UNSYS digital

Journal of Unmanned System Technology (JUST)

Vol.2 No.3 (2014), pp. 105–118.

Analytic Optimal-Based Time Horizon Concept Using Pontryagin's Maximum Principle for Autonomous Navigation

OREN GAL

Mapping and Geoinformation Engineering
Technion, Israel Institute of Technology
E-mail: orengal@technion.ac.il

Abstract. This paper addresses the issue of agent motion planning in dynamic environments using an analytic optimal time horizon solution as basic character of our search. Specifically, we propose the optimal time horizon concept as a leading feature for our local on-line planner for omni-directional robots using Velocity Obstacle (VO). Using unified strategy, we propose a solution to the basic limitation of the VO search and planning method, i.e. when all the dynamic available velocities for the next time step are blocked in the velocity space and there is no legal node at the next time step of the greedy search. The computation of the minimum time horizon is formulated as a minimum time problem for omni-directional models. The analytic solution describes minimal and safe VO and allows efficient on-line planning in dynamic environments searching safe nodes. At each time step, a local greedy search in velocity space is explored. The analytic solution define VO shape and by that set the bounded velocity space and the next optimal node outside VO explored in the next time step. We introduce on-line planner for omni-directional robot that generates near-time optimal trajectories to the goal by using optimal time horizon. We demonstrate the solution of our approach showing the efficiency relative to the traditional VO for on-line motion planning in dynamic environments.

The paper was submitted on November 2, 2014; revised on December 20, 2014; accepted on December 24, 2014.:

1. INTRODUCTION

The main challenge in motion planning in dynamic environments is reaching the goal while searching and selecting only safe maneuvers. While reaching the goal cannot be guaranteed with an on-line planner, one can reduce the state space search to only safe states, i.e. states outside an obstacles that from which at least one other safe state is reachable.

Generally, we distinguish between local and global planners. The local planner generates one, or a few steps at every time step, whereas the global planner uses a global search to the goal over a time spanned tree. We can divide this work into global and local (reactive) planners. The global planners generate complete trajectories to the goal in static [1] and dynamic [2, 3] environments.

Examples of local (reactive) planners are [4, 5, 6], but most do not guarantee safety as their ability to look-ahead and avoid states of inevitable collision is very limited in a dense and fast changing environments and narrow passages such as indoor environments. Recently, iterative planners [7, 8, 9, 10, 11, 12] were developed that compute several steps at a time, subject to the available computation time. The trajectory is generated incrementally by exploring a search-tree and choosing the best branch. These planners too do not address the issue of safety and completeness.

Only a few works have addressed the safety issue in dynamic environments, which is crucial for partial (local) planning. One approach of safe planning is to use braking policies [13]; another is to ensure local avoidance for a limited time [11]. However, neither considers the dynamic of the moving robot. A promising approach to safe motion planning in dynamic environment is the consideration of "Regions of Inevitable Collision" (RIC) first introduced in [14] and later extended to Inevitable Collision States (ICS) in [15, 16, 17, 18].

We address the issue of searching safe maneuvers for an on-line local planner in dynamic environments for a single agent using an improved Velocity Obstacles (VO) method. Safety is guaranteed by searching robot velocity which does not penetrate the VO, which is generated for a carefully selected time horizon. The analytic time horizon is calculated at each time step allowing the robot to escape unsafe states in a blocked velocity space and to explore and search a new states, *a limitation that was not treated yet by using VO-based search planners.*

Computation of the analytic optimal time horizon, which is obstacle specific, is formulated as a minimum time problem that minimizes the time for the robot velocity to exit the velocity obstacle including penetration states. Searching solution for this minimum time problem is demonstrated for omni-directional robots satisfying Pontryagin's Maximum Principle providing necessary conditions for optimality. Determining the safe time horizon is computationally efficient and it does not require a prior mapping of Inevitable Collision States (ICS). By that, an efficient and safe search can be done in a very complicated dynamic environments which can not be handle with the traditional VO planner search [19].

Since the time horizon is obstacle specific, motion safety is guaranteed if obstacles can be avoided individually or if the state-space between the current position and the goal state stays connected during planner search process. The modified VO planner based on analytic time horizon search are used for an on-line planner that generates near-time optimal trajectories by minimizing at each time step the time-to-go to the goal. The planner is demonstrated for on-line motion planning in static and dynamic environments.

2. The Velocity Obstacle

The velocity obstacle represents the set of all colliding velocities of the robot with the neighboring obstacles. It essentially maps static and moving obstacles into the robot's velocity space. The velocity obstacle (VO) of a planar circular obstacle, B, that is moving at a constant velocity v_b, is a cone in the velocity space at point A, as shown in Figure 1. In Figure 1, the position space and velocity space of A are overlaid to illustrate the relation between the two spaces. The VO is generated by first constructing the Relative Velocity Cone (RVC) from A to the boundaries of B, then translating RVC by v_b.

Each point in VO represents a velocity vector that originates at A. Any velocity of A that penetrates VO is a colliding velocity that would result in collision between A and B at some future time. Figure 1 shows two velocities of A: one that penetrates VO, and is hence a colliding velocity, and one that does not. All velocities of A that are outside of VO are safe as long as B stays on its current course. The velocity obstacle thus allows determining if a given velocity is potentially dangerous, and suggesting possible changes to this velocity to avert collision. If B is known to move along a curved trajectory or at varying speeds, it would be best represented by the nonlinear velocity obstacle discussed next.

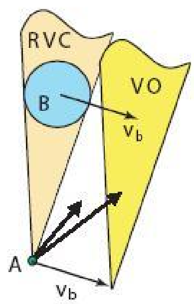

Figure 1. A Linear Velocity Obstacle

3. Nonlinear Velocity Obstacle

The nonlinear velocity obstacle ($NLVO$) accounts for a general trajectory of the obstacle, while assuming a constant velocity of the robot [20]. It applies to the scenario shown in Figure 2, where, at time t_0, a point A attempts to avoid an obstacle, B, that is following a general known trajectory, $c(t)$, and at time t_0 is located at $c(t_0)$. B represents the set of points that define the geometry of the obstacle, grown by the radius of the robot. If B is a circle, then $c(t)$ represents the trajectory followed by its center.

The $NLVO$ consists of all velocities of A at t_0 that would result in collision with the obstacle at any time $t > t_0$. Selecting a single velocity, v_a, at time $t = t_0$ outside the $NLVO$ thus guarantees to avoid collision at all times. It is constructed as a union of its temporal elements, $NLVO(t)$, which is the set of all absolute velocities of A, v_a, that would result in collision at a specific time t.

Referring to Figure 2, v_a that would result in collision with point p in B at time $t > t_0$, expressed in a frame centered at $A(t_0)$, is simply

$$v_a = \frac{c(t) + r}{t - t_0}, \tag{3.1}$$

where r is the vector to point p in the obstacle's fixed frame. The set, $NLVO(t)$ of all absolute velocities of A that would result in collision with any point in B at time $t > t_0$ is thus:

$$NLVO(t) = \frac{c(t) + B}{t - t_0}. \tag{3.2}$$

Clearly, $NLVO(t)$ is a scaled B, located at a distance from A that is inversely proportional to the collision time t. The entire $NLVO$ is the union of its temporal subsets from t_0, the current time, to some set time horizon t_h:

$$NLVO(t) = \bigcup \frac{c(t) + B}{t - t_0}. \tag{3.3}$$

The smallest safe time horizon is the one that allows sufficient time to avoid or escape collision as detailed next in Section 4. The non-linear v-obstacle is a warped cone. If $c(t)$ is bounded over $t = (t_0, \infty)$, then the apex of this cone is at $A(t_0)$. The boundaries of the $NLVO$ represent velocities that would result in A grazing B.

4. Analytic Optimal Time Horizon

The time horizon plays an important role in selecting feasible avoidance maneuvers. It allows considering only those maneuvers that would result in a collision within a specified time interval and efficiently search safe maneuvers in velocity space. Setting the time horizon too high would be too prohibitive, as it would mark as dangerous maneuvers resulting in collision at a distant time; selecting a too small time horizon would permit dangerous maneuvers that are too close and at too high speeds to avoid the obstacle.

It is essential that the proper time horizon ensures that a safe maneuver, even if temporarily pointing toward the obstacle, is selected. We define the smallest safe time horizon as the minimum time to exit the velocity obstacle from a given state, subject to

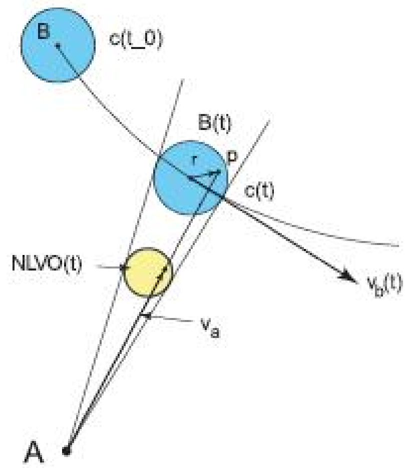

FIGURE 2. A Non-Linear Velocity Obstacle

robot dynamics and actuator constraints. It is the solution of the minimization problem satisfying system dynamic and control constrains:

$$min \int_{t_0}^{t_h} 1dt \tag{4.1}$$

with the initial condition

$$x(t_0), \dot{x}(t_0) \tag{4.2}$$

the final condition

$$\dot{x}(t_h) \notin \sum_{i=1}^{n} VO_{i\infty} \tag{4.3}$$

avoiding collision

$$x(t) \notin \sum_{i=1}^{n} B_i(t) \forall t \tag{4.4}$$

The main significance of the time horizon parameter using VO was first introduced in [21]. The time horizon calculated as the minimum between stopping and passing time for each obstacle as approximations to the exact optimization problem. Numeric solution of the optimal time horizon for point mass model with cubic control constraints were presented in [21] based on extremal trajectories generated from the boundary of the control effort. These formulation of time horizon defines conservative approximation of VO as the boundary of ICS without analytic solution escaping VO in a case of bounded velocity space.

We address time-optimal control problem, searching the best maneuver in case of blocked nodes, i.e. all of the possibles nodes in the next time step are located inside

VO. The omni-directional model for point mass system dynamics is:

$$\ddot{x} = u_1; \tag{4.5}$$

$$\ddot{y} = u_2; \tag{4.6}$$

and control constraints:

$$\left\{(u_1, u_2 \in U) | u_1^2 + u_2^2 \leq R_r^2\right\} \tag{4.7}$$

where $(x, y)^T \in R^2$ and $(u_1, u_2)^T \in U$ represent the joint variables and actuator efforts, respectively. The state vector is $x = (x_1, x_2, x_3, x_4)^T = (x, \dot{x}, y, \dot{y})^T$ and $f = (x_2, u_1, x_4, u_2)$.

For such control constraints and system dynamics, we introduce analytic solution to the optimal time horizon problem. The meaning of the boundary condition ψ is that the control signal $(u_1, u_2)^T$ at t_h is tangent to the closest boundary of the infinite VO from the current position of the robot.

For our case, based on Pontryagin's Maximum Principle necessary condition for optimality can be written as:

$$\psi_{t_h} = A_1 \cdot x_4(t_h) + A_2 \cdot x_2(t_h) + A_3 = 0 \tag{4.8}$$

where A_i are the coefficients of the closest boundary of the infinite VO as can be seen in Figure 3:

$$A_1 = v_{bx} - x_{b1} \tag{4.9}$$

$$A_2 = y_{b1} - v_{by} \tag{4.10}$$

$$A_3 = v_{bx} A_2 - v_{by} A_1 \tag{4.11}$$

where (x_{b1}, y_{b1}) is one of the tangent point between the obstacle B and its apex at v_b, separated to v_{bx} and v_{by}. The slope of the VO boundary is $-\frac{A_2}{A_1}$ as can be shown in Figure 3. A_i depend on time, but can be computed as constant for one time step. The Hamiltonian function is expressed by:

$$H(x, u, \lambda) = l(x, u) + \sum_{i=1}^{n} \lambda_i f_i(x, u) \tag{4.12}$$

As well known for the optimal time trajectory case [22]

$$l(x, u) = 1 \tag{4.13}$$

The control signal can be written in polar coordinates:

$$H(x, u, \lambda) = 1 + \lambda_1 x_2 + \lambda_2 u_1 + \lambda_3 x_4 + \lambda_4 u_2 \tag{4.14}$$

$$H(x, u, \lambda) = 1 + \lambda_1 x_2 + \lambda_2 r cos(\theta) + \lambda_3 x_4 + \lambda_4 r sin(\theta) \tag{4.15}$$

We can see that the signs of λ_2,λ_4 determine the value of the optimal control solution:

$$u^* = argmin_{u \in U} \left\{1 + \lambda_2 r cos(\theta) + \lambda_4 r sin(\theta)\right\} \tag{4.16}$$

The adjoint variables defined as:

$$\dot{\lambda}_i = -\frac{\partial H}{\partial x_i} \tag{4.17}$$

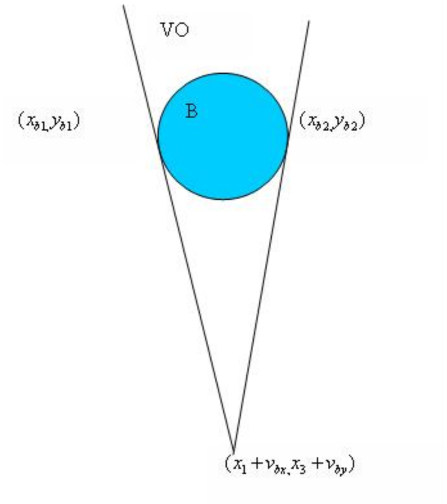

FIGURE 3. The Velocity Obstacles boundaries

therefore:

$$\lambda_1(t) = C_1 \tag{4.18}$$
$$\lambda_2(t) = C_2 - C_1 t \tag{4.19}$$
$$\lambda_3(t) = C_3 \tag{4.20}$$
$$\lambda_4(t) = C_4 - C_3 t \tag{4.21}$$

The coefficients also satisfying the boundary condition:

$$\left[\lambda_i + \vartheta^T \frac{\partial \psi}{\partial x_i}\right]_{t_h} = 0 \tag{4.22}$$

Therefore there's no switching along the trajectories and the initial values determine the optimal solution:

$$\lambda_1(t) = \lambda_3(t) = 0 \tag{4.23}$$
$$\lambda_2(t) = C_2 = -\vartheta^T A_2 \tag{4.24}$$
$$\lambda_4(t) = C_4 = -\vartheta^T A_1 \tag{4.25}$$

The optimal angular from omni-directional model:

$$\frac{\partial u^*}{\partial \theta} = -\lambda_2 r sin(\theta) + \lambda_4 r cos(\theta) \tag{4.26}$$
$$\theta^* = arctan\left(\frac{\lambda_4}{\lambda_2}\right) = arctan\left(\frac{A_1}{A_2}\right) \tag{4.27}$$

The geometric interpretation for the optimal angular from the omni-directional model reveals that the robot should move vertically to the boundary line of the infinite VO as described in this section.

The suggested solution determines the optimal time horizon for each obstacle with very low computation effort. Mapping the inevitable collision velocities (into the VO for this specific obstacle) to the robot velocity space ensures safe and sub-optimal trajectories, guiding the greedy search planner with an analytic solution. In cases of crowded or dynamic environments with high obstacle's velocities, the velocity space can be bounded at a current time step, and an extended strategy should be defined rather than detailed in [21], escaping VO of these unique cases. The optimal time horizon for omni-direction model is computed for each obstacle in each time step in order to define VO's.

5. Analytic-Based Search Planner

Our planner is a local search planner with one step ahead in time, exploring nodes and choosing the best next safe state (known as greedy search i.e. making the locally optimal choice at each stage). Most of the local planners cannot ensure convergence to the goal and in some cases may lead to Inevitable Collision States (ICS) [16]. The main advantage of our search is based on the fact that computation time allows on-line implementing in real-time platforms, and above all, our heuristic of an optimal time horizon and near-time optimal cost function maps beforehand ICS and generates time optimal trajectory. The well known off-line global search algorithms (such as A* and Dijkstra or Uniform Cost Search) can not be used in fast changing environments where the future obstacle's trajectories are unknown, where the level of risk to the robot can not be predicted beforehand.

The analytic time horizon enable the robot to avoid ICS and to search the next safe state based on a "legal" velocity space while the traditional VO concept dooes not allow to find a safe velocity for the next time step. The analytic solution extend the traditional VO planner search method and defines the strategy search for a cases of blocked attainable velocity space for the next time step in velocity space.

The search is guided by a cost function planner applying the most safe maneuver at every time step. An unsafe states ahead in time are recognized before the robot enters to an unsafe states without escaping option. For one obstacle, our planner can ensure safety, but the planner is not a complete one. By using an analytic search, the planner computes near-time optimal and safe trajectory to the goal. Our planner search extends the VO planner method, but also relevant to the case of non-linear obstacle's trajectories case known as the Non Linear Velocity Obstacle ($NLVO$) [20].

System dynamics of the planner is the same as introduced in Section 4.

5.1. **Attainable Cartesian Velocities.** The set of attainable Cartesian velocities (ACV) of the maneuvering robot represents the avoidance maneuvers that are dynamically feasible over a given time interval, Δt [23].

The attainable *Cartesian* velocities, $ACV(t+\Delta t)$ are integrated from the current state $(x_1(t), x_2(t))$ by applying all admissible controls $u(t) \in U$. The geometric shape of $ACV(t+\Delta t)$ depends on the specific system dynamics: For a point mass model, with constant control constraints, it is a rectangle that coincides with the set of admissible controls:

$$\dot{x}_1 = x_2 \quad (5.1)$$

$$\dot{x}_2 = u. \quad (5.2)$$

where $x_1, u \in R^2$.

$$ACV(t + \Delta t) = \{v | v = v(t) + \Delta t u, u \in U\}. \quad (5.3)$$

The attainable velocities at time $t + \Delta t$ apply to the position $x(t + \Delta t)$. Thus, the attainable velocities, when intersected with VO that correspond to the same position, would indicate those velocities that are safe if selected at time $t = t + \Delta t$.

This integration is approximated by Euler integration, integrating the current velocity by Δt to yield $x_1(t+\Delta t)$ and adding the set of attainable velocities to $v(t+\Delta t)$. The analytic time horizon gives an optimal solution escaping from VO in case of blocked ACV in velocity space.

5.2. **Search Tree.** The planner use a local search tree that expand over t to the goal. Each node contains the current position and velocity of the robot at the current time step, the planner compute the position and velocity at $t+\Delta t$ for ACV velocities options as detail at Section 5.1. The planner inform the value of the node at each step to the best next value if the relevant branch is safe .i.e. his velocity is out of the velocity space with the proper time horizon, and the time to go to the goal is minimal from his current state to the goal as detail at Section 5.3.

The main contribution of our work can be demonstrated in case of blocked nodes in the velocity space in the search tree for the next time step. In case of blocked nodes, i.e. all of the nodes located inside the VO, the planner choose the node that lead outside VO as soon as possible avoiding collision and formulated as analytic time horizon based search. Without using analytic time horizon formulation, there was no legal option for the next node, and a safe trajectory to the goal could not be found. An example for such cases will be demonstrated in Section 6.

5.3. **Cost Function.** Our search is guided by a minimum time cost function to produce near-time optimal trajectories to the goal. The cost function for each primitive is the minimum time to the goal from that state to the goal. It is determined by first computing the minimum time to the goal $w(x_1, x_2, x_{1f}, x_{2f})$ from the current state (x_1, x_2) to the goal (x_{1f}, x_{2f}) for each axis [22, 24]:

$$w(x_1, x_2, x_{1f}, x_{2f}) =$$

$$-x_2 - x_{2f} + 2\sqrt{-x_1 + x_{1f} + \frac{x_2^2}{2} + \frac{x_{2f}^2}{2}}, \quad (5.4)$$

$$if(x_1, x_2) \in R$$

$$x_2 + x_{2f} + 2\sqrt{x_1 - x_f + \frac{x_2^2}{2} + \frac{x_{2f}^2}{2}}, \quad (5.5)$$

$$if(x_1, x_2) \notin R$$

$$(5.6)$$

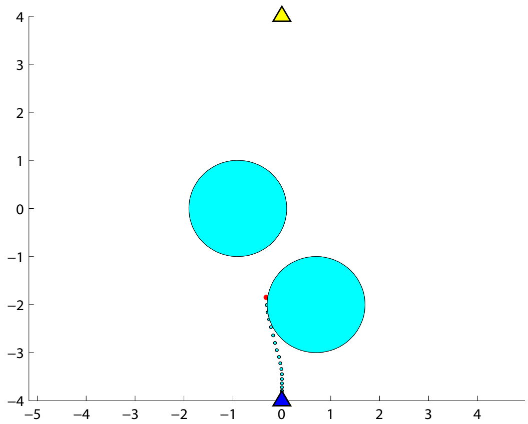

FIGURE 4. Avoiding Two Obstacles Using Analytic Time Horizon

where R is the region below and above the switching curve in the state space:

$$R(x_1, x_2) =$$
$$\{x_2^2 - 2(x_1 - x_{1f} + \frac{x_{2f}^2}{2}) > 0,$$
$$\{x_2^2 + 2(x_1 - x_{1f} - \frac{x_{2f}^2}{2}) < 0\} \tag{5.7}$$

where x_1 and x_2 are the start point and the velocity at starting point respectively, and x_{1f} and x_{2f} are the target point and the velocity at the target, respectively. Considering both axes, the minimum time to the goal used in the cost function is the largest of the times computed for both axes [22]. The cost function compute time-optimal trajectory without obstacles and near-time optimal trajectory with obstacles. This cost function does not depend on the planner and can be implement in different planners.

6. EXAMPLES

The on-line planner was implemented and tested for obstacle-free, and for static and dynamic environments. In the example shown in Figure 4, the robot, represented by a point, starts near point (0,-4) at zero speed, attempting to reach the goal at point (0,4) (marked by a yellow triangle) at zero speed, while avoiding two static obstacles. The trajectory is doted with red dot representing the current position of the robot. The bounded velocity space, representing velocity obstacles as yellow cycles and velocity vector (with green triangles) can be shown in Figure 5 related to the state space position as shown in Figure 4.

Clearly, that there is no gap to enter between $VO's$ in Figure 5 and the velocity vector is bounded in the velocity space. The trivial VO, with conservative and constant time horizon can not find the ultimate solution in such a case, and as a result, a

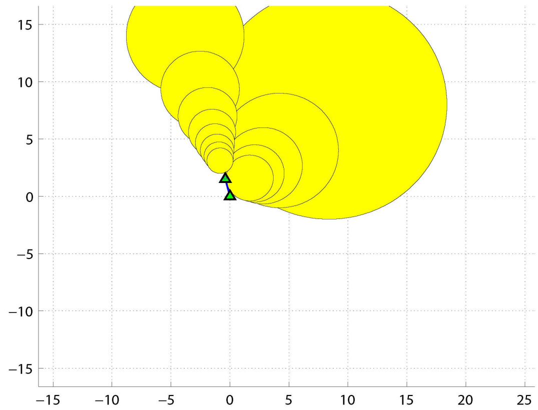

FIGURE 5. Blocked Velocity Space Avoiding Two Obstacles

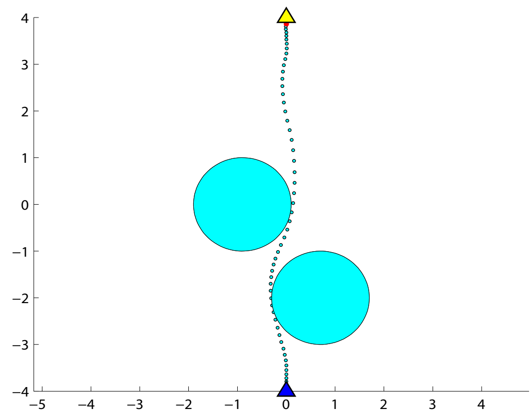

FIGURE 6. Final Trajectory Avoiding Two Obstacles Using Analytic Time Horizon

conservative trajectory will be computed. The robot avoids the obstacles to the left with high time horizon values, as can be shown in Figure 8. Moreover, in some other cases of dense and bounded velocity space no solution will be available at all. By using analytic time horizon, the robot escapes velocity obstacles and search for a safe maneuver in state space, as shown in Figure 6 and velocity space respectively, as shown in Figure 7.

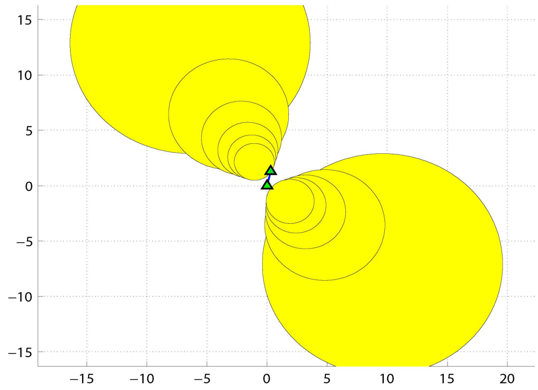

FIGURE 7. Escaping Blocked Velocity Space Using Analytic Time Horizon

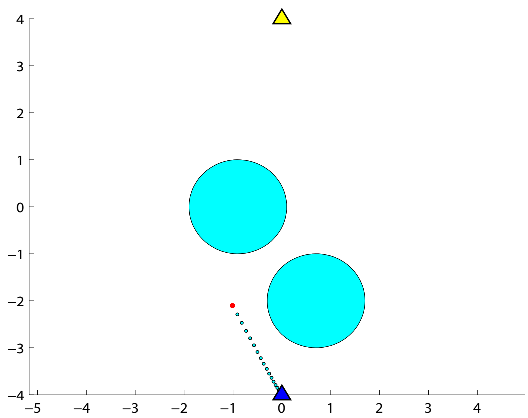

FIGURE 8. Conservative Solution of Avoiding Two Obstacles Using Constant Time Horizon: Blocked Velocity Space Caused to Conservative Trajectory Turning Left vs. Sliding on their Edges and Passing Between

Same analysis can be described consistently for dynamic obstacles. Since the analytic time horizon is set for each step according to the optimal escaping time to VO boundary, same planning behavior is still valid, the dynamic case acts in the same way.

7. Conclusions

A new analytic time horizon solution guided search in velocity space for omnidirectional robot in dynamic environments was presented. This time horizon, refers to the common cases in which the velocity space is bounded and the robot velocity can not escape from the velocity obstacles from the current state. These states can occur in very crowded environments or in dynamic environments with high obstacle's velocities. The analytic solution describes minimal and safe VO and allows efficient on-line planning in dynamic environments searching safe nodes. At each time step, a local greedy search in velocity space is explored. The analytic solution set the bounded velocity space and determine the next node that can be explored in the next time step. For other cases of unbounded velocity space, other time horizon strategies can be used, but a conservative or a collision will be the outcome. The planner generates near time-optimal trajectories, using the minimum time-to go to guide the tree search. The planner was demonstrated for a omni-directional dynamic model. The planner was successfully tested for static and dynamic environments, and as far as we know for the first time defines analytic strategy for bounded space using VO.

References

[1] J.-C. Latombe, *Robot Motion Planning.* Kluwer Academic Publishers, 1990.

[2] M. Erdman and T. Lozano-Perez, "On multiple moving objects," *Algorithmica*, vol. 2, pp. 447–521, 1987.

[3] K. Fugimura and H. Samet, "A hierarchical strategy for path planning among moving obstacles," *IEEE Transactions on Robotics and Automation*, vol. 5, pp. 61–69, 1989.

[4] L. Ulrich and J. Borenstien, "Vfh+: Reliable obstacle avoidance for fast mobile robots," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1998, pp. 1572–1577.

[5] N. Ko and R. Simmons, "The lane-curvature method for local obstacle avoidance," in *International Conference on Intelligence Robots and Systems*, 1998, pp. 1615–1621.

[6] J. Minguez and L. Montano, "Nearest diagram navigation. a new real-time collision avoidance approach," in *International Conference on Intelligence Robots and Systems*, 2000, pp. 2094–2100.

[7] T. Fraichard, "Planning in dynamic workspace: a state-time space approach," *Advanced Robotics*, vol. 13, pp. 75–94, 1999.

[8] D. H. R. K. J.-C. Latombe and S. Rock, "Randomized kinodynamic motion planning with moving obstacles," *Algorithmics and Computational Robotics*, vol. 4, pp. 247–264, 2000.

[9] O. Brock and O. Khatib, "Real time replanning in high- dimensional configuration spaces using sets of homotopic paths," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2000, pp. 550–555.

[10] N. S. J. Minguez L. Montano and R. Alami, "Global nearest diagram navigation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2001, pp. 33–39.

[11] E. Feron and M. D. E. Frazzoli, "Real time motion planning for agile autonomous vehicles," *AIAA Journal of Guidance Control and Dynamics*, vol. 25, pp. 116–129, 2002.

[12] D. Fox, W. Burgard, and S. Thrun, "The dynamic window approach to collision avoidance," *IEEE Robotics and Automation Magazine*, vol. 4, pp. 23–33, 1997.

[13] T. Wikman and W. N. M.S. Branicky, "Reflexive collision avoidance: a generalized approach," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1993, pp. 31–36.

[14] S. L. J. Kuffner, "Randomized kinodynamic planning," *International Journal of Robotics Research*, vol. 20, pp. 378–400, 2001.

[15] T. Fraichard, "A short paper about safety," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2007, pp. 1140–1145.

[16] S. P. T. Fraichard, "Safe motion planning in dynamic environment," in *International Conference on Intelligence Robots and Systems*, 2005, pp. 885–897.

[17] T. Fraichard and H. Asama, "Inevitable collision state-a step towards safer robots?" *Advanced Robotics*, vol. 18, pp. 1001–1024, 2004.

[18] N. Chan and M. Z. J. Kuffner, "Improved motion planning speed and safety using region of inevitable collision," in *ROMANSY*, July 2008, pp. 103–114.

[19] O. Gal, Z. Shiller, and E. Rimon, "Efficient and safe on-line motion planning in dynamic environment," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2009, pp. 88–93.

[20] Z. Shiller, F. Large and S. Sekhavat, "Motion planning in dynamic environments: Obstacle moving along arbitrary trajectories," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2001, pp. 3716–3721.

[21] Z. Shiller, O. Gal and T. Fraichard, "The nonlinear velocity obstacle revisited: the optimal time horizon," in *Workshop of Safety Motion Planning 2010.*

[22] S. Sundar and Z. Shiller, "Time optimal obstacle avoidance," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 1995, pp. 3075–3080.

[23] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *International Journal of Robotics Research*, vol. 17, pp. 760–772, 1998.

[24] S. Sundar, "Near-time optimal feedback control of robotic manipulators," Ph.D. dissertation, UCLA, 1995.