UNSYS digital

Journal of Unmanned System Technology

Trajectory Planning of Hovering Autonomous Underwater Vehicle in Complex Environments

Oren Gal

Department of Marine Technologies, Charney School of Marine Sciences, University of Haifa, Israel

Abstract—In this paper, we present trajectory planning method for Hovering Autonomous Underwater Vehicle (HAUV), based on Extended Kalman-Filter (EKF) process. The proposed planner is based on Rapid Random Trees (RRT) concept, generating visibility motion primitives in 3D underwater environments with obstacle avoidance capability. As far as we know, we present for the first time HAUV planner with visibility analysis for different kind of applications such as hull inspection where visibility analysis can be very crucial.

Keywords—Hovering Autonomous Underwater Vehicle, Extended Kalman-Filter, Rapid Random Trees.

DOI: 10.21535/just.v5i1.952

I. INTRODUCTION

TRAJECTORY Planning has developed alongside the increasing numbers of Autonomous Underwater Vehicles (AUV) and HAUV all over the world, with a wide range of applications such as surveillance, information gathering, suppression of enemy defenses, underwater mapping and hull inspection, etc. Most of these applications are involved in very complicated environments, with complex terrain for military domains [22]. With these growing needs, several basic capabilities must be achieved. One of these capabilities is the need to avoid obstacles or other moving objects, while autonomously navigating in 3D underwater 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 [17]. Path planning becomes trajectory planning when a time dimension is added for dynamic obstacles [5,18]. Later on, a vehicle's dynamic and kinematic constraints have been taken into account, in a process called kinodynamic planning [19]. All of these methods focus solely on obstacle avoidance.

Trajectory planning for air traffic control and ground vehicles has been well studied [21], based on short path algorithms using 2D polygons, 3D surfaces [1]. AUVs navigation has also been explored with vision-based methods [26], with local planning or a predefined global path [25]. AUV path planning is different from simple robot path planning, due to the fact that a AUV cannot stop, and must maintain its velocity above the minimum, as well as not being able to make sharp turns.

AUV path planning methods usually decompose the path planning into two steps: first, using some common path planning method in a polygonal environment [17], then, considering AUV dynamic and kinematic constraints into the trajectory [2]. These methods assume decoupling which affects the trajectory, as stated by all authors.

However, most of the effort focused on HAUV trajectory planning is related to obstacle avoidance with kinodynamic constraints, without taking into account visibility analysis.

The main challenge in motion planning 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 obstacles from which at least one other safe state is reachable.

Generally, we distinguish between local and global planners. The local planner generates one step, or a few steps, at every time step, whereas the global planner uses a global search toward 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 [4] and dynamic [5,6] environments.

Examples of local (reactive) planners are [7,8,9], 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 environment, and in narrow passages such as indoor environments. Recently, iterative planners [10,11,12,15,17,18], have been 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 also 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 to safe planning is to use braking policies [19]; another is to ensure local avoidance for a limited time [16]. However, neither considers the dynamic of the moving robot. A promising approach to safe motion planning in

Corresponding author: Oren Gal (e-mail: orengal@technion.ac.il)
This paper was submitted on May 29, 2017; revised on July 4, 2017; and accepted on August 7, 2017.

a dynamic environment is the consideration of ”Region of Inevitable Collision” (RIC) first introduced in [8] and later extended to Inevitable Collision States (ICS) in [9,23,27,28].

Efficient solutions for an approximated problem were investigated by LaValle and Kuffner, addressing non-holonomic constraints by using the Rapidly Random Trees (RRT) method [18,19]. Over the years, many other semi-randomized methods were proposed, using evolutionary programming [3,20,24]. The randomized sampling algorithms planner, such as RRT, explores the action space stochastically. The RRT algorithm is probabilistically complete, but not asymptotically optimal [13]. The RRT* planner [14] challenges optimality by a rewiring process each time a node is added to the tree. However, in cluttered environments, RRT* may behave poorly since it spends too much time deciding whether to rewire or not.

In this paper, we present, for the first time as far as know, a unique conceptual Spatial Trajectory Planning (STP) method based on RRT planner. The generated trajectories are based on visibility motion primitives set by SVC Optimal Control Points (OCP) as part of the planned trajectory, which takes into account exact 3D visible volumes analysis clustering that can be used in underwater environments.

The proposed planner includes obstacle avoidance capabilities, satisfying dynamics' and kinematics' agent model constraints in 3D environments, guaranteeing probabilistic completeness. The HAUV platform is a two-man-portable hovering AUV designed for ship hull inspection. Equipped with high-resolution imaging sonar, it surveys, by autonomous execution, ship hulls and other structures with minimal prior knowledge.

This platform is commonly involved in AUV systems but its control is usually relied on manual operations. Therefore, this diversity raises the need to establish autonomous abilities to the HAUV’s control and motion, and it can be achieved by forming equations that estimate its motion due to HAUV’s properties.

Estimating HAUV’s motion is based on EKF discretization- a process that inputs a start state vector of the HAUV's platform and updates it recursively at each step due to constant time periods during its motion. We present the RRT planner and EKF process with simulations and conclusion for our planner.

II. SPATIAL RAPID RANDOM TREES

In this section, the RRT path planning technique is briefly introduced with spatial extension. RRT was first introduced in [18,19], dealing with high-dimensional spaces by taking into account dynamic and static obstacles including dynamic and non-holonomic robots' constraints.

The main idea is to explore a portion of the space using sampling points in space, by incrementally adding new randomly selected nodes to the current tree's nodes.

RRTs have an (implicit) Voronoi bias that steers them towards yet unexplored regions of the space. However, in case of kinodynamic systems, the imperfection of the underlying metric can compromise such behavior. Typically, the metric relies on the Euclidean distance between points, which does not necessarily reflect the true cost-to-go between states. Finding a good metric is known to be a difficult problem. Simple heuristics can be designed to improve the choice of the tree state to be expanded and to improve the input selection mechanism without redefining a specific metric.

III. RAPID RANDOM TREES STAGES

The RRT method [19] is a randomized one, typically growing a tree search from the initial configuration to the goal, exploring the search space. These kinds of algorithms consist of three major steps:

a. Node Selection: An existing node on the tree is chosen as a location from which to extend a new branch. Selection of the existing node is based on probabilistic criteria such as metric distance.
b. Node Expansion: Local planning applied a generating feasible motion primitive from the current node to the next selected local goal node, which can be defined by a variety of characters.
c. Evaluation: The possible new branch is evaluated based on cost function criteria and feasible connectivity to existing branches.

These steps are iteratively repeated, commonly until the planner finds feasible trajectory from start to goal configurations, or other convergence criteria.

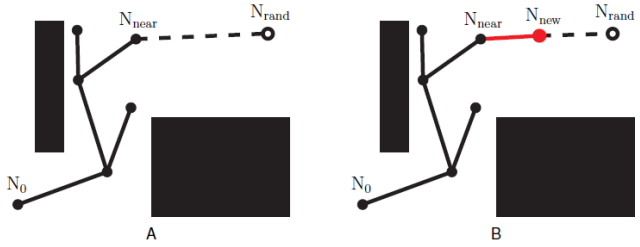

Figure 1 The RRT algorithm: (A) Sampling and node selection steps; (B) Expansion step

A simple case demonstrating the RRT process is shown in **Figure 1**. The sampling step selects Nrand, and the node selection step chooses the closest node, Nnear, as shown in **Figure 1.A**. The expansion step, creating a new branch to a new configuration, Nnew, is shown in **Figure 1.B**. An example for growing RRT algorithm is shown in **Figure 2**.

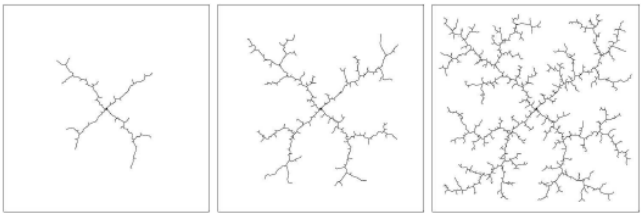

Figure 2 Example for growing RRT algorithm (source [19])

A. *Spatial RRT Formulation*

We formulate the RRT planner and revise the basic RRT planner [19] for a 3D underwater spatial analysis case for a continuous path from initial state xinit to goal state xgoal:

1. State Space: A topological space, X.
2. Boundary Values: $x_{init} \subset X$ and $x_{goal} \subset X$.
3. Free Space: A function $D: X \to \{true, false\}$ that determines whether $x(t) \subset X_{free}$ where X_{free} consist of the attainable states outside the obstacles in a 3D underwater environment.
4. Inputs: A set, *U*, contains the complete set of attainable control efforts u_i, that can affect the state.
5. Incremental Simulator: Given a current state, $x(t)$, and input over time interval Δt, compute $x(t+\Delta t)$.
6. 3D Spatial Analysis: A real value function, *f* (*x; u, OCP*$_i$) which specifies the cost to the center of 3D visibility volumes cluster points (OCP) between a pair of points in *X*

We present a revised RRT pseudo code described in Table 1, for spatial case generating trajectory *T*, applying *K* steps from initial state x_{init}. The *f* function defines the dynamic model and kinematic constraints, $\dot{x} = f(x; u, OCPi)$, where *u* is the input and OCP$_i$ set the next new state and the feasibility of following the next spatial visibility clustering point.

TABLE 1 SPATIAL RRT PSEUDO CODE

Generate Spatial RRT (x_{init}; *K*; Δt)
T.init (x_{init});
For *k* = 1 to *K* do
$x_{rand} \leftarrow$ *random.state*();
$x_{near} \leftarrow$ *nearest.neighbor* (x_{rand}; *T*);
u ← *select.input* (x_{rand}; x_{near});
$x_{new} \leftarrow$ *new.state* (x_{near}; *u*; Δt; *f*);
T.add.vertex (x_{new});
T.add.edge (x_{near}; x_{new}; *u*);
End
Return *T*

B. *Spatial Trajectory Planning (STP)*

In this section, we present a conceptual STP method based on RRT planner. The method generates visibility motion primitives in urban environments. The STP method is based on a RRT planner extending the stochastic search to specific OCP. These primitives connecting between nodes through OCP are defined as visibility primitives.

A common RRT planner is based on greedy approximation to a minimum spanning tree, without considering either path lengths from the initial state or following or getting close to specific OCP. Our STP planner consist of a tree's extension for the next time step with probability to goal and probability to waypoint, where trajectories can be set to follow adjacent points or through OCP. The planner includes obstacle avoidance capabilities, satisfying dynamics' and kinematics' agent model constraints in 3D environments. As we demonstrated in the previous section, the OCP are dynamic during daylight hours. Due to OCP's dynamic character, the generated trajectory is also a dynamic one during daylight hours.

We present our concept addressing the STP method formulating planner for a HAUV model, integrating OCP's as part of the generated trajectories along with obstacle avoidance capability.

IV. ESTIMATION AND MOTION PRIMITIVES

HAUV state vector can be described as

$$[u_t, v_t, \dot{\varphi}_t, x_t, y_t, \varphi_t]^T$$

is considered to inform about the HAUV's state at second t where: $u_t, v_t, \dot{\varphi}_t$ are sway velocity, heave velocity and roll rate of the HAUV, respectively and x_t, y_t, φ_t are horizontal, lateral position and angular orientation of the HAUV, respectively.

A waypoint is defined as same as last three parameters of a HAUV's state vector $[x_t, y_t, \varphi_t]^T$. A trajectory is defined as a series of a consecutive and nearby state vectors of the HAUV's vehicle.

A. *EKF discretization formulation*

As mentioned before, the process of the EKF is recursive and it's formulated by the following equation:

$$\begin{bmatrix} u_{k+1} \\ v_{k+1} \\ \dot{\varphi}_{k+1} \\ x_{k+1} \\ y_{k+1} \\ \varphi_{k+1} \end{bmatrix} = \begin{bmatrix} I_{3X3} & 0_{3X3} \\ I_{3X3}\Delta T & I_{3X3} \end{bmatrix} \begin{bmatrix} u_k \\ v_k \\ \dot{\varphi}_k \\ x_k \\ y_k \\ \varphi_k \end{bmatrix} + \begin{bmatrix} I_{3X3} \\ 0_{3X3} \end{bmatrix} \begin{bmatrix} w_{1,k} \\ w_{2,k} \\ w_{3,k} \end{bmatrix}$$

where ΔT is the constant time periods between two consecutive states, $\begin{bmatrix} I_{3X3} & 0_{3X3} \\ I_{3X3}\Delta T & I_{3X3} \end{bmatrix}$ is the movement matrix between the states, $\begin{bmatrix} w_{1,k} \\ w_{2,k} \\ w_{3,k} \end{bmatrix}$ is a zero mean Gaussian white noise vector used as a noise factor for the k-th step of the EKF's estimation.

In our case we define:

$$\Delta T = 1, p_t = \text{HAUV's state vector at second t}$$
$$p_{start} = p_0 = \text{start state of the HAUV}$$
$$F = \begin{bmatrix} I_{3X3} & 0_{3X3} \\ I_{3X3} & I_{3X3} \end{bmatrix}, \underline{w}_t = \begin{bmatrix} w_{1,t} \\ w_{2,t} \\ w_{3,t} \end{bmatrix},$$
$$\Gamma = \begin{bmatrix} I_{3X3} \\ 0_{3X3} \end{bmatrix}$$

Therefore, the EKF discretization is represented by the following equation:

$$p_t = \begin{cases} t = 0 & p_{start} \\ \text{else} & Fp_{t-1} + \Gamma \underline{w}_t \end{cases}$$

B. *EKF-The non-recursive formula*

By activating the EKF rule over and over again until reaching to the start state, we get a non-recursive version of the process:

$$p_t = \begin{cases} t = 0 & p_{start} \\ else & F^t p_{start} + \sum_{i=1}^{t} F^{t-i} \Gamma \underline{w}_i \end{cases}$$

Since for a non-negative integer h exists that:

$$F^h = \begin{bmatrix} I_{3X3} & 0_{3X3} \\ hI_{3X3} & I_{3X3} \end{bmatrix}$$

By matrix multiplication we get:

$$p_t = \begin{cases} t = 0 & p_{start} \\ else & \begin{bmatrix} I_{3X3} & 0_{3X3} \\ tI_{3X3} & I_{3X3} \end{bmatrix} p_{start} + \begin{bmatrix} \sum_{i=1}^{t} \underline{w}_i \\ \sum_{i=1}^{t} (t-i)\underline{w}_i \end{bmatrix} \end{cases}$$

and by defining $W_t = [\underline{w}_1, \underline{w}_2, \dots, \underline{w}_t]$ (a.k.a noise matrix) we get:

$$p_t = \begin{cases} t = 0 & p_{start} \\ else & \begin{bmatrix} I_{3X3} & 0_{3X3} \\ tI_{3X3} & I_{3X3} \end{bmatrix} p_{start} + \begin{bmatrix} W_t \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \\ W_t \begin{pmatrix} t-1 \\ \vdots \\ 0 \end{pmatrix} \end{bmatrix} \end{cases}$$

C. *Two-States Constraints*

Let k_1, k_2 be two consecutive waypoints that the HAUV's vehicle has to pass with time period of s seconds (when s is a positive integer). What are the suitable state values of the vehicle at waypoint k_1 in order to achieve this mission?

In order to solve this problem, we define p_{k_1}, p_{k_2} as the suitable states of the vehicle at k_1, k_2. Due to the non-recursive formula, it exists that:

$$p_{k_2} = \begin{bmatrix} I_{3X3} & 0_{3X3} \\ sI_{3X3} & I_{3X3} \end{bmatrix} p_{k_2} + N_s$$

$$N_s = \begin{bmatrix} W_t \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \\ W_t \begin{pmatrix} t-1 \\ \vdots \\ 0 \end{pmatrix} \end{bmatrix} = \begin{bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \\ N_5 \\ N_6 \end{bmatrix}$$

This leads to:

$$\begin{pmatrix} u_{k_2} \\ v_{k_2} \\ \dot{\varphi}_{k_2} \\ x_{k_2} \\ y_{k_2} \\ \varphi_{k_2} \end{pmatrix} = \begin{pmatrix} u_{k_1} + N_1 \\ v_{k_1} + N_2 \\ \dot{\varphi}_{k_1} + N_3 \\ s u_{k_1} + x_{k_1} + N_4 \\ s v_{k_1} + y_{k_1} + N_5 \\ s\dot{\varphi}_{k_1} + \varphi_{k_1} + N_6 \end{pmatrix}$$

According to these formulations, sway velocity, heave velocity and the roll rote of the HAUV's vehicle at k_1 can be calculated as:

$$u_{k_1} = \frac{x_{k_2} - x_{k_1} - N_4}{s}$$

$$v_{k_1} = \frac{y_{k_2} - y_{k_1} - N_5}{s}$$

$$\dot{\varphi}_{k_1} = \frac{\varphi_{k_2} - \varphi_{k_1} - N_6}{s}$$

Therefore, the state vector p_{k_1} equals to:

$$p_{k_1} = \begin{pmatrix} u_{k_1} \\ v_{k_1} \\ \dot{\varphi}_{k_1} \\ k_1 \end{pmatrix} = \begin{pmatrix} \frac{1}{s}(k_2 - k_1 - W_s \begin{pmatrix} s-1 \\ \vdots \\ 0 \end{pmatrix}) \\ k_1 \end{pmatrix}$$

These outcomes set HAUV trajectory which passes through a set of waypoints $\{b_i\}_{i=1}^{n}$ with integer time periods $\{s_i\}_{i=1}^{n-1}$, solved by the following algorithm:

For i=1…n-1, Create randomly $3Xs_i$ sized noise matrix W_{s_i} which each of its columns is a zero mean Gaussian white noise vector. This matrix will feature the noise at EKF processes along the algorithm progress.

Compute the state of the HAUV's vehicle at waypoint b_i using the formula above for the consecutive waypoints b_i, b_{i+1} and time period s_i:

$$p_{b_i} = \begin{pmatrix} \frac{1}{s_i}(b_{i+1} - b_i - W_{s_i} \begin{pmatrix} s_i - 1 \\ \vdots \\ 0 \end{pmatrix}) \\ b_i \end{pmatrix}$$

Activate the EKF discretization process over b_i

$$s_{end} = \begin{cases} s_i & i = n-1 \\ s_i - 1 & else \end{cases}$$

where s_{end} set the number of times in order to get the trajectory between b_i to b_{i+1}.

The EKF process can be described as:

$$\begin{cases} p_{start} = p_{b_i} = p_{b_i,0} \\ p_{b_i,m+1} = F p_{b_i,m} + \Gamma C_m(W_{s_i}) \\ (C_m(W_{s_i}) - \text{ column m of } W_{s_i}) \end{cases}$$

where trajectory itself:

$$Traj(b_i, b_{i+1}, s_i) = \{p_{b_i,m}\}_{m=0}^{s_{end}}$$

After getting all the trajectories between the waypoints unite them and get the requested trajectory, as can also be seen in **Figure 3**.

$$Traj(\{b_i\}_{i=1}^{n}, \{s_i\}_{i=1}^{n-1}) = \bigcup_{i=1}^{n-1} Traj(b_i, b_{i+1}, s_i)$$

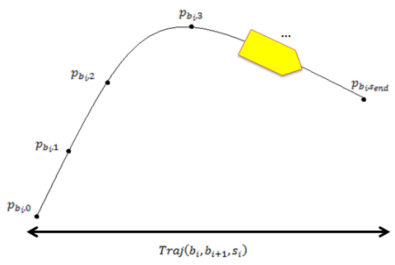

Figure 3 Trajectory states between waypoints using EKF

V. Search Method

Our search is guided by following spatial clustering points based on 3D visible volumes analysis in 3D urban environments, i.e., Optimal Control. The cost function for each next possible node (as the target node) consists of probability to closest *OCP*, P_{OCPi} , and probability to random point, P_{rand} .

In case of overlap between a selected node and obstacle in the environment, the selected node is discarded, and a new node is selected based on P_{OCPi} and P_{rand}. Setting the probabilities as P_{OCPi} =0.9 and P_{rand}=0.1, yield to the exploration behavior presented in **Figure 4**.

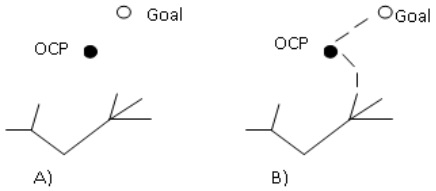

Figure 4 STP Search Method: (A) Start and Goal Points; (B) Explored Space to the Goal Through OCP

A. STP Planner Pseudo-Code

We present our STP planner pseudo code described in Table 2, for spatial case generating trajectory *T*. The search space is based on P_{OCPi} and P_{rand}. We apply *K* steps from initial state x_{init}. The *f* function defines the dynamic model and kinematic constraints, $\dot{x} = f(x; u)$, where *u* is the input and *OCPi* are local target points between start to goal states. Based on *f* function, one can assure that each node of the HAUV trajectory satisfying the dynamic and kinematic constraints.

Table 2 STP Planner pseudo code

STP Planner (x_{init}; x_{Goal} ;K; Δt; OCP)
  T.init (x_{init});
    x_{rand} ← *random.state*();
    x_{near} ← *nearest.neighbor*(x_{rand}; *T*);
    u ← *select.input*(x_{rand}; x_{near});
    x_{new} ← *new.state.OCP* (OCP_1; *u*; Δ*t*; *f*);
  While $x_{new} \neq x_{Goal}$ do
    x_{rand} ← *random.state*();
    x_{near} ← *nearest.neighbor*(x_{rand}; *T*);
    u ← *select.input*(x_{rand}; x_{near});
    x_{new} ← *new.state.OCP* (OCP_i; *u*; Δ*t*; *f*);
    T.add.vertex(x_{new});
    T.add.edge(x_{near}; x_{new}; *u*);
  end
  return *T;*

Function new.state.OCP (OCP_i;*u*; Δ*t*; *f*)
Set P_{OCPi} , *Set* P_{rand}
p←*uniform_rand[0..1]*
if $0 < p < P_{OCPi}$
  return $x_{new} = f(OCP_i,u,\Delta t)$;
else
if $P_{OCPi} < p < P_{rand} + P_{OCPi}$
then
  return RandomState();
end.

VI. Simulations

We have implemented the presented algorithm and tested some urban environments on a 1.8GHz Intel Core CPU with Matlab. In the first simulation planner avoided one obstacle reaching to the goal. In the second simulation presented in Figure 6, the planner generated trajectory avoiding four obstacle, taking into account EKF primitives and visibility analysis in all cases.

A. Completeness

Motion-planning and search algorithms commonly describe 'complete planner' as an algorithm that always provides a path planning from start to goal in bounded time. For random sampling algorithms, 'probabilistic complete planner' is defined as: if a solution exists, the planner will eventually find it by using random sampling. In the same manner, the deterministic sampling method (for example, grid-based search) defines completeness as resolution completeness.

Sampling-based planners, such as the STP planner, do not explicitly construct search space and the space's boundaries, but exploit tests with preventing collision with obstacles and, in our case, considering spatial considerations. Similarly, to other common RRT planners, which share similar properties with the STP planner, our planner can be classified as a probabilistic complete one.

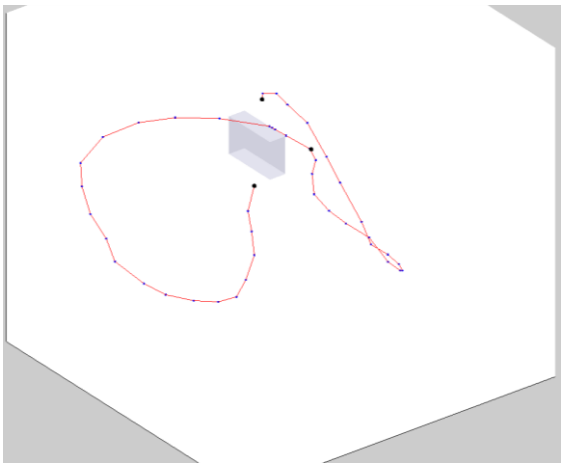

Figure 5 HAUV trajectory planning with one obstacle

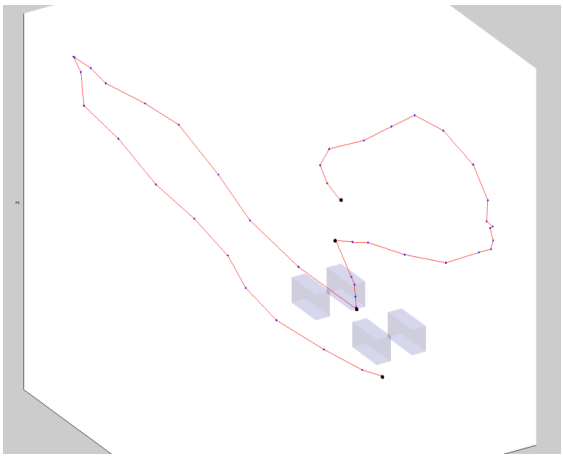

Figure 6 HAUV trajectory planning with four obstacles

VII. CONCLUSIONS

In this paper, we have presented a unique planner concept, STP, generating trajectory in 3D underwater environments for HAUV platforms. The planner takes into account obstacle avoidance capabilities and passes through optimal control points. The trajectory consists of EKF estimation waypoints between time constraints.

The unique RRT concept includes probabilistically complete properties. The planner allows us to generate trajectory for various applications in underwater domain taking into account visibility aspects.

REFERENCES

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

[2] Bortoff, S.A. "Path planning for UAVs," In Proc. of the American Control Conference, Chicago, IL, pp. 364–368, 2000. CrossRef

[3] Capozzi, B.J. and Vagners, J. "Navigating Annoying Environments Through Evolution," Proceedings of the 40th IEEE Conference on Decision and Control, University of Washington, Orlando, FL, 2001. CrossRef

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

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

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

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

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

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

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

[11] Fraichard, T. "Trajectory planning in a dynamic workspace: A 'state-time space' approach," Advanced Robotics, vol. 13, pp. 75–94, 1999. CrossRef

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

[13] Karaman, S. and Frazzoli, E. "Sampling-based algorithms for optimal motion planning," Int. J. Robot. Res., vol. 30, no. 7, pp. 846–894, 2011. CrossRef

[14] Karaman, S. Walter, M. Perez, A. Frazzoli, E. and Teller, S. "Anytime motion planning using the RRT*," in Proc. IEEE Int. Conf. Robot. Autom., Shanghai, pp. 1478–1483, May 2011.

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

[16] Latombe, J.-C. Kuffner, J. and Rock, S. "Randomized kinodynamic motion planning with moving obstacles", Algorithmics and Computational Robotics, vol. 4, pp. 247–264, 2000.

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

[18] LaValle, S.M. "Planning Algorithms," Cambridge,U.K.:Cambridge Univ. Press, 2006. CrossRef

[19] LaValle, S.M. "Rapidly-exploring random trees: A new tool for path planning," TR 98-11, Computer Science Dept., Iowa State University, 1998.

[20] Lum, W.C. Rysdyk, R.T. and Pongpunwattana, A. "Occupancy Based Map Searching Using Heterogeneous Teams of Autonomous Vehicles," Proceedings of the 2006 Guidance, Navigation, and Control Conference, Autonomous Flight Systems Laboratory, Keystone, CO, August 2006.

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

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

[23] Pandey, G. McBride, J.R. and Eustice, R.M. "Ford campus vision and lidar data set", International Journal of Robotics Research, 30(13):1543-1552, November 2011. CrossRef

[24] Pongpunwattana, A. and Rysdyk, R.T. "Real-Time Planning for Multiple Autonomous Vehicles in Dynamic Uncertain Environments," AIAA Journal of Aerospace Computing, Information, and Communication, pp. 580–604, 2004.

[25] Sasiadek, J. and Duleba, I., "Nonholonomic Motion Planning based on Newton Algorithm with Energy Optimization", IEEE Transaction Journal on Control Systems Technology, Vol. 11, No.3, May 2003, pp.355-363. CrossRef

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

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

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