Assessing path-following performance for Unmanned Marine Vehicles with algorithms from Numerical Commutative Algebra

E. Saggini\textsuperscript{1,2}, M. Torrente\textsuperscript{1}, E. Riccomagno\textsuperscript{1}, M. Bibuli\textsuperscript{2}, G. Bruzzone\textsuperscript{2}, M. Caccia\textsuperscript{2}, E. Zereik\textsuperscript{2}

Abstract— A current trend in marine robotics consists of performance evaluation of Unmanned Marine Vehicles (UMVs) guidance systems. This paper contributes to this by defining and testing performance indices and metrics to quantitatively measure and compare path-following performance. It focuses on the definition of a new criterion for evaluating the capability of an Unmanned Surface Vehicle (USV) to follow a desired generic curvilinear path. The main advantages of the method are represented by the possibility to compute the proposed index online and in a very general situation. Theoretical bases of the presented approach and preliminary results on simulated data that highlight the versatility of this criterion are presented.

I. INTRODUCTION

Technology transfer within many domains is strongly hampered by the lack of common and established methodologies to compare robot performances and to be able to reproduce experiments. This fact contributes to the growth, especially within the marine community, of discussion and research (both theoretical and application-oriented) about possible techniques for quantitatively measuring the robotic system performances, without neglecting the experimental method. Objective of this discussion is the definition of standards and rules to be applied to unmanned vehicles (like USVs) in civilian scenarios, in order to be able to assess safety [1] and also to establish insurance fees. Currently, the definition of benchmarks and metrics for quantitative evaluation of system performance is being addressed, as well as the creation of protocols for executing repeatable experiments. Moreover, the importance of the availability of common platforms and data sets for the testing phase is being highlighted [2].

Preliminary work has been done, for example in [3], defining fundamental guidelines that are circulating as recommendations for important robotics journals and conferences in order to effectively write experimental papers. Interest groups (such as Euron GEM Sig, \url{www.heronrobots.com/EuronGEMSig/}), technical committees (like IEEE Robotics & Automation TC-PBRAS16), dedicated workshops and special issues of international journals arose and are supporting discussions on these topics. The authors of [4] propose general guidelines to improve methodologies and reporting, with the aim of simplifying robotic experiment replication, comparison and performance evaluation. Work described in [2] reports, from an interdisciplinary point of view, considerations about experimental localization and mapping for mobile robots. Performance methodologies for robots belonging to search&rescue contexts are reported in [5], underlining the interactions with the standardization ‘Committee on Homeland Security Applications - Operational Equipment - Robots’ (E54.08.01). Data sets, mainly for navigation and SLAM applications, can be found in [6] for benchmarking, including for example measurements from Doppler Velocity Logger, compass and mechanically scanned sonar obtained in an abandoned marina with the Ictineu underwater vehicle [6].

Controllability of experimental conditions (e.g. waves, wind, sea currents and presence of recreational and commercial traffic), restricted number of executable experiments (due to costs and logistics), and uncertainty in the system inputs (e.g. forces and torque generated by propellers, rudders and planes known with a degree of uncertainty due to hydrodynamic interactions, in particular, for small vehicles) represent the major challenges in the application of rules and standards to marine robotics. Within marine robotics, basic performance metrics have been defined in [7], evaluating USV path-following performance and achieving test repeatability by moving along a straight or curved path in opposite directions with U-turns. USV path-tracking performance evaluation have been defined in [8] via repeating experiments by sending pre-logged target navigation data to the follower USV control system. Work in [9], [10] and [11], about ROV and AUV repeatable experiments, pertains to the execution of experiments at sea, in presence of significant constraints in terms of controllability of experimental conditions, restricted number of executable experiments and uncertainty in the inputs assigned to the system. Recent work by the authors deals with the identification of good experimental methodologies and practices to carry out repeatable experiments with USVs [12] and with the definition and validation of performance indices in marine robotics applications [13].

The present paper aims at establishing a new performance evaluation criterion apt to estimate the capability of an USV to follow a desired generic curvilinear path. The main advantages of the proposed criterion consist in the possibility to compute the described index both online and offline and in the fact that the target path does not need to be a mathematical function. For clarity of exposition and to highlight the main features of the proposed index, results are obtained through simulations of the CNR-ISSIA Charlie USV, described in [14] and depicted in Figure 1. The Charlie vehicle has an integrated simulation system (hardware-in-the-loop) allowing to perform experiments, simulating real conditions and accurate vehicle behaviour.

The paper is organized as follows. In Section II a brief
overview of the application field and of the problem statement is given. In Section III the new metrics and algorithms to compute it are described. In Section IV-B results from a simulation study are reported, validating the algorithm and highlighting the versatility of the proposed criterion. In Section V some conclusions are drawn, together with suggestions on future work.

II. APPLICATION FIELD AND STATEMENT OF THE PROBLEM

Methodologies proposed in this research to evaluate marine vehicles’ performances are applied to the field of UMVs. Such a research line assumes great importance for technology transfer and implementation of robotic technologies in civilian contexts, as it could greatly speed up the process of introducing robotic platform in the everyday civil life. As already mentioned, the problem consists in evaluating robotic systems’ performances, assessing their safety and employing them in civilian scenarios. The proposed solution consists of the following three steps:

1. define a performance metrics \( M \), based on reference, measurements, state signals and control actions, if accessible. It could be a combination of different performance indices and should be computed for each experiment;
2. define a system performance \( P \) as a function of \( M \) computable (at least theoretically) with respect to all possible reference signals and initial conditions;
3. design a limited suitable set of experiments, i.e. define reference signals and initial conditions, with respect to which to evaluate \( P \).

In the following the definition of a new criterion to evaluate system performance is presented to evaluate the robot performances online and in a very general situation. This new criterion is fundamental in the definition of \( M \).

III. DEFINITION AND COMPUTATION OF THE PATH FOLLOWING PERFORMANCE METRICS

Let \( \mathcal{X} = \{(x_{R,i}, y_{R,i}), i = 1, \ldots, n\} \subset \mathbb{R}^2 \) and \( \mathcal{Y} = \{(x_{V,i}, y_{V,i}), i = 1, \ldots, m\} \subset \mathbb{R}^2 \) be two sets identifying the reference and vehicle paths, respectively. The proposed method to evaluate quantitatively when the vehicle is close to the reference path, is composed of two steps:

1. computation of an algebraic curve \( f = 0 \) that approximates the points in \( \mathcal{X} \) within a tolerance \( \varepsilon_1 \) and
2. identification of the points in \( \mathcal{Y} \) far from the reference path \( f = 0 \) for more than a tolerance \( \varepsilon_2 \).

There are several methods in the literature to address Step 1. An interesting class of recently developed algorithms relies on tools from Numerical Commutative Algebra [15], [16], [17], [18], [19], [20], [21]. The input is a set of points \( \mathcal{X} \) possibly in \( d \)-dimensions and the output is a polynomial \( f \) in \( d \)-variables whose zero-locus defines a curve or surface or more generally an algebraic variety. The polynomial \( f \) gives an approximation of the points of \( \mathcal{X} \) and can be interpreted as an implicit polynomial regression model [22, Ch 2]. The algorithm presented in [21] is particularly interesting for our purposes because it returns a polynomial \( f \) that not only almost vanishes at \( \mathcal{X} \), but also is an implicit minimum least square regressor in some sense to be explained in Subsection III-A. The algorithm is tested on simulated data in Section IV-B. For Step 2 a rule is required to classify whether each point in \( \mathcal{Y} \) is close to the algebraic variety defined by \( f = 0 \) for less than \( \varepsilon_2 \). To this aim, the idea illustrated in [23] is exploited. The general methods behind the two steps hold in higher dimension but we illustrate the basic ideas for \( d = 2 \) because of the application to USV paths.

A. First step: approximation of the target path through a polynomial curve

In [21] a new algorithm, called Low-degree Polynomial Algorithm (LPA), is presented that returns a “simple” polynomial \( f \) whose zero-locus “almost” contains the points of \( \mathcal{X} \). The simplicity of a polynomial \( f \) is measured by its total degree, i.e. the largest sum of exponents of the monomials in the support of \( f \). A point is said to be almost contained in the zero-locus of \( f \) if the \( \varepsilon_1 \)-ball (w.r.t. a given norm) centred at the point intersects the variety \( f = 0 \). Here we use the \( L_\infty \)-norm and \( \varepsilon_1 \) represents the maximum error of the coordinates of a point. More formally, the polynomial \( f \) sought in Step 1 has to meet the two following requirements:

1) its total degree has to be bounded by the smallest total degree amongst the total degrees of all polynomials vanishing at all points in \( \mathcal{X} \). The set of all such polynomials is called the vanishing ideal of \( \mathcal{X} \) or design ideal [22]; and
2) the zero-locus of \( f \) lies close to the points of \( \mathcal{X} \) by less than the tolerance \( \varepsilon_1 \) (the distance in \( \mathbb{R}^2 \) is induced by the given norm).

For further details on the algebraic perspective see [21].

The current implementation of LPA is in C++ and uses some routines of the GSL-GNU Scientific Library [24]. All computations in the paper have been performed on an Intel Core 2 Duo processor at 1.86 GHz. LPA assumes an ordering of monomials. Below the degree lexicographic term ordering is used: a polynomial of total degree \( t \) precedes polynomials.
with total degree higher than $t$ and ties between polynomials of the same total degree are resolved by a lexicographic principle. In Section IV-B we consider both $x < y$ and $y < x$. Furthermore in Section IV-B the coordinates of the points and the coefficients of the polynomials are displayed as truncated decimals.

In order to get an estimation on the quality of the approximation obtained through the polynomial $f$ we compute a variant of the adjusted determination coefficient, defined as

$$adj R^2 = 1 - \frac{n - 1}{n - k - 1} \frac{\sum_{i=1}^{n} (y_{R,i} - \hat{y}_{R,i})^2}{\sum_{i=1}^{n} (y_{R,i} - \bar{y}_R)^2},$$

where $n$ is the number of points in $X$, $k - 1$ is the number of monomials in $f$, $\bar{y}_R$ is the sample mean of $y_{R,1}, \ldots, y_{R,n}$ and $\hat{y}_{R,i}$ is the predicted value at $x_{R,i}$ by $f$. Note that $0 \leq adj R^2 \leq 1$ and $adj R^2 = 1$ indicates perfect fit. There are two slight differences w.r.t. the classical adjusted $R^2$: the computation of the number of predictors and the definition of predicted values. Specifically, for $i = 1, \ldots, n$ consider $f(x_{R,i}, y)$ as a polynomial in $y$, solve the equation $f(x_{R,i}, y) = 0$ and take as $\hat{y}_{R,i}$ the real solution closest to $y_{R,i}$ in Euclidean norm.

B. Second step: the Crossing Cell algorithm

Let $f = 0$ be the algebraic plane curve found in Subsection III-A and let $\varepsilon_2$ be a fixed tolerance for the closeness of a generic point $(x_V, y_V)$ to $f$. The Crossing Cell algorithm is developed in [23] where details are reported. The crossing criterion depends on a tolerance $\varepsilon_2$, on the evaluation $f(x_V, y_V)$ and on the local differential geometry of $f = 0$. In [23] analytic bounds are given that can be exploited to produce a rule for establishing whether a point is close to $f = 0$ by less than $\varepsilon_2$, distant from $f = 0$ by more than $\varepsilon_2$ or the question remains unresolved. To decrease computation time an approximation is adopted and, in this case, $\varepsilon_2$ has to be smaller than 1. For each pair $(x_{V,i}, y_{V,i}) \in \mathbb{V}$ the output of the Crossing Cell algorithm returns a value in $\{0, 1, \xi\}$. If the current $i$-th position contains

- 0, then the curve $f(x, y) = 0$ does not cross the $\varepsilon_2$-ball centred in $(x_{V,i}, y_{V,i})$;
- 1, then the curve $f(x, y) = 0$ crosses the $\varepsilon_2$-ball;
- $\xi$, then the crossing problem remains undecided.

IV. SIMULATION STUDY

This section reports, firstly, the description of the simulative environment which is based on the Charlie USV architecture and next the simulative results, obtained by the employment of the proposed method.

A. The Charlie USV simulator

The Charlie USV system is a modular architecture allowing the integration of multiple payload devices, the hardware-in-the-loop (HIL) simulation and the interaction of the human operator at different levels of command. The overall system is composed by three main elements: i) the Charlie vehicle; ii) the control system; iii) the human operator remote interface.

![Fig. 2. The Charlie USV architecture.](image)

The flexible architecture scheme allows the Charlie vehicle element to be the real USV or a simulation system interacting with the control and remote interface systems, without needing any change of the two latter elements. As depicted in Figure 2, the Charlie simulator receives the command packages (i.e. the actuation directives) from the control system and sends the sensors’ data streams, depending on the actual proprioceptive sensor and payload configuration. The simulator implements the dynamical and kinematic models of the vehicles; on the basis of the actuation input received, combined with artificial environmental disturbances, vehicle’s velocities, position and attitude are updated at each sampling step. From the vehicle’s state values, the sensors’ set generates the simulated measurements, corrupting the signals with specific noise, and sends them to the control system. Each simulated sensor channel relies on a connectionless oriented socket service. This allows a complete modular configuration between simulated signals and real measurements allowing a selective analysis of the Charlie subsystems. The simulation system allows the execution of all the different guidance techniques, implemented in the control system, representing a valuable tool for new algorithms validation.

The comparison carried out between simulated data and real application measurements validated the good response of the simulation system, thus allowing the development of navigation, guidance and control techniques, on-line perception algorithms, high-level mission execution. Such tasks can be debugged and tested by means of the simulator, obtaining the same behaviour once the system is exploited in real conditions.

B. Simulative results

Two sets $X, Y \subset \mathbb{R}^2$ identifying respectively the reference path (140 points) and the vehicle path (2113 points) have been simulated. For Step 1 in Section III LPA is applied to a subset of 24 points of $X$, almost one every six available points. Two plane curves have been generated by adopting a
degree lexicographic term ordering

1) with \( y < x \) and \( \varepsilon_1 = (0.01, 0.01) \) the outcome is
\[
f_1(x, y) = x^5 - x^4y - 0.3x^3y^2 + 0.1x^2y^3 + \ldots
\]
be addressed in the near future, as working with a symbolic representation is not relevant for the robotic purposes.

2) with \( x < y \) and \( \varepsilon_1 = (0.05, 0.05) \) the outcome is
\[
f_2(x, y) = x^4 + 284.5x^4 + 3.4x^3y + 8.9x^2y^2 - 23.8y^3 + 38.4y^4 + 24.3x^2 - 184.5y^2 + 4978xy^2 - 92.2y^3 + 533729x^2 - 384797xy + 882521y^2 - 3791490x - 55608300y + 1323870000.
\]

From Figure 3 both approximations seem to be very good fits. This is confirmed by the values of the adjusted \( R^2 \)
\[
adj R^2_f = 0.9999985 \quad \text{and} \quad adj R^2_f = 0.9999986.
\]
Both \( adj R^2 \)'s are computed for the whole dataset \( X \). It is worth noting that overfitting issues, troublesome in more classical regression problems, are not relevant here.

The Crossing Cell algorithm for \( f_1 \) and \( f_2 \) with \( \varepsilon_2 = (0.9, 0.9) \) leads to very similar results (only around 20 points over 2113 are differently classified). Figure 4 shows the \( \{0, 1, \xi\} \) coding for \( f_1 \). We choose \( \varepsilon_2 = 0.9 \) because it represents the radius of the ball in \( L_\infty \)-norm and that Charlie USV is \( 2.5m \) long and \( 1.7m \) wide.

In order to start evaluating the performance of the overall path-following manoeuvre, the transient state of the vehicle approaching the path has to be neglected and the steady state instant \( t^* \) determined while applying the Crossing Cell algorithm. A moving time window of 10 seconds is chosen for identifying the first instant where all points in the time window are classified as close to the reference path. In Figure 4 the transient state is given by the blue line on the right (see also \( t^* \) in Figure 5).

In Figure 4 the vehicle seems to be “far” from the path (red dots) mainly during steering manoeuvres. A measure for evaluating the path-following manoeuvre from \( t^* \) until time \( t \) is given by the percentage of points which are close to the path and defined as
\[
P(t) = \frac{\# \text{ of points close to the path in } [t^*, t]}{\# \text{ of total points in } [t^*, t]}.
\]

Analogously, an online measure \( Q(t) \) of the percentage of non-classified points of \( Y \) is defined. Figure 6 shows the evolution of \( P(t) \) from \( t^* \) to \( t_{total} = 2113 \) and highlights \( P(t_{total}) \) as overall measure of the path-following performance. The green line shows that the number of unclassified points is small and increases, being localised mainly in the left side of Figure 4. The value \( P(t_{total}) = 0.55 \) highlighted in Figure 5, represents a good result in relation to the vehicle’s behaviour during motion. Indeed the path employed in the experiment, that is a sinusoidal-like trajectory, has been selected to stress the guidance system’s performance as it is characterized by a very few straight transects. Usually USV missions require the motion along straight paths, and hence, in real scenarios, a greater value for \( P(t_{total}) \) has to be expected.

With the final aim to introduce the proposed method in online applications, the computation time needed for the simulation example has also been taken into account. In “online application” we intend to take the current vehicle position as the new input for the Crossing Cell algorithm, whilst a polynomial curve \( f = 0 \) approximating/giving the reference has been computed at the start of the experiment or, in case of vehicle-following, can be computed as well online with a small delay depending on the distance between the two vehicles. The computation of the curve \( f_1 = 0 \) by choosing \( \varepsilon_2 = (0.001, 0.001) \) is 320 seconds, while for \( f_2 = 0 \) and \( \varepsilon_2 = (0.005, 0.005) \) the time needed is only 83 seconds. As already mentioned, recalling that the two values of \( adj R^2 \) are very close (and both close to 1), we can agree that the higher value for \( \varepsilon_2 \) is preferable, considering both the accuracy of approximation and the computation time. Note that the most significant value to be taken into account for online applications is the mean value of the time needed for a point in \( Y \) to be classified as “close” or “far” to the reference curve path, or “uncertain”. The mean value from the whole data set \( Y \) is 0.015 seconds (with minimum 0.010s and maximum 0.126s). Considering the common GPS rates (about 4-5Hz), the possibility to employ the algorithm online during field experiments is validated but it is worth noting that the used versions of the two algorithms are far from being optimized. Two different programming languages have been adopted: LPA is implemented in C++ and uses some routines of the GSL-GNU Scientific Library [24], while Crossing Cell algorithm is currently written in the CoCoA language [25], a software for symbolic exact computations with polynomials. This means that, porting the algorithm implementation in C++ or similar language, a strong reduction in the computational time above reported is expected. This topic will be addressed in the near future, as working with a symbolic representation is not relevant for the robotic purposes.
and vehicle-following, to paths with very general shape, in particular not necessarily expressible as univariate functions. For path-following the polynomial $f$ can be computed once for each experiment while for vehicle-following it has to be updated/recomputed during the experiment. The authors are planning to carry out more technical/theoretical work to consider which general characteristics of path following algorithms are fully captured by the proposed metrics, and how this metrics could be related to manoeuvrability characteristics of the vehicle(s).

ACKNOWLEDGMENT

This work has been supported by the EU FP7 MORPH (Marine robotic system of self-organizing, logically linked physical nodes) project, grant agreement no. 288704. The authors thank anonymous referees for the useful remarks.

REFERENCES


