# Towards a Cooperative Robotic System for Autonomous Pipe Cutting in Nuclear Decommissioning

Thomas Burrell, Craig West, Stephen D. Monk, Allahyar Montazeri, C. James Taylor,
Engineering Department, Lancaster University, Lancaster, UK, Email: c.taylor@lancaster.ac.uk

***Abstract*— A mobile camera is used to support an assisted teleoperation pipe–cutting system for nuclear decommissioning. The base system consists of dual–manipulators with a single mounted Kinect camera. The user selects the object from an on–screen image, whilst the computer control system automatically grasps the pipe with one end–effector and positions the second for cutting. However, the system fails in some cases because of data limitations, for example a partially obscured pipe in a challenging decommissioning scenario (simulated in the laboratory). Hence, the present article develops a new method to increase the use case scenarios via the introduction of mobile cameras e.g. for mounting on a drone. This is a non-trivial problem, with SLAM and ArUco fiducials introduced to locate the cameras, and a novel error correction method proposed for finding the ArUco markers. Preliminary results demonstrate the validity of the approach but improvements will be required for robust autonomous cutting. Hence, to reduce the pipe position estimation errors, suggestions are made for various algorithmic and hardware refinements.**

***Keywords*— *Cooperative Robotics, Nuclear Decommissioning, Computer Vision, ArUco Fiducials, OpenCV, SLAM, Kinect.***

## I. Introduction

Nuclear decommissioning is an obvious environment for the use of autonomous robots. Not only is much of the environment inaccessible to humans, but a high level of precision is needed to maintain safety. In fact, robots have been used in nuclear environments from the beginning, and they were often the only way to access highly contaminated areas. However, as modern robots have become more intelligent and autonomous, the nuclear industry has tended to lag behind, largely due to safety and reliability concerns. For example, high doses of nuclear and electromagnetic radiation have a negative effect on electronics [1], along with other elements of robots such as actuators and sensors [2]. Hence, nuclear robots usually emphasise simplicity and the use of highly trained human control, as opposed to artificial intelligence. Reference [3], for example, considers the development of an inspection robot for radioactive areas of a nuclear plant. The robot is mounted on a rail system, vastly reducing mobility of the robot, while increasing simplicity and reliability. The nuclear decommissioning robotic platform in reference [4] favours hydraulic actuators over electric ones.

With many nuclear facilities coming to the end of their lives, the need for robotic systems becomes more pressing. Nuclear installations have materials that present various radioactive and chemical hazards; their architecture is often complex; and they were not necessarily designed with the decommissioning problem in mind. For any new robotic system, the case for increased autonomy must outweigh the potential disadvantages. Autonomous robots can be faster and more precise than tele-operated ones. During the Fukushima incident many of the robots used for the initial response, and for clean-up, were designed especially for decommissioning [5]. These, and other forms of emergency response robots [6], must generally work independently, increasing the need for greater autonomy [7].

In fact, most robotic systems presently in the nuclear sector are directly teleoperated, as was originally the case for the robotic platform considered in this article: the laboratory demonstrator is based on a BROKK base unit, linked via a bespoke back-plate to two HYDROLEK manipulators [4]. The present authors have recently developed and evaluated for this machine, a vision based semi-autonomous object grasping system for decommissioning applications, using a single mounted Kinect camera [8]. The system has some similarities to that described by [9], although the latter consider a single manipulator with two 3D cameras. The new system here, presents a straightforward graphical interface to the operator, who uses a mouse to, for example, select target positions for each manipulator to perform a pipe grasp and cut action [8].

However, the new system fails in some cases because of data limitations, for example a partially obscured pipe in a challenging decommissioning scenario (simulated in the laboratory). Hence, the present article develops a new method to increase the use case scenarios via the introduction of mobile cameras e.g. for mounting on a drone. This is a non-trivial problem, with SLAM and ArUco fiducials necessarily introduced to locate the cameras, and a novel error correction method proposed for finding the ArUco markers. Section II of the article provides more background information about the existing BROKK system and the new multi-robot approach to broaden its use case. Section III describes methods for locating multiple robots and for sharing coordinate frames. Section IV presents the results using two robots to locate and position for a pipe cutting task, followed by the conclusions in Section V.

## II. Existing System and Proposed Improvements

BROKK tracked robots are commonly used in nuclear decommissioning, with multiple tools such as excavators and cutters. The laboratory-based BROKK robotic platform at Lancaster University has been developed for research into system identification, control and decommissioning e.g. [10-11]. In this case, the two hydraulic manipulators are used to provide multiple degrees of freedom (DoF). However, control requires a skilled human operator and multiple joysticks, with the difficulty of teleoperation. By contrast, with the assisted teleoperation system alluded to above, the machine can now autonomously grasp and position for a cutting task, with minimal human decision making [8].

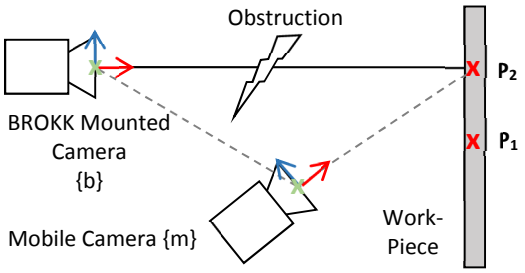


Fig. 1. In the event of an obstruction, postion in the {b} camera frame can be estimated using a second camera with an unobstructed view of the workpeice.

The new vision system seeks to remove the skilled human from selected elements of the control loop, replacing complex controls with a simple point and click interface, while controlling the robot automatically. A Microsoft Kinect camera (selected for convenient R&D purposes) is mounted between the two manipulators, looking over the workspace. A human user is presented with the image, and can use a mouse to press on the pipe in two locations, one for grasping and the other for the line of cut. Edge detection software is used to segment the pipe from its surroundings. Note that the Kinect camera produces RGB-D images, including the depth element.

### A. *Failure Cases of Current System*

One major problem with the current system is the position of the Kinect camera, on the shoulders of the manipulators. This fixed view means that a pipe cannot be cut if it is obscured, if there is some unseen branch, or if there is little clearance from a wall. Human operators may be able to correctly infer these situations and correct for them, while autonomous robots would struggle without more information. This information can be obtained from other viewpoints, for example, moving round an obstruction. Although the whole BROKK robot can potentially be moved by means of its caterpillar tracks, it is cumbersome and inefficient to regularly redeploy.

Another option would be to move the camera onto the end of one or both manipulators. These could subsequently be used to point the camera at any angle. However, the current mounting position offers many advantages. Mounting the Kinect camera on the shoulders of the BROKK robot gives a fixed frame of view, simplifying inverse kinematics and giving a full view of the manipulator workspace. Furthermore, it places the delicate camera away from cutting tools and collisions, reducing the risk of debris on the lens, and vibrations from the manipulators.

An autonomous robot may also fail due to lack of information about the wider environment. For example, cutting one pipe may cause others far above to collapse. To solve this problem, a specific order of tasks must be followed i.e. to prevent or control a collapse. Such on the fly planning, and re-planning, requires much more information than can be gathered by fixed cameras on large, relatively cumbersome hydraulic robots.

### B. *Solution Overview*

A full solution to this problem is beyond the scope of this work, which instead concentrates on a single simplified situation, as a first step towards development of the full system. In this case, we consider a single work-piece, a pipe section, placed vertically within the workspace of the manipulators. The work-piece is totally obscured from view of the Kinect camera, but can still be reached by the manipulators. One solution is to use another camera with an unobscured view of the work-piece. This camera can be mounted on a smaller robot (for example a drone) that can move to obtain an unobstructed view. To cut the pipe, we require a point $\boldsymbol{P_1}\{b\} \in \mathbb{R}^3$ for grasping and a second point $\boldsymbol{P_2}\{b\} \in \mathbb{R}^3$ for where to cut. Both are needed in the coordinate frame of the BROKK camera {b}. The mobile camera can see the same points in its coordinate frame {m}, $\boldsymbol{P_1}\{m\} \in \mathbb{R}^3$, and $\boldsymbol{P_2}\{m\} \in \mathbb{R}^3$. The points can be translated from {m} to {b} using a vector between the two systems, as shown in Fig. 1. The translation is dependent upon knowing the location and orientation (pose) of the two cameras.

## III. Locating Two Robots

To translate the points $\boldsymbol{P1} \in \mathbb{R}^3$ and $\boldsymbol{P2} \in \mathbb{R}^3$ between camera systems, one of two things is required; the pose of one camera in the frame of the second, or the pose of both cameras in another reference frame. The former means that one camera must keep the other in its field of view; this is impractical and adds another point of failure when this view is also obstructed. Instead, an additional world frame can be created and the pose of both cameras can be estimated in this new frame. The initial location of the mobile camera is chosen for the reference origin for a new frame {o}. This is a fixed point that does not move during the experiment. The reason for choosing this frame becomes apparent when looking at the methods to locate the two cameras with respect to each other.

### A. *Locating the Mobile Camera*

The field of mobile robotics provides various solutions for determining the position of a robot over time. Inertial Measurement Units (IMU) and GPS [12] and vision [13] can be used to estimate pose. The state of the art in visual pose estimation is Visual SLAM (Simultaneous Localisation and Mapping) [14]. This can be used to estimate pose and build a map of the environment, with both reducing uncertainty in the other. One open source SLAM algorithm is ORB-SLAM2 [15]. ORB (Oriented FAST and Rotated BRIEF) is a method for extracting features from an image, providing them with a description so that they can be matched in future frames [16]. ORB-SLAM2 tracks the relationships between points over time, estimating the motion of the camera. Also implemented is loop-closure, i.e. by recognising visited areas of a map, it can be retroactively altered to fix any drift between visits.

The key advantage for this project, is that ORB-SLAM2 is an “out-of-the-box” solution. With an input of mono or stereo images, the software performs the whole SLAM process and outputs real-time camera pose and a map of the environment. A 7 DoF pose of the mobile camera ($\boldsymbol{M} \in \mathbb{R}^7$) is produced, consisting of an xyz coordinate and a quaternion. This can be tracked a frame with the origin defined at the initialisation of the SLAM system. This is the transformation between {o} and {m}.

### B. *Locating the BROKK Robot*

The pose of the BROKK mounted camera is the next challenge. Again, there are many options. The ideal solution would be to implement SLAM upon this camera too, sharing maps between the robots.

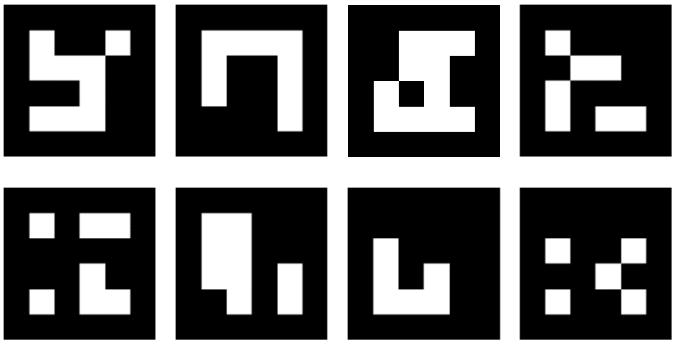

Fig. 2. Examples of ArUco fiducials.

Tracking the robots in the same map automatically ties the origins of their coordinate frames together and leads to easy transformation between the camera frames. However, this method could increase bandwidth use, not idea in a high radiation/interference area, and adds complexity to the system. A simpler method is to locate the BROKK camera using the mobile camera. Locating the BROKK camera means recognising it. This is not a trivial task, even state of the art object recognition and learning techniques are only just becoming realistic to use [17]. A simple approach is using fiducials, easily recognised objects placed into the world. An example of this is ping-pong balls used for motion tracking of actors. Another example is ArUco fiducials [18], as illustrated in Fig. 2.

Using the ArUco library with openCV, binary codes are generated from a dictionary. The patterns of black and white pixels are uniquely identifiable and easy to detect. Once detected in an image, the position of the centre is found. If the square length is known, then the pose of the markers can be calculated [18]. Just like the SLAM system, the ArUco system is a black box. Images with at least one ArUco marker are input, and the identity, position in the camera frame, and the orientation of the markers are the output. The 6 DoF pose of any fiducial ($\boldsymbol{F} \in \mathbb{R}^6$), contains a three-dimensional coordinate and three angles forming a Tait–Bryan rotation. The transformation is between the mobile camera frame {m} and the fiducial frame {f}. If the BROKK camera and a fiducial are placed together, with axes aligned, the transformation between the two can easily be measured with standard tools. Therefore, a path exists from frame {m} to frame {b} by passing through frame {f}.

The accuracy of the fiducial system is unknown; it relies on factors that could introduce inaccuracy. Accurately calculating pose of the markers needs the side length of the markers, and calibration properties of the camera. The latter is most likely to be incorrect; properties such as distortion model are only estimated using calibration techniques such as [19]. Another source of errors occurs for low resolution cameras or when detecting markers at a distance. Position is defined from discrete pixels and the width of a pixel may become significant. In general, errors can be reduced with more information. Specifically using multiple markers, the known inter marker distances can be compared to those measured by the software.

Consider $n$ ArUco markers, with ID numbers 1 to $n$, printed onto a single sheet of paper. Defined in a page frame {p}, the centres of each marker are $\boldsymbol{C}_n\{p\} \in \mathbb{R}^2$ (1). The inter marker distances are stored in a matrix $\boldsymbol{D} \in \mathbb{R}^{nxn}$, which is invariant of frame, such that each element is the magnitude of the difference between two markers, with IDs equal to the row and column numbers (2).

$$\boldsymbol{C}_n\{p\} = (x_n\{p\}, y_n\{p\}) \tag{1}$$

$$\boldsymbol{D}_{i,j} = \left\| \boldsymbol{C}_i\{p\} - \boldsymbol{C}_j\{p\} \right\| \tag{2}$$

$$\widehat{\boldsymbol{C}}_n\{c\} = (xm_n\{c\}, ym_n\{c\}, zm_n\{c\}) \tag{3}$$

The error comes from multiple sources and so each have different models. In the camera frame $\{c\}$, the centre position ($\widehat{\boldsymbol{C}}_n\{c\} \in \mathbb{R}^3$) (3) has three components that may be affected by different error sources. For example, if the side length of the markers is incorrectly defined, then the 3 components of $\widehat{\boldsymbol{C}}_n$ should be out by a scale factor. However, errors due to resolution will cause the centre to move only in the x-y plane of the camera. Furthermore, this effect is more apparent at greater distances, and this model would be a function of the z-component.

Here we consider a scalar error. We assume that the positions in the camera frame ($\boldsymbol{C}_n\{c\} \in \mathbb{R}^3$) are equal to the measured, multiplied by a scaler factor $S$ (4). To find the scale factor, the measured positions are compared directly to the known positions. This is achieved through the inter market distances which are invariant with frame. The measured inter marker distance $\widehat{\boldsymbol{D}} \in \mathbb{R}^{nxn}$ is the magnitude of the measured distance between markers i.e. those with ID numbers that match the row and column numbers (5). The same relationship as (4) exists, where the real inter marker distances should be approximately equal to the measured values, multiplied by a scalar factor (6).

$$\boldsymbol{C}_n\{c\} \approx S\, \widehat{\boldsymbol{C}}_n\{c\} \tag{4}$$

$$\widehat{\boldsymbol{D}}_{i,j} = \left\| \widehat{\boldsymbol{C}}_i\{c\} - \widehat{\boldsymbol{C}}_j\{c\} \right\| \tag{5}$$

$$\boldsymbol{D} \approx S\, \widehat{\boldsymbol{D}} \tag{6}$$

To estimate a value for $S$, we consider a matrix of scale factors $\boldsymbol{S} \in \mathbb{R}^{nxn}$. The Hadamard product of $\boldsymbol{S}$ and $\boldsymbol{D}$ will be exactly equal to the known inter marker distances (7). The elements of $\boldsymbol{S}$ are be calculated by elementwise division of the two $\boldsymbol{D}$, ignoring the zero diagonal elements (8). If the error model is correct then all off-diagonal elements of $\boldsymbol{S}$ will be equal; however, due to noise and other errors this will not be the case. A value of $S$ can instead be estimated by taking the mean of off-diagonal elements of $\boldsymbol{S}$ (9), used in (4) to correct the measured position of each marker. Section IV.A shows the results of this process.

$$\boldsymbol{D} = \boldsymbol{S} \odot \widehat{\boldsymbol{D}} \tag{7}$$

$$\boldsymbol{S}_{i,j} = \frac{\boldsymbol{D}_{i,j}}{\widehat{\boldsymbol{D}}_{i,j}},\ for\ i \neq j \tag{8}$$

$$S = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} \boldsymbol{S}_{i,j}}{n^2 - n},\ for\ i \neq j \tag{9}$$

### *C. Translating Between Coordinate Systems*

Returning to the problem of translating the position of a pipe in one camera field of view to a second camera's field of view. The coordinate frame {o} is so far used only to give a reference for the position of the mobile camera. However, this can also be used to solve the problem when the mobile camera cannot see the BROKK camera and work-piece in the same field of view.

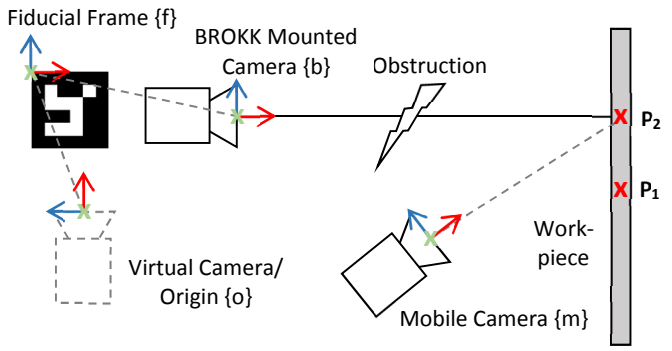


Fig. 3. Measuring the position of {b} with a fiducial and the position of {m} with SLAM.

Consider that the BROKK camera does not move and only needs to be located once during the experiment. In fact, this measurement can be performed by the mobile camera before it begins to move, i.e. the position of the BROKK camera and the mobile camera are now both within the origin frame {o} i.e. a virtual third camera that remains at the origin of the experiment. This forms a closed loop of frames with known transformations. Fig. 3 shows this loop. First the mobile camera sees the points $\mathbf{P_1}\{m\}$, and $\mathbf{P_2}\{m\}$. Finding these points is hardware specific to the type of camera used for the mobile camera. Next, the points are transferred into the origin frame {o}. The transformation {o} to {m} is known from SLAM, so the reverse is applied. Then, the points are transferred to the fiducial frame {f}, the transformation is gathered using ArUco software and the mobile camera at the beginning of the experiment. Finally, the points are transferred from {f} to {b} to obtain the corresponding position relative to the BROKK camera for inverse kinematics.

Such coordinate frame transforms consist of two parts: a translation to bring the origins of the two systems together, and a rotation to align the axes of the two systems. The SLAM system outputs a pose $\boldsymbol{M}$ in frame {o}, containing a Cartesian coordinate $\boldsymbol{T}_M \in \mathbb{R}^3$, and a quaternion $\boldsymbol{q}_M \in \mathbb{R}^4$ (10). Taking the four quaternion elements, yields rotation matrix $\boldsymbol{R}_M \in \mathbb{R}^{3x3}$. Transforming any Cartesian point from {m} to {o}, $\boldsymbol{P}\{m\} \in \mathbb{R}^3$ to $\boldsymbol{P}\{o\} \in \mathbb{R}^3$, means rotating by $\boldsymbol{R}_M$ and translating by $\boldsymbol{T}_M$ as in (11).

$$\boldsymbol{M} = [x_M\ y_M\ z_M\ q_{1M}\ q_{2M}\ q_{3M}\ q_{4M}]^T \tag{10}$$

$$\boldsymbol{P}\{o\} = \boldsymbol{R}_M\boldsymbol{P}\{m\} + \boldsymbol{T}_M \tag{11}$$

Next is to transform the point with respect to the origin frame into the fiducial frame $\boldsymbol{P}\{f\} \in \mathbb{R}^3$. The pose $\boldsymbol{F}$ contains a Cartesian coordinate $\boldsymbol{T_F} \in \mathbb{R}^3$, and a rotation vector $\boldsymbol{r}_F \in \mathbb{R}^3$, of rotations about each of the three axes $(r_x, r_y, r_z)$. The matrix representing this rotation, $\boldsymbol{R}_F \in \mathbb{R}^{3x3}$, is found using the Rodrigues function (12). This pose represents a transformation from {o} to {f} frames, but to transform a point in frame {o} to {f} requires the inverse. Therefore, $\boldsymbol{P}\{f\}$ is found using the inverse rotation and a negative translation as in (13).

$$\boldsymbol{R}_F = \cos\theta\, \boldsymbol{I} + (1 - cos\theta)\boldsymbol{r}_F\boldsymbol{r}_F{}^T + \sin\theta \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \tag{12}$$

Where, $\theta = \|\boldsymbol{r}_F\|$

$$\boldsymbol{P}\{f\} = \boldsymbol{R}_M{}^{-1}\boldsymbol{P}\{o\} - \boldsymbol{T}_M \tag{13}$$

Finally, the position of the point in the BROKK frame $\boldsymbol{P}\{b\} \in \mathbb{R}^3$ is found. The pose of the camera is not measured in the origin frame {o} but in the fiducial frame {f}. This is highly dependent upon the choice of position for the printed fiducial. Placing the fiducial so that any axis is collinear to any axis of the BROKK camera, and the other two axes parallel, reduces the rotation $\boldsymbol{R}_B \in \mathbb{R}^3$ to two ±90°, and the translation $\boldsymbol{T}_B \in \mathbb{R}^3$ to a single dimension, with the other elements zero. Furthermore, considering the values of $\boldsymbol{T}_B$, and $\boldsymbol{R}_B$ from {b} to {f} means the rotation does not need to be inverted. Therefore, the final transformation into frame {b} is (14), creating full loop for any point in {m} to the {b} frame.

$$\boldsymbol{P}\{b\} = \boldsymbol{R}_B\boldsymbol{P}\{f\} + \boldsymbol{T}_B \tag{14}$$

Using the point P{b} is equivalent to measuring the position using the BROKK Kinect camera alone and provides a numerical answer that can be used directly with the existing inverse kinematics and manipulator control software.

## IV. Experimental Results

Two experiments have been performed to test the new system presented in this work. The first tests the accuracy of poses found using ArUco markers, as discussed in section III.B. The second experiment tests the full system described in section III.C, integrating a mobile camera into the existing system.

### A. ArUco Feducial Validation

We test the ArUco markers as a method for locating the BROKK camera, looking to validate the pose given by software, and to calibrate for reduced errors. The experimental setup is shown in Fig. 4, with four random ArUco fiducials printed with side lengths of 75mm onto white A4 paper. A 1920x1080, monocular, colour camera is placed in plane with the sheet at 400mm distance. A measuring tape is placed perpendicular to the sheet so that the distance to the camera can be altered. The camera sensor is within the body of the camera and so its position cannot be known precisely, but is assumed to be 20mm from the lens; this is an uncertainty in the results. We have one independently measured element, the distance perpendicular to the sheet to the camera, or the z component of the measured pose.

The ArUco software is used to measure the positions of all 4 markers at varying distances. The camera is moved from the starting position in increments of 100 mm to a total of 1200 mm, with 5 measurements taken for each marker. Fig. 5 shows the measured depth compared to the actual position where each image was taken. If the measurements produced are accurate they should lay upon the perfect measurements line. However, due to the precision of placing the camera by hand, and the unknown position of the sensor, we assume a 20 mm tolerance. It is shown that for small distances the measurements match well. For large distances this is not true, with a maximum error of 52 mm (5%) at the furthest distance. Furthermore, the range

of values is large at up to 30 mm. For Fig. 6 a similar experiment is run, with the side length of the markers entered erroneously. In this case the markers are said to be 50 mm instead of 75 mm. As should be expected the software fails to measure the length correctly; again, there is a large range of values at the same position. Both figures have a second set of "Normalised Measurements", namely the results of applying (1) to (9) to the raw data. Once scaled, the results are much closer to the expected in both cases, with much smaller ranges of values. This shows that while the ArUco software on its own can provide acceptable measurements, when provided with the correct information, the new correction system can provide even more accurate results.

### *B. Shared Vision Grasping*

The final experiment is to locate a pipe in the BROKK reference frame using a hand held mobile camera that performs SLAM to locate itself, and using ArUco fiducials to locate the BROKK at the start of the experiment. The camera used is a DJI guidance camera. This device consists of 5 stereo pairs that can be pointed in different directions. For this experiment a single pair is used. The camera comes with a dedicated processing unit outputting many useful measurements, such as the distance to nearby objects, the velocity of the camera, and processed depth images, as well as the raw images from each individual camera. Images are produced at 20 Hz of 360x240 resolution.

Fig. 7 shows the full setup for this experiment. The outer loop (red) represents the existing system using a Kinect camera to find the position of the pipe and controlling the robot to grasp and cut. The new loop uses the guidance camera connected to a laptop computer running Linux Ubuntu, using ROS. Images are published to the ROS Master to be used by the main program. The program starts by detecting the same ArUco markers from Fig. 4, and the method presented in section IV.B, to estimate the pose of the board. A user chooses to initialise, the ArUco detector is stopped and SLAM begins to estimate the pose of the camera. The two poses are published by the ROS Master.

A windows PC running MATLAB then uses ROS to subscribe to all information published by the master. A GUI is drawn; this shows in real time the origin axes, the position of the mobile camera, the position of the Kinect camera, the position of the ArUco board, and a stream of the view from the left camera overlaid with the depth map. A user can click on any pixel in the camera stream to extract the position of that pixel in {m}. Using the method presented in section IV.C, this coordinate is converted to the {b} coordinate system. The position of the pipe is now fed into the existing inverse Kinematics algorithm to get joint angles for the manipulators. The joint angles are transferred to the National Instruments Labview program, via local TCP/IP, which controls the robot to grasp and cut the pipe. To test this system, the pipe position given by the new system needs to be validated. The best approach is to assume that the position given by the Kinect camera is accurate, then to compare this value to the coordinate given by the new system translated into the {b} frame.

The pipe was placed in front of the Kinect camera and its position measured; this forms the benchmark. Next the new system is initialised and the camera is moved to 3 locations; at each location the position of the same point on the pipe is measured. This is repeated 3 times by re-initialising each time for a total of 9 measurements of the same point. For the first two sets of 3 results, the camera was moved slowly and smoothly with minimal rotation. The final set was taken with sharp movements and large rotations. The magnitude of error between the systems is taken. Under good conditions the average error is 225 mm, while the worst-case error is 589 mm. Two errors seem to be predominant: the SLAM system gives good positional location for the camera but poor orientation, and the depth from the Guidance camera consistently under estimates.

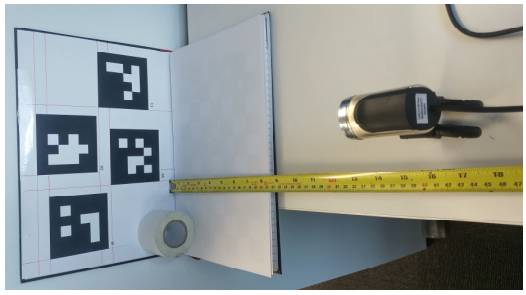

Fig. 4. Checking the accuracy of ArUco fiducial markers.

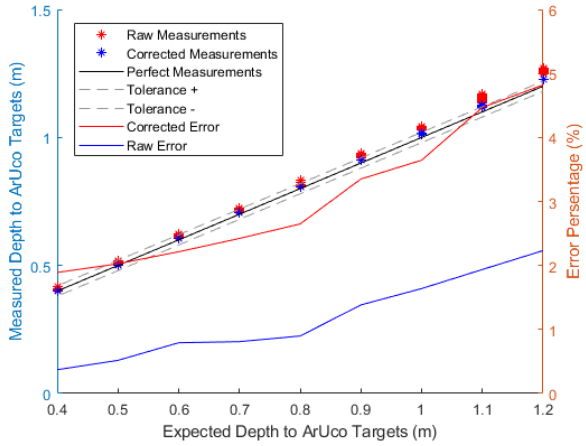


Fig. 5. Comparing the measured distance to ArUco targets against the distance the camera was placed at. Normalised Measurements have been corrected.

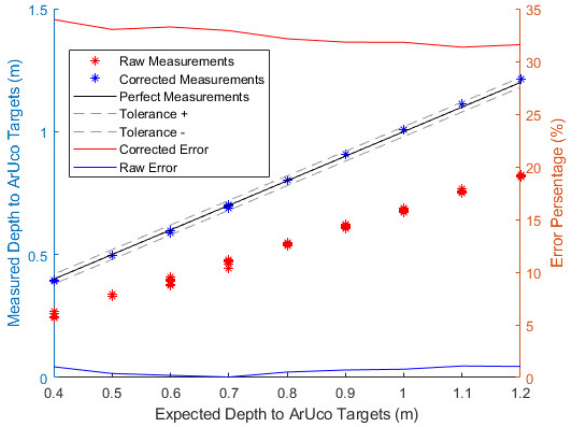


Fig. 6. Comparing the measured distance to ArUco targets against the camera distance. Normalised Measurements have been corrected. Raw measurements have been deliberately corrupted to demonstrate the correction method.

## V. Conclusions

The article has proposed a method to increase the use case scenarios for a nuclear decommissioning robot via the introduction of mobile cameras, requiring SLAM and ArUco fiducials. Preliminary results demonstrate the validity of the approach but improvements will be required for robust autonomous pipe cutting. Significant errors are due to the currently used low resolution mobile camera, which should be replaced. These errors are accumulated as the images from the mobile camera are used to calculate the {f} frame, the {m} frame and the position of the pipe. The future of the project is to expand the use case. Currently the mobile camera is used when the Kinect camera has no vision of the pipe at all. However, situations exist in which Kinect can see the pipe but additional information would be valuable. Therefore, fusing the vision of the two cameras would be useful. For example, the Kinect camera can be used to find the location of the pipe, while the mobile camera can be used to find more qualitative information such as pipe branches and mounting points, and to facilitate improved decision making. Also, the mobile camera will be mounted onto a mobile robot. Most likely a UAV i.e. small, mobile and unaffected by terrain. This UAV can use SLAM alongside greater intelligence for path and mission planning. Furthermore, SLAM, currently used just to locate the robot, produces maps as a by-product. These could be highly useful for demolition planning, and can potentially replace the images currently used to locate the pipes.

## Acknowledgments

This work is supported by the UK Engineering and Physical Sciences Research Council (EPSRC) grant EP/R02572X/1. The authors are also grateful for the support of the Nuclear Decommissioning Authority and National Nuclear Laboratory.

## References

[1] Schrimpf, R.D. and D.M. Fleetwood, Radiation effects and soft errors in integrated circuits and electronic devices, *World Scientific*, Vol. 12. 2004.
[2] Sharp, R. and M. Decreton, Radiation tolerance… nuclear robot applications. *Reliability Eng. & System Safety*, 53 (3), pp. 291-299, 1996.
[3] Yamamoto, S., Development of inspection robot for nuclear power plant. *IEEE International Conference on Robotics and Automation*, 1992.
[4] Bakari, M.J., K.M. Zied, and D.W. Seward, Development of a multi-arm mobile robot for nuclear decommissioning tasks. *International Journal of Advanced Robotic Systems*, 4 (4), pp. 51, 2007.
[5] Kawatsuma, S., M. Fukushima, and T. Okada, Emergency response by robots to Fukushima-Daiichi accident: summary and lessons learned. *Industrial Robot: An International Journal*, 39 (5), pp. 428-435, 2012.
[6] Murphy, R.R., A decade of rescue robots. *IEEE/RSJ Intelligent Robots and Systems (IROS)*, 2012.
[7] N. Marturi, A. Rastegarpanah, C. Takahashi, M. Adjigble, R. Stolkin, S. Zurek, M. Kopicki, M. Talha, J. A. Kuo, and Y. Bekiroglu, Towards advanced robotic manipulation for nuclear decommissioning: A pilot study on tele-operation and autonomy, *IEEE Robotics and Automation for Humanitarian Applications*, Kollam, 2016.
[8] West, C., S.D. Monk, A. Montazeri and C.J. Taylor, A vision–based positioning system with inverse dead–zone control for dual–hydraulic manipulators, *UKACC 12th Int. Conf. Control*, Sheffield, 2018.
[9] D. Kent, C. Saldanha, and S. Chernova, A comparison of remote robot teleoperation interfaces for general object manipulation, *ACM/IEEE International Conference on Human Robot Interaction*, Vienna, 2017
[10] Taylor, C.J. and Robertson, D., State–dependent control of a hydraulically–actuated nuclear decommissioning robot, *Control Engineering Practice*, 21 (12), pp. 1716–1725, 2013.
[11] Montazeri, A., West, C., Monk, S. and Taylor, C.J., Dynamic modeling and parameter estimation of a hydraulic robot manipulator using a multi-objective genetic algorithm, *Int. J. Control*, 90 (4) pp. 661–683, 2017.
[12] Wendel, J. *et al*., An integrated GPS/MEMS-IMU navigation system for an autonomous helicopter. *Aerospace Science and Technology*, 10 (6), pp. 527-533, 2006.
[13] Nistér, D., O. Naroditsky, and J. Bergen. Visual odometry. *IEEE Computer Vision and Pattern Recognition*, 2004.
[14] Durrant-Whyte, H. and T. Bailey, Simultaneous localization and mapping: part I. *IEEE Robotics & Automation Magazine*, 13 (2), pp. 99-108, 2006.
[15] Mur-Artal, R. and J.D. Tardós, ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 33 (5), 2017.
[16] Rublee, E., *et al*. ORB: An efficient alternative to SIFT or SURF. *IEEE Computer Vision* (ICCV), 2011.
[17] Hashimoto, M., Y. Domae, and S.i. Kaneko, Current Status and Future Trends on Robot Vision Technology. *Journal of Robotics and Mechatronics*, 29 (2), pp. 275-286, 2017.
[18] Garrido-Jurado, S., *et al*., Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, 47 (6), pp. 2280-2292, 2014.
[19] Zhang, Z., A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22 (11), pp. 1330-1334, 2000.

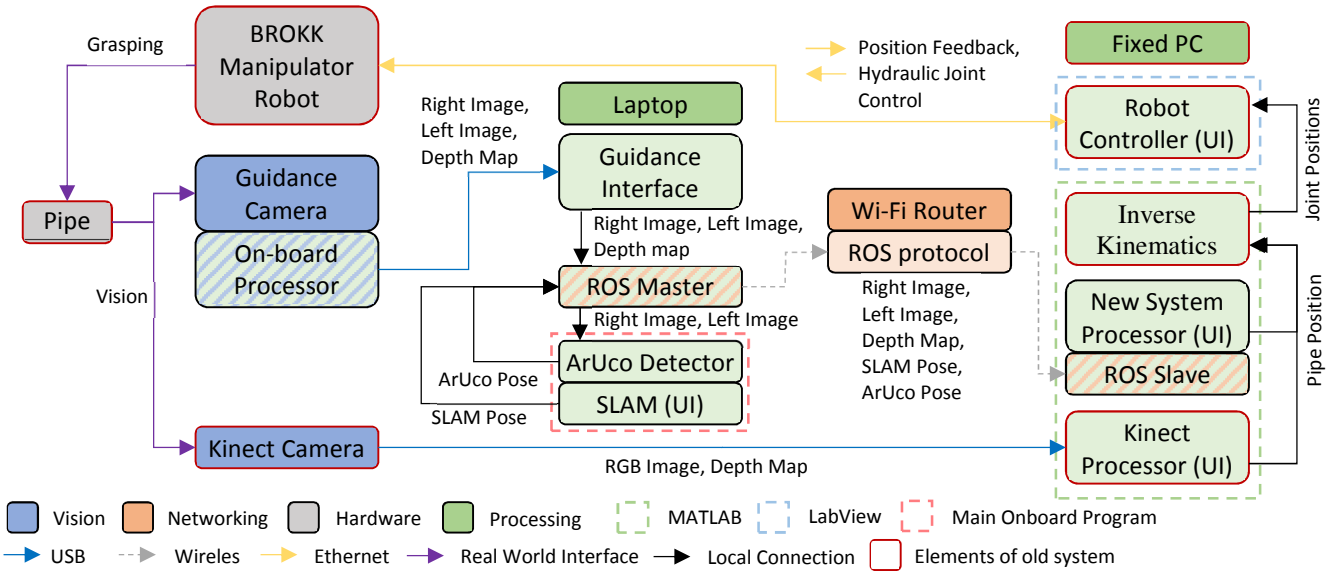


Fig. 7. Flow of information between all elements of the experiment.