Collision Avoidance of a Kinodynamically Constrained System from Passive Agents

Khalil Muhammad Zuhaib
Department of Electronics Engineering
Quaid-e-Awam University of Engineering, Science & Technology, Larkana Campus, Pakistan
kmzuhaib@quest.edu.pk

Junaid Iqbal
Department of Mechanical Engineering
Quaid-e-Awam University of Engineering, Science & Technology, Larkana Campus, Pakistan
ji.bhatti@quest.edu.pk

Ahsin Murtaza Bughio
Department of Electronics Engineering
Quaid-e-Awam University of Engineering, Science & Technology, Nawabshah, Pakistan
ahsan.murtaza@quest.edu.pk

Syed Abid Ali Shah Bukhari
Department of Electrical Engineering
Quaid-e-Awam University of Engineering, Science & Technology, Larkana Campus, Pakistan
abidshah@quest.edu.pk

Kelash Kanwar
Department of Electronics Engineering
Quaid-e-Awam University of Engineering, Science & Technology, Nawabshah, Pakistan
kelashkanwar@quest.edu.pk

Abstract-Robot motion planning in dynamic environments is significantly difficult, especially when the future trajectories of dynamic obstacles are only predictable over a short time interval and can change frequently. Moreover, a robot's kinodynamic constraints make the task more challenging. This paper proposes a novel collision avoidance scheme for navigating a kinodynamically constrained robot among multiple passive agents with partially predictable behavior. For this purpose, this paper presents a new approach that maps collision avoidance and kinodynamic constraints on robot motion as geometrical bounds of its control space. This was achieved by extending the concept of nonlinear velocity obstacles to incorporate the robot's kinodynamic constraints. The proposed concept of bounded control space was used to design a collision avoidance strategy for a car-like robot by employing a predict-plan-act framework. The results of simulated experiments demonstrate the effectiveness of the proposed algorithm when compared to existing velocity obstacle based approaches.

Keywords-collision avoidance; dynamic environment; motion planning; navigation; mobile robot; kinodynamic constraint; velocity obstacle; pedestrian environment

I. INTRODUCTION

Most modern applications of mobile robots require them to navigate in a dynamic environment among moving obstacles such as people, pets, cars, etc. Robots working in such environments must be able to successfully navigate while multiple agents are moving around them. To achieve this objective, mobile robots need to generate a short collision-free path in real-time concerning an obstacle's future position. Furthermore, the generated path should be system compatible, as most of the mobile robots have kinodynamic constraints, such as car-like robots [1, 2].

Many motion planning approaches, such as in [3-5], were proposed to navigate a robot with kinodynamic constraints in a static environment, and extended their application in dynamic environments [6-8]. Some important works in the direction of robot navigation in a dynamic environment were introduced in [9-11], focusing on a complete trajectory plan to the goal. These approaches don't perform well in a dynamic environment, especially when it is possible to predict only partially the future motion of other agents, e.g. pedestrian environment, for many reasons. The first reason is that these approaches don't consider predicting the future motion of other agents, thus the robot becomes blinded to a possible future collision. Secondly, a limited time is available for computing a solution in a dynamic environment. Therefore, it is very likely that the planner will be unable to compute the complete collision-free trajectory during the available time slot. Other works, such as [12, 13], proposed approaches that search in state time-space to compute a system compliant collision-free trajectory to the goal, based on the other agent's future path. These methods assumed that the complete path of the agents is known in advance, limiting their application on pedestrian or multiple passive agents environments. The moving agents have free will in such environments. Their future behavior can only be partially predictable, if at all. Agents' future behavior is predicted through various on-line prediction approaches, such as the ones proposed in [14-17]. In such prediction approaches, the model of the future motion is obtained at an instant and it is valid for a short period. Therefore, a navigation approach should be designed to take into account the duration of the environment model's validity and the limited available time for computing a collision-free solution. An alternative approach to achieve motion planning is to plan a motion locally. In such approaches, the robot employs a continuous predict-act-plan cycle. In each cycle, the robot must compute its action based on predictions to avoid static and moving obstacles as it moves towards the goal. Some of the dominant works in this direction were based on the concept of Velocity Obstacle (VO) [18],

Corresponding author: Khalil Muhammad Zuhaib

which considers the future behaviors for collision avoidance. The VO-based navigation approach consists of planning a local motion toward the next (sub) goal, which is extracted from a global waypoint plan. VO was formulated for a simple robot model, where the obstacle path was assumed to be along a straight line. In [19], the Nonlinear Velocity Obstacle (NLVO) concept was proposed for collision avoidance of a robotic system with a linear equation of motion, where the passive agent was assumed to be moving along an arbitrary possibly nonlinear path. To expand the concept of VO, authors in [20] considered the kinematic car-like robot avoiding collision with multiple passive agents, proposing the concept of Generalized Velocity Obstacle (GVO). This approach principally acts as the opposite of NLVO, where a car-like system with a nonlinear motion equation should avoid collision with obstacles having linear motion equations (straight paths). VO was expanded in [21] proposing the Acceleration Velocity Obstacle (AVO) method for a system with acceleration constraints. This approach was proposed for reciprocal collision avoidance among active agents, where each one shares the responsibility. This approach assumed that every active agent involved in a collision should run a similar collision avoidance algorithm for avoiding it. In [22], the concept of safe control space for collision avoidance was proposed to deal with various sources of un-modeled uncertainties. Some other studies which examined collision avoidance were presented in [23-24].

This study addressed the problem of navigating a car-like robot system under kinodynamic constraints among multiple passive agents with partially predictable behaviors. At first, the idea of NLVO [19] was extended to define a generalized approach for handling collisions of a car-like robot with obstacles moving along arbitrary paths. A formulation was given to represent geometrically the bounds of robot motion as bounds of the robot's control space. The bounded control space was named Valid Control Space. Moreover, this paper provides a framework for finding the optimal control within the bounded control space, which can be used to locally navigate a robot to a goal. The proposed method's performance was evaluated for a dynamic environment similar to [20], where the entire responsibility of collision avoidance belongs to the robot. As shown in Figure 1, passive agents' predicted trajectories can change frequently. Simulations showed that the proposed approach performs better than existing VO based approaches.

II. VALID CONTROL SPACE

This section presents the idea of Valid Control Space, which addresses the difficulty of overcoming the kinodynamic constraints of a car-like robot. This was achieved by introducing a control obstacle concept, which was a generalization of the NLVO.

A. Notations and Assumptions

$S_r \subset \mathbb{R}^n$ denotes the robot's state space, and $s_r(t)$ denotes the state of the robot R_r at time t. The robot workspace is $\mathbb{R}^d$, where typically $d = 2$. The robot position at time t in robot configuration space, denoted by $p_r(t)$, is obtained from the state of the system through a nonlinear projection function:

$$p_r(t) = h(s_r(t)) \quad (1)$$

where $h \in S_r \rightarrow \mathbb{R}^d$. Furthermore, it is assumed that the dimension of workspace is equal to the robot's input control space $U \subset \mathbb{R}^d$. The robot's state transition equation is probably a nonlinear function $f: S_r \times U \rightarrow \mathbb{R}^n$:

$$\dot{s}_r(t) = f(s_r(t), u(t)) \quad (2)$$

where $s_r(t) \in S_r$ is the robot state at time t, and $u(t) \in U$ is the control input. The state at any given time greater than zero for a given current state $s_r = s_r(0)$ and a constant input control $u = u(0)$ is given by:

$$s_r = g(s_r, u, t) \quad (3)$$

where $g: S_r \times U \times \mathbb{R} \rightarrow S_r$ is the solution of (2).

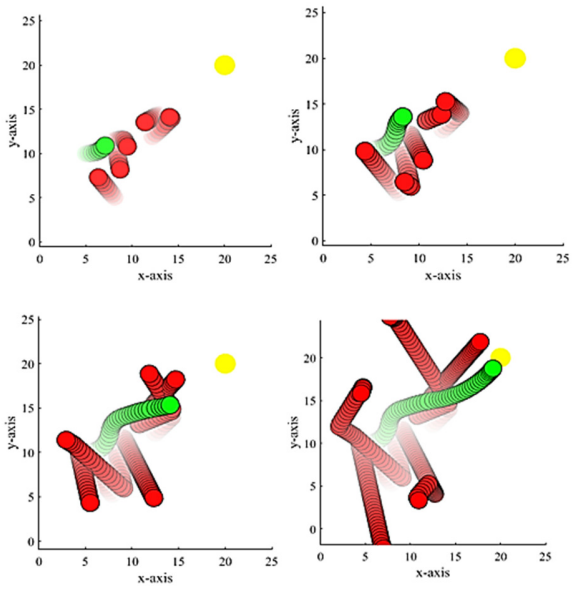


Fig. 1. Multiple simulation snapshots at different time instances. The robot in green starts from its initial position at (5,10) and moves towards the goal marked in yellow while avoiding collision with the multiple passive agents marked in red.

B. Dynamic Obstacle Representation

This study assumed that there will be a system in addition to the mobile robot, as introduced in [16-17], tracking passive agent states and estimating their future trajectories. Each agent has a disc-like shape with a radius and a trajectory. The list of passive agents and their future behaviors is given as the set of points representing time and their estimated future positions.

C. Control Obstacle

The geometry of the robot was considered to be its bounding circle, as in [18]. The mobile robot shares its workspace with several passive agents. A and O_i are sets of points representing the robot and the i-th agent geometry respectively, while B_i is the Minkowski sum of A and O_i geometries $B_i = A \oplus O_i$. At $t = 0$, the position robot's center and the i-th passive agent are represented by $p_r(0)$ and $p_i(0)$ respectively. The predicted position of the i-th passive agent at $t > 0$ is given as $p_i(t)$. The robot will avoid collision with the

i-th passive agent within a time interval $[0,\tau]$ if the relative position vector for input u remains outside B_i.

$$h\big(g(s_r,u,t)\big) - p_i(t) \notin B_i, \forall t \in [0,\tau] \quad (4)$$

The temporary component of control obstacle $UO_i(t)$ induced for the robot due to *i-th* agent at time $t \in [0;\tau]$, can be defined as:

$$UO_i(t) = \{u | h\big(g(s_r,u,t)\big) - p_i(t) \in B_i\} \quad (5)$$

where $UO_i(t)$ is the set of all $u \in U$ for which the relative position vector $h\big(g(s_r,u,t)\big) - p_i(t)$ will be inside B_i at time $t \in [0,\tau]$. Thus, the control obstacle induced for robot over time horizon $[0,\tau]$ due to the *i*-th agent can be defined as:

$$UO_i^\tau = \cup_{0<t\leq\tau} UO_i(t) \quad (6)$$

where UO_i^τ is the set of all $u \in U$ that will maneuver the robot towards a collision with an *i*-th agent at time $t \in [0,\tau]$. The robot will remain collision-free from the *i*-th agent within the time horizon $[0,\tau]$ if the selected robot's control input does not belong in the control obstacle ($u \notin UO_i^\tau$).

This collision avoidance approach can be extended for multiple dynamic agents. $N = \{1,2,...n\}$ is the index of the dynamic agents to avoid. The control obstacle induced for the robot by the dynamic agents can be given as:

$$UO^\tau = \cup_{i\in N} UO_i^\tau \quad (7)$$

A robot selecting $u \notin UO^\tau$ will navigate without collision with *n* dynamic agents for at-least τ seconds.

D. Valid Control Space

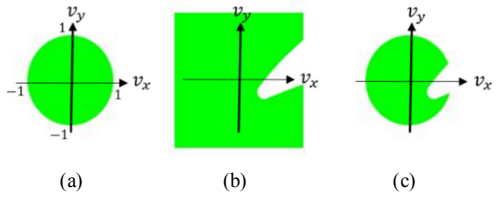
The constraints of robot motion are defined by constraint functions $f_c(s,t)$ bounded in $[M_-;M_+]$, where $(M_-;M_+ \in \mathbb{R})$. These constraints restrict the admissible controls for the robot in a control space over a time horizon $[0,\tau]$. The admissible space of the controls is denoted by U_{ad}. The valid control space at time $t \in [0,\tau]$, denoted by $U_{valid}^\tau(t)$, is:

$$U_{valid}^\tau = U_{ad} \backslash UO^\tau \quad (8)$$

U_{valid}^τ models all possible sub-trajectories satisfying all constraints. Every point chosen in U_{valid}^τ will satisfy the kinodynamic constraints and collision avoidance over $t \in [0,\tau]$. This approach is based on the geometrical construction of U_{valid}^τ as it is described in forthcoming sections.

E. Example – Single Integrator

The concept of U_{valid}^τ is explained considering a mobile robot with simple dynamics as of a single integrator. System's position at any time t for constant input u can be given as $p_r(t) = p_r(0) + tu$, where $p_r(0)$ is the current position of the robot. The motion of the system is constrained by $\sqrt{\dot{p}_r(t)} \leq v_{max}$, where v_{max} is the system's maximum speed limit. As the state transition equation of a single integrator is $\dot{p}_r(t) = u$, U_{ad} will be a set of input where $\|u\| \leq v_{max}$. U_{ad} has the shape of a disc with a center at the origin and a radius of v_{max} in robot's input control space. $UO_i(t)$ will be equivalent to $NLVO(t)$ for the single integrator. It also has the shape of a disc, with its center at $(p_i(t) - p_r(0))/t$ and radius equal to $(r_r + r_i)/t$ in robot input control space U, where r_r and r_i are the radius of the robot and the $i-th$ passive agent. Figure 2 shows U_{valid}^τ where $UO_i(t)$ is considered at equally spaced time instances by $\delta t = 0.1s$ and up to $\tau = 5s$ time horizon. The other parameters were set as: $p_r(0)$ =(0,0), $r_r = r_i = 0.4$, $v_{max} = 1\ unit/s$, and $p_i(t) = (2,0)$.



(a) (b) (c)

Fig. 2. The constraints on control space. The green region shows: (a) U_{ad}, (b) $U\backslash UO_i^\tau$, (c) U_{valid}^τ, for a considered robotic system as a single integrator.

III. COLLISION AVOIDANCE APPROACH

This section presents the proposed collision avoidance approach based on the concept of U_{valid}^τ, formulating the problem of robot navigation in a dynamic environment as an optimization problem. Given a random starting point, the proposed procedure will navigate the robot towards the goal without collision with any moving passive agent present. The goal position is assumed to be extracted from some waypoint plans for global navigation. The proposed algorithm allows the robot to quickly re-plan its course as new passive agents are discovered or new information regarding their future predicted states are received. u_{pref} is a control input to the kinodynamic model that leads the robot towards its goal. The exact value of $u = u_{pref}$, which is required to reach the goal, can be found by solving the equation $h(g(s_r,u,\tau)) = p_g$. If this point belongs to U_{valid} then all motion constraints are satisfied and the shape of the trajectory can be defined. Otherwise, control u^* that is given to the robot is the solution that minimizes the following cost:

$$u^* = \arg\min_{u\in U_{valid}^\tau} \|u_{pref} - u\| \quad (9)$$

This navigation problem is formalized as the problem of finding the $u \in U_{valid}^\tau$ that is nearest to u_{pref}, in terms of the Euclidean distance between them in control space U. The solution lies in the boundaries of U_{valid}^τ and is denoted by ∂U_{valid}^τ. The procedure used to find a roughly optimal solution is as follows: U_{ad}^τ represents a square grid of size d denoted by $U_{ad}^\tau(d)$, and the grid elements belonging to UO^τ are discarded. The algorithm in Figure 3 summarizes the procedure for finding and selecting an optimal control input by the robot for collision avoidance. A quadtree [25] was used to obtain precise tiling with rectangles over the boundaries of U_{valid}^τ. The best point is selected from all the remarkable points (center and/or corners) of rectangles. A rough optimization is performed by obtaining an optimal point from all rectangles, i.e. the controls that are closest to the preferred control.

Predict-plan-act Cycle: After each time step, the robot receives the predicted states of passive agents over a future time horizon of τ seconds, and U_{valid}^τ is built. The best control $u^* \in U_{valid}^\tau$ is computed with (9). The trajectory corresponding

to u^* will be collision-free for the next τ seconds. The computed trajectory is executed for one scan cycle of $\Delta t \ll \tau$, until a new scan cycle begins. This results in a continuous predict-plan-act navigation framework.

$UO^{\tau} \leftarrow \emptyset$
min $\leftarrow \infty$
for $j = 0$ to n **do**
  for all $u_j \in U_{ad}(d)$ **do**
    for $t = 0{:}\ \delta t{:}\ \tau$ **do**
      if $u_j \in \cup_{i \in N} UO_i(t)$
        $UO^{\tau} = UO^{\tau} \cup u_j$
        break loop
      end if
    end for
  end for
end for
$U^{\tau}_{valid} \leftarrow U_{ad}(d) \backslash UO^{\tau}$
  $\partial U^{\tau}_{valid} \leftarrow QuadTree(U^{\tau}_{valid})$
for all $u_j \in \partial U^{\tau}_{valid}$
  if $\|u_{pref} - u\| <$ min **then**
    min $\leftarrow \|u_{pref} - u\|$
    $argmin \leftarrow u_i$
  end if
end for

Fig. 3. Algorithm: find best control input u^*.

IV. IMPLEMENTATION AND RESULTS

The proposed approach was applied to a car-like robot model for evaluating its performance.

A. Considered Robot Model

A robot with car-like kinodynamic was considered with the following parameters: its real-wheel axle center is $p_r = [x_r, y_r]^T$, its orientation is θ, and its steering wheel angle is ϕ. The state transition equations of the robot are given by:

$$\dot{x}_r(t) = v cos\theta;\ \dot{y}_r(t) = v sin\theta;\ \dot{\theta}_r(t) = vk \quad (10)$$

where v is the input speed control. As in [20], the curvature was taken directly as input, while the steering angle ϕ can be computed for k as $\phi = \tan^{-1} kL$, where $L$is the wheelbase of the car. The motion of a car-like robot is considered to be constrained as:

$$\sqrt{\dot{x}_r(t)^2 + \dot{y}_r(t)^2} \in [0, v_m] \quad (11)$$

$$\frac{\dot{\theta}_r(t)}{v(t)} \in [-k_m, k_m] \quad (12)$$

where v_m is the maximum speed constraint and k_m is the robot's maximum curvature constraint. U_{ad} for this system was defined by k_m and v_m, and its geometry was a rectangular region of the input control space. Robot's position at time t can be obtained by integrating (10) assuming that controls will remain constant over the time horizon $[0, t]$, as:

- if $k \neq 0$:

$$p_r(t) = p_o + k^{-1} R(\theta_o)[\sin(vkt) \quad 1 - cos(vkt)]^T \quad (13)$$

- if $k = 0$:

$$p_r(t) = p_o(t) + R(\theta_o)[tv \quad 0]^T \quad (14)$$

where p_o is the current position and θ_o is the current orientation of the robot, and the rotation matrix $R(\theta_0)$ is:

$$R(\theta_0) = [cos\theta_o \quad -sin\theta_o\ ; sin\theta_o \quad cos\theta_o] \quad (15)$$

B. Implementation Details - Simulations Setup

The maximum speed and curvature were set to $1.5 unit/s$ and $1.5 unit^{-1}$ respectively. The radius of the robot was set to 1. An open environment was constructed for the simulations, bounded by (0,0) and (22,22). This environment was set to be crowded by multiple passive agents of circular shape and radius equal to one. The scan cycle was set to Δt=0.05s. Each passive agent was set to move along a specific trajectory, i.e. straight lines or arcs, with speed less than or equal to 1. The probability of an agent changing its path and speed within one second was set to 0.2. The interaction time horizon τ was set to 3.5s. Figure 1 shows the snapshots of simulations at multiple time instances, where the green marked robot successively avoids multiple red marked passive agents and reaches the yellow marked goal. Experiments are carried out on an Intel Core i5-3550/4GB RAM computer and the proposed algorithm was implemented using C++.

C. Performance Results

Each result value was a mean of 20 trials. Figure 4 shows the performance comparison with grid size d as 16 and 32. Comparisons were drawn for a varying number of passive agents using three parameters. The time needed for computing optimal control u^* in a scan cycle is shown in Figure 4(a). The time needed by the robot to reach the goal is shown in Figure 4(b). The success rate, which indicates the percentage of the trail in which the robot successfully reaches the goal from the starting position, is shown in Figure 4(c). For d=16 the computation time is much less on average, whereas its success rate is nearly the same when comparing it with d=32. However, when d=32 the robot takes a more direct path to the goal, and the time to reach a goal is less on average, as shown in Figure 4(b).

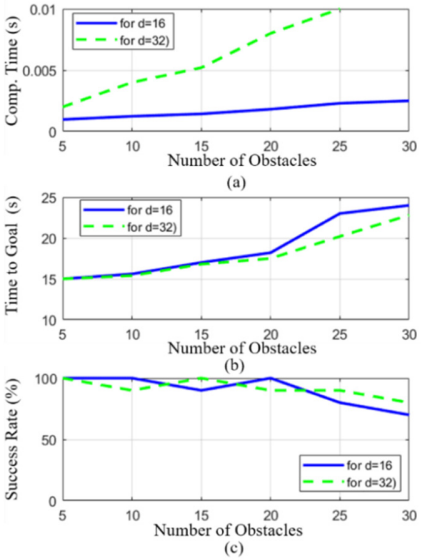


Fig. 4. (a) Computation time, (b) time to goal, and (c) success rate on various numbers of passive agents with grid size d={16,32}.

D. Comparative Analysis

The proposed approach was compared with GVO and AVO for a grid size of d=16. Figure 5 shows the comparison of the proposed approach with a GVO having 256 controls sampled. Results showed that the proposed algorithm performed better than GVO on all three parameters. The proposed approach's computation time and the time to goal was much less, while the success rate was averagely higher compared to GVO. An additional advantage emerges as GVO is restricted to a linear equation of motion of passive agents, while the proposed approach has no such restriction.

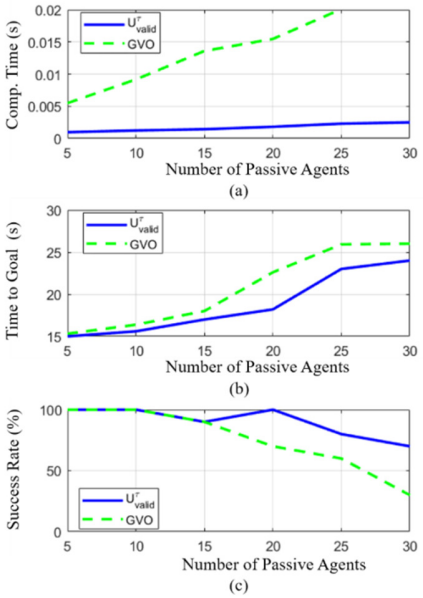


Fig. 5. (a) Computation time, (b) time to goal, and (c) success rate on various numbers of passive agents for the proposed approach vs GVO.

Figure 6 shows a comparison of the proposed approach with AVO. The reciprocal collision avoidance aspect of AVO was removed to make a relative comparison, while the linear programming based optimization aspect was kept. The results showed better performance for the proposed approach U^{τ}_{valid} on all three parameters, pointing out its effectiveness on the considered problem.

V. CONCLUSION

This paper presented a new approach that maps different constraints on a robotic system as geometrical bounds on the robot's input control space, allowing the computation of the optimal control input for collision-free local navigation using Quadtree. As it was shown, this computation can be executed in real-time. Furthermore, the proposed collision avoidance approach had superior performance compared to other velocity-based approaches for a considered environment. Moreover, an implementation of the proposed approach on a car-like robot was presented. This implementation can be easily adapted to other types of robotic systems, which is an additional advantage which can be studied in the future.

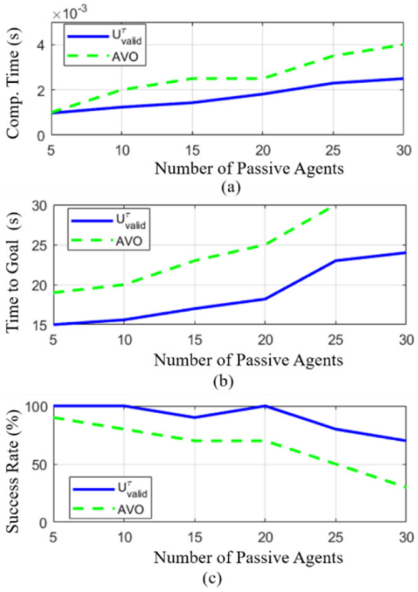


Fig. 6. (a) Computation time, (b) time to goal, and (c) success rate on various numbers of passive agents for the proposed approach vs AVO.

REFERENCES

[1] E. Prassler, J. Scholz, and P. Fiorini, "A robotics wheelchair for crowded public environment," *IEEE Robotics Automation Magazine*, vol. 8, no. 1, pp. 38–45, Mar. 2001, https://doi.org/10.1109/100.924358.

[2] A. Breitenmoser, F. Tâche, G. Caprari, R. Siegwart, and R. Moser, "MagneBike: toward multi climbing robots for power plant inspection," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Industry track*, Richland, SC, USA, May 2010, pp. 1713–1720.

[3] J.- Laumond, P. E. Jacobs, M. Taix, and R. M. Murray, "A motion planner for nonholonomic mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 5, pp. 577–593, Oct. 1994, https://doi.org/10.1109/70.326564.

[4] A. Scheuer and T. Fraichard, "Continuous-curvature path planning for car-like vehicles," in Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS '97, Grenoble, France, Sep. 1997, vol. 2, pp. 997–1003, https://doi.org/10.1109/IROS.1997.655130.

[5] F. Lamiraux and J.- Lammond, "Smooth motion planning for car-like vehicles," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 4, pp. 498–501, Aug. 2001, https://doi.org/10.1109/70.954762.

[6] E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-Time Motion Planning for Agile Autonomous Vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 1, pp. 116–129, 2002, https://doi.org/10.2514/2.4856.

[7] D. Hsu, R. Kindel, J.-C. Latombe, and S. Rock, "Randomized Kinodynamic Motion Planning with Moving Obstacles," *The International Journal of Robotics Research*, vol. 21, no. 3, pp. 233–255, Mar. 2002, https://doi.org/10.1177/027836402320556421.

[8] M. Zucker, J. Kuffner, and M. Branicky, "Multipartite RRTs for Rapid Replanning in Dynamic Environments," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, Apr. 2007, pp. 1603–1609, https://doi.org/10.1109/ROBOT.2007.363553.

[9] K. Kant and S. W. Zucker, "Toward Efficient Trajectory Planning: The Path-Velocity Decomposition," *The International Journal of Robotics Research*, vol. 5, no. 3, pp. 72–89, Sep. 1986, https://doi.org/10.1177/027836498600500304.

[10] J. Peng and S. Akella, "Coordinating Multiple Robots with Kinodynamic Constraints Along Specified Paths," *The International Journal of*

Robotics Research, vol. 24, no. 4, pp. 295–310, Apr. 2005, https://doi.org/10.1177/0278364905051974.

[11] J. van den Berg and M. Overmars, "Kinodynamic motion planning on roadmaps in dynamic environments," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Diego, CA, USA, Oct. 2007, pp. 4253–4258, https://doi.org/10.1109/IROS.2007.4398968.

[12] F. Gaillard, M. Soulignac, C. Dinont, and P. Mathieu, "Deterministic Kinodynamic Planning with hardware demonstrations," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, San Francisco, CA, USA, Sep. 2011, pp. 3519–3525, https://doi.org/10.1109/IROS.2011.6094769.

[13] C. Chen, M. Rickert, and A. Knoll, "Kinodynamic motion planning with Space-Time Exploration Guided Heuristic Search for car-like robots in dynamic environments," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, Sep. 2015, pp. 2666–2671, https://doi.org/10.1109/IROS.2015.7353741.

[14] M. Bennewitz, W. Burgard, and S. Thrun, "Learning motion patterns of persons for mobile service robots," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, Washington, DC, USA, May 2002, vol. 4, pp. 3601–3606 vol.4, https://doi.org/10.1109/ROBOT.2002.1014268.

[15] D. Vasquez and T. Fraichard, "Motion prediction for moving objects: a statistical approach," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, New Orleans, LA, USA, Apr. 2004, vol. 4, pp. 3931-3936 Vol.4, https://doi.org/10.1109/ROBOT.2004.1308883.

[16] S. Kim *et al.*, "BRVO: Predicting pedestrian trajectories using velocity-space reasoning," *The International Journal of Robotics Research*, vol. 34, no. 2, pp. 201–217, Feb. 2015, https://doi.org/10.1177/0278364914555543.

[17] A. Bera, S. Kim, T. Randhavane, S. Pratapa, and D. Manocha, "GLMP-realtime pedestrian path prediction using global and local movement patterns," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, May 2016, pp. 5528–5535, https://doi.org/10.1109/ICRA.2016.7487768.

[18] P. Fiorini and Z. Shiller, "Motion Planning in Dynamic Environments Using Velocity Obstacles," *The International Journal of Robotics Research*, vol. 17, no. 7, pp. 760–772, Jul. 1998, https://doi.org/10.1177/027836499801700706.

[19] Z. Shiller, F. Large, and S. Sekhavat, "Motion planning in dynamic environments: obstacles moving along arbitrary trajectories," in *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, Seoul, South Korea, May 2001, vol. 4, pp. 3716–3721 vol.4, https://doi.org/10.1109/ROBOT.2001.933196.

[20] D. Wilkie, J. van den Berg, and D. Manocha, "Generalized velocity obstacles," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, USA, Oct. 2009, pp. 5573–5578, https://doi.org/10.1109/IROS.2009.5354175.

[21] J. van den Berg, J. Snape, S. J. Guy, and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," in *2011 IEEE International Conference on Robotics and Automation*, Shanghai, China, May 2011, pp. 3475–3482, https://doi.org/10.1109/ICRA.2011.5980408.

[22] K. M. Zuhaib *et al.*, "Collision Avoidance from Multiple Passive Agents with Partially Predictable Behavior," *Applied Sciences*, vol. 7, no. 9, p. 903, Sep. 2017, https://doi.org/10.3390/app7090903.

[23] D. R. Pawar and P. Poddar, "Car Black Box with Speed Control in Desired Areas for Collision Avoidance," *Engineering, Technology & Applied Science Research*, vol. 2, no. 5, pp. 281–284, Oct. 2012, https://doi.org/10.48084/etasr.194.

[24] M. B. Ayed, L. Zouari, and M. Abid, "Software In the Loop Simulation for Robot Manipulators," *Engineering, Technology & Applied Science Research*, vol. 7, no. 5, pp. 2017–2021, Oct. 2017, https://doi.org/10.48084/etasr.1285.

[25] R. A. Finkel and J. L. Bentley, "Quad trees a data structure for retrieval on composite keys," *Acta Informatica*, vol. 4, no. 1, pp. 1–9, Mar. 1974, https://doi.org/10.1007/BF00288933.