Koren, 1991). In APF method, one has to attentively generate two antithetical potential functions (attractive and repulsive) such that the combined potential function provides the preferred result. Hence, while APF method is vastly applicable for a static situation, it is extremely hard to use the technique for a dynamic environment. To remove intrinsic limitations of APF approach, VFH has been proposed. But it needs a grueling procedure of manual parameters adjustment, making it unpractical to be used for a changing or dynamic environment.

To date, a few studies such as Wu (2005) and Purwin and D'Andrea (2006) in the path and trajectory planning have taken into account the KDCs of the robot. In Wu (2005), the concepts of velocity and acceleration cone for an omnimobile robot are proposed, by which the KDCs of the robot is being taken into account. The aforementioned study, however, does not consider the weight transfer effect which is generated due to robot acceleration.

This paper proposes a novel method for online motion planning of omnimobile robots in dynamic environments while considering the real capabilities of the robot. The novelties introduced throughout this study can be listed as follows:

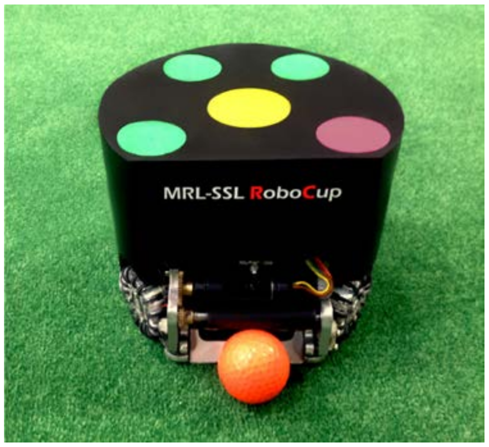
- It derives the feasible velocity and acceleration spaces for a wheeled platform, the so-called mechatronics research laboratory-small size league (MRL-SSL) RoboCup robot (Figure 1) with asymmetric geometry while considering the whole dynamic factors, such as weigh transfer, affecting the robot. These spaces, which represent the robot limitations from the viewpoint of kinematical and dynamical issues, can then be used for kinodynamically consistent selection of horizon time.
- It improves the GVO method by suggesting a systematic approach for nominating appropriate horizon time. As seen later, this time horizon generates a near-optimal movement of the system.
- In contrast to the most of velocity-space-based approaches (Fiorini and Shiller, 1998; Gal *et al.*, 2009; Berg *et al.*, 2011; Shiller *et al.*, 2011), which do not derive the real KDCs of the robot and apply their techniques just on the computer simulations, the current study derives the KDCs (Mahmoodi *et al.*, 2014), and the performance of the presented method has been examined not only via computer simulations but also through real-world experiments.

The remainder of this paper is organized as follows. The bases of kinodynamic velocity obstacle (KidVo) including collision avoidance procedure, time horizon and ICS are first explained. The admissible velocity and acceleration spaces are then derived addressing the KDCs in Section 3. The way of exploitation of KidVO is illustrated in Section 3.2. To support the usefulness of the suggested method, its response has been compared to the other planning methods in Section 4. Some concluding remarks are drawn in the last section along with several directions for further research and development.

2. Elements of the KidVO approach

In the current section, we first explore how the data from environments and obstacles are collected for the robot navigation? We address this question by describing our

Figure 1 The MRL-SSL RoboCup robot



external tracking system (ETS). ETS plays a vital role in successfully accomplishing the desired tasks by the robot. According to Figure 2, the SSL-Vision camera captures global images of the environment by 60 fps (Li *et al.*, 2010). The computer vision system processes these images to identify and locate the robot and neighboring obstacles. These data are sent to the multi-stage planning system (MSPS). MSPS is a high-level planner which computes the robot's actions and drives it to move around the field by wireless communication. This high-level representation leads to reduced effort for planning robot motion (Behnisch *et al.*, 2010). One of the most crucial stages in MSPS, which can be obtained by exploiting the suggested motion planner, is computing collision-free trajectories for the robot. Collision detection and setting a proper time horizon are two key aspects of the suggested planner, as will be elaborated later. Time horizon is the minimum time required for the robot to successfully avoid the $\mathcal{ICS}$ (all sets in this paper are expressed by calligraphy letters) while considering its KDCs. It is also pointed out that an $\mathcal{ICS}$ for a robotic system can be defined as a state or set where there is no feasible trajectory to avoid collision with an obstacle. $\mathcal{ICS}$ takes into account both the KDCs of the robot and the obstacles.

2.1 Collision detection procedure

In this section, a procedure to detect a collision between the robot and stationary/moving obstacles is presented based on GVO technique. The GVO concept defines the obstacle in the velocity space as:

$$\{\mathbf{v} \,|\, \exists\, t > 0, \|A(t,\mathbf{v}) - B(t)\| < r_{AB}\}. \tag{1}$$

Where $B(t)$ defines the trajectory of the moving obstacle B, and $A(t, \mathbf{v})$ is the trajectory of the robot A for a given velocity $\mathbf{v}$. Additionally, $r_{AB} = r_A + r_B$ (Figure 3), and $\|.\|$ stands for the 2-norm of its vector argument. To determine when a collision occurs, one should compute the time at which the distance between A and B is minimal for a given velocity $\mathbf{v}$. Then, the following optimization problem should be solved:

$$t_{\min}(\mathbf{v}) = \arg\min_{t>0} \|A(t,\mathbf{v}) - B(t)\|. \tag{2}$$

If a closed-form solution for equation (2) is obtained, then the $\mathcal{VO}_{AB}$ (Figure 3) could be computed by an explicit equation upon solving the following equation:

$$\|A(t_{\min},\mathbf{v}) - B(t_{\min})\| < r_{AB}. \tag{3}$$

Figure 2 Diagram of the ETS and MSPS architecture

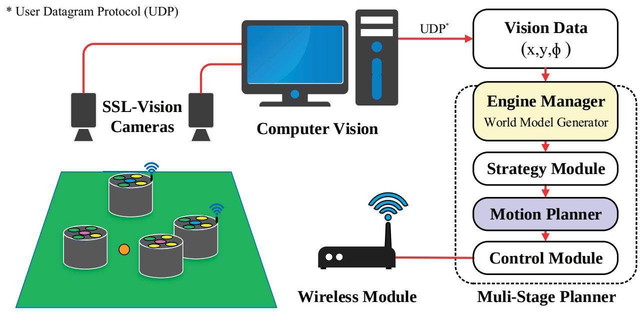
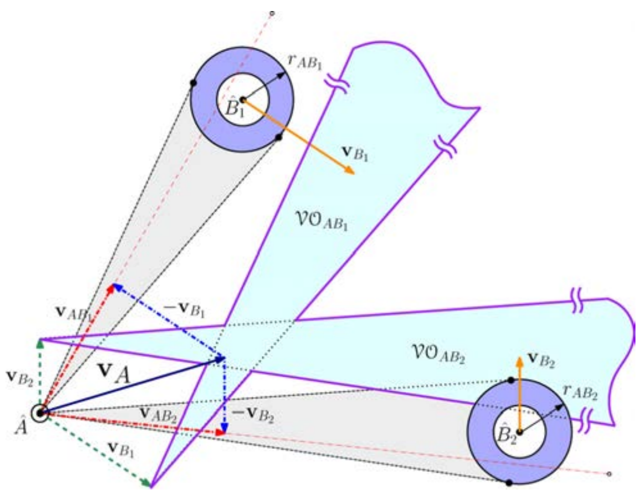


Figure 3 VOs for obstacles B_1 and B_2



According to the above relation, $\mathcal{VO}_{AB}$ can be defined as the set of all admissible robot velocities, which satisfy equation (3). To clarify this approach by refering to Figure 4, assume $A(t, \mathbf{v}_A) = \mathbf{p}_A + \mathbf{v}_A \cdot t$ and $B(t) = \mathbf{p}_B + \mathbf{v}_B \cdot t$, where $\mathbf{p}_A$ and $\mathbf{p}_B$ are the current positions of A and B, respectively. Then, solving equation (2) for t leads to:

$$t_{\min} = \frac{-\mathbf{p}_{AB} \cdot \mathbf{v}_{AB}}{\|\mathbf{v}_{AB}\|^2}. \tag{4}$$

in which $\mathbf{p}_{AB} = \mathbf{p}_A - \mathbf{p}_B$ and $\mathbf{v}_{AB} = \mathbf{v}_A - \mathbf{v}_B$ are the current relative position and velocity of A with respect to B, respectively. According to the above assumptions, equation (3) can be redefined at time t as below:

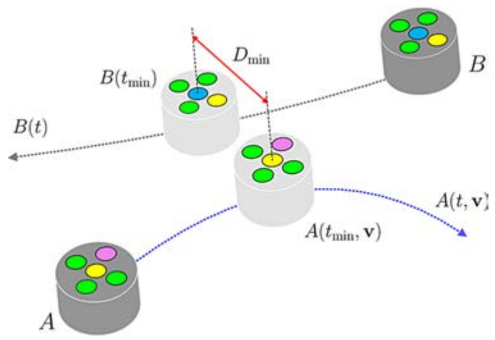
$$\|\mathbf{p}_{AB} + \mathbf{v}_{AB} \cdot t\| < r_{AB}. \tag{5}$$

Dividing both sides of equation (5) by t and rearranging leads to:

$$\left\|\mathbf{v}_{AB} - \left(-\frac{\mathbf{p}_{AB}}{t}\right)\right\| < \frac{r_{AB}}{t}. \tag{6}$$

which defines the disk of all relative velocities $\mathbf{v}_{AB}$ that allow the robot A and the moving obstacle B collide at time t.

Figure 4 The robot A is moving along trajectory $A(t, \mathbf{v})$ for a given velocity $\mathbf{v}$ and trying to avoid obstacle B that is moving along $B(t)$. The minimum distance between them occurs at $t_{\min}$ which is not satisfying equation (3). Therefore, velocity $\mathbf{v}$ is not in the $\mathcal{VO}_{AB}$



Consequently, one can define $\mathcal{VO}_{AB}$ as a union of disks in infinite time horizon as:

$$\mathcal{VO}_{AB} = \bigcup_{t_0 < t \leq \infty} \mathrm{Disk}\left(-\frac{\mathbf{p}_{\mathrm{AB}}}{\mathrm{t}} \oplus \mathbf{v}_{\mathrm{B}}, \frac{\mathbf{r}_{\mathrm{AB}}}{\mathrm{t}}\right). \tag{7}$$

2.2 Setting time horizon

The set of safe velocities of the robot for avoidance maneuvers is generated by selecting robot velocities outside of the $\mathcal{VO}_{AB}$, which is calculated based on $t_0 < t \leq \infty$, see equation (7). An infinite time horizon is safe, but overly conservative and significantly restricts the feasible velocity search space. This in turn may lead to a far-away optimal motion plan. On the other hand, setting the time horizon too small would increase the possibility of a collision. Therefore, it is important to consider a proper time horizon. As a result, in this study, a finite time horizon will be selected based on a heuristic approach presented in Gal *et al.* (2009). The robot is in safe state whenever it can avoid the obstacles either by stopping or by passing manoeuvres. To find the safe states of the robot, one should calculate the KDC-based stopping time, t_s, and passing time, t_p, for each obstacle as follows:

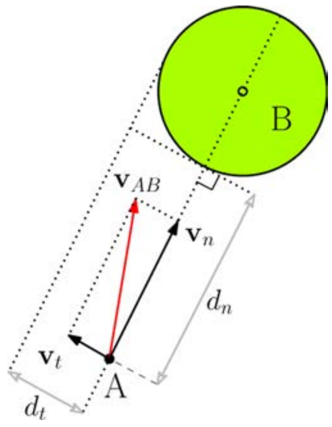
$$t_s = \frac{\mathbf{v}_n}{-2a_n}. \tag{8}$$

$$t_p = \text{positive min}\left(\frac{-\mathbf{v}_t \pm \sqrt{\mathbf{v}_t^2 + 2a_t d_t}}{a_t}\right). \tag{9}$$

Where $a_n < 0$ is the constant longitudinal deceleration and a_t is the constant lateral acceleration, which are elaborated and discussed in detail in the next section. Besides, d_n and d_t are depicted in Figure 5. As mentioned previously, we consider t_h is the minimum required time for the robot to successfully avoid the $\mathcal{ICS}$ while considering its KDCs; hence, one can conclude:

$$t_h = \min\{t_s, t_p\} \tag{10}$$

Figure 5 The robot and obstacles on a collision course with stopping and passing maneuvers by velocity $\mathbf{v}_n$ and $\mathbf{v}_t$, respectively. $\mathbf{v}_n$ and $\mathbf{v}_t$ are two components of relative velocity $\mathbf{v}_{AB} = \mathbf{v}_A - \mathbf{v}_B$ which are parallel and normal to the line connecting A and the center of B



2.3 Inevitable collision states

Safety concerns during motion planning specifically should be addressed if the robot planned trajectory involves an $\mathcal{ICS}$. As mentioned earlier, an $\mathcal{ICS}$ for a robotic system can be defined as a set where there is no feasible trajectory to avoid collision with an obstacle. It takes into account both the KDCs of the robot and the obstacles. Therefore, combining the equations (7) and (10) leads to equations (11) that is proposed to identify the $\mathcal{ICS}$. To clarify this equation, see Figure 6.

$$\mathcal{ICS} = \bigcup_{t_0 < t \leq t_h} \mathrm{Disk}\left(-\frac{\mathbf{p}_{AB}}{t} \oplus \mathbf{v}_B, \frac{r_{AB}}{t}\right). \tag{11}$$

3. Kinematic and dynamic constraints of the robot

In this section, the kinematics and dynamics of the four-wheeled MRL-SSL RoboCup robot (see Figure 1) will be discussed, and for the first time, the admissible velocity and acceleration spaces for such a platform with asymmetric geometry are derived considering various effects including mass-transfer effect. Table I represents the parameters of this robot.

3.1 Robot velocity space

Figure 7 is the free body diagram of MRL-SSL RoboCup robot. As it can be observed, $\{w\}$ is the world coordinate system and $\{b\}$ is the body coordinate system attached to the center of the robot. The kinematics equations of this robot are expressed in $\{w\}$ as follows:

$$^{w}\dot{\mathrm{X}} = {}^{w}_{b}\mathrm{R}(\phi)^{b}\dot{\mathrm{X}} = \mathrm{R}(\phi)r(\mathrm{G}^T)^{\dagger}\dot{\mathrm{q}} \tag{12}$$

Figure 6 $\mathcal{ICS}$ for obstacles B_1 and B_2

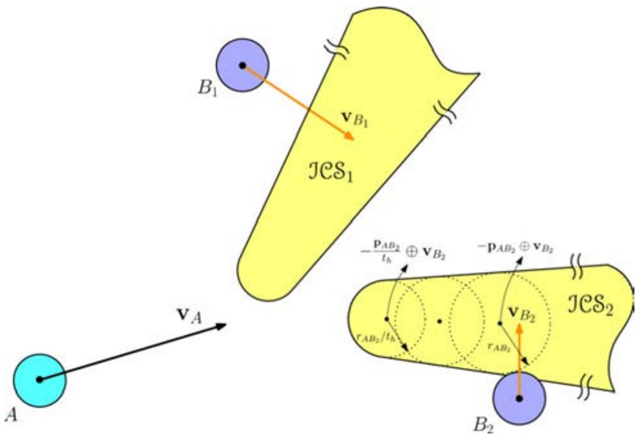
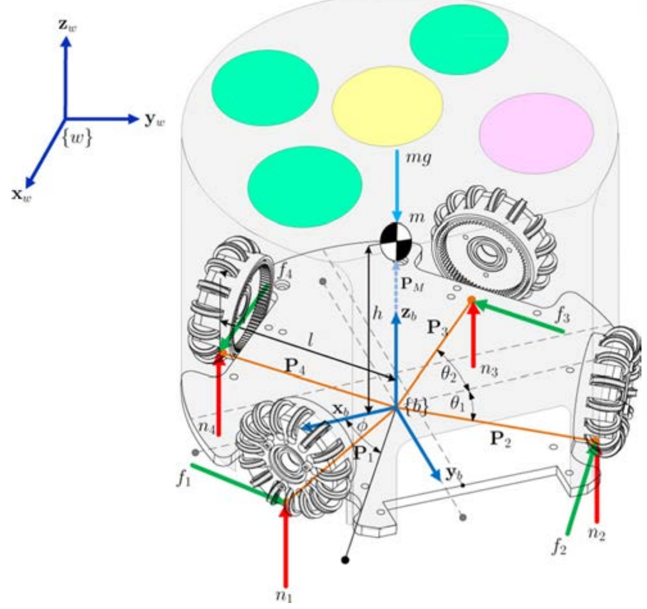


Table I MRL-SSL RoboCup robot parameters

Symbol	Quantity	SI
r	Wheel radius	0.026 (m)
l	Wheelbase	0.08 (m)
h	Center of mass height	0.052 (m)
m	Robot mass	2.5 (kg)
J	Robot inertia	0.0116 (kg.m^2)
μ	Friction coefficient	0.8
g	Gravity acceleration	9.81 (m/sec^2)

Figure 7 Free body diagram of the MRL-SSL RoboCup Robot



$$\mathbf{G} = \begin{pmatrix} -\sin\alpha_1 & -\sin\alpha_2 & -\sin\alpha_3 & -\sin\alpha_4 \\ \cos\alpha_1 & \cos\alpha_2 & \cos\alpha_3 & \cos\alpha_4 \\ l & l & l & l \end{pmatrix}_{3\times 4} \tag{13}$$

Where r and l are wheel radius and wheelbase, respectively; $\mathbf{G}$ is the geometrical matrix and $\alpha_i = (33° \; 147° \; 225° \; 315°)^T$ is the angle of each wheel with respect to $\mathbf{x}_b$; ${}^w\dot{\mathbf{X}} = (\dot{x}_w \; \dot{y}_w \; \dot{\phi}_w)^T$ and ${}^b\dot{\mathbf{X}}$ are the Cartesian velocity of the robot in $\{w\}$ and $\{b\}$, respectively; $\dot{\mathbf{q}} = (\dot{q}_1 \;\; \dot{q}_2 \;\; \dot{q}_3 \;\; \dot{q}_4)^T$ is the vector containing the wheels angular velocities. Moreover, ${}^w_b\mathrm{R}(\phi)$ is the orthonormal rotation matrix from frame $\{b\}$ to frame $\{w\}$. As described in Wu (2005), the Cartesian velocity ${}^w\dot{\mathbf{X}}$ of the robot can be determined from the wheel angular velocities $\dot{\mathbf{q}}$. First, assume $|\dot{\mathbf{q}}| \leq s$, which means that $\dot{\mathbf{q}}$ is bounded by maximum allowable wheel angular velocity s. The value of s could be obtained experimentally and, for our case, it equals $s = 1{,}280$ rev/min. Therefore, the range of ${}^b\dot{\mathrm{X}}$ can be obtained using the linear transformation ${}^b\dot{\mathrm{X}} = r(\mathrm{G}^T)^{\dagger}\dot{q}$.

As illustrated in Figure 8, the robot velocity space is not rotationally symmetric, which means that the maximum magnitude of the velocity is based on the direction of the velocity vector. The intersection-set theory (Halmos, 1960) will be used to obtain $\mathscr{H}$, which is the admissible set of velocities that the robot can achieve, independent of the current orientation ϕ which can be formulated as follows:

$$\mathscr{H} = \bigcap_{0\leq\phi\leq 2\pi} {}^w_b\mathbf{R}(\phi)^b\dot{\mathbf{X}} \;\equiv\; \left\{\mathscr{H}{:}\forall\mathscr{A} \in {}^w_b\mathbf{R}(\phi)^b\dot{\mathbf{X}}, \mathscr{H} \in \mathscr{A}\right\}. \tag{14}$$

$\mathscr{H}$ can be approximated as a right circular cone as shown in Figure 9. It is noteworthy that $\mathscr{H}$ is also known as the robot *velocity cone* (Wu, 2005; Purwin and D'Andrea, 2006). The equation of the velocity cone's generator line, as depicted in Figure 9, is as follows:

$$\phi + \frac{\dot{\phi}_{\max}}{v_{\max}}\|\mathbf{v}\| = \phi_{\max}. \tag{15}$$

The above equation will be used for the velocity filtering purpose in our robot motion planning process.

3.2 Robot acceleration space

The robot dynamics model is derived considering the approach proposed in Purwin and D'Andrea (2006), and then the maximum accessible acceleration is extracted. This method focuses on dynamics equations of robot which govern

Figure 8 The robot velocity space

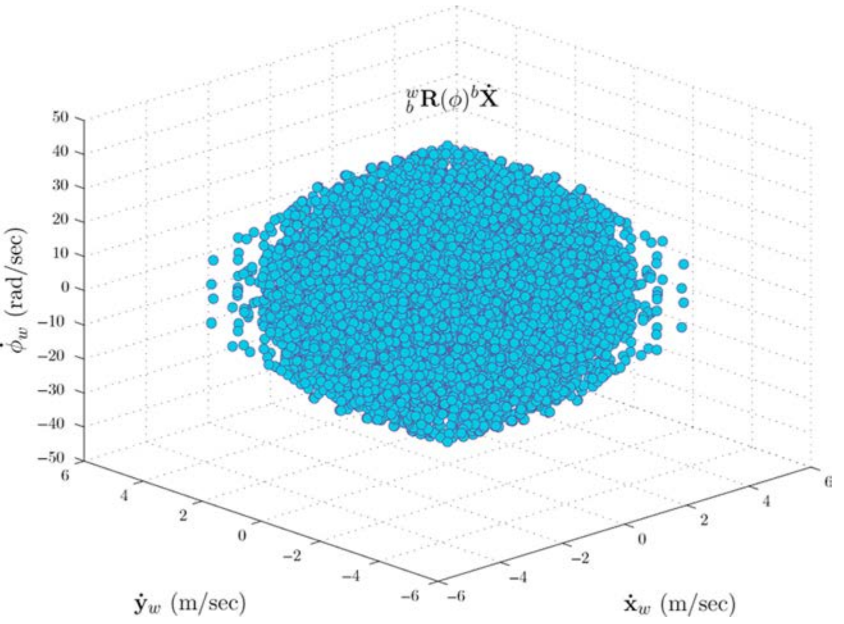
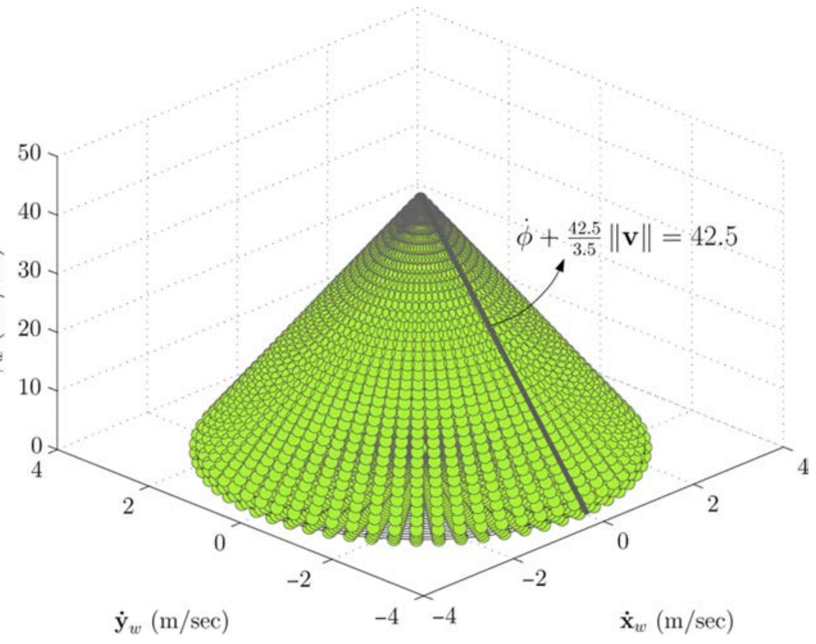


Figure 9 The robot velocity cone, $\mathscr{H}$. The maximum linear velocity is $v_{\max} = 3.5$ m/sec and maximum angular velocity is $\dot{\phi}_{\max} = 42.5$ rad/sec



the vehicle dynamics. The force and moment balance equations have been described in the world coordinate system $\{w\}$ as follows:

$$m^{w}\ddot{\mathbf{X}} = {}^{w}_{b}\mathbf{R}(\phi)\left(\sum_{i=1}^{4} f_i\mathbf{D}_i + \sum_{j=1}^{4} n_j\begin{pmatrix}0\\0\\1\end{pmatrix} - mg\begin{pmatrix}0\\0\\1\end{pmatrix}\right). \quad (16)$$

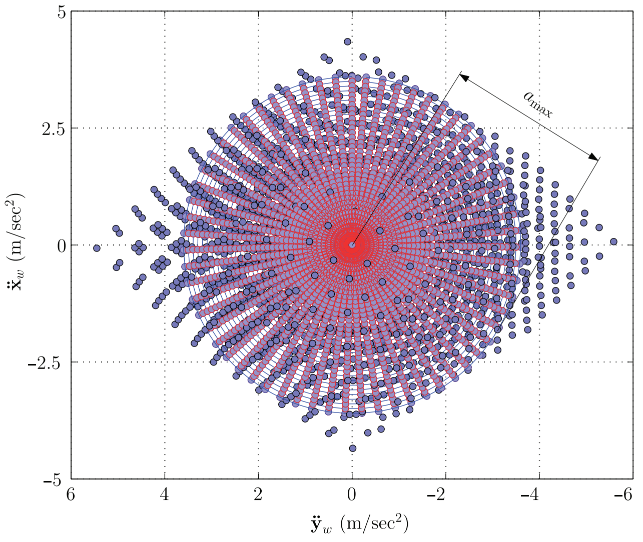
$$^{w}\dot{\mathrm{H}}_G = {}^{w}_{b}\mathrm{R}(\phi)\left(\sum_{i=1}^{4} P_{iM} \times f_i D_i + \sum_{j=1}^{4} P_{jM} \times n_i\begin{pmatrix}0\\0\\1\end{pmatrix}\right) \quad (17)$$

Where m is the robot mass; $^{w}\ddot{\mathrm{X}} = (\ddot{x}_w\, \ddot{y}_w\, \ddot{z}_w)^T$ is the Cartesian acceleration and $^{w}\dot{\mathrm{H}}_G = (\dot{H}_{Gx}\, \dot{H}_{Gy}\, \dot{H}_{Gz})^T$ is the time derivative of angular momentum of the robot in $\{w\}$. Additionally, $\mathbf{P}_{iM} = \mathbf{P}_i - \mathbf{P}_M$ and $\mathbf{P}_{jM} = \mathbf{P}_j - \mathbf{P}_M$. As it can be observed from Figure 7, f_i is the friction force, and n_i is the normal force between the wheel i and contact surface. $\mathbf{P}_M$ is the position of the center of mass and $\mathbf{P}_i$ is the position vector of wheel i with respect to the center of mass. $\mathbf{D}_i$ is the traction direction vector of wheel i which is orthogonal to the $\mathbf{P}_i$. Finally, by considering the Columb friction model for f_i and some mathematical manipulations, equation (18) could be determined which describes the robot acceleration space with control inputs $u_i \in [-1, 1]$.

$$\begin{pmatrix}\ddot{x}_w\\ \ddot{y}_w\\ \ddot{\phi}_w\end{pmatrix} = {}^{w}_{b}\mathrm{R}(\phi)\frac{\lg \mu}{\xi(u_i)}\begin{pmatrix}\psi_1(u_i)\\ \psi_2(u_i)\\ \psi_3(u_i)\end{pmatrix}., \; i = 1, \ldots, 4; \quad (18)$$

Where μ is the friction coefficient and g is the gravity acceleration. See Appendix 1 for more details on equation (18). Figure 10 shows the robot acceleration space and the corresponding *acceleration cone*, $\mathscr{Q}$, that is extracted following the same method as described in the previous section.

Figure 10 The robot acceleration space and the corresponding acceleration cone, $\mathscr{Q}$. In this figure, maximum linear acceleration for acceleration filtering is $a_{\max} = 3.6$ m/sec^2 and maximum angular acceleration is $\ddot{\phi}_{\max} = 111$ rad/sec^2



4. KidVO motion planning strategy

The form of an efficient motion planner depends heavily on properties of the robot (Choset, 2005). These properties were described for our robot in the previous section as velocity and acceleration constraints and ability to move instantaneously in any direction. Moreover, identification of the planner algorithm properties such as completeness and computational complexity is required to ensure convergence to the goal. Assuming known position and velocity for adjacent obstacles, the completeness of the proposed local algorithm cannot be guaranteed. Our motion planner approach to generate collision-free trajectory among a collection of immobile and moving obstacles is described in Algorithm 1. According to this algorithm, in the first time step, it is supposed that the robot initiates its motion from rest. Note that, the algorithm is completely general and independent of the initial velocity of the robot. The desired velocity, $\mathbf{v}_d$, is defined as follows:

$$\mathbf{v}_d = a_{\max} \cdot \Delta t \frac{\mathbf{v}_{\text{ref}}}{\|\mathbf{v}_{\text{ref}}\|}. \tag{19}$$

where $\Delta t = 20$ msec is the time step for our experiment. In the above, $\mathbf{v}_{\text{ref}}$ is the *refrence velocity*. In this paper, it is the velocity that would bring the robot as close as possible to the target without considering any obstacles. Then, at each time step, the algorithm computes equation (4) for each obstacle. Next, it should determine the relative distance between the robot and each obstacle $D_{\min}(i)$ at related $t_{\min}$ (i). Then, algorithm checks collision condition with each obstacle based on equation (3) to generate admissible velocity $\mathbf{v}$ for the robot to reach the goal. Subsequently, the time horizon t_h should be calculated for all obstacles which satisfy equation (5). When the collision condition returns true, the algorithm according to line 10 determines $\mathcal{VO}^{t_h}_{AB}$ for all obstacles that collide with the robot. Then, the set of *Attainable Velocity Increments* ($\mathcal{AVI}$) should be computed. This set contains all of avoidance maneuvers, $\mathbf{v}'$, that are dynamically feasible from a given state. $\mathcal{AVI}(t + \Delta t)$, over the time step Δt, is defined as (Figure 11):

$$\mathcal{AVI}(t + \Delta t) = \{\mathrm{v}' \,|\, \|\mathrm{v}' - \mathrm{v}_A\| \le a_{\max} \cdot \Delta t\}. \tag{20}$$

Algorithm 1. Compute best admissible velocity
```
 1: let $\mathbf{v}_d = a_{\max} \cdot \Delta t \mathrm{v}_{\text{ref}} / \|\mathrm{v}_{\text{ref}}\|$
 2: for all obstacle $B_i$ do
 3:     $\mathrm{v}_A \leftarrow \mathrm{v}_d$
 4:     $t_{\min}(i) \leftarrow -\mathrm{P}_{AB} \cdot \mathbf{v}_{AB} / \|\mathbf{v}_{AB}\|^2$
 5:     $D_{\min}(i) \leftarrow \|A(t_{\min}, \mathbf{v}_A) - Bi(t_{\min})\|$
 6:     if $D_{\min}(i) < r_{AB_i}$ then
 7:         $t_s(i) \leftarrow \|\mathbf{v}_n\| / -2a_n$        $\triangleright \|a_n\| = 2\, a_{\max}$
 8:
            $t_p(i) \leftarrow$ positive $\min\left(-\|\mathbf{v}_t\| \pm \sqrt{\|\mathbf{v}_t\| + 2a_t d_t} / a_t\right)$
                                                              $\triangleright \|a_t\| = a_{\max}$
 9:         $t_h(i) \leftarrow \min\{t_s(i), t_p(i)\}$
10:         $\mathcal{TCS} \leftarrow \bigcup_{t_0 < t \le t_h} \mathrm{Disk}\left(-\mathrm{P}_{AB_i}/t \oplus \mathbf{v}_B r_{AB_i}/t\right)$
            $\triangleright \oplus$ is the Minkowski vector sum operator
11:         $\mathcal{AVI} \leftarrow \mathbf{v}_A \oplus a_{\max} \cdot \Delta t$
12:         $\mathcal{FS}(i) \leftarrow \mathcal{AVI} \backslash \mathcal{ICS}$
13:         $\mathbf{v}_d \leftarrow \arg\min_{\mathbf{v}* \in \mathcal{FS}(i)} \|\mathbf{v}* - \mathbf{v}_{\text{ref}}\|$
14:     else
15:         $\mathbf{v}_d \leftarrow \operatorname{argmin}_{\mathbf{v}* \in \mathcal{AVI}} \|\mathbf{v}* - V_{\text{ref}}\|$
16:     end if
17:     $\dot{\phi}_d \leftarrow 42.5 - 42.5 / 3.5 \|\mathbf{v}_d\|$
18: end for
19: return $\mathbf{v}_d$ and $\dot{\phi}_d$
```
Then, the algorithm computes the *Feasible States* ($\mathcal{FS}$). The $\mathcal{FS}$ set is the relative complement of $\mathcal{ICS}$ in $\mathcal{AVI}$. This set is defined as follows:

$$\mathcal{FS} = \{\mathcal{ICS}\}^c \cap \mathcal{AVI}. \tag{21}$$

Formally:

$$\mathcal{FS} = \mathcal{AVI} \backslash \mathcal{ICS} = \{\mathbf{v}' \in \mathcal{AVI} \,|\, \mathbf{v}' \notin \mathcal{ICS}\}. \tag{22}$$

After determining the $\mathcal{FS}$, one can preferably examine this set to find a proper point so as a near-optimal motion is produced. In our study, this point is selected such that the shortest path to the goal is generated. Toward this goal, the next velocity is chosen as follows:

Figure 11 The attainable velocity increments (AVI) of the agent shown as the gridded circle. The points inside feasible states ($\mathcal{FS}$), marked green, and points inside ICS, marked red, are inadmissible. The best point, marked blue, is then explored in the next time step. This is repeated until reaching the target

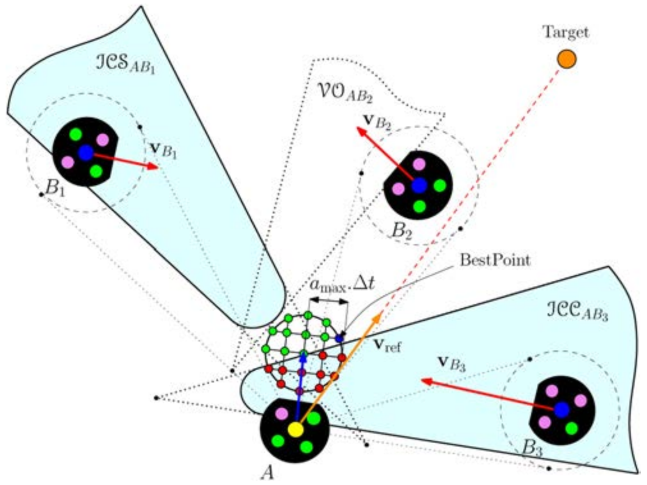
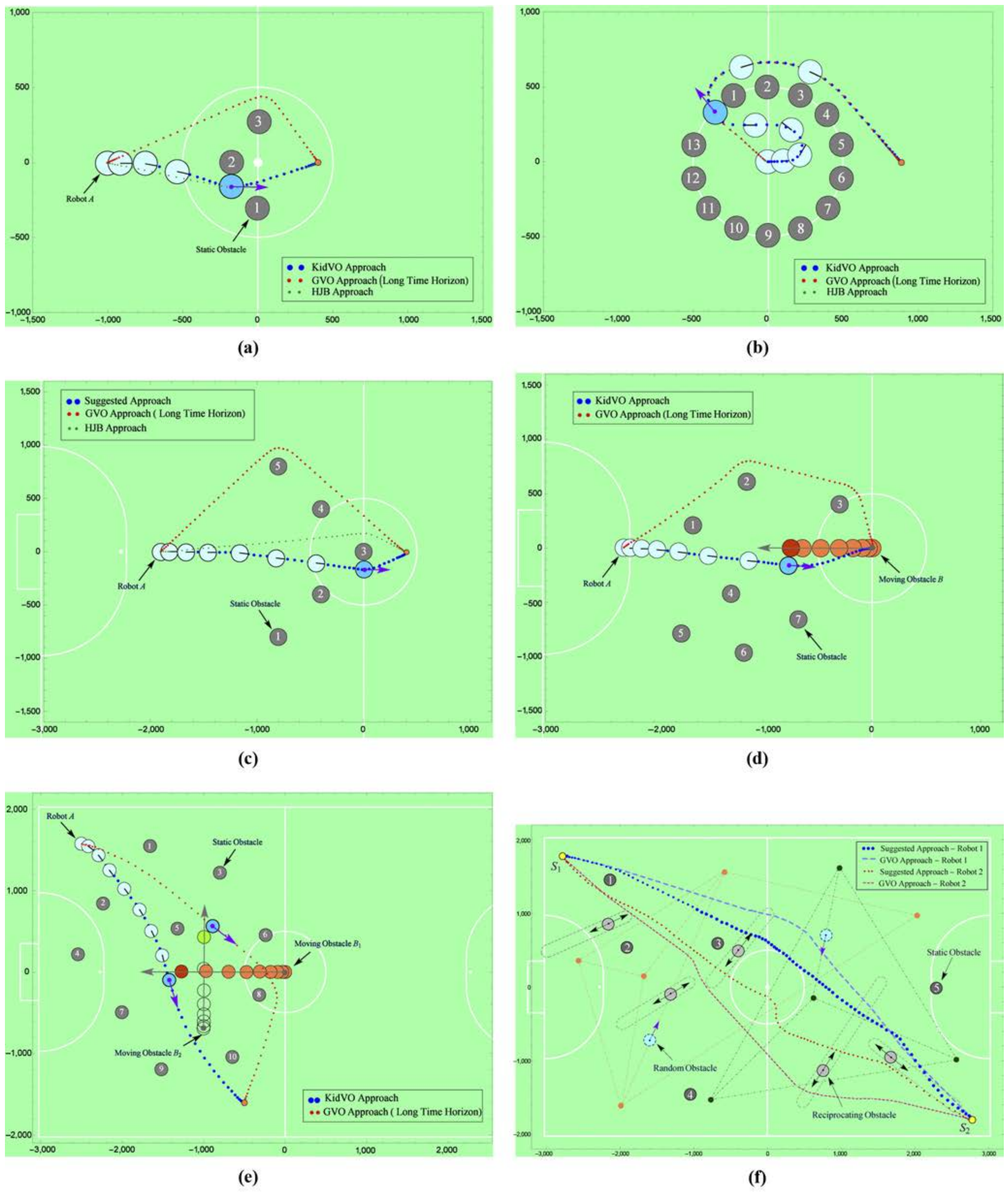


Table II Performance of the KidVO motion planner

	KidVO approach		GVO approach (long time horizon)		HJB approach	
Scenario no.	**Time (sec)**	**Length (m)**	**Time (sec)**	**Length (m)**	**Time (sec)**	**Length (m)**
1	1.20	1.45	1.44	1.69	1.08	1.43
2	2.42	2.94	2.03	2.45	1.97	2.39
3	2.04	2.47	2.57	3.11	1.98	2.44
4	1.91	2.31	2.49	3.01	–	–
5	3.17	3.83	3.72	4.49	–	–
6 (Robot 1)	8.73	6.88	8.91	7.76	–	–
6 (Robot 2)	10.12	7.31	11.02	8.25	–	–

Figure 12 Computer simulation results



Notes: (a) Simulated scenario-1 (b) simulated scenario-2; (c) simulated scenario-3; (d) simulated scenario-4; (e) simulated scenario-5; (f) simulated scenario-6

$$\mathrm{v} = \underset{v^{*}\in FS}{\arg\min}\ \|\mathrm{v}^{\star} - \mathrm{v}_{\mathrm{ref}}\| \qquad (23)$$

As seen form the above relation, the velocity is being planned such that the minimum angle between the next step velocity and $\mathbf{v}_{ref}$ is resulted. The algorithm returns output of equation (23) as desired velocity, $\mathbf{v}_d$, to be explored to the next time step, and this is repeated until reaching the goal. From line 17 of Algorithm 1, it can be inferred that by using velocity filtering equation, equation (15), the robot heading angle can simply be computed. This means in each time step, the suggested planner can compute not only the best admissible linear velocity but also the desired orientation of the robot. During the experiment, the y_b-axis of the robot (Figure 7) should always be tangent to the trajectory (for dribbling and ball handling maneuvers). Therefore, at each time step, to make the angle between $\mathbf{v}_d$ and the robot y_b-axis equal to zero, the command $\dot{\phi}_d$ is produced by the planner.

5. Implementation and results

The present section is divided into two parts, each of which presents the results relating to our research contributions. To demonstrate the performance and effectiveness of the KidVO approach for the omnimobile robots, some practical scenarios have been simulated in the first part. The second part moves on to evaluate the simulation results using real-world experimental tests. It is worth noting that all the computer simulation results are taken on an Intel core i5 CPU at 2.5 GHz with 4 GB RAM. Also, in the practical implementations, as described earlier in Figure 2, the robot's actions are computed on off-field PC with Intel core i7 CPU at 2.4 GHz and 4 GB RAM.

Notice that the used robot software is based on c# programming language. It was implemented and compiled on visual studio 2010 which is directly combined with a visualizer that lets users to run their codes on the robots. Furthermore, it is capable to visualize actual data from a vision system and a simulator by which the desired codes can be executed on a simulated physics-based environment (Poudeh *et al.*, 2014).

5.1 Computer simulations

The response of the suggested motion planner has been investigated through devising several simulations where the robot is commanded to move toward a target in a crowded environment including static and/or dynamic obstacles. The results of this investigation have been reported in Table II. It is pointed out that the considered scenarios contain several cases, where in the first three cases, all obstacles are stationary. Also, the rest of cases involve both static and dynamic obstacles.

5.1.1 Static environment

The first scenario indicates the robot motion while avoiding three immobile obstacles. As illustrated in Figure 12(a), the robot starts its motion from a point near to (−1000, 0) at zero speed. It should move toward a goal point located at (400, 0) which is marked by an orange-colored disk. Note that, all dimensions are expressed in millimeters. The obstacle avoidance maneuver for the suggested approach and that of GVO has been represented by blue- and red-dotted trajectory, respectively. The green-dotted trajectory indicates the shortest path which is generated by Hamilton-Jacobi-Bellman (HJB) equation (Sundar and Shiller, 1994). The dots are spaced at constant time intervals, thus indicating the acceleration and deceleration of the robot along the path. Now, a known case study of planning problem is being considered, as shown in Figure 12(b), where getting stuck in local minima is completely probable (in the sense of algorithms like APF-based techniques).

Figure 13 Performance and scalability of the KidVO planner

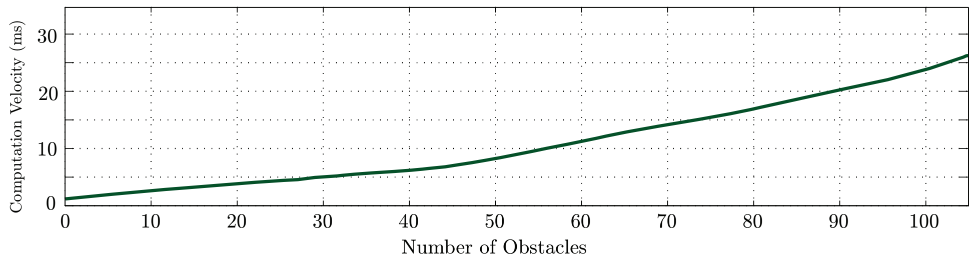


Figure 14 The real-world experimentation: performance of the KidVO motion planner

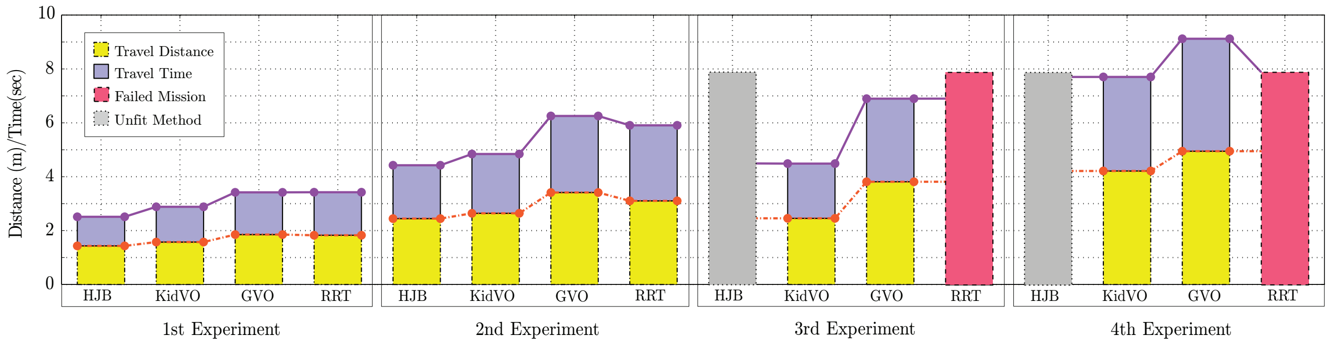
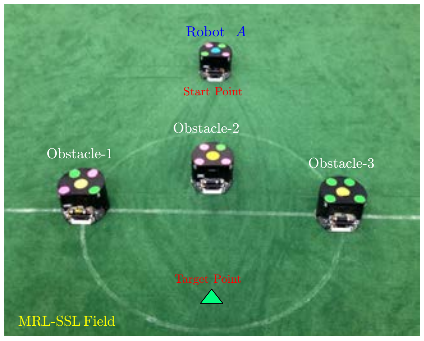
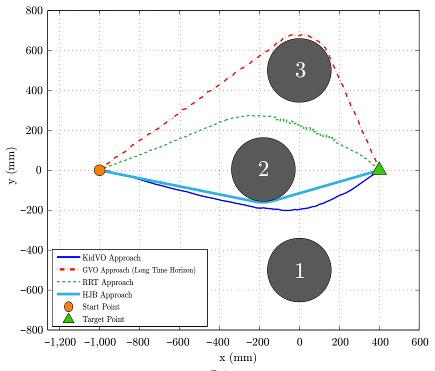


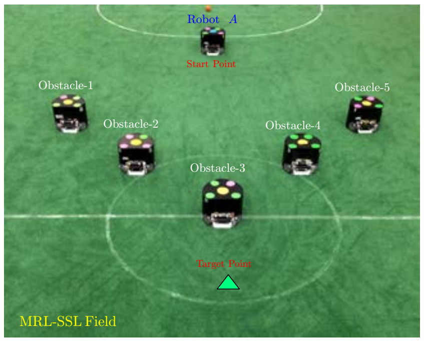
Figure 15 Implementation results in the first four real-world scenarios



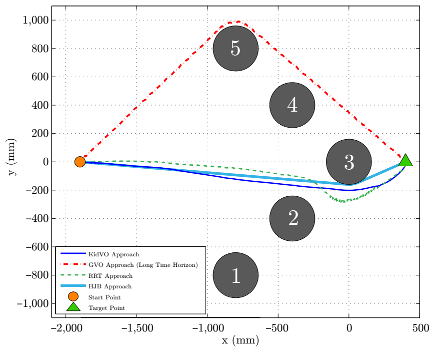
(a)



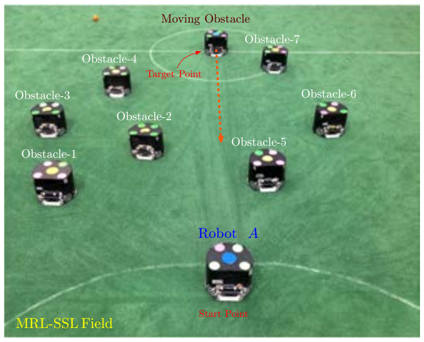
(b)



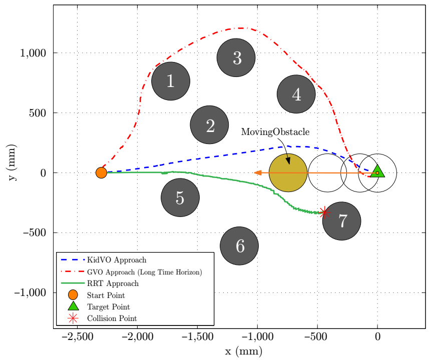
(c)



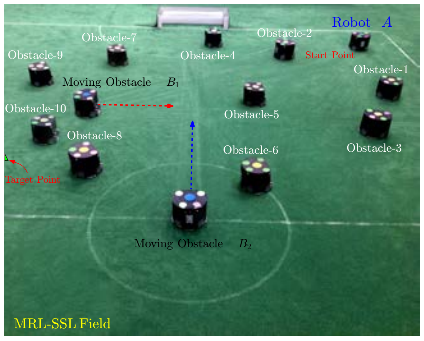
(d)



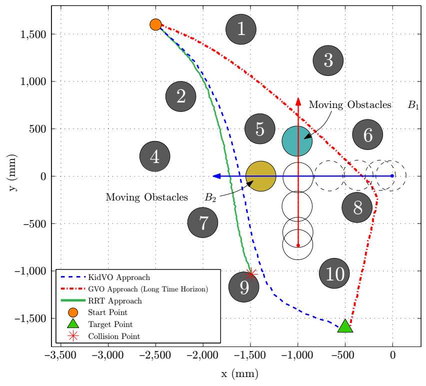
(e)



(f)



(g)



(h)

Notes: (a) The 1st Experiment; (b) results of the 1st Experiment; (c) the 2nd Experiment; (d) results of the 2nd Experiment; (e) the 3rd Experiment; (f) results of the 3rd Experimen; (g) the 4th Experiment; (h) results of the 4th Experiment

In this case, the robot initiates its travel from (0, 0) and tries to get to the orange-colored target point which is located out of circular space formed by obstacles. As seen, the suggested planner can successfully guide the robot to the goal position and avoid this potential local minimum condition. The corresponding time/distance performance of the algorithms has been reflected in Table II. In the third scenario, as illustrated in Figure 12(c), the robot starts at (−1900, 0), attempting to move toward the target at (400, 0). The travel distance along the red-dotted trajectory (GVO approach) was about 25 per cent longer than the corresponding one of the blue-dotted trajectory (KidVO approach). According to Table II, the travel distances and the travel times along the generated trajectories by KidVO planner and optimal solution are quite similar for the first two simulations.

5.1.2 Dynamic environment

The first scenario was implemented with seven stationary and one moving obstacles. As it can be observed from Figure 12(d), the robot starts at (−2300, 0) and should move to the goal point which is located at (0, 0). The moving obstacle starting from the rest and moving along a horizontal line with an upper speed equals ±1800 mm/sec. The obtained simulation results again indicate the merits of the KidVO planning method in terms of 30 per cent decreased travel distance as compared with the GVO method with consideration of long time horizon. The next scenario has been implemented with the presence of ten immobile and two moving obstacles. According to Figure 12(e), the robot starts from the top left at point (2500, 1600) and moves to the target point at (−500, −1600). One of the moving obstacles starts from (0, 0) at zero speed and traversing horizontally and the other one starts from rest at point (−1000, −700) and moving vertically. The upper limit of the moving obstacles' speeds is the same as the previous scenario. The time that takes robot to reach the goal using the suggested method is about 17 per cent less than the GVO method. The latest case of this part is a complicated one where there are 12 obstacles in the environment and two robots should exchange their locations, i.e. S_1 and S_2, as depicted in Figure 12(f). It is pointed out that among 12 obstacles, five ones are moving along specified straight line reciprocally. Besides, there are two robots moving randomly along paths depicted by orange- and green-colored roadmaps. Also, the remaining five obstacles are stationary. The results of GVO as well as those of KidVo can be observed in Table II. As seen, while the results are close, the performance of the suggested method is superior. Notice that

Figure 16 The real-world complicated scenario

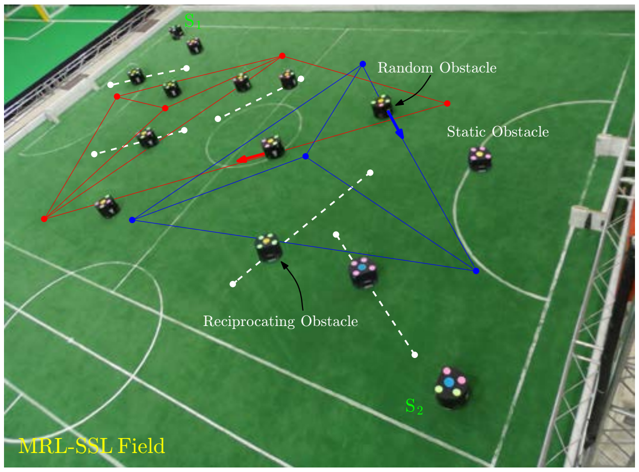
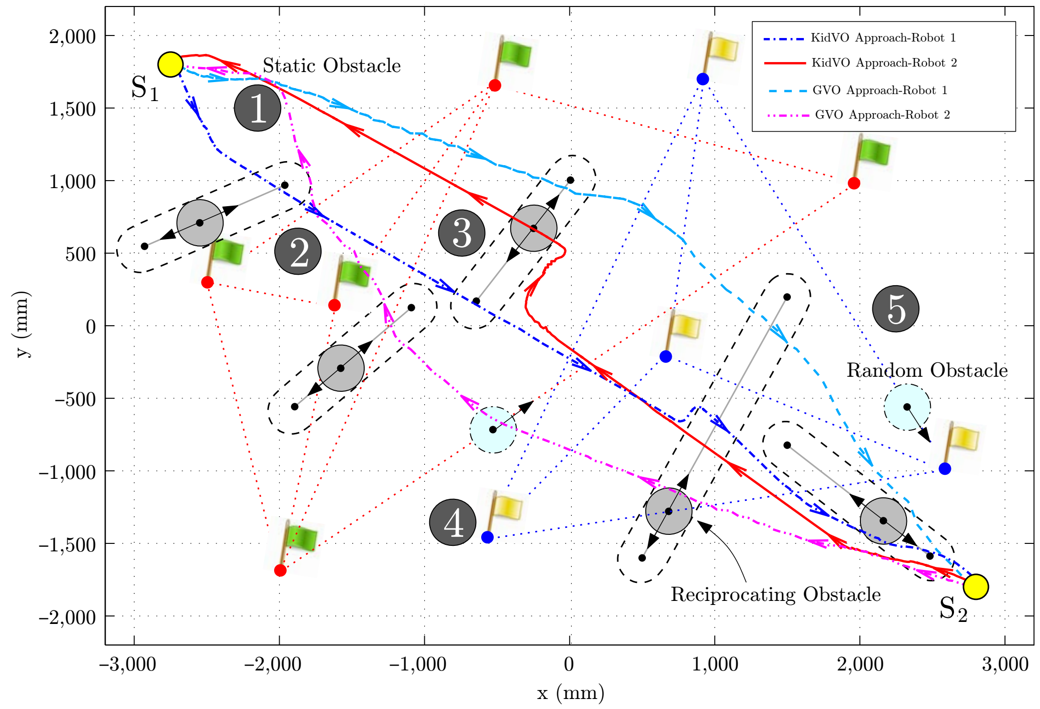


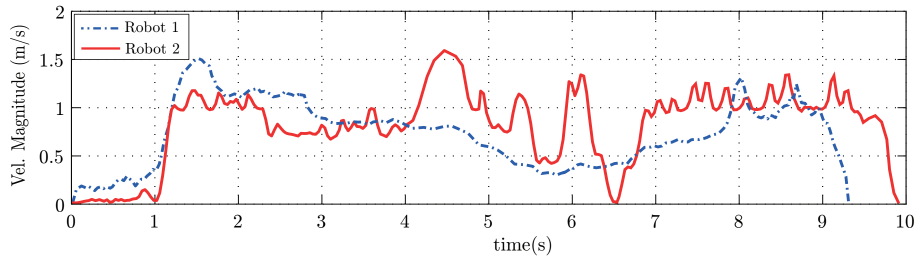
Figure 17 The real-world experimentation results in a complex dynamic environment



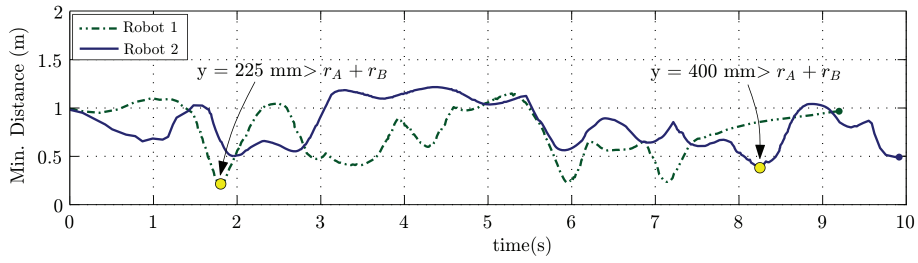
although the results of the current case have just been reported once in Table II, the performances of these two algorithms have been compared for several times using the final scenario, and the whole obtained results showed that the suggested method has better response. Note that in the last case study, as mentioned earlier, the two obstacles are moving randomly along colored roadmaps, and the start point of the reciprocating obstacles are also random which returns a distinct environment for each execution. It is worth mentioning that during the whole scenarios, two used algorithms were implemented in parallel using multiple cores.

To examine the scalability of the KidVO planner versus the number of obstacles, the last scenario, Figure 12(f), has been repeated using different number of obstacles. The achieved results have been shown in Figure 13. As seen, the vertical axis indicates the time required to compute a new velocity for each robot.

Figure 18 (a) The robots velocity profiles; (b) the robots minimum distances from obstacles



(a)



(b)

Figure 19 The real-world experimentation results in a complex dynamic environment

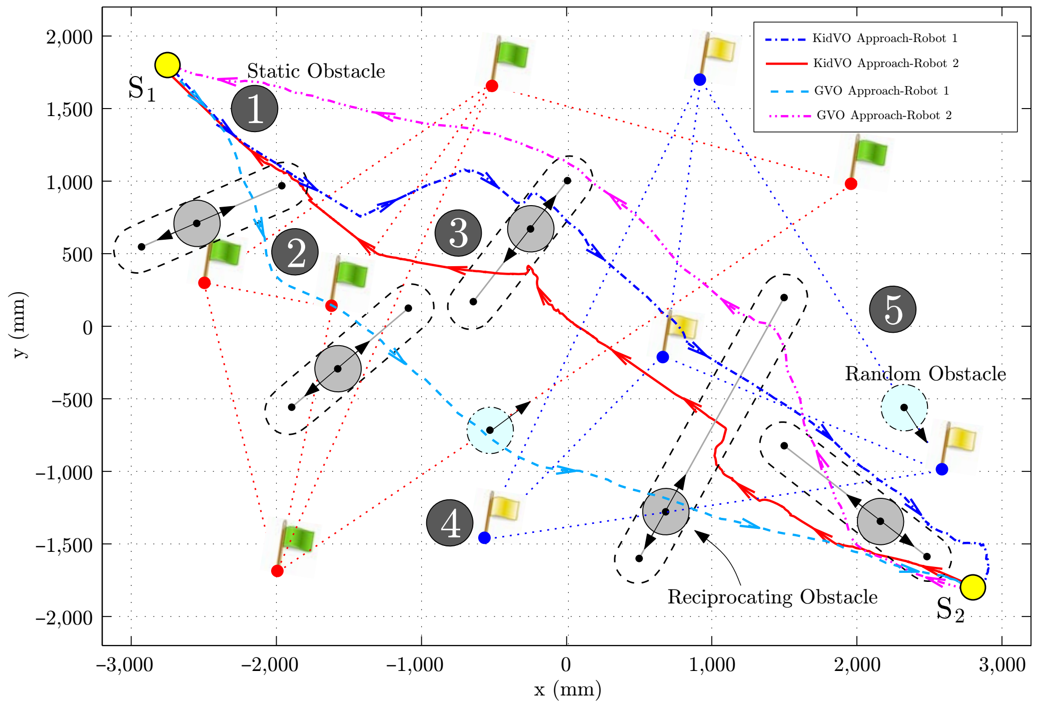
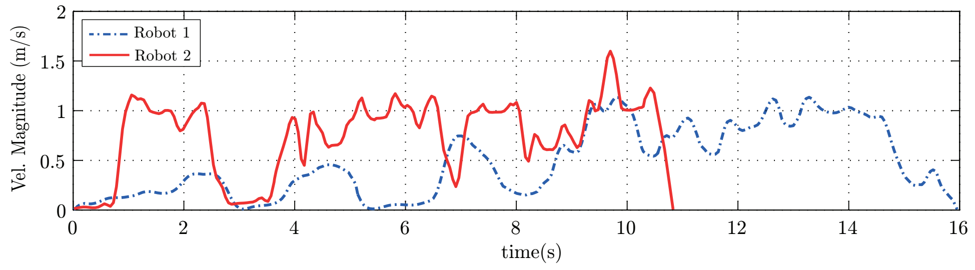
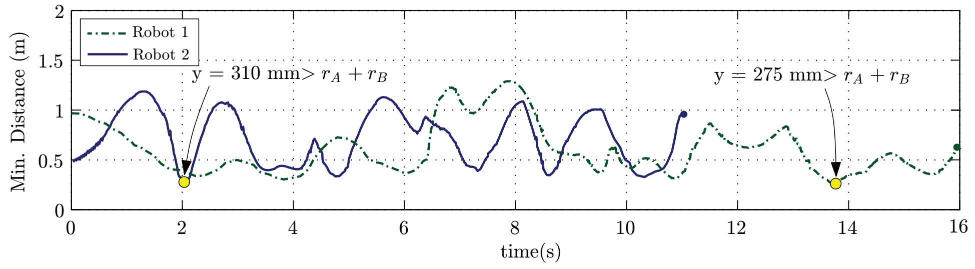


Figure 20 (a) The robots velocity profiles; (b) the robots minimum distances from the obstacles



(a)



(b)

5.2 Practical implementation

This part, similar to the previous one, examines the response of various planning methods through real-world experiments. To this end, the KidVO motion planning strategy was tested on the MRL-SSL RoboCup Robot, considering the scenarios Figure 12(a) and 12(c)-12(f) of the previous subsection, as depicted in Figure 12. It is noteworthy that, all configuration settings and initialization parameters in these experiments are the same as the computer simulations discussed earlier. Some of these include: number of stationary/moving obstacles, upper limit speed of the robot and moving obstacles and KDCs of the robot, etc. The results of our technique in comparison with the results of RRT (as a global planner) and HJB approaches under different environmental conditions indicate the good performance of our suggested idea.

Figure 14 provides comparison of KidVO and other mentioned approaches for four different experimental results. In the first two experiments, as illustrated in Figure 15(a to d), the travel distances and the travel times along the generated trajectories by proposed planner and HJB method are quite similar, which again confirms the merits of the suggested planning method. The real-world experimental results of application of proposed method and GVO have been shown in Figures 16-20 for two cases of the last complicated scenario (see Figure 16), i.e. the scenario corresponding to Figure 12(f). To better reflect the behavior of the suggested method, the velocity profile of robots has been demonstrated in Figure 18(a) and Figure 20(a) for two distinct cases. In Figure 20(b) and Figure 20(b), the minimum distances of robots relative to all obstacles have been drawn to illustrate the utilized core concept for successful obstacle avoidance for the two considered cases.

6. Conclusions and future works

An online motion planner for crowded dynamic environments was presented which improves GVO notion into two ways. First, it adds a systematic approach to tune the time horizon. Second, added value of the suggested method is considering the admissible velocity and its increment during the process of velocity planning. Besides, most of presented velocity-based planning algorithms demonstrate their results based on only computer simulations. However, in contrast to the previously proposed methods, the merits of the suggested KidVO method have been shown not only by computer simulations but also by real-world experiments imitating the situation of complex dynamic environments.

The future works include the addition of Model Predictive Controller to enable the robot to choose the optimal time horizon and to affect the future events during planning. In addition, reaching the goal within the specified time and robustness issues will be our next research directions.

References

Aivar, M., Brenner, E. and Smeets, J. (2008), "Avoiding moving obstacles", *Experimental Brain Research*, Vol. 190 No. 3, pp. 251-264.

Behnisch, M., Haschke, R. and Gienger, M. (2010), "Task space motion planning using reactive control", *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Taipei*, pp. 5934-5940.

Berg, J.P.V.D. and Overmars, M.H. (2005), "Roadmap-based motion planning in dynamic environments", *IEEE Transactions on Robotics*, Vol. 21 No. 5, pp. 885-897.

Berg, J.P.V.D. and Overmars, M.H. (2007), "Kinodynamic motion planning on roadmaps in dynamic environments", *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, San Diego, CA*, pp. 4253-4258.

Berg, J.P.V.D., Snape, J., Guy, S.J. and Manocha, D. (2011), "Reciprocal collision avoidance with acceleration-velocity obstacles", *2011 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Shanghai*, pp. 3475-3482.

Borenstein, J. and Koren, Y. (1991), "The vector field histogram-fast obstacle avoidance for mobile robots", *IEEE Transactions on Robotics and Automation*, Vol. 7 No. 3, pp. 278-288.

Brock, O. and Khatib, O. (1999), "High-speed navigation using the global dynamic window approach", *1999 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Detroit, MI*, Vol. 1, pp. 341-346.

Canny, J.F. and Lin, M.C. (1993), "An opportunistic global path planner", *Algorithmica*, Vol. 10 Nos 2/4, pp. 102-120.

Cherubini, A., Grechanichenko, B., Spindler, F. and Chaumette, F. (2013), "Avoiding moving obstacles during visual navigation", *2013 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Karlsruhe*, pp. 3069-3074.

Choset, H.M. (2005), *Principles of Robot Motion: Theory, Algorithms, and Implementation*, MIT press, Cambridge, MA.

Fiorini, P. and Shiller, Z. (1998), "Motion planning in dynamic environments using velocity obstacles", *The International Journal of Robotics Research*, Vol. 17 No. 7, pp. 760-772.

Fox, D., Burgard, W. and Thrun, S. (1997), "The dynamic window approach to collision avoidance", *IEEE Robotics and Automation Magazine*, Vol. 4 No. 1, pp. 23-33.

Fulgenzi, C., Spalanzani, A. and Laugier, C. (2009), "Probabilistic motion planning among moving obstacles following typical motion patterns", *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, St. Louis, MO*, pp. 4027-4033.

Gal, O., Shiller, Z. and Rimon, E. (2009), "Efficient and safe on-line motion planning in dynamic environments", *2009 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Kobe*, pp. 88-93.

Halmos, P. (1960), *Naive Set Theory*, Van Nostrand, reprinted by Springer-Verlag, Undergraduate Texts in Mathematics, 1974.

Hsu, D., Kindel, R., claude Latombe, J. and Rock, S.M. (2002), "Randomized kinodynamic motion planning with moving obstacles", *The International Journal of Robotic Research*, Vol. 21 No. 3, pp. 233-256.

Jing, X.-J. (2005), "Behavior dynamics based motion planning of mobile robots in uncertain dynamic environments", *Robotics and Autonomous Systems*, Vol. 53 No. 2, pp. 99-123.

Kant, K. and Zucker, S.W. (1986), "Toward efficient trajectory planning: the path-velocity decomposition", *The International Journal of Robotic Research*, Vol. 5 No. 3, pp. 72-89.

Khatib, O. (1985), "Real-time obstacle avoidance for manipulators and mobile robots", *1985 IEEE International Conference on Robotics and Automation (ICRA), IEEE, St. Louis, MO*, Vol. 2, pp. 500-505.

Kimmel, R., Kiryati, N. and Bruckstein, A. (1998), "Multivalued distance maps for motion planning on surfaces with moving obstacles", *IEEE Transactions on Robotics and Automation*, Vol. 14 No. 3, pp. 427-436.

Koenig, S. and Likhachev, M. (2002), "D* lite", *Eighteenth National Conference on Artificial Intelligence, American Association for Artificial Intelligence, Menlo Park, CA*, pp. 476-483.

Kuffner, J. and LaValle, S. (2000), "RRT-connect: an efficient approach to single-query path planning", *2000 IEEE International Conference on Robotics and Automation (ICRA), IEEE, San Francisco, CA*, Vol. 2, pp. 995-1001.

Kuwata, Y., Karaman, S., Teo, J., Frazzoli, E., How, J. and Fiore, G. (2009), "Real-time motion planning with applications to autonomous urban driving", *IEEE Transactions on Control Systems Technology*, Vol. 17 No. 5, pp. 1105-1118.

Li, C., Li, H., Watanabe, T., Huangfu, Y. and Wu, Z. (2010), "The real-time and embedded soccer robot control system", in Papić, V. (Ed.), *Robot Soccer*, INTECH Open Access Publisher, pp. 1-18.

Mahmoodi, M., Alipour, K., Tale Masouleh, M. and Beik Mohammadi, H. (2014), "Real-time safe navigation in crowded dynamic environments using Generalized Velocity Obstacles", *ASME 2014 International Design Engineering Technical Conferences and Computer and Information in Engineering Conference, Buffalo, New York*, American Society of Mechanical Engineers, Vol. 5B, pp. V05BT08A060-V05BT08A060.

Minguez, J. and Montano, L. (2000), "Nearness diagram navigation (ND): a new real time collision avoidance approach", *2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Takamatsu*, Vol. 3, pp. 2094-2100.

Nilsson, N.J. (1982), "Principles of Artificial Intelligence", Springer, available at: www.springer.com/la/book/9783540113409 (accessed 2 September 2015).

Petti, S. and Fraichard, T. (2005), "Safe motion planning in dynamic environments", *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Edmonton*, pp. 2210-2215.

Poudeh, A.G., Esmaeelpourfard, S., HosseiniKia, A., Tafti, H.J. and Adhami-Mirhosseini, A. (2014), "MRL extended team description 2014", *Proceedings of the 18th International RoboCup Symposium*, available at: http://robocupssl.cpe.ku.ac.th/_media/robocup2014:etdp:mrl_2014_etdp.pdf.

Purwin, O. and D'Andrea, R. (2006), "Trajectory generation and control for four wheeled omnidirectional vehicles", *Robotics and Autonomous Systems*, Vol. 54 No. 1, pp. 13-22.

Shiller, Z., Gal, O. and Raz, A. (2011), "Adaptive time horizon for on-line avoidance in dynamic environments", *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, San Francisco, CA*, pp. 3539-3544.

Simmons, R. (1996), "The curvature-velocity method for local obstacle avoidance", *1996 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Minneapolis, MN*, Vol. 4, pp. 3375-3382.

Sundar, S. and Shiller, Z. (1994), "Optimal obstacle avoidance based on the hamilton-jacobi-bellman equation", *1994 IEEE International Conference on Robotics and Automation (ICRA), IEEE*, Vol. 3, pp. 2424-2429.

Toit, N.E.D. and Burdick, J.W. (2012), "Robot motion planning in dynamic, uncertain environments", *IEEE Transactions on Robotics*, Vol. 28 No. 1, pp. 101-115.

Vannoy, J. and Xiao, J. (2008), "Real-time adaptive motion planning (RAMP) of mobile manipulators in dynamic environments with unforeseen changes", *IEEE Transactions on Robotics*, Vol. 24 No. 5, pp. 1199-1212.

Wang, D. and Qi, F. (2001), "Trajectory planning for a four-wheel-steering vehicle", *2001 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Seoul*, Vol. 4, pp. 3320-3325.

Wilkie, D., van den Berg, J. and Manocha, D. (2009), "Generalized velocity obstacles", *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, St. Louis, MO*, pp. 5573-5578.

Wu, J. (2005), "Dynamic path planning of an Omni-directional robot in a dynamic environment", PhD Thesis, Ohio University, Athens, OH.

Xu, B., Stilwell, D.J. and Kurdila, A.J. (2010), "A receding horizon controller for motion planning in the presence of moving obstacles", *2010 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Anchorage, AK*, pp. 974-980.

Zucker, M., Kuffner, J.J. and Branicky, M.S. (2007), "Multipartite RRTs for rapid replanning in dynamic environments", *2007 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Roma*, pp. 1603-1609.

Appendix 1. The robot acceleration space equations

In equations (A1-A4), $s\theta_i = \sin\theta_i$; $s2\theta_i = \sin 2\theta_i$; $s\theta_{12} = \sin(\theta_1 + \theta_2)$; $s\theta_{21} = \sin(\theta_1 - \theta_2)$ and, $c\theta_i = \cos\theta_i$; $c2\theta_i = \cos 2\theta_i$; $c\theta_{12} = \cos(\theta_1 + \theta_2)$; $c\theta_{21} = \cos(\theta_1 - \theta_2)$.

$$\begin{aligned}\xi_1(u_i) = {} & h\mu\big[u_4(h\mu s2\theta_2 u_3 - 4l(c2\theta_2 + c\theta_{12})) + u_2(h\mu s\theta_{12} u_3 \\ & - 4l(c2\theta_1 + c\theta_{12})) + 4lu_3(c2\theta_2 + c\theta_{12}) + u_1(4l(c\theta_{12} \\ & + s2\theta_1) + h\mu s2\theta_1 u_2 + h\mu s\theta_{12} u_4)\big] + 16l^2(s2\theta_1 \\ & + s2\theta_2 + 2s\theta_{12}).\end{aligned} \tag{A1}$$

$$\begin{aligned}\psi_1(u_i) = {} & 2(c\theta_1 + c\theta_2)\big[8ls\theta_1 s\theta_2(u_3 + u_4) - u_2(8ls\theta_1 s\theta_2 \\ & + h\mu s\theta_{12} u_3 - h\mu s\theta_{21} u_4) - u_1(8ls\theta_1 s\theta_2 h\mu s\theta_{21} u_3 \\ & - h\mu s\theta_{12} u_4)\big]\end{aligned} \tag{A2}$$

$$\begin{aligned}\psi_2(u_i) = {} & 2u_2\big[-8lc\theta_1(c\theta_1 + c\theta_2)s\theta_2 + h\mu(c\theta_1 - c\theta_2)((1 \\ & + c\theta_{12})u_3 - (1 + c\theta_{21})u_4)\big] + 2u_1\big[8lc\theta_1(c\theta_1 \\ & + c\theta_2)s\theta_2 + 2h\mu s2\theta_1 s\theta_2 u_2 + h\mu(c\theta_1 - c\theta_2)(-(1 \\ & + c\theta_{21})u_3 + (1 + c\theta_{12})u_4)\big] + 8c\theta_2 s\theta_1\big[2l(c\theta_1 \\ & + c\theta_2)u_4 - u_3(2l(c\theta_1 + c\theta_2) + h\mu s\theta_2 u_4)\big].\end{aligned} \tag{A3}$$

$$\begin{aligned}\psi_3(u_i) = {} & \frac{m}{\mathcal{J}}\big[16l^2(c\theta_1 + c\theta_2)s\theta_1(u_3 + u_4) \\ & + u_2\big[4l(c\theta_1 + c\theta_2)(4ls\theta_2 + h\mu(-c\theta_1 + c\theta_2)u_3) \\ & + h\mu(-2l(c2\theta_1 + c2\theta_2 + 2c\theta_{12}) + h\mu c\theta_2(s\theta_1 \\ & + s\theta_2)u_3)u_4\big] + u_1\big[16l^2(c\theta_1 + c\theta_2)s\theta_2 \\ & + h\mu(s\theta_1 + s\theta_2)(-4ls\theta_1 + 4ls\theta_2 + h\mu c\theta_1 u_2)u_4 \\ & + h\mu u_3\big[2l(c2\theta_1 + c2\theta_2 + 2c\theta_{12}) + h\mu(s\theta_1 \\ & + s\theta_2)\,(c\theta_1 u_2 + c\theta_2 u_4)\big]\big]\big].\end{aligned} \tag{A4}$$

Corresponding author

Khalil Alipour can be contacted at: k.alipour@ut.ac.ir