MANOEUVRE-AUTOMATON BASED MOTION PLANNING AND TRAJECTORY TRACKING OF AN AUTONOMOUS MARINE VEHICLE

W. Naeem [1] **, C. S. Tan***, **R. Sutton and J. Chudley**

{wnaeem, rsutton, jchudley}@plymouth.ac.uk

*chiew-seon.tan@power.alstom.com

Marine and Industrial Dynamic Analysis Research Group
The University of Plymouth, Drake Circus
Plymouth PL4 8AA, UK

Abstract: This paper presents a motion planning and trajectory tracking strategy for an autonomous vehicle operating in an obstacle populated environment. The vehicle's trajectory is generated using a quasi-random based manoeuvre automaton (MA) approach whilst the tracking autopilot selected is a genetic algorithm based model predictive controller. The MA method, which is based on quantisation of the system dynamics, is used here to enhance the rapid-exploring random tree (RRT) strategy. Later, the resulting trajectory is employed as a guidance system for the vehicle to manoeuvre safely around the obstacles to reach the final destination. Simulation results are shown to demonstrate the efficacy of the proposed approach.

Keywords: Autonomous vehicles, manoeuvre automation, rapid-exploring random tree, model predictive control, genetic algorithms and optimization.

1. INTRODUCTION

Recently, there is a significant increase in the deployment of autonomous underwater vehicles (AUVs) in missions such as mine-hunting, reconnaissance operations, sea bed exploration etc. This development has resulted in an urgent demand to improve the autonomy of the pertinent system. Whilst a remotely operated vehicle is guided and controlled by a trained human operator, an AUV has its own guidance and control processors onboard and thus capable of carrying out missions without any constant external intervention. Several inherent characteristics of an AUV particularly its highly nonlinear coupled dynamics, underactuated, non driftless, nonminimum phase behaviour and its subjection to unpredictable exogenous disturbances make controller design nontrivial. The last attribute is especially relevant for small, lightweight AUVs such as *Remus* (Prestero, 2001).

Typically, a few linearised models are utilised for the AUV controller design, thus artificial operational constraints must be imposed to avoid violation of the linearity assumption. Consequently, this introduces additional restriction on the system performance envelope. Alternatively, a more preferable approach is to quantise the AUV dynamics, transforming a continuous dynamic model governed by complex nonlinear differential equations into a hybrid model which not only possesses higher levels of abstraction but is also more beneficial computationally. With an added advantage, this approach also conveniently permits the incorporation of complex and aggressive manoeu-

[1] corresponding author

vres into the AUV. The process is achieved via the manoeuvre automation (MA) representation (Frazzoli *et al.*, 1999).

Moreover, most path planning techniques introduced to date are based firmly on deterministic methods and graph searches. Unfortunately, due to sample space discretisation issues, the generated trajectories frequently require extra smoothing and interpolation. Notwithstanding this, in some cases, the system dynamics are neglected in order to avoid the state explosion effect. Collectively, these factors ensue a very conservative system performance. Randomisation methods are becoming popular as they are inherently more robust to the state explosion effect. One version of the randomised algorihtm is the RRT (LaValle, 1998). The feasibility of employing RRT algorithm for AUV obstacle avoidance purpose has been studied by Tan *et al.* (2004). In this paper, a hybrid RRT which has been combined with MA representation is employed (Tan, 2005). This trajectory is then used as an input to a genetic algorithm based model predictive controller (GA-MPC) for tracking purposes and the performance is evaluated in simulations.

The organisation of the paper is as follows. Section 2 briefly describes the MA approach including AUV dynamics whereas Section 3 outlines the motion planning algorithm. Whilst the controller design is presented in Section 4 and simulation results are demonstrated in Section 5. Finally, concluding remarks are made in Section 6.

2. THE MANOEUVRE AUTOMATION APPROACH

The MA, a form of finite state machine, was proposed by Frazzoli *et al.* (1999) as a unified framework for formalising the control of nonlinear systems with symmetries. In essence, the main idea is to generate a complete trajectory via sequential combination of the copies of motion primitives from a library set. These motion primitives are extracted from the vehicle in an open loop mode.

The approach relies primarily on two different types of motion primitives: trim trajectories and manoeuvres. Trim trajectories correspond to steady state behaviour *in situ* when the velocities in body-axes and inputs are constants. Whilst manoeuvres can be seen as finite time motion primitives that interconnect two trim trajectories together.

Similar to a differential or difference equation, the MA transcription describes a dynamic system, differing only in that it has hybrid elements in both its control inputs and state vector. At each particular moment, the system is constrained to be either in a trim condition or performing a manoeuvre.

The MA representation allows one to express easily the optimal motion planning problem. This is true for certain cost functions that share the symmetry properties of the system, such as minimum time, minimum length and minimum control effort. Herein the time optimization criterion is adopted which is solved using the linear programming method (Frazzoli, 2002*a*).

2.1 AUV Dynamic Model and MA Implementation

In this section, the AUV model and techniques for synthesising the motion primitives are presented. This is done, in order to convert the continuous system model into the MA representation. The nonlinear AUV model was supplied by the QinetiQ, based on the *Autosub* vehicle (Millard *et al.*, 1998), which has a torpedo shaped hull. Dimensionally, the vehicle is $7m$ long and approximately $1m$ in diameter and has a nominal displacement of $3600kgs$. In this paper, the model is restricted to only latitude dynamics whilst yaw control is limited to the bow rudders.

Figure 1 depicts a possible MA representation of the *Autosub* dynamics. Both $2m/s$ and $5m/s$ cruising speeds are illustrated. However, the following simulations are constrained to only one speed regime of $2m/s$. The *Autosub* model is discretised using the zero-order-hold method and the sampling frequency set to $10Hz$. Figure 1 clearly shows a library that constitute manoeuvres of 15^0, 30^0, 60^0 and 120^0. The opposite manoeuvres are assumed to be identical. The manoeuvres are chosen so that they encompass the important performance envelope of the AUV, and their generation can be attained via a human operator or a controller input. Herein, the latter method is selected by designing a simple PD autopilot and the input and state histories are recorded.

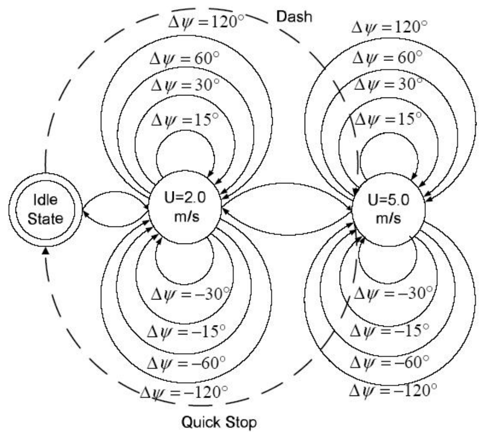

Fig. 1. AUV dynamics in MA representation

The next section outlines the steps of the proposed motion planning strategy.

3. MOTION PLANNING ALGORITHM

Frazzoli (2002*b*) advocated enhancing the RRT algorithm by fusing it with the MA to solve motion planning problems with obstacles. The algorithm assumes that one has an embedded planner that can plan an optimal trajectory in an obstacle free environment between two arbitrary states. The approach employed in this paper is that of multiple nested nodes, since every state in a trim trajectory can be considered as a starting point of a manoeuvre. This algorithm generates child nodes at every connection point between a trim trajectory and manoeuvre. This improves the RRT branching capability by increasing the probability of finding a solution. A brief explanation of this enhanced algorithm with reference to Figure 2 is outlined below. The reader is referred to Tan (2005) for a comprehensive treatment.

(1) Check to see if a direct connection to the goal from the initial states based on the minimum time criterion is possible. If this is attained then the algorithm terminates.
(2) Failing that, generate a subgoal, R1 using a quasi-random generator and attempt to connect to it using the embedded planner, again based on the minimum time objective function.
(3) If there is no collision, then generate an edge with new vertices at all inter-connecting points of trim trajectories and manoeuvres. Explicit connection to the goal is attempted from all the new vertices (Greedy algorithm). If this is successful, the algorithm terminates.
(4) If failed, generate another random subgoal, R2. Sort the shortest time trajectories from all vertices to R2 in an ascending order and attempt to connect to R2. Apply this to only the first few near-optimal trajectories to avoid vertices saturation. In this algorithm, only the first 3 near-optimal trajectories are stored and tested.
(5) The whole process is repeated from 1 to 4 until a feasible trajectory to the final state is found, the maximum vertices size or the time limit is reached. Figure 2 shows that vertex, nc1 has connected successfully to the final state.

Once the trajectory is generated, it is fused with an autopilot to produce the required performance.

4. GENETIC ALGORITHM BASED MODEL PREDICTIVE CONTROL

The GA-MPC algorithm was first proposed by Duwaish and Naeem (2001) for chemical processes identified as Hammerstein and Wiener models. This was later modified and implemented in the *Hammerhead* AUV in real time (Naeem *et al.*, 2005). The GA-based controller uses the process model to search for the control moves, which satisfy the process constraints and optimizes a cost function. The cost function to be minimised here is given by Equation 1.

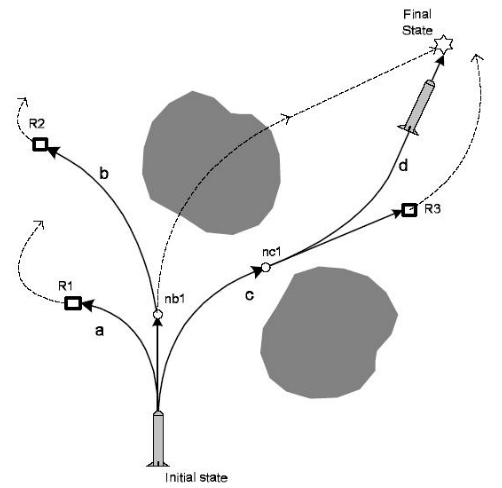

Fig. 2. Simplified illustration of the RRT algorithm

$$\begin{aligned} J = &\sum_{i=1}^{H_p} e(k+i)^T Q e(k+i) \\ &+ \sum_{i=1}^{H_c} \Delta u(k+i)^T R \Delta u(k+i) \\ &+ \sum_{i=1}^{H_p} u(k+i)^T S u(k+i) \end{aligned} \tag{1}$$

subject to,

$$\begin{aligned} u^l &\leq u(k+i) \leq u^u \\ \Delta u^l &\leq \Delta u(k+i) \leq \Delta u^u \end{aligned}$$

where the superscripts l and u represents the lower and upper bounds respectively. H_p and H_c denotes the prediction and control horizons respectively and Q is the weight on the prediction error,

$$e(k) = \hat{y}(k) - w(k) \tag{2}$$

where $w(k)$ is the reference or the desired setpoint obtained from the guidance mechanism and $y(k)$ is the predicted process output, also shown in Figure 3. R and S are weights on the change in the input Δu and magnitude of the input u respectively. Adjusting the input weighting matrices could add damping to the closed loop control system.

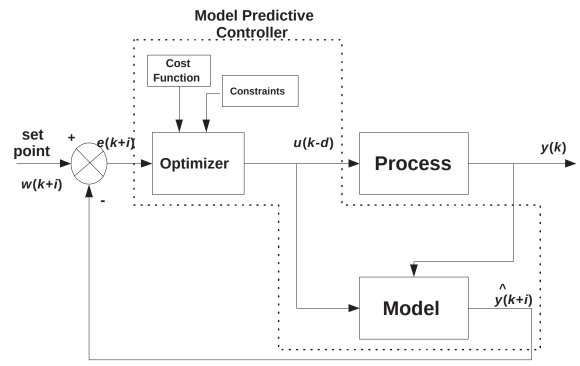

Fig. 3. Model predictive controller block diagram showing the optimizer, cost function and constraints as part of the controller

The following steps describe the operation of the GA-based MPC algorithm and the flow chart of the algorithm is shown in Figure 4. At every sample time k

(1) Evaluate process outputs using the process model.
(2) Use a GA search to find the optimal control moves which optimize the cost function and satisfy process constraints. This can be accomplished as follows.
    (a) generate a set of random possible control moves. The control moves or population consists of real values which is reasonable in a real world environment. The width of the population represents H_p and is doubled for every input added to the process.
    (b) find the corresponding process outputs for all possible control moves using the process model over H_p.
    (c) evaluate the fitness of each solution using the cost function and the process constraints. The fitness function used here is given by

$$fitness = \frac{1}{1+J} \tag{3}$$

    where J is the cost function given by Equation 1.
    (d) apply the genetic operators (selection, crossover and mutation) to produce new generation of possible solutions. Herein, single point crossover is used for parents mating whereas roulette wheel strategy is employed for selection procedure.
    (e) repeat until predefined number of generations is reached and thus the optimal control moves are determined.
(3) Apply the optimal control moves generated in step 2 to the process.
(4) Repeat steps 1 to 3 for time step $k+1$.

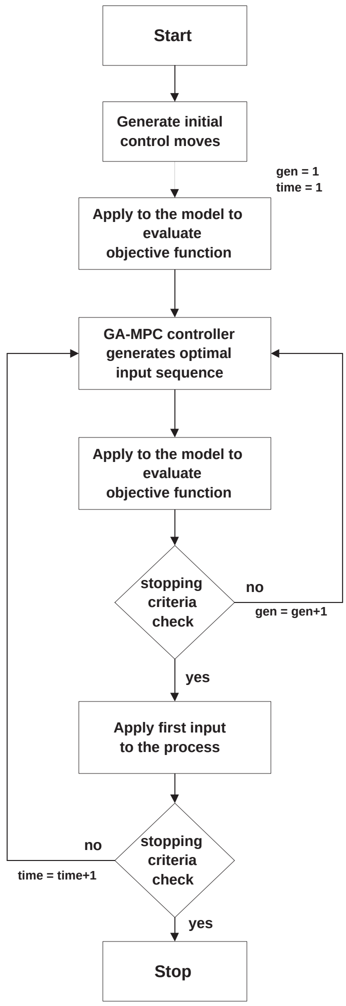

Fig. 4. Flow chart of the GA-based MPC

4.1 Constraints Formulation

Constraints represent limitations on different physical quantities involved in a process. For instance, the input or output of a certain process is restricted beyond a specified value due to economical or environmental reasons or the input cannot be changed abruptly due to the hardware dynamics. One of the most powerful and distinguishing features of MPC is its ability to handle constraints in a natural way during the controller design at every sample time. Generally, two types of constraints are considered in controller design. Soft constraints are employed in the cost function as a penalty factor and can be violated to fulfil some other criteria. On the other hand, hard constraints represent physical limitations on actuators and cannot be violated. Herein, hard constraints are placed on the input variables whereas a penalty factor is introduced to the rate of change of input in the cost function. Since the population in a GA represents the input variable, therefore, hard input constraints are implemented by generating random initial population in the desired range.

5. SIMULATION RESULTS

As mentioned earlier, the proposed algorithm is simulated on the *Autosub* dynamic model cruising at 2 m/s. The objective here is to demonstrate the trajectory generation and path following by the AUV in an obstacle populated environment. The obstacles could be visualised as buoys and other boats near a docking terminal where the AUV returns at the end of a mission for battery recharging and data downloading purposes.

The vehicle launching coordinates are assumed to be at $(0, 0)$ whereas the target is located at $(140, 170)$. The obstacles are placed with their geometric centres at $(20, 50)$, $(-5, 220)$, $(50, 220)$, $(75, 120)$, $(75, 200)$, $(75, 160)$.

The autopilot tuning parameters are provided in Table 1 and the simulated output trajectory is plotted in Figure 5 showing both the desired path and the actual trajectory of the vessel. Clearly, the vehicle's trajectory coincides with the desired path which was generated using the MA-RRT approach. The manoeuvres shown in this figure were obtained from the library set depicted in Figure 1.

Table 1. GA-MPC tuning parameters for LOS tracking for all cases

Controller Parameters	Value
Q	1
R	0.5
S	0.0
H_p	15
H_c	1
Mutation prob.	0.01
Crossover prob.	1
No. of generations	5
Population size	100
Insertion rate	0.5

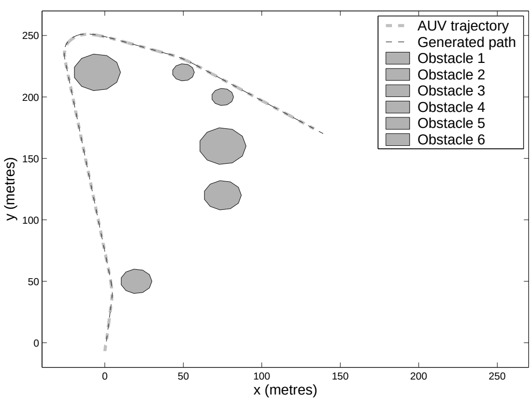

Fig. 5. Desired trajectory and actual AUV path

Figure 6 shows the heading response of the vehicle. In practice, this could be obtained from on an onboard compass. Except the small lag between the actual and the desired yaw angles, the AUV closely follows all the heading changes.

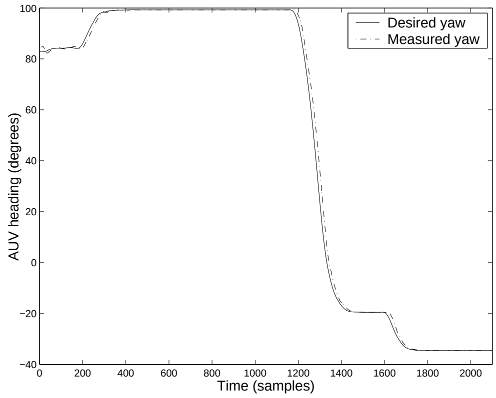

Fig. 6. Desired and actual AUV heading angle

Also shown here is the control effort in Figure 7 in terms of rudder angle which was constrained between $\pm 25^0$. The rudder is positioned in the centre corresponding to 0^0 when the vehicle is moving in a straight line. A maximum deflection of 20^0 is observed when changing the heading angle from 100^0 to -20^0.

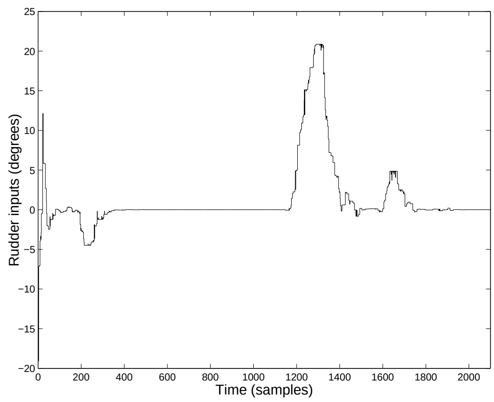

Fig. 7. Optimal rudder inputs generated by the autopilot

6. CONCLUSIONS

This paper integrates a motion planning algorithm with a genetic algorithm based autopilot. The path planning strategy is based on the MA approach combined with a RRT methodology. Whilst the autopilot selected is a model predictive controller which has already been tested in real time by the authors in the *Hammerhead* AUV. Simulation results are demonstrated showing a feasible trajectory generated by the guidance system and the simulated output of the vehicle.

REFERENCES

Duwaish, H. and W. Naeem (2001). Nonlinear model predictive control of hammerstein and wiener models using genetic algorithms. In: *Proceedings of the 2001 IEEE In-*

ternational Conference on Control Applications (CCA01). IEEE. Mexico City, Mexico. pp. 465–469.

Frazzoli, E. (2002*a*). Manoeuvre-based motion planning and coordination for single and multiple uavs. In: *Proc. of AIAA/IEEE Digital Avionics Systems Conference 2*. AIAA/IEEE. pp. 8D31–8D312.

Frazzoli, E. (2002*b*). Quasi-Random Algorithms for Real-Time Spacecraft Motion Planning and Coordination. *Acta Astronautica 53* pp. 485–495.

Frazzoli, E., M. A. Dahleh and E. Feron (1999). Hybrid control architecture for aggressive maneouvring of autonomous helicopters. In: *Proceedings of The IEEE Conference on Decision and Control 3*. IEEE. pp. 2471–2476.

LaValle, S.M. (1998). *Rapidly-exploring Random Trees: A New Tool for Path Planning*. World Wide Web. `http://janowiec.cs.iastate.edu/papers/rrt.ps`.

Millard, N. W., G. Griffiths, G. Finegan, S. D. McPhail, D. T. Meldrum, M. Pebody, J. R. Perrett, P. Stevenson and A. T. Webb (1998). Versatile Autonomous Submersibles - The Realising and Testing of a Practical Vehicle. *Underwater Technology* **23**(1), 7–17.

Naeem, W., R. Sutton, J. Chudley, F. R. Dalgleish and S. Tetlow (2005). An online genetic algorithm based model predictive control autopilot design with experimental verification. *International Journal of Control* **78**(14), 1076–1090.

Prestero, T. (2001). Verification of a Six-Degree of Freedom Simulation Model for the REMUS Autonomous Underwater Vehicle. Master's thesis. MIT. Massachusetts, USA.

Tan, C. S. (2005). A Collision Avoidance System for Autonomous Underwater Vehicles. PhD thesis. School of Engineering. University of Plymouth, Plymouth, UK.

Tan, C. S., R. Sutton and J. Chudley (2004). An incremental stochastic motion planning technique for autonomous underwater vehicles. In: *IFAC Conference on Control Applications in Marine Systems (CAMS 2004)*. IFAC. Ancona, Italy. pp. 483–488.