Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 58-20 (2024) 368–373

MPC Path Following with Macroscopic Shape Adjustment for AIAUVs

Ivan Gushkov * **Kristin Y. Pettersen** * **Jan Tommy Gravdahl** *

* *Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim, Norway*

Abstract: This paper treats the problem of adjusting the macroscopic shape of an articulated robotic system following a path, such that the shape fits the path curve as closely as possible. We propose a way of quantifying the fit of the robot's shape to the path through the errors of the individual links, and investigate a kinematic MPC solution to the problem. The MPC generates position and velocity reference trajectories, which respect the constraints of the system, to be fed to a low-level controller. The scheme is evaluated in a kinematic level simulation study with constant irrotational ocean currents.

Keywords: marine applications, path-following, MPC, snake robots

1. INTRODUCTION

Snake robots present a promising new alternative in a variety of industrial and inspection applications (Kelasidi et al. 2016). Featuring an elongated articulated design, they merge features from energy efficient Autonomous Underwater Vehicles (AUVs) and versatile robotic manipulators. When equipped with thrusters, they are usually called Articulated Intervention Autonomous Underwater Vehicles (AIAUVs) Borlaug et al. (2019). It is this particular combination of articulation and actuation redundancy that makes AIAUVs promising for operations in tight confined spaces. Inspection operations at subsea installations or historic shipwrecks demand robots which can thread tight obstacle ridden environments, an application where the articulation of the AIAUV gives it an advantage, see Figure 1.

Moving the robot from A to B is a crucial part of an inspection procedure. Planning algorithms can generate a curve which has been deemed safe from obstacles, and motion along that curve can be achieved through various control strategies, e.g. path-following or trajectory tracking (LaValle 2006). In this work we will consider path-following, where no explicit timing requirements are placed on the motion along the curve.

Path-following has been an active research topic within vehicle control for decades. The case of a straight-line path following with and without constant irrotational currents has well established solutions via the Line-of-sight (LOS) and Integral Line-of-sight (ILOS) methods Xu et al. (2020); Fossen and Pettersen (2014); Kohl et al. (2016). Vector field path-following approaches on the other hand explicitly design a guiding field, see Yao and Cao (2020); Kapitanyuk et al. (2018); Goncalves et al. (2010), obtaining similar stability guarantees as the LOS family of guiding laws. The vector field literature usually treats the path as the 0-level set of a so-called potential function of the variables where the path is defined, avoiding the use of parametrization and treating the path in its purely geometrical form.

A limitation of the standard path-following formulations is that they usually consider a single output of interest, i.e. the position of the center of mass or another point on the vehicle, while other parts of a robot's body do not follow the path, and might thus collide with obstacles in the environment. One way to work around this is at the planning level, where the path is chosen with sufficiently large safety region around it, so that no part of the robot can collide with the environment, so long as the path-following objective is realized. This approach, however, seems overly conservative when working with an articulated robot, as it can adjust its overall, or *macroscopic*, shape to fit the curve and thus be safe from obstacles. This gives rise to the problem treated in this paper, namely path-following control with macroscopic shape adjustment, or more concisely, snake-on-path following.

Adjusting the macroscopic shape of a snake robot to fit a curve has been investigated in the 2D case in Liljebäck et al. (2014a) and the 3D case in Liljebäck et al. (2014b). Said references proposed an algorithmic solution, where the configuration was recovered through an iterative procedure. This approach has the benefit of being computationally efficient, but it does not work with an explicit mathematical formulation of the problem, and its properties are thus difficult to verify. It is desirable to express the deviation of the robot's shape from the curve in a mathematical way, so as to hopefully treat the problem more generally and robustly. Moreover the motivation for shape control in Liljebäck et al. (2014a,b) was gait synthesis and generation of propulsive motion, as they considered terrestrial snake robots that were not equipped with thrusters.

In order to achieve snake-on-path following, while respecting the kinematic structure of the snake robot, we will introduce state constraints to the path-following problem. To this end we will use Model Predictive Control

10.1016/j.ifacol.2024.10.081

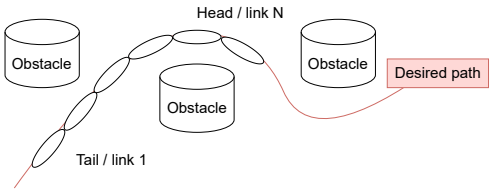

Fig. 1. An example of how the snake-on-path problem can be useful in a cluttered environment.

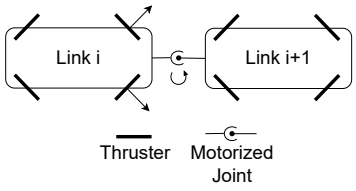

Fig. 2. An example actuator configuration for two generic AIAUV links. The vehicle is equiped with thrusters, which produce linear forces and motorized joints, which can apply torques.

(MPC), as this allows for systematic treatment of system constraints. Use of MPC for the purposes of path-following has been investigated in, among others, Böck and Kugi (2014); Yu et al. (2012); Faulwasser and Findeisen (2015). In Böck and Kugi (2014) the real time viability of a sub-optimal non-linear MPC formulation for path-following was demonstrated in experiments with a crane system. In Yu et al. (2012) the problem is formulated as path-following in the state space of a system, while in Faulwasser and Findeisen (2015) a quite general framework for output path-following is presented. The main benefit of MPC is that state- and input constraints can be treated in a systematic manner. Safety with respect to objects is readily encoded into the MPC problem formulation. Moreover, cost function design provides great freedom in encoding high level specifications, such as macroscopic shape control.

In this paper we will thus use an MPC approach towards achieving path-following control with macroscopic shape adjustment. We note that Sæbø et al. (2024) also presents a method for snake-on-path following. Their method combines a waypoint line-of-sight approach with a closed loop inverse kinematics task priority framework to set desired velocities of each link of the vehicle relative to the path. While our method has the advantage of treating system constraints like joint angle saturation through the MPC framework, the method in Sæbø et al. (2024) is decidedly computationally simpler and the waypoint scheme allows for a flexible definition of path-following scenarios. We will consider a class of paths which can be treated under the framework of vector field guidance. The vector field literature treats paths without the need for parametrization, which offers a particularly convenient expression of the error and guidance law in terms of the system's state variables, making the design of the MPC simpler. In the proposed kinematic level MPC algorithm, motion along the path is only prescribed to the head link, while the rest of the body is tasked to fit its shape to the path. Specifically, the errors of the individual links are used as proxy for the robot's shape fit to the path. A simulation study is conducted to evaluate the viability of the scheme. Finally, limitations are discussed and future improvements proposed.

The rest of the paper is organized as follows. Section 2 presents the working assumptions, choice of coordinates and the mathematical model used in the MPC scheme. Next, section 3 details the design of the kinematic MPC algorithm we propose for the snake-on-path following problem. Section 4 presents a simulation study and discussion of the method, and in Section 5 we present conclusions and future work.

2. ASSUMPTIONS, COORDINATES AND MODEL

2.1 Notation and useful mappings

In this section we introduce notation and matrices which are common in the snake robotics literature. We denote the snake robot's position in the plane $\boldsymbol{p} = (p_x, p_y) \in \mathbb{R}^2$, and the angles of its links w.r.t. the global x-axis by the vector $\boldsymbol{\theta} \in \mathbb{R}^N$. The links are indexed from 1 (tail link) to N (head link). The relative link angles, also called joint angles, are the differences between subsequent link angles, denoted $\boldsymbol{\phi} \in \mathbb{R}^{N-1}$ where $\phi_i := \theta_i - \theta_{i+1}$. The mapping $\sin(\boldsymbol{\theta})/\cos(\boldsymbol{\theta})$ is defined as the $\mathbb{R}^N$ vector of sines/cosines of the link angles $\boldsymbol{\theta}$. We define the matrices $\boldsymbol{S}(\boldsymbol{\theta})/\boldsymbol{C}(\boldsymbol{\theta}) = \mathrm{diag}(\sin(\boldsymbol{\theta}))/\mathrm{diag}(\cos(\boldsymbol{\theta}))$, where $\mathrm{diag}(\cdot)$ is a diagonal matrix. The vector $\boldsymbol{e} \in \mathbb{R}^N$ is a vector of ones. The matrix $R(\alpha) \in \mathcal{SO}(2)$ denotes the 2D rotation matrix with angle α. The matrices $\boldsymbol{A}, \boldsymbol{D} \in \mathbb{R}^{(N-1)\times N}$ are called the addition and difference matrices, respectively, and when multiplied by an N-dimensional vector produce a vector of sums/differences between its adjacent elements, i.e.

$$\boldsymbol{A} = \begin{bmatrix} 1 & 1 & & \\ & \ddots & \ddots & \\ & & 1 & 1 \end{bmatrix} \quad \boldsymbol{D} = \begin{bmatrix} 1 & -1 & & \\ & \ddots & \ddots & \\ & & 1 & -1 \end{bmatrix}. \tag{1}$$

2.2 Assumptions on the snake robot

We consider snake robots moving in a horizontal plane, i.e.

Assumption 1 (Planar snake robot)**.** *We assume that the snake robot is passively and/or actively stabilized to a virtual horizontal plane, i.e. its pitch and roll angles are zero and its depth is held constant.*

As we work with AIAUVs, the robot can be considered to have a fully actuated configuration space. Figure 2 shows a diagram of a possible actuator configuration on two generic links of an AIAUV.

Assumption 2 (AIAUV)**.** *We assume that the configuration space of the robot is fully actutated.*

The path-following problem can thus be treated on the kinematic level, with the goal being generation of desired configuration and velocity trajectories for a low-level controller. Under these considerations we will model the robot on kinematic level in the next section.

2.3 Model and choice of coordinates

The choice of configuration coordinates is important for practical purposes. Since the model we present in this

section will be used to formulate an open-loop MPC optimization problem, we can utilize the freedom that lies in choosing coordinates to achieve a more opportune formulation of the MPC problem. A common set of configuration coordinates for the snake robot is the relative link angles and the pose, i.e. position and orientation, of the tail or head link. The tail and head frame are also referred to as the base and end effector frames, respectively, in the snake robotics literature. This set of coordinates is meaningful when the control problem is formulated at the force/torque level, as the inputs from the motors of the joints enter directly the dynamics of the relative joint angles, and thus they constitute a subspace of the configuration space which is directly and fully actuated by the joint torques. However, we conjecture that for the purposes of *macroscopic shape control* the following set of coordinates makes the problem more interpretable and tractable:

$$\boldsymbol{q} = (p_x, p_y, \theta_1, \cdots, \theta_N) \in \mathbb{R}^{N+2}, \tag{2}$$

where $(p_x, p_y) \in \mathbb{R}^2$ is the position of the center of mass of the snake and $\boldsymbol{\theta} = (\theta_1, \cdots, \theta_N) \in \mathbb{R}^N$ are the angles of the links with respect to the global x-axis. As we will later see, this choice of coordinates results in a very compact model, and can be related to objectives concerning the macroscopic shape of the snake robot.

In the kinematic controller we will consider the relative velocities of the configuration as input, i.e. $\boldsymbol{u} := \dot{\boldsymbol{q}}_r$, which when ocean currents are added results in the following kinematic model:

$$\dot{\boldsymbol{q}} = \boldsymbol{u} + \boldsymbol{V}_{\text{c,gen}}. \tag{3}$$

Here $\boldsymbol{V}_{\text{c,gen}}$ denotes the generalized ocean currents given in the global frame. We will consider the case of constant irrotational currents, i.e. $\boldsymbol{V}_{\text{c,gen}} = (V_{c,x}, V_{c,y}, 0, \cdots, 0) \in \mathbb{R}^{N+2}$. Note that for the path-following objective we will define later, we want to also recover the positions and velocities of the links in the world frame. These are solved for via forward kinematics (Liljebäck et al. 2013):

$$\boldsymbol{X} = -l\boldsymbol{K}^T \cos(\boldsymbol{\theta}) + \boldsymbol{e}p_x \tag{4a}$$

$$\boldsymbol{Y} = -l\boldsymbol{K}^T \sin(\boldsymbol{\theta}) + \boldsymbol{e}p_y. \tag{4b}$$

Where $\boldsymbol{K} = \boldsymbol{A}^T(\boldsymbol{D}\boldsymbol{D}^T)^{-1}\boldsymbol{D}$. Differentiation of (4) w.r.t. time gives the linear velocities of the links

$$\dot{\boldsymbol{X}} = l\boldsymbol{K}^T\boldsymbol{S}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} + \boldsymbol{e}\dot{p}_x \tag{5a}$$

$$\dot{\boldsymbol{Y}} = -l\boldsymbol{K}^T\boldsymbol{C}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} + \boldsymbol{e}\dot{p}_y. \tag{5b}$$

3. PATH-FOLLOWING MPC FORMULATION

In this section we propose and design an MPC controller on the kinematic level to solve the problem of macroscopic shape path-following. Section 3.1 gives an overview of how the modules in the algorithm fit together. Section 3.2 deals with the design of a guidance law for the path-following. We note that the guidance scheme is not the main contribution of the paper, as we simply need a working guidance law for the path-following. Section 3.3 contains the main contribution of the paper, namely the MPC design. The proposed solution is not unique, in the sense that *macroscopic shape fit* to the path is a high level concept and can be quantified in many ways. MPC is a natural fit for this problem, as the cost function can be engineered to encode a high level specification, and the generated trajectories will respect the system's kinematic constraints.

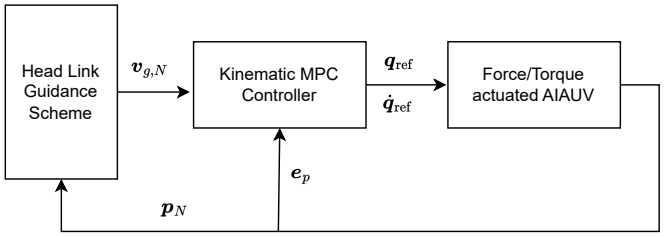

Fig. 3. A block diagram giving overview of the modules in the scheme. Here $\boldsymbol{e}_p$ is the vector of path-following errors of the individual links, while $\boldsymbol{p}_N$ is the position of the head link and $\boldsymbol{v}_{g,N}$ is the desired velocity of the head link.

3.1 Overview of the algorithm

Here we present an overview of the different modules of the algorithm and how they fit together. A block diagram of the scheme is given in Figure 3. The MPC receives as inputs the *path-following error signals* of the different links, as well as a *desired velocity reference* for the head link only. The desired velocity reference is generated by a guidance module. Often, the guidance law is used to generate heading or course references. However, we will treat the guidance law as a direct velocity vector command, which will fit neatly into the formulation of the MPC problem in section 3.3. Note that from the MPC's perspective, the choice of guidance scheme is arbitrary, so long as a desired reference velocity vector is provided. The outputs of the MPC module are reference configuration and configuration velocity trajectories, which can be given to an arbitrary force and torque level controller for the AIAUV.

3.2 Guidance design

For the guidance we will consider a vector field guidance law similar to the ones in Yao and Cao (2020) and Gushkov et al. (2023). We introduce the generic $\mathbb{R}^2$ variable $\boldsymbol{\xi} = (x, y)$. Note that this should not be confused with the position of the snake robot's Center of Mass (CoM) $\boldsymbol{p}$, as the path-following objective will not be defined for the CoM. The path in this guidance paradigm is defined as the sub-level set of a $\mathcal{C}^2$ so-called potential function $\Phi : \mathbb{R}^2 \to \mathbb{R}$:

$$\mathcal{P} = \{\boldsymbol{\xi} \in \mathbb{R}^2 \mid \Phi(x, y) = 0\}. \tag{6}$$

The function Φ essentially encodes the path, and can be chosen as $\Phi = y - ax$ for straight lines, $\Phi = y - A\sin(x)$ for sinusoids, $\Phi = y^2 + x^2 - R^2$ for circles etc. The goal of vector field path-following is to engineer a guidance field whose integral curves converge to the path, i.e. $\lim_{t\to\infty} \text{dist}(\boldsymbol{\xi}, \mathcal{P}) \to 0$, and once on the path they traverse the path for all future time, i.e. $\boldsymbol{\xi}(t_0) \in \mathcal{P} \Rightarrow \boldsymbol{\xi}(t) \in \mathcal{P}\ \forall t \geq t_0$. Note that the function Φ can be interpreted as a signed distance from the path under some technical conditions. These are omitted in this paper for the sake of brevity and readability, but are somewhat trivially fulfilled for the paths we will consider, see assumptions 1-3 in Yao and Cao (2020) for details. The gradient of Φ is given as

$$\boldsymbol{\nabla}\boldsymbol{\Phi} = \left[\frac{\partial\Phi}{\partial x}, \frac{\partial\Phi}{\partial y}\right]^T, \tag{7}$$

and we consider the path-following error function to be given by Φ itself

$$e := \Phi(\boldsymbol{\xi}) \tag{8a}$$
$$\dot{e} = \boldsymbol{\nabla\Phi}^T\dot{\boldsymbol{\xi}}. \tag{8b}$$

As shown in Gushkov et al. (2023) and Yao and Cao (2020), under these considerations, a vector field given by

$$\boldsymbol{\chi}(\boldsymbol{\xi}) = h\mathbf{R}(\tfrac{\pi}{2})\boldsymbol{\nabla\Phi}(\boldsymbol{\xi}) - ke(\boldsymbol{\xi})\boldsymbol{\nabla\Phi}(\boldsymbol{\xi}) \tag{9}$$

solves the path-following problem kinematically. Here h is either 1 or -1 and determines the direction the path is to be traversed in, and k is a gain giving how aggressive the guidance scheme is. We will now show a simple Lyapunov argument for the stability of the path-following error when a generic position output $\boldsymbol{\xi}$ evolves according to the guidance field, i.e. $\dot{\boldsymbol{\xi}} = \boldsymbol{\chi}$. For a complete formal proof which also considers the saturation of the vehicle's velocity, see Gushkov et al. (2023). Choosing the Lyapunov function candidate $V = \frac{1}{2}e^2$ and finding its derivative along the solutions of (8b) with $\dot{\boldsymbol{\xi}} = \boldsymbol{\chi}$ given by (9), we obtain

$$\dot{V} = e\dot{e} = e\boldsymbol{\nabla\Phi}^T\boldsymbol{\chi} \tag{10a}$$
$$= -k\|\boldsymbol{\nabla\Phi}\|^2e^2. \tag{10b}$$

The derivative of V is negative definite so long as $\boldsymbol{\nabla\Phi}$ does not vanish anywhere in $\mathbb{R}^2$, and the origin of (8b) is thus Globally Uniformly Asymptotically Stable (GUAS). The non-vanishing of $\boldsymbol{\nabla\Phi}$ is essentially a condition on the path, and is fulfilled for straight lines and sinusoidal paths. The guidance vector in (9) will be used to generate a desired reference trajectory for the head link's position output. The scheme's immediate benefit is that (9) constitutes an expression only dependent on the position of the head link, which in turn can be related to the states of the system through (4) and thus easily encoded into the MPC.

3.3 MPC design

In this section we design the open loop optimization problem, which makes use of the model in (3) and uses the guidance vector in (9) in order to solve the snake-on-path following problem for the path in (6).

We design the cost function to include the errors of the links, and encode a velocity prescription for the head link. We define $\boldsymbol{e}_p(\boldsymbol{X}, \boldsymbol{Y}) \in \mathbb{R}^N$ a vector containing the *path-following errors* of the links of the snake robot. Here we will use *error-like* functions defined as in (8a), i.e. potential functions, instead of the real distance to the path. The reason for this is that these functions are less computationally demanding to evaluate online than the actual distances, and are moreover a closed-form expression of the states, meaning they can be readily included into the MPC problem. Further, define $\dot{\boldsymbol{p}}_n = (\dot{X}_N, \dot{Y}_N)$ as the velocity of the head link, i.e. the last components of the vectors defined in (5). Lastly, define $\boldsymbol{v}_{g,N} = \boldsymbol{\chi}_{|\xi=\boldsymbol{p}_n}$ as the guidance vector (9) evaluated at the head link's position. We are now ready to present the design of the MPC problem

$$\min_{\boldsymbol{q},\boldsymbol{u}} \int_{t_k}^{t_k+T} L(\boldsymbol{q}(\tau), \boldsymbol{u}(\tau))d\tau + E(\boldsymbol{q}(t_k+T)) \tag{11a}$$
$$\text{s.t.} \quad \dot{\boldsymbol{q}} = \boldsymbol{u} + \hat{\boldsymbol{V}}_{c,gen}(t) \tag{11b}$$
$$\boldsymbol{\phi}_{\min} \leq |\boldsymbol{D\theta}| \leq \boldsymbol{\phi}_{\max} \tag{11c}$$
$$\boldsymbol{u}_{\min} \leq |\boldsymbol{u}| \leq \boldsymbol{u}_{\max} \tag{11d}$$
$$\boldsymbol{q}(t_k) = \boldsymbol{q}(t). \tag{11e}$$

The problem minimizes the cost functional composed of the integral of the stage cost L over the prediction horizon, and the terminal cost E, which take the following forms

$$L(\boldsymbol{q}, \boldsymbol{u}) = \boldsymbol{e}_p^T\boldsymbol{Q}\boldsymbol{e}_p + \boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u} + w_N\|\dot{\boldsymbol{p}}_N - \boldsymbol{v}_{g,N}\| \tag{12a}$$
$$E(\boldsymbol{q}) = \boldsymbol{e}_p^T\boldsymbol{Q}\boldsymbol{e}_p. \tag{12b}$$

The $\boldsymbol{Q}$ and $\boldsymbol{R}$ matrices are diagonal weight matrices for the link errors and inputs, respectively, and w_N is a weight for the deviation of the head link's velocity vector from the guidance vecotor (9). The last term in the cost function, E, is a terminal cost on the errors. In practice, snake robots always have saturation on the joint angles and their velocities which have to be respected, something which is readily accommodated for by constraints in the MPC framework. Specifically, the (linear) constraint in (11c) ensures that the relative angle constraints are satisfied, while the constraint on the input (11d) encodes a saturation on the system's velocities. The kinematic model from (3) is encoded in the MPC's dynamic constraint (11b), and initial condition for the optimization problem are encoded in (11e). It is assumed that an estimate of the initial condition can be recovered from a measurement or a filter at each time step. Note that in the dynamic constraint (11b), the term $\hat{\boldsymbol{V}}_{c,gen}$ represents an estimate of the ocean currents, which are assumed constant for the length of the prediction horizon.

Let us now interpret the cost function and how the MPC is intended to solve the problem. The term $w_N\|\dot{\boldsymbol{p}}_N - \boldsymbol{v}_{g,N}\|^2$ in the stage cost penalizes the deviation of the velocity of the head from the guidance vector $\boldsymbol{v}_{g,N}$. The guidance vector is thus interpreted as a *desired velocity vector* of the head. The head thus leads the way, and drags the rest of the snake robot with it. The other term in the stage cost is a quadratic form in the path-following errors of the rest of the links, minimization of which implies that the body fits to the path.

Remark 1. *Note that here we have used the path-following error as a proxy for the fit of the snake robot to the path. This is clearly not the only metric we could have used. Another representation of the fit could be to minimize the errors at both ends of each link, or to minimize the absolute value of the area between the link and the path. These would, however, clearly, result in more complicated MPC problem formulations, with higher computational demands.*

4. SIMULATION STUDY SETUP AND RESULTS

The simulation study is performed in Python, where the kinematic model (3) is integrated with a RK4 scheme. The MPC problem is implemented with CasADi's Opti class (Andersson et al. 2019), where the equality constraints of the model (11b) are discretized with a RK4 scheme and the *ipopt* solver is used to solve the open loop optimization problem [1]. We have considered the case of a sinusoidal path with $\Phi(x, y) = y - \sin(x)$ which we recently investigated as an applications of snake robots, see Gushkov et al. (2023), where the snake robot is tasked to converge to a path between vortices in the wake of a bluff body. The linear components of the ocean currents were set to $\boldsymbol{V}_c = (-0.2, 0.1)$ m/s, and the controller was tested

[1] The reader is invited to run our code, which can be found at (http://tinyurl.com/pkp6nxcx), in order to retrieve the simulation parameters or reproduce the results.

with ideal knowledge of the ocean currents, i.e. $\hat{\boldsymbol{V}}_c = \boldsymbol{V}_c$. The position was initialized at $(-1, -1)$ m and all the link angles at zero. For the main simulation results the MPC prediction horizon in (11a) was set to $T_{\text{mpc}} = 2$ s with discretization $\delta_{\text{mpc}} = 0.1$ s. The guidance gain was set to $k = 2.7$ and the tuning parameters for the MPC were chosen as:

$$\boldsymbol{Q} = \text{diag}(1, 1, 1, 1, 1, 15), \qquad w_N = 1 \tag{13a}$$

$$\boldsymbol{F} = \text{diag}(1, 1, 1, 1, 1, 15) \tag{13b}$$

$$\boldsymbol{R} = \text{diag}(0.01 \cdot \boldsymbol{I}_2, \frac{30\pi}{180}\boldsymbol{I}_4). \tag{13c}$$

The constraints for the link angles were set to 25 °, and the constraints of the link angle velocities to 40 °/s. The linear velocities were constrained to 1 m/s each. Figure 4 shows the relative angles and the state velocities generated by the scheme, while Figure 5 shows the individual link errors. Note that there are several instances where both the position and velocity constraints are active, but not violated. Figure 6 shows a visualization of the scheme in action, where it can be seen that the shape of the robot fits to the path over the simulation. This validates the design of the MPC cost function, showing that the link errors are a good choice of metric for shape fit. Moreover the choice of coordinates in (2) results in the non-linearities showing up in the cost function, rather than in the constraints. With $T_{\text{mpc}} = 2$ s and $\delta_{\text{mpc}} = 0.1$ s, the average solution time was around 0.28 s. While this is slower than what a force/torque level controller usually runs at, it should be noted that the MPC outputs desired trajectories over a horizon, and in a practical setting one can use several steps of the MPC solution while one waits for a new solution, or employ a sub-optimal scheme, as in Böck and Kugi (2014). Alternatively the horizon can be reduced for a quicker solution. Table 1 compares the solution times and errors for different horizon lengths, and it can be seen that the solution speed can be increased by an order of magnitude, while the error remains reasonable. Interestingly, the benefit of increasing the horizon diminishes quite quickly after 0.5 s. On the other hand the benefit of a longer horizon is seen in the number of active constraints over the simulation.

In the simulation above we assumed ideal knowledge of the ocean currents. In order to investigate the scheme's robustness w.r.t. the quality of the current estimate, two more cases were investigated: stochastic sampling of the current estimate and the case of no knowledge of the ocean currents. For the stochastic case, the estimate was drawn from a normal distribution with mean equal to the real currents, i.e. $\hat{\boldsymbol{V}}_c \sim \mathcal{N}(\boldsymbol{V}_c, 0.04^2 \cdot \boldsymbol{I}_2)$. The estimate was drawn at each step and held constant for the length of the MPC horizon. No knowledge of the ocean currents corresponds to the case where $\hat{\boldsymbol{V}}_c = 0$. Table 1 shows statistical data from the various cases. Unsurprisingly, the performance of the scheme deteriorates with the quality of the current estimate, but the stochastic case is still quite close to the case of a perfect estimate, indicating a level of robustness towards uncertainty in the kinematics, and that including an estimator in the loop is a viable strategy.

Table 1. Statistical data for some cases of interest. In the first part of the table the ocean current is assumed known. In the second part the horizon length used is $T_{\text{mpc}} = 2$ s.

Case	Avg. link error	Avg. solution time [s]	Nr. active rel. angle constraints
$T_{\text{mpc}} = 0.2$	0.00703	0.0193	51
$T_{\text{mpc}} = 0.5$	0.00336	0.0536	35
$T_{\text{mpc}} = 1$	0.00294	0.1237	20
$T_{\text{mpc}} = 1.5$	0.00296	0.1872	14
$T_{\text{mpc}} = 2.5$	0.00302	0.3489	12
Known current	0.00299	0.2779	12
Stochastic current est.	0.00359	0.2516	13
Unknown current	0.01928	0.2754	14

5. CONCLUSIONS AND FUTURE WORK

This paper has proposed an MPC framework for generating trajectories for an AIAUV performing path-following while adjusting its shape to the path, what we call the *snake-on-path following* problem. The simulations indicate a) that the individual link errors are a viable metric for representing the snake robot's shape fit to a path b) that the generated closed loop trajectories of the MPC indeed offer a viable solution to the problem, even in the case of limited knowledge of the ocean currents. The simulations also highlighted some interesting features of the scheme, namely that increasing the horizon beyond a certain point has diminishing returns in the sense that it results in higher computational times with no tangible benefit to error reduction. We conjecture that for this problem, the relevant part of the solution does not lie very far off into the future, and so predicting on a shorter absolute time horizon with finer discretization should be investigated. In future work the scheme should also be tested in experiments or simulation studies where the dynamics of the robot are considered. Extension beyond kinematics can be done either by giving the generated MPC trajectories as references to low level controllers, or by reformulating the MPC directly at the force/torque level, including a dynamic model in the optimization problem. Computing suitable terminal ingredients such that stability of the closed loop is guaranteed should also be investigated.

ACKNOWLEDGEMENTS

This result is part of a project that has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, through the ERC Advanced Grant 101017697-CRÈME. The work is also supported by the Research Council of Norway through the Centres of Excellence funding scheme, project No. 223254 – NTNU AMOS.

REFERENCES

Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., and Diehl, M. (2019). CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1), 1–36.

Borlaug, I.L.G., Pettersen, K.Y., and Gravdahl, J.T. (2019). Tracking control of an articulated intervention auv in 6dof using the generalized super-twisting algorithm. In *Proc. 2019 American Control Conference*, 5705–5712. Philadelphia, USA.

Böck, M. and Kugi, A. (2014). Real-time nonlinear model predictive path-following control of a laboratory tower crane. *IEEE Transactions on Control Systems Technology*, 22(4), 1461–1473.

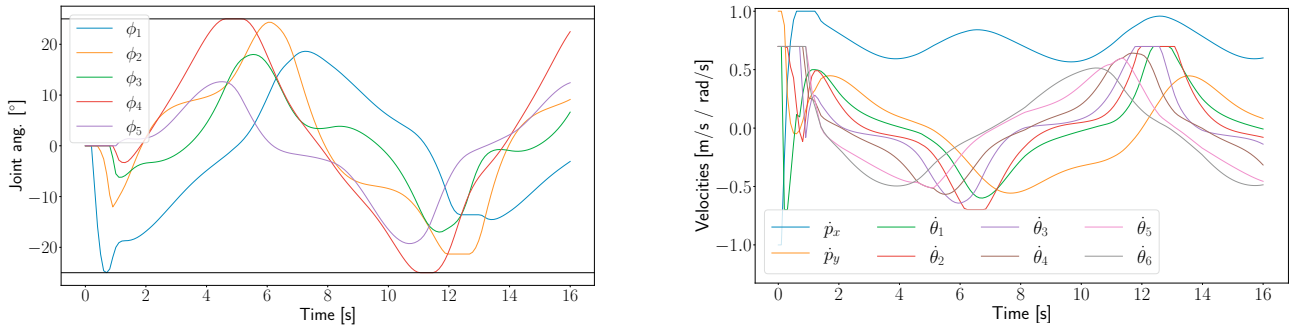

Fig. 4. The relative joint angles $\boldsymbol{\phi}$ (left) and configuration velocities $\dot{\boldsymbol{q}}$ resulting from the kinematic MPC scheme (right).

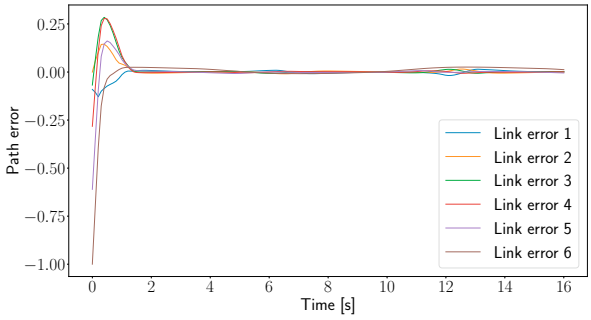

Fig. 5. Closed-loop path errors for the links.

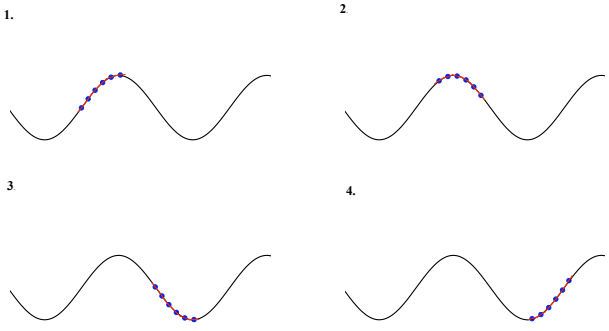

Fig. 6. Snapshots of the AIAUV and path at various points in the simulation of the scheme. The blue dots are the positions of the links, while the red lines represent the links themselves. The snapshots are taken rougly at times $t = 3, 6, 10, 13$ s

Faulwasser, T. and Findeisen, R. (2015). Nonlinear model predictive control for constrained output path following. *IEEE Transactions on Automatic Control*, 61, 1026–1039.

Fossen, T.I. and Pettersen, K.Y. (2014). On Uniform Semiglobal Exponential Stability (USGES) of Proportional Line-of-Sight Guidance Laws. *Automatica*, 50(11), 2912–2917.

Goncalves, V.M., Pimenta, L.C.A., Maia, C.A., Dutra, B.C.O., and Pereira, G.A.S. (2010). Vector Fields for Robot Navigation Along Time-Varying Curves in n -Dimensions. *IEEE Transactions on Robotics*, 26(4), 647–659.

Gushkov, I., Orucevic, A., Pettersen, K.Y., Yao, W., and Gravdahl, J.T. (2023). Vector Field Path Following of Static Sinusoidal Paths for Underwater Snake Robots. In *Proc. 2023 IEEE Conf. Control Tech. Applications*. Bridgetown, Barbados.

Kapitanyuk, Y.A., Proskurnikov, A.V., and Cao, M. (2018). A Guiding Vector-Field Algorithm for Path-Following Control of Nonholonomic Mobile Robots. *IEEE Transactions on Control Systems Technology*, 26(4), 1372–1385.

Kelasidi, E., Liljebäck, P., Pettersen, K.Y., and Gravdahl, J.T. (2016). Innovation in underwater robots: Biologically inspired swimming snake robots. *IEEE Robotics & Automation Magazine*, 23(1), 44–62.

Kohl, A.M., Pettersen, K.Y., Kelasidi, E., and Gravdahl, J.T. (2016). Planar path following of underwater snake robots in the presence of ocean currents. *IEEE Robotics and Automation Letters*, 1(1), 383–390.

LaValle, S.M. (2006). *Planning Algorithms*. Cambridge University Press.

Liljebäck, P., Pettersen, K.Y., Stavdahl, Ø., and Gravdahl, J.T. (2013). *Snake Robots*. Springer London.

Liljebäck, P., Pettersen, K.Y., Stavdahl, Ø., and Gravdahl, J.T. (2014a). Compliant control of the body shape of snake robots. In *Proc. 2014 IEEE International Conference on Robotics and Automation*. Hong Kong, China.

Liljebäck, P., Pettersen, K.Y., Stavdahl, , and Gravdahl, J.T. (2014b). A 3d motion planning framework for snake robots. In *Proc. 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Chicago, USA.

Sæbø, B.K., Pettersen, K.Y., and Gravdahl, J.T. (2024). Kinematic task-priority path following for articulated marine vehicles. *In Proc. 2024 IFAC Conf. Control Applications in Marine Systems, Robotics and Vehicles*.

Xu, H., Fossen, T.I., and Guedes Soares, C. (2020). Uniformly semiglobally exponential stability of vector field guidance law and autopilot for path-following. *European Journal of Control*, 53, 88–97.

Yao, W. and Cao, M. (2020). Path following control in 3D using a vector field. *Automatica*, 117, 108957.

Yu, S., Li, X., Chen, H., and Allgöwer, F. (2012). Nonlinear model predictive control for path following problems. In *4th IFAC Conference on Nonlinear Model Predictive Control*. Noordwijkerhout, Netherlands.