Visibility Velocity Obstacles (VVO): Visibility-Based Path Planning in 3D Environments

Oren Gal, Yerach Doytsher
Mapping and Geo-information Engineering
Technion - Israel Institute of Technology
Haifa, Israel
e-mail: {orengal,doytsher}@technion.ac.il

Abstract - **In this paper, we present as far as we know for the first time, a unique method combining visibility analysis in 3D environments with dynamic motion planning algorithm, named Visibility Velocity Obstacles (VVO). Our method is based on two major steps. The first step is based on analytic visibility boundaries calculation in 3D environments, taking into account sensors' capabilities including probabilistic consideration. In the second stage, we generate VVO transferring visibility boundaries from the position space to the velocity space, for each object. Each VVO represents velocity's set of possible future collision and visibility boundaries. Based on our analysis in velocity space, we plan our trajectory by selecting future robot's velocity at each time step, tracking after specific target considering visibility constraints as integral part of the velocities space. We formulate the tracked target in the environment as part of our planner and include visibility analysis for the next time step as part of our planning in the same search space. For the first time, we define visibility aspects as part of velocity space, where all the objects are modeled from visibility point of view. We introduce potential trajectory planner combining unified 3D visibility analysis for target tracking as part of dynamic motion planning.**

Keywords - Visibility; Motion planning; 3D; Urban environment; Spatial analysis.

I. INTRODUCTION

Trajectory planning has developed alongside the increasing numbers of Unmanned Aerial Vehicles (UAVs), drones unmanned ground vehicles all over the world, with a wide range of applications such as surveillance, information gathering, suppression of enemy defenses, air to air combat, mapping buildings and facilities, etc.

Most of these applications are involved in very complicated environments (e.g., urban), with complex terrain for civil and military domains [1].

With these growing needs, several basic capabilities must be achieved. One of these capabilities is the need to avoid obstacles such as buildings or other moving objects, while autonomously navigating in 3D urban environments.

Path planning problems have been extensively studied in the robotics community, finding a collision-free path in static or dynamic environments, i.e., moving or static obstacles. Over the past twenty years, many methods have been proposed, such as starting roadmap, cell decomposition, and potential field [6].

In this paper, we present visibility aspects as part of velocity space, where all the objects are modeled from visibility point of view. We introduce potential trajectory planner combining unified 3D visibility analysis for target tracking as part of dynamic motion planning. In the first part, we formulate visibility boundaries problem and introduce analytic solution that in the following sub-section integrated with sensor's limitations. Later on, we present the VVO method, demonstrated with visibility boundaries with cars, pedestrians and buildings visibility boundaries. In the last part, we suggest pursuer planner using VVO for UAV test case.

II. RELATED WORK

Path planning becomes trajectory planning when a time dimension is added for dynamic obstacles [7][8]. Later on, a vehicle's dynamic and kinematic constraints have been taken into account, in a process called kinodynamic planning [9]. All of these methods focus solely on obstacle avoidance.

Trajectory planning for air traffic control and ground vehicles has been well studied [10], based on short path algorithms using 2D polygons, 3D surfaces [11]. UAVs navigation has also been explored with vision-based methods [12], with local planning or a predefined global path [13].

UAV path planning is different from simple robot path planning, due to the fact that a UAV cannot stop, and must maintain its velocity above the minimum, as well as not being able to make sharp turns.

UAV path planning methods usually decompose the path planning into two steps: first, using some common path planning method in a polygonal environment [6], then, considering UAV dynamic and kinematic constraints into the trajectory [14]. These methods assume decoupling, which affects the trajectory, as stated by all authors.

However, most of the effort focused on UAV trajectory planning is related to obstacle avoidance with kinodynamic constraints, without taking into account visibility analysis as part of the nature of the trajectory in urban environments.

The visibility problem has been extensively studied over the last twenty years, due to the importance of visibility in GIS and Geomatics, computer graphics and computer vision, and robotics. Accurate visibility computation in 3D environments is a very complicated task demanding a high computational effort, which could hardly have been done in a very short time using traditional well-known visibility methods [15]. The exact visibility methods are highly complex, and cannot be used for fast applications due to their long computation time. Previous research in visibility computation has been devoted to open environments using DEM models, representing raster data in 2.5D (Polyhedral model), and do not address, or suggest solutions for, dense built-up areas. Most of these works have focused on approximate visibility computation, enabling fast results using interpolations of visibility values between points, calculating point visibility with the Line of Sight (LOS) method [16]. Other fast algorithms are based on the conservative Potentially Visible Set (PVS) [17]. These methods are not always completely accurate, as they may render hidden objects' parts as visible due to various simplifications and heuristics.

A vast number of algorithms have been suggested for speeding up the process and reducing computation time. Franklin [18] evaluated and approximated visibility for each cell in a DEM model based on greedy algorithms. Wang et al. [19] introduced a Grid-based DEM method using viewshed horizon, saving computation time based on relations between surfaces and the LOS method. Later on, an extended method for viewshed computation was presented, using reference planes rather than sightlines [20].

One of the most efficient methods for DEM visibility computation is based on shadow-casting routine. The routine cast shadowed volumes in the DEM, like a light bubble [21]. Extensive research treated Digital Terrain Models (DTM) in open terrains, mainly Triangulated Irregular Network (TIN) and Regular Square Grid (RSG) structures. Visibility analysis in terrain was classified into point, line and region visibility, and several algorithms were introduced, based on horizon computation describing visibility boundary [22].

Only a few works have treated visibility analysis in urban environments. A mathematical model of an urban scene, calculating probabilistic visibility for a given object from a specific viewcell in the scene, has been presented by [23]. This is a very interesting concept, which extends the traditional deterministic visibility concept. Nevertheless, the buildings are modeled as cylinders, and the main challenges of spatial analysis and building model were not tackled. Other methods were developed, subject to computer graphics and vision fields, dealing with exact visibility in 3D scenes, without considering environmental constraints. Plantinga and Dyer [15] used the aspect graph – a graph with all the different views of an object. Due to their computational complexity, all of these works are not applicable to a large scene with near real-time demands, such as UAV trajectory planning.

III. VISIBILITY BOUNDARIES ANALYSIS

A. Problem Statement

We consider visibility problem in a 3D urban environment, consisting of static constant objects and dynamic objects.

Given:

- Static objects:
  3D buildings modeled as 3D cubic parameterization $\sum_{i=1}^{N_{of\_build}} C_i(x, y, z = {}^{h_{\max}}_{h_{\min}})$
- Dynamic objects:
  Moving cars modeled as 3D cubic parameterization, $C_{car}(x, y, z)$
- Pedestrian modeled as cylinder parameterization, $C_{peds}(x, y, z)$
- Trees modeled with two cylinder parameterization, $C_{tree}(x, y, z)$
- Wind profile $v_w(z)$.
- Viewpoint $V(x_0, y_{0,} z_0)$, in 3D coordinates.

Computes:

Set of all visible points from $V(x_0, y_{0,} z_0$, $\sum_{i=1}^{N}[C_{building_i}, C_{car_i}, C_{tree_i}, C_{peds_i}]$.

We extend our previous work [2], developed for a fast and efficient visibility analysis for buildings in urban environments, and consider also a basic structure of cylinders, which allows us to model pedestrians and trees. Based on our probabilistic visibility computation of dynamic objects, we test the effect of these by using data gathered from web-oriented GIS sources to update our estimation and prediction on these entities.

B. Dynamic Objects – Modeling and Probabilistic Visibility

Dynamic objects such as moving cars and pedestrians, directly affect visibility in urban environments.

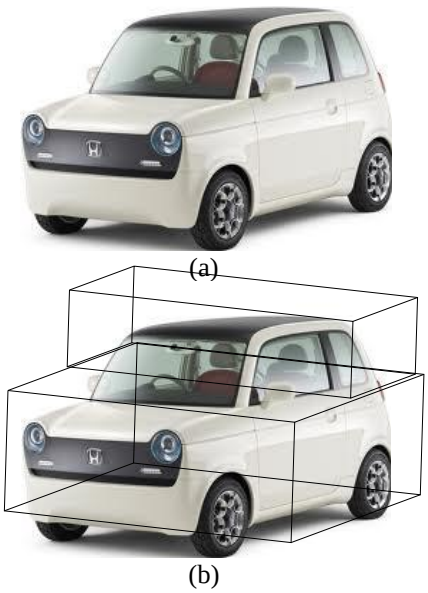
Due to modeling limitations, these entities are usually neglected in spatial analysis aspects. We focus on three major dynamic objects in an urban case: moving cars and pedestrians. Each object is modeled with 3D boxes or 3D cylinders, which allow us to extend the use of our previous visibility analysis in urban environments presented for static objects [2].

1) *Moving Car*

3D Modeling: As we mentioned earlier, web-cameras in urban environments can record the moving cars at any

specific time. Image sources such as web cameras, like other similar sensors sources, demand an additional stage of Automatic Target Detection (ATD) algorithms to extract these objects from the image [31]. In this research we do not focus on ATD, which must be implemented when shifting from the research described in the paper toward an applicable system.

The common car structure can be easily modeled by two 3D boxes, as can be seen in Fig. 1(b), which is similar to the original car structure presented in Fig. 1(a).



(a) (b)

Fig. 1. Car Modeling Using 3D Boxes: (a) the Original Car, (b) the Modeled Car

We define the Car Boundary Points (CBP) as the set of visible surfaces' boundary points of 3D boxes modeling the car presented in Fig. 1(b). Each box is modeled as 3D cubic $C_{car}(x, y, z)$ as presented extensively in [2] for a building model case:

$$C_{car}(x,y,z) = \begin{pmatrix} x = t \\ y = \begin{pmatrix} x^n - 1 \\ 1 - x^n \end{pmatrix} \\ z = c \end{pmatrix}$$

$$-1 \le t \le 1$$

$$n = 350$$

$$c = c + 1$$

Car Boundary Points (CBP) - we define CBP of the object i as a set of boundary points $j = 1..N_{CBP\_bound}$ of the visible surfaces of the car object, from viewpoint $V(x_0, y_0, z_0)$, where the maximum surface's number is six and each surface defined by four points, $N_{CBP\_bound} \le 24$.

In Fig. 2, the car is modeled by using two 3D boxes. Visible surfaces colored in red, CBP marked with yellow points.

$$CBP_{i=1..N_{CBP\_bound}}(x_0, y_0, z_0) = \begin{bmatrix} x_1, y_1, z_1 \\ x_2, y_2, z_2 \\ .. \\ x_{N_{CBP\_bound}}, y_{N_{CBP\_bound}}, z_{N_{CBP\_bound}} \end{bmatrix}$$

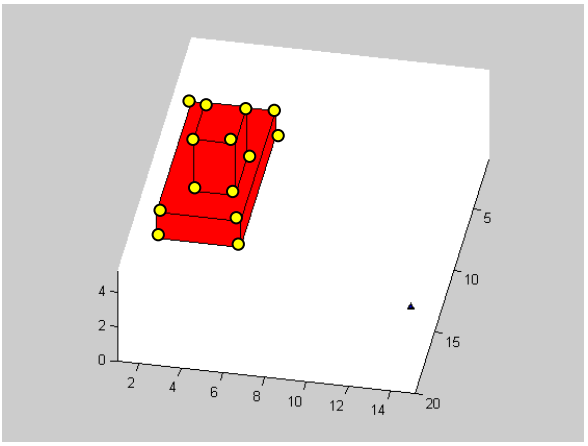


Fig. 2. Modeling Car Using 3D Boxes (CBP Marked with Yellow Points)

Probabilistic Visibility Analysis

Visibility has been treated as Boolean values. Due to incomplete information and the uncertainties of predicting the car's location at future times, visibility becomes much more complicated.

As it is well known from basic kinematics, CBP can be estimated in future time $t + \Delta t$ as:

$$CBP_i(t + \Delta t) = CBP_i(t) + V(t)\Delta t + \frac{A(t)\Delta t^2}{2}$$

Where $V(t)$ is the car velocity vector $V(t) = (v_x v_y)^T$, and the acceleration vector $A(t) = (a_x a_y)^T$. Estimation of a car's location in the future based on a web camera is not a simple task. Driver behavior generates multi-decision modeling, such as car-following behavior, gap acceptance behavior, or lane-change cases including traffic flow, speed etc. [32].

Our probabilistic car model is based on microscopic simulation models that were properly calibrated and validated using VISSIM simulation. VISSIM is a time-based microscopic simulation tool that uses various driver behaviors and vehicle performances to accurately represent

an urban traffic model. The VISSIM simulation model has been validated when compared to the data from various real-world situations [33] and used for the test-bed network by [34][35], and on driver behavior research defining average speed and acceleration [32].

The average speed in urban environments is about 45 [km/hr], from a minimum of 40 [km/hr] up to a maximum of 50 [km/hr]. In the situation of a free driving case, which is the common mode in urban environments [36], the acceleration of family car can change between 1 to 3.5 $\left[{}^{\mathrm{m}}/_{\mathrm{sec}^2}\right]$, and on average 2.5 $\left[{}^{\mathrm{m}}/_{\mathrm{sec}^2}\right]$, as seen in Fig. 3.

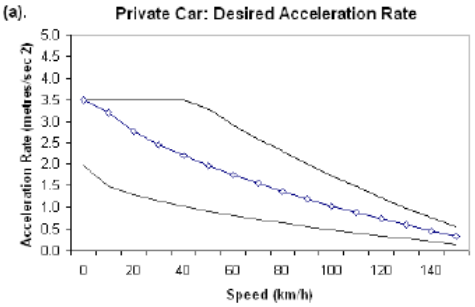


Fig. 3. Average Acceleration Rate of a Family Car in an Urban Environment [32]

As can be seen from several validations of car and driver estimation, velocity and acceleration are distributed as normal ones, and lead to normal location distribution:

$$\mathrm{V(t)}\sim\mathrm{N}(\mu = 45, \sigma^2 = 10)$$
$$\mathrm{A(t)}\sim\mathrm{N}(\mu = 2.5, \sigma^2 = 1)$$
$$\mathrm{CBP(t+\Delta t)}\sim \sum \mathrm{N}$$

In time step t, where the car's location is taken from a web-camera, visibility analysis from $\mathrm{CBP(t)}$is an exact one, based on our previous visibility analysis [2], as seen in Fig. 2 Visibility analysis becomes probabilistic for future time $\mathrm{t}+\Delta\mathrm{t}$, applying the same visibility analysis for $\mathrm{CBP(t+\Delta t)}$ presented in Fig. 4.

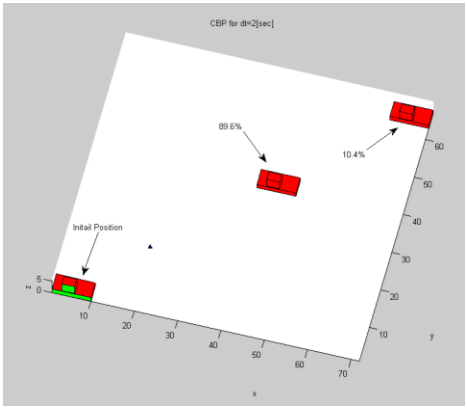


Fig. 4. Probabilistic Visibility Analysis for CBP

In Fig. 4, the car's location from a web-camera appears in the bottom left side. For $\Delta\mathrm{t} = 2[\mathrm{sec}]$, the car's location is marked by two 3D boxes, where CBP for each of them is the boundary of visible surfaces marked in red. The probability that the visible surfaces, which are bounded by CBP, will be visible in future time is based on the last update taken from the web application (depicted with arrows in Fig. 4, computed by using two different random normal PDF values for V and A based on eq. (4).

2) *Pedestrians*

3D Modeling: Pedestrian modeling can be done in high resolution, but due to ATD algorithms capabilities, pedestrians are usually bounded by a 3D cylinder and not as an exact detailed model [31]. For this reason, we model pedestrians as 3D cylinders, which is somewhat conservative but still applicable.

Pedestrian can be easily modeled by 3D cylinder, as seen in Fig. 5 (marked in red), which is similar to the output from ATD methods tested on a web-camera output recognizing walkers in urban environments.

We extend our previous visibility analysis concept [2] and include new objects modeled as cylinders as continuous curves parameterization$\mathrm{C_{Peds}(x,y,z)}$.

Cylinder parameterization can be described as:

$$C_{Peds}(x,y,z) = \begin{pmatrix} r\sin(\theta) \\ r\cos(\theta) \\ c \end{pmatrix}$$
$$0 \le \theta \le 2\pi$$
$$c = c+1$$
$$0 \le c \le h_{peds\_\max}$$

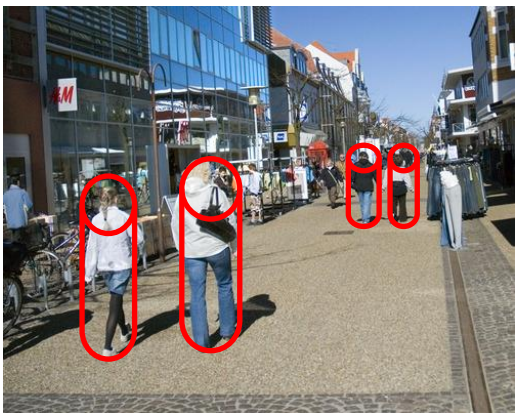


Fig. 5. Modeling Pedestrians in Urban Scene Using Cylinders (Colored in Red)

We define the visibility problem in a 3D environment for more complex objects as:

$$C'(x,y)_{z_{const}} \times (C(x,y)_{z_{const}} - V(x_0, y_0, z_0)) = 0$$

where 3D model parameterization is $C(x,y)_{z=const}$, and the viewpoint is given as $V(x_0, y_0, z_0)$. Extending the 3D cubic parameterization, we also consider the cylinder case. Integrating eq. (5) to (6) yields:

$$\begin{pmatrix} r\cos\theta \\ -r\sin\theta \\ 0 \end{pmatrix} \times \begin{pmatrix} r\sin\theta - V_x \\ r\cos\theta - V_y \\ c - V_z \end{pmatrix} = 0$$

$$\theta = \arctan\left(-\frac{-r - \frac{\left(-vy\, r + \sqrt{vx^4 - vx^2\, r^2 + vy^2\, vx^2}\right)}{vx^2 + vy^2}}{vx} \right.$$
$$\left. -\frac{-vy\, r + \sqrt{vx^4 - vx^2\, r^2 + vy^2\, vx^2}}{vx^2 + vy^2} \right)$$

As can be noted, these equations are not related to Z axis, and the visibility boundary points are the same for each x-y cylinder profile.

The visibility statement leads to complex equation, which does not appear to be a simple computational task. This equation can be efficiently solved by finding where the equation changes its sign and crosses zero value; we used analytic solution to speed up computation time and to avoid numeric approximations. We generate two values of θ generating two silhouette points in a very short time computation. Based on an analytic solution to the cylinder case, a fast and exact analytic solution can be found for the visibility problem from a viewpoint.

We define the solution presented in eq. (8) as x-y-z coordinates values for the cylinder case as **Pedestrian Boundary Points** (PBP). PBP are the set of visible silhouette points for a 3D cylinder modeling the pedestrian, as presented in Fig. 6:

$$PBP_{i=1..N_{PBP\_bound}=2}(x_0, y_0, z_0) = \begin{bmatrix} x_1, y_1, z_1 \\ x_{N_{PBP\_bound}}, y_{N_{PBP\_bound}}, z_{N_{PBP\_bound}} \end{bmatrix}$$

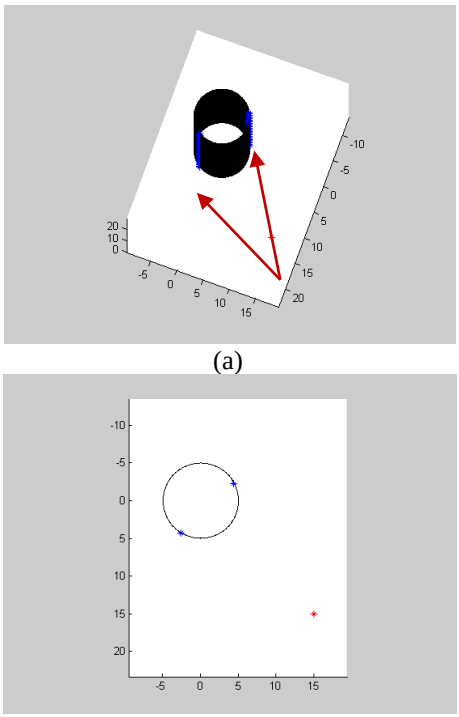


Fig. 6. PBP for a Cylinder using Analytic Solution marked as blue points, Viewpoint Marked in Red: (a) 3D View (Visible Boundaries Marked with Red Arrows); (b) Topside View

C. *Visibility Analysis Considering Sensor's Stochastic Character*

In this section, we extend our visibility model by exploring and including sensors' sensing capabilities and physical constraints. Our visibility analysis is based on the fact that sensors are located at specific visibility points. Sensors are commonly treated as deterministic detectors, where a target can only be detected or undetected. These simplistic sensing models are based on the disc model [37][38].

We study sensors' visibility-based placement effected by taking into account the stochastic character of target detection. We present a single sensor model, including noisy measurement, and define the necessary condition for visibility analysis with false alarm and detection probabilities for each visibility point's candidate.

1) *Single Visibility Sensing Model*

Most of the physical signals are based on energy vs. distance from single source model. Different kind of sensors such as: radars, lasers, acoustics, etc., are based on this signal character. Like other signal models presented in the literature [39][40][41] we use signal decay model as follows:

$$L(d) = \begin{cases} \frac{L_0}{(\frac{d}{d_0})^k}, \text{if } d > d_0 \\ L_0, \quad \text{if } d \le d_0 \end{cases}$$

where L_0 is the original energy emitted by the target, k is the decaying factor (typical values from 2 to 5), and d_0 is a constant determined by the size of the target and the sensor.

We model the sensor's noise N_i located at visibility point V_i, using zero-mean normal distribution, $N_i \sim N(0, \sigma^2)$. Sensor signal energy including noise effect, S_i, can be formulated as:

$$S_i = L(d_i) + N_i^2$$

In practice, S_i parameters are set by empiric datasets.

2) Necessary Condition for Visibility

Nowadays, detection systems use more and more data fusion methods [42][43]. In order to use multi sensors benefits, fusion and local decision-making using several sensors' data is a very common capability. As with other distributed data fusion methods, we assume that each sensor sends the energy measurement to a Local Decision Making Module (LDMM). Similar to other well known fusion methods [41], the LDMM integrates and compares the average sensors' measurements n against detection threshold τ.

Detection probability, denoted by P_D, is the probability that a target is correctly detected. Supposing that n sensors take part in the data fusion applied in the LDMM, detection probability is given by:

$$P_D = P(\frac{1}{n}\sum_{i=1}^{n}(L(d_i) + N_i^2) > \tau)$$

$$P_D = 1 - P(\sum_{i=1}^{n}(\frac{N_i}{\sigma})^2 \le \frac{n\tau - \sum_{i=1}^{n} L(d_i)}{\sigma^2})$$

$$P_D = 1 - X_n(\frac{n\tau - \sum_{i=1}^{n} L(d_i)}{\sigma^2})$$

Where $N_i/\sigma \sim N(0,1)$ and X_n denote the distribution function. In the same way, false alarm rate probability is the probability of making a positive detection decision when no target is present. False alarm rate probability, denoted by P_F, is given by:

$$P_F = P(\frac{1}{n}\sum_{i=1}^{n} N_i^2 > \tau) = 1 - P(\sum_{i=1}^{n}\left(\frac{N_i}{\sigma}\right)^2 \le \frac{n\tau}{\sigma^2})$$

$$P_F = 1 - X_n(\frac{n\tau}{\sigma^2})$$

Conditions Necessary for Visibility: *Given two real numbers,* $a \in (0,1)$ *and* $b \in (0,1)$. *Visibility Point* $V_i(x, y, z)$ *can be defined as visible point* ***if and only if*** $P_F(V_i) \le a$ *and* $P_D(V_i) \ge b$.

We integrate our unique concept of probabilistic visibility into the velocity space. We transform the visibility's boundaries from location to velocity space.

IV. VISIBILITY VELOCITY OBSTACLES (VVO)

The visibility velocity obstacle represents the set of all velocities from a viewpoint, occluded with other objects in the environment. It essentially maps static and moving objects into the robot's velocity space considering visibility boundaries.

The VVO of an object with circular visibility boundary points such as the pedestrians case, PBP, that is moving at a constant velocity v_b, is a cone in the velocity space at point A. In Fig. 7, the position space and velocity space of A are overlaid to illustrate the relationship between the two spaces. The VVO is generated by first constructing the Relative Velocity Cone (RVC) from A to the boundaries of the object, i.e., PBP, then translating RVC by v_b.

Each point in VVO represents a velocity vector that originates at A. Any velocity of A that penetrates VVO is a occluded velocity that based on the current situation, would result in a occlusion between A and the pedestrian at some future time. Fig. 7 shows two velocities of A: one that penetrates VVO, hence a occluded velocity, and one that does not. All velocities of A that are outside of VVO are visible from the current robot's position as the obstacle denotes as B, stays on its current course. The visibility velocity obstacle thus allows determining if a given velocity is occluded, and suggesting possible changes to this velocity for better visibility. If PBP is known to move along a curved trajectory or at varying speeds, it would be best represented by the nonlinear visibility velocity obstacle case discussed next.

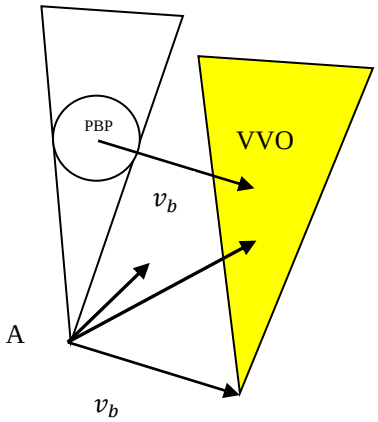


Fig. 7. Visibility Velocity Obstacles

The VVO consists of all velocities of A at t_0 predicting visibility's boundaries related to obstacles at the environment at any time $t>t_0$. Selecting a single velocity, v_a, at time $t = t_0$ outside the VVO, guarantees visibility to this specific obstacle at time t. It is constructed as a union of its temporal elements, VVO(t), which is the set of all absolute velocities of A, v_a, that would allow visibility at a specific time t.
Referring to Fig. 8, v_a that would result in occlusion with point p in B at time $t > t_0$, expressed in a frame centered at $A(t_0)$, is simply:

$$v_a = \frac{VBP_i}{t - t_0}$$

where r is the vector to point p in the blocker's fixed frame, and visibility boundaries denoted as Visibility Boundary Points (VBP). The set, VVO(t) of all absolute velocities of A that would result in occlusion with any point in B at time $t > t_0$ is thus:

$$VVO(t) = \frac{VBP_i(t)}{t - t_0}$$

Clearly, VVO(t) is a scaled B for two dimensional case with circular object, located at a distance from A that is inversely proportional to time t. The entire VVO is the union of its temporal subsets from t_0, the current time, to some set future time horizon t_h:

$$VVO(t) = \bigcup_{t=t_0}^{t_h} \frac{VBP_i(t)}{t - t_0}$$

The presented VVO generate a warped cone in a case of 2D circular object. If VBP(t) is bounded over $t = (t_0, \infty)$, then the apex of this cone is at $A(t_0)$.We extend our analysis to 3D general case, where the objects can be cubes, cylinders and circles. The mathematical analysis with visibility boundaries is based on VBP presented in the previous part for different kind of objects such as buildings, cars and pedestrians.

We transform the visibility's boundaries into the velocity space, by moving the VBP to the velocity space, in the same analysis presented for 2D circle boundary's.
Following that, we present 3D extension for VBP case, transformed to the velocity space.

Given two objects, VBP_1, VBP_2 will create a VVO representing VBP_2 (and vice-versa) such that VBP_1 wishes to choose a guaranteed collision-free velocity for the time interval τ, and visibility boundary in velocity space.

The Nonlinear Visibility Velocity Obstacle (NVVO) accounts for a general trajectory of the object, while assuming a constant velocity of the robot. It applies to the scenario shown in Fig. 8, where, at time t_0, a point A attempts to plan visible trajectory related an object, PBP, that is following a general known trajectory, c(t), and at time t_0 is located at $c(t_0)$. PBP represents the set of points that define the geometry of the visibility boundaries of the object, grown by the radius of the robot. In case of pedestrians where PBP is a circle, then c(t) represents the trajectory followed by its center.

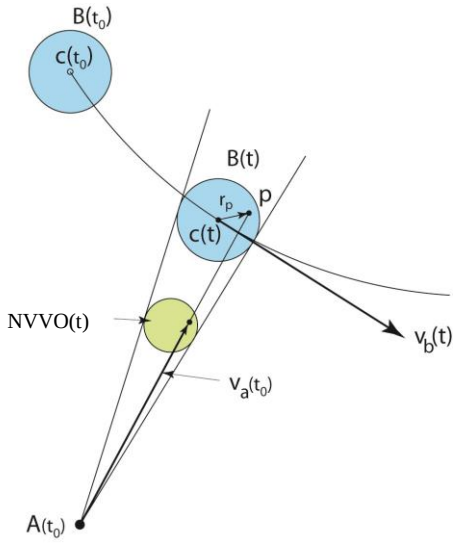


Fig. 8. Nonlinear Visibility Velocity Obstacles (NVVO), based on Nonlinear Velocity Obstacles (NLVO) (source [44])

Our method, based on visibility boundaries transformation from position to velocity space, can be formulated as homothetic transformation [44] that is centered at $A(t_0)$ and having the ratio $k = 1/(t - t_0)$:

$$v_a = H_{A(t_0),k}(c(t) + r), k = \frac{1}{t - t_0}.$$

The set, NVVO(t) of all absolute velocities of A that would result in occlusion with objects B at time $t> t_0$ is thus:

$$NVVO(t) = H_{A(t_0),k}(c(t) \oplus B), k = \frac{1}{t - t_0},$$

where $\oplus$ represents the Minkowski sum. Clearly, NVVO(t) is a scaled B, located at a distance from A that is inversely proportional to time t. To emphasize the geometric shape of the NVVO(t), we rewrite it as:

$$NVVO(t) = \frac{c(t)}{t - t_0} \oplus \frac{B}{t - t_0}$$

The entire NVVO is the union of its temporal subsets from t_0, the current time, to some set time horizon t_h:

$$NVVO = \bigcup_{t_0<t<t_h} \frac{c(t)}{t-t_0} \oplus \frac{B}{t-t_0}$$

Truncating the NVVO at t_h allows focusing the analysis till limited future time, time horizon. In case of cars, buildings and pedestrians where visibility boundaries can be expressed by geometric operations of 3D boxes, where VVO for the linear and NVVO for the non linear case analyzed in the same concept and formulation presented so far, as can be seen in Fig. 9.

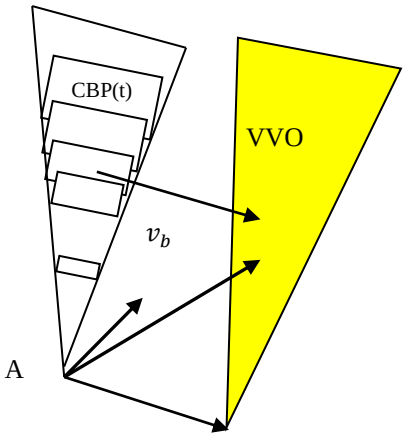


Fig. 9. Visibility Velocity Obstacle for visibility boundaries consist of 3D boxes

V. PURSUER PLANNER USING VVO

Our planner, similar to previous work [45] is a local one, generating one step ahead every time step reaching toward the goal, which is a depth first A* search over a tree. We extend previous planners, which take into account kinematic and dynamic constraints [9][14] and present a local planner for UAV as case study with these constraints, which for the first time generates fast and exact visible trajectories based on VVO, tracking after a target by choosing the optimal next action based on velocity estimation.

The fast and efficient visibility analysis of our method, allows us to generate the most visible trajectory from a start state q_{start} to the goal state q_{goal} in 3D urban environments, which can be extended to real performances in the future. We assume knowledge of the 3D urban environment model, and by using Visibility Velocity Obstacles (VVO) method to avoid occlusion, exploring maximum visible node in the next time step and track a specific target.

At each time step, the planner computes the next eighth Attainable Velocities (AV), as detailed in the next sub-section. The nodes, which are not occluded, i.e., nodes outside Visibility Velocity Obstacles, are explored. The planner computes the cost for these visible nodes and chooses the node with the optimal cost according to mission type. In our case, the optimal cost related to the node with minimum velocities difference between the robot and the tracked target.

1) Attainable Velocities

Based on the dynamic and kinematic constraints, UAVs velocities at the next time step are limited. At each time step during the trajectory planning, we map the AV, the velocities set at the next time step $t+\tau$, which generate the optimal trajectory, as is well-known from Dubins theory [28].

We denote the allowable controls as $u = (u_s, u_z, u_\phi)$ as U, where $V \in U$.

We denote the set of dynamic constraints bounding control's rate of change as $\dot{u} = (\dot{u}_s, \dot{u}_z, \dot{u}_\phi) \in U'$.

Considering the extremal controllers as part of the motion primitives of the trajectory cannot ensure time-optimal trajectory for Dubins airplane model [28], but is still a suitable heuristic based on time-optimal trajectories of Dubin - car and point mass models.

We calculate the next time step's feasible velocities $\tilde{U}(t+\tau)$, between $(t, t+\tau)$:

$$\tilde{U}(t+\tau) = U \cap \{u \mid u = u(t) \oplus \tau \cdot U'\}$$

Integrating $\tilde{U}(t+\tau)$ with UAV model yields the next eight possible nodes for the following combinations:

$$\tilde{U}(t+\tau) = \begin{pmatrix} \tilde{U}_s(t+\tau) \\ \tilde{U}_z(t+\tau) \\ \tilde{U}_\phi(t+\tau) \end{pmatrix} = \begin{pmatrix} u_s^{\min}, u_s(t) + a_s\tau \\ -u_s^{\max} \tan\phi^{\max}, u_s(t)\tan u_\phi(t) + u_s^{\max} \tan a_\phi \\ u_z^{\max}, u_z(t) - a_z\tau \end{pmatrix}$$

At each time step, we explore the next eight AV at the next time step as part of our tree search, as explained in the next sub-section.

2) Tree Search

Our planner uses a depth first A* search over a tree that expands over time to the goal. Each node $(q, \dot{q})$, where

$q = (x, y, z, \theta)$, consist of the current UAVs position and velocity at the current time step. At each state, the planner computes the set of AV, $\tilde{U}(t+\tau)$, from the current UAV velocity, $U(t)$, as shown in Fig. 10. We ensure the visibility of nodes by computing a set of Visibility Velocity Obstacles (VVO).

In Fig. 10, nodes inside VVO, marked in red, are occluded. Nodes out of VVO are further evaluated; visible nodes are colored in blue. The safe node with the lowest cost, which is the next most visible node, is explored in the next time step. This is repeated while generating the most visible trajectory, as discussed in the next sub-section.

Attainable velocities profile is similar to a trunked cake slice, as seen in Figure 10, due to the Dubins airplane model with one time step integration ahead. Simple models attainable velocities, such as point mass, create rectangular profile [4].

3) Cost Function

Our search is guided by minimum invisible parts from viewpoint V to the 3D urban environment model, with minimal difference between robot's velocity v_a and tracked target v_{tck}.

The cost function is computed for each visible node $(q, \dot{q}) \ni VVO$, i.e., node outside *VVO*, considering UAV velocities at the next time step:

$$w\big(q(t+\tau)\big) = abs(v_a(q(t+\tau) - v_{tck}(q(t+\tau))$$

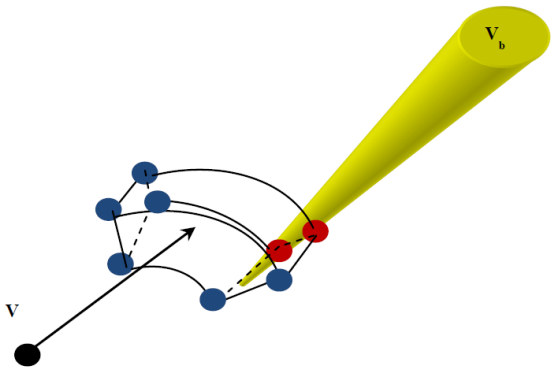


Fig. 10. Tree Search Method

4) Planner Pseudo-Code

The Pseudo-Code of the UAV Planner is as follows in Fig. 11:

$t = t_0$. $q_{best} = q_{start}$

1. While $(q_{best} \neq q_{goal})$ do:

1.1. Calculate AV nodes from q_{best},

$AV_{i=1}^{8} = \tilde{U}_{i=1}^{8}(t+\tau)$.

1.2. For each node $q_i \in AV$ check:

if $\dot{q}_i \in \cup_{j=1}^{n=N_{onj}} VVO_j$

q_i is illegal.

Else

Calculate node cost $w(q_i)$

1.3. If all nodes are illegal

STOP! No trajectory to the goal

Else

1.3.1. Find node with minimal cost

$q_{\min} = \{q_i \mid \min w(q_i)\}$.

1.3.2. Update $q_{best} = q_{\min}$

1.3.3. $t = t + dt$

End

Fig. 11. UAV Planner Pseudo-Code

VI. CONCLUSIONS

This paper proposes an online motion planning algorithm in 3D environments for tracking target, taking into account visibility analysis. The planner is based on local search and includes dynamic and kinematic constraints as complete part of the planner. Visibility boundaries, which are based on analytic solution for several kind of objects in 3D urban environments, also include uncertainty and probabilistic factors. Each VVO represents velocity's set of possible future collision and visibility boundaries. Based on our analysis in velocity space, we plan our trajectory by selecting future robot's velocity at each time step, tracking after specific target considering visibility constraints as integral part of the velocities space. We formulate the tracked target in the environment and include visibility analysis for the next time step as part of our planning in the same search space.

REFERENCES

[1] O. Gal and Y. Doytsher, "Motion Planning in 3D Environments Using Visibility Velocity Obstacles," in Proc. of the Tenth International Conference on Advanced Geographic Information Systems, Applications, and Services, Athens, Greece, pp: 60-65, 2018

[2] O. Gal and Y. Doytsher, "Fast and Accurate Visibility Computation in a 3D Urban Environment," in Proc. of the

Fourth International Conference on Advanced Geographic Information Systems, Applications, and Services, Valencia, Spain, pp: 105-110, 2012

[3] O. Gal and Y. Doytsher, "Fast Visibility Analysis in 3D Procedural Modeling Environments," in Proc. of the, 3rd International Conference on Computing for Geospatial Research and Applications, Washington DC, USA, 2012

[4] P. Fiorini and Z. Shiller, "Motion Planning in Dynamic Environments Using Velocity Obstacles," Int. J. Robot. Res.17, pp. 760–772, 1998

[5] Office of the Secretary of Defense, Unmanned Aerial Vehicles Roadmap, Tech. rep., December 2002

[6] J.C. Latombe, "Robot Motion Planning," Kluwer Academic Press, 1990

[7] M. Erdmann and T. Lozano-Perez, "On Multiple Moving Objects," Algorithmica, 2, 477–521, 1987

[8] T. Fraichard, "Trajectory Planning in a Dynamic Workspace: A 'State-Time Space' Approach," Advanced Robotics, 13:75–94, 1999

[9] S.M. LaValle and J. Kuffner, "Randomized Kinodynamic Planning," In Proc. IEEE Int. Conf. on Robotics and Automation, Detroit, MI , USA, pp: 473–479, 1999

[10] Z.H. Mao, E. Feron, and K. Bilimoria, "Stability and Performance of Intersecting Aircraft Flows Under Decentralized Conflict Avoidance Rules," IEEE Transactions on Intelligent Transportation Systems, 2: 101–109, 2001

[11] J. Bellingham, A. Richards, and J. How, "Receding Horizon Control of Autonomous Aerial Vehicles," in Proceedings of the IEEE American Control Conference, Anchorage, AK, USA, pp. 3741–3746, 2002

[12] B. Sinopoli, M. Micheli, G. Donata, and T. Koo, "Vision Based Navigation for an Unmanned Aerial Vehicle," in Proc. IEEE Int'l Conf. on Robotics and Automation, 2001

[13] J. Sasiadek and I. Duleba, "3D Local Trajectory Planner for UAV," Journal of Intelligent and Robotic Systems, 29: 191–210, 2000

[14] S.A. Bortoff, "Path Planning for UAVs," In Proc. of the American Control Conference, Chicago, IL, USA, pp: 364–368, 2000

[15] H. Plantinga and R. Dyer, "Visibility, Occlusion, and Aspect Graph," The International Journal of Computer Vision, 5,137-160, 1990

[16] Y. Doytsher and B. Shmutter, "Digital Elevation Model of Dead Ground," Symposium on Mapping and Geographic Information Systems (Commission IV of the International Society for Photogrammetry and Remote Sensing), Athens, Georgia, USA, 1994

[17] F. Durand, "3D Visibility: Analytical Study and Applications," PhD thesis, Universite Joseph Fourier, Grenoble, France, 1999

[18] W.R. Franklin, "Siting Observers on Terrain," in Proc. of 10th International Symposium on Spatial Data Handling. Springer-Verlag, pp. 109–120, 2002

[19] J. Wang, G.J. Robinson, and K. White, "A Fast Solution to Local Viewshed Computation Using Grid-based Digital Elevation Models," Photogrammetric Engineering & Remote Sensing, 62, 1157-1164, 1996

[20] J. Wang, G.J. Robinson, and K. White, "Generating Viewsheds without Using Sightlines," Photogrammetric Engineering & Remote Sensing, 66, 87-90, 2000

[21] C. Ratti, "The Lineage of Line: Space Syntax Parameters from the Analysis of Urban DEMs'," Environment and Planning B: Planning and Design, 32,547-566, 2005

[22] L. De Floriani and P. Magillo, "Visibility Algorithms on Triangulated Terrain Models," International Journal of Geographic Information Systems, 8, 13-41, 1994

[23] B. Nadler, G. Fibich, S. Lev-Yehudi, and D. Cohen-Or, "A Qualitative and Quantitative Visibility Analysis in Urban Scenes," Computers & Graphics, 5, 655-666, 1999

[24] S.M. LaValle, "Planning Algorithms," Cambridge,U.K.:Cambridge Univ. Pr., 2006

[25] M. Hwangbo, J. Kuffner, T. Kanade, "Efficient Two-phase 3D Motion Planning for Small Fixed-wing UAVs," In proceeding of: 2007 IEEE International Conference on Robotics and Automation, ICRA 2007, 10-14 April 2007, Roma, Italy

[26] http://www.asctec.de/uav-applications/research/products/asctec-hummingbird/

[27] A. Bhatia, M. Graziano, S. Karaman, R. Naldi, E. Frazzoli, "Dubins Trajectory Tracking using Commercial Off-The-Shelf Autopilots," AIAA Guidance, Navigation and Control Conference and Exhibit 18 - 21 August 2008, Honolulu, Hawaii.

[28] H. Chitsaz and S.M. LaValle, "Time-Optimal Paths for a Dubins Airplane," in Proc. IEEE Conf. Decision and Control., USA, pp. 2379–2384, 2007

[29] S. Zlatanova, A. Rahman, and S. Wenzhong, "Topology for 3D Spatial Objects," International Symposium and Exhibition on Geoinformation, pp. 22-24, 2002

[30] W.R. Franklin and C. Ray, "Higher isn't Necessarily Better: Visibility Algorithms and Experiments," In T. C. Waugh & R. G. Healey (Eds.), Advances in GIS Research: Sixth International Symposium on Spatial Data Handling, pp. 751–770. Taylor & Francis, Edinburgh, 1994

[31] Y. Song, "The research of a new Auto Target Recognition directed Image compression," in 3th Int. Congress on Image and Signal Processing (CISP), 16-18 Oct, China, 2010

[32] J. Archer, "Methods for the Assessment and Prediction of Traffic Safety at Urban Intersections and their Application in Micro-simulation Modeling," Centre for Traffic Simulation Research, CTR, Sweden. Technical Report, 2010

[33] M. Fellendorf and P. Vortisch, "Validation of the Microscopic Traffic Flow Model VISSIM in Different Real world Situations," 79th Annual meeting of Transportation Research Board, UK, 2001

[34] B. Park and J. D. Schneeberger, "Microscopic Simulation Model Calibration and Validation: Case Study of VISSIM Simulation Model for a Coordinated Actuated Signal System," Transportation Research Record 1856 , Paper No. 03-2531

[35] D. Parker and T. Lajunen, "Are Aggressive People Aggressive Drivers? A Study of the Relationship between Self-Reported General Aggressiveness Driver Anger and Aggressive Driving," Accident Analysis and Prevention, 33(2), 243-255, 2001

[36] R. Wiedemann and U. Reiter, "Microscopic Traffic Simulation: The Simulation System MISSION," Background and Actual State. Project ICARUS (V1052), Final Report, Brussels CEC.2: Appendix A, 1992

[37] K. Chakrabarty, S. Iyengar, H. Qi and E. Cho, "Grid Coverage for Surveillance and Target Location in Distributed Sensor Networks," IEEE Trans. Comput, vol. 51, no. 12, 2002

[38] D. Tian and N.D. Georganas, "A coverage-preserved node scheduling scheme for large wireless sensor networks," In WSNA, 2002

[39] M.F. Duarte and Y.H. Hu, "Vehicle classification in distributed sensor networks," Journal of Parallel and Distributed Computing, vol. 64, no. 7, 2004

[40] D. Li and Y.H. Hu, "Energy based collaborative source localization using acoustic micro-sensor array," EUROSIP J. Applied Signal Processing, vol. 4, 2003

[41] P. Varshney, "Distributed Detection and Data Fusion," Spinger-Verlag, 1996

[42] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan and K.K. Saluja, "Sensor deployment strategy for target detection," In WSNA, 2002

[43] T. Clouqueur, K.K. Saluja and P. Ramanathan, "Fault tolerance in collaborative sensor networks for target detection," IEEE Trans. Comput, vol. 53, no. 3, 2004

[44] Z. Shiller, R. Prasanna, J, Salinger, "A Unified Approach to Forward and Lane-Change Collision Warning for Driver Assistance and Situational Awareness," SAE Technical Paper 2008-01-0204, 2008, https://doi.org/10.4271/2008-01-0204

[45] O. Gal and Y. Doytsher. ”Patrolling Strategy Using Heterogeneous Multi Agents in Urban Environments Using Visibility Clustering”, Journal of Unmanned System Technology, ISSN 2287-7320, 2016