International Journal of Control, 2014
Vol. 87, No. 4, 787–801, http://dx.doi.org/10.1080/00207179.2013.858829

Navigation of autonomous vehicles for oil spill cleaning in dynamic and uncertain environments†

Xin Jin[a] and Asok Ray[b,*]

[a]*A.O. Smith Corporate Technology Center, Milwaukee, WI 53224, USA;* [b]*Department of Mechanical and Nuclear Engineering, Pennsylvania State University, University Park, PA 16802, USA*

(*Received 12 June 2013; accepted 19 October 2013*)

In the context of oil spill cleaning by autonomous vehicles in dynamic and uncertain environments, this paper presents a multi-resolution algorithm that seamlessly integrates the concepts of local navigation and global navigation based on the sensory information; the objective here is to enable adaptive decision making and online replanning of vehicle paths. The proposed algorithm provides a complete coverage of the search area for clean-up of the oil spills and does not suffer from the problem of having local minima, which is commonly encountered in potential-field-based methods. The efficacy of the algorithm is tested on a high-fidelity player/stage simulator for oil spill cleaning in a harbour, where the underlying oil weathering process is modelled as 2D random-walk particle tracking.

Keywords: autonomous agents; oil spill; complete coverage; unknown environment; path planning

1. Introduction

The recent Deepwater Horizon oil spill in the Gulf of Mexico has attracted the attention of the world community due to its colossal ecological, economic and social impacts. Over 210 million gallons of crude oil was released and the slicks and sheen of the surface oil directly affected over 180,000 km^2 of ocean surface. To clean this oil spill, over 39,000 personnel, 5000 vessels and 110 aircraft were involved, over 700 km of booms have been deployed, 275 controlled burns have been carried out, approximately 27 million gallons of oily liquid has been recovered by skimmers, and more than 1.5 million gallons of chemical dispersant has been used in these efforts (Zhong & You, 2011).

In view of the facts that the current oil spill cleaning technology is labour-intensive and the toxic chemicals and oil vapours are pernicious to the health of the cleaning crews, there is a pressing need for the development and implementation of new technologies for combating oil spills. To mitigate the adverse environmental effects of an oil spill, research efforts are being focused on development of technologies to remove the oil *in situ*, minimise operational time, and protect the health and safety of the cleaning crew (Kakalis & Ventikos, 2008). To this end, several novel methods have been developed to make use of autonomous vehicles for effective oil spill confrontation, such as Seaswarm (MIT Senseable City Lab, 2010) and Protei (Protei: Open Source Sailing Drone, 2010) that are intended to work as a fleet or 'swarm' of vehicles to create an organised system for autonomous ocean skimming and oil removal. While the current trend emphasises hardware improvement, advanced software-based navigation algorithms are yet to be developed.

This paper develops a multi-resolution method for autonomously cleaning oil spills in dynamic and uncertain environments by alleviating the restriction of static environments (Acar & Choset, 2002; Cai & Ferrai, 2009; Choset, 2000; Garcia & de Santos, 2004; Jin, Gupta, Luff, & Ray, 2012; Oh, Choi, Park, & Zheng, 2004) that may forbid adaptation to the changing weathering process of the oil spill. In particular, dynamic adaptation after detection of oil spills would allow the autonomous vehicle to replan its actions online. Furthermore, the proposed method enables navigation at different levels of resolution to achieve complete coverage of the environment. The underlying algorithm is validated on a Player/Stage platform that is capable of high-fidelity simulation of autonomous vehicles and oil weathering processes for comparison with the benchmark algorithm of back and forth (i.e. zigzag) motion. In this context, major features of the proposed multi-resolution method are delineated as follows:

- The algorithm provides a complete coverage of the unknown environment for oil spill cleaning.
- The algorithm enables effective cleaning of oil spills via dynamic adaptation after their detection.

* Corresponding author. Email: axr2@psu.edu

† A preliminary version of this paper was presented by X. Jin and A. Ray as 'Coverage Control of Autonomous Vehicles for Oil Spill Cleaning in Dynamic and Uncertain Environments', Proceedings of the American Control Conference, Washington, DC, June 2013, pp. 2600–2605.

- The low computational complexity of the algorithm is expected to enable real-time implementation on commercial autonomous vehicles.

The paper is organised in six sections (including the present section) and two appendices. Section 2 reviews the related work and highlights the distinct features of the proposed method. Section 3 presents the modelling of the oil spill process and construction of the search space. Section 4 describes the multi-resolution navigation algorithm for autonomous cleaning of oil spills in dynamic and uncertain environments. Section 5 provides an illustrative example of oil spill cleaning in a harbour environment by the proposed algorithm. Section 6 summarises and concludes the paper with a brief discussion on future work. Appendix 1 presents a brief background of oil weathering process, as well as its modelling and simulation. Appendix 2 addresses the important features of the multi-resolution navigation algorithm.

2. Related work

Technical literature abounds with diverse path-planning algorithms that address the above problem to various degrees of autonomy. Several algorithms have been proposed based on the artificial potential functions (Kim & Khosla, 1992; Valavanis, Hebert, & Tsourveloudis, 2000), where the local potentials generate a virtual force field to navigate the autonomous vehicles. Potential-field-based methods have been widely studied due to their simplicity and elegance; however, these methods have inherent problems such as trap situations (e.g. local minima and cyclic behaviour) and no passage between closely spaced obstacles (Koren & Borenstein, 1991). A variety of methods have been proposed to circumvent these problems (Ge & Cui, 2000; Mabrouk & Mclnnes, 2008). Recent path-planning algorithms rely on sensor-based exploration (Lee & Choset, 2005), which focus on real-time planning to reach the goal from a start point by avoiding obstacles based on sensor readings. For example, Cowlagi and Tsiotras (2012) have proposed a multi-resolution path-planning algorithm via wavelet-based cellular decompositions.

The *a priori* information, as needed by autonomous vehicles for oil spill cleaning in dynamic and uncertain environments, is often either incorrect or incomplete. Therefore, time-critical operations of these vehicles require real-time decisionmaking to facilitate continuous adaptation of the evolving information that is generated *in situ* by onboard sensing and pattern analysis. The generated information refers to the observed phenomena that relate to dynamic unfolding of the search area (e.g. detection of unknown obstacles and boundaries) and environmental changes (e.g. spreading and drift of oil spills). Although such information can be obtained through remote sensing Engelhardt (1999), it may not be always available due to communication constraints and high operational cost. Under these circumstances, the autonomous vehicle is required to scan all points in the search area while dynamically discovering new oil spills and avoiding obstacles at a priori unknown locations. This is known as the *complete coverage problem* (Hert, Tiwari, & Lumelsky, 1996; Lumelsky, Mukhopadhyay, & Sun, 1990). A variety of path-planning algorithms exist in technical literature for coverage of the search area using autonomous vehicles; a review of such algorithms is reported in Choset (2001). Recent coverage algorithms are based on cellular decompositions of the search area at critical points of the Morse functions that correspond to obstacle extremities (Acar & Choset, 2002; Choset, 2000; Garcia & de Santos, 2004). An approximate cell decomposition method is developed in Cai and Ferrai (2009), where a connectivity graph is constructed to generate an optimal sensing strategy. In the authors' recent work (Jin et al., 2012), a multi-resolution navigation algorithm is proposed to enable autonomous vehicles to explore the unknown and static environment, and cover the entire search area. A similar multi-resolution algorithm is proposed in Cowlagi and Tsiotras (2012) using wavelet-based cellular decompositions, but not in the context of complete coverage. None of these completecoverage methods address the issue of dynamic adaptation to the changes in the environment, which is essential for adaptive cleaning of oil spills.

Yang and Luo (2004) have proposed a neural network-based approach, where the robot is able to achieve complete coverage in an environment with abrupt changes that are the only dynamic phenomena in the environment, and the issue of adaptation to the target is not addressed. Gupta, Ray, and Phoha (2009) used the concept of a generalised Ising model for dynamic adaptation to the lattice sites in the presence of targets. However, the modelling of the environment in Gupta et al. (2009) is overly simplified in the sense that no obstacles are considered. This issue is addressed by the authors in their previous work (Jin, Luff, Gupta, & Ray, 2010), where the concept of path potential is used for obstacle avoidance. To achieve complete coverage of the search space (i.e. when there is no unexplored lattice site left in the current neighbourhood of the autonomous vehicle), the neighbourhood size is expanded to find the closest unexplored lattice site, and then a series of waypoints are created for the autonomous vehicle by using a backtracking algorithm. However, the method proposed in Jin et al. (2010) can be extremely computationally expensive and time consuming, especially in a large and complex environment. It is also noted that, in the work reported in Jin et al. (2010), the target is static, and it may not be a good representation of oil spills.

As an augmentation of the authors' previous work (Jin et al., 2010, 2012), the current paper addresses the issue of oil spill cleaning as a tradeoff between complete coverage and dynamic adaptation. The notions of local navigation

and global navigation are adopted to enable navigation at different levels of resolution, depending on the available spatio-temporal information. During local navigation, the autonomous vehicle operates at the level of finest resolution and can deviate from its original navigation plan upon detection of oil spills and search the neighbouring areas where spills are detected. The autonomous vehicle is able to 'zoom out' on the grid map and navigates at a level of coarser resolution when no unexplored cells are left nearby. With the proposed method, the autonomous vehicle is able to clean up the spills more efficiently than the existing methods that may not be able to adapt to the spreading and drift of the spills.

3. Modelling of oil spill phenomena and cleaning

Although many prototypes of autonomous vehicles have been developed for oil spill cleaning, navigation algorithms for these prototypes are not adequately addressed (MIT Senseable City Lab, 2010; Protei: Open Source Sailing Drone, 2010). For example, several researchers (e.g. Kakalis & Ventikos, 2008; Pereda, de Marina, Giron-Sierra, & Jimenez, 2011; Zhang, Fricke, & Garg, 2011) have tested the algorithms for control of autonomous vehicles with relatively simple scenarios; however, in the real-world applications, the autonomous vehicles must carry out the clean-up tasks in more complicated scenarios, such as obstacle-rich environments that contain islands and other cleaning vessels. This evinces the need for development of navigation algorithms that will enable autonomous vehicles to perform oil cleaning tasks in dynamic and uncertain environments, where the locations of obstacles and oil spills are *a priori* unknown. From these perspectives, the operation of the oil spill process is modelled under the following assumptions:

(1) After occurrence of an oil spill at a physical location, the spillage stops before initiation of the clean-up task.
(2) The location and volume of oil spill are unknown to the autonomous vehicle but the exact location of the vehicle is known through a localisation system.
(3) The autonomous vehicle uses mechanical clean-up to remove the oil at its current position (see Appendix 1.1)
(4) The dynamic behaviour of the oil weathering process can be predicted by an oil transport and weathering model.

The remaining part of this section presents the formulation of oil weathering model, the autonomous vehicle, and the search space. The spreading and drift process of the oil spill is modelled by two-dimensional (2D) random walk. The autonomous vehicle is equipped with localisation system, navigation sensor, and oil detection sensor. The search space is partitioned and a lattice structure is constructed for autonomous exploration of the search space. Formulation of the search space is then extended in the multi-resolution sense to enable navigation in diverse and complex environments.

3.1 *Oil spill modelling*

As stated in Appendix 1.1, over 50 oil weathering models have been reported in the literature. The 2D random-walk particle tracking model has been adopted in this paper because the model is computationally tractable when simulating a large number of particles online, and it predicts the time trajectories of the spill size and the probability distribution of the oil spill.

In the random-walk particle tracking model, spilled oil consists of a large number of particles, with each particle representing a defined quantity of oil. Effectively, model particles are treated as 'mass points', with their transport determined by tidal currents, wind-driven current, turbulent eddies, gravitational spreading, and buoyancy. The 2D update equations (Arega & Sanders, 2004; Dimou & Adams, 1993) for particle positions are given by

$$\mathbf{X}^n = \mathbf{X}^{n-1} + \mathbf{A}(\mathbf{X}^{n-1})\Delta t + \mathbf{B}(\mathbf{X}^{n-1})\mathbf{Z}\sqrt{2\mathbf{K}\Delta t}, \quad (1)$$

where Δt is the time interval, $\mathbf{X}^n$ is the position at time $n\Delta t$ (i.e. at the step number n), $\mathbf{A}$ is a forcing vector that models the drift process due to currents and wind, $\mathbf{B}$ is a deterministic scaling matrix, $\mathbf{Z}$ is a vector of two independent random numbers taken from a uniform distribution in the range $[-1, 1]$ and $\mathbf{K}$ is a vector of the turbulent coefficients. In this model, the motion of one particle is statistically independent of other particles. As seen in Equation (1), the displacement of each particle is determined by its previous position and the effects of drift and spreading. The effects of other weathering processes (e.g. evaporation, natural dispersion and emulsification) are not included in this model.

The oil weathering process is simulated by following the model in Equation (1), where the simulated scenario is an oil spill incident in a harbour. The oil spillage is assumed to stop before the clean-up process is initiated; however, the spill starts to spread and drift and would eventually hit the port if preventive actions are not taken. Figure 1 shows oil particle distribution in the top row and concentration in the bottom row at different time instants, namely, at the beginning of the clean-up (left column) and after 10,000 steps of simulation (i.e. after 3000 seconds for $\Delta t =$ 0.3 seconds). It is seen in Figure 1 that the size of the spill increases due to spreading, and the spill moves toward the port due to tidal current and wind. This scenario is also used in Section 5 where autonomous vehicle is dispatched to clean up the oil spill.

Unlike several previous studies in the complete coverage problem (Acar & Choset, 2002; Choset, 2000; Hert

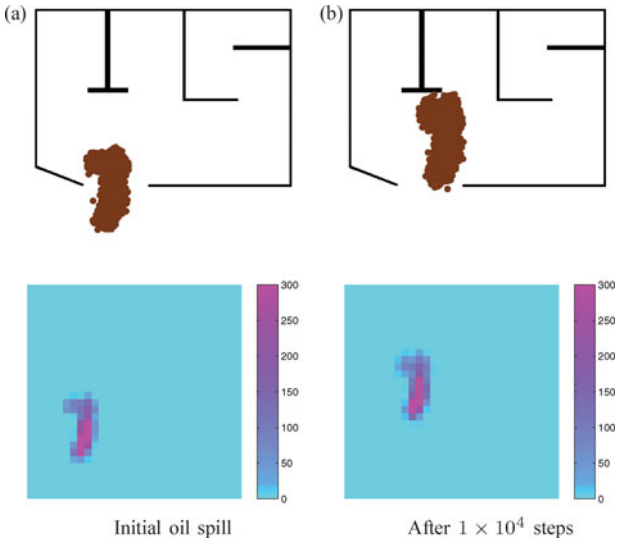

Figure 1. Spreading and drift of the oil spill due to wind and current. (a) Initial oil spill. (b) After 1×10^4 steps.

et al., 1996; Lumelsky et al., 1990), this paper assumes that the autonomous vehicle occupies a finite area instead of being a point mass. For example, the autonomous vehicle could be considered to have a circular body or a rectangular body. Thus, the opening space between any two obstacles through which the vehicle can pass is lower bounded by its size parameters (e.g. diameter or width). Furthermore, the autonomous vehicle has the knowledge of its exact location at all times (which is available from a *Global Positioning System*) and is equipped with navigation sensors (e.g. ultrasonic, infrared or LIDAR) to detect obstacles within an area of radius R around the vehicle body, as illustrated in Figure 2, where three areas are defined in the current formulation. The largest one is the navigation scan area, which is covered by the navigation sensors. The medium one is the local navigation area, where the locally optimal navigation

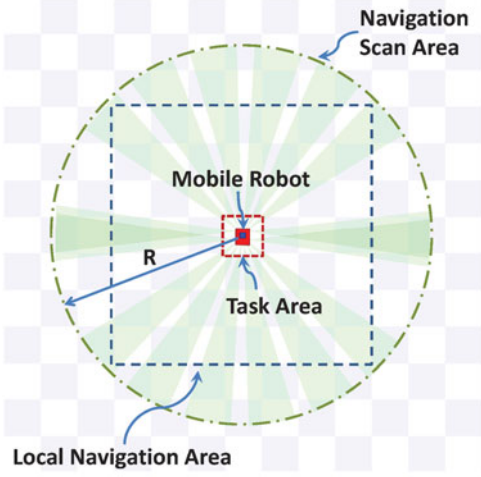

Figure 2. Autonomous vehicle and its sensing and computation areas.

decisions are made. The smallest one is the task area within which the autonomous vehicle carries out oil clean-up task. The autonomous vehicle is also equipped with an oil sensor that is able to detect oil and other hydrocarbons in water.

3.2 *Search space: partitioning and lattice formulation*

The environment to be explored is considered to be a planar area populated with a finite but unknown number of obstacles. The obstacles may have arbitrary shapes and sizes, and their exact locations are a priori unknown. The terrain limits are defined either by a hard boundary (e.g. a wall) or by a soft boundary (e.g. subarea of a larger field). To this end, pertinent definitions are presented below.

Definition 3.1 (Partition): Let $\mathcal{S} \subset \mathbb{R}^2$ be the search space. A partition $\mathcal{P}$ of $\mathcal{S}$ is defined as $\mathcal{P} \triangleq \{P_\xi : \xi = 1,), |\mathcal{P}|\}$, such that $\mathcal{S}$ is divided into mutually exclusive and exhaustive cells, i.e. $P_\xi \bigcap P_\nu = \emptyset \, \forall \, \xi \neq \nu$ and $\bigcup_{\xi=1}^{|\mathcal{P}|} P_\xi = \mathcal{S}$, respectively.

Definition 3.2 (State of a lattice site): Let Ω: $\mathcal{P} \times \mathcal{T} \rightarrow \Sigma$ be a mapping that assigns each symbol in the alphabet Σ to a cell of the partition $\mathcal{P}$ at each time epoch $t \in \mathcal{T}$, where $\mathcal{T}$ is the time span. Then, the state of a lattice site ξ at time $t \in \mathcal{T}$ is defined as

$$\gamma_\xi(t) \triangleq \Omega(P_\xi, t). \tag{2}$$

Definition 3.3 (Lattice system): The lattice system is defined as a set $\mathcal{L} \triangleq \{\mathcal{P}, \Gamma\}$, where $\mathcal{P}$ is a partition set of the search space $\mathcal{S}$ and Γ is the collection of the state of the lattice.

Remark 3.1: It follows from Definitions 3.1 and 3.2 that the partition $\mathcal{P}$ forms a grid of the search space $\mathcal{S}$. The partition is constructed such that the dimensions of each element (i.e. cell) of the grid structure fall within the task area where the autonomous vehicle carries out its oil spill cleaning task.

The search region is partitioned into a grid to form a finite-dimensional lattice structure such that each grid element (i.e. a cell) represents a lattice site. A generalised Ising model (Gupta et al., 2009) is constructed over the lattice, which involves an exogenous time-varying potential function term to control the movement of the autonomous vehicle in the search space. The construction of an energy potential of this spin model is similar to the pheromone of biological systems, which acts as a navigator to search for critical targets.

Once the spatial partitioning is done to construct a grid, the next step is to define a lattice, where each site of the lattice is isomorphic to a grid cell and represents a physical state of that cell. Therefore, the terms 'grid cell' and 'lattice site' are used interchangeably in this paper. Let $\Sigma \triangleq \{\sigma_j : j = 1, \ldots, |\Sigma|\}$ be a finite set of symbols,

called the alphabet, which also determines all possible states for each partition cell $P_\xi \in \mathcal{P}$. For a standard Ising model Pathria (1996), such an alphabet denotes the up and down states of the spin orientations. In this paper, the physical description of the state of each partition cell is described by an alphabet Σ that is constructed with four possible symbols (i.e. $|\Sigma| = 4$). These symbols are defined as: $\sigma_1 = T$, $\sigma_2 = E$, $\sigma_3 = U$ and $\sigma_4 = O$, representing the following possible states of each partition cell: (1) *explored and target present*, (2) *explored and target not present*, (3) *unexplored* and (4) *explored and obstacle detected*, respectively. The term *target* refers to oil spill in this paper. In essence, these states represent the four possible conditions of a partition cell. The objective here is to facilitate adaptive decision making based on the observed states of the neighbourhood.

The configuration space of the autonomous vehicle at time $t \in \mathcal{T}$ is constructed as the Cartesian product,

$$\Gamma(t) = \bigotimes_{\xi=1}^{|\mathcal{P}|} \gamma_\xi(t), \tag{3}$$

where Γ is the collective state of the lattice that can have at most $|\Sigma|^{|\mathcal{P}|}$ possible state configurations and $|\mathcal{P}|$ is the cardinality of the (finite) partition set $\mathcal{P}$.

As the autonomous vehicle continuously searches different cells on the lattice, while moving from one cell to another, the collective state Γ of the lattice unfolds in space and time by exhibiting different configurations that represent an evolution of the checkerboard pattern in the search space.

3.3 *Multi-resolution formulation of the search space*

The local navigation area of an autonomous vehicle is usually the local neighbourhood of the current grid cell for computational efficiency. However, when there are no unexplored grid cells left in the local navigation area, the autonomous vehicle has to increase the size of the neighbourhood to find the possible unexplored grid cell further away (Jin et al., 2010). This method is not efficient, especially when the unexplored grid cells are far away from the vehicle, the local neighbourhood size needs to be increased to as large as the entire search space to cover these cells. This paper introduces the concept of multi-resolution navigation that partitions the search space at various levels of resolutions and uses the corresponding grid map for navigation according to the available spatio-temporal information.

The search space is coarsely and uniquely partitioned so that information can be consistently stored by the autonomous vehicles. The entire search space is approximately divided into two halves in both dimensions, keeping the fine grid cells intact, to form four cells. Then, the search space is divided into two halves in each dimension, again keeping cells intact and forming 16 cells in total. The procedure continues until the size of the local navigation area is reached. As the last step, the local navigation area is further divided into finer grid cells such that these cells have the same size or slightly smaller than the local neighbourhood of the vehicle. This size requirement is that the entire cell at the lowest level falls within the local neighbourhood of an autonomous vehicle and can be completely scanned at the local level. The level of the finest grid cells is denoted as level 0 and the lowest level where each cell has about the same size with the local neighbourhood is denoted as level 1. This procedure continues until the coarsest level that covers the entire search space.

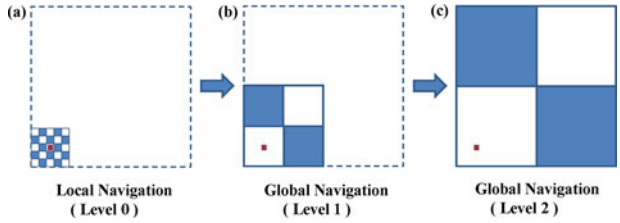

Figure 3. Illustration of the switch between local and global navigation. (a) Local navigation (Level 0). (b) Global navigation (Level 1). (c) Global navigation (Level 2).

Figure 3 shows the switch between the local and global navigation. In Figure 3, the search space is partitioned at three levels of resolution. The lattice system described in Section 3.2 corresponds to the level 0 that is the level of the finest resolution. The grid cell at level 1 defines the local navigation area that consists of 5×5 finest grid cells at level 0. Four grid cells of level 1 form a grid cell of level 2, which is the coarsest level. The autonomous vehicles first operate in level 0 until no unexplored grid cell remains in its local neighbourhood. Then global navigation with level 1 is implemented to find unexplored cells, and the autonomous vehicles moves toward the centroid of the cell that has the most unexplored fine grid cells. If no unexplored grid cells are found, then the autonomous vehicle continues to switch to the coarser level until unexplored cells are found. If no unexplored cells are found at the coarsest level, then the complete coverage task has been accomplished. As shown in Figure 3, among the various levels, only the coarsest level covers the entire search space and has the real 'global' view of the search space. This formulation avoids unnecessary global calculations and reduces the computational complexity in real-time implementation. Integration of local to global navigation is described in the next section.

4. Algorithms of multi-resolution navigation

This section presents the algorithms of multi-resolution navigation with the capability of dynamic adaptation. The multi-resolution framework seamlessly integrates the

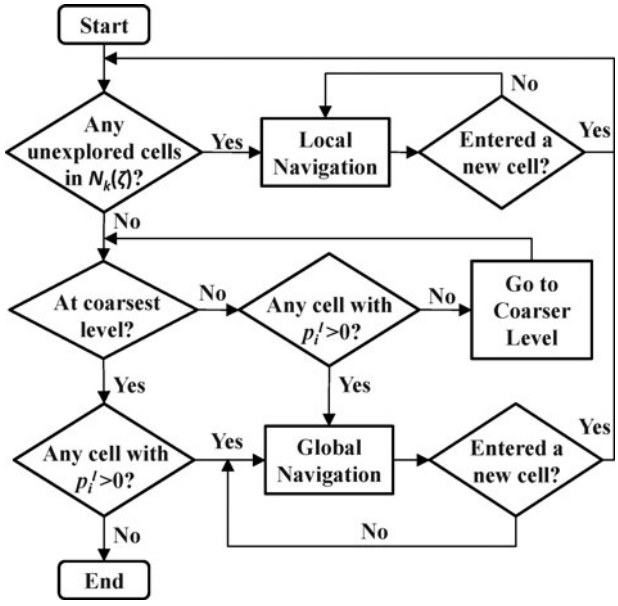

Figure 4. Flowchart of a multi-resolution navigation algorithm.

algorithms of local and global navigation. In this paper, the generalised Ising model is used for local navigation, and a probability vector is used for navigation when there is no ideal or unexplored points within the local neighbourhood. Generating the probability vector is an effective way to circumvent the difficulties encountered in Jin et al. (2010), where expansion of the neighbourhood size can be computationally expensive in complex environments. The flowchart of the multi-resolution navigation is shown in Figure 4.

The autonomous vehicle is assumed to have access to its current coordinate $c(t) \in \mathcal{P}_\mu$ with the grid cell coordinate μ and navigation sensor readings $\{s_i(t)\}$ at all times. It stores the environment information by building the grid map $\mathcal{M}(t)$. As shown in Figure 4, the autonomous vehicles first check if there is any unexplored grid cells in its local neighbourhood. If so, it performs local navigation; otherwise, it enters the global navigation mode. In both local navigation and global navigation, the autonomous vehicle computes the goal when it enters a new grid cell instead of computing it at every step. This reduces the computation load while maintaining the adaptability to environment changes. The switching between local navigation and global navigation is unidirectional, i.e. the coarseness of the level can only increase during the operation and is reset to the level 0 when it enters a new grid cell (see Figure 3). It is noted that the switch can be made bidirectional at the expense of complexity and loss of robustness in real-time implementation.

Several important features of the multi-resolution algorithm are outlined in Appendix 2. The features include complete coverage of the unknown environment, clean-up of all oil spills, no local minima problem, and low computational complexity.

4.1 *Generalised Ising model for local navigation*

A four-state generalised Ising model is now constructed over the lattice system $\mathcal{L}$ by extending the earlier work of Gupta et al. (2009). A local (implicitly time-dependent) energy term $E_\xi^{\mathcal{L}}$ at a lattice site ξ is defined as

$$E_\xi^{\mathcal{L}}(t) = \sum_{<\xi,\nu>_{\kappa_1}} J_{\xi\nu}\Psi(\gamma_\xi(t), \gamma_\nu(t)) + \Phi(B_\xi(t), \gamma_\xi(t)), \tag{4}$$

where $< \xi, \nu >_{\kappa_1}$ implies summation over a κ_1-neighbourhood of ξ, for some $\kappa_1 \in \mathbb{N}$. The κ-neighbourhood of a lattice site ξ is defined as

$$\mathcal{N}_\kappa(\xi) = \{\nu : \max(|\xi_x - \nu_x|, |\xi_y - \nu_y|) \leq \kappa\}, \tag{5}$$

where $\xi_x, \xi_y \in \mathbb{N}$ and $\nu_x, \nu_y \in \mathbb{N}$ denote the x and y coordinates of lattice sites ξ and ν, respectively. The computation of Equation (4) is carried out in $\mathcal{N}_{\kappa_0}(\mu)$, i.e. the κ_0-neighbourhood of μ, where μ is the grid cell occupied the autonomous vehicle, and κ_0 is the distance between μ to the boundary of local navigation area.

Figure 5 illustrates different neighbourhoods in the local navigation. The dashed-boundary box indicates κ_1-neighbourhood of ξ, where the summation in the first part of Equation (4) is implemented. The solid-boundary box shows κ_0-neighbourhood of the grid cell μ occupied by the autonomous vehicle. $\mathcal{N}_{\kappa_0}(\mu)$ is equivalent to the local navigation area that consists of 5 $\times$ 5 grid cells. Part of the dashed-boundary box is out of bound, since ξ is on the boundary of $\mathcal{N}_{\kappa_0}(\mu)$. These out-of-bound cells are not considered in the summation of Equation (4). The energy term shown in Equation (4) is calculated for each $\xi \in \mathcal{N}_{\kappa_0}(\mu)$. In the example shown in Figure 5, $\kappa_0 = 2$ and $\kappa_1 = 1$.

The first term in the right-hand side of Equation (4) defines the total interaction potential due to the sum of the effects of neighbours on the state at a lattice site ξ. This term

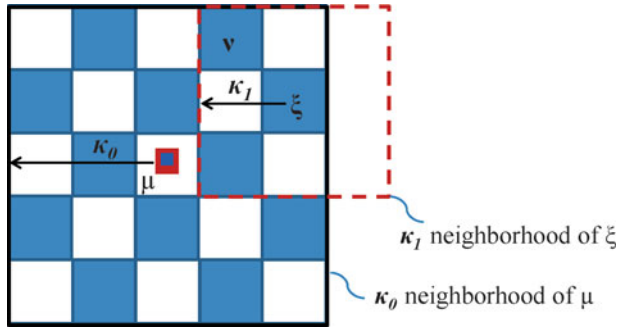

Figure 5. Illustration of different neighbourhoods in local navigation.

is called *adaptation term* because the effects of the observed states in the neighbourhood cause changes in the resultant energy potential at a lattice site ξ, which enables real-time adaptation in the navigation path trajectory. The coefficient $J_{\xi\nu}$ denotes the interaction strength between two distinct lattice sites ξ and ν. For $\eta = \max(|\xi_x - \nu_x|, |\xi_y - \nu_y|)$, i.e. the distance between two neighbourhood sites, $J_{\xi\nu}$ is given as

$$J_{\xi\nu} = \begin{cases} \eta^{-\alpha}, & \forall \xi \neq \nu \text{ and } \eta \in \{1, \ldots, \kappa_1\}, \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

where $\alpha \in (0, \infty)$ is a control parameter. The (implicitly time-dependent) interaction function Ψ is defined as

$$\Psi\left(\gamma_\xi(t), \gamma_\nu(t)\right) = \begin{cases} \psi_T, & \text{for } \gamma_\xi(t) = U, \gamma_\nu(t) = T \\ \psi_E, & \text{for } \gamma_\xi(t) = U, \gamma_\nu(t) = E, \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

which implies the following conditions:

(1) Any lattice site ξ, whose state is either $\gamma_\xi(t) = T$ (i.e. *an explored site where a target is present*) or $\gamma_\xi(t) = E$ (i.e. *an explored site where no target is present*) or $\gamma_\xi(t) = O$ (i.e. *an obstacle*) is not influenced by the state of the neighbourhood. Therefore, the interaction potentials of sites that are in the above states are zero.
(2) All neighbourhood sites with a state $\gamma_\nu(t) = U$ (i.e. *unexplored site*) exert no influence on any lattice site because they provide no information to the neighbourhood.
(3) ψ_T defines the influence of a site ν in the neighbourhood with $\gamma_\nu(t) = T$ (i.e. *an explored site where a target is present*) on a site ξ with $\gamma_\xi(t) = U$ (i.e. *an unexplored site*).
(4) ψ_E defines the influence of a site ν in the neighbourhood with $\gamma_\nu(t) = E$ (i.e. *an explored site where no target is present*) on an unexplored site ξ.

The interaction function Ψ has the following physical interpretation. A target detection at a certain lattice site causes distortion in the space-time potential field in the local neighbourhood of the target resulting in an increase in energy by ψ_T, scaled to the interaction strength, thereby creating a dome-like structure. Therefore, as the autonomous vehicle scans the area, the collective state Γ of the lattice (see Equation (3)) unfolds in the space-time coordinates and a detected target's neighbourhood with localised increase in energy becomes a high priority area. In this neighbourhood, the unexplored sites (i.e. having a state $\gamma_\xi(t) = U$) with a high interaction potential tend to settle down to low energy states (i.e. *explored sites*). Thus, the autonomous vehicle follows the high potential sites and, by scanning, turns them to low energy states by conversion to explored sites that have no neighbourhood interactions. If another target is detected in the neighbourhood, it generates its own high interaction potential, which leads to constructive interference with the potential of an earlier target, and so on. Therefore, following high potential sites leads to an adaptation in the nominal trajectory of the autonomous vehicle such that the high priority areas are scanned earlier, thereby improving localised search performance. The interaction function Ψ shows that the explored sites, where no target is detected, also exert relatively small influences on the neighbouring unexplored sites, where the energy increases by a factor ψ_E. Therefore, Γ tends to unfold in the neighbourhood of explored sites with a priority, thereby generating a more uniform and orderly search. The construction of energy functional for adaptation of an autonomous vehicle is conceptually similar to that of the chemical pheromone used for tracking by *biological systems* such as an ant.

Algorithm 1 Local Navigation Algorithm.

Input: current position $c(t) \in \mathcal{P}_\mu$, grid map $\mathcal{M}$
Output: updated grid map $\mathcal{M}$

```
calculate the local energy term E^L_{ξ_i}, ∀ cell ξ_i ∈ N_{κ_0}(μ)
set the centroid of grid cell ξ* = argmax_i E^L_{ξ_i} as the navigationgoal
if front distance to obstacle ≤ threshold d then
    use Bug2 algorithm for obstacle avoidance
else
    move toward the navigation goal
    get current coordinate c(t) and sensor readings {s_i}
    update grid map M using c(t) and {s_i}
end if
```

The second term on the right-hand side of Equation (4) defines the navigation control function Φ that depends on an exogenous time-varying potential field $B_\xi(t)$ and the state $\gamma_\xi(t)$ at a lattice site ξ. The (implicitly time-dependent) function Φ is defined as

$$\Phi\left(B_\xi(t), \gamma_\xi(t)\right) = \begin{cases} \phi_T, & \text{for } \gamma_\xi(t) = T \\ \phi_E, & \text{for } \gamma_\xi(t) = E \\ \phi_O, & \text{for } \gamma_\xi(t) = O \\ B_\xi(t), & \text{for } \gamma_\xi(t) = U, \end{cases} \tag{8}$$

where the constants $\phi_T \leq 0$, $\phi_E \leq 0$ and $\phi_O < 0$ correspond to low energy states of the explored sites in the presence (i.e. $\gamma_\xi(t) = T$) and absence (i.e. $\gamma_\xi(t) = E$) of a target, and the presence of obstacle (i.e. $\gamma_\xi(t) = O$), respectively. As described earlier, the explored sites have zero neighbourhood interaction potential; and therefore, they settle down to these low energy states. On the other hand, the exogenous potential field $B_\xi(t)$ defines the time-varying potential at unexplored sites (i.e. $\gamma_\xi(t) = U$) and is given as

$$B_\xi(t) = B^\star_\xi - C_{\xi,\mu}(t), \tag{9}$$

where $B_\xi^\star$ represents the constant potential field that is constructed to navigate the autonomous vehicle with no *in situ* adaptation. The relative cost potential function $C_{\xi,\mu}(t)$ defines the total decrease in potential at a grid cell ξ due to travel and turn costs that are incurred to reach the grid cell ξ from a current position $c(t) \in \mathcal{P}_\mu$ with the grid cell coordinate μ at time $t \in \mathcal{T}$. The cost function is constructed as

$$C_{\xi,\mu}(t) \triangleq \chi_{\text{tr}} T_{\text{tr}} + \chi_{\text{tu}} T_{\text{tu}}, \tag{10}$$

where T_{tr} is the cost of traveling across a single grid cell; T_{tu} is the cost of turning; and χ_{tr} and χ_{tu} define the total number of grid cells and the total number of turns, respectively, that are required to reach ξ from $\mu(t)$ along the shortest path. Equation (9) implies that the potential of a lattice site depends on the position and orientation of the autonomous vehicle. For example, a far away site on the lattice with respect to the current position of the autonomous vehicle will have less potential as compared to a nearby site, because of high travelling and turning costs, unless the far away site has a neighbourhood target.

Therefore, Equation (4) describes the total energy potential at a lattice site ξ, which is the sum of: (1) neighbourhood interaction potential due to nearby target locations and (2) a time-varying field that depends on an externally applied potential and the travelling and turning costs. The autonomous vehicle computes the values of the energy potentials $E_\xi^{\mathcal{L}}(t)$ for all $\xi \in \mathcal{N}_{\kappa_0}(\mu)$, and sets the centroid of the grid cell $\xi^*(t)$ that has the highest potential as the goal for local navigation,

$$\xi^*(t) = \underset{\xi \in \mathcal{N}_{\kappa_0}(\mu)}{\text{argmax}}\, E_\xi^{\mathcal{L}}(t). \tag{11}$$

The local navigation algorithm is summarised in Algorithm 1 using a pseudo code. The Bug2 algorithm (Ng & Bränl, 2007) is used as a subroutine for obstacle avoidance. Many other obstacle avoidance algorithms are available, such as Bug 1 and TangentBug. The Bug2 algorithm is chosen over its counterparts for its simplicity in implementation and execution.

4.2 *Probability vectors for global navigation*

Global navigation is usually operated over large search space that involves a large number of the finest level grid cells. Calculation of the energy for all the grid cells requires high computation power, especially when the search space is large. This may undermine the real-time implementation capability of the algorithm and its further application in the multi-agent cooperation. To resolve this problem, a lightweight probability vector is used to store the environment information of each cell at the coarse levels. This vector mirrors the spins associated in an area of the environment in a probabilistic manner.

Algorithm 2 Global Navigation Algorithm.

Input: current position $c(t) \in \mathcal{P}_\mu$, grid map $\mathcal{M}$, level ℓ
Output: updated grid map $\mathcal{M}$

```
generate grid map M^ℓ of level ℓ from grid map M
compute probability vectors p_i^ℓ for all cells at level ℓ
set the centroid of cell i* = argmax_i p_i^ℓ as the navigation goal
if front distance to obstacle ≤ threshold d then
    use Bug2 algorithm for obstacle avoidance
else
    move toward the navigation goal
    get current coordinate c(t) and sensor readings {s_i}
    update grid map M using c(t) and {s_i}
end if
```

Each coarse-cell partition is assigned a probability vector that records the spins of the fine cells located in it,

$$\begin{aligned} p_i^\ell &= [|\gamma_\xi(t) = \sigma_1|, |\gamma_\xi(t) = \sigma_2|, |\gamma_\xi(t) \\ &= \sigma_3|, |\gamma_\xi(t) = \sigma_4|]^T, \end{aligned} \tag{12}$$

where $|\gamma_\xi(t) = \sigma_j|$ with $j = 1, 2, 3, 4$ signifies the number of fine cells in that region with spin σ_j, and p_i^ℓ refers to the ith cell of the ℓth level. Evidently, the sum of the four elements of the vector is the number of cells in the region. Then, the probability of finding a cell with given spin w in region i in level l is given by

$$p_i^\ell(\gamma_\xi(t) = w) = \frac{|\gamma_\xi(t) = w|}{\sum_{j=1}^4 |\gamma_\xi(t) = \sigma_j|}. \tag{13}$$

This probability vector is very lightweight and extremely easy to store. In global navigation, the probability vectors of all the grid cells at the current coarse level are calculated, and the one with highest probability of unexplored cells is set as the goal. The details of global navigation algorithm are summarised in Algorithm 2, where the Bug2 algorithm (Ng & Bränl, 2007) is also used as a subroutine for obstacle avoidance.

5. Validation on a simulation test bed

This section presents validation of the multi-resolution algorithm for oil spill cleaning on a simulation test bed. The test bed is built upon the player/stage platform that is a high-fidelity open source robotic simulator. The multi-resolution algorithm, together with the 2D-random-walk oil weathering model, is implemented on the simulator. Several performance metrics are used to comparatively evaluate the performance of the proposed algorithm and the benchmark algorithm in oil spill cleaning. The simulation results of both algorithms are given at the end of the section.

5.1 *Player/stage simulator*

The player/stage project is one of the most widely used robotics software packages, which consists of libraries that provide access to communication and interface functionality. The robot server Player provides an architecture where many modules can be independently written and connected through a custom middleware relying on transmission control protocol communication. Stage is a lightweight, highly configurable robot simulator that supports large populations. All sensor and actuator models are available through Player's standard interfaces (Gerkey, Vaughan, & Howard, 2003).

A Pioneer 2AT robot is modelled in the simulator. The Pioneer robot is equipped with sonar sensors for navigation. The range of the sonar sensors is 5 m, and in total 16 sonar sensors are installed around the Pioneer robot. The robot has access to its position information in real time. The robot is also equipped with oil detection sensor that is implicitly modelled. It is assumed the oil detection sensor has limited range and is only able to detect oil spill that is within the same grid cell occupied by the robot. Kinematic constraints were modelled in the Pioneer robot, such as minimum turn radius, top speed, and maximum acceleration. Although we use Pioneer robot for validation, the navigation algorithm presented in the paper is meant to be generic and not platform dependent.

The oil spill is modelled as a 2D random walk process. The number of oil particles and their initial locations are specified in the program. The weathering processing of the oil spill is modelled by using a large number of oil particles at every step in the simulation. The navigation algorithm and the oil spill simulation are implemented in separate subroutines in Player such that the autonomous vehicle has no access to the distribution of the oil spill until it enters the grid cell that is occupied by oil particles. A vector is used to keep track of the status of every oil particle. At the beginning of the simulation, the oil particles are at their initial locations and all of them being in the 'active' status. The location of each oil particle is updated every step until the autonomous vehicle enters the same grid cell. The oil particles that are in the same grid cell with the autonomous vehicle are labelled to be 'inactive'. The 'inactive' oil particles are considered to have been removed from the water and thus their locations are no longer updated in the following simulation. This process continues until all oil particles are in the status 'inactive', i.e. the oil spill clean-up task is completed.

In the oil spill simulation, $N = 5000$ oil particles, forcing vector $\mathbf{A} = [2 \times 10^{-5}, 5 \times 10^{-5}]^T$, scaling matrix $\mathbf{B} = [2 \times 10^{-2}, 0;\ 0, 5 \times 10^{-2}]^T$, and vector of turbulent coefficients $\mathbf{K} = [0.5, 0.5]^T$ are used. The vector $\mathbf{Z}$ of two independent random numbers is obtained from a uniform distribution with range $[-1, 1]$. Each time step approximately corresponds to $\Delta t = 0.3$ seconds in real time. Figure 1 shows the results of oil spill simulation without any cleaning action. Figure 1(a) shows the distribution and concentration of oil spill at the beginning of simulation ($n = 1$) and Figure 1(b) shows the same after $n = 10{,}000$ steps of simulation.

A 30 $\times$ 30 m simulated harbour map has been used to test the performance of the proposed algorithms. The size of the grid cell at the finest level is 1 $\times$ 1 m, which is slightly larger than the size of the Pioneer 2AT robot. The selection of the starting point of the operation depends on the direction of tidal current and wind. In this simulation exercise, the starting point is the bottom left corner because the wind and tidal current make the oil spill drifts toward the harbour.

Since the initial location of the oil spill is unknown to the autonomous vehicle and both the size and the location of the oil spill change with time due to spreading and drift, the autonomous vehicle needs to cover the entire search area to assure complete clean-up of the oil particles. The typical back and forth motion is optimal for searching an area in terms of minimum number of turns when no adaptation to target and obstacle avoidance is needed. Therefore, for area coverage planning, the exogenous potential field $B^{\star}_{\xi}$ in Equation (9) is designed for back and forth motion such that the potential field has a decreasing magnitude from column to column, starting from a maximum value of magnitude 10,000 at the start point, while having equipotential sites on each column. The other parameters in Equations (4)–(10) have been selected to be $\kappa_0 = 3, \kappa_1 = 2, \phi_T = 0,\ \phi_U = 0,\ \phi_O = -40,000, \psi_T = 2000, \psi_E = 0, T_{\text{tr}} = 600, T_{\text{tu}} = 1000,$ and $\alpha = 0.8$ in the simulation exercises; however, the specific values of these parameters do not have very significant effects on the algorithm performance since the behaviour of the autonomous vehicle depends on the ratio of the parameters (ϕ, ψ, and T) instead of the magnitude.

To comparatively evaluate the performance of the proposed multi-resolution navigation algorithm, a benchmark algorithm (Jin et al., 2012) is also tested in the same scenario. In the benchmark algorithm, the autonomous vehicle implements back and forth motion and avoids obstacles as needed. When no unexplored grid cells are in its local neighbourhood, the vehicle moves toward the direction of the most unexplored grid cells. In essence, the benchmark algorithm is a simplified version of the multi-resolution navigation algorithm without the adaptation term $\sum_{<\xi,\nu>_\kappa} J_{\xi\nu}\Psi\left(\gamma_\xi(t), \gamma_\nu(t)\right)$ as shown in Equation (4). Without the adaptation term, the benchmark algorithm is not affected by the detection of the oil spill in the current grid cell and thus does not deviate from the back and forth motion to search the neighbourhood of current grid cell for oil spills.

5.2 *Performance metrics*

Four different performance metrics are used to compare the effectiveness of the proposed algorithm with that of the

benchmark algorithm in oil spill cleaning:

- T_{total}: total time to cover the entire search area.
- T_{clean}: time to clean up all the oil spills.
- AUC: area under the curve in the oil spill cleaning profile.
- EIM: environmental impact measure.

The most intuitive performance metric is the completion time. Since the location of the oil spill is unknown and varying with time, the autonomous vehicle needs to cover the entire search area to guarantee removal of all the oil spills. Therefore, two performance metrics regarding the completion time are used in this paper, namely, the total time in covering the entire search area T_{total} and the time taken to clean up all oil spills in the search area T_{clean}. It is obvious that, in most cases, $T_{\text{total}} > T_{\text{clean}}$. The performance metric T_{clean} is more important than the metric T_{total}, as it reflects the ability of the system to remove the environmental risk. Thus, in the case where one algorithm has a lower T_{total}, but the other has a lower T_{clean}, the latter is considered more effective.

To keep track of the history of oil spill cleaning, a profile is generated to record the remaining oil spill at each time step. The profiles of the proposed algorithm and the benchmark algorithm are visualised in Figure 8. The metric *Area Under the Curve* (AUC) has been used to quantitatively compare these two profiles. The formal definition of AUC is given as follows:

Definition 5.1 (AUC): Let $\mathcal{H}$ be the profile that records the remaining oil spill at each time step, then the metric AUC is defined as

$$\text{AUC} = \sum_{n=1}^{T} \mathcal{H}(n), \tag{14}$$

where T is the time taken to complete the task. This metric also emphasises the efficiency of cleaning the oil spill. The algorithm with smaller AUC is considered to be more effective.

Oil spill has significant ecological impacts to the environment. The longer the oil spill remains in the water, the more damage it will incur. The *environmental impact measure* (EIM) is defined to quantify the impact.

Definition 5.2 (EIM): Let $\mathcal{H}$ be the profile that records the remaining oil spill at each time step, then the metric EIM is defined as

$$\text{EIM} = \sum_{n=1}^{T} n\mathcal{H}(n) \tag{15}$$

The performance metrics T_{clean}, AUC and EIM are related to each other. The clean-up time T_{clean} shows how fast the algorithm is able to finish the cleaning of all oil spills, the AUC takes the remaining oil spill at each time step into consideration, and the EIM emphasises the impact to the environment due to the cumulative effect of the oil spill.

5.3 *Simulation results*

This simulation exercise aims to validate the proposed algorithms for a real-life oil spill cleaning scenario, and compare their performance with the benchmark algorithm. A map with the layout of a typical harbour is designed. The harbour consists of several ports and has one entrance that connects to the open sea, as shown in the top row of Figure 1. The initial oil spill is located at the entrance, probably due to a maritime accident outside the harbour or drifting of the oil spill from the site of an offshore platform accident. As shown in Figure 1, the oil spill will hit the port in a certain amount of time due to drift and spreading if no proper response is taken.[1]

The multi-resolution algorithm is used to navigate an autonomous vehicle to clean up the oil spill. Four snapshots are taken and shown in Figure 6(a)–(d). The top row shows the layout of the harbour, the oil spill, and the trajectory of the autonomous vehicle, while the bottom row shows the environment map in the vehicle's onboard memory. Figure 6(a) shows the vehicle follows the offline plan and implements back and forth motion to explore the search area. Figure 6(b) shows the vehicle deviates from the offline plan once it detects the oil spill and explores the neighbourhood of the grid cells where the oil spill is detected. Figure 6(c) shows the moment when the vehicle successfully cleans-up all the oil spills in the search area, and Figure 6(d) shows the vehicle explores the remaining search area using back and forth motion and avoids obstacles.

In the environment maps, the yellow box around the search area in Figure 6(a) is the buffer specified at the beginning of the task to prevent the vehicle from leaving the search area. As the exploration goes on, the vehicle detects the jetty and the port, which are marked in dark green in the environment map. Upon detection of the jetty and the port, the buffer is automatically labelled around them in order to force the vehicle to keep a safe distance and avoid collision. The grid cells where oil spill is discovered are labelled in a different colour, and the vehicle spends more time to search the neighbouring area of the cells. As seen in Figure 6(d), the complete perimeter of the harbour is mapped in dark green.

The benchmark algorithm is also implemented in the same map to compare with the proposed algorithm, and the results are shown in Figure 7. The only difference between Figures 6 and 7 is that the autonomous vehicle does not adapt to the neighbourhood of the grid cells with oil spill detected. Instead, it follows the offline plan and implements the back and forth motion until it covers the entire search

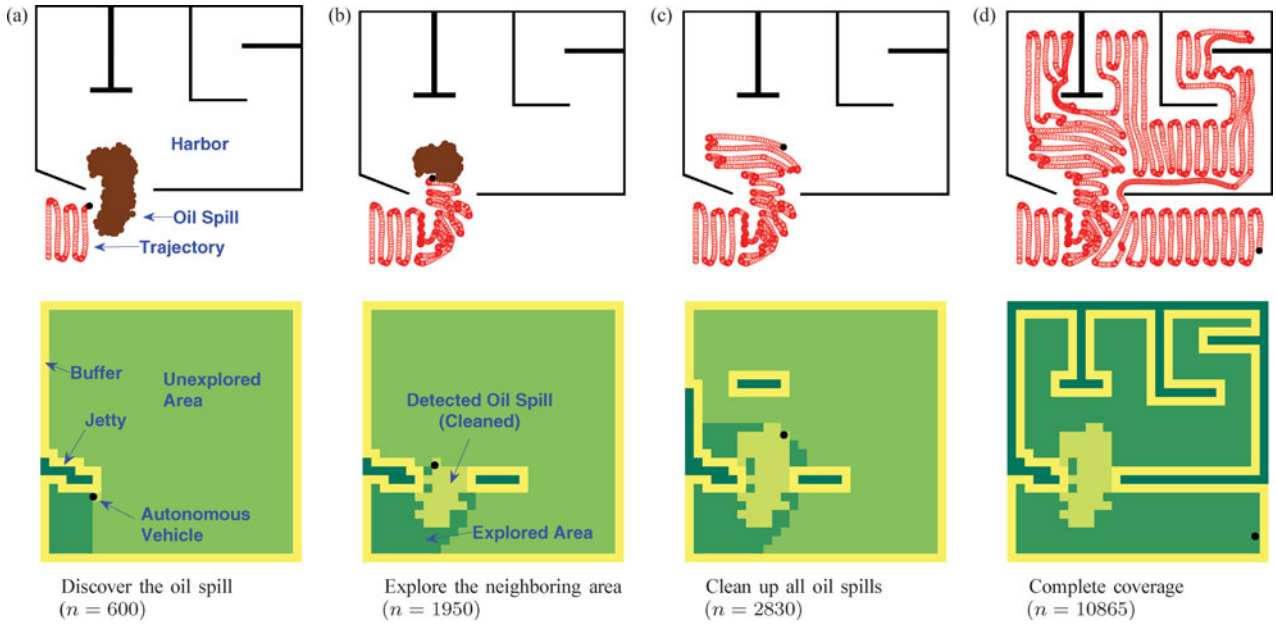

Figure 6. Example of oil spill cleaning in a harbour using the proposed multi-resolution navigation algorithm. (a) Discover the oil spill ($n = 600$). (b) Clean-up all oil spills ($n = 2830$). (c) Complete coverage ($n = 10865$). (d) Complete coverage ($n = 10865$).

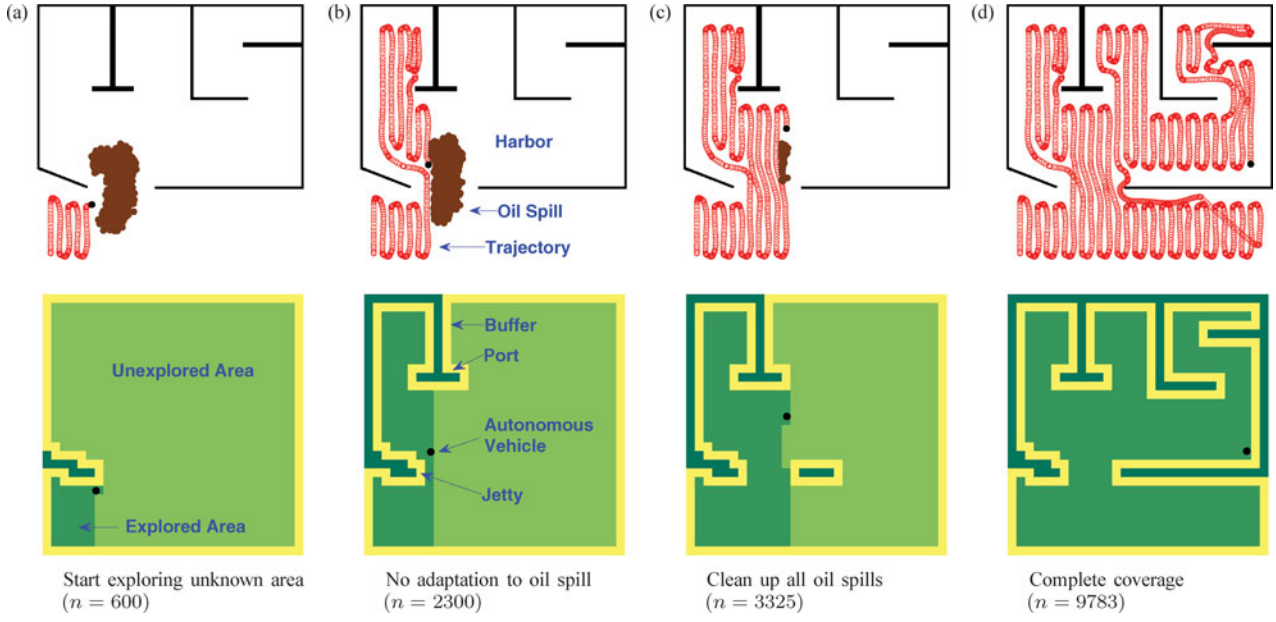

Figure 7. Example of oil spill cleaning in a harbour using the benchmark algorithm (Jin et al., 2012) that has no adaptation to the oil spill. (a) Start exploring unknown area ($n = 600$). (b) No adaptation to oil spill ($n = 2300$). (c) Clean-up all oil spills ($n = 3325$). (d) Complete coverage ($n = 9783$).

area. As a result, the trajectory recorded in Figure 7 is neater than the one in Figure 6.

The profiles of oil spill cleaning using the proposed and benchmark methods are shown in Figure 8. The y axis in Figure 8 indicates the normalised number of the remaining oil particles, and the x axis is the time steps in the simulation. It is seen in Figure 8 that the proposed method takes much less time in cleaning up all the oil spills, although it takes slightly more time to finish scanning the entire search area. This is understandable since in the proposed method the autonomous vehicle deviates from the offline plan and exploits the neighbouring areas when it detects oil spills.

The proposed method and the benchmark method are comparatively evaluated using the performance metrics

Table 1. Comparative evaluation of the proposed and benchmark methods in oil spill cleaning in a harbour environment.

Method	T_{total}	T_{clean}	AUC	EIM
Benchmark	9783	3387	2578	3.51×10^6
Proposed	10865	2826	1618	1.45×10^6
Improvement	−11.1%	16.6%	37.2%	58.7%

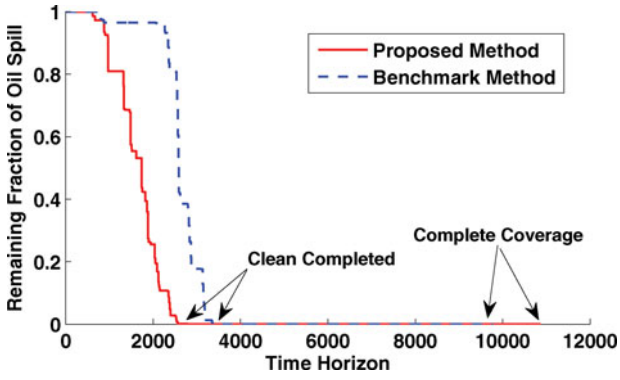

Figure 8. Oil spill cleaning profiles of the proposed and benchmark methods.

defined in Section 5.2, and the results are shown in Table 1. Although it takes 11.1% more time to finish scanning the entire search area, the proposed method is much more efficient in oil spill cleaning and takes 16.6% less time to clean up all the oil spills. As indicated by EIM, the proposed method significantly reduces the impact of the oil spill to the environment by dynamically adapting to the oil spill and replanning online.

6. Summary, conclusions and future work

This paper presents a multi-resolution navigation algorithm for oil spill cleaning in dynamic and uncertain environments using autonomous vehicles as a single agent. The concepts of local and global navigation are integrated in the algorithm for adaptive decision making according to the available spatio-temporal information. The local navigation provides a reduced computational complexity in local decision making, while the global navigation is organised in a hierarchical manner to prevent the robot from being stuck into a local minimum. Dynamic adaptation significantly improves the cleaning efficiency and reduces the impact of the oil spill to the environments. The proposed algorithm, while motivated by the oil spill application, can also be applied to other problems that require coverage guarantees with mapping of a dynamic field as a goal, such as plume detection.

The proposed algorithm has been validated in a harbour example. With the multi-resolution navigation algorithm, the autonomous vehicle manages to explore the complex and unknown environment, and cleans up all the oil spills in a timely manner. Compared to the benchmark algorithm that uses back and forth motion and obstacle avoidance algorithm, the proposed multi-resolution algorithm is more efficient in oil spill cleaning and significantly reduces the impact of the oil spill to the environment.

While there are many research issues that need to resolved before exploring commercial applications of the proposed algorithms, the following topics are under active research.

- *Inference of oil spill distribution to facilitate adaptation*: The autonomous vehicle under consideration in this paper is able to detect an oil spill only if the spill is present within a grid cell. If additional sensors are available to measure the information on oil concentration, then the oil spill distribution could be inferred to facilitate adaptation. In that case, the autonomous vehicle would be able to move in the steepest ascendent direction to clean up the high-concentration area first.
- *Experimental validation of the algorithm on hardware platform(s)*: The proposed algorithm has not been experimentally validated primarily due to the difficulties in creating oil spills in a laboratory environment and also due to high costs of constructing a prototype of the oil spill cleaning robot. The current plan is to implement the proposed algorithm on existing land robots in a laboratory environment and perform co-simulation on a computer that is dedicated to simulate the oil spill process.
- *Extension of the algorithm for multi-agent cooperation*: Multi-agent cooperation is critical for reducing the time required to clean up oil spills. The proposed algorithm could be extended to investigate the multi-agent case by dividing the entire search space into several regions, where an agent is assigned to each region. In this way, the cleaning efficiency could be further improved if communications among the agents are available such that the idle agents, who have already finished their assigned tasks, are allowed to assist their neighbours to clean up the remaining oil spills.

Acknowledgements

Any opinions, findings and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

Funding

This work has been supported in part by the US Army Research Laboratory (ARL) and the US Army Research Office (ARO) [grant number W911NF-07-1-0376]; US Office of Naval Research (ONR) [grant number N00014-09-1-0688].

Note

1. A video of this simulation exercise is available at http://goo.gl/R0SXw for downloading or viewing online.

References

Acar, E.U., & Choset, H. (2002). Sensor-based coverage of unknown environments: Incremental construction of Morse decompositions. *International Journal of Robotics Research, 21*(4), 345–366.

Arega, F., & Sanders, B. (2004). Dispersion model for tidal wetlands. *Journal of Hydraulic Engineering, 130*(8), 739–754.

Brebbia, C. (2001). *Oil spill modeling and processes*. Southampton: WIT Press.

Cai, C., & Ferrai, S. (2009). Information-driven sensor path planing by approximate cell decomposition. *IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics, 39*(3), 672–689.

Carmo, J., Pinho, J., & Vieira, J. (2005). *Oil spills in coastal zones: Environmental impacts and practical mitigating solutions*. Lisboa: International Maritime Association of the Mediterranean.

Choset, H. (2000). Coverage of known spaces: The boustrophedon cellular decomposition. *Autonomous Robots, 9*, 247–253.

Choset, H. (2001). Coverage for robotics: A survey of recent results. *Annals of Mathematics and Artificial Intelligence, 31*, 113–126.

Cowlagi, R., & Tsiotras, P. (2012). Multiresolution motion planning for autonomous agents via wavelet-based cell decompositions. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), 42*(5), 1455–1469.

Dimou, K., & Adams, E. (1993). A random-walk, particle tracking model for well-mixed estuaries and coastal waters. *Estuarine, Coastal and Shelf Science, 37*(1), 99–110.

Engelhardt, F. (1999). Remote sensing for oil spill detection and response. *Pure and Applied Chemistry, 71*(1), 103–111.

Fingas, M. (1995). Oil spills and their cleanup. *Chemistry & Industry, 24*, 1005–1008.

Garcia, E., & de Santos, P.G. (2004). Mobile-robot navigation with complete coverage of unstructured environments. *Robotics and Autonomous Systems, 46*, 195–204.

Ge, S.S., & Cui, Y.J. (2000). New potential functions for mobile robot path planning. *IEEE Transactions on Robotics and Automation, 16*(5), 615–620.

Gerkey, B., Vaughan, R., & Howard, A. (2003). The Player/Stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the International Conference on Advanced Robotics (ICAR 2003)* (pp. 317–323). Coimbra, Portugal.

Grisolía-Santos, D., & Spaudlding, M. (2000). The influence of oil particle size distribution as an initial condition in oil spill random walk models. In G. Rodríguez & C. Brebbia (Eds.), *Oil and hydrocarbon spills, modeling, analysis and control II* (pp. 19–28). Southampton: WIT Press.

Gupta, S., Ray, A., & Phoha, S. (2009). Generalized Ising model for dynamic adaptation in autonomous systems. *European Physics Letters, 87*, 10009. doi:10.1209/0295-5075/87/10009

Hert, S., Tiwari, S., & Lumelsky, V. (1996). A terrain-covering algorithm for an AUV. *Journal of Autonomous Robots, 3*, 91–119.

Jin, X., Gupta, S., Luff, J., & Ray, A. (2012). Multi-resolution navigation of mobile robots with complete coverage of unknown and complex environments. In *Proceedings of 2012 American Control Conference* (pp. 4867–4872). Montréal, Canada.

Jin, X., Luff, J., Gupta, S., & Ray, A. (2010). Autonomous vehicle navigation with dynamic adaptation and complete coverage of unknown environments. In *Proceedings of the ASME 2010 Dynamic Systems and Control Conference* (pp. 241–248). Cambridge, MA.

Kakalis, N.M., & Ventikos, Y. (2008). Robotic swarm concept for efficient oil spill confrontation. *Journal of Hazardous Materials, 154*(1–3), 880–887.

Kim, J., & Khosla, P.K. (1992). Real-time obstacle avoidance using harmonic potential functions. *IEEE Transactions on Robotics Automation, 8*(3), 338–349.

Koren, Y., & Borenstein, J. (1991). Potential field methods and their inherent limitations for mobile robot navigation. In *Proceedings of the IEEE Conference on Robotics and Automation* (pp. 1398–1404). Sacramento, CA.

Korotenko, K., Mamedov, R., Kontar, A., & Korotenko, L. (2004). Particle tracking method in the approach for prediction of oil slick transport in the sea: Modelling oil pollution resulting from river input. *Journal of Marine Systems, 48*(1–4), 159–170.

Lee, J.Y., & Choset, H. (2005). Sensor-based exploration for convex bodies: A new roadmap for a convex-shaped robot. *IEEE Transactions on Robotics, 21*(2), 240–247.

Lonin, S.A. (1999). Lagrangian model for oil spill diffusion at sea. *Spill Science & Technology Bulletin, 5*(5–6), 331–336.

Lumelsky, V., Mukhopadhyay, S., & Sun, K. (1990). Dynamic path planning in sensor-based terrain acquisition. *IEEE Transactions on Robotics and Automation, 6*(4), 462–472.

Mabrouk, M.H., & Mclnnes, C.R. (2008). Solving the potential field local minimum problem using interanal agent states. *Robotics and Autonomous System, 56*, 1050–1060.

MIT Senseable City Lab. (2010). *Seaswarm*. Retrieved from http://senseable.mit.edu/seaswarm/

Mullin, J.V., & Champ, M.A. (2003). Introduction/overview to in situ burning of oil spills. *Spill Science & Technology Bulletin, 8*(4), 323–330.

Ng, J., & Bränl, T. (2007). Performance comparison of bug navigation algorithms. *Journal of Intelligent & Robotic Systems, 50*, 73–84.

Oh, J.S., Choi, Y.H., Park, J.B., & Zheng, Y. (2004). Complete coverage navigation of cleaning robots using triangular-cell-based map. *IEEE Transactions on Industrial Electronics, 51*(3), 718–726.

Pathria, R.K. (1996). *Statistical mechanics*. Boston, MA: Butterworth-Heinemann.

Pereda, F., de Marina, H., Giron-Sierra, J., & Jimenez, J. (2011). Towards automatic oil spill confinement with autonomous marine surface vehicles. In *OCEANS, 2011 IEEE* (pp. 1–6). Santander, Spain.

Pinho, J., Antunes do Carmo, J., & Vieira, J. (2002). Numerical modelling of oil spills in coastal zones – a case study. In C. Brebbia (Ed.), *Oil and hydrocarbon spills III: Modelling, analysis and control* (pp. 35–45). Southampton: WIT Press.

Protei: Open Source Sailing Drone. (2010). Retrieved from http://protei.org/

Valavanis, K.P., Hebert, R.K.T., & Tsourveloudis, N. (2000). Mobile robot navigation in 2-D dynamic environments using an electrostatic potential field. *IEEE Transactions On Systems, Man, and Cybernetics–Part A: Systems and Humans, 30*(2), 187–196.

Ventikos, N., Vergetis, E., Psaraftis, H., & Triantafyllou, G. (2004). A high-level synthesis of oil spill response equipment and countermeasures. *Journal of Hazardous Materials, 107*(1–2), 51–58.

Yang, S.X., & Luo, C. (2004). A neural network approach to complete coverage path planning. *IEEE Transactions on Systems, Man and Cybernetics – Part B: Cybernetics, 34*(1), 718–725.

Zhang, G., Fricke, G., & Garg, D. (2011). Spill detection and perimeter surveillance via distributed swarming agents. *IEEE/ASME Transactions on Mechatronics, 18*(1), 121–129.
Zhong, Z., & You, F. (2011). Oil spill response planning with consideration of physicochemical evolution of the oil slick: A multiobjective optimization approach. *Computers & Chemical Engineering, 35*(8), 1614–1630.

Appendix 1. Background of oil spill

This appendix briefly reviews the background of oil spill. The relevant information regarding the oil weathering process and the oil spill cleaning methods are discussed.

1.1 *Oil weathering processes*

The spilled oil is normally a mixture of hydrocarbon compounds whose chemical and physical properties vary among oil types. When an oil spill occurs, the nature of the oil undergoes a series of changes in chemical and physical properties over time, that in combination, is termed 'weathering'. The oil weathering processes include short-term processes such as spreading, drift, evaporation, emulsification, dispersion, and dissolution, and long-term processes such as photo-oxidation, biodegradation, and sedimentation (Brebbia, 2001; Zhong & You, 2011).

In the event of a massive oil spill on water, the oil begins to spread by gravity, wind, and current, with the process resisted by inertia, viscosity, and surface tension, until the slick reaches a thickness of 0.1 mm or less. The process, known as *spreading*, can affect other weathering processes such as evaporation, dispersion, and emulsification. The environmental impact of oil spills largely depends on the spreading area. The clean-up operations also require information on the spill size. Because of the influence of the winds and wind-induced surface currents, the oil slick may move downward with respect to the wind direction. This movement, called *drift*, results in a displacement of the centre of the oil slick and contributes to the non-symmetrical spreading (Zhong & You, 2011). This paper considers both spreading and drift in the oil weathering process. Other weathering processes include evaporation, natural dispersion, emulsification, dissolution, photo-oxidation, sedimentation, and biodegradation.

In the oil weathering process, a variety of complex physical, chemical and biological phenomena take place simultaneously. The weathering process depends on the initial oil properties, the spilled amount, hydrodynamics, and weather conditions (Zhong & You, 2011). It is estimated that over 50 oil weathering models have been developed. The transport of oil can be modelled either by solving the advection–diffusion equation on the node of a grid (Eulerian models) (Carmo, Pinho, & Vieira, 2005; Pinho, Antunes do Carmo, & Vieira, 2002), or by tracking particles, which represent individual portions of the spill (Lagrangian models) Lonin (1999). Random walk models (Arega & Sanders, 2004; Dimou & Adams, 1993; Grisolía-Santos & Spaudlding, 2000; Korotenko, Mamedov, Kontar, & Korotenko, 2004), which solve the nonlinear Langevin equation, belong to the latter category. For the 2D random walk models, initial conditions including the number of particles and time interval need to be specified. For random-walk spill models involving the vertical direction, the oil particle size distribution must also be specified, since oil buoyancy influences oil spill movement and spreading (Grisolía-Santos & Spaudlding, 2000). In this paper, we use the 2D random-walk particle tracking technique (Arega & Sanders, 2004; Dimou & Adams, 1993) to follow the motion of individual particles (oil droplets), the total amount of which constitutes the oil spill.

1.2 *Oil spill clean-up methods*

When a massive oil spill occurs, quick and effective response is critical in order to minimise the economic impact and the damage to both the ecology and the quality of human life. Five methods are commonly used to contain and clean up a spill: mechanical methods, chemical methods, use of absorbents, oil burning and bioremediation (Ventikos, Vergetis, Psaraftis, & Triantafyllou, 2004).

The current state of the art in spill countermeasures varies from mechanical and chemical methods, to the usage of absorbents, oil burning and bioremediation (Ventikos et al., 2004). Mechanical methods, referring to the use of booms and skimmers, are the most widely used combat means. Booms are used to confine the oil to a specific area, hence controlling its spreading, and/or stop the oil from entering a given area, while skimmers are used to recover the oil from the water surface (Fingas, 1995). Other clean-up methods may be effective under certain circumstances, but they also suffer from many limitations and are not as cost effective as the mechanical methods (Kakalis & Ventikos, 2008).

The chemical methods are more efficient than mechanical method when properly applied, but the dispersed oil droplets in high concentrations could have an acute lethal toxicity for many species. Despite burning of the spilled oil is an efficient, versatile and lower cost method, there could be possibility of sinking extremely viscous and dense residues. The idea of bioremediation is attractive; however, its practical use is restricted.

Mechanical clean-up refers to the use of booms and skimmers, and are the most widely used combatting means. Mechanical recovery operations for large spills are in general considered to be expensive, complex and labour-intensive, with their recovery efficiency not to usually exceed 20% (Mullin & Champ, 2003). The chemical methods are based on the transformation of the physicochemical properties of the oil, and the most common method is the use of dispersants. Although dispersants, when properly applied, are more efficient than skimmers, for large spills can also be expensive, complex and labour-intensive. *In situ* burning corresponds to controlled ignition and burning of the oil at or near the spill site, on the surface of the water or in a marsh (Mullin & Champ, 2003). Bioremediation refers to the addition of microorganisms that are able to degrade hydrocarbons, or the addition of fertilizers to the hydrocarbon-degrading bacteria and fungi that naturally exist in marine environments. Although the idea of bioremediation is attractive, its practical use is restricted.

Current technology to skim oil spill harms the health of cleaners, cannot operate at night safely, is not in operation far from shore because it is too costly, and cannot operate in rough weather.

Appendix 2. Important features of the algorithm

This appendix outlines important features of the multi-resolution algorithm, which includes complete coverage of the unknown environment, clean-up of all oil spills, no local minima problem, and low computational complexity.

2.1 *Complete coverage*

The multi-resolution algorithm provides complete coverage of the unknown search space, as explained below.

Let there be an unexplored grid cell ξ that is far away from the autonomous vehicle. Then the autonomous vehicle may switch to global navigation because no unexplored grid cells are within its κ_0-neighbourhood. The coarse-level ℓ increases until a coarse-level cell ς with $\xi \in \varsigma$ and $p_\varsigma^\ell > 0$ is found, and the vehicle

navigates toward the centroid of the cell ς. As the vehicle approaches the fine-level grid cell ξ, the coarse-level ℓ gradually decreases because a lower level global view is large enough to cover the unexplored cell ξ. Eventually, the grid cell ξ falls within the κ_0-neighbourhood of the vehicle, and the vehicle finishes the complete coverage task by moving into the grid cell ξ.

2.2 *Clean-up capability*

The multi-resolution algorithm is capable of cleaning oil spills whose locations may not be known, as explained below.

The autonomous vehicle carries out back and forth motion until it detects oil spill. Upon detection of the oil spill, the autonomous vehicle explores the κ_0-neighbourhood of the current grid cell where oil is discovered. Due to the fact that oil distribution is continuous, the autonomous vehicle discovers and cleans-up the entire oil spill before exploring the remaining areas. The starting point of the operation is chosen at the windward side such that remaining oil spill does not enter the area that has been cleaned. Following Appendix 2.1, the autonomous vehicle eventually finishes covering the entire search area and cleans-up all the oil spills.

2.3 *Local minima problem*

The multi-resolution algorithm does not suffer from the local minima problem, as explained below.

Local minima problem occurs when there is no unexplored grid cell left in the local navigation area. In this case, the autonomous vehicle switches from local navigation to global navigation and increases the coarse level until it finds unexplored grid cells. The vehicle then moves toward the centroid of the coarse-level cell and resolves the local minima problem.

2.4 *Computational complexity*

The multi-resolution algorithm has very low computational complexity that enables the real-time implementation of the algorithm, as explained below.

The computational complexities of the local and global navigation are evaluated separately. In the local navigation, the autonomous vehicle searches for the grid cell with highest energy within a square area of length $2\kappa_0 + 1$, which has $(2\kappa_0 + 1)^2 - 1$ grid cells excluding the grid cell occupied by the autonomous vehicle. Therefore, the local navigation algorithm has a computational complexity of $\mathcal{O}({\kappa_0}^2)$. In the global navigation, as illustrated in Figure 3, the autonomous vehicle only needs to search a 2×2 area, no matter which coarse level it is at. Overall, the proposed algorithm has a computational complexity of $\mathcal{O}({\kappa_0}^2)$. Usually $\kappa_0 \simeq \frac{1}{10}L$ suffices full coverage of a local neighbourhood, where L is the length of the search area; so the proposed algorithm has a low computational complexity.