3D Adaptive Coverage Planning for Confined Space Inspection Robots

Rowan PIVETTA, Andrew LAMMAS and Karl SAMMUT[1]
Centre for Maritime Engineering, Control & Imaging,
Flinders University, SA, Australia

Abstract. The manual inspection of confined spaces, such as ballast tanks, for corrosion is a hazardous, time consuming, expensive, and subjective process. Robotic inspection techniques are being considered as they promise greater objectivity and comprehensive coverage as well as repeatability, reduced risk, and shorter maintenance downtime. To enable a robot to comprehensively inspect all surfaces within the tank, it needs an inspection plan. Although coverage planning algorithms can be used to automatically generate optimal plans, complex environments and robot kinematics can make the problem challenging. Most existing coverage algorithms can only produce offline plans, however these cannot cope with unforeseen obstacles and would consequently fail when placed in modified environments. Such problems present the need for adaptive coverage-planning algorithms that can replan routes autonomously. To date, the focus of our research has been to evolve an offline *redundant roadmap* based algorithm into an enhanced algorithm capable of adapting its plan online. The online solution utilises LIDAR scan updates to rectify compromised paths through region-based replanning, allowing localised replanning around identified areas of change without recalculating the whole tour from scratch. Preliminary results reveal that the online planner is capable of replanning multiple regions concurrently, faster, and with minimal path degradation compared to its offline counterpart.

Keywords. Coverage planning, adaptive planning, robotics, confined space inspection.

1. Introduction

Performing a manual inspection inside an enclosed, confined space is an inherently hazardous task for humans. Inspectors are required to wear protective equipment and conduct a thorough examination for defects while being subjected to physically and mentally uncomfortable conditions. The objective of the work described in this paper is to replace the human inspector with an autonomous robotic system capable of navigating the confined space to perform a comprehensive inspection and provide a complete suite of observations for later analysis.

The inspiration for this work derives from the Australian Collins Class submarine sustainment program. A submarine contains many tanks with variable dimensions, consisting of multiple T frame stiffeners and internal pipe networks only accessible through small manholes. The challenges in developing a solution for an unsupervised

[1] Assoc. Prof. Karl Sammut, Centre for Maritime Engineering, Control & Imaging, Flinders University, Adelaide, South Australia, Australia; E-mail: karl.sammut@flinders.edu.au

inspection is addressed in [1] which shows that a multiple degree of freedom (DOF) platform that can freely climb around the internal structures using magnetic clamping is best suited to such a role. From our earlier work, a modified Phantom-X concept platform equipped with two cameras mounted on separate limbs and a Velodyne Puck-LITE LIDAR was developed as the concept demonstrator for the project.

High-quality tank inspection by the robot requires the capability to perform live mapping to generate an updated model of the environment. The objective is to generate a set of robot configurations from which a sensor can acquire images of all the surfaces of the tank. Such a comprehensive inspection plan can be generated offline using coverage planning algorithms from pre-existing models. However, when unforeseen objects are detected during execution, possibly due to minor changes not documented during the construction process, they are excluded from the offline plan resulting in an incomplete survey. Therefore, for this application, it is essential that newly detected changes are adequately covered and reflected in the output of the coverage algorithm without the need to recompute the whole inspection plan again.

This paper extends the functionality of a known offline coverage algorithm and utilises its functionality to create an adaptive online solution capable of receiving regular mapping updates, producing coverage for new information and factoring the new viewing positions back into an existing offline inspection plan. This paper is organised as follows; Section 2 briefly discusses coverage planning techniques which enable autonomous robots to conduct high quality inspection plans. Section 3 introduces the offline algorithm and presents the extensions that allow the algorithm to work adaptively online. Section 4 demonstrates the capability of the adaptive planner against a naïve planner in simplified test environments. Finally, Section 5 summarises the conclusions.

2. Coverage Planning

Coverage planning is the process of developing a path for an autonomous robot that enables it to view all points within an area or volume [2]. Surveys [2, 3] demonstrate various offline and online coverage planning techniques in both 2D and 3D environments for applications such as agriculture, manufacturing and inspection. However, many of the algorithms presented, segment the environment to generate plans of continuous sweeping motions. Furthermore, the different tank variations on-board a submarine are not easily parameterised and run the risk of missing critical information if not discretised, segmented or sliced correctly. An inspection plan that can utilise a map update, and cover an environment with a set of high quality images lends itself to a coverage planning technique called view planning.

2.1. View Planning to Solve Coverage Planning Problem

View planning is the process of finding a suitable set of discrete viewing positions that provide full coverage of an object or structure [4]. Instead of the coverage being determined from the layout of the environment, these methods base the coverage on the robot's viewing constraints. A survey [4] on 3D view planning describes the common approaches employed. Applying these approaches into a full inspection plan which includes the robot's motion through the environment involves breaking the planning into two steps. The first step is the generation of viewpoints. This is best represented by the art gallery problem which searches for the minimum number of guards to cover a

boundary environment. The second is the Travelling Salesman Problem (TSP), which finds the shortest paths between viewing positions to form an inspection plan [5].

View planning can fall into two categories called (i) model-based and (ii) non-model-based view planning. Model-based planning generates a plan offline from a known model, while non-model-based planning executes online with no prior knowledge and uses exploration algorithms such as Next-Best View (NBV) to generate coverage.

A useful technique for online planning is to first generate a plan offline. The benefit provided is that if no changes occur, the robot will execute the plan as expected. However, if new information is discovered while online, the algorithm can adapt the current plan and resolve the new coverage without replanning the entire route. Similar methods have been demonstrated for coverage planning [6, 7]. The preservation of pre-calculated plans is especially important for large planning problems in complex environments with robots of high DOF, due to the costly overheads associated with naïvely replanning everything, each time a new piece of information is detected.

The offline algorithm that we extend to work adaptively online is the sampling-based coverage planner algorithm presented in [8] developed for the full inspection of ship hulls using an autonomous underwater vehicle. This planner solves a *coverage sampling problem* (CSP) by constructing a *redundant roadmap* and then generating an inspection plan using a *multi-goal planning problem* (MPP) without placing assumptions on the environment. The online adaptive variant presented in this work extends the capability of the sampling-based coverage algorithm to reuse an offline plan, provide coverage for any new information, and only replan as appropriate to minimise recalculating a new inspection plan.

3. The Adaptive Sampling-Based Coverage Planner

The goal of this adaptive coverage planner is to maximise the coverage of the environment by adapting to newly detected features and replanning the coverage of these features into an existing inspection plan. It achieves this by extending the fundamental CSP and MPP of the offline algorithm to produce '*coverage*', and replanning new information which is bounded by the region of interest (ROI). A diagram of the adaptive algorithm is shown in Figure 1. The adaptive planner consists of three phases, (i) the *ROI Validation*, (ii) *Online CSP* and (iii) *Online MPP*. The *ROI Validation* is responsible for creating ROIs to determine which elements of the inspection plan are preserved or replanned. The *Online CSP* uses the functionality of the offline algorithm to generate coverage for the new information and refine any observations made by configurations of the existing plan within the ROIs. Finally, the *Online MPP* is responsible for replanning new paths, allowing the robot to adapt its plan online.

3.1. The Offline Sampling-Based Coverage Planner

The offline algorithm solves an inspection plan by solving the CSP and MPP in series. The definitions for the CSP, MPP and an analysis of the probabilistic completeness of the planner are found in [8]. The CSP is represented as a *set system (P, Q)*, where P is the finite set of geometric primitives p_i that comprise the structure, and Q is the robot configuration space, comprising configurations q_j. The CSP role is to generate a redundant set of discrete robot configurations q_j inside Q to cover all elements $p_i \in P$.

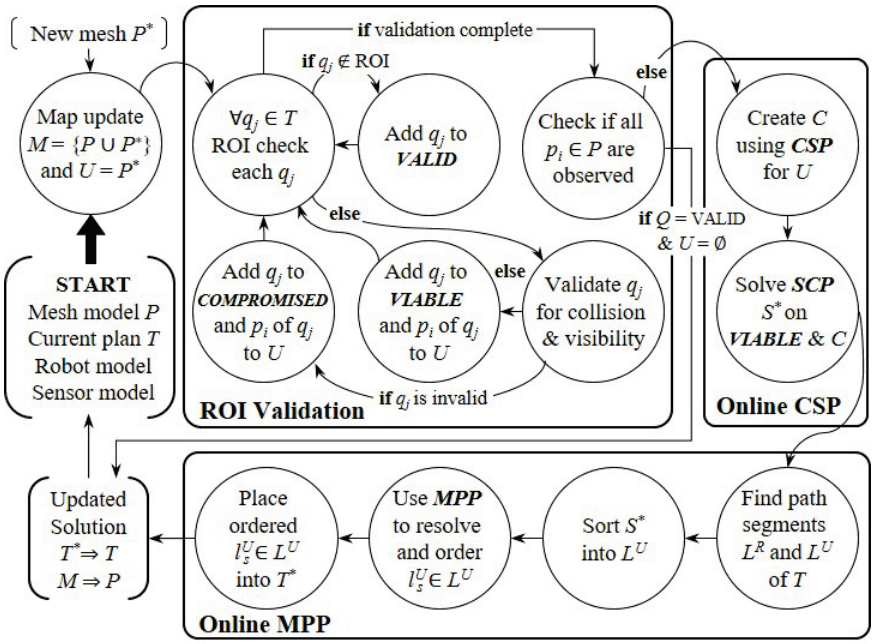

Figure 1. The adaptive sampling-based coverage planner

Therefore, as the *set system* implies, the observations made by a sensor at q_j, will map to a subset of P. To remove the redundancy, a set cover problem (SCP), a greedy algorithm is applied to approximate a minimum feasible set S that contains configurations that obtain distinct observations of all $p_i \in P$. A further pruning procedure is applied to remove any $q_j \in S$, that do not contain any unique $p_i \in P$ to create a set of goal configurations, G, that cover all P. The MPP uses the functionality of the *lazy point-to-point planner* to generate paths between $q_j \in G$ to generate a single collision-free inspection plan T.

3.2. Creating Newly Detected Features

Online adaptability depends upon the robot's perception and the ability of the mapping system to identify new information P^*. Let P^* be the set of all new primitives p_k^* that comprise a set of newly identified structures s.t $p_k^* \in P^*$. A simultaneous localisation and p_k^* mapping (SLAM) system is responsible for generating P^* from the LIDAR scans. These updates are created by determining which parts of the current scan returned by the sensor are associated with P, utilising Euclidean Segmentation [9]. Those parts of the scan associated with P are registered using Iterative Closest Point [10] generating global pose information which is used to correct the dead reckoning pose via an Extended Kalman Filter. The scans not associated with P, are meshed, using the Chisel algorithm [11], to form $p_k^* \in P^*$. P^* is merged into P by assimilating triangle vertices to create $M = \{P \cup P^*\}$. Therefore, when a new updated map M becomes available from the mapping system, M is supplied to the coverage planner to generate coverage for P^* and present an updated inspection plan T^*.

3.3. Generating Regions of Interest

For each M received, the adaptive coverage planner aims to preserve as much of the current T from replanning as possible whilst factoring in the new configurations q_n^* into T^*. This task is accomplished by bounding every $p_k^* \in P^*$ within its own region of interest (ROI_k). The neighbourhood that bounds each p_k^* is based on the sensor's maximum field of depth (FOD_{MAX}). Therefore, the set of all ROI_k is defined as,
$ROI = \{Q \mid dist(f_{xyz}(Q) - f_{xyz}(p_k^*)) < FOD_{MAX} \in \mathbb{R}, \forall\ p_k^* \in P^*\}$, for which f_{xyz} projects elements of a set into the Cartesian space. By bounding each p_k^* within its own $ROI_k \in ROI$, allows for disconnected sets of $p_k^* \in P^*$ to be replanned as separate regions. This approach makes no assumption to structures being received, whether they be complete or incomplete and will replan accordingly. The ROI also defines the space in Q to generate new configurations q_n^* to view P^* and to determine which $q_j \in T$ are within the ROI (Q^{ROI}). This separation determines which sections of T remain within T^* as calculated, or become candidates for replanning.

3.4. ROI Validation

Any $q_j \notin ROI$ are added to a *VALID* set. Elements of the *VALID* set are all assumed to be free of collision and still observe the same subsets of P as originally calculated. However, $q_j \in Q^{ROI}$ will have to be revalidated for collision and visibility. Depending on the influence of P^*, $q_j \in Q^{ROI}$ will either pass and be added to a *VIABLE* set or fail and be marked for removal in a *COMPROMISED* set. All elements in *COMPROMISED* are removed completely before replanning. As *VIABLE* configurations may now potentially observe P^*, they all become candidates for replanning within the ROI.

The ROI also serves to determine P^{ROI}, the subset of P that are viewed by Q^{ROI}. As P^* can impact the visibility of p_i viewed by Q^{ROI} it is important that P remains covered. Therefore, all p_i observed by Q^{ROI} are marked as unobserved U, where $U = \{P^{ROI} \cup P^*\}$. Despite elements of *VIABLE* observing a subset of P and P^* during re-validation, these primitives are not considered observed until new configurations q_n^* have the opportunity to observe U. This allows the configurations of the set *VIABLE* to be replaced with q_n^* that provide better coverage of U. However, if *VIABLE* covers all P^* with no $q_j \in$ *COMPROMISED*, no q_n^* are required. Therefore, the *Online CSP* is skipped, *VIABLE* is added to the *VALID* set and is accepted as the covering set of M. Otherwise, new paths need to be recomputed for configurations in *VIABLE*.

3.5. Online CSP

The online CSP is responsible for sampling new $q_n^* \in C$ to cover U. The same sampling procedure is implemented as in the offline algorithm, except it is generating coverage over U instead of P. Similarly, the offline SCP and pruning procedures are applied over the combination of sets C and *VIABLE* to produce the feasible set S^*. Any members of *VIABLE* that are removed by these procedures from S^*, are added to *COMPROMISED*.

In the offline variant, the CSP exploits areas of local planarity by generating sweep paths over flat surfaces [12]. Whilst beneficial for an offline approach, this online variant opts for a CSP that produces random samples instead of determining the structures to form sweep paths on P^*.

3.6. Region-Based Replanning via MPP

The generation of an *ordered set* of configurations has been identified as the most expensive operation of the original algorithm [13]. Therefore, the preservation of paths between configurations outside the *ROI* will minimise the computation needed to generate T^*. This problem can be solved by breaking T into path segments. When generating an offline T, there exists only one path segment from start to finish. However, when replanning occurs online, T may intersect multiple ROI_k. To distinguish between paths that are either outside or within the *ROI*, T is broken down into multiple path segments. Thus, separating T into path segments, allows segments outside the *ROI* to be preserved and those intersecting the *ROI*, to be replanned. That is, $T^*=\{L^R \cup L^U\}$, where L^R is the set of *resolved path segments* l_s^R outside the *ROI* and L^U is set of *unresolved path segments* l_s^U inside the *ROI*.

The *resolved path segments* l_s^R are constructed from stepping along T, and adding to l_s^R all $q_j \in VALID$ until q_j encounters an *entry gate* q_g^{Entry}. A q_g^{Entry} is the last $q_j \in VALID$ before entering the *ROI*, indicating that the path between $q_j \in VALID$ and $q_{j+1} \in Q^{ROI}$ will need to be replanned. Therefore, the set of all *entry gates* is defined as,
$q^{Entry} = \{q_j \in VALID \mid (q_j \in VALID) \wedge (q_{j+1} \in Q^{ROI}), \forall\ q_j \in T\}$. Conversely, the path from $qj \in Q^{ROI}$ to $q_{j+1} \in VALID$ will indicate the departure of the *ROI* and will create an *exit gate* q_g^{Exit}. Therefore, the set of *exit gates* q_g^{Exit} is defined as,
$q^{Exit} = \{q_j \in VALID \mid (q_j \in Q^{ROI}) \wedge (q_{j+1} \in VALID), \forall\ q_j \in T\}$. New l_s^R will resume construction upon encountering the next $q_j \in VALID$ in T.

Unresolved path segments l_s^U are constructed from the pairing of *ROI entry* and *exit gates* and serve as a path segment to plan into T^* subsets $r_a \subset S^*$, therefore defining
$L^U = \{q^{Entry} \cup r_a \subset S^* \cup q^{Exit}\}$. The subset r_a is added into each l_s^U by naïvely evaluating the point to line distances between each element of S^* between each pair of q_g^{Entry} and q_g^{Exit}. When all $l_s^U \in L^U$ are created, each l_s^U represents an unordered sub-plan that can be individually ordered back into T^* using the MPP. To restrict the TSP solver in the *lazy point-to-point planner* from forming a looped sub-plan but start and finish at the q_g^{Entry} and q_g^{Exit} respectively, calls on a solution to finding the shortest Hamiltonian path by introducing a *dummy configuration* [5]. A *dummy configuration* ties both gates together in an artificial loop resulting in the ordering of l_s^U. A similar method was presented in [12]. As the path segments of T are generated in order, upon completing each $l_s^U \in L^U$ the result forms T^*. The updated solution is provided to the robot and before the next M is supplied, the current $M \Rightarrow P$ and $T \Rightarrow T^*$. This process continues until robot has reached its final configuration and no map updates are detected.

4. Results and Discussion

To demonstrate the capability of the adaptive planner it is compared against a naïve planner. The naïve planner implements the offline CSP and MPP procedures to replan everything except the start and finish locations upon receiving M. The complexity of the environment and robot model are simplified to demonstrate the capability of the algorithm in the simplest form. It can be inferred that planning for a higher DOF robot in a complex environment will only increase computation time for both the CSP and MPP procedures. The first experiment demonstrates the scalability of the adaptive planner. A singular change (represented by a cylinder) is placed centrally in both a 2x2m and 4x4m box. This experiment demonstrates the planner's ability to preserve pre-

calculated plans for small changes in increasing sizes of environments (Figure 2a). A second experiment demonstrates the ability to handle multi-regions as it plans for three simultaneous changes concurrently (Figure 2b) within the 4x4 environment. Both planners were supplied with the same pre-calculated raster inspection plan, required to replan with a 3-redundancy using only random samples and provided with the same sensor configuration (FOD_{MAX} = 340mm). Each planner was initiated for replanning upon reaching the first q_j. Each planner was executed 50 times for each case experiment. The results for both planners are compared in Table 1.

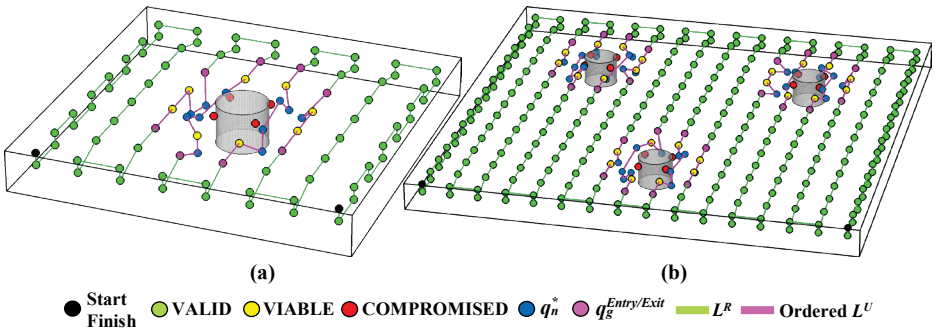

Figure 2. The adaptive planner replanning a) one change in a 2x2m environment and b) three concurrent changes in a 4x4 environment.

Table 1. Scalability of the adaptive planner vs naïve replanning over 50 trials.

Planner (Changes)	Time	CSP	MPP	$\#q_j$ [T^* (T)]	$\#[q_n^*$ (COMP.)]	Length
adaptive2x2(1)	0.26s	0.23s	0.006s	105.68 (98)	11.68 (4)	18.88m
naive2x2(1)	0.48s	0.34s	0.14s	129.7 (98)	127.7 (96)	19.30m
adaptive 4x4(1)	0.27s	0.23s	0.006s	330.30 (322)	12.3(4)	69.59m
naïve4x4(1)	2.22s	0.37s	1.84s	398.66 (322)	396.66 (320)	68.60m
adaptive 4x4(3)	0.75s	0.69s	0.02s	347.16 (322)	37.16 (12)	71.43m
naïve4x4(3)	3.68s	0.85s	2.82s	408.28 (322)	406.28 (320)	69.13m
Relative Performance – Adaptive relative to naïve (%)						
2x2(1)	53.47	68.14	4.12	81.48	9.15 (4.17)	97.74
4x4(1)	12.01	62.46	0.32	82.85	3.10 (1.25)	101.45
4x4(3)	20.37	81.04	0.74	85.03	9.14 (3.75)	103.32

Table 1 shows that the adaptive planner scales to the change in the environment while the naïve scales to the size of the environment. This behaviour is apparent in the one change examples as the adaptive planner shares similar execution times as opposed to the naïve approach. This scaling to change is also present in the experiments that contain three changes as the adaptive planner takes 2.7 times compared to the 4x4 with one change. This is due to the adaptive planner attempting to retain as much prior knowledge while the naïve planner discards all prior knowledge to solve the global solution again therefore increasing the MPP time. Despite the naïve planner generating more configurations as the global solution is resolved again, it results in the overall length being shorter than the adaptive solutions. This is expected as the adaptive planner does not consider that the environment topology may change and therefore replans problems locally within a ROI to bound start and finish locations. This results in a 47-88% reduction in computational time with at most a 3.3% increase in path length compared to the naïve approach. For more complex environments this may be evident and will

result in longer path lengths, but the adaptive solution will provide a feasible solution to overcome change at a cheaper computational cost than replanning naïvely.

5. Conclusion

This paper demonstrates an adaptive implementation of the coverage planner presented in [8] that can replan coverage for new detected features into an existing inspection plan. The adaptive algorithm attempts to make minimal changes by bounding new information into ROIs, therefore preserving any robot positions and paths that are not influenced by the new features. Results indicate that this method is suitable to provide feasible solutions that, scale to changes in and not the size of the environment, replans multiple changes concurrently, and is faster with minimal path degradation compared to a naïve approach.

Acknowledgment

The authors would like to acknowledge ASC Pty Ltd and the ARC Research Training Centre for Naval Design and Manufacturing (RTCNDM) for their support of this project, and Richard Bowyer for his assistance in helping to prepare this paper.

References

[1] Pivetta R., et al, (2017), “Submarine Tank Inspection Robots.”, 4th SIA Submarine Science, Technology and Engineering Conference 2017 (SubSTEC4).

[2] Galceran, E. and M. Carreras (2013). "A survey on coverage path planning for robotics." Robotics and Autonomous Systems 61(12): 1258-1276.

[3] Choset, H. (2001). “Coverage for robotics - A survey of recent results.” Annals of Mathematics and Artificial Intelligence 31: 113-126.

[4] Scott, W. R., et al. (2003). “View planning for automated three-dimensional object reconstruction and inspection.” ACM Computing Surveys. 35(1): 64-96.

[5] Cook, W. J. (2011). “In pursuit of the traveling salesman: mathematics at the limits of computation.” Princeton University Press.

[6] Galceran, E., et al. (2015). “Coverage Path Planning with Real-time Replanning and Surface Reconstruction for Inspection of Three-dimensional Underwater Structures using Autonomous Underwater Vehicles.” Journal of Field Robotics 32(7): 952-983.

[7] Sehestedt, S., et al. (2013). “Prior-knowledge assisted fast 3D map building of structured environments for steel bridge maintenance.” 2013 IEEE Int. Conference on Automation Science and Engineering.

[8] Englot, B. and F. S. Hover (2013). “Three-dimensional coverage planning for an underwater inspection robot.” The International Journal of Robotics Research 32(9-10): 1048-1073.

[9] Rusu, R. B. (2010). "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments." KI - Künstliche Intelligenz 24(4): 345-348.

[10] Besl P. J., and McKay N. D., (1992). "A method for registration of 3-D shapes." in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 2, pp. 239-256.

[11] Klingensmith M., Dryanovski I., Srinivasa S. S., Xiao J., (2015), “Chisel: Real time large scale 3D reconstruction onboard a mobile device using spatially hashed signed distance fields.”

[12] Englot, B. and F. S. Hover (2012). “Sampling-based sweep planning to exploit local planarity in the inspection of complex 3D structures.” 2012 IEEE/RSJ Intel. Conf. on Intelligent Robots and Systems.

[13] Englot, B. and F. Hover (2017). “Planning Complex Inspection Tasks Using Redundant Roadmaps.” Robotics Research: The 15th International Symposium ISRR. H. I. Christensen and O. Khatib. Cham, Springer International Publishing: 327-343.