Adaptive mission planning: the embedded OODA loop

Pedro Patrón, David M. Lane
Ocean Systems Laboratory, Heriot-Watt University
Edinburgh, Scotland, UK, EH14 4AS

Abstract *This paper proposes a novel technique for autonomous mission plan recovery in order to increase operability of unmanned underwater vehicles. It combines the benefits of knowledge-based ontology representation, autonomous partial ordering plan repair and robust mission execution. The approach is based on a combination of unrefinement and refinement stages on the plan-space domain in order to adapt declarative mission plans. It can handle uncertainty and action scheduling in order to maximize mission efficiency and minimise mission failures due to external unexpected factors. Its performance is presented in a set of simulated scenarios for different concepts of operations for the underwater domain. The paper concludes by showing the results of a trial demonstration carried out on a real underwater platform. The results of this paper are readily applicable to land and air robotics.*

Keywords – misssion plan repair, mission adaptation, ontology knowledge representation, autonomous decision making

Introduction

Motivation

Autonomous Underwater Vehicles (AUVs) have become a standard tool for data gathering for military applications. In these environments, mission effectiveness directly depends on vehicle's *operability*. Operability underlies the ultimate vehicle's *availability*. Two main vehicle characteristics can improve the vehicle's operability: *reliability* depends on the internal hardware components of the vehicle, and *survivability*, a concept that is closely related with vehicle failures due to external factors or damages.

In recent years, emphasis for increasing AUV's operability has been focused in increasing AUV's survivability by reducing the *susceptibility* and *vulnerability* of the platform. Recent approaches in rules of collision [3] and wave propagation techniques [18] for obstacle avoidance, collision avoidance and escape scenarios [7] have focused on reducing susceptibility by looking at the adaptation of the vehicle's trajectory plan.

However, when active and passive measures fail to protect the vehicle, or unexpected hardware failures occur, the focus of the mission should shift to 'reconfigure' itself to use alternative combinations of the remaining resources. Scarce bandwidth and tight response constraints in some domains force the operator out of the decision making loop. In such challenging environments, autonomous embedded *recoverability*, is a key capability for vehicle's endurance. This can be achieved via adaptation of the vehicle's mission plan.

Current AUV mission plan solutions are procedural and static [12]. If behaviors are added [15], they are only to cope with possible changes that are known a-priori by the operator. In order to achieve higher levels of autonomy, an evolution in plan adaptability from current waypoint-based approaches to declarative goal-based solutions is required.

Declarative mission planning solutions for unmanned vehicles leverage from the research carried out on autonomous planning generation [11]. However, almost as important as the

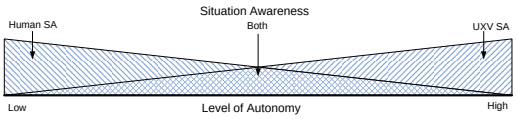

Figure 1: Human and AUV SA across the levels of autonomy

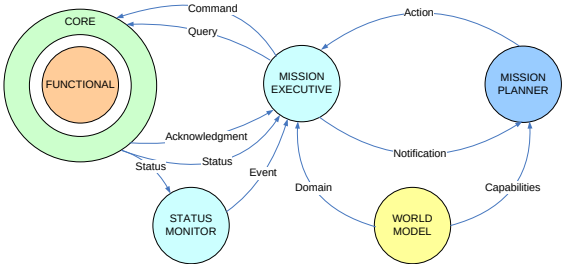

Figure 2: Interaction of the $\mathcal{M}_{ME}$, $\mathcal{M}_{MP}$, $\mathcal{M}_{SM}$ and $\mathcal{M}_{WM}$ architecture modules with a generic platform functional layer.

mission plan generation (if no more), is the capability of the mission to adapt and recover from failures. The aim is to be effective and efficient and a mission plan costs time to prepare. This time has been already invested once (to compute the mission that is now failing), so it might be wise to try to reuse of previous effort by repairing it. Also, commitments might have been made to the current mission plan: trajectory reported to other intelligent agents, assignment of resources, assignment of part of mission plan to executors,... Repairing an existing mission ensures that few commitments as possible are invalidated. Finally, often several planners (usually autonomous and human planners combined) could be performing together to achieve the goals. In such cases, it is more likely that a similar mission plan will be accepted by the operator than one that is potentially completely different.

Related work

Space operations are driving the motivation behind autonomous planning and adaptability for unmanned vehicles. The *Remote Agent Experiment* (RAX) [17], which flew on Deep Space-1, demonstrated the three tier architecture and its ability to autonomously control an active spacecraft. One problem discovered involved using a batch planner on a reactive system. It took hours to replan after updates had invalidated the original plan.

Another system, called $CLEaR$ [8], used the $CASPER$ system [6] in conjunction with a sequencer/controller in order to provide a planning and execution framework for closed loop commanding. The deliberative and reactive methods operated in parallel at run time to determine how to best respond to failures and take advantage of opportunities. The system highlighted the difference between 'local' conflicts - errors that require changes only to the currently executing part of the plan - and 'global' conflicts - errors that occur which require changes to future parts of the plan. Local conflicts were managed by the executive while global ones were passed back to the planner to be fixed. Van der Krogt later formalised this two levels of handling events in what was called *executive repair* and *planning repair* [21]. His approach combined unrefinement and refinement stages in order to provide faster performance than planning from scratch. However, it failed to produce the most optimal plan. This could be considered an issue in domains requiring optimality. This is not generally the case in unmanned vehicle mission plans where optimality can be sacrified for operability.

Recently, Fox proposed an on-board planning assistant to the operator to adjust and repair plans on-board [9]. It can handle idle times due to conservative mission planning and plan failures. The approach relaxes methodological constraints and fills opportunity gaps only in situations where resources would otherwise go unused. A wider discussion on other related research that goes from strong executors and formal planning approaches to strong deliberators can be found in Knight's review of the field [14].

In the underwater domain, several challenges have been identified requiring adaptive plannning solutions of the mission [20] [2]. The potential benefits of the adaptive planning capabilities has been promoted by Rajan [19]. Together with Fox [10], they have started using sensor data to adapt the decision making on the

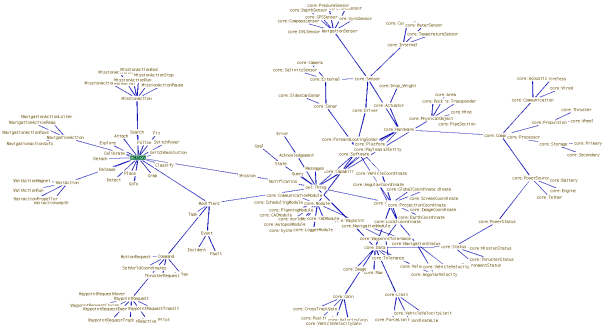

Figure 3: Core domain and planning concepts and relationships represented in the TGVizTab plugin on Protege.

vehicle's mission.

In our work, we present an approach for fault tolerant adaptive mission planning for AUVs. It combines semantic knowledge representation with fault prognosis in order to observe changes in the platform capabilities and orient the decision making process.

Decision making loop for mission adaptation

The human capability of understanding highly dynamic and complex environments is known as *situation awareness* (SA_H). SA_H breaks down into three separate levels: perception of the environment, comprehension of the situation and projection of the future status.

According to Boyd, decision making occurs in a cycle of observe-orient-decide-act ($OODA$) [4]. The *Observation* ($\mathcal{OB}_{ooda}$) component corresponds to the Perception level of SA_H. The *Orientation* ($\mathcal{OR}_{ooda}$) component contains the previously adquired knowledge and understanding of the situation. The *Decision* ($\mathcal{D}_{ooda}$) component represents the SA levels of comprehension and projection. This decision stage is the central mechanism enabling adaptation before closing the loop with the final *Action* ($\mathcal{A}_{ooda}$) stage. Note that it is possible to take decisions by looking only at orientation inputs without making any use of observations.

Based on the autonomy levels and environmental characteristics, situation awareness definitions can be directly applied to the notion of *unmanned vehicle situation awareness* (SA_V) [1]. The levels of situation awareness for individual unmanned vehicle systems (SA_S) spans from full human control to fully autonomous unmanned capabilities (see Fig. 1).

In current implementations, the human operator constitutes the $\mathcal{D}_{ooda}$ phase. The embedded architecture is formed by the *world model* module ($\mathcal{M}_{WM}$) and the *mission plan executive* module ($\mathcal{M}_{ME}$). $\mathcal{M}_{WM}$ stores $\mathcal{OB}_{ooda}$ and $\mathcal{OR}_{ooda}$ and $\mathcal{M}_{ME}$ performs $\mathcal{A}_{ooda}$. When high-bandwidth communication links exist, the operator remains in the $OODA$ loop during the mission execution. However, when communication link is poor, unreliable or not allowed, the operator tries, based only on $\mathcal{OR}_{ooda}$, to include all possible behaviours to cope with execution alternatives. This has unpredictable consequences, in which unexpected situations can cause the mission to fail or to abort and might even cause the loss of the vehicle.

In order to achieve autonomous adaptive mission planning, two additional components are required: a *status monitor* ($\mathcal{M}_{SM}$) and a *mission plan adapter* ($\mathcal{M}_{MP}$). The status monitor reports any changes detected during the execution of a plan. These modifications might change the SA_V *Perception*. When the $\mathcal{M}_{ME}$ is unable to handle the changes coming from the $\mathcal{M}_{SM}$, the $\mathcal{M}_{MP}$ is called to generate a new modified mission plan that agrees with the updated constraints (see Fig. 2). The RAX project was the first attempt of implementing this type of architecture on a real environment. However, tight time deadlines, more restricted communications and environment constraints existing in general AUVs applications have led to the research of new approaches.

Observation and orientation

SA_V consists in making the vehicle to autonomously understand the '*big picture*'. This picture is composed by the experience achieved from previous missions and the information obtained from the sensors while on mission. It is composed by the world model (long term memory) and the communication messages (short access memory) providing knowledge access to all

Mission domain ontology $Tbox$
• CAD Module - hasRequirement: Sidescan driver - isCapableOf: Mine-like objects detection • CAC Module - hasRequirement: Mine-like objects detection - isCapableOf: Mine-like objects classification • CAD/CAC Payload - hasSoftware: CAD/CAC Module • Platform - hasPayload: Payload • Action - hasPlatform: Platform • Mission - hasPlatform: Platform ...

Figure 4: Example of mission plan domain $Tbox$ concepts and relationships.

Mission problem ontology $Abox$
• SBCAC $\in$ CAC • SBCAD $\in$ CAD • SB337 $\in$ CAD/CAC Payload • Remus337 $\in$ Platform • Explore $\in$ Action - hasRequirement: Sidescan driver • Detect $\in$ Action - hasRequirement: Mine-like objects detection • Detect $\in$ Action - hasRequirement: Mine-like objects classification • Clearance $\in$ Mission - hasRequirement: Mine-like object classification ...

Figure 5: Example of mission plan problem $Abox$ instances.

the phases of the autonomous decision making process.

$TBox$ and $ABox$ statements make up an ontology, or knowledge base. These two components contain respectively the terminological and assertion concepts and relationships for each particular context. A set of ontologies has been developed as knowledge repositories for the different components in the architecture (see Fig. 3). They allow the mission adapter to be independent from the domain and platform where the mission is being executed. This section focuses on the domains concepts required at the $\mathcal{M}_{ME}$, the planning concepts handled at the $\mathcal{M}_{MP}$ and the $\mathcal{M}_{SM}$ that modifies them.

Domain ontology

Domain concepts are handled by $\mathcal{M}_{ME}$. They are used to ground the abstract planning concepts managed by $\mathcal{M}_{MP}$ to a particular domain. This allows transition from the $\mathcal{D}_{ooda}$ phase to the $\mathcal{A}_{ooda}$ phase of the $OODA$ loop. Our domain ontology heritates from other well stablished generic ontologies such as the Suggested Upper Merged Ontology (SUMO) and the Ontology for Geography Markup Language of Open GIS Consortium (GML-OGC). Some of the knowledge concepts identified in the unmanned vehicle domain are: *Platfom* (static or mobile - ground, air, underwater vehicles -), *payload* (hardware with particular properties, sensors or modules), *module* (software with specific capabilities), *sensor* (a device that receives and responds to a signal or stimulus), *waypoint* (position in space with coordinate and tolerance), *coordinate* (local frame, global frame, angular), *velocity* (linear, angular),...

Planning ontology

Planning concepts are handled by $\mathcal{M}_{MP}$. In order to provide a solution to a mission failure, the decision making requires, not only concepts to generate a mission, but also concepts capable of representing incidents or problems ocurring during the mission. Some of the most important concepts identified for mission plan adaptability are: *Resource* (state of an object in the environment - physical or abstract), *action* (resource modifier), *catalyst resource* (resources that are not consumed for an action but needed for the proper execution of the action), *plan gap* (actions that may no longer be applicable), *execution* (when an action is executed successfully), *failure* (an unsuccessful execution), *incident* (a combination of failures),...

Diagnosis system

In order to identify any problems ocurring during the mission, a status monitor system is required capable of transforming domain dependant information into planning knowledge. For demonstration purposes, we concentrate in changes ocurring to vehicle's health via use of

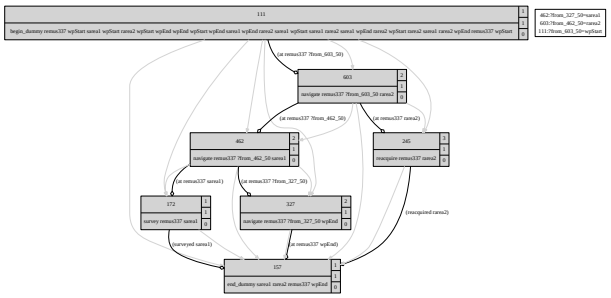

Figure 6: Ordering constraints (grey-arrows) and interval preservation contraints (black arrows) between actions in a partial ordered plan representation of an AUV mission.

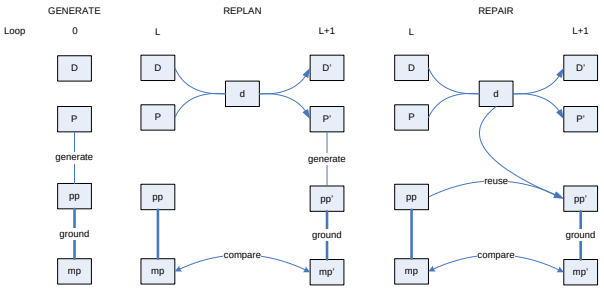

Figure 7: Replan and repair processes for mission plan adaptation.

the outputs of the $RECOVERY$ intelligent fault detection and diagnosis system ($\mathcal{M}_{SM} = \mathcal{M}_{Recovery}$) [13].

Decision

Knowledge consistency and inference

A set of pairs of $TBox$-$ABox$ at the $\mathcal{OB}_{ooda}$ and $\mathcal{OR}_{ooda}$ phases, not only describes the relationships between concepts but also facilitates the decision making process. An ontology facilitates reasoning over concepts. Concepts consistency can be checked to reassure that SA_V remains stable through the development of the mission. Also, inference of concepts and relationships allows new knowledge to be extracted or inferred from the observed data.

Fig. 4 and Fig. 5 show the main elements of an ontology representing the underwater mine countermeasure scenario. Reasoning over this knowledge base can autonomously provide answers to questions such as:

- Has the 'Remus337' Platform 'Detection' capability ?
- Can the mission 'Clearance' be performed successfully with 'Remus337' ?
- Could actions 'Explore', 'Detect', 'Classify' be performed by 'Remus337' ?

This kind of initial reasoning at the $\mathcal{D}_{ooda}$ level prepares the information required to correctly adapt the mission plan in $\mathcal{M}_{MP}$ accordingly to the platform status updated by $\mathcal{M}_{SM}$.

Mission plan adaptation

Following classical planning problem representation, an instance of a vehicle mission problem can be simply defined as $\Pi = \{\mathcal{P}, \mathcal{A}, \mathcal{I}, \mathcal{G}\}$, where $\mathcal{P}$ is the set propositions defining the available resources in the vehicle, $\mathcal{A}$ is the set of actions, $\mathcal{I}$ is the initial platform state and $\mathcal{G}$ is the set of possible mission accomplished states. $D = \{\mathcal{P} \cup \mathcal{A}\}$ defines the mission domain and $P = \{\mathcal{I} \cup \mathcal{G}\}$ the mission problem. Given an instance Π, the *mission generation problem* consists in finding if there exists a mission plan (mp), using $a \in \mathcal{A}$, such that satisfies any $g \in \mathcal{G}$.

Several approaches exist in the AI literature capable of solving this problem. In a real environment where optimality can be sacrified by operability, partial ordered planning (pp) is seen as a suitable approach because it produces a flexible structure capable of being adapted (see Fig. 6). The implemented approach can deal with extensions from the classical representation. It can handle durative actions, fluents, functions and different search metrics in order to minimise resource consumption, such as remaining battery, or total distance travelled.

Fig. 7 shows the difference between replanning and repairing a mission plan. At the initial loop, a partial ordered plan pp_0 is generated satisfying the given mission domain D_0 and problem P_0. The pp_0 is then grounded into the minimal mission plan mp_0 including all contraints in pp_0. At an iteration L, the knowledge base is updated by the diagnosis information d_L providing a modified mission domain D'_{L+1} and problem

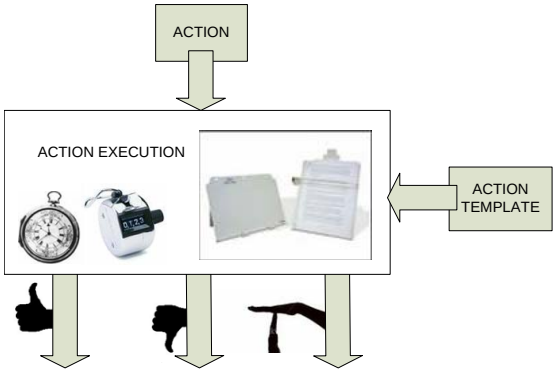

Figure 8: Action execution grounded from the mission plan action and the action concept on the domain $TBox$.

P'_{L+1}. From here, two mission recovery options are possible: The *mission replan* process generates a new partial plan pp'_{L+1}, as done at the first stage, based only in the D'_{L+1} and P'_{L+1} information. On the other hand, the *mission plan repair* process re-validates the original plan by ensuring minimal perturbation of it. Given a mp to D and P in L, the mission repair problem produces a solution mission plan mp' that solves the updated mission problem D' and P', by minimally modifying mp.

When a mission failure occurs during the execution, two possible repair levels can be identified: *mission execution repair* and *mission plan repair*. Execution repair changes the instantiation mp such that either: an action a_i that was previously instantiated by some execution α is no longer instantiated, or an action a_i that was previously instantiated is newly bound by an execution α already part of the mp. Plan repair modifies the partial plan pp itself, so that it uses a different composition, though it still uses some of the same constraints between actions. It might also entail the elimination of actions which have already been instantiated.

Executive repair will be less expensive and it is expected to be executed by $\mathcal{M}_{ME}$. Plan repair, however, will be computationally more expensive and requires action of $\mathcal{M}_{MP}$. The objective of the approach is to maximise the number execution repairs over plan repairs and, at the plan repair level, maximise the number of decisions reused from the old mission plan.

Mission plan refinement

A practical approach following the previously described concepts has raised interest recently in the AI community providing a novel solution for all drawbacks identified during the replanning process. This set of methods is known as plan recovery methods. Plan recovery methods are based on plan-space searches and are able to adapt the existent plan to the new state of the world. They can be divided into two stages:

The first stage, known as *plan diagnosis*, analyzes the effects of the updated platform status on the current mission. According with the new updated constraints received from the $\mathcal{M}_{SM}$, it identifies the failures and gaps existent in the current mission plan. These plan gaps are causing the inconsistency between the existent plan and the current status of the platform. They are, therefore, preventing the correct execution of the mission. The approach developed at this stage is based on unrefinement planning strategies.

The second stage is known as *plan repair*. The strategy during this stage is to repair with new partial plans the gaps or failures identified during the plan diagnosis stage. The plan repair stage is based on refinement planning strategies for plan recovery.

In simple terms, when changes on the world are sensed (d) that affect the consistency of the current mission plan pp_L, the plan diagnosis stage starts an unrefinement process that relaxes the constraints in the mission plan that are causing the mission plan to fail. The remaining temporal mission partial plan pp_t is now relaxed to be able to cope with the new sensed constraints. This will be the simplest stage of recovery necessary to continue with the execution of the plan but it does not guarantee that all the mission goals will be achieved. The plan repair stage then executes a refinement process searching for a new mission plan pp'_{L+1} that is consistent with the new world status D' and P'. By using this technique, it can be seen that the new mission plan mp' is not generated again from D' and P' (re-planned) but recycled from pp_L (repaired).

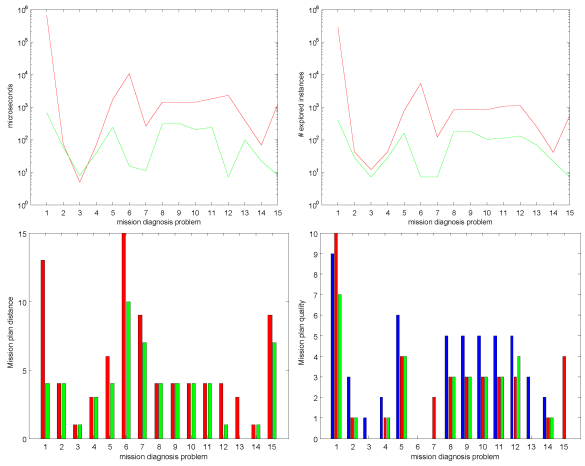

Figure 9: CPU time (on a logaritmic scale), number of explored nodes, $D(\Pi_0)$ and number of instantiated actions for the 15 REMUS mission adaptations.

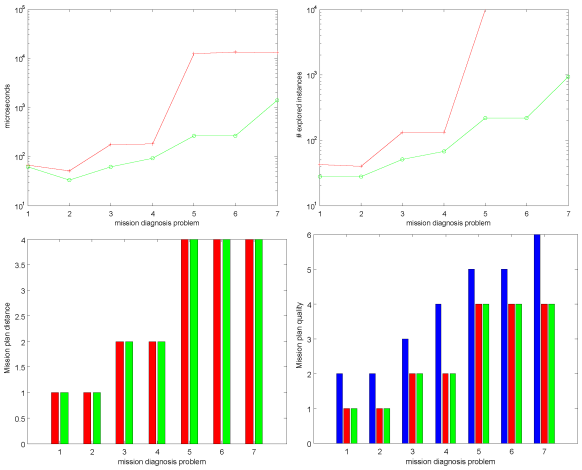

Figure 10: CPU time (on a logaritmic scale), number of explored nodes, $D(\Pi_0)$ and number of instantiated actions for the 7 smart AUV operation adaptations.

This allows re-use of the parts of the plan pp_L that were still consistent with D' and P'. At the $\mathcal{D}_{ooda}$ phase, a partial plan is maintained that gets adapted based on the updates obtained during the $\mathcal{OB}_{ooda}$ phase. The plan adaptability process is initiated when any of the concepts from the domain $ABox$ used inside the mission change.

Action

Action execution

The translation from the $\mathcal{M}_{MP}$ to the $\mathcal{M}_{ME}$ is done via a sequence of instances of action executions. An action execution is defined by the domain ontology $TBox$ in $\mathcal{M}_{WM}$ and gets instantiated by the action grounded on mp. The instance contains the script of commands necessary to perform the action (see Fig. 2). The action execution contains a timer, an execution counter, a timeout register and a register of the maximum number of executions. The success, failure or timeout outputs control the robust execution of the mission and the executive repair process (see Fig. 8).

State machine

Once mp is obtained and the list of grounded action is generated, the mission plan gets transformed into a state machine of action execution instances. A action execution graph gets generated that contains all the possible options for the new plan. This deals with the robustness of the execution and the execution repair process. This minimises the number of calls to the $\mathcal{M}_{MP}$ and therefore the response time for adaptation.

Experimental results

Simulation

A set of synthetic simulated scenarios have been implemented in order to test the performance of $\mathcal{M}_{MP}$. Search time, number of instances created, distance to the original mission plan ($D(\Pi_0)$) and mission plan quality metrics have been analysed. Graphs display metrics for the original, replanned and repaired mission plans in blue, red and green respectively.

• Mine countermeasures scenario (MCM)

MCM is maybe the most problematic mission for AUVs. AUVs support and provide solutions for mine-hunting and neutralisation. The operation involves high levels of uncertainty and risk of damage in the vehicle. Navigating in such a hazard environment is likely to compromise the vulnerability of the platform. If a vehicle is damaged and some of its components failed, mission adaption will be required to cope with the new restricted capabilities.

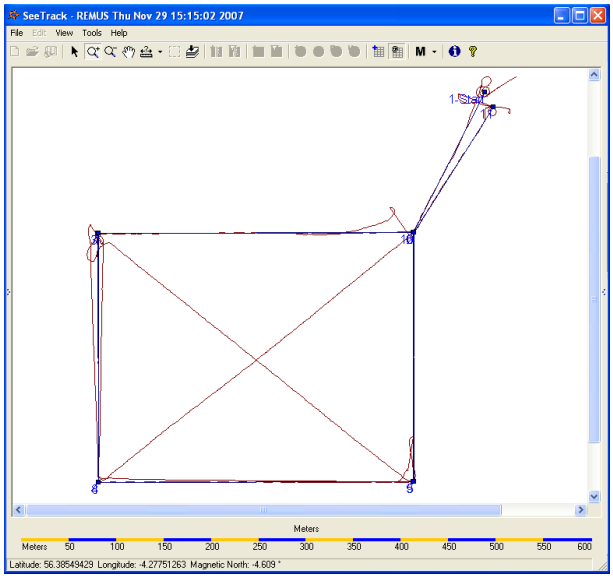

Figure 11: Field trials navigation track diverts from the original square mission baseline adapting to perform two additional diagonal tracks.

Fig. 9 shows metric results of 15 selected MCM scenarios. Each scenario simulates a failure detection d_L. It can be seen that mission repair compared with replanning improves performance and time response while maintaining mission quality and reducing distance to the original mission plan.

• Smart AUV operations

Smart AUVs can detect anomalies and provide reassesment of the status of underwater structures. As described in [16], compared with current state-of-the-art ROV-based methods, fully autonomous inspection methods are able to provide in record times data about the status of their underwater structures. Mission plan adaptation for interchange of objectives between different tasks can maximise efficiency of water time and reduce the overall inspection cost.

Fig. 10 shows metric results of 7 selected Smart AUV scenarios. A performance of the system similar to the MCM scenario can be observed.

Real environment

In order provide architecture independency to the vehicle's functional layer, an Abtract Layer Interface (ALI) has been developed. For demonstration purposes, we have concentrated on the AUV REMUS 100 platform. A PC-104 payload has been installed in the vehicle capable of communicating with its control module via the *REMUS Remote Control* protocol.

A set of trials were performed in Loch Earn (Scotland) in which different component failures have been simulated. Fig. 11 shows an example of one of the original missions programmed by the operator in comparison with the final mission executed by the vehicle. The alternative mission was generated via the mission adaptive planner running on the payload computer when the diagnosis system reported the sensor failure. It can be observed how the mission track was altered in order to cope with the failure and maintain the vehicle operative and maximize the number of goals achieved.

Conclusions and future work

Conclusion

The underwater domain is a challenging environment for maintaining AUV's operability. This can be gained via embedded adaption of the mission plan. We propose a new system capable of repairing mission plans in order to adapt the current mission and maintain vehicle's operability. A combination of ontological representation of knowledge, diagnosis and adaptive mission plan repair techniques is used. The approach combines robust execution and mission plan repair to maximize system performance and response time. The decision making process for mission adaption uses a combination of refinement and unrefinement phases of the constraints of a partial ordered mission plan. The system performance and benefits for two different applications have been demonstrated in simulation. Additionally, the mission adaption capability is shown during an in-water field trial demonstration. The system developed is platform independent and the results of this paper are readily applicable to other platforms, including land and air robotics.

Future Work

The proposed architecture can use other sources of information for the Observation

phase that make the mission adaptation. We are planning to extend the single vehicle mission plan recovery to a mission plan recovery for a group or team of vehicles performing a collaborative mission. We are currently working for an SA for a team of vehicles (SA_T) to which every team member possess the SA_S required for its responsibilities.

Acknowledgements

The work reported in this paper was funded by the Systems Engineering for Autonomous Systems (SEAS) Defence Technology Centre established by the UK Ministry of Defence. Our thanks to all at the Ocean Systems Laboratory and SeeByte Ltd. for their inputs and helpful expertise running the experiments in the real world.

References

[1] Julie A. Adams. Unmanned vehicle situation awareness: A path forward. In *Proceedings of the 2007 Human Systems Integration Symposium*, 2007.

[2] J. Bellingham, B. Kirkwood, and K.Rajan. Tutorial on issues in underwater robotic applications. 2006.

[3] Michael Benjamin, Joseph Curcio, John Leonard, and Paul Newman. Navigation of unmanned marine vehicles in accordance with the rules of the road. *International Conference on Robotics and Automation (ICRA)*, May 2006.

[4] John Boyd. Ooda loop, 1995.

[5] John Bresina, Ari Jnsson, Paul Morris, and Kanna Rajan. Mixed-initiative activity planning for mars rovers. In *International Joint Conference on Artificial Intelligence*, page 1709, 2005.

[6] S. Chien, R. Knight, A. Stechert, R. Sherwood, and G. Rabideau. Integrated planning and execution for autonomous spacecraft. In *Proceedings of the 1999 IEEE Aerospace Conference*, Aspen, CO, march 1999.

[7] J. Evans, P. Patrón, B. Smith, and D.M. Lane. Design and evaluation of a reactive and deliberative collision avoidance and escape architecture for autonomous robots. *Journal of Autonomous Robots*, In Press.

[8] F. Fisher, R. Knight, B. Engelhardt, S. Chien, and N. Alejandre. A planning approach to monitor and control for deep space communications. In *Proceedings of the IEEE Aerospace Conference*, 2000.

[9] M. Fox, D. Long, L. Baldwin, G. Wilson, M. Wood, D. Jameux, and R. Aylett. On-board timeline validation and repair: a feasibility study. In *Proceedings of 5th International Workshop on Planning and Scheduling in Space*, pages 22–25, Baltimore, USA, October 2006.

[10] M. Fox, D. Long, F. Py, K. Rajan, and J. Ryan. In situ analysis for intelligent control. In *Proceedings of IEEE/OES OCEANS Conference, Vancouver*, 2007.

[11] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning*. Morgan Kaufmann, 2004.

[12] P.E. Hagen. Auv/uuv mission planning and real time control with the hugin operator system. In *IEEE OCEANS 2001*, volume 1, pages 468–473, 5-8 Nov 2001.

[13] K. Hamilton. *An Integrated Diagnostic Architecture for Autonomous Robots*. PhD thesis, Heriot-Watt University, 2002.

[14] R. Knight, F. Fisher, T. Estlin, B. Engelhardt, and S. Chien. Balancing deliberation and reaction, planning and execution for space robotic applications. In *International Conference on Intelligent Robots and Systems (IROS 2001)*, Maui, HI, November 2001.

[15] S. Pang, J.A. Farrell, R.M. Arrieta, and W. Li. Auv reactive planning: deepest point. In *IEEE OCEANS 2003*, volume 4, pages 2222–2226, 22-26 Sept 2003.

[16] P. Patrón, J. Evans, J. Brydon, and J. Jamieson. Autotracker: Autonomous pipeline inspection: Sea trials 2005. In *World Maritime Technology Conference - Advances in Technology for Underwater Vehicles*, March 2006.

[17] B. Pell, E. Gat, R. Keesing, N. Muscettola, and B. Smith. Robust periodic planning and execution for autonomous spacecraft. In *Proceedings of the International Conference on Artificial Intelligence*, 1997.

[18] C. Pêtrès, Y. Pailhas, P. Patrón, Y. Petillot, J. Evans, and D. M. Lane. Path planning for autonomous underwater vehicles. *IEEE Transactions on Robotics*, April 2007.

[19] K. Rajan, C. McGann, F. Py, and H. Thomas. Robust mission planning using deliberative autonomy for autonomous underwater vehicles. In *International Conference on Robotics and Automation workshop on Robotics in challenging and hazardous environments*, 2007.

[20] R.M. Turner. Intelligent mission planning and control of autonomous underwater vehicles. In *International Conference of Autonomous Planning and Scheduling, Workshop on Planning under uncertainty for autonmous systems*, 2005.

[21] Roman van der Krogt. *Plan repair in single-agent and multi-agent systems*. PhD thesis, The Netherlands TRAIL Research School, 2005.