Localisation and policy synthesis for underwater swarming autonomous vehicles with probabilistic guarantees about safe exploration and reachability requirements

Ada Alevizaki
Centre for Doctoral Training in Autonomous Intelligent Machines and Systems
University of Oxford
antigoni.alevizaki@eng.ox.ac.uk

Abstract—This project is concerned with the case of an autonomous underwater vehicle (AUV) submerging in deep ocean to locate and inspect underwater infrastructure, i.e. a pipeline. For this task, proper decision-making and positional accuracy are key, both for getting closer to infrastructure whilst keeping safe and for recording data more accurately. The underwater environment raises significant difficulties to these requirements, due to signal attenuation, multipath fading as well as the presence of extreme and unpredictable currents. To address the aforementioned challenges, we estimate that better localisation can be achieved if the receiver nodes (which in this case are also AUVs floating on the surface) stay in an equilateral triangle formation. On the other hand, to synthesise the appropriate policy that satisfies the requirements for safe exploration and reachability of the pipeline, we employ a reinforcement learning (RL) inspired control policy, that satisfies the required properties, which we express as linear temporal logic (LTL) formulae. The performance of our method is satisfactory and evaluated through several test case scenarios, where different parameters involved are taken into account.

Index Terms—autonomous underwater vehicles, localisation, localisation error, markov decision process, linear temporal logic, limit-deterministic buchi automata, product MDPs, Q-learning

I. INTRODUCTION

Autonomous underwater vehicles (AUVs) are unmanned, untethered, subsea vehicles. They are equipped with sensors, actuators and appropriate intelligence algorithms, in order to carry out missions without human operator interference or support from a ship. Common missions include pipe assessment, cable inspection, seabed exploration, surveying etc. [1].

In this context, BP is moving to fully autonomous inspection of the subsea pipelines and infrastructure using AUVs endowed with a range of sensors [2]. The AUVs explore the underwater environment and call for intervention when identifying a problem. For this task, proper decision-making and positional accuracy are key, both for getting closer to infrastructure and for recording data more accurately. However, underwater wireless communication channels are subject to harsh and unique environmental conditions and hence often display severe attenuation characteristics, time-varying multipath fading, limited bandwidth and power resources of acoustic devices, limiting both the localisation accuracy and the estimation of obstacles’ position [3]. Moreover, the presence of extreme and unpredictable currents in deep ocean environments, combined with the complex non-linear hydrodynamic behaviour of the AUVs can affect their intended actions, either by diverting them from their heading direction or by requiring additional time to complete a specified task [4].

In this project, we try to address the aforementioned challenges, in the context of an AUV exploring a predefined underwater area in quest of locating and inspecting an existing pipeline. We propose a simple approach aiming to keep the AUV localised and the localisation error bounded, as well as a reinforcement learning (RL) inspired control policy, that satisfies the required properties, which we express as linear temporal logic (LTL) formulae; exploration of the underwater environment and reachability and inspection of the pipeline.

The rest of the report is organised as follows: in section II, we present the necessary theoretical background, in order for a reader to comprehend the concepts and terminologies introduced later in this report. In section III we explain our proposed approach and implementation details, while in section IV we present the results of our method through different test-case scenarios. Finally, in section V we recapitulate on our work, discuss its current limitations and suggest potential future directions.

II. THEORETICAL BACKGROUND

A. Localisation

Location knowledge of nodes in a network is essential for many tasks; geo-routing, data-centric storage or collaborative signal processing rely on knowing geographic locations of sensor nodes or physical phenomena of interest. When GPS or other device assisted location technology is not available due to shadowing or cost, it is important to develop alternative localisation approaches. The methods employed can vary from range free to range based and from single to multi-stage, depending on the number of hops between the reference node and the receiver to be localised [5]. In this project, we will focus on range-based localisation techniques, which are
more common in the underwater environment. A wide variety of sensors is employed to assist the localisation problem, obtaining different information about the received signal, such as the signal strength, time of arrival, time difference of arrival and angle of arrival. The received signal strength indicator (RSSI), though very effective in terrestrial missions, is not very common in the underwater set-up our problem lies, due to multipath fading and signal attenuation, so we will not discuss this any further. In the rest of this section, we will present the rest of the aforementioned sensors, as well as the respective localisation algorithms employed in each case.

1) Time of arrival (ToA) or Time of Flight (ToF): The time-of-arrival (ToA) method performs ranging based on the relationship among transmission time, speed and distance. It is estimated as the difference between the received time of an acoustic signal (according to local clock measurements) and the transmission time (according to global time measurements through GPS), or vice versa [6]. This subtle point of using both local and global measurements implies the requirement for time synchronisation between network nodes, which can make the design of the network more complicated.

Knowing the ToA and the propagation speed of the acoustic signal, we can estimate the equivalent distances through basic physics laws, i.e.:

\[
d = cT,
\]

where \( d \) is the calculated range measurement, \( c \) is the propagation speed of the acoustic signal and \( T \) is the ToA, as computed by the sensors. The node then estimates its position, according to the intersection of various circles centred at each reference node with radii corresponding to the range measurements. In general, to localise a node in \( d \)-dimensional space, the number of independent range measurements required should be at least \( d + 1 \) [7].

For our problem, where we need to localise an AUV in 3D space, we should normally require 4 independent range measurements. AUVs however are equipped with high accuracy pressure sensors and altimeters, whose measurements translate to depth information. As a result, we only have to solve the equivalent 2D problem, which is the solution to the following set of equations:

\[
(x - x_1)^2 + (y - y_1)^2 + d^2 = d_1^2,
\]

\[
(x - x_2)^2 + (y - y_2)^2 + d^2 = d_2^2,
\]

\[
(x - x_3)^2 + (y - y_3)^2 + d^2 = d_3^2,
\]

where \( x_i, y_i \) are the \( x \) and \( y \) coordinates of the reference nodes, \( d \) is the depth of the node to be localised and \( d_i \) are the calculated range measurements from node \( i \), with \( i = 1, 2, 3 \).

2) Time Difference of Arrival (TDoA): The time-difference-of-arrival (TDoA) sensors compute distances by translating the difference between the measured time of arrival of acoustic signals received from a pair of reference nodes [8], to the difference in range estimates of those reference nodes. As a result each estimate of time-difference gives rise to a hyperbola for the unknown position of the node. A unique estimate of the submersibles position can be obtained by intersecting two such hyperbolas. However, this technique requires that beacons transmit at near concurrent times. Otherwise, the hyperbolas no longer intersect at a common point, requiring for additional approaches, such as least squares, to obtain an estimate for the position.

Knowing the TDoA, we can calculate distances using multilateration. Similarly to the ToA case, we can solve the 2D equivalent problem. Supposing we have \( N + 1 = 3 \) receivers at known locations \( P_i, i = 0, ..., N \) with \( P_0 \) located at the origin of our coordinate system, we can calculate the distances \( R_i \) from the transmitter to the \( i^{th} \) receiver:

\[
R_i = \sqrt{(x - x_i)^2 + (y - y_i)^2 + d^2},
\]

where \( x_i, y_i \) are the \( x \) and \( y \) coordinates of the reference nodes and \( d \) is again the depth of the AUV. If \( c \) is the propagation speed of the acoustic signal and \( \tau_i \) is the time difference of the acoustic signal arriving at receivers 0 and \( i \), we have:

\[
c\tau_i = R_i - R_0.
\]

From (5), (6) we obtain the following set of linear equations:

\[
x(\frac{2x_i}{c\tau_i} - \frac{2x_1}{c\tau_1}) + y(\frac{2y_i}{c\tau_i} - \frac{2y_1}{c\tau_1}) + c\tau_i - c\tau_1 - \frac{x_i^2 + y_i^2}{c\tau_i} + \frac{x_1^2 + y_1^2}{c\tau_1} = 0.
\]

3) Angle of Arrival (AoA): The angle-of-arrival (AoA) sensors are used to indicate the direction of the signal sources. The AoA is defined as the angle between the propagation direction of an incident wave and some reference direction, which is known as orientation. Orientation, defined as a fixed direction against which the AOAs are measured, is represented in degrees in a clockwise direction from the North. When the orientation is 0° or pointing to the North, the AOA is absolute, otherwise, relative [9]. Basic idea of AoA is that the delay of the signal arriving at two antennas varies corresponding to the direction of the transmitter. Thus by measuring the phase difference between received signals of two antennas we can recover the direction of the coming signal. That is, if we know the angles of a signal received at two receivers with known positions, the position of the unknown node can be calculated as the third point of a triangle with one known side (the distance between the two receivers) and two known angles (the AoAs) [10]. In our set-up, the AoA method requires triangulation over three receivers’ pairs, in order to obtain the 3D position of the AUV as the intersection of three circles.

B. Formal Verification

Formal verification of a computing system entails a mathematical proof showing that the system satisfies its desired property or specification. This requires some mathematical structure to model the system of interest and derive its desired properties as theorems about the structure. In this project, we employ Markov Decision Processes (MDPs) as the mathematical structure that models the system, and express the desired properties in Linear Temporal Logic (LTL), a modal logic
able to express rich logical time-dependent properties such as safety and liveness. An LTL property can be represented as an automaton. In the following sections, we will discuss the notions of MDPs and LTL, as well as some automata details that are relevant to our problem formulation.

1) Markov Decision Processes (MDPs): An MDP is a tuple \( M = (S, \bar{s}, A, P, R) \) where:
- \( S \) is a set of states;
- \( \bar{s} \in S \) is an initial state;
- \( A \) is a set of actions;
- \( P : S \times A \rightarrow \text{Dist}(S) \) is a partial probabilistic transition function, with \( \text{Dist}(S) \) being the set of discrete probability distributions over \( S \);
- \( R = (R_S, R_A) \) is a reward structure, where \( R_S : S \rightarrow \mathbb{R} \) is a state reward function and \( R_A : S \times A \rightarrow \mathbb{R} \) is an action reward function.

An MDP represents the evolution of a system exhibiting both probabilistic and nondeterministic behaviour through states from the set \( S \). Each state \( s \in S \) of \( M \) has a set \( A(s) = \{ a \in A \mid P(s, a) \text{ is defined} \} \) of available actions, the choice of which is nondeterministic. If action \( a \in A(s) \) is chosen, then the probability of moving to state \( s' \) from \( s \) is \( P(s, a)(s') \). The aforementioned rewards are accumulated when passing through states or state-action pairs; more specifically, a reward of \( R(s) \) is accumulated when passing through state \( s \), and a reward of \( R(s, a) \) is accumulated when taking action \( a \) while being in state \( s \) [11].

2) Stationary Policy: Having defined the MDP, we define a stationary randomised policy \( \text{Pol} : S \times A \rightarrow [0, 1] \) as a mapping from each state \( s \in S \) and action \( a \in A \) to the probability of taking action \( a \) in state \( s \). A deterministic policy is a degenerate case of a randomised policy which outputs a single action at given state, that is \( \text{Pol}(s) \in A(s) \).

3) End Component: An end component of an MDP \( M = (S, \bar{s}, A, P, R) \) is a directed graph induced by a pair \( (S, A) \) such that it is strongly connected [12].

4) Maximal End Component (MEC): And end component \( (S, A) \) of an MDP \( M = (S, \bar{s}, A, P, R) \) is maximal if there exists no end component \( (S', A') \) such that \( (S, A) \not\subseteq (S', A') \) and \( S \subseteq S' \) and \( A_s \subseteq A_s' \) for all \( s \in S \) [12].

5) Linear Temporal Logic (LTL): Propositional linear temporal logic (LTL) is a logical formalism that is suited for specifying linear temporal properties [12]. The elementary temporal modalities that are present in most temporal logics include the operators \( \Diamond \) ("eventually") and \( \Box \) ("always"), which translate as "eventually in the future" and "now and forever in the future" respectively. Combination of these two modalities leads to new, dual temporal modalities; \( \Box \Diamond \phi \) means "infinitely often \( \phi \)", while \( \Diamond \Box \phi \) means "eventually forever \( \phi \)".

The set of LTL properties is defined over a set of atomic propositions \( AP \) as follows:
\[
\phi ::= \text{true} \mid a \in AP \mid \phi \wedge \phi \mid \neg \phi \mid \Box \phi \mid \phi_1 \cup \phi_2,
\]
where the operators \( \Box \) and \( \cup \) translate as "next" and "until" respectively. More specifically, \( \Box \phi \) indicates that at the next timepoint property \( \phi \) will be satisfied, while \( \phi_1 \cup \phi_2 \) indicates that property \( \phi_1 \) holds at all times until we reach a time where property \( \phi_2 \) is satisfied.

6) Semantics of LTL: If \( \phi \) is an LTL formula over \( AP \), then the linear temporal property induced by \( \phi \) is:
\[
\text{Words}(\phi) = \{ \sigma \in (2^AP)^\omega \mid \sigma \models \phi \} \tag{9}
\]

From the above it follows that the semantics of an LTL formula could logically be determined with respect to a transition system (TS). A TS is a tuple \( TS = (S, Act, \rightarrow, I, AP, L) \), from which we can derive that MDPs are also a form of TS. Thus, we can define LTL formulae interpreted over MDPs. More specifically, using an LTL formula we can specify a set of properties (that have to be satisfied) over the sequence of states that are generated by a policy in the MDP.

Since the set of associated words of an LTL formula \( \phi \), \( \text{Words}(\phi) \) can often be difficult to obtain, we can use an alternative method to identify it using a limit-deterministic Büchi automaton [13].

7) Limit-deterministic Büchi Automata (LDBAs): To better understand the concept of LDBAs, we will begin by defining the notion of Generalised Büchi Automata (GBA). A GBA \( \mathcal{N} = (Q, q_0, AP, F, \Delta) \) is a structure where \( Q \) is a finite set of states, \( q_0 \in Q \) is the set of initial states, \( AP \) is a finite alphabet, \( F = \{ F_1, ..., F_j \} \) is the set of accepting conditions where \( F_j \subseteq Q, 1 \leq j \leq f \) and \( \Delta : Q \times 2^AP \rightarrow 2^Q \) is a transition relation. If \( \Sigma = 2^AP \) and \( \Sigma^\omega \) is the set of all infinite words over \( \Sigma \), we allow an infinite word \( \omega \in \Sigma^\omega \) to be accepted by \( \mathcal{N} \) if there exists an infinite run \( \theta \in \Sigma^* \) starting from \( q_0 \) where \( \theta[i+1] \in \Delta(\theta[i], \omega[i]), i \geq 0 \) and for each \( F_j \subseteq F \), \( inf \omega \cap F_j \neq \emptyset \). \( inf \omega \) is the set of states that are visited infinitely often in the sequence \( \theta \).

Following the above, a GBA \( \mathcal{N} = (Q, q_0, AP, F, \Delta) \) is an LDBA if \( \mathcal{N} \) can be partitioned into two disjoint sets \( Q = Q_N \cup Q_D \) such that:
- \( \Delta(q, a) \subseteq Q_D \) and \( |\Delta(q, a)| = 1 \) for every state \( q \in Q_D \) and for every \( a \in AP \),
- for every \( F_j \subseteq F, F_j \subseteq Q_D \).

8) Product MDP: In order to create a model that incorporates both the system-environment interaction modelled by the MDP and the LTL properties expressed by the LDBA, [14] proposes the notion of a product MDP, upon which a policy can be synthesised to satisfy both the environment and the LTL property. Given an MDP \( M = (S, \bar{s}, A, P, R) \) and an LDBA \( \mathcal{N} = (Q, q_0, \Sigma, F, \Delta) \) with \( \Sigma = 2^AP \), the product MDP can be constructed as \( \mathcal{M} \otimes \mathcal{N} = (S^\otimes, A, S_0^\otimes, P^\otimes, AP^\otimes, L^\otimes) \), where:
- \( S^\otimes = S \times Q \);
- \( S_0^\otimes = (s_0, q_0) \);
- \( AP^\otimes = Q \);
- \( L^\otimes = S \times Q \rightarrow 2^Q \) such that \( L^\otimes(s, q) = q \);
- \( P^\otimes : S^\otimes \times A \times S^\otimes \rightarrow [0, 1] \) is the transition probability function such that \( P^\otimes((s, a, q_j), (s, a, q_j')) = P(s, a, q_j) \).

Over the states of the MDP, we also define accepting condition \( F^\otimes \) where \( F_j^\otimes \subseteq F_j \), \( 1 \leq j \leq f \) is a subset of \( S^\otimes \) such that for each \( s^\otimes = (s, q) \in F_j^\otimes, q \in F_j \).
C. Q-Learning

Reinforcement learning (RL), also known as dynamic programming, is an area of machine learning, where the agents try to learn from their environment by taking actions that are coupled with some cumulative reward. RL is associated with the notion of exploration vs. exploitation, that is finding the balance between uncharted territory and current knowledge [15]. Usually, in problems where RL is employed, the environment is modelled as an MDP whose exact mathematical model is unknown [16]. In III, we will see how this notion will be useful to the formulation of our problem.

Q-learning is a sub-class of RL, where the reward function of the MDPs is unknown, in the sense that the rewards for all state-action pairs are initialised with an arbitrary common value. As the system starts taking random actions and receiving rewards, the Q function is updated as follows [15]:

\[
\begin{align*}
Q(s, a) &\leftarrow Q(s, a) + \\
& + \mu \left[ R(s, a) - Q(s, a) + \gamma \max_{a'} Q(s', a') \right], \quad (10)
\end{align*}
\]

III. PROPOSED APPROACH

The task we are addressing in this project includes constant localisation of an AUV, that submerges into deep ocean to inspect a pipeline in the underwater infrastructure. For this mission, we assume no stationary anchor points; instead, we employ a swarm of four AUVs in total, that have been classified in two categories depending on their role [17]; one AUV submerges and explores the underwater environment in search of the pipeline, which it aims to inspect fully once found. The remaining three AUVs stay on the surface at known positions, so as to act as stationary known nodes, since they are GPS enabled. Their task is to localise the submerged AUV whilst keeping the localisation error bounded. In the remaining of this report, we will refer to the surface AUVs as “vessels” or “localisers”, to avoid confusion with the submerged vehicle, which we will call an “AUV” or “explorer”.

In terms of the localisation task, we will consider the following scenario. The localisers will remain stationary on the surface, trying to localise the AUV that moves underseas, using the ToA method. The Internal Measuring Unit (IMU) of the AUV is also taken into account, to ensure that the proposed location is meaningful with respect to the previous known location of the AUV. The localisers always remain in an equilateral triangle formation, as this ensures that the localisation error is minimised [18]. The AUV’s initial position lies at the centre of mass of this triangle.

Also, we assume that all communication between the two groups of vehicles takes place when they are both stationary, to avoid the doppler effect at the transmission of the signal.
$Q(s_{obs}, a)$ values, based on the AUV’s current observations, where $s_{obs}$ indicates the observed states that are identified either as hazardous or as pipe. Thus, once an unsafe state is identified, all the rewards for the state-action pairs leading to that state are assigned with a negative value (penalty), while once the pipe appears in the observation area, all the rewards for the state-action pairs guiding the AUV to the pipe are assigned with a positive value (reward). Following these updates, the action that has the maximum Q-value for the current state is chosen and the $Q(s, a)$ of the current state is updated with (10).

It should be noted that all sensors obtain range measurements, which are very noisy in deep sea environment. As a result, calculated distances include error, which can be dangerous when trying to take an action in the vicinity of an obstacle. In order to account for this, we make sure that all the surrounding area of the identified danger are also avoided.

Similarly, in order to ensure that no part of the pipe is missed during inspection, we handle the $Q(s, a)$ values of the state-action pairs of the pipe as follows: once we reach the pipe, we first identify the neighbouring states that also belong to the pipe, and choose randomly one of them as the next state. After this first step, we favour all states in this half-plane that include the pipe, until we reach the end of the pipe towards this side. This is effectively accounted in our product automaton, that rewards all actions that will keep the AUV on the pipe, but we need to ensure that the neighbouring states of the pipe are also rewarded so that the AUV can return to the pipe in case of deviation because of e.g. an unexpected current. Following, the AUV turns a 180°, and tries to repeat the last sequence of actions in reverse, which entails the appropriate $Q(s, a)$ values update. This now guarantees that the AUV will constantly move towards the unexplored end of the pipe, unless a sudden disturbance that takes it further away from its predefined path appears.

Algorithm 1 summarises the aforementioned approach.

IV. EXPERIMENTS AND RESULTS

In this section, we will first begin with a general description of the implementation description of this project. More details will be added where necessary in the different test-cases we present later in this section. All modules of this project are implemented in MATLAB.

The inspection area has dimensions $1500 \times 1500 \times 1515$. The initial location of all vehicles is on the surface, with the AUV located at the centre of the inspection area. The localisers surround it in an equilateral triangle formation, of which the AUV is the centre of mass. The speed of all AUVs is assumed constant $v = 0.76 m/sec$.

The vehicles communicate by exchanging acoustic signals. The AUV always broadcasts its position so to call for intervention at the proper location if necessary. The propagation speed of acoustic signals is assumed to be constant, $c = 1520 m/sec$. We also assume the existence of multipath fading, that results to $0.5 dB$ loss every time the signal is reflected in the bottom. The transmitter of the AUV operates at $f_c = 20 kHz$, emitting a rectangular waveform. We assume that the projector is isotropic, i.e. it has the same response in all directions and similarly, that the receiver consists of an isotropic hydrophone.

In order to model the aforementioned characteristics, we make use of the Phased Array System Toolbox, that includes models for transmitters and receivers, underwater propagation, moving targets and platforms, jammers, and clutter. Since at least one of our vehicles is moving, this toolbox gave us the required tools to model the communication of acoustic signals to serve the localisation demands.

No special toolbox is required for the LCRL part of the project. However, it demands a transformation from continuous to discrete space. Movement and localisation takes place in the continuous space, but to study the decision making, we need to discretise the space. As a result, we discretise the space appropriately to a $300 \times 300 \times 303$ grid. This grid represents the MDP that models the environment. As mentioned earlier in II, we construct the product MDP between the MDP and the LDBA and run the LCRL algorithm, updating the Q-values where necessary where a danger or a target appears.

In all the above cases, we use maps of the environment,
constructed as follows: except for a special-case scenario, we 
assume the pipe to be a straight line in the bottom of the 
inspection area. Obstacles are also included near the seabed, 
in the shape of pyramids with dimensions $3 \times 3$ grids for the 
base and $3$ grids for the height. The position and the number 
of the obstacles is selected randomly for each map, varying 
from $1\%$ to $1\%$ of the total number of a 2D grid.

In the following sections, we present different test cases 
based on the outlined set up and discuss their results re-
spectively. The maps of the environments upon which our 
algorithm was tested are shown in Fig. 2, Fig. 3 and Fig. 
4.

1) Unobscured straight pipe: In our first experiment, we 
evaluate our algorithm to the simplest case, where the pipe is 
completely unobscured. For this experiment, we assume the 
length of the equilateral triangle of the localisers to be $l = 
1500$.

As it can be seen in Fig. 5, localisation accuracy is perfect 
for the vertical movement, when the AUV submerges. This 
is because the AUV remains in the vertical line going trough 
the centre of the equilateral triangle formed by the localisers, 
ensuring that the signal is received at equal times by all of 
them. This is the case for all the experiments, so we will not 
discuss this any further in the following cases.

Fig. 6 shows the navigation policy chosen by the AUV and 
the location estimate in black and green, respectively. As it 
can be seen, the policy synthesised by the AUV fully satisfies 
the required properties of keeping safe, locating the pipe and 
following the pipe fully. Localisation accuracy however is not 
as effective: since the odometry measurements are normally 
very noisy, a large number of random movements may lead 
to false corrections, resulting to the localisation error being 
accumulated. Fig. 7 summarises the above mentioned; the 
localisation error is acceptable when the number of iterations 
in small, but it can increase drastically when then number of 
iterations for the experiment to be completed increases.

2) Effect of length of equilateral triangle: For this ex-
periment, we evaluate the performance of our method over 
four different lengths of triangle, $l = 500$, $l = 1000$ and 
$l = 1500$. Fig. 8 shows that in terms of navigation, all cases 
perform very similarly, covering the whole length of the pipe. 
In terms of localisation, it appears that a length of $l = 500$ 
performs much better than the rest of the cases. This is because 
larger lengths of triangle, even though they might ensure 
that the receivers are well separated, increase the distance 
between the transmitter and the receiver, resulting to more
noise in range measurements and thus to distance estimates. Unfortunately, we did not realise this soon enough, so the rest of our experiments run with \( l = 1500 \). The error distance in this case can be seen in Fig. 9.

3) Behaviour of AUV on obscured straight pipe: In this case, we examined the ability of the AUV to inspect the whole pipe, even if it is partially obscured by obstacles. The navigation policy is still very effective, going around the obstacles until the next part of the pipe appears where necessary.

4) Behaviour of AUV on angular pipe: In this scenario, we forget our assumption of the pipe being straight. We create a special map of the environment, where the pipe curves at one position, and we evaluate the ability of the AUV to follow it. As it can be seen in Fig. 11, the AUV perfectly identifies the angle and follows it until it reaches the end, and then effectively moves in the reverse direction.

V. CONCLUSIONS

In this project, we developed an algorithm for localising an AUV that submerges into deep ocean to inspect underwater infrastructure and simultaneously synthesising the appropriate navigation policy, that keeps the AUV safe and ensures that the targeted pipeline is reached and fully inspected. We tested the performance of our algorithm through different test case scenarios. The synthesised policy is very satisfactory. The constraints are satisfied in all test-case scenarios: the AUV keeps safe, reaches the pipe and then traverses the whole pipe. Unfortunately, the localisation accuracy still remains a problem that we will have to work on and improve. It seems that odometry corrections are not generally a very wise choice, since these measurements are very noisy. We should thus search for alternative methods to handle the localisation error that appears due to the noisy range measurements.

Though we did address a few of the factors that are important in our problem set up, there are still questions that we did not discuss in this project. We will refer to some of them as issues that could be addressed in future work.

First and foremost, we did not account for energy constraints. Given that we have developed our LCRL policy synthesis algorithm to be very risk-averse when taking actions (moving only one grid at a time), and demanded all vehicles to be stationary when engaging in communication, it is most probable that the AUV consumes much more energy than it is actually required. Relaxation of these two factors would be necessary if energy consumption was to be taken into account. The algorithm though would have to become more complex: movement at the time of exchange of acoustic links requires taking into account the doppler effect in order to ensure accurate localisation, while moving across further distances could increase the risk of not avoiding obstacles appropriately.

Also, similar to the assumption that each action only moves the AUV along 1 grid, we assumed that the currents will only
move the AUV in 1 grid-distance, which keeps it relatively safe unless the probability of currents to occur is rather high. Should this be relaxed, the choice of action should be adapted appropriately to try to avoid dangers as much as possible.

Lastly, it would be much more efficient if more than one AUVs submerged for two main reasons; first, the exploration task to locate the pipe would be split among them and thus it should, logically, be completed faster. Second, having 3 AUVs inspecting the pipe would give us more accurate information of the pipe, since we could have 3D visual data. These two scenarios would require a multi-agent approach, which would be rather interesting since the agents should exchange more information than in our simpler set-up. Also, an iterative localisation algorithm should be employed, to ensure that the localisation error remains bounded and minimised, since it could only be 1 AUV on the surface to know its exact location through GPS, resulting to the localisation error being accumulated.

ACKNOWLEDGEMENTS

This project was offered by BP in the context of mini projects carried out by students of the Centre for Doctoral Training in Autonomous Intelligent Machines and Systems at the University of Oxford. We would very much like to thank the BP representatives, Mr. Joe Little, Senior Technology Innovation Manager, Mr. Richard Bailey, Senior Process Engineer and Mr. Peter Collinson, Global Environmental Response Expert, for their time and guidance.

REFERENCES


