RESEARCH ARTICLE

A Receding Horizon Framework for Autonomy in Unmanned Vehicles

Marina Murillo* | Guido Sánchez | Lucas Genzelis | Nahuel Deniz | Leonardo Giovanini

Abstract

In this article we present a unified framework based on receding horizon techniques that can be used to design the three tasks (guidance, navigation and path-planning) which are involved in the autonomy of unmanned vehicles. This tasks are solved using model predictive control and moving horizon estimation techniques, which allows us to include physical and dynamical constraints at the design stage, thus leading to optimal and feasible results. In order to demonstrate the capabilities of the proposed framework, we have used Gazebo simulator in order to drive a Jackal unmanned ground vehicle (UGV) along a desired path computed by the path-planning task. The results we have obtained are successful as the estimation and guidance errors are small and the Jackal UGV is able to follow the desired path satisfactorily and it is also capable to avoid the obstacles which are in its way.

KEYWORDS: Model predictive control, Moving horizon estimation, Receding horizon techniques, Unmanned vehicle

1 | INTRODUCTION

In the field of unmanned vehicles (UVs), autonomy refers to the ability of such vehicles to perform a set of predefined tasks without human interaction. To do this, UVs should be able of sensing their environment to measure their positions and velocities, as well as the obstacles present in their proximity, in order to navigate through the environment and achieve the targeted positions to perform the tasks. The execution of these tasks involves the acquisition and processing of a wide variety of sensors (accelerometers, gyroscopes, GPS and cameras, among others), as well as the solution of coupled optimization problems at different time scales. In this context, these activities can be organized in three interrelated tasks (See Figure 1):

- **Path-planning** task which aims to find a feasible path across the surrounding environment to achieve the next waypoint \( \tilde{w}_k \in \tilde{\mathcal{W}} \), where \( \tilde{\mathcal{W}} \) is the waypoint constraint set. It employs all the information available of the environment such as...
obstacles, limits, forbidden regions, among others, (represented through a set of constraints in the position and velocities called here as environmental constraints $c^E_k(x_k) \in C^E \subseteq \mathbb{R}^{n_x+n_u}$, where $C^E$ is the environmental constraint set) to define the space where the UV is allowed to move; and the current estimated position $\hat{y}_k$ and states $\hat{x}_k$ of the UV to find a feasible path $x_{sp}^k$ defining the future behaviour of the UV on a time horizon longer than the UV dynamic.

- **Navigation** task which aims to estimate the position $y_k$ and states $x_k$ of the UV using information from sensors that measure the dynamic states (translational accelerations, rotational velocities, absolute and relative positions, among others) of the UV. Taking into account that sensors have constraints $c^S_k(x_k) \in C^S \subseteq \mathbb{R}^n$, where $C^S$ is the sensors constraint set, and the estimated noise measurement $\hat{v}_k$, this task employs estimations algorithms to estimate the current position ($\hat{y}_k$), attitude and states ($\hat{x}_k$) of the vehicle.

- **Guidance** task which aims to control the current state $x_k$ (position, attitude and velocity) of the UV, moving the vehicle from its current estimated state $\hat{x}_k$ (which is estimated by the navigation task) towards the computed path $x_{sp}^k$ (which is computed by the path-planning task) taking into account the disturbances and vehicle constraints $c^UV_k(x_k,u_k) \in C^UV \subseteq \mathbb{R}^{n_x+n_u}$, where $C^UV$ is the vehicle constraint set. This task computes the control actions $u_k$ (position of the actuators) which are necessary to control the UV.

![FIGURE 1 Tasks interaction scheme](image-url)

In the following paragraphs, we will present the state of the art concerning the three aforementioned tasks. As we have mentioned before, the guidance task refers to the control of the position, attitude and velocity of a vehicle along a pre-defined path. To do this, a suitable control algorithm should be used in order to compute which actuators’ deflections and/or motors’ speeds the vehicle should have so as to accomplish a certain mission. A wide range of control techniques have been tested, ranging from classical to modern ones. Proportional-integral-derivative (PID) control is indeed the most popular control technique used to control UVs. This is mainly due to its simplicity and because its parameters are easy to adjust. Several works dealing with PID
control of UV have been found in the specialized literature. Based on the classic scheme of PID control, Li and Li have designed a controller which aims to regulate the position and orientation of a six degree-of-freedom (DOF) quadrotor. They obtain the PID control parameters by means of simulations and then they perform experimental tests in a free-obstacle low-speed wind environment. Zhao et al. present the control system architecture of an autonomous vehicle called "Intelligent Pioneer". They use a two DOF dynamic model and an adaptive-PID controller in order to provide more flexibility to the vehicle control system. PID parameters can be selected online in order to obtain better control performance than with conventional PID. However, when using a PID controller a decoupled version of the mathematical model of the UV is used. This may lead to unexpected results in the presence of disturbances and even limit its performance due to unmodelled dynamics. Also, in most cases when performing experimental tests PID parameters should be re-tuned by a trial and error approach. Another technique that has been successfully applied to control UVs is the linear quadratic regulator (LQR) control algorithm. Khamseh et al. use an LQR scheme to design a coupled controller that enables simultaneous control of an unmanned quadcopter which is equipped with a two DOF robotic manipulator. By adding an integral action to the LQR controller, the authors achieve better path following characteristics as the steady state error is removed. Within optimal control techniques model predictive control (MPC) can be found. Unlike LQR, the MPC algorithm allows to include constraints in the optimization problem. This is very useful as physical and dynamical characteristics of the vehicle and different types of obstacles can be taken into account just by the inclusion of proper constraints in the minimization stage. Recently intelligent algorithms, including neural networks and genetic algorithms, have been used to solve the guidance problem. However, this kind of algorithms involve high computational complexity, which limits their usage with systems with fast dynamics.

The navigation task aims to solve the problem of determining the position, velocity and orientation of a vehicle in space using different sources of information (inertial measurement units, GPS, among others). Traditionally, the Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF) or the Particle Filter (PF) are used to solve the navigation problem. Recently, the use of non-linear observers have been proposed as an alternative to the different types of Kalman filters and statistical methods. However, there is still little literature on the subject. Grip et al. present an observer for estimating position, velocity, attitude, and gyro bias, by using inertial measurements of accelerations and angular velocities, magnetometer measurements, and satellite-based measurements of position and (optionally) velocity. Vandersteen et al. use a Moving Horizon Estimation (MHE) algorithm in real-time to estimate the orientation and the sensor calibration parameters applied to two space mission scenarios. In the first scenario, the attitude is estimated from three-axis magnetometer and gyroscope measurements. In the second scenario, a star tracker is used to jointly estimate the attitude and gyroscope calibration parameters. In order to solve this constrained optimization problem in real time, an efficient numerical solution method based on the iterative Gauss–Newton scheme has been implemented and specific measures are taken to speed up the calculations by exploiting the sparsity and band structure of matrices to be inverted. Both EKF and MHE are based on the solution of a least-squares problem. While EKF
uses recursive updates to obtain the estimates and the error covariance matrix, MHE uses a finite horizon window and solve a constrained optimization problem to find the estimates. In this way, the physical limits of the system states and parameters can be modeled through the optimization problem’s constraints. The omission of this information can degrade the estimation algorithm performance\(^\text{17}\). Unfortunately, the Kalman based filters do not explicitly incorporate restrictions in the estimates (states and/or parameters) and, because of this, several ad-hoc methods have been developed\(^\text{18–23}\). These methods lead to sub-optimal solutions at best and can obtain non-realistic solutions under certain conditions, specially when the statistics of the unknown variables are chosen poorly. On the other hand, MHE solves an optimization problem to find the system estimates on each sample step, providing a theoretical framework for constrained state and parameter estimation.

The path-planning task has attracted substantial attention\(^\text{24,25}\). It deals with searching a feasible path between the present location and the desired target while taking into consideration the geometry of the vehicle and its surroundings, its kinematic constraints and other factors that may affect the feasible path. Different methodologies are used to find feasible paths\(^\text{26}\). In the literature, some recent path-planning algorithms can be found\(^\text{27–29}\). For example, Saska et al.\(^\text{27}\) introduce a technique that integrates a spline-planning mechanism with a receding horizon control algorithm. This approach makes it possible to achieve a good performance in multi-robot systems. Xue et al.\(^\text{28}\) present an offline path-planning algorithm for UAVs in complex terrain. The authors propose an algorithm which can be divided into two steps: firstly a probabilistic method is applied for local obstacle avoidance and secondly a heuristic search algorithm is used to plan a global trajectory. As it can be seen, there are many methods to obtain feasible paths for UVs; however, most of them do not consider the dynamics of the UV that should follow the path. In their review article, Yang et al.\(^\text{30}\) have surveyed different path-planning algorithms. The authors discuss the fundamentals of the most successful robot 3D path-planning algorithms that have been developed in recent years. They mainly analyze algorithms that can be implemented in aerial robots, ground robots and underwater robots. They classify the different algorithms into five categories: i) sampling based algorithms, ii) node based algorithms, iii) mathematical model based algorithms (which include optimal control and receding horizon strategies), iv) bioinspired algorithms, and v) multifusion based algorithms. From these, only mathematical model based algorithms are able to incorporate in a simple way both the environment (kinematic constraints) and the vehicle dynamics in the path-planning process. Recently, Hehn and D’Andrea\(^\text{31}\) introduced a trajectory generation algorithm that can compute flight trajectories for quadcopters. The proposed algorithm computes three separate translational trajectories (one for each degree of freedom) and guarantees the individual feasibility of these trajectories by deriving decoupled constraints through approximations. The authors consider the quadcopter dynamics when they compute the flight trajectories but the proposed technique is not a general one (it can not be used with ground vehicles, for example). Even though the feasibility is guaranteed for each separate trajectory, the resulting vehicle trajectory might not be necessarily feasible (e.g., when perturbations are present). Minh and Pumwata\(^\text{32}\) present three conventional holonomic trajectory generation algorithms (flatness, polynomial and symmetric) for ground vehicles subject to constraints on their steering angle. In order to satisfy this constraint, they propose
to lengthen the distance from the initial position to the final position until the constraint is satisfied. This process might be tedious and it may not be applicable in dynamic environments. Besides, it can only be used with ground vehicles and it can only handle steering constraints violations.

As it can be seen in previous paragraphs, there are a wide range of techniques that can be used to implement the guidance, navigation and path-planning tasks. The main drawback of this approach is the need of assuming the validity of the separation principle (which is not always true) and the global analysis of the whole system is difficult to perform due to the lack of analytical tools applicable to different techniques simultaneously. This is the main reason why we propose to use a unified framework based on receding horizon techniques to design the three aforementioned tasks. To design the path-planning and guidance modules we propose the use of the MPC algorithm\(^{33}\) and to solve the navigation task we propose the use of the MHE algorithm\(^{34}\). In this way, physical and dynamical constraints can be considered in the path-planning, guidance and estimation stages. The advantages of using the proposed framework are: i) the obtained path is guaranteed to be optimal and feasible, ii) the estimates of states and parameters of the system are improved as they are guaranteed to satisfy physical limits, iii) the position of actuators and motors’ speeds are computed in an optimal fashion satisfying their physical limits, and iv) the three modules are in state-space form which is very useful when working with multiple-input multiple-output (MIMO) systems.

This article is organized as follows. In Section 2 we provide a general overview of receding horizon techniques. In Section 3 the guidance problem is formulated. In Section 4 the way in which we estimate states and parameters is presented. The path-planning problem is described in Section 5. Simulation results are presented in Section 6. Finally, conclusions are stated in Section 7.

2 RECEDING HORIZON PRINCIPLE

In this section we provide an overview of receding horizon strategies commonly used in control and estimation. Unlike classical control and estimation techniques, the main objective of receding horizon techniques is to solve an explicit inverse problem that allows the incorporation, at the design stage, of different types of constraints to obtain the best feasible solution. For both control and estimation, the inverse problem to solve is the minimization of a cost function that quantifies the performance of the system (for control case) and how well we estimate unknown states and parameters (for estimation case). This constrained minimization process is done over a fixed-time horizon window of a certain length. This is shown in Figure 2 for time \(k\), being the cyan dashed-dot line the control window of length \(N_c\) and the red dashed line the estimation window of length \(N_e\). The arrival cost provides a mean to incorporate information from previous measurements to the current estimates and the cost to go includes the missing information due to the use of a finite horizon approach. At the next sampling instant \(k + 1\), new information is included
and old one is discarded by shifting both windows one step in time and the constrained minimization process is restarted at the next sampling instant. This is also shown in Figure 2 for time $k + 1$.

When we refer to a control receding strategy we think in model predictive control. As we mentioned before, the goal of this kind of receding strategy is to minimize a cost function that measures the system performance. For control case, we generally solve at each sampling instant, a constrained minimization problem of the form

$$
\min_{U_k} J_{\text{MPC}}(x_{k+1|k}, u_{k+1|k}, x_{sp}^{k}, N_c)
$$

subject to

$$
\begin{align*}
x_{k+i|k} &= f(x_{k+i|k}, u_{k+i|k}), & i \in [0, 1, \ldots, N_c - 1] \\
x_{k|k} &= x(k), \\
u_{k+i|k} &\in U, \\
x_{k+i|k} &\in X,
\end{align*}
$$

(1)

where $J_{\text{MPC}}(\cdot)$ denotes the cost function to be minimized, $x_{k+i|k} \in X \subseteq \mathbb{R}^n_x$ is the state vector, $u_{k+i|k} \in U \subseteq \mathbb{R}^n_u$ is the control input vector, $X$ and $U$ are the state and input constraint sets, respectively, $U_{k|k} = [u_{k|k}, \ldots, u_{k+N_c-1|k}]^T$ is the control input sequence and $f(\cdot)$ is a continuous and differentiable vector function that describes the dynamics of the system. The solution of problem (1) is an optimal control input sequence (denoted here with an asterisk) $U_k^* = [u_{k|k}^*, \ldots, u_{k+N_c-1|k}^*]^T$, but only the first control input of this sequence is applied to the system, i.e. $u_k = u_{k|k}^*$. Then, the horizon is shifted forward to the next sampling instant ($k \leftarrow k + 1$) in a receding horizon fashion (see the cyan dashed-dot line windows in Fig. 2), discarding old information (orange dot in Figure 2) but including the new one (green dot in Figure 2), thus compensating for unmeasured disturbances and/or unmodelled dynamics. As the reader can see, the cost function $J_{\text{MPC}}$ in (1) plays a key role in obtaining the optimal control sequence and it should be carefully designed in order to fulfill the goals of the system. The different cost functions used to design the guidance and path-planning tasks will be properly defined in Section 3 and 5, respectively.

FIGURE 2 Scheme of the receding horizon principle
To perform states and/or parameters estimation in a receding horizon fashion we use moving horizon estimation. In this case, we also solve a constrained minimization problem at each sampling instant but the solution obtained is different from the previous one as it includes estimates of states, parameters and noises. The MHE problem has the following form:

\[
\min_{\hat{x}_{k+1|k}, \hat{z}_{k+1|k}, \hat{w}_{k+1|k}, N_e} J_{\text{MHE}}(\hat{x}_{k+1|k}, \hat{z}_{k+1|k}, \hat{w}_{k+1|k}, N_e)
\]

\[
\begin{align*}
\hat{x}_{k+1|k} &= \hat{f}(\hat{x}_{k+1|k}, \hat{z}_{k+1|k}, \hat{w}_{k+1|k}), \\
\hat{y}_{k+1|k} &= \hat{h}(\hat{x}_{k+1|k}, \hat{v}_{k+1|k}), \quad i \in [-N_e, \ldots, -1] \\
\hat{x}_{k+1|k} &\in \hat{X}, \quad \hat{z}_{k+1|k} \in \hat{Z}, \\
\hat{w}_{k+1|k} &\in \hat{W}, \quad \hat{v}_{k+1|k} \in \hat{V},
\end{align*}
\]

where the hat (\(\hat{\cdot}\)) denotes estimated value, \(\hat{x}_{k+1|k} \in \hat{X} \subseteq \mathbb{R}^{n_x}\), \(\hat{z}_{k+1|k} \in \hat{Z} \subseteq \mathbb{R}^{n_z}\) stand for the estimated state and algebraic state vectors, respectively, \(\hat{w}_{k+1|k} \in \hat{W} \subseteq \mathbb{R}^{n_w}\) and \(\hat{v}_{k+1|k} \in \hat{V} \subseteq \mathbb{R}^{n_v}\) are the estimated process and measurement noises vectors, respectively, \(\hat{y}_{k+1|k} \in \hat{Y} \subseteq \mathbb{R}^{n_y}\) is the measurement vector, \(\hat{f}\) and \(\hat{h}\) are differentiable vector functions that define the system dynamics and the measurement equation, respectively. \(\hat{X}, \hat{Z}, \hat{W}, \hat{V}\) are the state, algebraic, state, process noise, measurement noise and measurement vector constraints sets, respectively. The sequences \(\hat{X}_{k|k} = [\hat{x}_{k-N_e+1|k}, \ldots, \hat{x}_{k|k}]^T\), \(\hat{Z}_{k|k} = [\hat{z}_{k-N_e+1|k}, \ldots, \hat{z}_{k|k}]^T\), \(\hat{W}_{k|k} = [\hat{w}_{k-N_e+1|k}, \ldots, \hat{w}_{k|k}]^T\) and \(\hat{V}_{k|k} = [\hat{v}_{k-N_e+1|k}, \ldots, \hat{v}_{k|k}]^T\) are the state, algebraic state, process noise and measurement noise sequences, respectively. The solution of problem (2) are the optimal sequences \(\hat{X}^*_{k|k}\) and \(\hat{W}^*_{k|k}\) from where we extract the current estimates, i.e. \(\hat{x}_k = \hat{X}^*_{k|k}\) and \(\hat{w}_k = \hat{W}^*_{k|k}\). Then, the estimation window is shifted forward to the next sampling instant \((k \leftarrow k + 1)\), see the red dashed line windows in Figure 2, in order to drop the oldest measurement (magenta dot in Figure 2) and to include the current one (yellow dot in Figure 2), and the minimization process is restarted. In this case, the cost function \(J_{\text{MHE}}\) also plays a key role in the behaviour of the MHE algorithm, and it should be carefully chosen in order to obtain the best estimates as possible. The cost function used in the navigation task will be properly described in Section 4.

## 3 | THE GUIDANCE PROBLEM

As we have mentioned before, based on the estimations \(\hat{x}_{k|k}\) performed by the navigation task, the guidance task aims to control the position, attitude and velocity of a vehicle in order to drive it along the path \(x^{\text{sp}}_{k+j|k}\) with \(j \in [0, 1, \ldots, N_c]\), which is computed by the path-planning task. To do this, a control algorithm should be implemented in this task in order to compute actuators’ positions and/or motors’ speeds so as the vehicle is able to follow the desired path. As we have reviewed in Section 1, the control of a vehicle can be held using many techniques. In this work, we propose to use the Nonlinear Model Predictive Control (NMPC) technique to solve this problem as it allows to take into account the vehicle’s dynamics and its physical constraints at the minimization stage, thus giving optimal and feasible results. To design the guidance task, we propose to solve problem (1)
using the following cost function:

\[
    J_g(\mathbf{x}_{k+j|k}, \mathbf{u}_{k+j|k}, \mathbf{x}^{sp}_{k+j|k}, N_g) = \sum_{j=0}^{N_g-1} L_j(\mathbf{x}_{k+j|k}, \mathbf{u}_{k+j|k}, \mathbf{x}^{sp}_{k+j|k}) + L_{N_g}(\mathbf{x}_{k+N_g|k}, \mathbf{x}^{sp}_{k+N_g|k}),
\]

where \( N_g \) is the guidance prediction horizon, \( L_j(\cdot, \cdot) \) is the stage cost and \( L_{N_g}(\cdot, \cdot) \) stands for the terminal cost (or cost to go), which are commonly adopted as follows:

\[
    L_j(\mathbf{x}_{k+j|k}, \mathbf{u}_{k+j|k}, \mathbf{x}^{sp}_{k+j|k}) = \| \mathbf{x}_{k+j|k} - \mathbf{x}^{sp}_{k+j|k} \|^2_Q + \| \Delta \mathbf{u}_{k+j|k} \|^2_R
\]

and

\[
    L_{N_g}(\mathbf{x}_{k+N_g|k}, \mathbf{x}^{sp}_{k+N_g|k}) = \| \mathbf{x}_{k+N_g|k} - \mathbf{x}^{sp}_{k+N_g|k} \|^2_P,
\]

where \( Q, R, P \) are positive definite matrices. \( \| \cdot \|^2_a \) stands for the \textit{alpha}-weighted 2-norm, \( \Delta \mathbf{u}_{k+j|k} = \mathbf{u}_{k+j|k} - \mathbf{u}_{k+j-1|k} \) and \( \mathbf{x}^{sp}_{k+j|k} \) is the desired path which is computed by the path-planning task.

Now that we have defined the guidance cost function, we should define which mathematical model will be used in the guidance task. This model can be either kinematic or dynamic, however, it should be as accurate as possible in order to improve the performance of the guidance task. Here, we choose to work with the best mathematical model we have available. In many cases, the UV model is a simplified and approximate model of the vehicle we want to drive along the pre-defined path. Note that as the adopted model is not exact, the control algorithm embedded in the guidance task should be robust enough in order to obtain good results. The model that will be used in the guidance task is described properly in Section 6.

4 | THE NAVIGATION PROBLEM

The navigation task uses the information from different sensors in order to estimate, as accurate as possible, the position, velocity and attitude of a vehicle. This task can be solved using several techniques, among which we can find KF and MHE. Both techniques are based on the solution of a least-squares problem but, while EKF uses recursive updates to obtain the estimates and the error covariance matrix, MHE uses a finite horizon window and solve a constrained optimization problem to find the estimates. In this way, the physical limits of the system states and parameters can be modeled through the optimization problem’s constraints, this is what motivates us to use MHE to solve the navigation task. In order to design this task, we propose to solve problem 2 with the following cost function

\[
    J_n(\tilde{\mathbf{x}}_{k+j|k}, \tilde{\mathbf{z}}_{k+j|k}, \tilde{\mathbf{w}}_{k+j|k}, \tilde{\mathbf{v}}_{k+j|k}, N_e) = P_b(1 - ||\mathbf{q}_b^{\gamma}(k - N_e)||^2_2) + ||\tilde{\mathbf{x}}_{k-N_e}|k - \tilde{\mathbf{x}}_{k-N_e}|k||^2_{P_i} + ||\tilde{\mathbf{z}}_{k-N_e}|k - \tilde{\mathbf{z}}_{k-N_e}|k||^2_{P_2} + \sum_{j=-N_e}^{0} ||\tilde{\mathbf{w}}_{k+j|k}||^2_{Q_w} + ||\tilde{\mathbf{v}}_{k+j|k}||^2_{R_v}
\]
where $N_e$ is navigation prediction horizon, $Q_{av}$ and $R_v$ are symmetric positive definite matrices that penalize the estimated noise vectors $\hat{w}_{k+j|k}$ and $\hat{v}_{k+j|k}$, respectively, $\hat{x}_{k-N,|k}$ and $\hat{z}_{k-N,|k}$ are the current knowledge of the initial states and algebraic states estimates, respectively, $P_0$ is a positive constant which penalizes the deviations of the quaternion $\hat{q}_k^m$ which must have unit norm, and $P_1$ and $P_2$ are symmetric positive semi-definite weighting matrices.

Now that we have defined our navigation cost function, we should determine the mathematical model that will be used in the navigation task. Let us choose $x(t)$ as our state vector and $z(t)$ as our algebraic state vector, which are defined as follows:

$$x(t) = [p_n(t), v_n(t), q_n^m(t), \alpha(t), \beta(t)]^T \quad \text{and} \quad z(t) = [\omega^b(t), \alpha^b(t)]^T$$

where where $p_n(t) \in \mathbb{R}^3$ is the position in East-North-Up reference frame (referred as ENU and by the superscript $n$), $v_n(t) \in \mathbb{R}^3$ is the linear velocity in ENU coordinates, the quaternion $q_n^m(t) \in \mathbb{R}^4$ determines the orientation of the rigid body in ENU coordinates and $\alpha(t) \in \mathbb{R}^3$ and $\beta(t) \in \mathbb{R}^3$ are the gyroscope and accelerometer bias, respectively. $\omega^b(t) \in \mathbb{R}^3$ and $\alpha^b(t) \in \mathbb{R}^3$ are the angular velocity and linear acceleration vectors in body reference frame (referred as Body and by the superscript $b$) and ENU coordinates, respectively.

The measurement vector $y(t)$ is defined as

$$y(t) = [\omega_m^b(t), \alpha_m^b(t), m_m^b(t), p_m^e(t), v_m^e(t)]^T$$

where the subscript $m$ stands for measurement, the superscript $e$ refers to the Earth-Centered Earth-Fixed (ECEF) reference frame, $\omega_m^b(t) \in \mathbb{R}^3$ and $\alpha_m^b(t) \in \mathbb{R}^3$ are the measured angular velocity and linear acceleration vectors in body coordinates, respectively, $m_m^b(t) \in \mathbb{R}^3$ is the measured magnitude of the terrestrial magnetic field in body frame given our current latitude and longitude, $p_m^e(t) \in \mathbb{R}^3$ and $v_m^e(t) \in \mathbb{R}^3$ are the measured position and linear velocity vectors in ECEF frame, respectively.

Using the equations that describe the rigid body dynamics (for a detailed description, see Poloni et al.\textsuperscript{35} and Bekir\textsuperscript{36}), the mathematical model used in the navigation task can be written in ECEF coordinates as follows:

$$\dot{x}(t) = \tilde{f}(x(t), z(t), w(t)) = \begin{bmatrix} v^e(t) \\ -2S(\omega^e(t))v^e(t) + \alpha^e(t) + g^e(p^e(t)) \\ \frac{1}{2}q^e_b(t) \cdot \tilde{\alpha}^b_{ib}(t) - \frac{1}{2}\tilde{\alpha}^e_{ie}(t) \cdot q^e_b(t) \\ 0 \\ 0 \end{bmatrix}.$$
simply use the transpose of the ENU to ECEF rotation matrix. Finally, the measurement equations with measurement noise

\[ \mathbf{v}(t) = [v_w(t), v_a(t), v_m(t), v_p(t), v_v(t)]^T, \]  

with \( v_{c,j}(t) \in \mathbb{R}^3 \), are given by:

\[ y(t) = \tilde{h}(\mathbf{x}(t), \mathbf{v}(t)) = \begin{bmatrix} \omega^b(t) + \alpha(t) + v_w(t) \\ \mathbf{R}(q^e_{b})^T \alpha^e(t) + \beta(t) + v_a(t) \\ \mathbf{R}(q^e_{b})^T m^e(t) + v_m(t) \\ p^e(t) + v_p(t) \\ t^e(t) + v_v(t) \end{bmatrix}, \]  

where \( m^e(t) \) is a known vector that contains the values of the magnitude of the terrestrial magnetic field given our current latitude and longitude. \( \omega^b(t) \) and \( \alpha^e(t) \) are the angular velocity and linear acceleration vectors in body and ECEF coordinates, respectively. The matrix \( \mathbf{R}(q^e_{b}(t)) \) is the rotation matrix associated with the current orientation quaternion. As it can be seen, model and the measurement equation are expressed in their continuous-time form, but in order to be used in problem, they should be in their discrete-time form. We have solved this issue using a discretization method called collocation since it provides great accuracy at a relatively low computational cost.

5 | THE PATH-PLANNING PROBLEM

The goal of the path-planning task, as we have mentioned before, is to find a feasible path \( \mathbf{x}_{k+j|k}^{sp} \) between the current location and each consecutive target waypoint \( \mathbf{w}_{k+j|k} \), taking into account all the information about the environment, the geometry of the vehicle, its kinematics constraints and any other factor that may affect the feasibility of the path. As we have seen in Section, this problem can be solved using many techniques, particularly using a receding horizon scheme. One of the advantages of using this approach is that the dynamic of the vehicle which should follow the path together with its constraints, can be embedded at the path-planning stage leading to an optimal and feasible path. In this work we propose to use an MPC based path-planning algorithm which uses a virtual particle vehicle (PV) model to compute the paths. For more information about this path-planning algorithm, the reader is referred to Murillo et al.

In order to design the path-planning task we propose to solve problem with the following cost function:

\[ J_p(\mathbf{x}_{k+j|k}, \mathbf{u}_{k+j|k}, \mathbf{w}_{k+j|k}, N_p) = \sum_{j=0}^{N_p-1} L_j(\mathbf{x}_{k+j|k}, \mathbf{u}_{k+j|k}, \mathbf{w}_{k+j|k}) + L_{N_p}(\mathbf{x}_{k+N_p|k}, \mathbf{w}_{k+N_p|k}), \]  

*This data is tabulated and can be obtained from https://www.ngdc.noaa.gov/geomag-web*
where $N_p$ is the path-planning horizon length. For this case, the stage cost and the terminal cost are adopted as follows:

$$
\mathcal{L}_j(x_{k+j|k}, u_{k+j|k}, \tilde{w}_{k+j|k}) = \|x_{k+j|k} - \tilde{w}_{k+j|k}\|^2_Q + \|\Delta u_{k+j|k}\|^2_R \tag{13}
$$

and

$$
\mathcal{L}_{N_p}(x_{k+N_p|k}, \tilde{w}_{k+N_p|k}) = \|x_{k+N_p|k} - \tilde{w}_{k+N_p|k}\|^2_P \tag{14}
$$

where $\tilde{Q}$, $\tilde{R}$ and $\tilde{P}$ are positive definite matrices.

In order to describe the UV movement, we used the PV model because they have similar characteristics and it is simpler. For the PV, let us choose $\tilde{x}(t)$ as our state vector and $\tilde{u}(t)$ as our control input vector, which are defined as

$$
\tilde{x}(t) = [\tilde{x}(t), \tilde{y}(t), \tilde{v}(t)]^T \quad \text{and} \quad \tilde{u}(t) = [\tilde{\psi}(t), T(t)]^T \tag{15}
$$

where $\tilde{x}(t)$ and $\tilde{y}(t)$ denote the PV $xy$-position of the PV and $\tilde{v}(t)$ is the modulus of the PV velocity vector, $\tilde{\psi}(t)$ and $T(t)$ denote the yaw angle and the thrust force of the PV, respectively. Then, the mathematical model that describes the movement of the PV can be written as follows:

$$
\tilde{x}(t) = f(\tilde{x}(t), \tilde{u}(t)) = \begin{bmatrix}
\dot{\tilde{x}}(t) \\
\dot{\tilde{y}}(t) \\
\dot{\tilde{v}}(t)
\end{bmatrix}
= \begin{bmatrix}
\tilde{v}(t) \cos \tilde{\psi}(t) \\
\tilde{v}(t) \sin \tilde{\psi}(t) \\
-\tau \tilde{v}(t) + \kappa T(t)
\end{bmatrix}, \tag{16}
$$

where $\tau$ is a damping constant that determines the rate of change of the PV velocity and $\kappa$ is a constant proportional to the thrust force $T$. In order to use this model in problem (1), model (16) is also discretized using collocation method.

### 6 SIMULATION EXAMPLE

In this section we present the usage of the proposed framework in order to estimate unknown states, compute a feasible path and follow this path with a Jackal unmanned ground vehicle (UGV) using Gazebo simulator, CasADi, MPCTools and Ipopt.

The Jackal UGV is a small, fast, entry-level field robotics research platform. It is fully compatible with Robot Operating System (ROS) and it can be simulated using Gazebo. At this point we need to emphasize that we do not know the exact mathematical model of the Jackal UGV, however, we do know that it is a complex model simulated by Gazebo. Here, we propose to model Jackal UGV with the mathematical model of a differential drive UGV. This last model is very simple, but for us is the best model at hand and, as it will be shown in the simulation example, it will allow us to control accurately the Jackal UGV along

---

1 [https://www.clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/](https://www.clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/)
2 [http://gazebosim.org/](http://gazebosim.org/)
3 [http://www.ros.org/](http://www.ros.org/)
the pre-defined path. Let us choose \( \mathbf{x}(t) \) as our state vector and \( \mathbf{u}(t) \) as our control input vector, which are defined as follows:

\[
\mathbf{x}(t) = [\mathbf{x}(t), \mathbf{y}(t), \mathbf{v}(t)]^T \quad \text{and} \quad \mathbf{u}(t) = [\mathbf{v}(t), \mathbf{\omega}(t)]^T,
\]

(17)

where \( \mathbf{x}(t) \) and \( \mathbf{y}(t) \) denote the UGV xy-position, \( \mathbf{v}(t) \) denotes its yaw angle, \( \mathbf{v}(t) \) and \( \mathbf{\omega}(t) \) stands for the UGV linear and angular velocities, respectively. Then, the mathematical model of the differential drive UGV can be written as follows:

\[
\dot{\mathbf{x}}(t) = \tilde{f}(\mathbf{x}(t), \mathbf{u}(t)) = \begin{bmatrix}
\dot{\mathbf{v}}(t) \cos \tilde{\psi}(t) \\
\dot{\mathbf{v}}(t) \sin \tilde{\psi}(t) \\
\dot{\mathbf{\omega}}(t)
\end{bmatrix}.
\]

(18)

As before, model (18) is discretized using collocation method.

The guidance task is configured as follows. We assume the initial condition of model (18) are \( \mathbf{x}_0 = [0, 0, 0]^T \) and \( \mathbf{u}_0 = [0, 0]^T \). This model is discretized using collocation method with a sampling rate \( T_s = 0.1 \) s and the guidance horizon is set to \( N_g = 10 \). The weight matrices defined in (4) and (5) are chosen as: \( \mathbf{R} = diag([20, 15]) \), \( \mathbf{Q} = diag([300, 300, 150]) \) and \( \mathbf{P} = diag([500, 500, 400]) \). The differential drive constraints are configured according to the capabilities of the Jackal UGV: \( 0 \leq \dot{\mathbf{v}}(t) \leq 2 \) (m/s), \( -2 \leq \dot{\mathbf{\omega}}(t) \leq 2 \) (rad/s), \( -0.3 \leq \Delta \dot{\mathbf{v}}(t) \leq 0.3 \) (m/s), \( -20 \leq \Delta \dot{\mathbf{\omega}}(t) \leq 20 \) (deg/s). \( \tilde{\psi}(t), \mathbf{x}(t) \) and \( \mathbf{y}(t) \) are unconstrained.

For the path-planning task we assume that the PV model (9) has the following initial conditions: \( \mathbf{x}_0 = [0, 0, 0]^T \) and \( \mathbf{u}_0 = [0, 0]^T \). This model is also discretized using collocation method with a sampling rate \( T_s = 0.1 \) s. The path-planning horizon is set to \( N_p = 10 \). The weight matrices defined in (13) and (14) are chosen as \( \mathbf{\tilde{R}} = diag([0.1, 0.1]) \), \( \mathbf{\tilde{Q}} = diag([100, 100, 100]) \) and \( \mathbf{\tilde{P}} = diag([500, 500, 500]) \). The PV constraints are configured as follows: \( 0 \leq \tau(t) \leq 2 \) (N), \( -5 \leq \Delta \tilde{\psi}(t) \leq 5 \) (deg/s), \( -0.5 \leq \Delta \tau(t) \leq 0.5 \) (N) and \( 0 \leq \dot{\mathbf{v}}(t) \leq 2 \) (m/s). \( \tilde{\psi}(t), \mathbf{x}(t) \) and \( \mathbf{y}(t) \) are unconstrained. Both constants of the PV model are set as \( \tau = 2 \) (1/s) and \( \kappa = 2 \) (1/kg). As we are interested in having the computed path pass sufficiently close to the waypoints, but not exactly through them, we define a circular area centered at each waypoint. If the path passes through this area, then we consider that the corresponding waypoint has been reached. For all the waypoints we set this area to a disk with a radius of \( r = 0.1 \) (m). Also, when computing the path we assume that there are two circular obstacles of radii \( r_{o_1} = r_{o_2} = 0.5 \) (m) and centres \( \mathbf{c}_{o_1} = [−2, 3]^T \) (m) and \( \mathbf{c}_{o_2} = [2, 7]^T \) (m).

For the navigation task we assume that there is no process noise, i.e. \( \mathbf{\hat{w}_{k+j}^j} = 0 \). Also, we assume that the Jackal UGV simulated by Gazebo has two sensors: i) IMU, and ii) GNSS. The data given by these sensors is corrupted by Gaussian noise and it is fused in order to estimate the Jackal’s xy-position and its yaw angle. In this task, we use model (9) in ENU coordinates and its initial conditions are \( \mathbf{x}_{-N_e} = [0, 0, 0.0635, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]^T \) and \( \mathbf{\hat{z}}_{-N_e} = [0, 0, 0, 0, 0, 0]^T \). This model is also discretized using collocation method with a sampling rate \( T_s = 0.1 \) s and the estimation horizon is set to \( N_e = 6 \). The weights defined in (6) are chosen as \( \mathbf{Q}_{\hat{w}} = \)
\[
\text{diag}(\{10^{-1}, 10^{-1}, 10^{-3}, 10^{-3}, 1, 1, 1, 1, 1, 1, 1, 1\}), \quad R_v = \text{diag}(\{1, 1, 1, 1, 1, 10^{-3}, 10^{-3}, 10^{-3}\}), \quad P_0 = 0.1, \quad P_1 = \text{diag}(\{10^{-1}, 10^{-1}, 10^{-1}, 10^{-1}, 10^{-3}, 10^{-3}, 10^{-3}, 0.5, 0.5, 0.5, 0, 0, 0, 0\}) \text{ and } P_2 = \text{diag}(\{1, 1, 1, 1, 1\}).
\]

The results obtained can be seen in Figures 3-6. In Figure 3, three different paths are shown: i) the setpoint path (squared-green line) which is computed by the path-planning task, ii) the estimated Jackal \(xy\)-position (rounded-blue line) which is computed by the navigation task and controlled by the guidance task in order to follow the setpoint path, and iii) the ground truth (diamond-red line) which is given by Gazebo. From this figure it can be seen that the interrelated work of the three tasks (path-planning, navigation and guidance) is done in such a satisfactory way that it allows to move the Jackal UGV along the desired path with minor errors, considering that sensors are corrupted by Gaussian noise. In Figure 4 the Jackal UGV control inputs are shown. Both control inputs are feasible and, as it can be seen from this figure, the linear velocity of the vehicle is limited up to \(2\) (m/s) which matches both constraints defined in the path-planning and guidance tasks. In Figure 5 the guidance errors are shown, where \(x^p\) and \(y^p\) are the desired \(xy\)-coordinates that define the desired path and \(\psi^p\) is the desired yaw angle. Note that we do not only define the Jackal UGV \(xy\)-coordinates but also we impose the orientation it should have along the desired path. This is very useful because without this angle condition the Jackal UGV could follow some part of the desired path in reverse gear, which is not a desirable movement, for example, if we want to use the vehicle for images acquisition. \(\hat{x}, \hat{y}\) and \(\hat{\psi}\) are the components of the estimated Jackal state vector. As it can be seen form this figure (top and middle), the guidance errors in \(xy\)-position are small. However, the guidance error in the yaw angle seems to be a bit bigger than the errors in the position. This is mainly due to the fact that the yaw angle is a control input for the PV model and consequently no dynamics is considered. This problem can be solved by adding this angle as a state variable and defining a first order dynamics for it, as we did with the velocity of the PV. Figure 6 depicts the estimation errors, where \(x^t, y^t\) and \(\psi^t\) denote the truth \(xy\)-component and the truth yaw angle, respectively, which can be obtained from Gazebo simulator in order to check if our estimations are close to this truth values. It can be seen that the navigation task is successfully employed as the errors we have made in the estimation of \(xy\)-position (between \(\pm0.3\) (m)) and the yaw angle (between \(\pm15\) (deg)) of the Jackal UGV are small.

7 | CONCLUSIONS

In this article, we have presented a unified receding horizon framework that can be used for the guidance and navigation of any UUV along any feasible path. This framework can be split into three interrelated task, all of them designed using a receding horizon principle, which allows us to include physical dynamics and constraints at the design stage. We have used MPC technique for guidance and path-planning tasks and MHE for the navigation task. To evaluate the performance of the proposed framework, we have used Gazebo simulator in order to drive a Jackal UGV model along the path computed by the path-planning task. As we have shown in the last section, the results we have obtained are satisfactory.
ACKNOWLEDGMENTS

The authors wish to thank the Universidad Nacional de Litoral (with CAI+D Jóven 500 201501 00050 LI and CAI+D 504 201501 00098 LI), the Agencia Nacional de Promoción Científica y Tecnológica (with PICT 2016-0651) and the Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) from Argentina, for their support.
FIGURE 5 Guidance Errors

FIGURE 6 Estimation Errors

References

1. Li Jun, Li Yuntang. Dynamic analysis and PID control for a quadrotor. In: :573–578IEEE; 2011.

2. Zhao Pan, Chen Jiajia, Song Yan, Tao Xiang, Xu Tiejuan, Mei Tao. Design of a control system for an autonomous vehicle based on adaptive-pid. International Journal of Advanced Robotic Systems. 2012;9(2):44.

3. Khamseh Hossein Bonyan, Janabi-Sharifi Farrokh. UKF–Based LQR Control of a Manipulating Unmanned Aerial Vehicle. Unmanned Systems. 2017;5(03):131–139.
4. Hang Peng, Luo Fengmei, Fang Shude, Chen Xinbo. Path tracking control of a four-wheel-independent-steering electric vehicle based on model predictive control. In: :9360–9366IEEE; 2017.

5. Salamat Babak, Tonello Andrea M. Altitude and attitude tracking of a quadrotor helicopter UAV using a novel evolutionary feedback controller. In: :327–331IEEE; 2017.

6. Liu Yan-Cheng, Liu Si-Yuan, Wang Ning. Fully-tuned fuzzy neural network based robust adaptive tracking control of unmanned underwater vehicle with thruster dynamics. *Neurocomputing*. 2016;196:1–13.

7. Carrio Adrian, Sampedro Carlos, Rodriguez-Ramos Alejandro, Campoy Pascual. A Review of Deep Learning Methods and Applications for Unmanned Aerial Vehicles. *Journal of Sensors*. 2017;2017.

8. Lefferts Ej, Markley Fl, Shuster Md. *Kalman filtering for spacecraft attitude estimation*. 1982.

9. Markley F, Sedlak Joseph. Kalman Filter for Spinning Spacecraft Attitude Estimation. *Journal of Guidance, Control, and Dynamics*. 2008;31(6):1750–1760.

10. Roumeliotis Stergios, Bekey George. 3-D Localization for a Mars Rover Prototype. In: :441–448; 1999.

11. Crassidis John, Markley F. Unscented Filtering for Spacecraft Attitude Estimation. *Journal of Guidance, Control, and Dynamics*. 2003;26(4):536–542.

12. Rhudy Matthew, Gu Yu, Gross Jason, Gururajan Srikant, Napolitano Marcello R.. Sensitivity Analysis of Extended and Unscented Kalman Filters for Attitude Estimation. *Journal of Aerospace Information Systems*. 2013;10(3):131–143.

13. Carmi Avishy, Oshman Yaakov. Adaptive Particle Filtering for Spacecraft Attitude Estimation from Vector Observations. *Journal of Guidance, Control, and Dynamics*. 2009;32(1):232–241.

14. Cheng Yang, Crassidis John. Particle Filtering for Sequential Spacecraft Attitude Estimation. In: American Institute of Aeronautics and Astronautics; 2004; Reston, Virigina.

15. Grip Håvard Fjær, Fossero Thor I, Johansen Tor A, Saberi Ali. A nonlinear observer for integration of GNSS and IMU measurements with gyro bias estimation. In: :4607–4612IEEE; 2012.

16. Vandersteen Jeroen, Diehl Moritz, Aerts Conny, Swevers Jan. Spacecraft Attitude Estimation and Sensor Calibration Using Moving Horizon Estimation. *Journal of Guidance, Control, and Dynamics*. 2013;36(3):734–742.

17. Haseltine Eric L, Rawlings James B. Critical evaluation of extended Kalman filtering and moving-horizon estimation. *Industrial & engineering chemistry research*. 2005;44(8):2451–2460.
18. Simon Dan. Kalman filtering with state constraints: a survey of linear and nonlinear algorithms. *Control Theory & Applications, IET*. 2010;4(8):1303–1318.

19. Simon Dan, Chia Tien Li. Kalman filtering with state equality constraints. *IEEE transactions on Aerospace and Electronic Systems*. 2002;38(1):128–136.

20. Hall David Lee, McMullen Sonya AH. *Mathematical techniques in multisensor data fusion*. Artech House; 2004.

21. Teixeira Bruno Otávio Soares, Chandrasekar Jaganath, Palanthandalam-Madapusi Harish J, Torres Leonardo Antônio Borges, Aguirre Luis Antonio, Bernstein Dennis S. Gain-constrained Kalman filtering for linear and nonlinear systems. *IEEE Transactions on Signal Processing*. 2008;56(9):4113–4123.

22. Simon Dan, Simon Donald L. Constrained Kalman filtering via density function truncation for turbofan engine health estimation. *International Journal of Systems Science*. 2010;41(2):159–171.

23. Ko Sangho, Bitmead Robert R. State estimation for linear systems with state equality constraints. *Automatica*. 2007;43(8):1363–1368.

24. United States Air Force . Unmanned aircraft systems flight plan 2009–2047. *Headquarters Department of the Air Force, Washington DC*. 2009;.

25. Weatherington Dyke, Deputy U. Unmanned aircraft systems roadmap, 2005–2030. *UAV Planning Task Force, OUSD (AT&L)*. 2005;.

26. LaValle Steven M. *Planning algorithms*. Cambridge university press; 2006.

27. Saska Martin, Spurný Vojtěch, Vonásek Vojtěch. Predictive control and stabilization of nonholonomic formations with integrated spline-path planning. *Robotics and Autonomous Systems*. 2015;.

28. Xue Qian, Cheng Peng, Cheng Nong. Offline path planning and online replanning of UAVs in complex terrain. In: 2287-2292; 2014.

29. Zhang Guoqing, Zhang Xianku. A novel DVS guidance principle and robust adaptive path-following control for underactuated ships using low frequency gain-learning. *ISA Transactions*. 2015;56:75 - 85.

30. Yang Liang, Qi Juntong, Song Dalei, Xiao Jizhong, Han Jianda, Xia Yong. Survey of Robot 3D Path Planning Algorithms. *Journal of Control Science and Engineering*. 2016;2016.

31. Hehn Markus, D’Andrea Raffaello. Real-Time Trajectory Generation for Quadrocopters. *IEEE Transactions on Robotics*. 2015;31(4):877–892.
32. Minh Vu Trieu, Pumwa John. Feasible path planning for autonomous vehicles. *Mathematical Problems in Engineering*. 2014;2014.

33. Murillo M., Sánchez G., Giovanini L.. Iterated non-linear model predictive control based on tubes and contractive constraints. *ISA Transactions*. 2016;62:120 - 128.

34. Sánchez G, Murillo M, Giovanini L. Adaptive arrival cost update for improving Moving Horizon Estimation performance. *ISA transactions*. 2017;68:54–62.

35. Polóni Tomáš, Rohal-Ikiv Boris, Arne Johansen Tor. Moving Horizon Estimation for Integrated Navigation Filtering. *IFAC-PapersOnLine*. 2015;48(23):519–526.

36. Bekir Esmat. *Introduction to modern navigation systems*. World Scientific; 2007.

37. J. Sanz-Subirana J. Zornoza, Hernández-Pajares M.. Transformations between ECEF and ENU coordinates. [http://www.navipedia.net/index.php/Transformations_between_ECEF_and_ENU_coordinates](http://www.navipedia.net/index.php/Transformations_between_ECEF_and_ENU_coordinates) [Online; accessed 30-07-2017]; 2011.

38. Sánchez Guido, Murillo Marina, Genzelis Lucas, Deniz Nahuel, Giovanini Leonardo. MPC for nonlinear systems: A comparative review of discretization methods. In: ;1–6IEEE; 2017.

39. Murillo M., Sánchez G, Genzelis L, Giovanini L. A Real-Time Path-Planning Algorithm based on Receding Horizon Techniques. *Journal of Intelligent & Robotic Systems*. 2017;:. In press.

40. Andersson Joel AE, Gillis Joris, Horn Greg, Rawlings James B, Diehl Moritz. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*. In Press, 2018;.

41. Risbeck MJ, Rawlings JB. *MPCTools: Nonlinear model predictive control tools for CasADi*. 2016.

42. Wächter Andreas, Biegler Lorenz T. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*. 2006;106(1):25–57.
How to cite this article: M. Murillo, G. Sánchez, L. Genzelis, N. Deniz, and L. Giovanini (20xx), A Receding Horizon Framework for Autonomy in Unmanned Vehicles, *Optimal Control Applications and Methods*, 20xx;xx:xx–xx.