Particle Diversity Reduction for AUV's Active Localisation

Francesco Maurelli and Yvan Petillot
Ocean Systems Laboratory
School of Engineering and Physical Sciences
Heriot-Watt University
EH14 4AS Edinburgh, Scotland (UK)
Email: {f.maurelli;y.r.petillot}@hw.ac.uk

***Abstract*— An Autonomous Underwater Vehicle (AUV) needs to demonstrate a number of capabilities, in order to carry on autonomous missions with success. One of the key areas is autonomous localisation, i.e. the capability of the AUV to estimate correctly its position and orientation in the environment. However, most of the proposed approaches are "passive", with no robot motion control involved. The "active" localisation incorporates the control of the robot motion, as a way to improve the robustness of the AUV localisation, finding the best path to follow in order to reduce the uncertainty in the state estimation. Representing the vehicle's belief of the state with particles, the active localisation module is triggered when there is a clear grouping in the particles and it produces as an output the path to be followed in order to reduce the uncertainty. Both simulation results and tank trials have shown the advantages of using this technique compared to the classical ones.**

I. Introduction

Day after day, the number of operations at sea has significantly increased in the last few years. The increased interest is related to many different fields, such as exploration, exploitation of resources (e.g. oil underwater infrastructures), security (e.g. harbour protection). The current vehicles used safely and routinely in the off-shore industry are the so-called ROVs, Remotely Operated Vehicles. They have no intelligence as they are driven by a human pilot, connected through an umbilical cable. The limitations are many and they rely both on the complete lack of autonomy and on the insfrustructures needed to operate with those vehicles. Autonomous Underwater Vehicles are addressing these problems as they do not need a human pilot and the costly infrastructures required by the ROVs, being able to perform more complex missions, due to the absence of a connecting cable between the vehicle and the mother ship. Being underwater intelligent vehicles, some of the research fields are similar to the ones for ground or aerial intelligent vehicles. The peculiar characteristics of the environment are different, so different sensors and different strategies, according to the constraints of the underwater world need to be used. A key area for intelligent vehicles is to correctly estimate the state in the operational environment, i.e. the capability of the AUV to estimate its position and orientation. The problem is usually known as *autonomous localisation*. Localisation techniques are required for many underwater applications involving autonomous and semi-autonomous robots. They are used, for example, to perform docking tasks, in order to determine the relative state of the vehicle with respect to the docking station or to navigate around underwater structures, for inspection or intervention. For both survey-class and intervention-class AUVs, a localisation system is a prerequisite for almost every task. Many approaches have been proposed to address the AUV localisation problem. However, most of the approaches are *passive*. There is no robot motion control involved. They are based exclusively on the analysis of the sensor data (usually motion estimation and measures of the environment). The *active* localisation incorporates the control of the robot motion, as a way to improve the robustness and the efficiency of the process. The key difference is that in this case there is the research of the best path to be followed, in order to reduce the uncertainty, rather than just trying to evaluate what the sensors are streaming. The authors want to clarify that with *active* localisation, they refer to the vehicle being *active*, i.e. capable of taking decisions and controlling actively the motion. It is not related to localisation with *active* features (e.g. active beacons, lights), like in [1].

Related Work

Most of the proposed approaches to localisation are *passive*, as the estimation problem has been considered in most of the cases totally unrelated to the robot control problem, which assumes usually a knowledge of the robot position. However, there is some work done in the field of *active* localisation, mainly in land robotics.

[2] proposed an active choice of the landmark to be observed. However, this approach does not solve the global localisation problem, but helps tracking the position of the robot. If the uncertainty grows, a specific module is called for global localisation, but there is no specific strategy to discriminate between different poses.

This idea is similar to the one proposed in [3], where the landmarks are actively selected, by a supervisor module. This approach is unsuitable for our purposes, both because it is landmark-based and because it does not address the initial pose estimation and the disambiguation between multiple hypotesis.

[4] proposed an approach based on multiple hypothesis Kalman filter based pose tracking. Using a topological map, the decision on the robot move is determined by the max-

978-1-4244-5222-4/10/$26.00 ©2010 CROWN

imization of the expected number of new features observed in the next possible moves. This approach works well when it is possible to identify a clear set of features, whereas we want to address the problem in a general way, regardless on the structure of the environment. The exploration is driven by some heuristics, including the avoidance of visiting the same location twice. We have also made this assumption, as visiting the same location is not giving any new information.

A cornerstone work in the field of mobile robot active localisation is represented by [5] and [6]. In their work they have been selecting actions by maximizing the weighted sum of the expected decrease in uncertainty (entropy) and the costs of moving to the target point. Target points are specified relative to the current robot position and can represent an arbitrary point in the space. Path planning is not involved in the active localisation module. The output is only a single point to be reached by the robot. Position probability grids are used to estimate the vehicle position.

[7] do active sensing, clustering the particles into groups and calculating the total expected entropy for the particle filter by a weighted average of the expected entropy for each group.

With respect to the underwater world, there is not much work in the field of active localisation. The approach of [8] uses active beacons deployed in the environment, in order to help the localisation process. The active strategy is however pretty limited, related only to the disambiguation between the two standard solutions in the beacon localisation problem.

The work carried on by [9] represents an important contribution. It uses active localisation on top of the map previously constructed by a SLAM approach. The set of possible actions are represented by the heading of the vehicle for the following 30 m. The action is selected in order to choose the most discriminative one. The vehicle state is represented with a particle filter, and only a subset of particles are used to evaluate the best action.

[10] propose an approach based on the active movement of an electric field emitter. Apart from the biology-inspired idea of using electric fields, which is out of our scope, the vehicle state is estimate with a particle filter and the control option chosen minimizes the expected variance of the particles at the next step.

Contribution

This paper aims to present a novel approach to the active localisation problem. With respect to passive approaches, the proposed one has the decision on where to move in the loop, allowing the vehicle to decide to follow a specific path in order to determine uniquely its position in the state space. This is very important as a random path often is not able to determine the vehicle position, if unique distinctive features are sparsely present in the environment. In this respect to other active localisation approaches, the proposed method is considering a full trajectory, composed by a set of basic moves. Considering only one basic move, like many approaches described in the previous section, (e.g. [10]) is often not enough. For example, if disambiguating the robots position requires the robot to move to a remote location, greedy single-step algorithms can fail to make the robot move there. With this respect, we fully agree with the work of [6]. However, our approach overcomes some of its limitations. The type of selectable actions are just random targets, with no trajectory planned. We incorporate the building of a trajectory in our approach, as the *pdf* can change significantly according to the single moves the vehicle chooses in order to reach a predefined location. The reward of an action is expressed by the entropy expectation and this calculation is time consuming. In this respect, we agree with [9] about the computational complexity of the entropy expectation calculation. We propose therefore a very computational efficient solution, based on diversity of measurements.

II. AUV Localisation Module

The proposed approach consists in a two-module localisation architecture: the first one is passive, while the second one is active, i.e. with the control of the vehicle's motion in the loop. The first module is the one running by default: it receives all the information from the vehicle's sensors (doppler velocity log, compass, depth estimation, sonar measures) and it tries to combine them in order to estimate correctly the vehicle state. When there is the need to actively localise, the second module starts and produces in output the best path to be followed, in order to drop the uncertainty. The decision to switch from passive localisation to active localisation is taken by the mission planner, based on the current goals and vehicle state estimation. The navigation architecture is highlighted in Fig. 5.

A. Passive Localisation

This module is based on an improved Particle Filter algorithm. It is assumed that the vehicle has a general (although not perfect) knowledge of the environment. Any particle filter rely on a Monte Carlo approximation of the conditional density $p(x_t|z_{0:t})$ using a finite set of points ξ_t^i in the state space called *particles*. The approximation is of the form

$$p(x_t|z_{0:t}) \simeq \sum_{i=1}^{N} w_t^i \delta_{\xi_t^i}(x_t) \quad \text{where} \quad \sum_{i=1}^{N} w_t^i = 1 \qquad (1)$$

where N represents the number of particles and w_t^i represents the weight of the particle i at time t. It can be interpreted in the following way: the denser the particles in a region of the state space and the higher their weights, the higher the probability that the state lies in this region. The most commonly used particle filter is the *sampling with importance resampling* (SIR) algorithm, whose performances are better ([11]) than the first *bootstrap filter*, introduced by [12]. The three steps of iteration $t \geq 1$ of the SIR algorithm are as follows.

1) **Selection:** Generate $\tau_t^i \sim (w_{t-1}^1, \ldots, w_{t-1}^N)$
2) **Propagation:** Generate $\xi_t^i \sim p(x_t|\xi_{t-1}^{\tau_t^i})$
3) **Correction:** Set $w_t^i \propto p(z_t|\xi_t^i)$

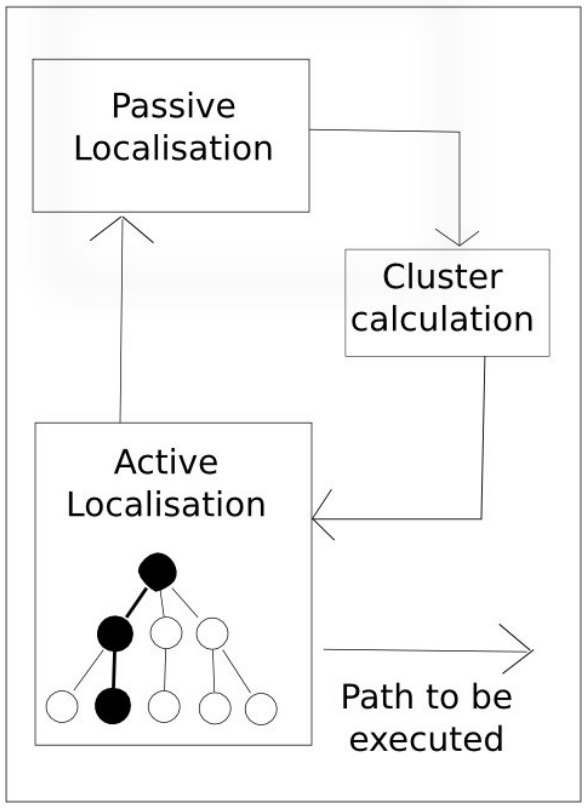


Fig. 1. The general architecture of the navigation system: the passive localisation module is running. According to the probability distribution of the state, the particles are clustered and the centroids are in input to the active localisation module. Through an exploration tree structure, it outputs the path to be executed by the vehile and gives the control back to the passive localisation.

More details about the authors previous work on passive localisation using particle filter can be found in [13] and [14].

With the filtering processing, the particles converge to the most likely positions in the map. According to the map and the motion error, it is possible that more than one position is very likely according to the observations and the motion estimation. This is the case, for example, when similar features are present in different parts of the map. In industrial off-shore oil applications, the map is often composed by a sequence of same elements (pipes, rises, underwater structures), while the navigation error from one site to another one is usually bigger than the distance between two same (or similar) elements, thus preventing a straightforward localisation. In this case, the mission planner triggers the second module in order to decide where to go for uncertainty minimization in the vehicle's state estimation. It is important to underline that the number of particle clusters is not predefined, but determined dynamically.

B. Active Localisation

A prerequisite to start the active localisation module is to have a finite number of possible current states. It provides in output a specific path to follow in order to discriminate between them. The input parameters of the this module are given by a clusterization of the particles. The active localisation module is triggered when there is a clear grouping in the particles and when the vehicle needs to find out precisely its state to carry on its mission. The general principle of the module is to find a set of actions that minimizes the entropy expectation in the particle distribution. Basic actions a_i that the vehicle can perform are identified:

$$\mathbb{A} = \{a^1; a^2; ...; a^n\} \tag{2}$$

The actions a^i are on the format: "*go forward for x meters*", "*go backwards for x meters*", "*go left for x meters*", "*go right for x meters*", "*go up for x meters*", "*go down for x meters*", "*turn x deg clockwise*", "*turn x deg anticlockwise*".

The module produces in output a list $a_{t_0}, ..., a_{t_s}$ which represents the s actions selected to be executed at times $t_0, ..., t_s$. It is important to stress that choosing only the best single move is not enough. It is often impossible to discriminate between possible positions, with only one basic step, while it is more often true that only a more complex trajectory can do that, in order to break the similarity of the environment. A simple greedy approach of building up a new move on top of the previous best move is again not suitable, due to the possibility of local minima and local maxima. The proposed approach is thus exploring a tree of basic actions, expanding each node and calculating the reward for each action. For each cluster, a tree is built having the root initialized to the centroid position and orientation. The complexity of the tree exploration is polynomial on the number n of actions and exponential on the depth d of the tree ($O(n^d)$). However, it is possible to reduce this complexity, considering that for every basic action a^i there is another basic action a^j which produces the opposite effect. As visiting a location already visited is not providing any new information, each node will not expand the action which balances the previous one. Following the same principle, loops on the same root-to-node path are not allowed, thus reducing the final complexity. It is possible to set other constraints, in order to cut the tree. They can be related to the specific vehicle used, and to some manouvres which should be avoided. For simplicity, we have decided not to add any other constraints, but they are easily pluggable in the module. Modeling the robot behavior as a set of basic actions is important in order to consider that different paths to the same location can produce a different probability density function of the vehicle state. The reward function is thus a critical one, as it needs to be evaluated for each node of the tree. Minimize the expected entropy means minimize the function:

$$\begin{aligned} E_{a_i}[H] = -\int\int p(z/x)Bel_{a_i}(x) * \\ log[p(z/x)Bel_{a_i}(x)p(z)^{-1}]dxdz \end{aligned} \tag{3}$$

for every action a^i, as explained in [6]. This calculation is however time consuming ([9]). Our approach deals with the information gain acquired after executing the actions $a_{t_0}, ..., a_{t_s}$. It is represented by the diversity in the measurements z_t^k from each cluster position k, at time t, after executing

all the previous actions linking the root to the node. The sensor measures are simulated from each cluster and from each cluster a *path tree* is generated in order to simulate the sensors over all the possible list of basic actions a^i. It is important to notice that this has the effect of minimizing the expected entropy, without the need of calculating it explicitely. Considering that the measurements z^k are represented by an array of distances (after processing the sonar data), the reward function for a single node n executing the action a^i becomes:

$$r_{a^i}(n) = \frac{\sum_{j=1}^{m} \sigma^2_{z^{j,k}_{a^i}(n)}}{m} \tag{4}$$

where m represents the length of the array and $z^{j,k}_{a^i}(n)$ represents the measures taken from the position of the node n, given by the centroid of the cluster k rototranslated according to the actions $a_0, ..., a_{t-1}$ (path root-node). It represents the average of the variance of the measurements z from the different clusters k for each index j of the measures. Each action has a cost associated, which may vary according to the specific constraints. For example, it is reasonable to assume that if the action a_t is the same than the action a_{t+1}, the cost for the vehicle is less than a completely new action. This is because there is no need to radically change the thruster behaviour. The total value assigned to a single node is given by the difference between the reward r and the cost c. The output of the module is therefore:

$$p^* = \{a^*_{t_0}, ..., a^*_{t_s}\} = argmax_{p_i}(r_{p_i} - c_{p_i}) \tag{5}$$

where p_i represent the path i^{th} and p^* represents the best path. The maximum depth of the tree is fixed *a-priori*. The module can stop either when the maximum depth of the tree is reached, returning the best node (and consequently the path root-node, to be executed by the vehicle) or if a node value (given by the diversity of the measures from the different clusters minus the cost of arriving to that node) is higher than a predefined threshold. This is because the need of a full exploration of the tree might not arise, if enough distinctive features have been recognized at a certain point.

Summarizing, the general principle of the module is to find a path that maximizes the diversity in the observations from the different initial possible positions. From the centre of each cluster, a trajectory tree is built. Each node represents a possible basic move. The output of the module is a path root-leaf (i.e. a sequence of basic moves) which maximizes the diversity in the observations and thus minimizing the expected entropy.

III. Experimental Results

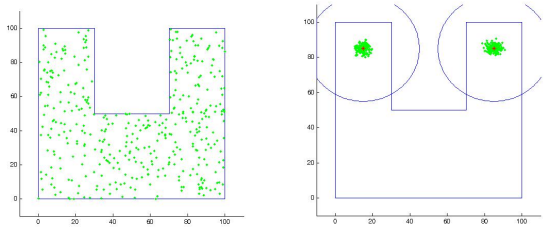
A. Simulated setup

The algorithm has been tested in a simulated setup. For simplicity, only 2D environments have been tested, in order to reduce the set of basic moves. The 3D extension is very straightforward, as the only change is the number of elements in the set of possible actions. Conceptually and practically, there is no difference, apart from the computational time, which is however relatively low. The sensor modeled is a Tritech Micron, currently mounted on our vehicle *Nessie IV*. It is a mechanically scanning imaging sonar (MSIS), with $360\,\mathrm{deg}$ field of view. For this reason, we have not set any rotation in our set of basic moves, as it does not provide any more information about the environment. In the case in which the field of view is limited, then the rotation basic actions are necessary. The set of basic actions is thus represented by:

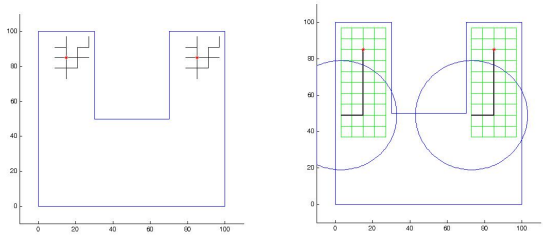
$$\begin{aligned}\mathbb{A} = \text{move}\{&\text{forward, backwards, left, right}\}\\ &\text{for 6 meters}\end{aligned} \tag{6}$$

The first environment has a *U configuration*. The vehicle is either at the end of one of the two legs of the *U*. With the vehicle positioned on the left leg, the particles, initially spread all over the environment, quickly converge in the two locations. Of course, the same result appears with the vehicle positioned on the right leg. As the observation model from the two points is completely the same, it is not possible to distinguish between the two hypotesis with classical passive techniques. The control is then given to the active localisation module which takes the two locations of the cluster centroids in input. The dimensions of the environment are 100x100 meters, with each leg long 50 meters and 30 meters wide. The two cluster centroids are located at $(15; 85)$ and at $(85; 85)$ With a sonar range of 40 meters, the output of the module is a path composed by six basic moves, all going backwards. This is actually the best path in order to discriminate between the two solutions, as the diversity in the environment can be sensed on the bottom of the *U*. Reducing the range to 30 meters, the output is composed by eight basic moves: six backwards and then two on the left. This is consistent with the expectations, as the generated path arrived to a point very close to the borders of the environemnt (presence of obstacles) for one cluster, while for the other cluster, the final point was far from any obstacle. This simulated environment shows the possibility to apply active localisation in closed environment, like it is often the case for man-made environments, like marinas. All the steps are highlighted in Fig. 2.

The second environment is more similar to open sea conditions and leading to off-shore applications. There are no boundaries, just three objects, which can represent an underwater site. Assuming that the vehicle is traveling from one site to another one, it is very likely that the navigation error is bigger than the distance between two objects and thus the vehicle needs to find a way to discriminate between the initial hypotesis (in this case, three). This case is also interesting as it shows how a small change in the parameters of the sonar can change significantly the results. Due to the location of the underwater objects, a range of the sonar over 27 meters can discriminate between the positions without need of any active localisation. Reducing the range gradually, the generated path change significantly. Between 26 and 27 meters, one move is enough to distinguish between the three hypotesis and the selected move is to go backwards. For the top right centroid, this has the effect to go nearer the two middle objects. When the range drop up to 23 meters, the selected move is to go right.



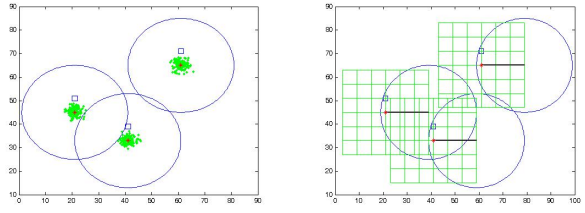
(a) Initial state: uniform distribution of the vehicle state (b) Clusterization of the particle



(c) Tree built from each centroid (d) Path chosen to maximize the information gain

Fig. 2. All the steps of the active localisation process: clusterization, tree construction and path building. First scenario: U-like closed environment

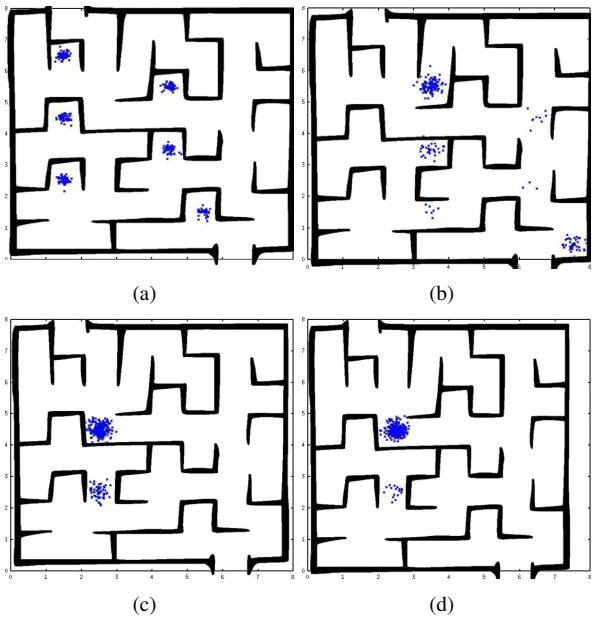
For the left centroid, this has the effect to go nearer the central object. Reducing again the range, the required trajectory is composed by two steps and again the first choice (up to 21 meters) is to go backwards, creating a measure discrepancy between the top centroid and the other two. At 20 meteres range, the two steps are on the right. Up to 17 meters range, the planned path is to go on the right for three steps: this helps to discriminate the left centroid (very near to the central object) with the respect to the other two objects. It is now interesting to see what happens for sonar range below 17 meters. There is no straight exit from the Active Localisation module, so the tree is fully explored until the maximum depth (fixed at nine). However, the best discriminant path that the algorithm can found is not a path of length 9, but it is the last path generated for a sonar range of 17-20 meters, of length 3. A representation of the environment, with particle clustering and chosen path is highlighted in Fig. 3.



(a) Particle clustered in front of the three objects (b) Path chosen to maximize the information gain

Fig. 3. Second scenario: three objects in an open environment.

A complex simulated test - a labyrinth-style environment, Fig. 4 - represents our third simulated environment. From one side, it proves the validity of the algorithm, while testing a possible real scenario, like underwater cave inspection. For such complex scenario with so many constraints, given by the walls, the algorithm needs to be applied iteratively, in order to arrive to a final unique determination of the robot pose. The first path generated by the algorithm drops the number of clusters from the inital six to three. The second path drops it from three to two, while the third path provides a unique solution. It is to be noted that the last path is actually a degenerated path, with the robot deciding not to move. This decision is very rare, if not impossible, at the very beginning of the active localisation module, otherwise it would mean that two (or more) different positions sensing substantial different measures would have all a certain likelihood to represent the robot pose. It is however possible when there is an iteration of the algorithm, like in this case. The second path has dropped one cluster (the fourth), while keeping the position is recognized to be the best action to discrminate between the last two clusters.



(a) (b) (c) (d)

Fig. 4. (a) inital distribution (b) pose estimation after first path (down-right-right) is executed. Three clusters are dropped, (c) pose estimation after second path is executed (left-down). An additional cluster is dropped. (d) execution of the third path (stay still): particle convergence.

B. Tank Trials

The algorithm has been succesfully tested in several tank trials, using the facilities at Heriot-Watt University. The test platform was the Autonomous Underwater Vehicle *Nessie IV* [15], equipped with a Tritech Micron sonar to sense the environment, and Doppler Velocity Log (DVL) for motion estimation. A first set of tests has taken place in a 3x4 m tank, as in Fig. **??**.

A panel has been put into the tank in order to create two identical parts in the environment, similar to the *U-scenario*

Fig. 5. The autonomous underwater vehicle Nessie IV in the OSL tank. It is to be noted the division panel, in order to create two identical environment sections.

described in the simulated setup. The results are highlighted in Fig. 6.

The second set of tests has taken place in a 10x12 m tank. Several environments have been tested, although we only present one set-up in this paper. The environment is composed by the tank walls plus four panels making two identical sections. The robot starts with no initial knowledge of its position, so particles are spread over all the environment. After a clear clustering of the particles is reached and stable over time, the active localisation module is triggered and the vehicle computes the path to be executed. After executing the path, the vehicle's position is determined without doubts. All the steps are highlighted in Fig. **??**.

IV. Conclusion

This paper has presented a novel approach to the active localisation problem. Driven by the maximization of the information gain, the algorithm has proven to be reliable and efficient. Its key point is the ability to efficiently build a path to be executed, in order to discriminate between possible poses. The experimental results have showed its capabilities in different scenarios, always providing excellent results. The next step will be to test the algorithm in a realistic off-shore scenario, using the *ARF* simulator ([16]). Real tests will then be performed during a long mission in a Scottish harbour and lake.

Acknowledgment

This research was sponsored by the Marie Curie Research Training Network FREEsubNET (MRTN−CT−2006−036186) and by the Scottish Funding Council for the Joint Research Institute in Signal and Image Processing between the University of Edinburgh and Heriot-Watt University (part of the Edinburgh Research Partnership in Engineering and Mathematics - ERPem). The authors would also like to thank all the people at the Ocean Systems Lab, for their invaluable support and advice.

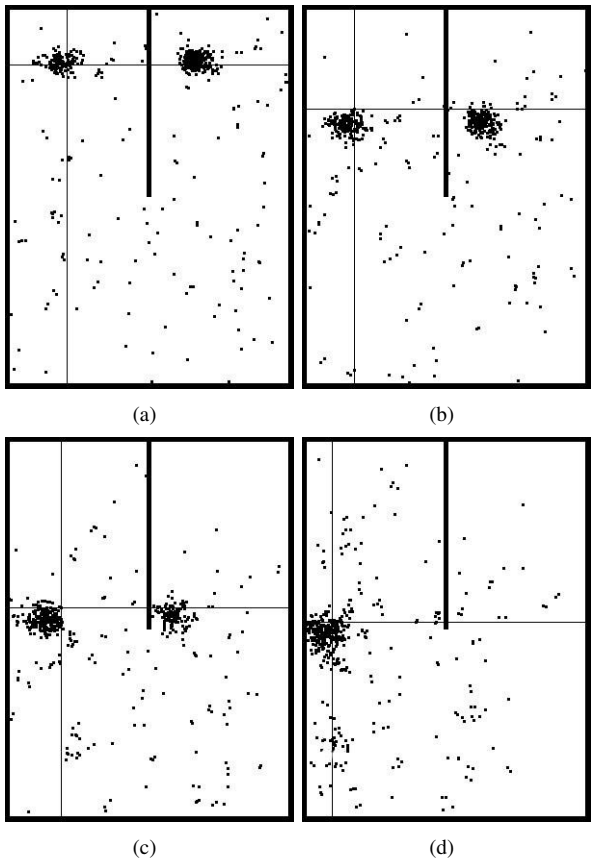


Fig. 6. (a) Initial distribution (b-c) executing the path (down-down-down-down-left-left) (d) Convergence after execution of the path generated by the Actve Localisation module.

References

[1] F. Giuffrida, P. Morasso, G. Vercelli, and R. Zaccaria. Integration of active localization systems in vehicles control for real-time trajectory tracking. pages 549–556, Dec 1996.

[2] A. Arsenio and M.I. Ribeiro. Active range sensing for mobile robot localization. 2:1066–1071 vol.2, Oct 1998.

[3] C. Tessier, C. Debain, R. Chapuis, and F. Chausse. Active perception strategy for vehicle localisation and guidance. pages 1–6, June 2006.

[4] P. Jensfelt and S. Kristensen. Active global localization for a mobile robot using multiple hypothesis tracking. *Robotics and Automation, IEEE Transactions on*, 17(5):748–760, Oct 2001.

[5] Wolfram Burgard, Dieter Fox, and Sebastian Thrun. Active mobile robot localization. In *In Proceedings of IJCAI-97. IJCAI, Inc*. Morgan Kaufmann, 1997.

[6] D. Fox, W. Burgard, and S. Thrun. Active markov localization for mobile robots. volume 25, pages 195–207, 1998.

[7] Rainer Kümmerle, Patrick Pfaff, Rudolph Triebel, and Wolfram Burgard. Active monte carlo localization in outdoor terrains using multi-level surface maps. In *AMS*, pages 29–35, 2007.

[8] E. Olson, J. Leonard, and S. Teller. Robust range-only beacon localization. In *Autonomous Underwater Vehicles, 2004 IEEE/OES*, pages 66–75, June 2004.

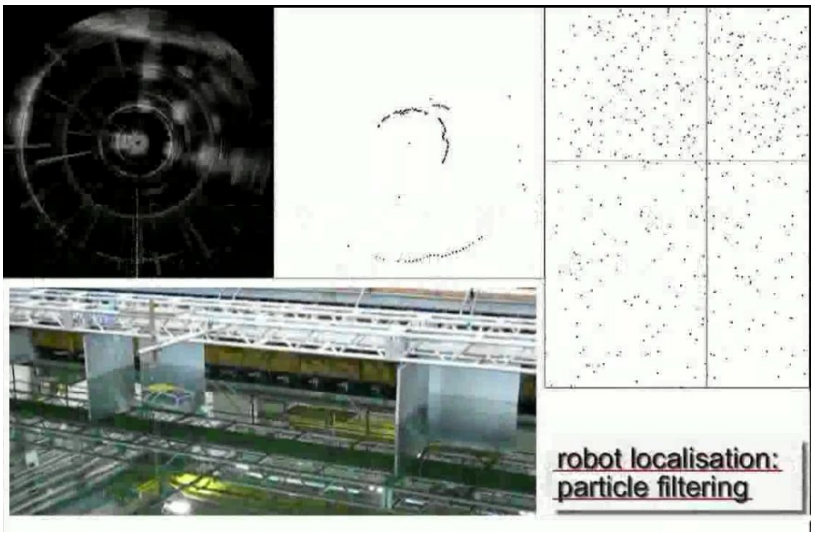
[9] Nathaniel Fairfield and David Wettergreen. Active localization on the ocean floor with multibeam sonar. In *Proceedings of MTS/IEEE OCEANS*, 2008.

[10] James R. Solberg, Kevin M. Lynch, and Malcolm A. MacIver. Robotic electrolocation: Active underwater target localization with electric fields. In *ICRA*, pages 4879–4886, 2007.

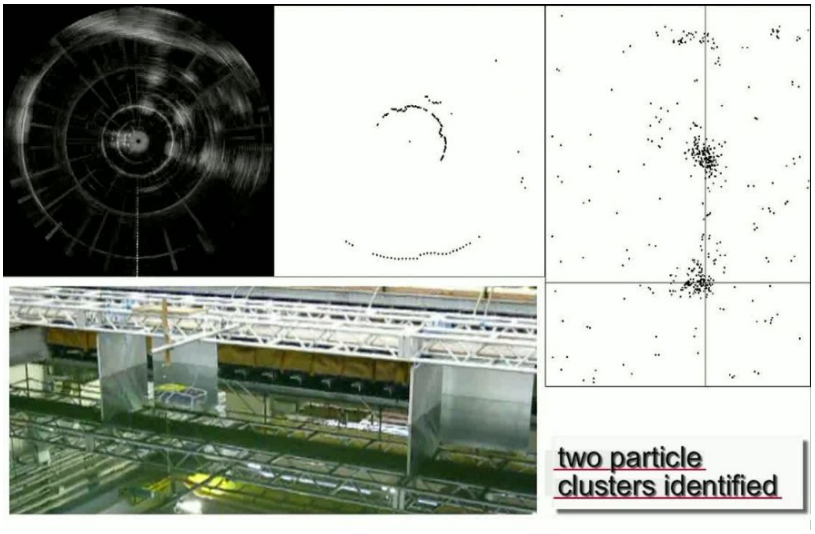
[11] W. Burgard, D. Fox, and S. Thrun. On sequential monte carlo sampling methods for bayesian filtering. *Journal of Artificial Intelligence*, 2001.

[12] N. Gordon, D. Salmond, and C. Ewing. A novell approach to nonlinear nongaussian bayesian estimation. In *Proc. IEEE Radar and Signal Processing*, 1993.

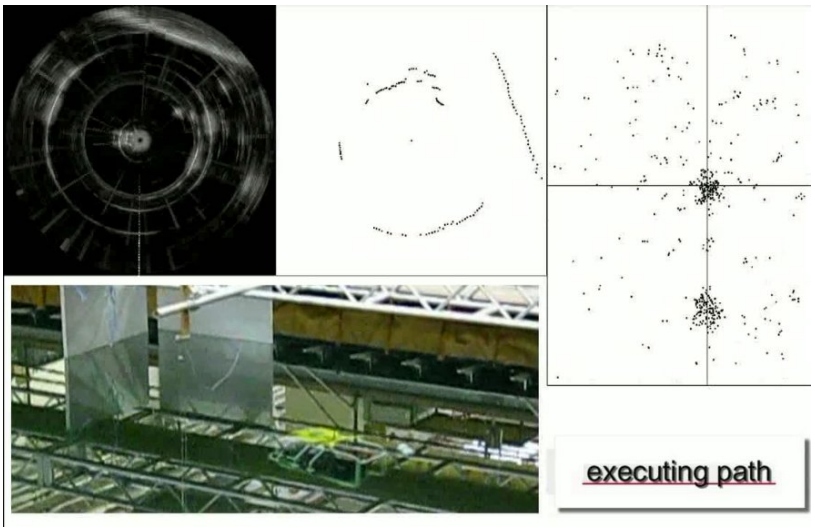
[13] F. Maurelli, S. Krupiński, A. Mallios, Y. Petillot, and P. Ridao. Sonar-



(a) Initial situation, the robot doesn't know its position

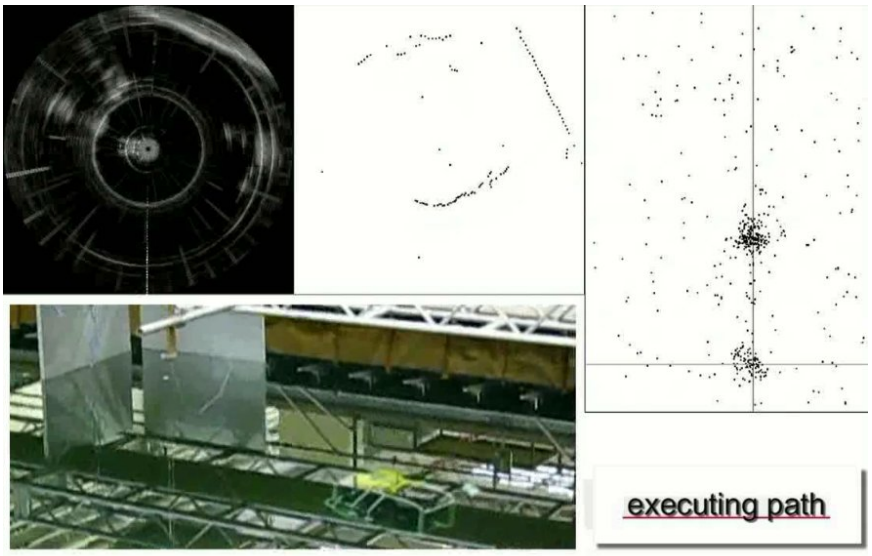


(b) Two clusters are identified. At this moment the Active Localisation module is triggered and a path is planned.

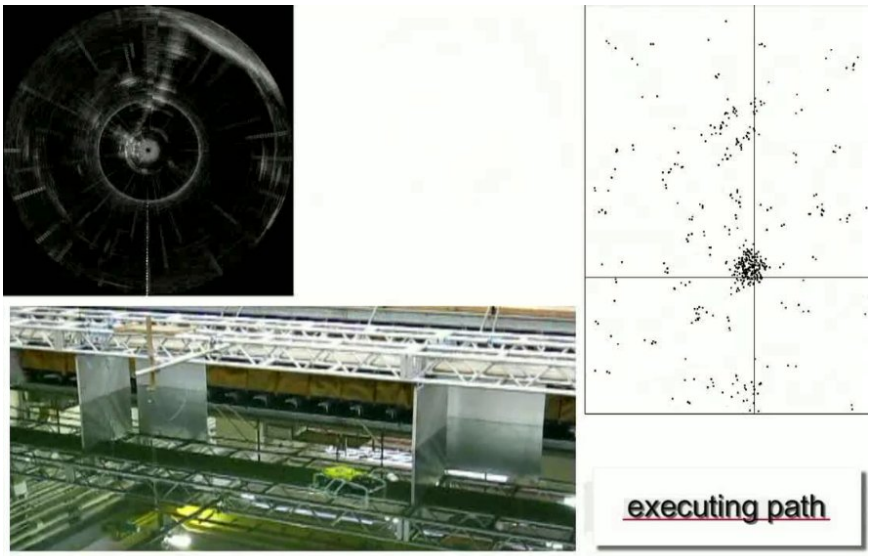


(c) Path execution I

Fig. 7. Second real scenario (I): raw sonar image (top left), processed sonar image (top centre), state estimation (top right), real vehicle position (centre-left).



(a) Path execution II



(b) Path execution III: particle convergence

Fig. 8. Second real scenario (II): raw sonar image (top left), processed sonar image (top centre), state estimation (top right), real vehicle position (centre-left).

based auv localization using an improved particle filter algorithm. In *Proceedings of IEEE Oceans '09, Bremen, Germany*, 2009.

[14] F. Maurelli, A. Mallios, D. Ribas, P. Ridao, and Y. Petillot. Particle filter based auv localization using imaging sonar. In *Proceedings of IFAC MCMC '09, Sao Paulo, Brazil*, 2009.

[15] F. Maurelli, J. Cartwright, N. Johnson, G. Bossant, P.L. Garmier, P. Regis, J. Sawas, and Y. Petillot. Nessie iv autonomous underwater vehicle. In *Proceedings of UUVS 2009, Southampton, UK*, 2009.

[16] Benjamin C. Davis, Pedro Patron, Miguel Arredondo, and David M. Lane. Augmented reality and data fusion techniques for enhanced Situational awareness of the underwater domain. In *OCEANS 2007 - EUROPE, VOLS 1-3*, pages 1090–1095, 2007. Oceans 2007 Europe International Conference, Aberdeen, SCOTLAND, JUN 18-21, 2007.