Automation in Construction 94 (2018) 244–256

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

Localisation of a mobile robot for bridge bearing inspection

H. Peel[a,*], S. Luo[a], A.G. Cohn[b], R. Fuentes[a]

[a] School of Civil Engineering, University of Leeds, United Kingdom
[b] School of Computing, University of Leeds, United Kingdom

ARTICLE INFO

Keywords:
Bridge bearings
Robot
Inspection
Localisation

ABSTRACT

Bridge bearings are a critical component of a bridge and require regular visual inspection to ensure the safe operation of the bridge throughout its life. However, the bearings are often located in spaces that are difficult or hazardous to reach, which can impact how often the bearings are inspected. In addition, these spaces are small and offer significant challenges for tele-operation due to line-of-sight restrictions; hence, some level of autonomy is required to make robotic inspection possible. In this work, a robotic solution to bridge bearing inspection is presented, and localisation methods are assessed as the first, and most, important step towards automation. Robot localisation is performed in both a lab environment and a real bridge bearing environment. In this paper, Adaptive Monte-Carlo Localisation is considered for localisation in a known map and gave comparable results to Hector-SLAM, with all results less than a defined error threshold of 10 cm. A combination of both of these methods are proposed to give a more robust approach that gives errors lower than the defined threshold in the real bridge. The experiments also show the need to provide an accurate starting point for each inspection within the bearing, for which we notionally suggest the use of a docking station that could also be used for improved autonomy, such as charging. In addition, proof-of-concept approaches for visual inspection tasks, such as geometry changes and foreign object detection are presented to show some of the proposed benefits of the system presented in this work.

1. Introduction

Bridge bearings transfer the loads from the superstructure of bridges (e.g., the deck) to the abutments or intermediate supports, which then transfer these loads to the bridge foundations. Bearings are therefore an integral part of bridge structures and their failure can have considerable impact on the bridge life [1, 2], leading to the overall failure of the entire bridge [3]. It is not uncommon for bridge bearings to be replaced at high costs and disruption (e.g., [4]). Some authors (e.g., [5] and [6]) have shown, through a life-cycle cost analysis, that replacement of bearings due to poor maintenance is significant and can be partially prevented, through appropriate inspection methods. The inspection requirements for structural bridge bearings are detailed in the relevant European Standard [7] as: “close visual inspection without measurements, spaced at equal, reasonably frequent, intervals ”, with inspections occurring at least as often as the bridge structure is assessed. Specifically, the standard requires that the bearings are assessed for visible defects including: cracks, incorrect position of the bearing, unforeseen movements and deformations of the bearing and visible defects on the bearing or surrounding structure.

Most of the main problems affecting bridge bearings are reflected by changes to geometry, regardless of the source of the problem or the type of bearing [8, 9]. These problems include: out-of-position translation, rotation or deformation of the bearing. Current methods to measure changes in the bearing geometry are somewhat rudimentary and involve inaccurate and non-repeatable measurements [9] such as: metric tapes, gap gauges, air bubble levels, quadrant rulers, compasses and verniers, levelling and topographic surveys or direct visual observations. Other, more sophisticated, systems include displacement transducers [10], tell-tales [11] and other instruments that do not measure geometry but measure the actual effect of changes on the bearing or structure directly (e.g., cells and strain gauges [3, 9], fibre optics [12], radar interferometries [13], magnetorheological elastomers [14]), but these are typically outside the norm, with most bridges being inspected via operative-led visual inspection [15].

Other main anomalies in bridge bearings are related to deterioration and degradation of the material itself. Similar to other civil engineering structures, these anomalies typically manifest as cracks, corrosion [16] or crushing [8] that are also visible during visual inspections; such information also has the potential to be extracted from vision sensors [17, 18]. In addition, a visual inspection will also record additional anomalies, such as build up of debris and vegetation growth [9].

* Corresponding author.
E-mail address: h.peel@leeds.ac.uk (H. Peel).

https://doi.org/10.1016/j.autcon.2018.07.003
Received 30 April 2017; Received in revised form 22 June 2018; Accepted 9 July 2018
Available online 17 July 2018

The examples above present inspection solutions for bridge bearings using different types of sensors that reduce the human labour involved and expertise required for visual inspection of the bearings. However, the bridge bearing space is often limited in size and not accessible or hazardous for human access. To this end, robotic platforms mounted with sensors have been deployed for bridge inspection, usually with a focus on structural condition or material degradation [19-21] of the bridge. However, there has been no development of a robotic platform specifically for bridge bearing inspection, where close access to the bearings is required to obtain sufficient detail. In addition, the robotic platforms presented in these examples are manually controlled and are therefore difficult to manipulate if the robot moves out of line-of-sight of the operator. In order to achieve autonomy when performing robotic inspections, a robust localisation approach is essential, especially since errors in the limited bearing space could lead to catastrophic failures such as the robot falling from height.

In this paper, localisation is performed on a robotic platform using a 2D LiDAR as the primary sensor. This proof-of-concept platform is tested in a controlled lab environment and on the cable-stayed Millennium Bridge in Leeds, United Kingdom. This paper focuses on the problem of localisation and mapping since it is critical for any further development of autonomous technology for bridge bearing inspection. The work presented here is not platform dependent and could be incorporated into different configurations with different control systems: a suggested work-flow is given in Fig. 1. In summary, the main novel contributions of this paper are:

- The novel combination of two localisation techniques, namely ACML and Hector-SLAM, to provide a robust localisation of the robot that meets the performance requirements, i.e. less than 10 cm accuracy, see Section 5.5.
- A demonstration of an in situ robotic platform for visual inspections in bridge bearings with two applications: geometry changes of the bearing and detection of foreign objects.

Methods to assess material degradation of bridge structures are not considered in this paper, but visual methods for the detection of cracks (e.g., [22]) and corrosion (e.g., [23, 24]) exist for a range of applications.

This paper is structured as follows: first, the related works on bridge bearing using mobile robots and robot localisation are reviewed and a solution for localising the robot in the bridge bearing environment is proposed (Section 2 and continued in Section 3). A description of the robotic platform and on-board sensors is provided in Section 4, followed by an introduction to the experimental set-up and testing environment. The comparison of the different maps used for localisation is then provided, with validation taking the form of a comparison against a ground-truth in Sections 5.1 and 5.5, for the lab environment and Section 6 for the real bridge site. The experimental results are discussed in Section 6.4, and preliminary inspection results and conclusions are given in Sections 7 and 8.

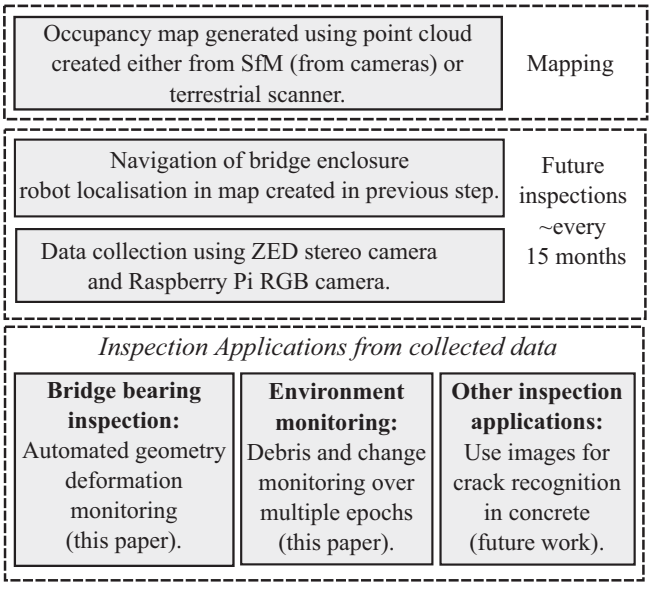

Fig. 1. Overview of how the methods proposed in this paper can contribute to the inspection procedures for a bridge bearing.

2. Literature review

2.1. Robotic inspection of bridges

It is common to use photographs for monitoring the corrosion and structural properties of a bridge [17, 18, 25]. Small cameras can be mounted on robotic platforms, allowing inspection of hard to reach and risky environments. For example, Jahanshahi and Masri [26] obtain depth information from Structure from Motion (SfM) to assist crack detection from photographs of concrete pillars. Torok et al. [27] perform crack detection directly from SfM 3D reconstructions of concrete structures by comparing the normal values of meshes created from SfM point clouds of damaged and undamaged surfaces, with the aim of performing robotic or remote structural assessment in disaster scenarios. The authors of [28] use both SfM and image mosaicing as a method for photo-realistic structural inspection of bridge elements, performed by robotic means. Such vision sensors have been mounted on wheeled robots [20] and legged walking robots [29]. However, the presented systems are bulky and would not fit into a bridge bearing enclosure. In contrast, the authors of [19] present a solution that is small enough to enable passage through narrow spaces. The platform can move on both concrete and steel surface types (including surfaces that had peeled due to corrosion) using six air pads, with air provided by an air supply connected to an air pump and compressor on the ground. The authors also perform testing in a real bridge environment using a CCD camera to inspect the surface of truss members on the bridge as the robot moves along.

Most recently, unmanned aerial vehicles (UAVs) equipped with sensors, such as GPS, gyroscopes and cameras, have allowed large scale inspection of bridges with relative ease. For example, the authors of [21] use a UAV with a top-mounted camera to take images of the under-side of a bridge to be later reviewed by an expert. Similarly, a small UAV for photographic data collection is presented by the authors of [30] in order to perform crack detection of steel bridge members. In both cases, the UAV is unable to get very close to the underside of the bridge, which makes the UAV unsuitable for the inspection of the bearings. In addition, the authors highlight current restrictions surrounding requirements for pilot certification for UAV use and the problems of using GPS for navigation under bridge structures [30].

Overall, technology is developing to allow inspection of structures to be performed remotely, mainly using visual sensors. In addition, the use of robotic platforms and UAVs is allowing the development of inspection methods of structures that are otherwise difficult or dangerous to reach for human inspectors. However, the current development of such platforms for bridge inspection focusses primarily on the platform development. Furthermore, in the reviewed literature, these platforms must be controlled by a human operator. In constrained bridge bearing spaces autonomous inspection is required and therefore, methods for localisation and navigation in these inspection environments should be considered; this topic is the focus of this paper.

2.2. Overview of SLAM and localisation methods

Simultaneous Localisation and Mapping (SLAM) is one particular area of research in robotics where a map is built whilst the robot finds

its position within the map. The SLAM research question has been investigated for many years and includes implementations from underwater inspection of ship hulls [31] to autonomous systems for planetary exploration scenarios [32]. There is almost always a requirement for this task to be completed in real-time as the robot navigates a previously unknown environment.

Localisation refers to the estimation of the time-dependent state of a robot, where the robot state (e.g., position, orientation and velocity) affects the future of the robot [33]. In the simplest case, localisation can be performed using odometry only (commonly wheel odometry, although LiDAR or visual odometry can also be used), with the current position of the robot being determined with respect to an initial position. However, over long periods of time, the position estimate can drift. Data from sensors (such as LiDAR, ultrasound, camera, etc.) and landmarks (such as walls, corners, etc.) can be recognised and used to reduce the drift of the robot's position within the map. The position estimate is improved using techniques such as Kalman Filters. For example, Moore and Stouch [34] implement an Extended Kalman Filter that combines sensor information from multiple GPS and IMU devices to vastly improve on the dead-reckoning position estimation of a robot. Although these approaches allow accurate position tracking, which is important for accurate navigation, they do not necessarily allow relocalisation if the position of the robot is lost. Global localisation (when the robot is placed in the map with no guess for the current location) is possible using grid-based methods, with some examples including Markov Localisation [35] and Monte-Carlo localisation [36].

For inspection applications, however, it is more efficient to use a pre-built map of the environment (rather than building a map from scratch each time), which is then used to direct the robot to a specific area that requires attention. There is still little literature on the implementation of robotic localisation or mapping to civil engineering applications, especially in infrastructure inspection, due to the difficult nature of inspection environments, and none exist for bridge bearings. Advances in automated infrastructure inspection often rely on laboratory-based or simulated environments. For example, the authors of [37, 38, 39] use a mobile robot equipped with a camera to inspect a bridge deck, and use the images to detect deck cracks and to create a map for robot localisation. Although testing in a controlled laboratory environment is essential for development, testing in a real bridge addresses the distinct challenges these environments offer.

Multiple sensors can be used for SLAM and localisation, including: LiDAR, GPS, cameras and ultrasound sensors, where the sensor choice depends on the application at hand. For example, GPS can be used to find the current location of the robot, but can be unreliable when navigating areas where the signal is weak. Since the robotic platform presented in this paper is also used for collecting visual data for inspection purposes, there is a possibility of using the visual sensor data for SLAM. However, there are several factors that need considering for visual SLAM, such as: available frame-rate, processing requirements and image resolution required to perform reliable visual odometry; the effect of changing lighting conditions on the navigation results and the types of motion required when performing the inspection. For example, certain types of visual SLAM (e.g. ORB-SLAM [40]) require a combination of translation and rotation in turning, which may be difficult to achieve in confined spaces. Hence, this method was deemed unsuitable for inspection environments, although it may be possible for use in a combined SLAM approach such as Google Cartographer[1], and this could be considered in future work.

3. Methods

In addition to a range of choice of sensors for SLAM and localisation, there is also an ever-growing number of implementations. Many of the implementations that have been used in outdoor or urban scenarios rely on visual sensors (e.g., RTAB-map [41]) or LiDAR (e.g, Hector SLAM [42]). However, due to the limitations caused by the size of the bearing enclosure, and the complex and changeable nature of the inspection environments, visual sensors are not suitable for navigation in the bridge environment. On the other hand, LiDAR is not limited by the motion of the robot in the bearing enclosure and has existing implementations in such environments. For example, Hector SLAM is a 2D-LiDAR SLAM approach that was originally designed to compete in the RoboCup competitions for robotic post-disaster search and rescue applications in challenging urban environments [43]. Sensor data from 2D LiDARs can also be used to navigate in existing maps. Monte-Carlo Localisation is an established and commonly used method for localisation in a known map and will be used in this work with maps created from point cloud data.

3.1. Hector SLAM

Hector SLAM [42, 43] is primarily a 2D SLAM approach that incorporates 2D LiDAR scans into a planar map. In contrast to other existing SLAM methods (e.g., Gmapping and Rao-blackwell), Hector SLAM does not require any external method of odometry (e.g., wheel encoders), but uses fast scan matching approaches to provide this information. Traditionally, scan matching is done using Iterative Closest Point (ICP) which is computationally expensive. In Hector SLAM however, a fast scan-matching approach is used, which takes advantage of the low distance measurement noise and high scan rates of modern LiDAR [42]. The endpoints of the LiDAR beams are aligned with a map generated by Hector SLAM and a Gauss-Newton approach is used to find the transformation of the current scan that best minimises a cost function to find the correct alignment of the scan to the map. In its current implementation, the map created using Hector SLAM cannot be reloaded for further mapping, although it is possible to save the 2D map and use a localisation only approach such as Monte-Carlo localisation for this purpose. Hector SLAM will be used in this paper for map creation, as an alternative method to AMCL and to provide odometry to the AMCL approaches.

3.2. Adaptive Monte-Carlo localisation

Monte-Carlo localisation [33, 36] uses a Bayes filter; it has a prediction step and a measurement update step. The knowledge that the robot holds about its environment (also known as belief) is represented by a set of particles, where a particle represents a ‘guess' location. If the starting position of the robot is unknown, these particles are spread across the map used for localisation, with each particle representing a potential position and orientation for the robot. Alternatively, a known starting location can be given as an input, around which the particles spread. The first of these cases is referred to as global localisation and the second as local localisation. In the prediction step, the motion commands given to the robot are applied to the particles, updating their position in the map. At the new particle locations, measurements to map landmarks are obtained by the 2D LiDAR and are compared to the expected values from the current particle positions. The set of particles begin with a uniform belief of the true location, but are then re-weighted depending on the probability that a given particle could obtain the current sensor readings in its current location. The particles are then re-distributed for the next time-step based on this probability. The estimated position of the robot at the current time step is centred on the accumulative probability mass of the particles [33]. For a more detailed explanation of Monte-Carlo localisation, refer to [33] and [36].

Adaptive Monte-Carlo Localisation (AMCL) is a version of Monte-Carlo Localisation where the number of particles are adapted over time using Kullback-Leibler Divergence Sampling (KLD-Sampling) to determine the number of particles that are required at each time step (a detailed description of the approach is given in [33]). Particles are

[1] https://google-cartographer-ros.readthedocs.io/en/latest/.

added to the map until a statistical bound is satisfied, where the error between the true pose and the sample-based approximation is less than a fixed value. Changing the number of particles over time allows better computational efficiency, since fewer particles are required if many particles have a similar belief about the robot's position. The number of particles required tends to decrease over time as the robot moves around the map and the belief regarding the current position improves. The implementation of AMCL used in this paper is an existing ROS package[2] based on the work in [33].

ROS is an open-source framework consisting of many libraries, packages and tools that allows the control of low-level sensor drivers. Odometry and some sensor input are required for AMCL to localise in a map provided by the user. In this paper, odometry is provided from the Hector SLAM ROS package (the mapping functionality of Hector SLAM is disabled) and the raw data from the 2D LiDAR is used to update the measurement model in AMCL.

In summary, Hector SLAM and AMCL were selected because:

- Their implementations can be performed using the same, easily accessible, type of sensor, i.e. a 2D-LiDAR.
- They use relatively low computing power, i.e., they can be implemented on the Raspberry-Pi and the Nvidia Jetson TX1.
- They have been demonstrated in urban environments (e.g., [42-44])

3.3. Maps for localisation

In robot navigation, maps are often represented in either a topological format, where the map is decomposed into significant places in the environment, or a metric representation, where the map is decomposed into fine-grained cells of uniform size [33]. One common form of the metric representation is an occupancy grid, where grid cells contain a probability that each cell is occupied by an object in the real world. If the probability that a cell is occupied by an object is high, the cell is filled in black, whilst white shows free space and grey is unknown as it is outside of the mapped area, i.e., areas out of the sensor range. This approach prevents transient objects, such as people, from affecting the map with artefacts.

Monte-Carlo Localisation requires an occupancy map in which to perform localisation. These maps are often created using the same sensors that will also be used for navigation, but they can also be created using other approaches. For example, the authors of [45] successfully perform localisation in simulated environments using hand-drawn maps and Monte-Carlo Localisation in areas where no accurate maps were available. This scenario is similar to the inspection environment, where it may not be possible to create maps using sensors on-board a robot. An alternative approach to obtaining maps for localisation in the inspection environment is described in Section 5.3.

4. Robotic platform

The robotic platform used in this work is a commercial product called a DiddyBorg (a six wheeled robot with a Perspex chassis, that is built around the Raspberry Pi single-board computer – see Fig. 2), which has been modified to accommodate additional sensors and hardware and is approximately 25 × 18 × 23 cm in size. The sensors mounted on the platform include: a RPLiDAR (a 2D LiDAR of size of 7 × 10 × 5 cm with a range of 6 m) mounted on top of the platform, a single 8 MegaPixel RGB camera (Raspberry Pi Camera V2) and the ZED Stereo Camera which can used from VGA at 60 frames-per-second (fps) to 2 K at 15 fps. Both cameras are used for inspection applications (see Section 7). The on-board processing, required for sensor operation and data collection, is provided by the Raspberry Pi and NVIDIA Jetson TX1 (TX1) which was added to the platform to allow real-time visual processing.

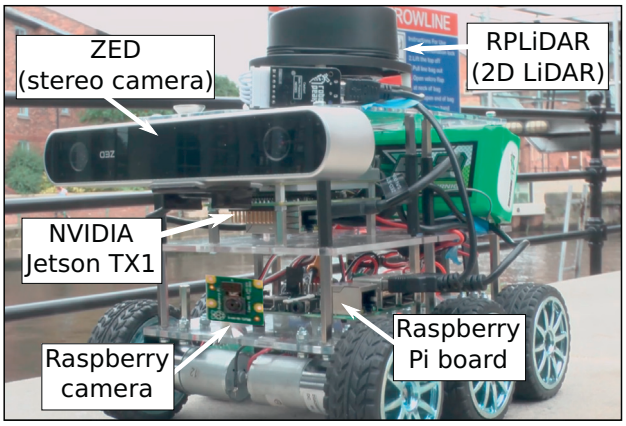

Fig. 2. A photograph of the modified DiddyBorg robotic platform used in this work, with the relevant on-board sensors labelled in the image.

In addition, the Robot Operating System (ROS) is used to facilitate data collection. Sensor data is transferred as messages to different machines (in our case, the Raspberry-Pi, TX1 and a laptop) for processing using a wide variety of open-source algorithms or for recording for later use. An overview of the system architecture is given in Fig. 3.

5. Experimental set-up of the laboratory environment

5.1. Description

The lab environment used for testing of the localisation approaches is an unfinished, domestic structure. This environment has similar textures to the bridge environment and similar lighting conditions, with bare concrete walls, naturally changing lighting conditions and dark or unlit areas. The testing area is similar in size to the bearing enclosure used in this work (approximately 2.5 m by 1.35 m). In addition, a mock-up of a bearing was created to allow testing for measuring the geometry of a bearing using the inspection methods described in Section 7.1.

5.2. Data collection in the testing environment

A map of the environment is required to be able to direct the robotic platform to specific areas in the bridge bearing environment. To compare the different mapping approaches for localisation, the robot was moved around the test enclosure to collect 2D LiDAR data using pre-programmed velocity commands. A ground-truth trajectory is determined from the robot odometry (provided by Hector-SLAM) and velocity commands. Repetitions of the trajectory were performed to test its accuracy and repeatability. The 2D LiDAR data is then post-processed for use in mapping and AMCL. In the lab environment, the maps tested for localisation with AMCL are:

1. A map saved from Hector SLAM.
2. A map created from a point cloud.

To create the first map, the 2D LiDAR data is processed using Hector SLAM, and the finished map is extracted and saved. The second type of map is created from a point cloud, which can either be produced using the 3D reconstruction method Structure from Motion (SfM) or using a 3D terrestrial laser scanner; in the lab enclosure a map is produced using SfM.

To obtain the data for SfM, photographs are first collected by teleoperating the robot around the test enclosure and taking photographs at multiple locations using the approach shown in Fig. 4. To ensure sufficient image overlap for the reconstructions, the robotic platform is rotated on the spot by a small increment, then stopped before capturing

[2] http://wiki.ros.org/amcl.

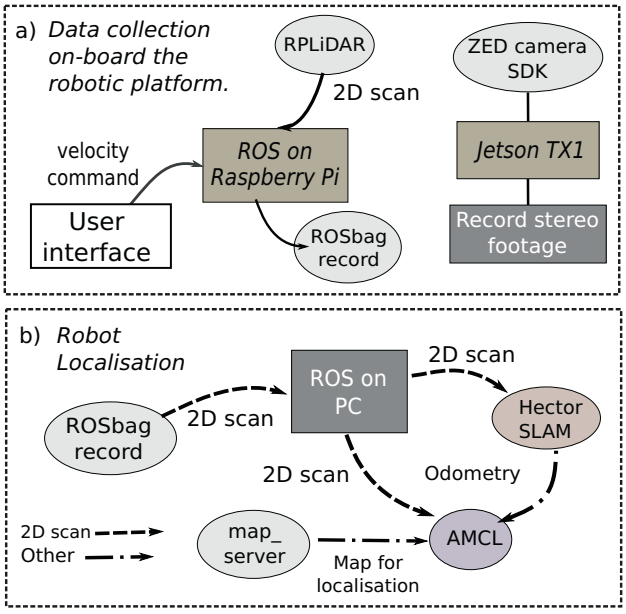

Fig. 3. a) An overview of the tasks completed by hardware and software on-board the robot in data-collection. b) An overview of the different types of data collection and processing used with the localisation methods in this work.

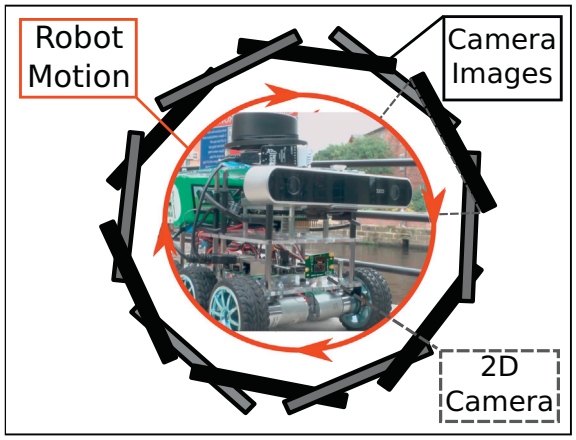

Fig. 4. The method used by the robot to collect photographs for SfM reconstructions. The figure does not represent how many photos were taken through this process, only the manner in which the photos were collected.

the next photo (see Fig. 4). This method may not be optimal for SfM reconstructions in general, however it is required when space is limited such as in bearing enclosures. The photographic data is then processed using SfM software to create a dense 3D point cloud of the enclosure and scaled using known measurements. The resulting point cloud is then made into the map (see Section 5.3).

5.3. Creating a two-dimensional map for robot localisation

In this work, an occupancy grid map is created in a binary format (i.e., a cell is either occupied or not and no probability information is stored) using the Cartesian coordinates of points from the input point clouds.

The steps performed to create the map for AMCL are as follows (also depicted in Fig. 5): the point cloud is cropped to the area of interest around the bearing enclosure using CloudCompare[3]; the floor level of

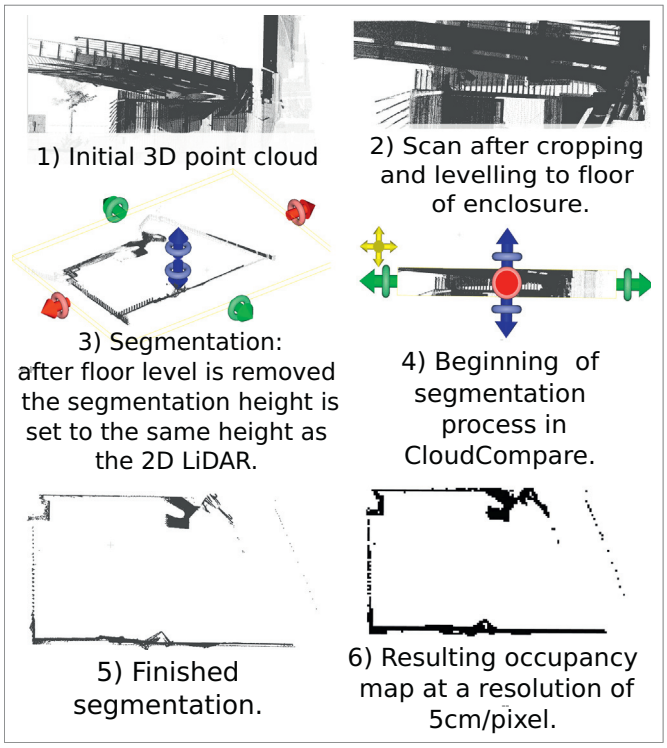

Fig. 5. An overview of the steps for processing a 3D point cloud (either from SfM or a 3D terrestrial LiDAR) into a 2D occupancy map for use in 2D localisation using Adaptive Monte-Carlo Localisation.

the point cloud is determined using the levelling tool in the software by manually selecting points that are at floor level; from this level, a slice through the point cloud is extracted (using the segmenting tool) corresponding to the location of the 2D LiDAR on the robot. The slice from the point cloud is three-dimensional, with two dimensions corresponding to the plane of the sensor data collected by the 2D LiDAR and a third dimension, of approximately the same height as the 2D LiDAR sensor, in order to ensure that the majority of points that could be detected by the 2D LiDAR are included in the map. Finally, a binary occupancy grid is extracted from this point cloud slice using the BinaryOccupancyGrid function in the MATLAB Robotics Systems Toolbox[4].

An occupancy map is split into uniform grid cells representing locations in the real-world. It is possible to vary the resolution of the occupancy maps by varying the grid size, e.g., to 1 cm /px or 1 mm /px (highest available resolution for this method). For Hector SLAM, these maps had a resolution of 5 cm/pixel, so that each grid cell represents 5 cm in the real-world (higher resolutions were tested, but gave noisy results that were unsuitable). Initially, as previously mentioned, the slice taken from the point cloud data is 3D, and some method is required to represent this slice in the 2D occupancy map. A grid cell in the map can be considered occupied if a point from the point cloud is present in the grid cell. Since the grid cell can only be set as occupied or unoccupied, multiple points in the same grid cell are discarded to give a 2D map. The effect of varying the resolution of the occupancy maps created from point clouds will be discussed further in Section 6.4.

5.4. Scaling the SfM point cloud

The 3D reconstruction software Zephyr-Aerial (Zephyr)[5], produced by the company 3Dflow, is used to produce point clouds from images.

[3] http://www.cloudcompare.org/.

[4] https://uk.mathworks.com/products/robotics.html.

[5] https://www.3dflow.net/.

The input to the software is raw RGB camera images collected by the robotic platform and the camera calibrations are performed automatically by Zephyr. The first output from the software is the SfM point cloud – a sparse point cloud. More details can be added to the sparse, scaled, point clouds using Multi-View Stereo (MVS), to create a dense point cloud.

It should be noted that the sparse point clouds generated by SfM are at an arbitrary scale; therefore some method of scaling is required to recover the global scale of the point cloud. To scale the point cloud, control points are selected from the photographs used for the SfM reconstruction using the control point selection software in Zephyr. Using this software, multiple instances of the control point in different images are selected. Each control point must be selected in a minimum of two image views [46], although accuracy of the control point location improves if more images are selected.

Visually distinguishable features, such as the corners of the bearing pads, were selected as control points. The world-scale was recovered using measurements taken manually with a tape measure, with three-dimensional co-ordinates from a reference point required for each control point. Control points were selected from regions across the area being reconstructed, with approximately eight control points being used in each case. It is also possible to provide scale to the point clouds using dimensions from construction drawings. However, it can be challenging to match these dimensions with visible features from the photographic dataset. Scaling can also be performed directly by matching points in the SfM cloud with points in the ground-truth laser scan, but this is only possible if laser scan data is available. The effectiveness of this method for scaling the SfM point clouds in validated against 3D terrestrial LiDAR data in Section 6.3.

5.5. Results and discussion

The trajectories resulting from AMCL when localising using the map created by Hector SLAM (referred to henceforth as AMCL-Hector) are plotted against the trajectories from AMCL when localising in the map created from a SfM point cloud (referred to henceforth as AMCL-SfM), and both trajectories are compared to the ground-truth trajectory of the robot and to the trajectory calculated by Hector SLAM (Fig. 6 for local, Fig. 9 for global). Global and local forms of AMCL are considered, as three repetitions of each test (Fig. 8). For local AMCL, an initial guess for the position and orientation of the robot is provided by a user-input as a Cartesian map coordinate, and orientation is provided as an angle with respect to the origin of the map. For global AMCL, particles are sampled across the full map and no additional input is provided to calculate the initial position of the robot. Furthermore, the spread (i.e., confidence) of the particles in the AMCL calculations are visualised in Fig. 6 by plotting ellipses representing the covariance of the robot pose at each point in the robot's trajectory and also graphically in Fig. 7b, where a small ellipse represents a small spread of particles in the AMCL calculations.

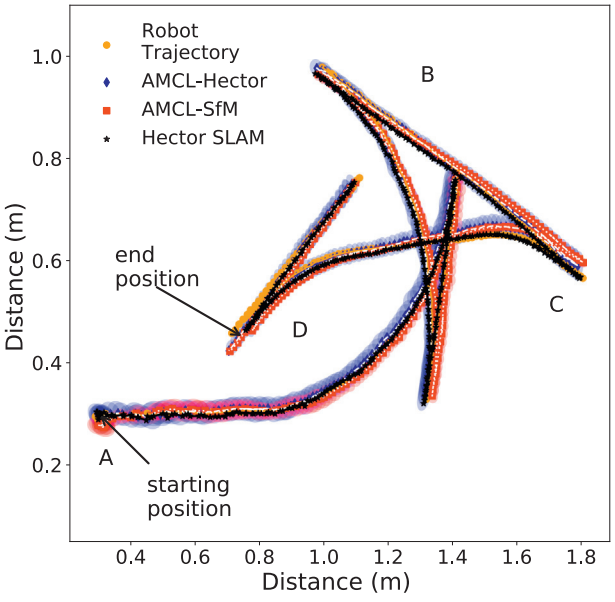

Fig. 6. A comparison of the trajectories calculated using local AMCL-SfM and AMCL-Hector, the trajectory calculated by Hector SLAM and the ground-truth robot trajectory. The ellipses that represent the covariance of the pose calculated by AMCL-Hector and AMCL-SfM are also plotted at each step in the trajectory.

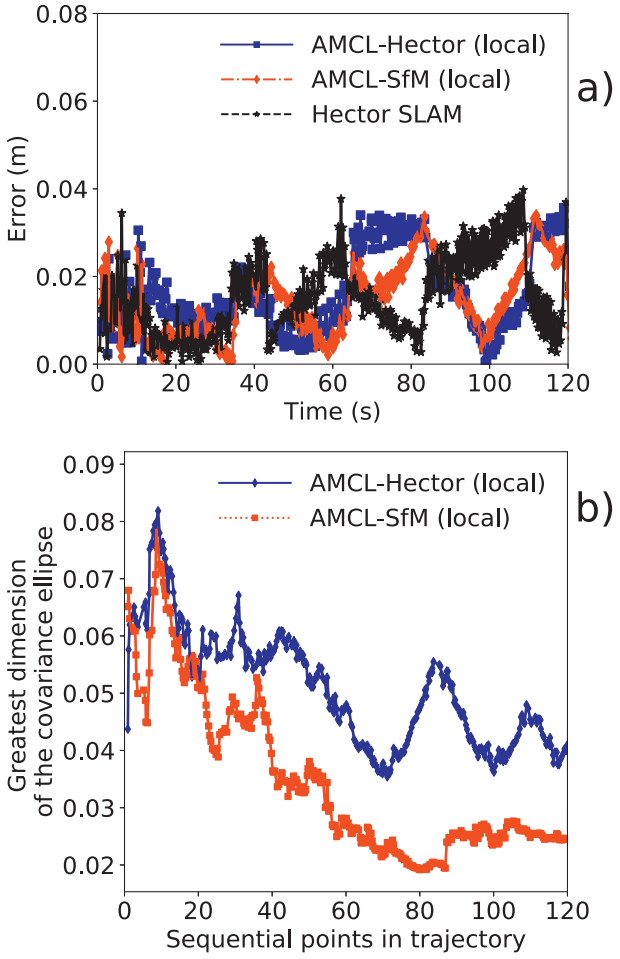

Fig. 7. Panel a): The error calculated between the ground-truth trajectory of the robot and the trajectories calculated by both local AMCL-Hector and AMCL-SfM for each point in the ground-truth robot trajectory. Panel b): The covariance of the AMCL particles over the duration of the trajectory, as calculated from the greatest dimension of the covariance ellipses seen in Fig. 6.

In Fig. 6, the initial errors for the local AMCL trajectories and the ground-truth trajectory (errors plotted in Fig. 7a) are less than 3 cm for all methods. Over several repetitions (see Fig. 8) this initial error is as large as 10 cm for AMCL-SfM and 8 cm and AMCL-Hector (both in the third repetition – see Fig. 8), where there is an error in the initial position guess of approximately 4 cm, which causes recalculation of the robot's position over the first 10 time-steps before converging to the ground-truth. Once the robot begins to move, the error remains below 4 cm for all AMCL methods for the remainder of the trajectory. This error is comparable to Hector SLAM for all repetitions, which also has a maximum error of 4 cm. Slight peaks in the error value occur at points where the robot changes direction, such as the regions labelled B and C in Fig. 6.

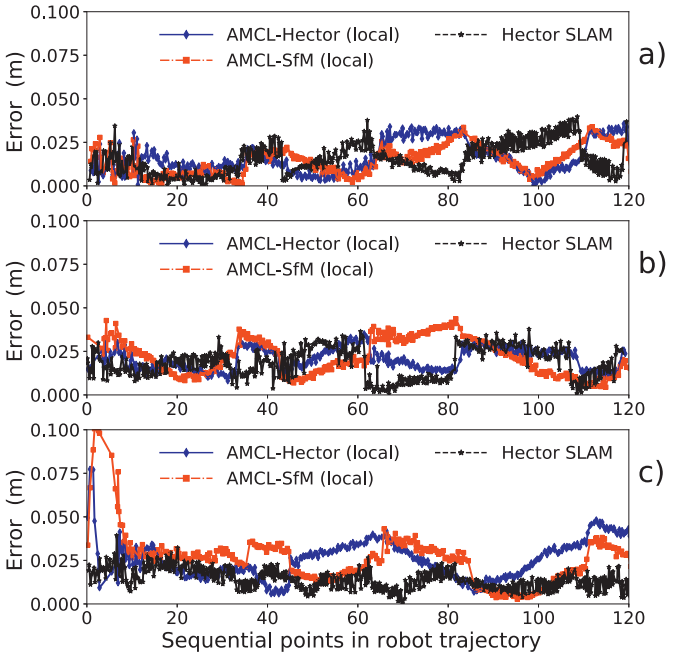

Fig. 8. A comparison of the error between local AMCL-Hector and AMCL-SfM when compared to the ground-truth robot trajectory for three repetitions of a set of input commands to the robot.

The uncertainty of the AMCL particles for local AMCL, is double the initial trajectory error with values around 6 cm in Fig. 7 (3 cm either side of the robot). The spread of particles is initially large to ensure calculation of the correct position and the uncertainty increases. Once this position is found, the uncertainty and the number of particles reduces and the covariance value fluctuates around 5 cm for AMCL-Hector and 3 cm for ACML-SfM.

Next, the difference between global and local AMCL is considered. As expected, the covariance of the particles is higher for global AMCL than for local AMCL (see Fig. 7b and Fig. 10b) since the particles begin spread across the map. In Fig. 9 and Fig. 10a the error between the global AMCL-Hector and the ground-truth trajectory starts at 1.25 m,

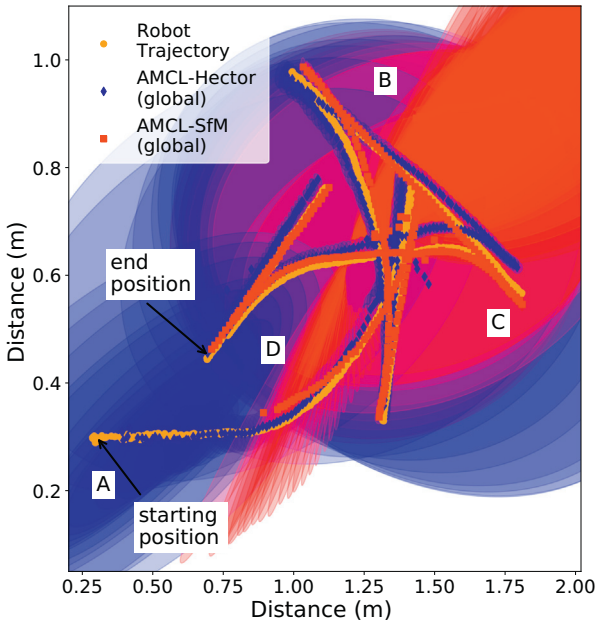

Fig. 9. A comparison of trajectories from AMCL-Hector and AMCL-SfM with the ground-truth robot trajectory for the global initialisation of AMCL.

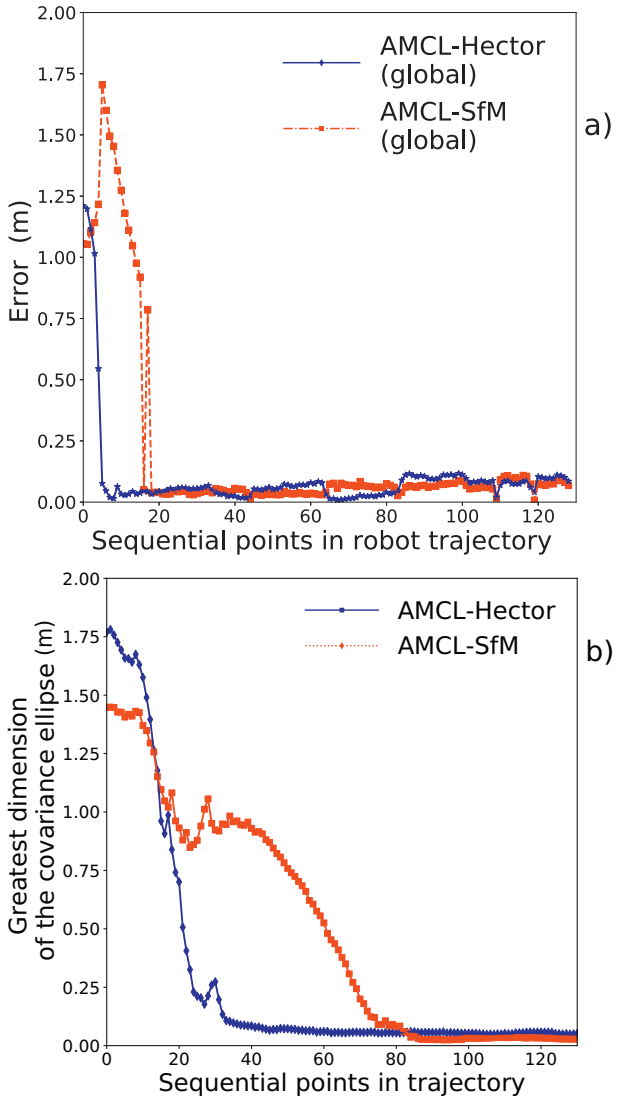

Fig. 10. a: The error calculated between the ground-truth trajectory of the robot and the trajectories calculated by both AMCL-Hector and AMCL-SfM when global localisation is implemented. b): The covariance of the AMCL particles over time, as calculated from the greatest dimension of the covariance ellipses seen in Fig. 9b.

with an initial guess being found at the centre of the map. The current position is recalculated and the trajectory begins to converge to the ground-truth. After seven time-steps the error approaches a value similar to the error seen in the local localisation approaches. Similarly, AMCL-SfM also begins with a large error of 1.1 m, which increases as the robot begins to move before converging to the ground-truth after 20 time steps. The initial location calculations can be seen to the right of label B in Fig. 9 as the ellipses move from the top right of the figure to the bottom left where the true starting pose is. Although the error in the robot position for AMCL-SfM converged after 20 time-steps, it took 80 time-steps for the covariance to converge (compared to 30 time-steps for AMCL-Hector, see Fig. 10). Due to the large errors in the initial time-steps, global localisation will be discounted at present, as it is not suitable for the bridge bearing environment.

The results in the laboratory environment showed that all local AMCL and Hector SLAM trajectories gave an error within 4 cm of the ground-truth. However, in the real bridge environment there is danger that the robot could fall from height. Therefore, some maximum bound of the error is required to reduce the risk of failure, especially since the maps in the bridge environment are more noisy than in the lab

environment. For the purposes of this work, this bound is defined as 10 cm, which is approximately the diameter of one of the robot's wheels, and it is anticipated that near edges in the bearing enclosure the robot could not recover from an error greater than this value. A maximum error of 10 cm is an error of 8% in the shortest dimension of the test enclosure and 3.6% in the longest dimension of the test enclosure.

Next, to understand whether such an error bound is achievable in practice, AMCL is tested in the bridge environment using the same system as tested in the lab environment. An additional map (created from 3D terrestrial LiDAR scans) will be introduced as another option for map creation. Again all results from AMCL are compared to Hector SLAM and the ground-truth robot trajectory.

6. Experimental set-up of the bridge environment

6.1. Description of the bridge environment

The bridge and associated bearings used this work are located at the The Millennium Bridge Leeds, UK. The Millennium Bridge is a cable-stayed footbridge crossing the River Aire, spanning approximately 57 m. The bearings on the north side of the river are studied in this work and are located in a space approximately 2.8 m by 1.2 m, which has a total height of 0.4 m. The layout of this site gives an enclosed region to be referred to as the bearing enclosure, see Fig. 11.

6.2. Data collection in the bridge environment

In all cases, only the local implementation of AMCL is considered as it proved to be more accurate than global localisation in the lab environment. The maps used for AMCL calculations:

1. Map created using Hector SLAM.
2. Map created using SfM.
3. Map created using 3D terrestrial LiDAR (map creation process identical to map created from SfM in Section 5.3). AMCL results using this method will henceforth be referred to as AMCL-LiDAR.

The results of AMCL-SfM and AMCL-LiDAR will also be considered for two different map resolutions: 5 cm/pixel and 1 cm/pixel. Examples of the different maps tested in the bridge environment are given in Fig. 12. Again, the results of AMCL are compared to the trajectory calculated by Hector SLAM. Sensor data is collected in the same manner as described in Section 5.2, with the robot traversing a pre-set trajectory from a given set of velocity commands.

6.3. Validation of SfM data against 3D terrestrial LiDAR data

In the bearing enclosure, point cloud data is also available from a 3D terrestrial laser scanner, the RIEGL VZ-1000. Detailed scans of the bearing enclosure were taken from three different perspectives and the clouds were registered using targets with a registration error of less than 3 mm. CloudCompare is then used to compare the SfM and 3D terrestrial LiDAR point clouds. The SfM and terrestrial laser scan point clouds are aligned by selecting common features that are visible and easy to select by eye. Iterative Closest Point (ICP) alignment is used to further align the point clouds. A cloud-cloud distance calculation allows the comparison of a point cloud to a reference point cloud, where the distance of points will be calculated relative to associated points in the reference cloud. A scalar field is generated showing the number of

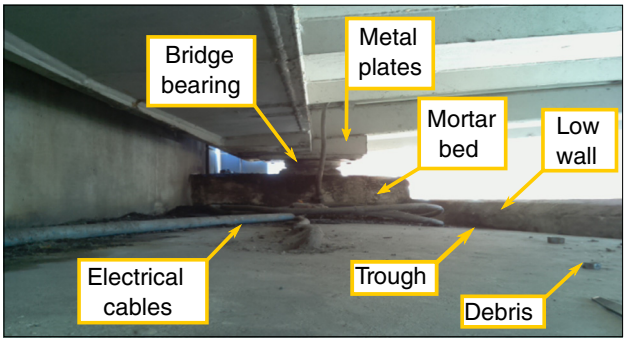

Fig. 11. Photograph of the bearing enclosure at Millennium bridge, on the north-side of the river. One of the bridge bearings is visible in the centre of the image. The low wall that surrounds the enclosure is visible on the right-hand side and some electrical cables on the left-hand side of the image.

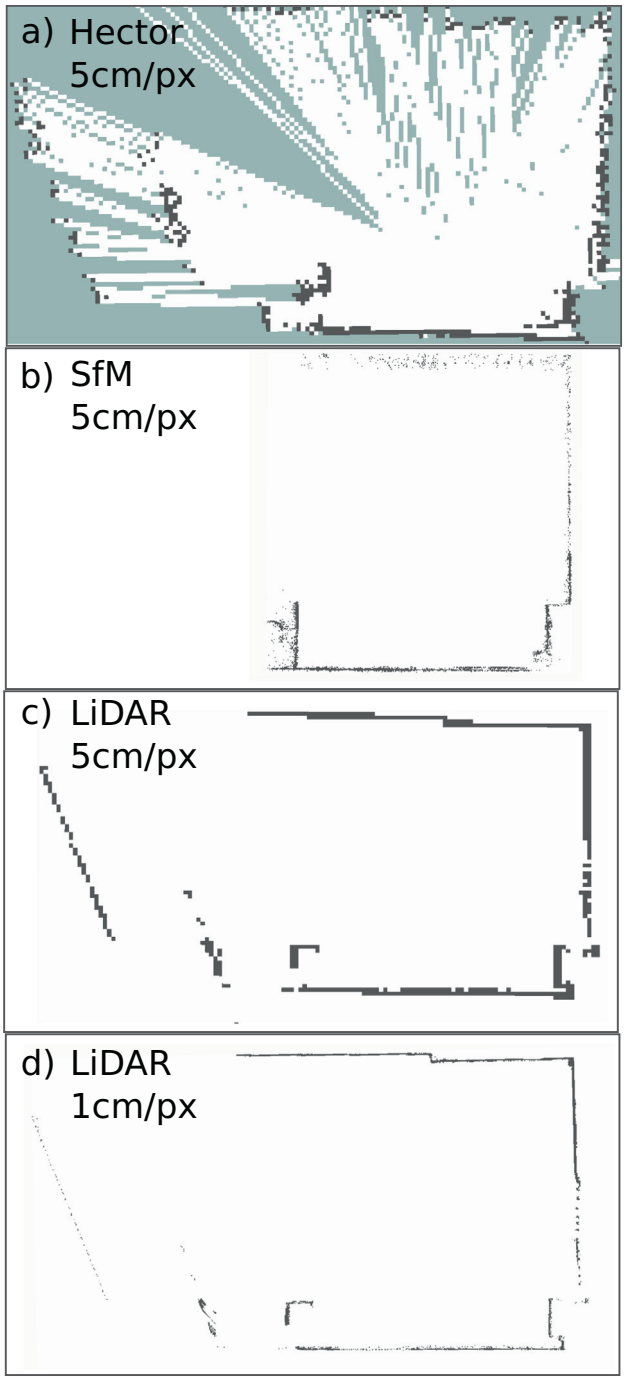

Fig. 12. a) 2D occupancy map created using Hector SLAM. b) 2D occupancy map created using SfM c) 2D occupancy map at 5 cm/pixel created from 3D terrestrial LiDAR point cloud. d) 2D occupancy map at 1 cm/pixel created from 3D terrestrial LiDAR point cloud.

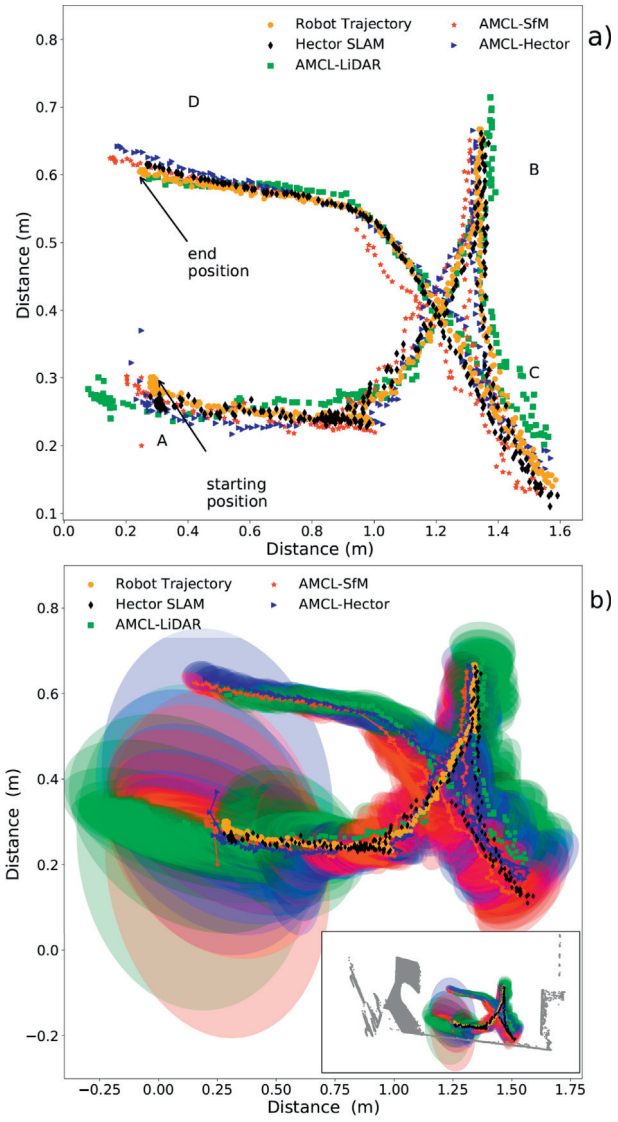

Fig. 13. a): A comparison of the trajectories calculated using AMCL in the bridge environment for three different maps (maps created using either SfM, 2D LiDAR data and Hector SLAM). b): The same trajectory as in a) overlaid with ellipses representing the covariance of the pose at each step. In the bottom right of b), map points from the terrestrial 3D LiDAR scan are shown with a zoomed-out view of b) to give a sense of scale of the trajectories relative to the bearing enclosure.

points and the distance of those points compared to the points in the reference cloud. 90% of points from the SfM point cloud are within a distance of 4 cm from the terrestrial 3D LiDAR point cloud and 60% of points are within 1 cm of the terrestrial 3D LiDAR point cloud. This result gave confidence that the SfM map was accurate and suitable for map creation. Furthermore, using two sets of maps gives an indication of whether localisation will be affected by scale. Further discussion on this point can be found in Section 5.5.

6.4. Results and discussion

In the bridge environment, variation from the ground truth in the robot trajectory and the associated uncertainty in the position can be seen in Fig. 13, with most variation seen in the areas labelled D and C in Fig. 13a. Similarly, the associated trajectory errors in the real bridge environment are greater than the laboratory results, with values varying between 2 cm and 10 cm. The maximum value for covariance is

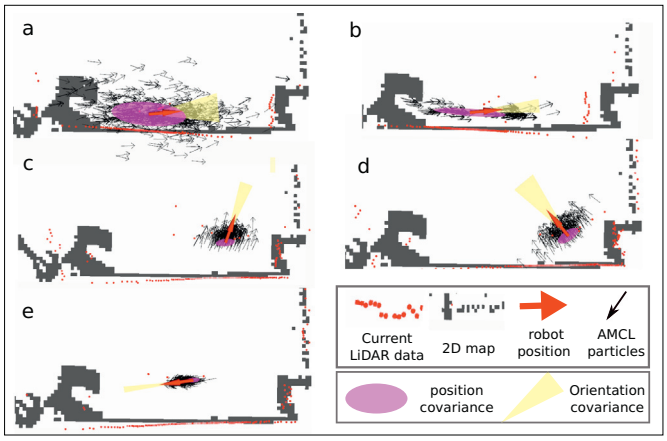

Fig. 14. The pictorial representation of AMCL-LiDAR plotted in RVIZ (ROS visualisation software) showing the robot position within the 2D LiDAR map. The covariance of the robot pose and robot orientation varies as the robot moves around the map. Panel a) shows the robot position shortly after initialisation of AMCL-LiDAR with particles widespread around the current guess for robot position; panel b) shows the AMCL particles in two small groups, but with misalignment in the current LiDAR data and the map there is uncertainty in the current robot position; panel c) shows the particles converge to one position and an improvement in the alignment of the current LiDAR data and the map; panel d) shows an increase in the covariance for robot position and orientation as the robot turns; panel e) is towards the end of the robot motion and shows that AMCL-LiDAR is converged on the current robot position.

also greater than in the lab environment, with values between 20 cm and 1 m, which covers a large portion of the bearing enclosure in Fig. 13b. Again, the largest error and covariance occurs after initialisation of the particles, with the largest error value of 9 cm produced by AMCL-SfM, and the largest covariance value of 1 m, produced by all methods. The error is slightly greater due to uncertainty in the initial position given to the robot, but this error converges to within 3 cm of the ground-truth value after three time-steps. Errors in the initial position guess occur due to the difficulty of measuring the location by hand in the bridge environment. However, in practice a docking station could be used to house the robot and create a repeatable starting location.

The initial error for AMCL-LiDAR is lower than AMCL-SfM, but slightly greater than AMCL-Hector and Hector SLAM (5 cm, 2 cm and 4 cm respectively). In addition, the covariance for AMCL-LiDAR takes the longest to converge. This initial uncertainty is visible in Fig. 14a, where the particles are spread over the majority of the bearing enclosure and in Fig. 14b appear to split into two groups of particles representing the robot position. However, by Fig. 14c the covariance has decreased to 20 cm (i.e., between the areas labelled A and B in Figs. 13 and 15). One potential reason for the greater uncertainty for AMCL-LiDAR is that the available detail in the original point cloud is greater than for the 2D LiDAR. Therefore, some features have been included in the map that are not representative of what is visible for the 2D LiDAR, and as a result, the AMCL particles group in the true position and another position, but as the robot begins to move, it becomes apparent which is the correct position and the certainty and error decreases.

An increase in error is observed at the points labelled B and C in Fig. 15 (time-steps 60 and 100 in Fig. 15) as the robot turns towards the less featured regions of the map (top left region of figures a–d in Fig. 12). At these points, potential features in the map are at a distance close to the maximum sensor range of the 2D LiDAR (6 m). When comparing the maps in Fig. 12 a–c, it is apparent that the sensor data from the 2D LiDAR can reach some of regions of the map but not others, which contributes to a greater error in the robot position. Although the increase in errors are similar for all methods, the error for AMCL-SfM does not recover as well as the other methods towards the end of the

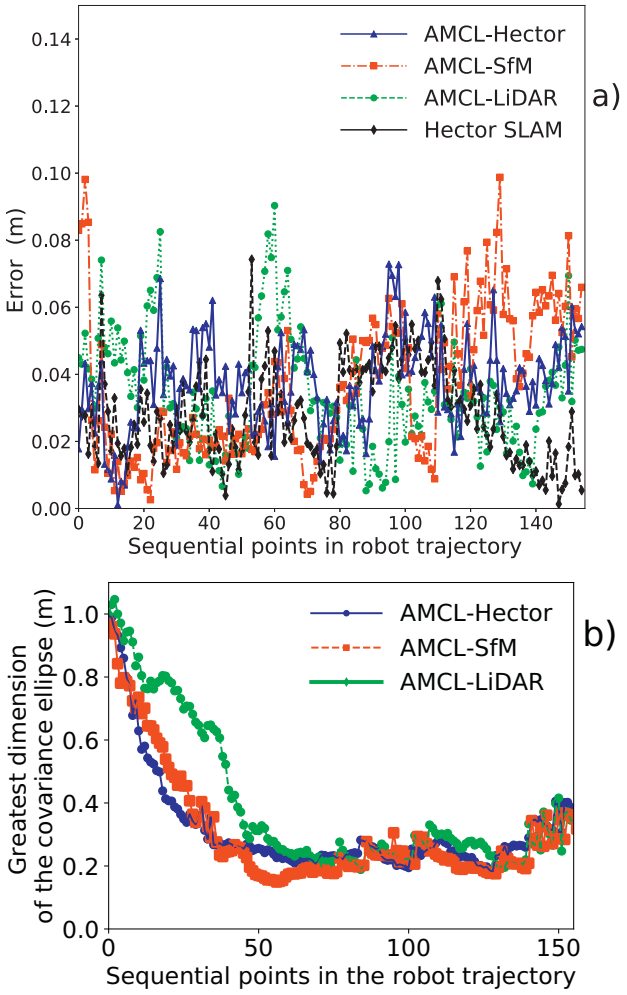

Fig. 15. a: The error calculated between the ground-truth trajectory of the robot and the trajectories calculated by AMCL-Hector, AMCL-LiDAR and AMCL-SfM in the bridge environment, where the error is calculated for each point in the ground-truth robot trajectory. Panel b): The covariance of the AMCL particles over time for the same approaches as in a), with the covariance value being calculated from the greatest dimension of the covariance ellipses seen in Fig. 9b.

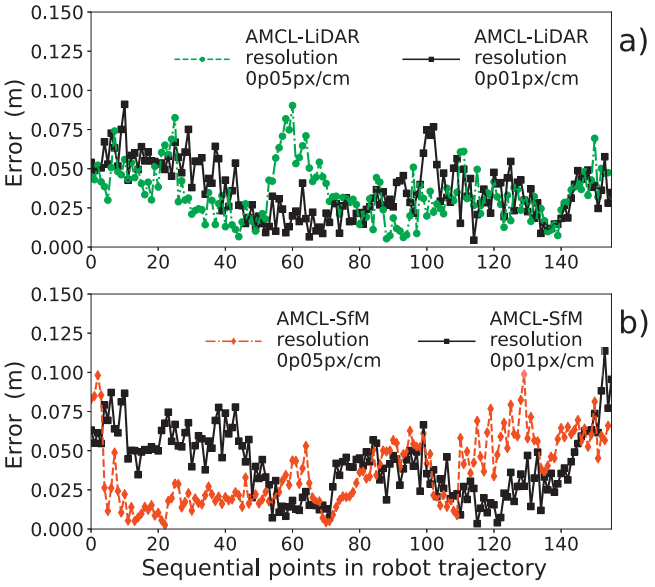

Fig. 16. The error calculated between the ground-truth trajectory of the robot and the trajectories calculated by AMCL-LiDAR and AMCL-SfM in the bridge environment, where the error is calculated for each point in the ground-truth robot trajectory. Map resolutions of 1 cm/pixel and 5 cm/pixel are compared for both methods.

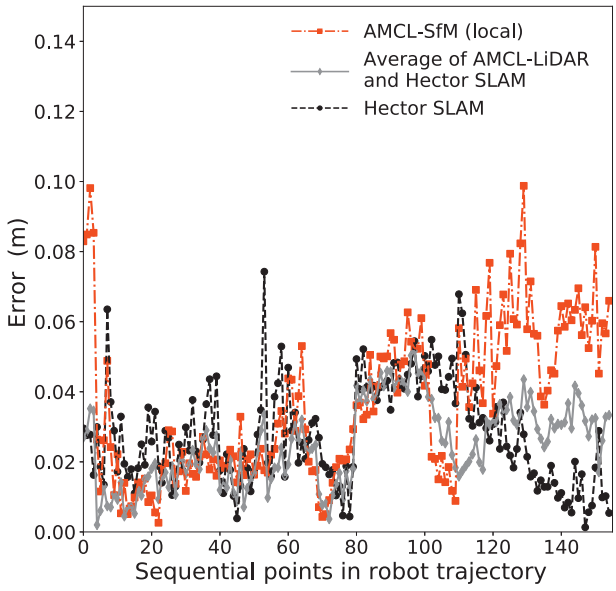

Fig. 17. The error calculated in the bridge environment between the ground-truth trajectory of the robot and the trajectories calculated by AMCL-SfM, Hector SLAM and a trajectory determined from the average of the two trajectories.

trajectory at point D. In addition, the error at D for AMCL-SfM is higher when the map resolution is increased to 1 cm/pixel. The map for AMCL-SfM in these regions is noisier, but also the sensor data does not overlap well, suggesting that the point cloud scaling is worse in these regions. AMCL-SfM has the least detail in this region since the image features used to create the SfM point cloud are far away are not as clear as nearby features.

Otherwise, there appears to be little difference in the errors produced from AMCL-Hector and AMCL-SfM when increasing the map resolution to 1 cm/pixel (Fig. 16). For AMCL-SfM, the error converges quicker for the lower map resolution of 5 cm/pixel, but the error for both resolutions of the AMCL-LiDAR maps match well. It is therefore recommended that the map resolution of 5 cm/pixel is used for creating the map as it allows some smoothing of noisy regions, but provides adequate detail for localisation.

For the majority of the trajectory recorded in Figs. 13 and 15, the error with respect to the ground-truth is equal to or below the desired bound of 10 cm. However, for the method with the largest trajectory error (AMCL-SfM) the error becomes greater than this bound. The method with the lowest trajectory error is Hector SLAM, and the associated error is below this threshold at all times. However, since a known map is required to give direction for inspection, the output from Hector SLAM is combined with a AMCL-SfM by taking the average of the trajectory coordinates from Hector SLAM and AMCL-SfM and plotting the error of this new trajectory with respect to the ground-truth trajectory (Fig. 17). By combining the two approaches, it is observed that the averaged trajectory maintains a value below 6 cm for all time. In addition, for the greatest error produced by AMCL-SfM (10 cm), the equivalent error for the averaged trajectory is 4 cm and when Hector SLAM spikes to 6 cm at time-step 50 in Fig. 17 the error for the averaged trajectory remains below 4 cm. This solution is recommended to provide robustness in the solution, particularly as a check when the error for the trajectory calculated by AMCL approaches the 10 cm error bound. In addition, if one of the individual methods fail, this solution

provides redundancy to reduce the risk of failure of the entire system.

7. Demonstration of inspection applications

Although localisation of the robotic platform in a bearing enclosure is the main focus of this work, two inspection applications are considered here that address the specific requirements discussed at the beginning of this paper, namely geometry monitoring and foreign object detection and show a specific use-case for the robot.

7.1. Inspection application one: geometry monitoring of the bridge bearing

Stereo-pairs from the ZED camera are used to obtain 3D information of points in the images and therefore monitor the geometry of the bridge bearing. The work-flow for a proof-of-concept method for automating bridge-bearing monitoring is presented here.

First, a 2D image (left view) of the bridge bearing in the stereo pairs is selected. During pre-processing, a Gaussian filter is applied to the image; a Canny edge detector is then used to detect edges of the bearing and a Hough line transform is applied to detect straight lines in the image. The profiles of the bearing can then be obtained, from which 12 points are selected to determine the geometry of the bearing. These points are the edge points of each edge of the bearing as shown in Fig. 18a.

Secondly, the 3D locations of the 2D points in the 2D image can be obtained using the ZED SDK. With such 3D locations, the dimensions of each bearing edge can be obtained by computing the distances between the mark points. The estimated dimensions are compared with ground truths and shown in Fig. 18b.

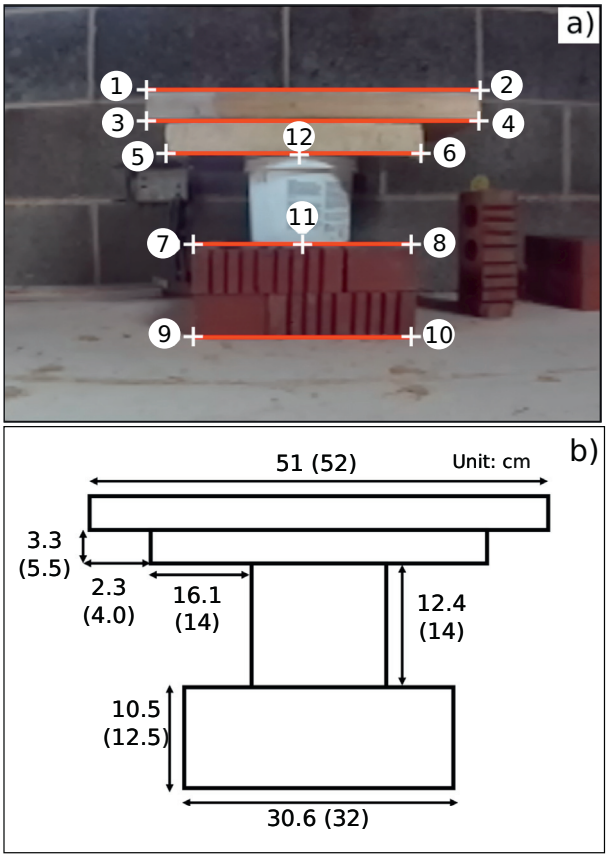

Fig. 18. a) The twelve mark points selected for calculating the bearing dimensions. The edges of the edges are detected by Hough transform and shown in red. The corner points of the bearing edges are selected and shown as crosses and numbered from 1 to 12. b) The estimated bearing dimensions against the ground truths (listed in the parentheses). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

From Fig. 18b, it can be seen that although most of the estimated dimensions of the bearing are similar in magnitude to the ground truth values, they are not accurate enough to use for inspection purposes. The main reason is that the 3D locations of the data points are not well estimated in the ZED SDK. It is expected to achieve better geometry measurement results by employing more accurate depth computing algorithms, which will be explored in future work. Another reason is that the corner points of the bearing profile may not be accurately detected. This is caused by the distortion effects of the stereo camera as shown in Fig. 18a and inaccurate edge/corner detection. In the future work, better preprocessing methods such as distortion correction and filtering are expected to be included to improve the measurement accuracy. By monitoring the changes in the bearing dimensions over time, problems such as compression, rotating and moving out of position of the bridge bearing, can be mitigated against before great damage is caused to the bridge. In addition, if all proper considerations are taken into account (e.g., the effect of changing light conditions and camera perspective), this system will provide a more repeatable and comparable solution than current visual inspection methods.

7.2. Inspection application two: foreign object detection

One advantage of using cameras to perform the inspection of bridge bearings is that extra details can be extracted that would otherwise be difficult to record by hand. One example is detection of foreign objects around the bearings, such as vegetation, rodents or litter. Debris and vegetation is listed as a type of anomaly to be recorded when performing inspections of bridge bearings [9]. In this section, a demonstration of change monitoring in the bearing enclosure using SfM is performed.

7.3. Debris monitoring

To show the process of change monitoring, an object is artificially introduced into the bearing enclosure. Once the object is introduced, another dataset for SfM reconstructions are collected using the robotic platform. This dataset is processed in the same manner as for the initial stages of the map creation process in 5.3, and again the resulting dense point cloud is loaded into CloudCompare, but in this instance only a Cloud-Cloud distance calculation is performed. The new point cloud is compared to the terrestrial laser scan in Fig. 19. A heat map shows the differences in distances between the two point clouds and the debris is clearly visible. Some regions in the heat map show large differences between the point clouds that are unrelated to the foreign object. These differences are due to gaps in the terrestrial scan point cloud caused by occlusion when collecting the data since the scanner was unable to fit inside the bearing enclosure, One advantage of using SfM rather than terrestrial laser scanners is that it is potentially simpler to create a point

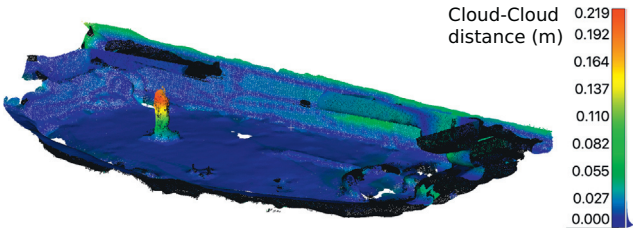

Fig. 19. A heat map resulting from the cloud-cloud distance calculation between the terrestrial LiDAR reference scan and the SfM scan with the foreign object. Areas of large distance are shown by a range of colours in the heat-map. The foreign object is clearly visible in the heat-map. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

cloud without occlusion because more viewpoints can be used to collect the data.

8. Conclusions

Autonomous inspection offers opportunities for reducing the risk, whilst also increasing the repeatability of bridge bearing inspection. Using an pre-defined map is important for autonomous navigation to allow targeted inspection. The map can be created from existing data, such as 3D terrestrial scans data, which is particularly useful in environments where the robot cannot be tele-operated in order to create a map. In this paper, 2D occupancy maps were created from 3D SfM terrestrial LiDAR point cloud data.

A particle filter localisation approach called Adaptive Monte-Carlo Localisation was used to localise the robot in each of the maps created from point cloud data and also a map created from Hector SLAM. Hector SLAM was also considered as an alternative solution to AMCL. Two environments were considered for testing, a lab environment and a real bridge environment.

In the laboratory environment, a typical trajectory error of 3 cm with respect to the ground-truth was obtained for all methods, with the largest error of 8 cm occurring on initialisation of the particle filter for AMCL-SfM. Global localisation was also successful, but with a greater error of 75 cm which was sustained for several time-steps and made it unsuitable for use in a real bridge environment. An error bound of 10 cm was defined in the laboratory environment, with considerations being made about operating in the real bridge environment.

In the bridge environment, the trajectory error was generally greater than in the laboratory due to increased noise in the maps and in the environment, with a typical error ranging between 1 cm and 8 cm. However, all methods remained below the defined error bound, with the exception of AMCL-SfM. However, by combining the worst performing results from AMCL-SfM with Hector SLAM, a more robust method for determining the trajectory is possible that keeps the trajectory error below 6 cm throughout.

In addition to testing localisation and SLAM approaches in a real bridge environment, two proof-of-concept approaches to visual inspection were briefly demonstrated to show potential applications of the robot. One method showed how stereo pairs might be used to calculate the geometry of the bearing and the second approach showed debris monitoring by comparing SfM and terrestrial LiDAR point clouds. These methods will be refined in future work.

Acknowledgments

This work was supported by the following grants: Developing Infrastructure Robotics, Royal Academy of Engineering / Leverhulme Trust LTRSF1516\12\9, Balancing the impact of City Infrastructure Engineering on Natural systems using Robots, UK Engineering and Physical Sciences Research Council EP/N010523/1.

References

[1] J.S. Cho, J.C. Park, H.B. Gil, H.J. Kim, H.H. An, Computer vision techniques for bridge bearing condition assessment using visual inspection photographs, International Association for Bridge and Structural Engineering Symposium Report, 2014, pp. 2697–2704, , https://doi.org/10.2749/222137814814070253.

[2] A. Niemierko, Modern bridge bearings and expansion joints for road bridges, Transportation Research Procedia, 14 2016, pp. 4040–4049, , https://doi.org/10.1016/j.trpro.2016.05.501.

[3] M. Aria, R. Akbari, Inspection, condition evaluation and replacement of elastomeric bearings in road bridges, Struct. Infrastruct. Eng. 9 (2013) 918–934, https://doi.org/10.1080/15732479.2011.638171.

[4] M. Yanagihara, T. Matsuzawa, M. Kudo, T. Turker, Replacement of bearings in the Golden Horn Bridge, Turkey, Struct. Eng. Int. 10 (2) (2000) 121–123, https://doi.org/10.2749/101686600780557857.

[5] N. Baimas, J. McClean, Mancunian Way bearing replacements, Proceedings of the Institution of Civil Engineers - Municipal Engineer, vol. 127, 1998, pp. 124–131, , https://doi.org/10.1680/imuen.1998.30988.

[6] T. Spuler, N. Meng, G. Moor, Life-cycle costs of bridge bearings - key considerations for bridge designers and owners, Multi-Span Large Bridges - Proceedings of the International Conference on Multi-Span Large Bridges, 2015, pp. 743–750, , https://doi.org/10.1201/b18567-95.

[7] British Standard, EN1337 Structural Bearings - Part 10 Inspection and maintenance, 2003, https://standards.globalspec.com/std/3192/bsi-bs-en-1337-10, date accessed: 2018-05-30.

[8] L. Freire, J. De Brito, Relationship between bearings type and their most common anomalies, Proceedings of the 3rd International Conference on Bridge Maintenance, Safety and Management, Life-Cycle Performance and Cost, Porto, 2006, pp. 205–206, , https://doi.org/10.13140/RG.2.1.3995.9769.

[9] L.M. Freire, J. De Brito, J.R. Correia, Management system for road bridge structural bearings, Struct. Infrastruct. Eng. 10 (8) (2014) 1068–1086, https://doi.org/10.1080/15732479.2013.788524.

[10] N. Shibasaki, C. Ikeda, C. Sakano, Maintenance, Monitoring, Safety, Risk and Resilience of Bridges and Bridge Networks - Proceedings of the 8th International Conference on Bridge Maintenance, Safety and Management, (2018) 978-984-33-9313-5http://www.iabse-bd.org/session/69.pdf , Accessed date: 7 September 2018.

[11] Y.-C. Shiau, C.-M. Huang, M.-T. Wang, J.-Y. Zeng, Discussion of pot bearing for concrete bridge, Proceedings from the 25th International Symposium on Automation and Robotics in Construction, Vilnius, 2008, pp. 213–223, , https://doi.org/10.22260/ISARC2008/0033.

[12] J. Liu, X. Miao, Y. Yuan, The rail bridge bearing monitoring system base on FBG, in: Y. Liao, W. Jin, D.D. Sampson, R. Yamauchi, Y. Chung, K. Nakamura, Y. Rao (Eds.), Proceedings of the 4th International Conference on Electronics, Communications and Networks, CRC Press 2015, Beijing, China, 2012, p. 8421AQ, , https://doi.org/10.1117/12.968607.

[13] M. Maizuar, L. Zhang, S. Miramini, P. Mendis, R.G. Thompson, Detecting structural damage to bridge girders using radar interferometry and computational modelling, Struct. Control. Health Monit. 24 (10) (2017) e1985, https://doi.org/10.1002/stc.1985.

[14] M. Behrooz, S. Yarra, D. Mar, N. Pinuelas, B. Muzinich, N.G. Publicover, G. Pekcan, A. Itani, F. Gordaninejad, A self-sensing magnetorheological elastomer-based adaptive bridge bearing with a wireless data monitoring system, Smart Structures and Materials and Nondestructive Evaluation and Health Monitoring, 9803 SPIE, 2016, p. 98030D, , https://doi.org/10.1117/12.2218691.

[15] M. Rossow, Section 9: inspection of bridge bearings, Tech. rep. Federal Highway Administration, 2006, https://www.cedengineering.com/userfiles/InspectionofBridgeBearings.pdf date accessed: 31/05/2018.

[16] L.J.L. Hoeke, P.M. Singh, R.D. Moser, L.F. Kahn, K.E. Kurtis, Degradation of steel girder bridge bearing systems by corrosion, National Association of Corrosion Engineers - International Corrosion Conference Series, 2009 https://www.onepetro.org/conference-paper/NACE-09228 date accessed: 31/05/18, isbn: 09228 2009.

[17] Y. Noh, D. Koo, Y.-M. Kang, D. Park, D. Lee, Automatic crack detection on concrete images using segmentation via fuzzy C-means clustering, International Conference on Applied System Innovation, IEEE, 2017, pp. 877–880, , https://doi.org/10.1109/ICASI.2017.7988574.

[18] P. Prasanna, K.J. Dana, N. Gucunski, B.B. Basily, H.M. La, R.S. Lim, H. Parvardeh, Automated crack detection on concrete bridges, IEEE Trans. Autom. Sci. Eng. 13 (2) (2016) 591–599, https://doi.org/10.1109/TASE.2014.2354314.

[19] A. Akutsu, E. Sasaki, K. Takeya, Y. Kobayashi, K. Suzuki, H. Tamura, A comprehensive study on development of a small-sized self-propelled robot for bridge inspection, Struct. Infrastruct. Eng. 13 (8) (2017) 1056–1067, https://doi.org/10.1080/15732479.2016.1236132.

[20] N.H. Pham, H.M. La, Design and implementation of an autonomous robot for steel bridge inspection, 54th Annual Allerton Conference on Communication, Control, and Computing, 10 2016, pp. 556–562, , https://doi.org/10.1109/ALLERTON.2016.7852280.

[21] C.-H. Yang, M.-C. Wen, Y.-C. Chen, S.-C. Kang, An optimized unmanned aerial system for bridge inspection, Proceeding of the 32nd International Symposium on Automation and Robotics in Construction, 2015, pp. 1–6, , https://doi.org/10.22260/ISARC2015/0084.

[22] L. Pauly, H. Peel, S. Luo, D. Hogg, R. Fuentes, Deeper networks for pavement crack detection, Proceedings of the 34th International Symposium on Automation and Robotics in Construction, 2017, pp. 479–485, , https://doi.org/10.22260/ISARC2017/0066.

[23] F. Bonnín-Pascual, A. Ortiz, Detection of cracks and corrosion for automated vessels visual inspection, Proceedings of the 2010 Conference on Artificial Intelligence Research and Development: Proceedings of the 13th International Conference of the Catalan Association for Artificial Intelligence, 2010, pp. 111–120, , https://doi.org/10.3233/978-1-60750-643-0-111.

[24] F.F. Feliciano, F.R. Leta, F.B. Mainier, Texture digital analysis for corrosion monitoring, Corros. Sci. 93 (2015) 138–147, https://doi.org/10.1016/j.corsci.2015.01.017.

[25] C.M. Yeum, S.J. Dyke, Vision-based automated crack detection for bridge inspection, Comput. Aided Civ. Inf. Eng. 30 (10) (2015) 759–770, https://doi.org/10.1111/mice.12141.

[26] M.R. Jahanshahi, S.F. Masri, Adaptive vision-based crack detection using 3D scene reconstruction for condition assessment of structures, Autom. Constr. 22 (2012) 567–576, https://doi.org/10.1016/j.autcon.2011.11.018.

[27] M.M. Torok, M. Golparvar-Fard, K.B. Kochersberger, Image-based automated 3D crack detection for post-disaster building assessment, J. Comput. Civ. Eng. 28 (5) (2014) A4014004, https://doi.org/10.1061/(ASCE)CP.1943-5487.0000334.

[28] D. Lattanzi, G.R. Miller, 3D scene reconstruction for robotic bridge inspection, J. Infrastruct. Syst. 21 (2) (2015) 04014041, https://doi.org/10.1061/(ASCE)IS.1943-

555X.0000229.
[29] A. Mazumdar, H.H. Asada, Mag-Foot: a steel bridge inspection robot, Int. Conf. Intell. Robot. Syst. (2009) 1691–1696, https://doi.org/10.1109/IROS.2009.5354599.
[30] S. Dorafshan, M. Maguire, N.V. Hoffer, C. Coopmans, Challenges in bridge inspection using small unmanned aerial systems: results and lessons learned, International Conference on Unmanned Aircraft Systems, IEEE, Miami, FL, USA, 2017, pp. 1722–1730, , https://doi.org/10.1109/ICUAS.2017.7991459.
[31] M. Meireles, R. Lourenco, A. Dias, J.M. Almeida, H. Silva, A. Martins, Real time visual SLAM for underwater robotic inspection, Oceans, IEEE, 2014, pp. 1–5, , https://doi.org/10.1109/OCEANS.2014.7003097.
[32] D. Droeschel, M. Schwarz, S. Behnke, Continuous mapping and localization for autonomous navigation in rough terrain using a 3D laser scanner, Robot. Auton. Syst. 88 (2017) 104–115, https://doi.org/10.1016/j.robot.2016.10.017.
[33] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, Chapter 8.3 - Monte Carlo Localization, pp 250–263, 10th ed., The MIT Press, 2006 isbn: 978-0-262-20162-9.
[34] T. Moore, D. Stouch, A generalized extended Kalman filter implementation for the robot operating system, in: E. Menegatti, N. Michael, K. Berns, H. Yamaguchi (Eds.), Intelligent Autonomous Systems 13: Proceedings of the 13th International Conference, Springer, Cham, 2016, pp. 335–348, , https://doi.org/10.1007/978-3-319-08338-4_25.
[35] W. Burgard, A. Derr, D. Fox, A. Cremers, Integrating global position estimation and position tracking for mobile robots: the dynamic Markov localization approach, Proceedings. 1998 International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications, vol. 2, IEEE, 1998, pp. 730–735, , https://doi.org/10.1109/IROS.1998.727279.
[36] D. Fox, W. Burgard, F. Dellaert, S. Thrun, Monte Carlo localization: efficient position estimation for mobile robots, The Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI'99), 1999, pp. 343–349 isbn: 978-0-262-51106-3.
[37] R.S. Lim, H.M. La, Z. Shan, W. Sheng, Developing a crack inspection robot for bridge maintenance, International Conference on Robotics and Automation, IEEE, 2011, pp. 6288–6293, , https://doi.org/10.1109/ICRA.2011.5980131.
[38] R.S. Lim, H.M. La, W. Sheng, A robotic crack inspection and mapping system for bridge deck maintenance, IEEE Trans. Autom. Sci. Eng. 11 (2) (2014) 367–378, https://doi.org/10.1109/TASE.2013.2294687.
[39] H.M. La, R.S. Lim, B.B. Basily, N. Gucunski, J. Yi, A. Maher, F.A. Romero, H. Parvardeh, Mechatronic systems design for an autonomous robotic system for high-efficiency bridge deck inspection and evaluation, Trans. Mechatron. 18 (6) (2013) 1655–1664, https://doi.org/10.1109/TMECH.2013.2279751.
[40] R. Mur-Artal, J.M.M. Montiel, J.D. Tardos, ORB-SLAM: a versatile and accurate monocular SLAM system, Transactions on Robotics 31 (5) (2015) 1147–1163, https://doi.org/10.1109/TRO.2015.2463671.
[41] M. Labbe, F. Michaud, Online global loop closure detection for large-scale multi-session graph-based SLAM, International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 2661–2666, , https://doi.org/10.1109/IROS.2014.6942926.
[42] S. Kohlbrecher, O. von Stryk, J. Meyer, U. Klingauf, A flexible and scalable SLAM system with full 3D motion estimation, International Symposium on Safety, Security, and Rescue Robotics, IEEE, 2011, pp. 155–160, , https://doi.org/10.1109/SSRR.2011.6106777.
[43] S. Kohlbrecher, J. Meyer, T. Graber, K. Petersen, U. Klingauf, O. von Stryk, Hector open source modules for autonomous mapping and navigation with rescue robots, RoboCup 2013: RoboCup 2013: Robot World Cup, vol. XVII, 2014, pp. 624–631, , https://doi.org/10.1007/978-3-662-44468-9_58.
[44] J. Levinson, M. Montemerlo, S. Thrun, Map-based precision vehicle localization in urban environments, Robot. Sci. Syst. III (2007) 121–128, https://doi.org/10.15607/RSS.2007.III.016.
[45] B. Behzadian, P. Agarwal, W. Burgard, G.D. Tipaldi, Monte-Carlo localization in hand-drawn maps, International Conference on Intelligent Robots and Systems, IEEE, 2015, pp. 4291–4296, , https://doi.org/10.1109/IROS.2015.7353985.
[46] 3Dflow, Structure from Motion Software (3DF Zephyr) User Manual, 2013, http://3dflow.net/zephyr-doc/3DFZephyrManual2500English.pdf, date accessed: 30/05/2018.