Abstract—Over the past few years we have witnessed an increasing use of multiple autonomous robotic vehicles in mission scenarios with increasing complexity and associated risks. The use of multiple vehicles provides a paradigm shift which involves using, instead of a large monolithic vehicle, a set of simpler vehicles, further enhancing flexibility and robustness against failures, while potentially lowering operation costs. For these reasons, there is great interest in developing efficient motion planning algorithms for cooperative missions.

This thesis aims to contribute to the development of a new generation of systems for mission planning that incorporates in its formulation the dynamics of the vehicles. The first part describes the development of a trajectory generator for a single vehicle. For this, we make use of the so-called Projection Operator Approach, a mathematical tool proposed by Professor John Hauser of the Univ. Colorado, USA to solve optimal control problems. The generator synthesizes trajectories that minimize time or energy usage, avoiding, at the same time, collisions with fixed obstacles.

The second part of the thesis focuses on the extension of the development algorithm to the motion planning of multiple vehicles. We consider the case in which a group of vehicles is expected to reach a target position simultaneously, while minimizing expended energy and avoiding collisions with each other. Further, we consider the important case in which the vehicles need to move in formation.

The overall system performance is assessed via computational simulations with realistic models of the Medusa marine vehicle.

Keywords—Trajectory Generation, Collision Avoidance, Time-Minimal Optimization, Energy-Minimal Optimization, Projection Operator

I. INTRODUCTION

The past few years have witnessed an increasing in the design, development, and operation of remotely operated and autonomous vehicles. These vehicles are highly mobile robots that can reach otherwise inaccessible places and perform tasks that are too hard or too dangerous for human beings to perform. They also have the potential to reduce the man-power and cost required to perform increasingly challenging scientific and commercial operations at sea. As the costs needed to produce autonomous vehicles have decreased significantly, the number of missions in which more than one vehicle is used has increased. This allows them not only to carry out more complex tasks than a single robot can handle, but also increases the efficacy and efficiency of the missions.

An important part in the success of any mission is motion planning, which consists of evaluating a collision free trajectory from an initial state to a desired final state taking into account geometrical, physical, and temporal constraints. The planner complexity increases in multiple vehicle mission scenarios, where the vehicles need to cooperate with each other, while at the same time avoiding collisions with one another.

The range of applications where single and multiple autonomous vehicles are used is very diversified. A representative example, is the use of unmanned aerial vehicles to detect and prevent fires[1]. The objectives are to determine the position of potential fire alarms and also to reduce the number of false alarms by means of cooperation.

Another example is the Nasa’s Mars Exploration Rover, Curiosity, that landed on Mars in 2012. This vehicle can analyse images taken during a drive to compute a safe driving path. This will enable the rover to cover the remaining ground of Mars, giving crucial information about the geological features of that planet.

Another field where multiple autonomous vehicles play an important role is ocean exploration. Exploring the oceans can be costly, hard and potentially dangerous, so the best option is to deploy a large number of unmanned autonomous vehicles.

An example of a multiple vehicle cooperation project using AUVs is Cooperative Cognitive Control for Autonomous Underwater Vehicles[2]. In this project, a number of autonomous surface craft are required to maintain a desired formation to guide a human diver. For this mission’s objective to be completed is not only necessary to have a reliable motion controller, but also a good mission planner.

A. General Statement

The general problem of motion planning can be divided in three different aspects: path planning, manoeuvre planning, and trajectory generation. In this thesis we will focus on the problem of generating trajectories that are feasible in terms of the vehicle’s dynamic model. Automatic trajectory generation is one of the most important functions which an autonomous vehicle must perform, since it must be able to move in his workspace whilst avoiding obstacles and reaching specific goal configurations.

New methods have been proposed for this problem, which include differential geometric and differential algebra techniques, control input parametrization, and optimal control approaches ([3]).

In this work, we adopt a control theoretic approach that builds on optimal control theory and yields trajectories for missions with single and multiple vehicles. To this effect, we borrow from and extend the formalism described in [4] and [5] that relies on the so-called Projection Operator approach for the optimization of dynamic systems. We consider problems
whereby a fleet of vehicles must manoeuvre from their initial to target positions and arrive at their destinations simultaneously, while minimizing the time to travel or energy expenditure. In contrast to what is normally reported in the literature, we include in the planning algorithm the explicit computation of the actual electrical energy consumed by the thrusters that power the Medusa vehicles considered in this thesis. We further consider the case where the vehicles are required to move while holding a desired geometrical formation is also proposed.

The work is based in a dynamical model for the Medusa vehicle but can also be extended to other types of vehicles.

C. Outline

The paper is structured as follows: Section II presents the model of the vehicles used throughout the work, the Medusa class of autonomous marine vehicles. Section III introduces the mathematical framework used to solve optimal control problems. Section IV describes the type of optimal control problems that will be solved in the simulations. Section V describes the implementation details of multiple vehicles mission scenarios. Section VI presents and comments the results of several simulations made to test the proposed implementation. Section VII concludes the work with a summary of the results obtained and some topics of future work.

II. AUTONOMOUS UNDERWATER VEHICLE MODEL

To obtain the model equations it is common practice to define two coordinate frames: an inertial reference frame \( \{ I \} \), composed by the orthonormal axes \( \{ x_I, y_I, z_I \} \) and a body-fixed reference frame \( \{ B \} \), composed by the orthonormal axes \( \{ x_B, y_B, z_B \} \).

Using the notation in [6], the kinematic equations of motion of a vehicle moving in the horizontal plane are

\[
\begin{align*}
\dot{x} &= u \cos \psi - v \sin \psi \\
\dot{y} &= u \sin \psi + v \cos \psi \\
\dot{\psi} &= r
\end{align*}
\]

which can be written in the more compact form

\[
\dot{p} = R(\psi) \nu
\]

where \( p = [x, y]^T \), \( \nu = [u, v]^T \) and \( R(\psi) \) is the transformation matrix from \( \{ R \} \) to \( \{ I \} \).

Neglecting roll, pitch and heave rates, the equations for surge, sway, and heading of an underactuated vehicle become

\[
\begin{align*}
m_u \dot{u} - m_v vr + d_u u &= \tau_u \\
m_v \dot{v} - m_u ur + d_v v &= 0 \\
m_r \dot{r} - m_u vr + d_r r &= \tau_r
\end{align*}
\]

where \( m_u = m - X_u, m_v = m - Y_v, m_r = I_r - N_r \) and \( m_{uv} = m_u - m_v \) are mass and hydrodynamic added mass terms, and \( d_u = -X_u - X[|u|v], d_v = -Y_v - Y[|u|v] \) and \( d_r = -N_r - N[|u|v|] \) are hydrodynamic damping effects.

The propelling thrust \( \tau_u \) and steering torque \( \tau_r \) are in turn defined by

\[
\begin{align*}
\tau_u &= T_{ps} + T_{sb} \\
\tau_r &= I(T_{ps} - T_{sb})
\end{align*}
\]

where \( T_{ps} \) and \( T_{sb} \) are the port side and starboard thrust forces on the horizontal plane, respectively, and \( I \) is the displacement of the propellers from the center of \( \{ B \} \) (see Figure 2).

When we introduce a irrotational ocean current, \( v_c \), forming an angle \( \phi \) with respect to \( \{ I \} \), the kinematic equations (1) hold but with \( u = u_r + u_c \) and \( v = v_r + v_c \), where \( u_r \) and \( v_r \) are the components of the vehicle velocity with respect to the current and \( u_c \) and \( v_c \) are the components of the ocean current velocity in \( \{ B \} \). Furthermore, dynamic equations (3) become

\[
\begin{align*}
m_u \dot{u}_r - m_v vr + d_u u_r &= \tau_u \\
m_v \dot{v}_r - m_u ur + d_v v_r &= 0 \\
m_r \dot{r} - m_u vr + d_r r &= \tau_r
\end{align*}
\]

where now \( d_u = -X_u - X[|u|v], d_v = -Y_v - Y[|u|v] \).

A. The Medusa Autonomous Marine Vehicle

Throughout this work we use a full dynamic model of the MEDUSA-class of autonomous semi-submersible robotic vehicles, developed at the Laboratory of Robotics and Systems in Engineering and Science (LARSyS), Instituto Superior Tecnico, shown in Figure 1. Each MEDUSA-class vehicle weighs approximately 30 Kg and consists of two longitudinal acrylic housings with a total length of around 1 m. The upper body is partially above the surface and carries an EPIC single-board computer, an RTK-enabled GPS receiver, a full navigation sensor suite and an underwater camera. Most of the lower body is taken up by the batteries. An 802.11 interface is used for surface communications, while a Tritech acoustic modem enables underwater communication. The vehicle is propelled by two side-mounted, forward-facing stern thrusters that directly control surge and yaw motion, and is capable of speeds up to 1.5 m/s. Roll and pitch motion are not actuated directly.

B. Thruster Model

The Medusa vehicle uses two Seabotix brushless HPDC1507 thrusters, with a maximum rotational speed of approximately 75 rotations per second.
For our objective of minimizing the electrical energy that the thrusters consume, a mapping between the rotational velocities of the propellers ($n_{ps}$ and $n_{sb}$ for the portside and starboard propeller, respectively) and the thrust $T_{ps}$ and $T_{sb}$ is necessary. Classically, this is done by computing the so-called open water coefficients [6] defined in terms of the open-water advance ratio

$$J_0 = \frac{v_0}{nD}$$

where $v_0$ is the propeller’s advance speed and $D$ its diameter. It is important to remark that the open-water coefficients can only be used when the advance velocity is non-zero together with a non-zero propeller velocity, where the rotational direction must be the one that drives the vehicle forward, i.e. $n > 0$. This stems from the fact that (7) is not defined for $n = 0$. To overcome this problem, a full four-quadrant model could have been used, as in [5] and [7]. For the sake of simplicity, we decided to use the open-water coefficients and constrain the propellers rotational velocities to be positive.

In the model chosen, the thrust and torque equations are given by

$$T = \rho D^4 K_T(J_0)n|n|$$

$$Q = \rho D^5 K_Q(J_0)n|n|$$

where $\rho$ is the water density (which we assume is constant) and $K_T$ and $K_Q$ are non-dimensional thrust and torque coefficients. These coefficients are obtained from self-propulsion tests, where $T$, $Q$ and $n$ are measured.

To compute the advance speed, we have to take into account that they are not the same for each propeller when the vessel is rotating. From [5], the following expressions are obtained:

$$v_{aps} = \sin(\text{atan2}(-p_y, p_x))l r + u = -p_y r + u$$

$$v_{asb} = \sin(\text{atan2}(p_y, p_x))l r + u = p_y r + u$$

where $(p_x, p_y)$ is the offset of the propellers from the vehicle center of mass in body coordinates and $l$ is their absolute distance to the center of mass (see Figure 2).

III. PROJECTION OPERATOR APPROACH

This section provides a brief overview of the mathematical framework used to solve the optimization problems. In particular, the ones that involve solving finite-horizon optimal control problems of the form

$$\text{minimize} \int_0^T l(t, s(t), u(s(t)))dt + m(s(T))$$

subject to $\dot{s}(t) = f(s(t), u(s(t)))$, $s(0) = x_0$

where $x \in \mathbb{R}^n$ is the state-vector, $u \in \mathbb{R}^m$ the control-vector, $l(\cdot)$ is the running cost and $m(\cdot)$ the terminal cost.

The Projection Operator based Newton method for Trajectory Optimization (PRONTO) approach, as introduced in [4], will be discussed along with the necessary formulation to implement it.

A. PRONTO

The main idea behind PRONTO is to define a projection operator to convert the constrained optimal control problem (12) into an unconstrained one to be solved using Newton’s method. To this end and since the vector field $f$ can be unstable, the PRONTO method takes a trajectory-tracking approach. Suppose that $\xi(t) = (\alpha(t), \mu(t))$, $t \geq 0$, is a bounded curve (e.g., an approximate trajectory of $f$) and let $\eta(t) = (x_s(t), u_s(t)), t \geq 0$, be the trajectory of $f$ determined by the nonlinear feedback system

$$\dot{x}_s(t) = f(x_s(t), u_s(t))$$

$$u_s(t) = \mu(t) + K_e(t)(\alpha(t) - x_s(t))$$

Under certain conditions, this feedback system defines a nonlinear projection operator

$$P: \xi = (\alpha, \mu) \mapsto \eta = (x_s, u_s)$$

that maps a certain trajectory to a trajectory that belongs to the trajectory manifold $T$ (the set of trajectories that satisfy the system dynamics).

Making $g(\xi) := h(P(\xi))$, with

$$h(\xi) = \int_0^T l(t, \alpha(t), \mu(t))dt + m(\alpha(T))$$

we define the following Newton based method.
Algorithm 1 Projection Operator Newton Trajectory Optimization

given initial trajectory \( \xi_0 \in \mathcal{T} \\
for i = 0, 1, 2, \ldots \ do
1. Compute a regulator \( K_r \) for \( \xi_i \)
2. Compute the search direction:
   \[
   \xi_i = \arg \min_{\xi \in \mathcal{T}} Dg(\xi_i) \cdot \xi + \frac{1}{2} D^2 g(\xi_i) \cdot (\xi, \xi)
   \]
   with \( g(\xi_i) = h(\mathcal{P}(\xi_i)) \)
3. Compute the step size: \( \gamma_i = \arg \min_{\gamma \in (0, 1]} g(\xi_i + \gamma \xi_i) \)
4. Update: \( \xi_{i+1} = \mathcal{P}(\xi_i + \gamma_i \xi_i) \)
end

This algorithm is quite similar to the usual Newton method for unconstrained optimization of a function \( g(\cdot) \). The main differences are that the search direction minimization is performed on the tangent space \( (T_\xi \mathcal{T}) \) about a trajectory manifold and the update step projects each iterate on to the trajectory manifold.

A suitable feedback gain \( K_r \) may be constructed by solving a finite horizon linear regulator problem (LQR) [8] about a trajectory \( \eta = (x(t), u(t)) \). The matrices \( Q_r = Q_r^T > 0 \) and \( R_r = R_r^T > 0 \) that penalize the state and control errors, respectively, are defined in order to ensure the desired stability-like property. Also, to ensure good regulation all the way up to the final time \( T \), we include a positive definite terminal cost, \( P_{fr} = P_{fr}^T > 0 \). The feedback is thus chosen as \( K_r(t) = R_r^{-1} B^T P_{fr}^{-1} P_r(t) \), \( t \in [0, T] \) where \( P \) is the solution of the usual Riccati differential equation.

To compute the step size of the optimization algorithm a backtracking Armijo line search will be used [9]. It could also have been used a fixed step size of \( \gamma_i = 1 \), as a pure Newton method would, but by using the line search the region of convergence is expanded.

B. Approximate Barrier Functional

In most optimal control problems, the system dynamics \( \dot{x}_s(t) = f(x_s(t), u_s(t)) \), \( x_s(0) = x_0 \) are not the only constraints of the problem. Given finite \( k \) constraints on the state and/or control variables, the constrained optimal control problem can be cast as

\[
\begin{align*}
\min & \int_0^T l(\tau, x_s(\tau), u_s(\tau))d\tau + m(x_s(T)) \\
\text{s.t.} & \dot{x}_s(t) = f(x_s(t), u_s(t)), \quad x_s(0) = x_0 \\
& c_j(x_s(t), u_s(t), t) \geq 0, \quad t \in [0, T], \quad j \in \{1, \ldots, k\}
\end{align*}
\]

where \( c(\cdot) \) is the constraint function.

To solve this problem we incorporate the constraints using the barrier functional method [10]. The direct translation is clearly

\[
\begin{align*}
\min & \int_0^T l(\tau, x_s(\tau), u_s(\tau)) \\
& - \epsilon \sum_j \log(c_j(x_s(\tau), u_s(\tau)))d\tau + m(x_s(T)) \\
\text{s.t.} & \dot{x}_s(t) = f(x_s(t), u_s(t)), \quad x_s(0) = x_0
\end{align*}
\]

The main difficulty of solving this problem is the fact that it is not possible to evaluate the objective in (17) unless \( \xi \) is a feasible curve (i.e. satisfies the constraints), since the logarithmic function is not defined for non-positive arguments.

To overcome that we define an approximate log barrier function \( \beta_\delta(\cdot) \), \( 0 < \delta \leq 1 \)

\[
\beta_\delta(z) = \begin{cases} 
- \log z & z > \delta \\
0 & \begin{cases}
\frac{k - 1}{k} \left( \frac{z - k\delta}{(k - 1)\delta} \right)^k - 1 - \log \delta & z \leq \delta \\
\delta & z < \delta
\end{cases}
\end{cases}
\]

where \( k > 1 \) is an even integer. Usually, \( k = 2 \) is used. This function is similar to the log barrier function for \( z < 0 \) but expands its domain from \((0, \infty)\) to \((-\infty, \infty)\).

The final formulation of the constrained optimal control problem (16) becomes

\[
\begin{align*}
\min & \int_0^T l(\tau, x_s(\tau), u_s(\tau)) \\
& + \epsilon \sum_j \beta_\delta(c_j(\tau, x_s(\tau), u_s(\tau)))d\tau + m(x_s(T)) \\
\text{s.t.} & \dot{x}_s(t) = f(x_s(t), u_s(t)), \quad x_s(0) = x_0
\end{align*}
\]

where we start by choosing reasonably large values for \( \epsilon \) and \( \delta \) and reduce them in every algorithm iteration to force the trajectories in to the valid region.

The fact that (18) tends to infinity as \( z \) tends to infinity may cause some undesired effects. For example, in a obstacle avoidance constraint this fact may result in a domination of the cost integral and effectively putting a reward on staying away from the obstacle as far as possible. To overcome that, we make use of the "hockey stick" function [11],[5]

\[
\sigma(z) = \begin{cases} 
\tanh(z) & z \geq 0 \\
\frac{z}{\delta} & \text{otherwise}
\end{cases}
\]

In this case, for the constraints that have this problem, we define a new barrier functional \( \beta_\delta(\sigma(z)) \) that behaves as the standard approximate barrier function (18) for small and negative \( z \), but goes to zero as \( z \) goes to infinity.

IV. PROBLEM FORMULATION

In this section we will define the various optimization problems that we are interested to solve, namely by defining the cost functional. The constraints added and the parameters needed by the optimization algorithm (Section III) is also discussed here.

A. Minimum Time Trajectories

In minimum time optimal control problems, the goal is to drive the vehicle from a given initial state to a desired final one in minimum time. A valid choice for the integral cost term in (16) can be

\[
l(t, x_s(t), u_s(t)) = 1
\]

and the optimal solution would be computed by optimizing the final time \( T \). However, the final time \( T \) cannot be directly used as an optimization variable in PRONTO. So, to compute
Algorithm 2 Minimum Time Algorithm using PRONTO

given initial small final time \( T \)

for \( i = 1, 2, 3, \ldots \) do

Solve problem using PRONTO, with final time \( T \)

if converged then

Stop for cycle

end if

Increase \( T \)

end for

it we solve several optimization problems with different final times. The algorithm is the following.

The idea is to start with a small final time \( T \) and keep increasing it until Pronto converges, in which case we have the optimal solution. However, sometimes the PRONTO algorithm can converge even when the solution does not satisfy the constraints or the final position of the vehicle is far from the desired one. In those cases we do not consider it as valid solution and continue to increase \( T \).

Another issue that arises with the chosen integral cost term (21) is that it needs to be twice differentiable in \( u_s \), as expected by PRONTO. As we can see from (21), this is not the case. Although, as we will need to add constraints to the optimization problem (Section IV-C) and convert it to (19), the new integral cost term will become twice differentiable in \( u_s \).

The terminal cost will be

\[
m(x_s(T)) = \|x_s(T) - x_{s_{des}}(T)\|^2_{K_1}/2 \tag{22}\]

where \( \|a\|^2_K = a^\top K a \).

B. Minimum Energy Trajectories

In these type of problems, the idea is to generate trajectories that minimize the energy usage throughout the mission. More precisely, we want to minimize the electrical energy the batteries need to use in order to drive the vehicle.

Taking the standard DC motor equation

\[
L_a \frac{dl}{dt} + R_a I = V - K_e \omega \tag{23}
\]

and neglecting the dynamics, since the terminal inductance \( L_a \) is small, we get the relation between the rotational velocities of the propellers and the armature voltage as

\[
V_{ps, sb} = R_a I_{ps, sb} + K_e \omega_{ps, sb} = R_a I_{ps, sb} + K_e 2 \pi n_{ps, sb} \tag{24}
\]

where \( K_e \) is the electrical constant of the DC motor, \( R_a \) is the resistance of the DC motor, and \( V_{ps, sb} \) and \( I_{ps, sb} \) are the armature voltage and current of the portside/starboard DC motor, respectively.

To determine the armature currents as a function of the rotational velocities, some experimental measures where made with the Medusa vehicle and a 3rd order polynomial was fitted to the data obtained [12]

\[
I_{ps} = 0.1142e^{-3}n_{ps}^3 + 0.4355e^{-3}n_{ps}^2 + 23.5331e^{-3}n_{ps} 
+ 8.2628e^{-3} \tag{25}
\]

\[
I_{sb} = 0.1175e^{-3}n_{sb}^3 + 0.5602e^{-3}n_{sb}^2 + 29.4185e^{-3}n_{sb} 
+ 14.3139e^{-3} \tag{26}
\]

The integral cost term for the optimization problem (16) then becomes

\[
l(t, x_s(t), u_s(t)) = V_{ps}(t)I_{ps}(t) + V_{sb}(t)I_{sb}(t) 
= (R_a I_{ps}(t) + K_e 2 \pi n_{ps}(t))I_{ps}(t) 
+ (R_a I_{sb}(t) + K_e 2 \pi n_{sb}(t))I_{sb}(t) \tag{27}
\]

and the terminal cost is the same as in (22).

C. Constraints

1) Constraints on the inputs: The use of the open-water coefficients for the propellers’ model implies that their rotational velocity has to be positive. Also, due to the thruster’s specifications we know that their top speed is approximately 75 rotation per second.

So, the constraint function for \( n_{ps} \) becomes

\[
c_{n_{ps}}(x_s, u_s(t)) = -(n_{ps} - n_{max})(n_{ps} - n_{min}) \tag{28}
\]

which is a second order polynomial that is positive between \( n_{min} \) and \( n_{max} \) and negative otherwise, as expected by the optimization algorithm. We chose \( n_{max} = 75 \) and \( n_{min} = 1 \).

The constraint for the other input (\( n_{sb} \)) is defined in the same way.

2) Obstacle Avoidance: One of the requirements is that the generated trajectories are collision free. To that end, it is necessary to add a constraint on the vehicle spatial position.

Considering an obstacle with spatial coordinates \( (x_{obs}, y_{obs}) \) in reference frame \( \{I\} \), we can define the constraint as

\[
c_{obs}(x_s(t), u_s(t)) = \frac{(x(t) - x_{obs})^2}{D_{obs}} + \frac{(y(t) - y_{obs})^2}{D_{obs}} - 1 \tag{29}
\]

with \( D_{obs} = r_{obs} + r_{v} \), where \( r_{obs} \) is the radius of the obstacle and \( r_{v} \) is the safety distance that must be kept between the obstacle and the vehicle.

There will be as much constraints as there are obstacles.

V. MULTIPLE VEHICLES

In multiple vehicles mission scenarios, the trajectory planner has to take into account the different dynamics of each vehicle as well as making sure they do not collide with each other.

In this work we will focus mainly on two types of cooperative missions. One of them is when the goal of each vehicle is to go from an initial to a final position independently. In other words, the vehicles only need to cooperate in order to avoid collisions with each other, while optimizing the overall mission. Another type of mission is when we want to move a group of vehicles from one area to another, but in a way
that the vehicles maintain a specified geometrical formation between each other.

The simplest way to take into account multiple vehicles in the optimization problem is by increasing the state vector. More precisely, consider $N_v$ vehicles then the new state vector becomes

$$x_s = \begin{bmatrix} x_s^1 \\ x_s^2 \\ \vdots \\ x_{s[N_v]} \end{bmatrix}$$ (30)

where $x_s^i, \quad i \in \{1, \ldots, N_v\}$ is the state vector of the $i$-th vehicle. The control vector and other vectors and matrices needed by PRONTO ($Q_r, R_r, P, \ldots$) are also increased in the same way. With this approach is possible to include vehicles that have different kinematic and dynamic equations.

A. Inter-vehicle collision avoidance

As said before we need to make sure that the optimal trajectory is collision free. We do that by adding constraints to the position of the vehicles, similar to what was done in Section IV-C2.

Thus, for each pair of vehicles $\{i, j\}$ we define the constraint

$$c_{col}(x_s^i(t), x_s^j(t)) = \frac{(x^i(t) - x^j(t))^2}{D_v^2} + \frac{(y^i(t) - y^j(t))^2}{D_v^2} - 1$$ (31)

with $i, j \in \{1, \ldots, N_v\}$ and where $D_v$ is the minimum safety distance that must be kept between two vehicles. The total number of constrains added is $\binom{N_v}{2}$.

B. Formation Motion

In some interesting scenarios, it is necessary for the vehicles to perform trajectories where they need to keep a certain configuration between each other. We do that by defining a leader vehicle that the others have to follow from a certain relative position, while the leader moves from an initial to a final position. In PRONTO, this is done by adding a penalization to the integral cost term, $l(x_s(t), u_s(t))$, in (19).

Defining $(x_w^i(t), y_w^i(t))$ as the desired position, expressed in $\{I\}$, the $i$-th vehicle should be at time $t$ (see Figure 3), the added penalization term becomes

$$l_{form}(x_s(t), u_s(t)) = \sum_{i=1}^{N_v-1} \left(\frac{(x^i(t) - x_w^i(t))^2}{D_v^2} + \frac{(y^i(t) - y_w^i(t))^2}{D_v^2}\right)$$ (32)

To compute $x_w^i(t)$ and $y_w^i(t)$ in each instant, a coordinate $(x_b^i, y_b^i)$ is defined for each vehicle in the leader body reference frame, which depends on the desired formation (e.g., in a side-by-side formation we could have $x_b^1 = 0$ and $y_b^1 = 10$).

Fig. 3. Diagram that represents the vectors used in the formation missions.

The mapping between them becomes

$$x_w^i(t) = x^i(t) + d_i^1 \cos(\theta_i^i)$$
$$y_w^i(t) = y^i(t) + d_i^1 \sin(\theta_i^i)$$ (33)

with

$$d_i^1 = \|x_b^i - y_b^i\|$$ (34)
$$\theta_i^i = \psi^i + \arctan(\frac{y_b^i}{x_b^i})$$ (35)

where $(x^i, y^i)$ and $\psi^i$ are, respectively the position and heading of the leader vehicle (see Figure 3).

VI. RESULTS

To assess the performance of the proposed implementation several simulations were made with simple examples that highlight the different aspects of the planner.

A. Example 1

The objective of this first example is to compare the optimal solutions for the time and energy minimization problems in the presence of space variant water currents. There will be only one vehicle, and the components of the ocean current velocity are

$$u_c = -0.8 \tanh(y)$$
$$v_c = 0$$ (36)

In other words, the current velocity only has a component in $x$ that depends on the $y$ position. The maximum value is $0.8ms^{-1}$. This example is quite similar to one in [13].

Figure 4 shows the obtained results for the minimum time of 22.5s.

By looking at the results, we can see that the vehicle uses the water current to its advantage by trying to get as fast as it can to a place with favourable currents (positive $y$), instead of trying
to go against them. We can also see that in order to minimize the consumed energy, the vehicle opts to do a wider curve so it does not need to turn so fast, which is energy expensive. The vehicle also goes upper to obtain stronger favourable currents.

It should also be noted that the constraints on the propellers’ rotational velocities are satisfied and the trade-off between time and energy is in accordance with the expected results, since as time increases a smaller energy is used.

B. Example 2

The objective of this example is to see how well the obstacle avoidance constraint works. In this scenario, to make a comparison with the previous case, this is example is similar to the previous one with the exception that an obstacle was added.

Figure 5 shows the optimal trajectory for the minimum energy problem with a simulation time of $T = 35s$.

As can be seen, the added obstacle was clearly in the way of the previous solution (Figure 4(a)). The new trajectory successfully avoids the obstacle by passing as close to it as it can. This solution, however, is much more energy expensive than the previous one, since the vehicle needs to rotate faster, especially after passing the obstacle.

C. Example 3

In this simulation we will have two vehicles that need to go from an initial position to a final one. These positions were chosen in way that the vehicles need to cooperate in order to not crash whit each other. This way, we can evaluate the performance of the inter-vehicle collision constraint. There is no water current in this example.

Figure 6 shows the optimal trajectory obtained for the minimum time ($56s$) and minimum energy ($65.5s$) problems.

As can be seen, to avoid having a collision the vehicles do not go in a straight line, but instead they opt to do a little curved trajectory. In the energy minimization problem the curve is wider so that the variation of angular speed (yaw rate) is not as big. This conclusion is actually similar to the one made in Example 1.

D. Example 4

For the final example, we will solve a minimum time optimal control problem where the vehicles need to pass trough several way-points before reaching the final position. More precisely, we want to do a square movement. To do that we are going to solve several optimization problems between each consecutive way-point.

In this example, there will be 3 vehicles that have to move in a triangular formation. The desired coordinates that the vehicles need to be on the leader body reference frame (see Section V-B) are

\[
\begin{align*}
  x_b^{[1]} &= 12m & y_b^{[1]} &= 6m \\
  x_b^{[2]} &= 0m & y_b^{[2]} &= 12m
\end{align*}
\]  

(37)
VII. Conclusions

In summary, this work addressed the problem of developing a motion planner for single and multiple autonomous vehicles. A simple model for an AUV was used, along with a propeller model based on the open-water coefficients. We solved minimum-time and minimum-energy optimal control problems making use of the Projection Operator Approach with some modifications so constraints could be added.

The generated trajectories were feasible in terms of the vehicles’ dynamics and input constraints and they were also free of obstacles and inter-vehicles collisions. An approach to perform mission with formation motions was developed by adding a term to the integral cost of the optimization problem. The proposed implementation has shown good behaviour in the computational simulations made.

A. Future Work

There are several issues that could be addressed in future research in this area, such as the study of time variant currents and other current configurations and the case of moving obstacles and communication constraints. Also, the time performance of the implementation should be improved for the planning to be done in real time so it can adapt to environment changes.

REFERENCES


