Navigation for Inspection

Alberto Ortiz, Javier Antich and Francisco Bonnin-Pascual

Synonyms

Localization and mapping (for inspection); Motion and path planning (for inspection); Survey of an Area of Interest (AOI); Complete coverage of an AOI

Definition

Navigation for inspection refers to both the hardware and software required to make a robotic platform undertake the different tasks involved in an inspection mission. This typically requires from the robot a number of capabilities at the locomotion, sensory and ”intelligence” levels which make it possible collecting inspection data of relevance, in accordance to the inspection procedures of application, while adequatelly covering the Area of Interest (AOI).

Alberto Ortiz
Dept. of Mathematics and Computer Science, University of the Balearic Islands, Cra. Valldemossa, km 7.5, 07122 - Palma de Mallorca, e-mail: alberto.ortiz@uib.es

Javier Antich
Dept. of Mathematics and Computer Science, University of the Balearic Islands, Cra. Valldemossa, km 7.5, 07122 - Palma de Mallorca, e-mail: javier.antich@uib.es

Francisco Bonnin-Pascual
Dept. of Mathematics and Computer Science, University of the Balearic Islands, Cra. Valldemossa, km 7.5, 07122 - Palma de Mallorca, e-mail: xisco.bonnin@uib.es

Overview

Installations and facilities such as industrial plants (e.g. from the power or petrochemical sectors), large-tonnage vessels, storage tanks, buildings, roads, bridges, tunnels, etc., to name but a few, deteriorate due to environmental factors or simply wear and tear. Therefore, the periodic maintenance of these facilities, including visual inspection of the relevant structures, is required to ensure a safe and proper use/operation. Typical maintenance programmes base their decisions on gathered visual evidences and on taking a number of measurements from specific areas of the infrastructure, in both cases following well-defined inspection procedures.

By way of example, we can refer to the particular case of large-tonnage vessels. The safety of these vessels is overseen by the so-called Classification Societies, who are continually seeking to improve inspection standards and reduce the risk of maritime accidents. Since structural failures are a major cause of accidents, and can usually be prevented through timely maintenance, this kind of vessels undergo annual inspections, with intensive Special and Docking Surveys every five years, which ensures that the hull structure and related piping are all in satisfactory condition and are fit for the intended use over the next five years. This is the reason why an important fraction of the inspection effort is spent on close-up visual surveys, where the involved personnel has to get within arms reach of the structural element under observation, as well as on taking thickness measurements at strategic points over the vessel hull.

By way of illustration of the enormity of the inspection task, one can notice that large-tonnage vessels (e.g. bulk carriers, containerships, oil tankers) can contain up to 600,000m^2 of steel, involving over 860m of web frames (primary stiffening members) and approximately 3.2km of longitudinal stiffeners (e.g. a VLCC, Very Large Crude Carrier). Furthermore, the inspection requires that many of the tanks of the vessel (cargo, fuel, ballast, etc.) are emptied, ventilated (because of the presence of flammable and/or toxic gases) and cleaned prior to the inspection, and that suitable access is arranged, typically using scaffolding or cherry-pickers to reach relevant heights (of around 15-20m in most cases). In the case of older ships, this preparation will also be useful to allow the repair crew to enter and access the element that has failed. However, in a younger well-run ship, the survey is likely to reveal no defects in the early years, the number of required repairs is small, and so the bulk of these arrangements are mostly required for the survey itself. Once you factor the vessels preparation in the use of yards facilities, cleaning, ventilation, and provision of access arrangements, the total cost of a single surveying can exceed $1M. In addition, the ship owner experience significant lost opportunity costs while the ship is inoperable (a detailed discussion can be found in [36]).

Other application scenarios for inspection, such as bridges, buildings, storage tanks, etc. share similar requirements as the aforementioned, although at their proper scale and following their corresponding inspection rules. These requirements and inspection rules explicit the kind of inspection data that is to be collected and the critical points from where to gather it. In many cases, it is required to video-document the area under inspection, complementing it with measurements of the state of the infrastructure at specific points using particular probes, e.g. ultrasound thickness measurement for the case of metallic structures, infrared thermography and ground-penetrating radar for the case of concrete structures, etc. All in all, the goal is to detect specific defects: to name but a few, corrosion, cracking, buckling, coating breakdown (steel structures), scaling, cracking, spalling, efflorescence, delamination, leakages (concrete structures), ...

The incorporation of suitable robotic platforms into the previous data collection tasks is to provide levels of assistance and/or automation that contribute to the success of the operation (improving the quality and accuracy of the data collected, as well as for intensifying data capture), reducing the execution time, lowering the costs involved (this is particularly true for the inspection of younger installations) and/or increasing the safety of the operation. Aside from these benefits, the automated approach is to permit focusing the corresponding officers close-up assessment on the areas that are most at risk. Furthermore, for the particular case of infrastructure inspection, the use of robots allows the owners to perform inspections more frequently, on their own initiative, and know well ahead of time any repair work that is necessary so they can schedule it at their best convenience, e.g. plan ship repairs at a competitively priced dock.

For those benefits to be finally available, the robots need to be fitted with the necessary capabilities at both the hardware and software levels. That is to say: (1) the platform will be required to adapt to the environment and the surfaces to inspect, what mainly refers to its locomotion abilities, partly implemented through the morphology and mechanical design of the robot, and partly by a suitable low-level control layer; (2) adequate mid- and high-control layers that, as part of the platform control architecture, are to fit the robot with a proper interface with the operator as well as with a number of autonomous, useful functionalities. Apart from specific requirements for a particular kind of application, inspection tasks usually require from the robotic solution the capability to self-localize and map inspection data, as well as be able to navigate safely through the area of interest ensuring a high level of coverage. These common abilities of inspection-oriented robots constitute the key research findings of this chapter.

Key Research Findings

Navigation is one of the fundamental problems to deal with when an application involving a mobile robotic system is addressed. As a rather complex capability, navigation comprises in turn functionalities such as, among others, self-localization, mapping, motion planning and path planning [45]. The designers of a purely tele-operated robot would rarely be concerned by these skills, but they will become relevant, alone or in combination, as soon as autonomy is incorporated into the system, either in full or partially. By way of example, semi-autonomous platforms adopting supervised/shared autonomy control paradigms allow an operator in the main control loop but provides assistance on platform navigation (and also self-preservation) to alleviate the pilotage stress and permit focusing on the inspection task at hand.

For the case of inspection applications, the locomotion capabilities can also become crucial for mission success if a specialized kind of mobility is dictated by the specific structures and surfaces to be inspected and the kind of inspection data to be collected, e.g. take thickness measurements of the surface of a storage tank or of a vessel's cargo hold (see Fig. 1(a) and (b) for a couple of examples), using a by-contact transducer. The particular kind of inspection to be performed can additionally state how and from where inspection data has to be collected for the inspection to be conclusive, i.e. in accordance to well-established rules, what can determine the kind of motion the platform has to perform (see Fig. 1(c)).

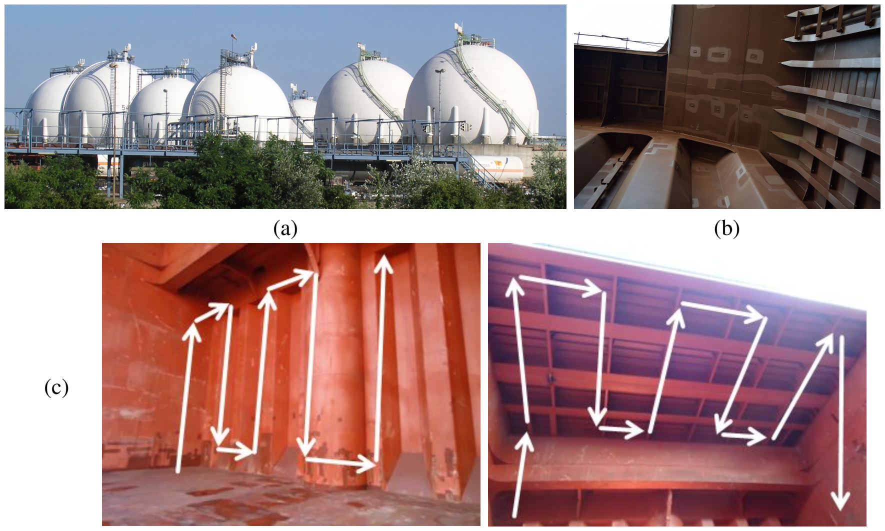

Fig. 1 Examples of structures and surfaces that an inspection application could involve: (a) storage tanks [source: https://de.wikipedia.org/wiki/Kugelgasbehälter, licensed according to https://creativecommons.org/licenses/by/2.0/de/deed.en], (b) view from bottom to top of a part of a bulk carrier's cargo hold, (c) examples of paths to be followed during an inspection.

What follows reviews briefly (and without pretending to be exhaustive) the aforementioned topics from the perspective of an inspection mission, irrespectively of the degree of autonomy intended to be attained. Additionally, the end of this section illustrates the most relevant topic regarding navigation for inspection, giving the details of Complete Coverage Path Planning (CCPP) algorithms.

Locomotion issues

One can find inspection robots for almost all kinds of inspection scenarios, ranging from land to submerged installations. While the latter requires vehicles able to operate underwater, which on most occasions lead to the use of standard ”sailing” platforms, either Remotely Operated Vehicles (ROV) or Autonomous Underwater Vehicles (AUV) [16, 39], land installations have been proposed to be inspected by means of vehicles with different kinds of locomotion: from purely wheeled vehicles able to displace over the floor, perhaps fitted with telescopic arms for reaching high points of interest [32], to generic aerial vehicles adapted to the inspection application through the integration of the necessary sensors [11, 25, 37, 42], going through robots specifically designed to move over or inside the structure to inspect, such as power lines [38] or pipes/sewerage [33, 40], to name but a few.

Nevertheless, without doubt, the kind of inspection-oriented robot for which a major diversity of designs can be found is the crawler/climber. In this case, the desired tasks and the field of application define the locomotion type and the adhesion method to be used, as well as the platform size [41]. The different locomotion types range from *arms and legs*, with two- to eight-, or even more, leg designs, to *wheels and chains*, for relative smooth surfaces, to *sliding frames*, typically involving two frames which can move against each other, to *wires and rails*, what simplifies the mechanics but limits the robot to the supporting setup. As for adhesion, probably the most common principle is the *magnetic adhesion*, intended for the inspection of ferromagnetic structures either through permanent magnets or electromagnets. A second technique involves *pneumatic* or *negative pressure adhesion*, either using passive suction cups, active suction chambers or vortex systems. *Mechanical adhesion* is another attraction principle, based on the use of claws, or through gripping/clamping systems. Less common adhesion principles comprise *electrostatic attraction*, by means of electroadhesive pads involving conductive electrodes and insulation substrates, and *chemical adhesion*, using sticky tapes attached to wheels and tracks or multi-legged systems equipped with tacky elastomere feet.

Basic navigation skills

First works on self-localization and mapping date back to the early 1990’s [15]. Since then, a vast number of different solutions have emerged under the categories of either *relative localization*, a.k.a. *dead reckoning* and *path integration*, or *absolute localization*. First attempts on *relative localization* adopted strategies for motion estimation based on proprioceptive sensors, such as wheel odometers and Inertial Mearurement Units (IMU) [21]. These methods, despite being simple, were soon detected to be affected by drift in the estimations, due to the unavoidable uncertainty of sensor data and the cumulative nature of the estimation process. Since the eruption of Simultaneous Localization and Mapping (SLAM) approaches in the late 1990’s [46], *relative localization* has been typically solved together with mapping within a single framework, what permits counteracting the aforementioned drift by incorporating into the calculations the relationships between sensor data and an abstract representation of the environment named as a map, together with the detection of loop closures, i.e. already visited places. More precisely, SLAM systems formulate the estimation process around a closed loop that can combine dead reckoning with exteroceptive sensing embedded within incremental filtering or non-linear optimization frameworks, via respectively variations of, among others, *Kalman filters*, *particle filters* or *information filters* [47], or through *factor graph-based Dynamic Bayes Networks (DBN) solvers* [26]. Nevertheless, recently, odometry algorithms combining visual and inertial information have been reported to result into very small drift [13].

The different approaches outlined above rely on the availability of sensing devices appropriate for the inspection scenarios, so that an adequate level of robustness in the final application is achieved. In this regard, a number of possibilities arise *a priori*. For a start, *ultrasounds* became a popular range sensing modality in a great amount of foundational works on self-localization for ground platforms [8], particularly after occupancy grid maps were introduced [34]. For the outdoor structures involved in certain inspection tasks, ultrasounds would be preferred, although its limited range and insufficient angular resolution can make them not useful at long range.

2D laser scanners have been extensively used for both ground and aerial vehicles due to their accuracy and speed [7]. During the last decade, a number of solutions can be found for the particular case of aerial inspection platforms [25, 35]. *3D laser scanners* have also been incorporated in both ground [17] and aerial applications [37, 9], using in the latter cases rotating 2D scanners.

Vision cameras have also become quite popular recently. Cameras' success in general robotics come mainly from the richness of the sensor data supplied, combined with low weight, low-power designs and a relatively low price, together with the increasing power of current state-of-the-art computers and embedded processors, which permits successfully running real-time image processing on-board. Latest developments comprise visual SLAM solutions either based on feature tracking [14] or adopting direct/semi-direct approaches [20], which optimise the scene geometry directly on the basis of image intensities. In this case, the challenge for inspection is that it can so be that the environments of interest exhibit almost no texture or the observations are performed at very small grazing angles, which limit the reliability of feature detection.

Depth and *RGB-D cameras* are the latest type of sensor which has been incorporated in relative localization approaches [27, 28]. These cameras allow the capture of reasonably accurate mid-resolution depth and appearance information at high data rates. The choice among them is typically between time-of-flight cameras, with usually lower resolution and depth precision, stereo vision, which requires a textured environment, or structured light, which can be affected by ambient lighting. The sensitivity of 3D vision to external conditions can make this a challenging sensor for inspection.

The possibilities enumerated above require different levels of computational capabilities onboard. Depending on the inspection platform to be used, it could be the case that not the most accurate localization (and mapping) algorithm can be integrated, but one has to choose a lower-performance solution compatible with the available resources as for volume, weight, power consumption, onboard sensors, etc. Actually, by way of example, in commercial inspection crawlers, the navigation capabilities are constrained mostly to odometry and inertial sensing, mainly because of limitations from lightweight designs. In these cases, displacement measurement can become particularly tricky because of the complexity of the vehicle kinematics due to e.g. the interleaving between wheel adhesion and wheel motion, which can be affected by slippage on the inspected structures, as well as by disturbing elements such as umbilical cords.

At the other end of the spectrum, *absolute localization* solutions yield a stable estimation error although at the expense of a specific infrastructure that has to be deployed in the environment, i.e. beacons whose position must be available at all times. At this point, it is important to notice that the installation of this equipment does not need to be necessarily a problem in an inspection application, provided the deploying time is not excessive.

The by far most adopted solution in this regard is the Global Positioning System (GPS). The use of GPS receivers, however, is subject to some important limitations that make them not usable in the context of an inspection mission because of degradation of the positioning signal in poor GPS reception areas, e.g. due to satellites being occluded by the inspected structures, multi-path effects, etc. To the contrary, wireless-based localisation [31] is especially suited for this kind of scenarios. Among the different possibilities available nowadays, Ultra-Wide Band (UWB) systems have emerged as one of the leading positioning technologies because the UWB ultra-short pulses are resilient to frequency-dependent absorption, thanks to their large bandwidth, and because ultimate accuracy can range from 2 cm (ideal conditions) to 50 cm (non-line of sight scenario) [6]. Several categories of UWB algorithms can be distinguished depending on how the position is inferred from the radio signals travelling between the beacons and the target node [2]: *based on time of arrival* (TOA), *based on time difference of arrival* (TDOA), *based on angle of arrival* (AOA), *based on received signal strength* (RSS), and *hybrid approaches*. TOA and TDOA have been shown to exhibit higher accuracy than the other algorithms, and even better results have been achieved by the hybrid approaches which combine some of the different alternatives.

Alternatively, camera-based absolute localization has been adopted as a very accurate pose estimation method (to the point that most times it becomes a source of ground truth data for performance assessment), particularly for micro-aerial systems. Nevertheless, the volume which can be covered is rather limited so as to be useful for real inspection applications, while their cost grows significantly with precisely the enclosed volume.

Overview of motion planning

Motion planning techniques can be roughly classified into deliberative/path planners and reactive strategies. Deliberative/path planners are generally characterized by computing, and later executing, the path to a given target based on a known map of the environment. Roughly speaking, there are two major classes of algorithms used for path planning: *heuristic search* [22] and *probabilistic sampling* [18]. As a general rule, the use of heuristic search-based algorithms is preferred when the dimensionality of the planning problem is low and there is an interest in obtaining a least-cost path. In the rest of cases, the use of probabilistic sampling-based algorithms is recommended.

Using abstract world models is, generally speaking, a time-consuming process [5]. To cope with this limitation, reactive approaches control the motion of the robot by means of, exclusively, the sensory inputs. Most popular reactive techniques comprise artificial potential fields, vector field histograms (VFH, VFH+, and VFH*), the nearness diagram strategy, the curvature-velocity method, the Bug family of algorithms, and the popular Dynamic Window Approach (DWA) [43]. The main advantage of these strategies lies in their high computational efficiency, which makes them well suited for navigating through unknown and dynamic scenarios. Nevertheless, they are only able to provide sub-optimal results due to the locality of the information which is managed. Some approaches have addressed the latter problem providing solutions ensuring global convergence at a low computational cost [3].

Since neither deliberative nor reactive strategies are really free of problems, new hybrid approaches combining the strengths of the two aforementioned strategies and minimizing their corresponding weak points have been proposed [48]. In this kind of control systems, two components, one reactive and the other one deliberative, interact via an intermediate layer. To be more precise, the deliberative layer generally computes first a global path to the target using an up-to-date model of the environment. Afterwards, this path is safely followed by the reactive layer avoiding any unexpected obstacle in real time.

Complete surface inspection through coverage-aware algorithms

This section illustrates the most relevant aspect of the navigation topic regarding inspection, which has to do with the automatic generation of inspection paths, i.e. paths that allow a robot to visit every relevant point, from an inspection point of view, of a target structure. The computation of these paths requires the use of a special type of path planning algorithms, collectively known under the name of Complete Coverage Path Planning (CCPP). The following sections consider and provide the details of some of the most representative algorithms in this class.

Key aspects of path planning

In robotics, the concept of path planning is intrinsically related to the concept of configuration space. Also known as C-space, the *configuration space* is the set of all possible configurations that a robot can assume, where a configuration is a complete specification of the position of every point on the robot. The minimum number of parameters needed to represent a configuration always coincides with the number of Degrees Of Freedom (DOF) of the robot, and, hence, the dimension of the C-space is given by the number of DOFs. In order to illustrate the concept of C-space with an example, let us consider the planar robot arm depicted in Fig. 2. As can be observed, this robot arm has two revolute joints, labeled as joint 1 and joint 2, which allow a rotation around the joint axis. These rotations are described by the angles θ_1 and θ_2, which actually represent the two DOFs of the robot arm, i.e. these angles constitute the robot arm configuration, and, thus, the set of all possible tuples $\langle\theta_1, \theta_2\rangle$ make up the C-space of the robot arm. Finally, the C-space is subdivided into two subspaces, namely the forbidden C-space and the free C-space. On the one hand, the *forbidden C-space* comprises those configurations that cause the robot to collide with an obstacle in the environment. On the other hand, the *free C-space* is the set of all configurations for which no collision occurs.

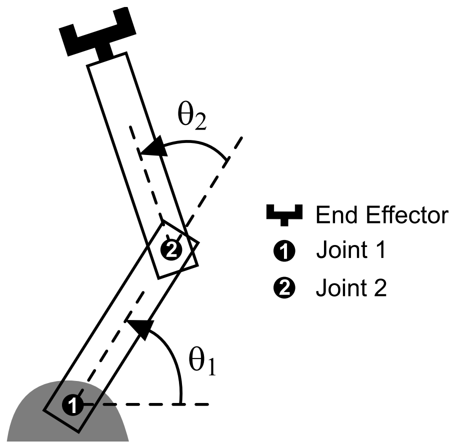

Fig. 2 A planar robot arm with two revolute joints.

Given the above definition, the problem of path planning can be stated as the task of calculating a *feasible* path, i.e. a path in the free C-space that connects some initial configuration q_{start} to some desired target configuration q_{target}. In path planning, each path has an associated cost which reflects the cost of moving the robot along that path. Keeping this in mind, a path is said to be *optimal* if its cost is minimal across all possible paths leading from q_{start} to q_{target}. Additionally, a path planning algorithm is said to be *optimal* if it always finds an optimal path between any two given configurations q_{start} and q_{target}, if one exists. As a final point, a path planning algorithm is said to be *complete* if it always finds a feasible path in finite time when one exists, and lets us know in finite time if there is no feasible path.

Some key facts about path planning are enumerated in the following:

- The cost of a path is generally computed by means of a cost function that represents a set of preferences over some features of the path, such as length, smoothness, clearance, time of travel and/or energy consumption. As it is obvious, the higher the degree to which preferences are satisfied, the lower the path cost.
- The computation of the robot's forbidden C-space —or equivalently, the computation of the robot's free C-space— is one of the most time-consuming operations in path planning, specially when the shapes of both the robot and obstacles are complex and when the dimension of the C-space is high. From a practical perspective, the above operation requires the use of a specific algorithm to detect intersections between each robot configuration and the obstacles which are present in the environment. Several strategies have been developed with the aim of significantly reducing the computational burden of detecting such intersections. Some of those strategies use a set of basic geometric primitives, e.g. spheres, cylinders, cones, convex hulls, etc. as approximations of both shapes to simplify the calculations. Other strategies reduce the number of intersections to be determined by only computing the portion of the robot's forbidden/free C-space which is really relevant to find a path that joins the given pair of configurations $\{q_{start}, q_{target}\}$.
- In heuristic search, the C-space is typically discretized and represented by a weighted graph, where nodes represent configurations. Two configurations, q_x and q_y, are connected by an edge if they belong to the free C-space and no intermediate configuration is visited by the robot when moving from either q_x to q_y or q_y to q_x. The weight of each edge reflects the cost of transitioning between the two corresponding configurations. After constructing the weighted graph, a graph search algorithm finds a path from q_{start} to q_{target}. These algorithms make use of a heuristic function to guide the search towards the least-cost path on the weighted graph. Some of the most popular graph search algorithms are Depth First Search, Breadth First Search, Dijkstra, A^* and D^*, just to mention a few.
- Alternatively, in probabilistic sampling, the C-space does not need to be discretized. More specifically, these algorithms sample the C-space by selecting a large number of configurations at random, and retain the valid configurations as nodes in a graph-like structure called *roadmap*. By definition, a configuration q_x is said to be *valid* if there exists a collision-free path that connects the candidate node q_x to another node in the roadmap. The construction of the roadmap is stopped when the initial configuration q_{start} and the desired final configuration q_{target} can be added as valid nodes to the roadmap. As soon as this happens, the query phase begins, where the roadmap is used to find a path between q_{start} and q_{target}. Probabilistic sampling-based algorithms are probabilistically complete, meaning that the probability of finding a path with endpoints q_{start} and q_{target}, if one exists, approaches 1 as more time is spent. Probabilistic RoadMaps (PRM) and Rapidly-exploring Random Trees (RRT) are two of the most extensively-used probabilistic sampling-based algorithms.

- Both the kinematic and dynamic constraints of the robot can be taken into account when planning a path. The most common way to do this is to transform the C-space into an equivalent space which additionally embeds such kinodynamic constraints. An example of this transformation can be found in [10, 12].
- Both heuristic and probabilistic strategies cover a useful class of algorithms named *anytime*, which is able to trade off running time and solution quality in domains where quick reactions are required. This kind of algorithms generates a succession of progressively better solutions, where the time needed for computing each solution is related to its quality. A bunch of anytime algorithms have been developed so far, including ARA* [30], deriving from A*, ARRT [23], deriving from RRT, and vABUG [4], just to cite some examples.

Overview of complete coverage algorithms

In short, Complete Coverage Path Planning (CCPP) is a special type of path planning task which requires the calculation of a path that passes over each relevant point of an area or volume of interest while avoiding collisions with obstacles [24]. This task is essential not only for inspection robots, but for other robotic applications including, among others, vacuum cleaning robots, lawn mowers, underwater imaging/scanning robots, window cleaners and painter robots. Robotic inspection of structures such as aircrafts, bridges, buildings and ships, just to name a few, can be regarded as a critical application since any defect in these structures could seriously affect their integrity.

Generally speaking, CCPP algorithms should find a path which satisfies the following constraints: (C1) the coverage is complete (i.e. all points of the area or volume of interest are covered); (C2) there is no overlapping or repetition (i.e. no point of the area or volume of interest is covered more than once); (C3) there is no collision with any obstacle in the environment; and, finally, (C4) the path consists of a set of simple motion trajectories such as straight lines or circles in order to make the control of the robot easier. The most popular algorithms that solve the CCPP problem are considered next:

- The most simple strategy to address the CCPP problem is randomization. More precisely, a randomized algorithm makes the robot move randomly in the target environment while avoiding obstacles. If this behavior is maintained for a long time, the robot is expected to completely cover the target environment. Some vacuum cleaning robots partially or fully adopt this technique to successfully clean the floor of an entire home. Nevertheless, it is also important to highlight that randomized CCPP algorithms are not well-suited to efficiently cover large spaces. Both underwater robots and aerial robots typically operate in large, 3D environments. Addressing these vast environments by following a random-movement strategy can be unaffordable in terms of execution time and energy consumption. This limitation explains the need for more sophisticated CCPP algorithms.
- The classical manner of addressing the CCPP problem consists in decomposing the free space in the target environment into simple and non-overlapping regions

known as *cells*. Among all the algorithms based on this classical approach, one of the most powerful is described in [1]. This algorithm applies a class of cellular decomposition termed Morse which involves the following steps:

S1. An imaginary line, formally named *sweep-line*, is moved across the target environment from left to right. At each new position of the sweep-line, a search for critical points is carried out. Plainly speaking, a *critical point* is a point where the sweep-line encounters an obstacle and where there are no obstacles at a short distance both above and below that point. By way of illustration, Fig. 3(a) shows a situation where a critical point is detected at point P_1.

S2. The second step begins once all critical points have been found (see Fig. 3(b)). These critical points are used to determine the location of the cell boundaries. By definition, a *cell boundary* is a vertical line segment which touches an obstacle at only one point, being this point a *critical point*. Figure 3(c) depicts the cell decomposition resulting from the set of critical points of Fig. 3(b).

S3. The third step is related to the construction of an adjacency graph. An *adjacency graph* encodes the adjacency relationships between cells. In this sense, two cells are said to be *adjacent* if they share a common cell boundary. In an adjacency graph, a node corresponds to a cell, while an edge represents an adjacency relationship between the connected cells. Figure 3(d) plots the adjacency graph associated with the cellular decomposition of Fig. 3(c).

S4. The goal of the fourth step is to find an exhaustive walk through the previously-constructed adjacency graph. In essence, an *exhaustive walk* is a path where each node is visited at least once. A path planning algorithm is required to compute this exhaustive walk. By way of example, Depth-First Search (DFS) and Breadth-First Search (BFS) are two algorithms for path planning which ensure traversing all the nodes of a given graph. Figure 3(e) shows the sequence of nodes corresponding to the exhaustive walk computed on the adjacency graph of Fig. 3(d).

S5. As a last step, all that remains is to decide the motion model that will be used to cover each cell individually and completely. With no doubt, the zigzag pattern is the most common approach to cover a cell. Figure 3(f) illustrates how cell C_2 of Fig. 3(c) would be covered according to a zigzag path.

- Nevertheless, the most widely-used solution to address the CCPP problem is by representing the target environment as a grid. More specifically, a grid is decomposed into a set of cells of equal area and shape. These cells are typically square (although other shapes, such as triangles, can be employed). Each cell is assigned a value that indicates the probability of that cell of being occupied by an obstacle. Based on this value, a cell is labeled as either free or not. By way of example, Fig. 4(a) depicts the grid-like representation of the environment of Fig. 3(a). As can be observed, due to the grid resolution, the shape of the obstacle can only be approximated. Grid-based CCPP algorithms provide thus a weaker form of completeness (see constraint C1 above) called *resolution completeness*, or, in other words, these algorithms ensure complete coverage only if the resolution of the grid is fine enough.

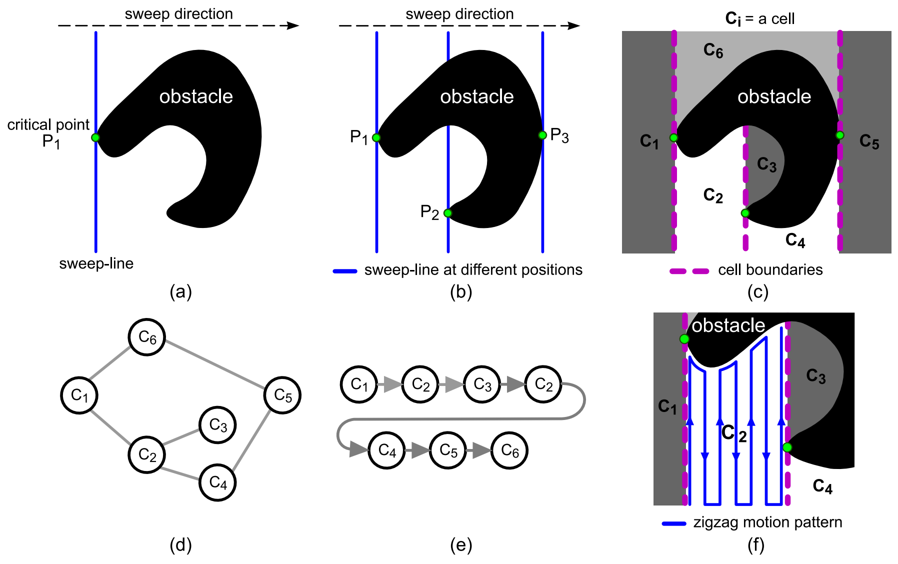

Fig. 3 Step-by-step explanation of how a Morse-based CCPP algorithm operates.

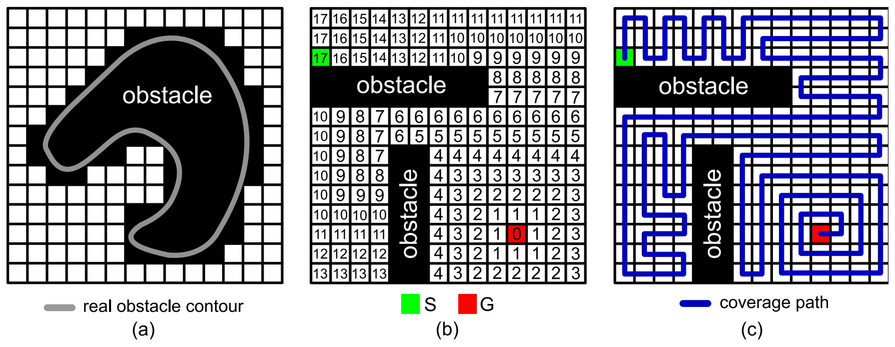

Fig. 4 Illustration of the step-by-step operation of a grid-based CCPP algorithm.

Among all grid-based CCPP algorithms reported in the literature, [49] is a popular choice due to its simplicity and unique features. This algorithm consists of three following steps:

S1. One cell of the grid is marked as the starting point S and another cell is marked as the goal point G $\neq$ S. Both S and G should be free cells.

S2. A wave-front is propagated from G to S through the free space (i.e. through the cells of the grid that are labeled as free). Figure 4(b) illustrates the way the wave-front propagates in an environment containing two rectangle-shaped obstacles. To begin with, G is assigned a label 0. Afterwards, all the cells adjacent to G are assigned a label 1. The third propagation step assigns a label

  2 to those non-labelled cells that are adjacent to some 1-labelled cell. This procedure finishes when a label has been given to each cell in the grid.

  S3. A coverage path is computed by starting from S and choosing the neighboring cell with the highest label that has not been visited yet (see Fig. 4(c)). In case two or more unvisited neighboring cells have the same highest value, one of them is randomly selected.

  To finish, notice that this grid-based CCPP algorithm allows, as a unique feature, specifying the start and the final point of the coverage path.

- All the CCPP algorithms reviewed up-to-now assume that the target environment can be represented as a planar surface. Obviously, this assumption is not valid when the inspection of the structure requires considering the surrounding three-dimensional space. The concept of CCPP is rather different when dealing with this kind of target environments: in this case, the aim is not to cover all the obstacle-free points of the space, but only a subset of these points. To be more clear, the aim for a three-dimensional CCPP algorithm is to cover those obstacle-free points of the space that allow the robot to make complete coverage of a certain three-dimensional structure. Taking an example from another domain, in the case of an automatic spray-painting application, the aim is to move the robot's spray gun through a path that completely paints a given workpiece.
  Many examples of three-dimensional CCPP algorithms are found in the literature for different problems. By way of illustration, the approach described in [19] is overviewed in the following. This algorithm is specially designed to perform complete coverage of complex three-dimensional structures, such as the in-water part of a ship's hull. To achieve this, the algorithm operates in three steps:

  S1. A graph of feasible paths is built until the set of nodes allows complete coverage of the desired three-dimensional structure by using a random sampling-based path planner.

  S2. The previous graph is used to look for a minimum-cost path that covers the entire three-dimensional structure. Figure 5(a) is an example of a path computed according to this step. More specifically, in this example, the aim was to plan a path that allows an AUV equipped with a sonar to inspect a ship's propeller.

  S3. As a final step, the previous path is smoothed and shortened, as shown in Fig. 5(b).

Examples of Application

Among the huge diversity of robot-based inspection approaches described in the related literature so far, this section reviews three of them to illustrate the different possibilities and how these specific designs take profit of their particularities to result useful for the respective inspection problems.

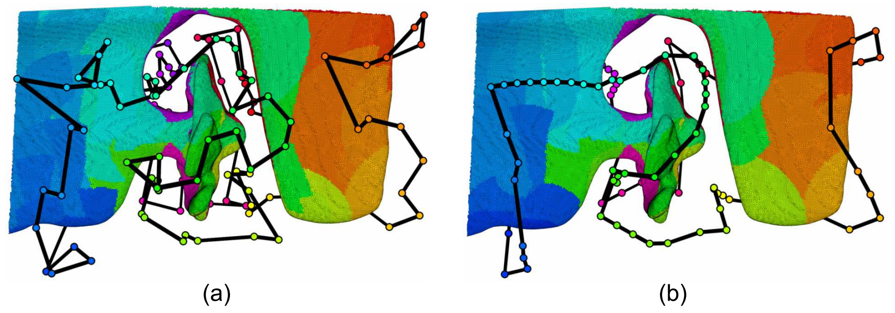

Fig. 5 Sampling-based path planning for autonomous ship hull inspection. (a) initial coverage path; (b) the path of (a) once it has been smoothed and shortened. Source: Ref. [19], with permission from the authors.

As a first example, we will consider the robotic device presented in [29], which consists in a platform developed for bridge deck inspection and evaluation. The vehicle consists in the commercial platform shown in Fig. 6(a) which has been equipped with a collection of non-destructive evaluation sensors for that purpose. In second place, we will overview the aerial platform presented in [11], devised for the visual inspection of vessels. This vehicle is fitted with cameras and it is supposed to be operated in the different vessel compartments, including wide areas, such as cargo holds, and other narrow and dark spaces, such as water ballast tanks. Figure 7 shows two realizations of the same approach. Finally, we will revise the platform presented in [44], consisting in a crawler which is capable of climbing on vertical and rough surfaces, what makes this vehicle also suitable for surveillance and SAR applications. This robotic platform is shown in Fig. 6(b).

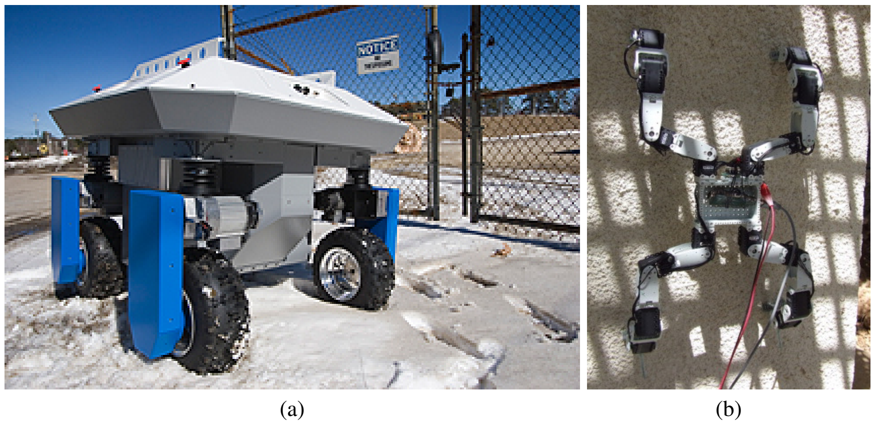

Fig. 6 (a) Commercial robotic platform used to implement the bridge inspector described in [29] [source: https://en.wikipedia.org/wiki/ActivMedia_Robotics, licensed according to https://creativecommons.org/licenses/by-sa/3.0/]. (b) Robotic crawler devised to climb vertical rough surfaces [source: Ref. [44], with permission from the authors].

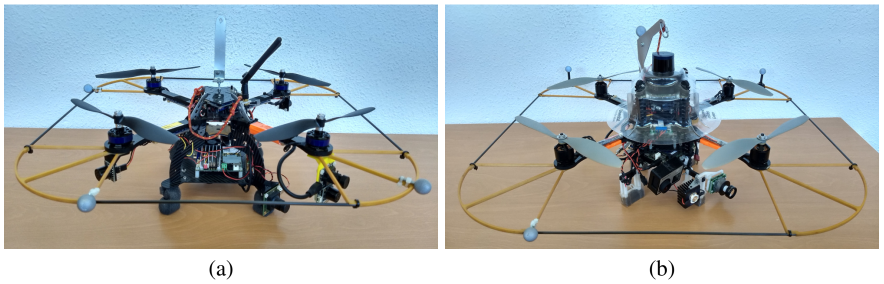

Fig. 7 Two realizations of the robotic platform for vessel inspection described in [11]: (a) lightweight platform equipped with two optical-flow sensors to estimate the vehicle speed, (b) higher-payload platform equipped with a laser scanner.

Regarding locomotion, the three platforms have been conceived to ensure a good perception of the element or surface to be inspected. In this regard, the bridge inspector is just intended for the bridge deck, so wheels are enough for the platform displacement. Concerning the vessel inspection device, it consists in a Micro-Aerial Vehicle (MAV) capable of reaching the highest parts of the vessel structure. Three alternative multi-rotor configurations, including two quadcopters and one hexacopter, are considered by the authors, what allows for different sensor suites in accordance to the available payload. Finally, unlike other crawlers which are based on the use of magnets, suction or adhesive methods for the attachment to the walls, the crawler selected here consists in a four-legged device fitted with claws devised for the attachment to vertical rough surfaces, such as stucco walls.

The control architectures of the three platforms also present important differences. The bridge inspector has been designed to be fully autonomous. Assuming that the bridge is straight and that the area to inspect presents a rectangular shape, the vehicle solves a coverage planning problem. To do that, the robotic platform is supplied with the GPS coordinates of three corners of the rectangle and the platform performs a zigzag path computed through trapezoidal decomposition. An artificial potential field approach is used to make the platform follow the trajectory.

On the contrary, the control architecture of the vessel inspector adopts a semi-autonomous approach. In more detail, it is based on the Supervised Autonomy (SA) paradigm, so that the human surveyor is kept in the positioning control loop, from which it sends displacement commands using a joystick/gamepad. At the same time, the vehicle control software is in charge of all the safety-related issues, such as collision avoidance and platform status monitoring. The idea behind this approach is to provide the vessel surveyor with a flying camera which can be easily operated by a non-expert, who maybe has never used an aerial robot.

Unlike the previous examples, the autonomous capabilities of the crawler are focused on solving a tracking problem. To be precise, the vehicle is able to track a predefined path specified by a collection of segments. To this end, the control architecture splits the segments into sub-segments and then computes and makes each leg perform the suitable movement that allows the central part of the platform to approximately follow the indicated path. At each step, each end effector is checked to detect whether it is released from the wall, or it is out of the allowed space, or a small force is applied on it. If some of these conditions is satisfied, then the corresponding leg is moved to the next possible position. On the contrary, if a leg is overloaded, the vehicle performs small leg orientation moves to decrease the load on that end effector.

The three robotic platforms do exhibit important differences regarding the sensor suites used for state estimation. To follow the computed trajectory, the control architecture of the bridge inspector requires an accurate localization of the vehicle. This is performed by means of an extended Kalman filter which combines RTK GPS data with IMU readings and wheel encoder measurements.

On the contrary, the estimation of the vehicle position is not required by the control software of the vessel aerial inspector, whose state comprises estimated 3-axis velocities and height. The authors propose two alternative sensor suites which can be installed on board a MAV depending on its payload capacity. On the one hand, a lightweight MAV such as the one shown in Fig. 7(a) makes use of two Optical Flow (OF) sensors (one OF sensor looks to the ground and the other looks to the inspected surface) for the estimation of the vehicle speed. On the other hand, a platform with a higher payload capacity, such as the one shown in Fig. 7(b) features a laser scanner to estimate its motion. In all cases, the flight height is estimated using a laser altimeter, and the final platform state is estimated involving the IMU accelerations through a Kalman filter.

Concerning the legged crawler, the position and the force applied on its end effectors are computed by means of inverse kinematics using the torque and angle measured at the actuators. No additional sensors are used to estimate the position of this platform.

Finally, only the two first platforms face more advanced navigation capabilities, such as collision avoidance. In the case of the bridge inspector, obstacle detection is performed using three laser scanners, one on the front and one at each side of the platform. In the case of the aerial vessel inspector, the platform equipped with a laser scanner makes use of this sensor for collision detection, while the remaining configurations, with less payload capacity, make use of US range sensors for that purpose. Within its control architecture, the collision avoidance is performed by a collection of robotic behaviours which are in charge of accomplishing the displacement commands indicated by the user, while preserving the physical integrity of the platform and checking the flight viability (e.g. monitoring the battery voltage level).

Further Directions for Research

On the one hand, future inspection systems are to be capable to reduce human intervention to the minimum, and even operate completely unsupervised in certain cases, what means fitting inspection robots with as many autonomous functionalities as feasible, in a way compatible with robustness, i.e. only reliable autonomy is going to be useful. On the other hand, and now speaking in terms closer to the practical application, research is to focus on the design of systems capable of performing accurate and cost-effective inspection and assessment, either of civil and/or private infrastructures, so that all this reverts in safer operation and less budget being spent in repairs.

References

1. E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull. Morse decompositions for coverage tasks. *The International Journal of Robotics Research*, 21(4):331–344, 2002.
2. A. Alarifi, A. Al-Salman, M. Alsaleh, A. Alnafessah, S. Al-Hadhrami, M. Al-Ammar, and H. Al-Khalifa. Ultra wideband indoor positioning technologies: Analysis and recent advances. *Sensors*, 16(5), 2016.
3. J. Antich and A. Ortiz. A convergent dynamic window approach with minimal computational requirements. In *Proc. International Conference on Intelligent Autonomous Systems*, pages 183–192, 2008.
4. J. Antich and A. Ortiz. A rapid anytime path planner with incorporated range sensing to improve control on solution quality. In *Proc. International Conference on Intelligent Autonomous Systems*, pages 207–216, 2010.
5. R. Arkin. *Behavior-based robotics*. The MIT Press, 1998.
6. A. Bahr, A. Feldman, J. Colli-Vignarelli, S. Robert, C. Dehollain, and A. Martinoli. Modeling and benchmarking ultra-wideband localization for mobile robots. In *Proc. IEEE International Conference on Ultra-Wideband*, pages 443–447. IEEE, 2012.
7. T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): part ii. *IEEE Robotics and Automation Magazine*, 13(3):108–117, 2006.
8. K. R. Beevers and W. H. Huang. Slam with sparse sensing. In *Proc. IEEE International Conference on Robotics and Automation*, pages 2285–2290, 2006.
9. M. Beul, N. Krombach, Y. Zhong, D. Droeschel, M. Nieuwenhuisen, and S. Behnke. A high-performance mav for autonomous navigation in complex 3d environments. In *Proc. IEEE International Conference on Unmanned Aircraft Systems*, pages 1241–1250, 2015.
10. J. L. Blanco, M. Bellone, and A. Gimenez-Fernandez. TP-Space RRT: Kinematic path planning of non-holonomic any-shape vehicles. *International Journal of Advanced Robotic Systems*, 12(5), 2015.
11. F. Bonnin-Pascual and A. Ortiz. A Flying Tool for Sensing Vessel Structure Defects using Image Contrast-based Saliency. *IEEE Sensors Journal*, 16(15):6114–6121, 2016.
12. R. Bordalba, J. Porta, and L. Ros. Kinodynamic planning on constraint manifolds. Technical report, Kinematics and Robot Design Research Group, Institut de Robòtica i Informàtica Industrial, CSIC-UPC, 2017.
13. C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.
14. G. Chowdhary, E. N. Johnson, D. Magree, A. Wu, and A. Shein. GPS-denied indoor and outdoor monocular vision aided navigation and control of unmanned aircraft. *Journal of Field Robotics*, 30(3):415–438, 2013.
15. I. J. Cox and G. T. Wilfong, editors. *Autonomous Robot Vehicles*. Springer-Verlag, 1990.
16. N. A. Cruz, A. C. Matos, R. M. Almeida, B. M. Ferreira, and N. Abreu. TriMARES - a hybrid AUV/ROV for dam inspection. In *Proc. IEEE/MTS OCEANS Conference*, pages 1–7, 2011.
17. D. Droeschel, M. Schwarz, and S. Behnke. Continuous mapping and localization for autonomous navigation in rough terrain using a 3d laser scanner. *Robotics and Autonomous Systems*, 88:104 – 115, 2017.
18. M. Elbanhawi and M. Simic. Sampling-based robot motion planning: A review. *IEEE Access*, 2:56–77, 2014.

19. B. Englot and F. Hover. Sampling-based coverage path planning for inspection of complex structures. In *Proc. International Conference on Automated Planning and Scheduling*, pages 29–37, 2012.
20. M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza. Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle. *Journal of Field Robotics*, 33(4):431–450, 2016.
21. L. Feng, H. Everett, and J. Borenstein. *Where am I? Sensors and methods for autonomous mobile robot positioning*. University of Michigan, 1996.
22. D. Ferguson, M. Likhachev, and A. Stentz. A guide to heuristic-based path planning. In *Proc. International Workshop on Planning under Uncertainty for Autonomous Systems (International Conference on Automated Planning and Scheduling)*, 2005.
23. D. Ferguson and A. Stentz. Anytime RRTs. In *Proc. International Conference on Intelligent Robots and Systems*, pages 5369–5375, 2006.
24. E. Galceran and M. Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258 – 1276, 2013.
25. P. Gohl, M. Burri, S. Omari, J. Rehder, J. Nikolic, M. Achtelik, and R. Siegwart. Towards autonomous mine inspection. In *Proc. International Conference on Applied Robotics for the Power Industry*, pages 1–6, 2014.
26. G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.
27. P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments. In O. Khatib, V. Kumar, and G. Sukhatme, editors, *Proc. International Symposium on Experimental Robotics, ISER2010*, pages 477–491. Springer, 2014.
28. C. Kerl, J. Sturm, and D. Cremers. Dense visual SLAM for RGB-D cameras. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2100–2106, 2013.
29. H. M. La, R. S. Lim, B. Basily, N. Gucunski, J. Yi, A. Maher, F. A. Romero, and H. Parvardeh. Autonomous Robotic System for High-Efficiency Non-Destructive Bridge Deck Inspection and Evaluation. In *Proc. IEEE International Conference on Automation Science and Engineering*, pages 1053–1058, 2013.
30. M. Likhachev, G. J. Gordon, and S. Thrun. ARA* : Anytime A* with provable bounds on sub-optimality. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Proc. Advances in Neural Information Processing Systems*, pages 767–774. MIT Press, 2004.
31. H. Liu, H. Darabi, P. Banerjee, and J. Liu. Survey of wireless indoor positioning techniques and systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(6):1067–1080, 2007.
32. E. Menendez, J. G. Victores, R. Montero, S. Martínez, and C. Balaguer. Tunnel Structural Inspection and Assessment using an Autonomous Robotic System. *Automation in Construction*, 87:117–126, 2018.
33. J. M. Mirats and W. Garthwaite. Robotic Devices for Water Main In-Pipe Inspection: A Survey. *Journal of Field Robotics*, 27(4):491–508, 2010.
34. H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proc. IEEE International Conference on Robotics and Automation*, volume 2, pages 116–121, 1985.
35. A. Ortiz, F. Bonnin-Pascual, and E. Garcia-Fidalgo. Vessel inspection: A micro-aerial vehicle-based approach. *Journal of Intelligent and Robotic Systems*, 76(1):151–167, 2014.
36. A. Ortiz, F. Bonnin-Pascual, A. Gibbins, P. Apostolopoulou, W. Bateman, M. Eich, F. Spadoni, M. Caccia, and L. Drikos. First steps towards a roboticized visual inspection system for vessels. In *Proc. IEEE International Conference on Emerging Technologies and Factory Automation*, pages 1–6, 2010.
37. T. Ozaslan, G. Loianno, J. Keller, C. J. Taylor, V. Kumar, J. M. Wozencraft, and T. Hood. Autonomous navigation and mapping for inspection of penstocks and tunnels with MAVs. *IEEE Robotics and Automation Letters*, 2(3):1740–1747, 2017.

38. A. Pagnano, M. Höpf, and R. Teti. A Roadmap for Automated Power Line Inspection. Maintenance and Repair. In *Proc. Conference on Intelligent Computation in Manufacturing Engineering*, pages 234–239, 2013.
39. P. Ridao, M. Carreras, D. Ribas, and R. Garcia. Visual Inspection of Hydroelectric Dams using an Autonomous Underwater Vehicle. *Journal of Field Robotics*, 27(6):759–778, 2010.
40. N. S. Roslin, A. Anuar, M. F. A. Jalal, and K. S. M. Sahari. A review: Hybrid locomotion of in-pipe inspection robot. In *Proc. International Symposium on Robotics and Intelligent Sensors*, volume 41, pages 1456–1462, 2012.
41. D. Schmidt and K. Berns. Climbing robots for maintenance and inspections of vertical structures—a survey of design aspects and technologies. *Robotics and Autonomous Systems*, 61(12):1288–1305, 2013.
42. M. Siegel and P. Gunatilake. Remote Enhanced Visual Inspection of Aircraft by a Mobile Robot. In *Proc. IEEE Workshop on Emerging Technologies, Intelligent Measurement and Virtual Systems for Instrumentation and Measurement*, 1998.
43. R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza. *Introduction to Autonomous Mobile Robots*. The MIT Press, 2nd edition, 2011.
44. A. Sintov, T. Avramovich, and A. Shapiro. Design and Motion Planning of an Autonomous Climbing Robot with Claws. *Robotics and Autonomous Systems*, 59(11):1008–1019, 2011.
45. C. Stachniss. *Robotic Mapping and Exploration*. Springer, 1st edition, 2009.
46. S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. The MIT Press, 2005.
47. S. Thrun, Y. Liu, D. Koller, A. Y. Ng, Z. Ghahramani, and H. Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *The International Journal of Robotics Research*, 23(7-8):693–716, 2004.
48. C. Urmson, C. R. Baker, J. M. Dolan, P. Rybski, B. Salesky, W. R. L. Whittaker, D. Ferguson, and M. Darms. Autonomous driving in traffic: Boss and the urban challenge. *AI Magazine*, 30(2):17–29, 2009.
49. A. Zelinsky, R. Jarvis, J. C. Byrne, and S. Yuta. Planning paths of complete coverage of an unstructured environment by a mobile robot. In *Proc. International Conference on Advanced Robotics*, pages 533–538, 1993.