PATH AND TRAJECTORY PLANNING FOR AUTONOMOUS UNDERWATER VEHICLES

Matko Barišić, Đula Nađ, Antonio Vasilijević, Nikola Mišković, *University of Zagreb, Faculty of Electrical Engineering and Computing, Unska 3, HR-10000 Zagreb, Croatia, {matko.barisic, dula.nad, antonio.vasilijevic, nikola.miskovic}@fer.hr*

ABSTRACT:
The paper details the advances made or currently being researched in the area of path and trajectory planning for marine vessels within the Laboratory for Underwater Systems and Technologies of the University of Zagreb, Faculty of Electrical Engineering. The paper outlines the differences in the definition of a path planning problem versus the trajectory planning problem. A novel algorithm for trajectory planning based on perfect a priori semantic knowledge of obstacles is presented.

Key words: path planning, trajectory planning, decentralized control functions, unmanned underwater vehicles

1. INTRODUCTION

Today, the unattended marine systems, a broad category consisting of various buoy-mounted systems, bottom-fixed collocated sensor measurement platforms and autonomous underwater vehicles of different configurations is moving towards maturity [1]. The most technologically advanced of these are the moving vessels – the autonomous underwater vehicles, since in addition to data-processing and telecommunication challenges, these must operate autonomous and robust motion control and motion decision algorithms. In order to resolve the complexity arising from the need to perform event-driven machine reasoning at the semantic level simultaneously with time-driven calculation and control at the syntactic level, AUVs are usually equipped with a hierarchical tiered structure of control algorithms [2,3], as shown in figure 1.

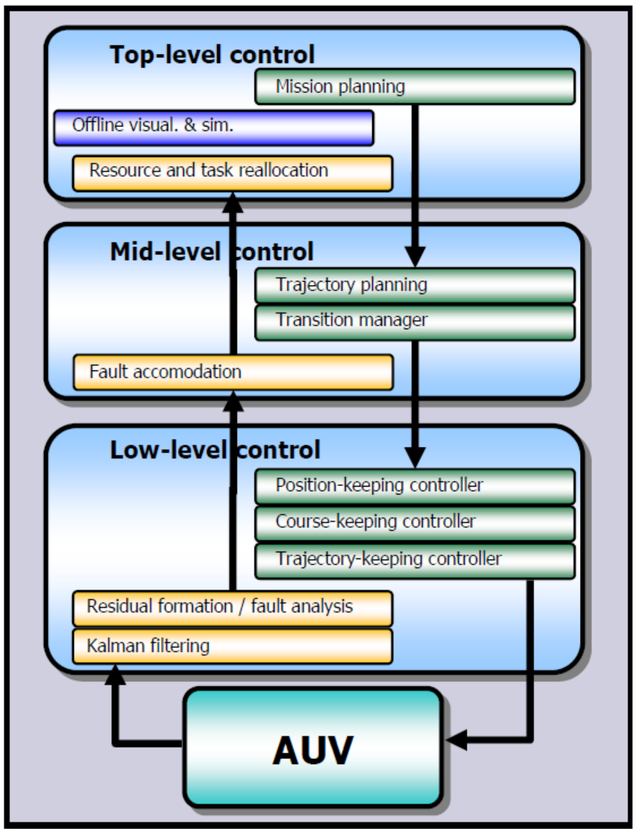

Figure 1: Hierarchy

The focus of this paper is the mid-tier, providing services of path and trajectory planning to an autonomous underwater vehicle. Chapter 2 gives a preliminary analysis of the mathematics involved and the definition of all terms. Chapter 3 develops from the defined terms and prerequisites assumed given an algorithm for autonomous path and trajectory planning in 2D, 3DOF[1] for a single AUV performing on the basis of perfect knowledge and perfect abstracting power. Chapter 4 concludes and presents on-going and future work.

2. PRELIMINARIES

In this section, a formal mathematical treatment of paths and trajectories is presented for the benefit of analysis in the following sections.

2.1. Paths

A path embedded in an N-dimensional space is, most generally, an N-dimensional real continuous single-valued vector function of the real parameter $\lambda \in [0,1]$, as denoted in (1).

$$\mathcal{P}(\lambda) = \begin{bmatrix} p_1(\lambda) & p_2(\lambda) & \dots & p_N(\lambda) \end{bmatrix}^{\mathrm{T}} \quad \lambda \in [0,1] \tag{1}$$

Since no further impositions are placed on the vector function, or individually on its functional elements, this specifies a broad class of possible paths. Amongst those is an

[1] Three degrees of freedom.

uncountable infinity of useless and unfeasible ones, in the physical sense. Also, note that every function element can itself be defined by piece-wise notation.
The solution of the vector function for the value of $\lambda = 0$ gives the initial point of the path, and the solution of the vector function for the value of $\lambda = 1$ the terminal point of the path. Since the vector function is continuous, there exists a finite length of the path. A graphical representation of a path is presented in figure 2.

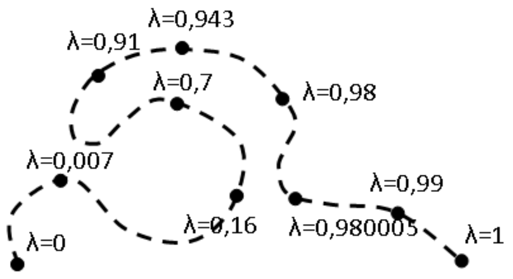

Figure 2. Path

The direction of the path is of some consideration for path-following control of an AUV. The direction is given by (2).

$$\vec{p}(\lambda) = \frac{\partial \mathcal{P}}{\partial \lambda} = \begin{bmatrix} \frac{\partial p_1}{\partial \lambda} & \frac{\partial p_2}{\partial \lambda} & \dots & \frac{\partial p_N}{\partial \lambda} \end{bmatrix}^{\mathrm{T}} \tag{2}$$

Since no prejudice is made to the differentiability of the function elements in (1), this definition is unsatisfactory for paths with cusps (i.e. points where there is a discontinuity in the path). In those cases, since AUVs have non-ideal finite dynamics and holonomic constraints, providing grounds for an expected lag between a command based on the direction of the path, and the AUV stabilizing on that path can be expected, therefore for all cusps, the direction can be redefined as in (3).

$$\forall \vec{c} \in \mathcal{P}([0,1]) \;\; \nexists \nabla \mathcal{P}(\vec{x})|_{\vec{x}=\vec{c}}, \quad \vec{p} \overset{\mathrm{def}}{=} \lim_{\vec{x} \to \vec{c}^+} \nabla \mathcal{P}(\vec{x}) \tag{3}$$

2.2. Piece-wise affine paths

The largest class of paths considered in AUV path planning, from the applicability point of view, are piece-wise affine (PWA) paths. These are paths for which certain other characteristics of the vector function “producing” the path on $\lambda \in [0,1]$. Namely, as denoted in (4), every element of the PWA path vector function can be expressed piece-wise, where each piece is an affine function of λ.

$$\forall i = 1 \dots N, \; \exists m_i \in \mathbb{N}, \; \exists \lambda_1^{(i)}, \lambda_2^{(i)}, \dots, \lambda_{m_i}^{(i)} \in (0,1),$$
$$p_i(\lambda) = \begin{cases} 0 \le \lambda < \lambda_1^{(i)} : & p_i(\lambda) = k_1^{(i)} \lambda + l^{(i)} \\ \lambda_1^{(i)} \le \lambda < \lambda_2^{(i)} : & p_i(\lambda) = k_2^{(i)} \lambda + k_1^{(i)} \lambda_1^{(i)} + l^{(i)} \\ \vdots & \vdots \\ \lambda_{m_i-1}^{(i)} \le \lambda < 1 : & p_i(\lambda) = k_{m_i}^{(i)} \lambda + \sum_{j=0}^{m_i-1} k_j^{(i)} \lambda_j^{(i)} + l^{(i)} \end{cases} \tag{4}$$

Every function element can be decomposed into the same number of pieces, m, and the condition for switching pieces can be expressed in terms of the intervals of values of λ, and the switching of every function element occurs for the same bounds of the intervals, as in (5).

$$\begin{gathered}\forall i = 1 \ldots N,\ j = 1 \ldots N,\ r = 1 \ldots m, \\ m_i \overset{\text{id}}{=} m_j = m,\ \lambda_r^{(i)} \overset{\text{id}}{=} \lambda_r^{(j)} = \lambda_r,\ \vec{k}_r = \begin{bmatrix} k_r^{(1)} & k_r^{(2)} & \ldots & k_r^{(N)} \end{bmatrix}^{\mathrm{T}},\ \vec{l} = \begin{bmatrix} l^{(1)} & l^{(2)} & \ldots & l^{(N)} \end{bmatrix}^{\mathrm{T}} \\ \mathcal{P}(\lambda) = \begin{cases} 0 \le \lambda < \lambda_1 : & \vec{k}_1\lambda + \vec{l} \\ \lambda_1 \le \lambda < \lambda_2 : & \vec{k}_2\lambda + \vec{k}_1\lambda_1 + \vec{l} \\ \vdots & \vdots \\ \lambda_{m-1} \le \lambda < 1 : & \vec{k}_m\lambda + \sum_{i=1}^{m-1} \vec{k}_i\lambda_i + \vec{l} \end{cases}\end{gathered} \tag{5}$$

The points in the N-dimensional water-space $\vec{x}_i = \mathcal{P}(\lambda_i)$ are then called *objective points*. Regarding directions of such paths, which are a necessary datum for line-following used to track them, they are defined in (6), and the linear vector equation of a directed line (for positive values of μ) in (7).

$$\forall i = 1 \ldots m,\ \vec{p}_i = \frac{\vec{k}_i}{\|\vec{k}_i\|},\ \vec{q}_i = \begin{cases} i = 1 : & \vec{l} \\ \text{otherwise} : & \sum_{j=1}^{i} \vec{k}_j\lambda_j + \vec{l} \end{cases} \tag{6}$$

$$\vec{x}_i = \vec{p}_i\mu + \vec{q}_i \tag{7}$$

A PWA path is presented in figure 4.a).

2.3. Trajectories

In addition to being defined as a path by (1), the trajectory is furnished with one-to-one and onto (injective), monotonously increasing mapping of the time variable t to the path length parameter λ, according to (8).

$$\mathcal{T}(\lambda, t) = (\mathcal{P}(\lambda),\ \lambda = \Lambda(t)) \tag{8}$$

In (8), the first member of the pair, $\mathcal{P}(\lambda)$ is the path of the trajectory, and the second member, $\Lambda(t)$, is the progress.

The ideal speed along a trajectory is a time derivative of the direction of $\mathcal{P}(\lambda)$, given by (9), and the ideal orientation can be obtained by (10).

$$\vec{v} = \left\| \nabla\mathcal{P}(\lambda) \cdot \frac{\partial\lambda}{\partial t} \right\| = \sqrt{\sum_{i=1}^{N} \left(\frac{\partial p_i(\lambda)}{\partial\lambda} \right)^2} \left| \frac{d\Lambda(t)}{dt} \right| \tag{9}$$

$$\begin{bmatrix}\varphi(t) & \vartheta(t)\end{bmatrix}^{\mathrm{T}} = \begin{bmatrix}\mathrm{atan}_2\left(\frac{\partial p_y}{\partial p_x}\right) & \mathrm{atan}_2\left(\frac{\partial p_z}{\sqrt{\partial p_x^2+\partial p_y^2}}\right)\end{bmatrix} = \begin{bmatrix}\mathrm{atan}_2\left(\frac{\frac{\partial p_y}{\partial\lambda}\cdot\frac{d\Lambda(t)}{dt}}{\frac{\partial p_x}{\partial\lambda}\cdot\frac{d\Lambda(t)}{dt}}\right) & \mathrm{atan}_2\left(\frac{\frac{\partial p_z}{\partial\lambda}\cdot\frac{d\Lambda(t)}{dt}}{\sqrt{\frac{\partial p_x}{\partial\lambda}^2+\frac{\partial p_y}{\partial\lambda}^2}\cdot\left|\frac{d\Lambda(t)}{dt}\right|}\right)\end{bmatrix} \tag{10}$$

An example of a trajectory is given in figure 3.

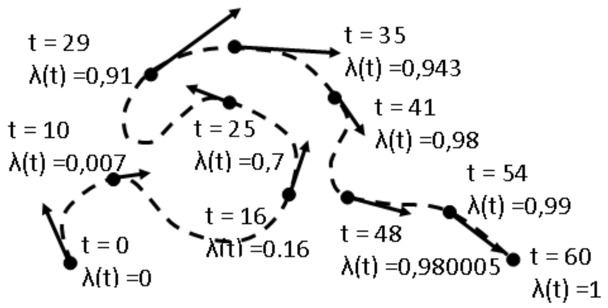

Figure 3. Trajectory

2.4. Piece-wise affine trajectories

Piece-wise affine trajectories are those trajectories arising from the furnishing of PWA paths with a piece-wise affine progress, i.e. a progress that is a linear spline interpolating on a set of points $\{(t_i, \lambda_i)\}$, $i = 1 \ldots m$ as per (4), (5).

From (4), (5) and (9) it then follows that the property of PWA trajectories is that the speeds v_i, $i = 1 \ldots m$ on directed line segments $\overrightarrow{\mathcal{P}(\lambda_i)\,\mathcal{P}(\lambda_{i+1})}$ are of constant norm.

A progress on a set PWA path can then be designed by working back from an m-tuple of speeds, by an algorithm in table 1.

Table 1. Algorithm of working out a progress of a PWA trajectory based on set speeds along PWA path legs

```
push onto T ← 0
push onto Λ ← 0
for i = 1: number of PWA path legs
    $l = \|\mathcal{P}(\lambda) - \mathcal{P}(\lambda_i)\|$
    push onto T ← $\mathrm{T} + l/v(i)$
    push onto Λ ← $\lambda_i$
end for
```

After performing the algorithm in table 1, the progress $\Lambda(t)$ is given by a linear spline between points (T_i,Λ_i) formed by members of stacks T and Λ filled by the algorithm.

PWA trajectories, of which an example is presented by figure 4.a,b) are by far the most common trajectories used as reference by AUVs. These are used almost exclusively with off-the-shelf state-of-the-art AUV technology used today to tackle most mission profiles listed in Section 2. The trajectories for such off-the-shelf systems are supplied by a human in the decision-making loop using a mission planning software.

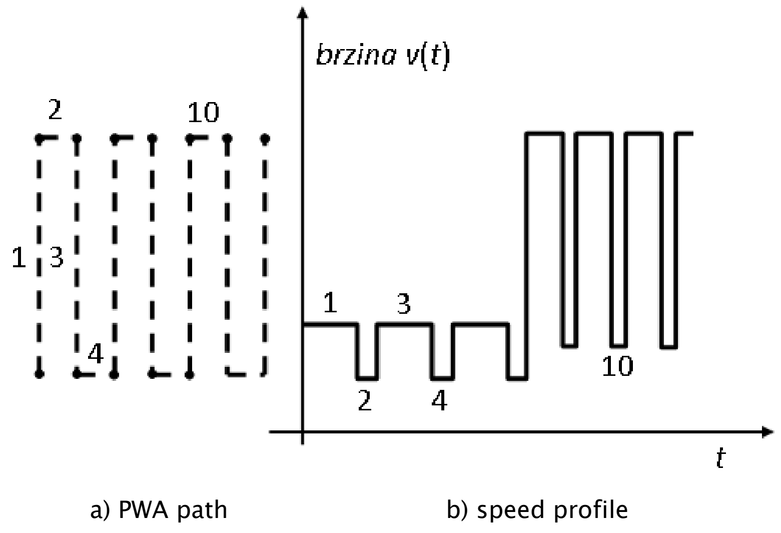

Figure 4. PWA trajectory consisting of a furnishing of a PWA path in a) with a velocity profile in b)

3. PATH AND TRAJECTORY PLANNING

When considering the problem of path planning which is far simpler than that of trajectory planning, and even constraining the planner to PWA paths, the problem formally becomes that of identifying safe, efficient and objective-fulfilling objective points in a possibly cluttered or unsafe water-space.

Even when this is performed by a human in pre-processing, or by a machine reasoning algorithm like in [4], representative of tier 1 in figure 1, the problem of path planning remains unsolved, Objective points might need to be interpolated between in order to avoid collisions or no-go areas (e.g. some critical vicinity of a tentatively mine-like object in autonomous mine identification and clearance missions).

Therefore, an algorithm for autonomous path interpolation is proposed in table 2. Prior to performing the off-line algorithm, all obstacles need to be extended by a safety-radius at least equal (or greater than) the radius of the AUV-circumscribing circle, as presented in figure 5. For simplicity purposes, rectangular obstacles are extended using ∞ – norm, remaining rectangular rather than being furnished with corners beveled by circular sections.

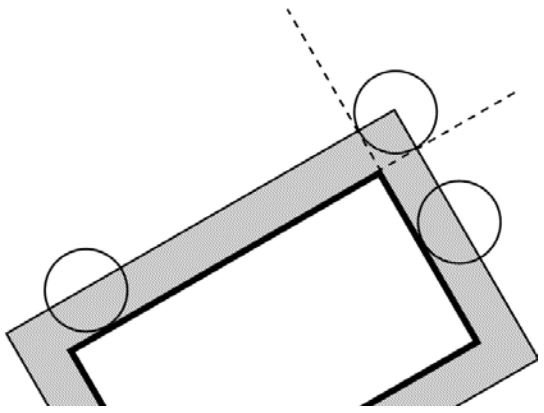

Figure 5. The extension of obstacles by a minimal safety-radius

Table 2. Algorithm for PWA path planning by addition of objective points circumnavigating obstacles

```
for i = 1 to m
  set initial point to $x_b = x_{i-1}$
  set final point to $x_e = x_i$
  push onto P ← $x_b$
  $x_c = x_b$
  while $x_c \neq$x_e
    S = Ø
    if $x_c$ on rectangle
      if $x_c$ isn't a vertex
        S = {vertices of current rectangle}
      end if
      $x_r = x_c$
    else
      $x_r$ = center of circular or elliptic obstacle
    end if
    S = S ∪ obstacles ∪ $x_e$
    $x_n = \mathrm{argmin}_x\, d(x_r,S)$
*   if $x_c$ not on rectangle
      $x_3$ = intersection of obstacle boundary
            and ray from obstacle center to $x_n$,
      $\alpha$ = mean between azimuths of $x_1$ and $x_2$
            w.r.t. the obstacle center
      $x_0$ = intersection of obstacle boundary
            and ray from obstacle center at
            azimuth $\alpha$
      $k_1$ = slope of tangent to boundary at $x_c$
      $k_0$ = slope of tangent to boundary at $x_0$
      $k_2$ = slope of tangent to boundary at $x_3$
      $x_1$ = intersection of $[1;k_1]\lambda+x_c$ and
            $[1;k_0]\mu+x_0$
      x_2 = intersection of $[1;k_0]\lambda+x_0$ and
            $[1;k_2]\mu+x_3$
      push onto P ← $x_c$, $x_1$, $x_2$, $x_3$
    end if
    push onto P ← $x_n$
    $x_c = x_n$
  end while
† i = length(P)
  while i>=2
    j = i-1
    while j>=1
      if P(j) visible from P(i)
        del = j+1
      end if
      j = j-1
    end while
    delete from P elements from i-1 down to
      del inclusive
    i = del
  end while
end for
```

Note that the lines under the asterix-marked if-clause of the algorithm assure circumnavigation of a circular or elliptical obstacle as presented in figure 6.

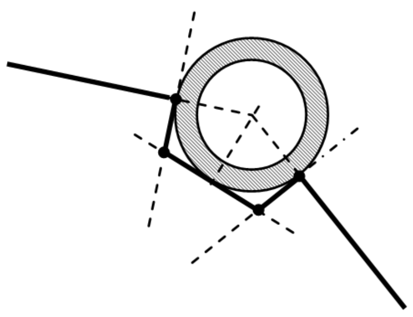

Figure 6. Circumnavigation of elliptical and circular obstacles

The lines in the two nested while-loops starting with the dagger-marked line in table 2 assure that the list of objective points is pruned of spurious objective-points that have arisen due to locally optimal, but globally sub-optimal path planning in congested areas, as presented by dotted lines in figure 7. The finished planned path is presented in the same figure using a solid line.

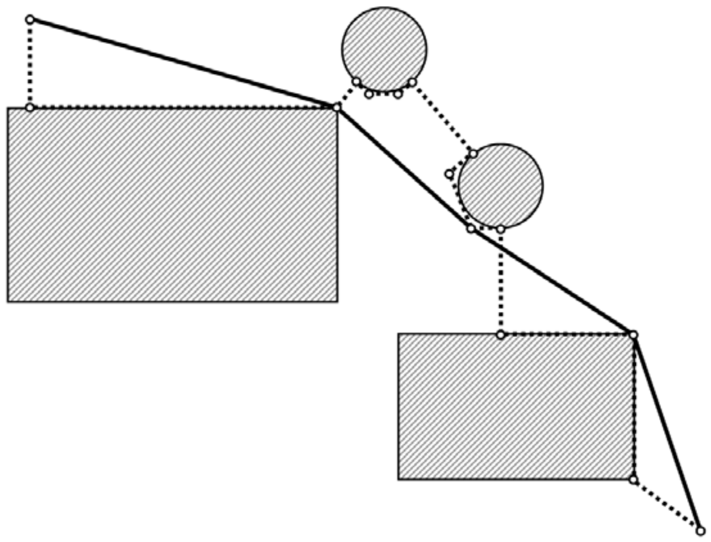

Figure 7. Pruning of spurious objective points in locally optimally traversed congested sub-water-spaces

There are many possibilities of turning the PWA path produced by a perfectly a priori known configuration of the water-space (obstacles and a priori objective points) into a trajectory. The trivial way is to furnish the PWA path with a PWA progress built from a sequence of pre-defined speeds along path legs, through the algorithm in table 1. Another algorithm, presented originally in [5], minimizes the wear-and-tear of the AUV's actuators, while enabling traversal of the congested water-space with the optimal speed. The algorithm contains two steps. The first is the complete algorithm presented originally in [5], using a set of objective points to plan a path through all of them that is not piecewise affine in the Euclidean water-space, but in terms of curvature[2] and sharpness[3]. Such a path will formally consist of in general $3n$ segments for n objective points. Depending on the spatial distribution of objective points, the finally planned path may consist of mathematically identically formulated segments that are smoothly joined in the λ parameter, effectively consisting of less than $3n$ actual distinguishable segments. Thus designed curves of constant sharpness in Euclidean 2-space are called *Clothoids* or *Euler spirals*, obeying (11 - 12).

$$x(L) = \int_0^L \cos l^2 \, dl \tag{11}$$

$$y(L) = \int_0^L \sin l^2 \, dl \tag{12}$$

Where L is the length along the *normalized* Clothoid. The above integral, called *Fresnel integrals* are not analytically solvable. A numerical approximation must be accepted, e.g. a Taylor series expansion to finitely many terms, or an approach using successive over-relaxation or other numerical mathematical technique. The second step of the algorithm

[2] Curvature is the differential of the tangent angle of the curve w.r.t. the length of the curve.

[3] Sharpness is the differential of the curvature w.r.t. the length of the curve, i.e. the second differential of the tangent angle w.r.t. the length of the curve.

is to assign to the clothoidal segments such progress (in effect – such speed through water) that angular and linear accelerations are maximized subject to (17), as follows from the analysis in (13 – 16).

$$\begin{aligned}\varphi(L) &= sL + c\\ \omega &= \frac{d\varphi}{dt}\\ &= \frac{\partial\varphi(L)}{\partial L}\frac{dL}{dt}\\ &= s\cdot v\end{aligned} \tag{13}$$

$$\begin{aligned}\alpha &= \frac{d^2\varphi}{dt^2} = \frac{d}{dt}\frac{d\varphi}{dt} = \frac{d}{dt}\left(\frac{\partial\varphi}{\partial t} + \frac{\partial\varphi}{\partial L}\frac{dL}{dt}\right)\\ &= \frac{d}{dt}\left(0 + s\cdot v\right)\\ &= sa\end{aligned} \tag{14}$$

$$\begin{aligned}E_k &= \tfrac{1}{2}mv^2 + \tfrac{1}{2}I\omega^2\\ &= \tfrac{1}{2}v^2(m + Is)\end{aligned} \tag{15}$$

$$\begin{aligned}P &= \tfrac{d}{dt}E_k\\ &= va(m + Is)\end{aligned} \tag{16}$$

$$\text{Maximize } a,\ \alpha \text{ subject to}: \begin{cases}(1): & \max E_k = \max \tfrac{1}{2}v^2(m+Is) = \overline{E_k}\\ (2): & \max P = \max va(m+Is) = \overline{P}\\ (3): & \max a = \overline{a}\\ (4): & \max v = \overline{v}\\ (5): & \max \omega = \overline{\omega}\\ (6): & \max \alpha = \overline{\alpha}\end{cases} \tag{17}$$

4. CONCLUSION

The paper has presented an overview of the theory of path and trajectory-planning, with visualization in 2D to facilitate intuitive understanding of results. Economic conside-rations of professional fields that stand to gain in expediency, efficiency, lowering of costs has led to the clear statement of motivations for the introduction of autonomous machine capability of path- and trajectory-planning. Definition of terms leading to clear problem statement allowing for diversified approaches to machine path- and trajectory-planning was presented. An algorithm was presented that breaks the task of planning a trajectory down into the task of planning a piecewise affine path through a set of objective points, modifying the lines in between the objective points into triplets of clothoid segments, and then assigning piece-wise affine signals for reference angular and linear speed along such a segmented clothoid path. The algorithm will be further augmented and employed in the spring-summer season of 2010 to prove collision avoidance and suitability of planned trajectories to necessary tasks in the example of marine biology and maritime archaeology surveys. Path- and trajectory-planning capabilities will be tested on the OceanServer Technologies Inc. Iver 2 autonomous underwater vehicle. They will be realized in the MOOS framework that has already been used by the authors for various autonomous control tasks [6]. To facilitate stable trajectory following, hierarchically lower control of the degrees of freedom of the vehicle (in the sense of figure 1), [2] will be included in the library of MOOS modules carried aboard the Iver during the missions.

REFERENCES

1. Bildberg, R.: *Foreword*. In Bildberg. R: Proceedings of the 16th International Symposium on Unmanned Untethered Submersibles Technology, Durham, NH, USA, August 2009, on CD, Autonomous Undersea Systems Institute, Durham, NH, USA
2. Barišić, Matko, Vukić, Zoran and Mišković, Nikola: *Design Of A Coordinated Control System For Marine Vehicles*, In Pascoal, A.: Proceedings of the 7th IFAC Conference on Manoeuvring and Control of Marine Craft, on CD, Lisabon, 2006, Institute for Systems and Robotics, Instituto Superior Tecnico, Lisabon, Portugal.
3. Mišković, Nikola, Bibuli, Marco, Bruzzone, Gabriele, Caccia, Massimo, and Vukić, Zoran: *Heuristic Parameter Tuning Procedures for a Virtual Potential Based AUV Trajectory Planner*. In Decio, C. D.: Proceedings of the 8th IFAC Conference on Manoeuvring and Control of Marine Craft, on CD, Gurauja, 2009, on CD, Escola Politecnica da Universidade de Sao Paolo, Sao Paolo, Brazil
4. Fiorelli, E. et al., *Multi-AUV Control and Adaptive Sampling in Monterey Bay*, In: Proceedings of the IEEE Autonomous Underwater Vehicles 2004: Workshop on Multiple AUV Operations, pp. 134-147.
5. Shin, D. H. and Singh, S.: *Path Generation For Robot Vehicles Using Composite Clothoid Segments*. © 1990, Carnegie-Mellon University, Pittsburgh, Pennsylvania 15213, USA
6. Barišić, M., Kragelund, S., Masek, T. D., and Vukić, Z.: *An Online AUV Trajectory Re-planning Software Architecture Based on the MOOS*. In Bildberg, R.: Proceedings of the 16th International Symposium on Unmanned Untethered Submersible Technology, Durham, NH, USA, August 2009, on CD, Autonomous Undersea Systems Institute, Durham, NH, USA