Determining Positions and Distances Using Collaborative Robots

Louis G. Clift, Adrian F. Clark
School of Computer Science and Electronic Engineering
University of Essex
Email: {lclift, alien}@essex.ac.uk

***Abstract*—This paper addresses the problem of accurate position estimation in both indoor and outdoor environments. It investigates the use of relative positioning using multiple robots to determine whether local landmarks can produce accurate results in unknown environments that lack external data such as maps or GPS. Specifically, it uses 2D barcodes as fiducials on mobile ground robots, observed by an aerial robot, to provide information such as altitude and pose to the aircraft's navigation stack and to control the motion of the ground robots. The accuracy of this approach is assessed and its potential for use in unknown, hazardous environments discussed.**

Keywords—UAV, 2D barcode, position monitoring, localisation, visual odometry

I. Introduction

Accurate position determination is a continual problem in robotics. Although there are many competing technologies, all have shortcomings that restrict their range of use. For example, conventional odometry may be acceptable for ground robots (though it suffers from problems such as wheel slippage) but is clearly inappropriate for aerial robots (drones). GPS, the Global Positioning System, can provide latitude and longitude but cannot be used successfully indoors and is too inaccurate for many uses with a root-mean-square (RMS) error of $\approx \pm 2$ m [1]. Visual odometry [2], [3] and its related technique of visual SLAM [4], which both use video streams from a camera, can perform well but require the matching of visual features at video rate, quite a difficult task. The state of the art is arguably provided by time-of-flight systems; while these can operate in any environment, they are expensive and cumbersome and this makes them unattractive for robotics. Nevertheless, they have become the *de facto* standard for capturing 3D models of archaeological and heritage sites [5].

Where one has the ability to instrument the environment, one can do a little better. Multiple-camera systems such as those sold by Vicon have become the industry standard for capturing human motion and involve attaching reflective markers to the objects being tracked [6]. Placing fiducials (markers) at known positions in the environment allows one to determine where one is relative to them (*e.g.*, [7]). In practice, installations such as these can be used only indoors.

This work is motivated by the wish to develop a technology that allows 3D shape to be captured in either indoor or outdoor environments using a combination of ground-based robots and drones, a field of research becoming known as *collaborative robotics*. The intention is for them to build a 3D model of an environment largely autonomously, with application in hazardous environments such as collapsed buildings or in the vicinity of fires or road traffic accidents. Positioning technology in this kind of situation needs to be quick to deploy and devoid of any need to instrument the environment.

The approach used to construct 3D models from images is *visual structure from motion* (SFM) (*e.g.*, [8]). Strictly, this can operate without any knowledge of where the images were captured. However, having such information available can help improve shape determination; and it is needed in order to command robots to move to positions that allow them to capture images to fill voids in the 3D model.

A technology that is attractive in principle is to attach fiducials to the various robots, and then to determine position relative to those markers. This approach employs only low-cost cameras, which can be used on ground or airborne vehicles, and is equally effective indoors or outside. The particular approach used here is to attach QR code fiducials [9] to the upper surfaces of ground robots so that a drone may determine its position relative to one or more of them. When a ground robot needs to move through a particular distance in a particular direction, the drone needs to hover (relative to other fiducials or to ground features) until it perceives the distance traversed to be correct. Clearly, there are inaccuracies in each stage of this scheme, and the principal purpose of this paper is to establish whether the errors introduced are random, and hence tend to cancel out over several movements, or systematic [10].

The remainder of this paper is structured as follows. Section II describes the principles of camera-based position determination used in this work. This is followed by section III which explains how this work fits into the field of collaborative robotics. Section IV describes the experimental set-up used to capture imagery, and the positioning accuracy resulting from it presented in section V. Finally, section VI draws conclusions.

II. Sensing Depth with Vision

Acquiring depth from images is a well-established problem in computer vision. The most common approach is to use two or more cameras, known as *computational stereo* for two cameras or *multi-view reconstruction* for many [11]. Recovering depth from stereo images relies on knowing the *intrinsic camera parameters* (focal length, sensor size, *etc*) and their arrangement in the environment (*extrinsic* parameters) —

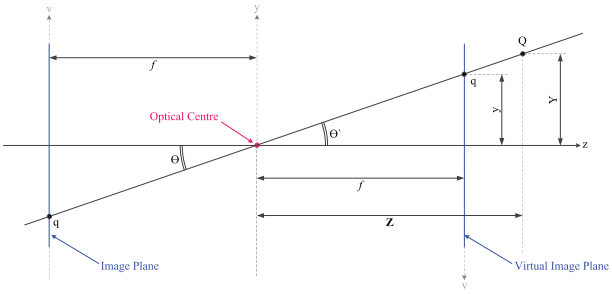

Fig. 1. Depth recovery of point 'P' using a pin-hole camera model

these can be found by calibration. When calibrated, a stereo camera system is accurate and able to operate in real time; the Vicon system mentioned above uses this approach.

Depth can also be extracted from a single camera source. The methods of extracting depth from a monocular camera are more complicated and typically require more information about the scene. However, SFM uses the motion of a single camera to mimic a computational stereo configuration and replaces the calibration stage with a global optimisation; the only information needed is the focal length and the size of the sensor in the camera.

To measure the altitude of the image we assume that the camera is facing downward from the base of the aircraft. The distance from the image plane to the camera can be calculated by using equations valid for a pin-hole camera where Z is the distance between the camera and the QR code; it is assumed that camera distortions are small enough to be ignored. Figure 1 shows the 'virtual image plane.' The relationship between a point seen in 3-D space, $Q = (X, Y)^T$, and the point seen in the corresponding 2-D image, $q = (x, y)^T$, can be modelled as

$$\begin{aligned} x &= \frac{fX}{Z} + t_x \\ y &= \frac{fY}{Z} + t_y \end{aligned} \tag{1}$$

where Z is the distance, f the focal length of the lens and t_x, t_y the optical axis offset. Digital images are captured with the origin in the top left-hand corner in the same way as a computer display starts from $(0, 0)$ in the top left. The offset shifts the origin back to the image centre to align with the optical centre of the camera.

A camera lens will be characterised by a single focal length measurement, though there are usually distortions present that introduce effects such as astigmatism, coma and distortion. The focal length needs to be converted into pixel units or *vice vers* for the algorithms to return a result that relates to the real world. The pin-hole model can be generalised to account for non-square sensors. The focal length (in millimetres) can be converted into pixel values using

$$\begin{aligned} f_x &= \frac{f}{\left(\frac{S_w}{I_w}\right)} \\ f_y &= \frac{f}{\left(\frac{S_h}{I_h}\right)} \end{aligned} \tag{2}$$

where $f_{x,y}$ is the focal length in pixels, f the focal length in millimetres, $S_{w,h}$ the width and height of the sensor in millimetres and $I_{w,h}$ the width and height of the image measured in pixels. A generalised camera model can then be represented by the matrix

$$\begin{bmatrix} \alpha_u & S & t_u \\ 0 & \alpha_v & t_v \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

where α_u, α_v are the focal lengths f_x, f_y from equation 2 and $t_{u,v}$ are the offsets in x and y respectively. These relationships enable the calculation of the distance of the QR target from the camera without needing pose information from the aerial vehicle.

Assuming that the camera has negligible distortion, one can use (2) and (1) to recover the depth of any point seen in an image. Most uncalibrated cameras will preform reasonably well if the region of interest is in the centre of the image. All cameras have a varying degree of distortion to them which can be corrected through calibration. There are several tools available to preform this task such as those in OpenCV and Matlab [12]. Calibration produces two matrices, the intrinsic parameters as shown in (3), and the distortion matrix. These matrices enable a pre-processing stage to correct the image and enable more accurate measurements to be calculated.

The algorithms discussed so far are based on the principles of stereo vision and require a disparity between two or more frames. Depth can also be recovered from a single image given that a known object is in the view. In a similar method to calibration, given a target of known shape and dimension, the distance to the object can be calculated:

$$\begin{aligned} Z_x &= \frac{Wf_x}{w} \\ Z_y &= \frac{Wf_y}{w} \end{aligned} \tag{4}$$

where W is the known width in millimetres, f_x, f_y the focal length in pixels and Z_x, Z_y are the calculated distances to the object. In this work Z_x, Z_y are then averaged together to return a combined result to reduce the effect of a non-square image due to perspective effects.

III. Collaborative Robotics

The notion of using fudicials as landmarks is not new in robotics; however, in this work the landmarks are the robots themselves. An aerial observer looks down at the ground team and uses the markers attached to the robots to provide the pose information back to the control system. A (human) controller provides high-level control to guide the aerial vehicle into a new environment with the support of one or more ground vehicle.

SLAM is used to explore an unknown environment, however these maps are generated with respect to the local coordinate frame as a global map is unavailable. This makes multi-robot SLAM a challenging task. In this work the aircraft uses QR codes to locate and determine the position of the ground robots. Given that there can be multiple vehicles in the environment we use the aerial view to orient each of the robot's sensor information in relation to the aircraft's

Fig. 2. Aerial View of ground team

TABLE I
QR READABILITY FROM A STATIC CAMERA

Camera	Sensor	Large QR	Small QR
DraganFly EyeCam	480×320	1.2m	0.5m
AR Drone Parrot 2	320×240	0.9m	0.6m
TP-Link IP CCTV Camera	1028×720	1.9m	1.5m
Samsung Galaxy S4	4128×3096	2.85m	1.8m

coordinate frame. This approach enables the global model to build faster as multiple views are captured simultaneously with a common coordinate frame.

IV. Experimental Procedure

The ground robots used in this study were equipped with QR ("Quick Response") 2D barcodes of size 17×17 cm on their upper surfaces. QR codes [9] are now in fairly widespread use because they contain only black and white regions and can be read (with appropriate software) using a conventional camera. They contain three 'finder' patterns in their corners which help in distinguishing the QR code from visual clutter, and the rest of the pattern encodes a series of characters. The 'finder' patterns allow the orientation of a QR code (and hence the vehicle to which it is attached) to be determined. In this work, QR codes with the largest pattern size and the most redundancy were used, encoding strings such as "UGV-1".

An open source library, ZBar [13], was used for reading QR codes. This interfaces to the popular OpenCV software, though it is necessary to use OpenCV to pre-process and manage the incoming image streams before presenting them to the routines that finds and decodes a bar code.

A. Determining the maximum usable altitude

The higher a drone flies, the smaller the QR code appears in the field of view of a downward-pointing camera attached to it. At some point, the pattern becomes too small to distinguish the individual black and white regions that comprise it and reading fails. Of course, different cameras have different focal lengths and sensors. Table I presents the maximum usable altitude for several popular cameras, measured using a Vicon MX motion capture system with an accuracy of ± 5 mm. The camera and the drone on which it was mounted were clamped to a camera crane, so practically usable altitudes will be lower than this due to effects such as motor vibration. It is interesting to note that the maximum usable altitude is generally *below* head height, a fact that is unattractive for using drones in emergency situations, where there are usually people milling about.

B. Measuring altitude and distance

Initial experiments used a live video stream from the camera into the control computer, which was running ROS, the Robotic Operating System [14]. ROS is a framework designed to work with a range of robots and provides and extensive range of modules designed to support the development of robotic platforms. In this work we use ROS to both control and utilise the on-board cameras of the UAVs, as well as capture the data being published from a Vicon system (to provide global 'ground truth') and the QR tracking software. The control software written for these experiments integrates with ROS to subscribe to the images and control topics, an approach that minimises latency and removes the need to transport images between packages. The full solution is capable of controlling the aircraft in real time to hover over a QR code. The IP CCTV and wireless DraganFly cameras were fed directly into the application software, which performs its processing using OpenCV; but this was not the case on the AR drone Parrot as its video feed was not exposed as a ROS stream.

V. Results

To evaluate the practical performance of the QR code reader these experiments use the downward facing camera of an AR Drone Parrot 2.0. The initial experiments required that the UAV remain static and explore the use of the sensor in an uncalibrated state. To achieve reliable results the UAV was clamped to a camera crane to enable the aircraft to be held at a range of fixed and stable altitudes in order to obtain best-case decoding frequency measurements as seen in figure 3.

The decoder, ZBar, is designed to run at video rate; however during testing and development it was observed that the library was not always able to decode a QR code when clearly visible in the image. Each of the cameras in table I was 'flown' above two QR codes while streaming imagery to the library. All of the cameras experienced the same issue of occasional cases where a clearly-visible code was not decoded. The results in figure 3 show the rate at which the software is able decode and the update the altitude estimation from the aircraft's on-board camera. It can be seen that the frequency varies as the distance increases until the aircraft reaches a critical point at which the update frequency of the decoder falls away. It has been established that this is due the contrast in the image becoming too poor to distinguish the blocks of the QR pattern.

The QR size ultimately determines both the minimum and maximum altitudes of the aircraft due to the field of view of the camera. Figure 4 clearly shows that there is a linear relationship between the altitude of the aircraft and the error in the height estimation from the QR code, regardless of the size of the QR code used. The results were obtained from an uncalibrated camera. Interestingly there appears to be a consistent gain in error for every 100 mm of altitude. The minimum reading altitude for the smaller QR was observed

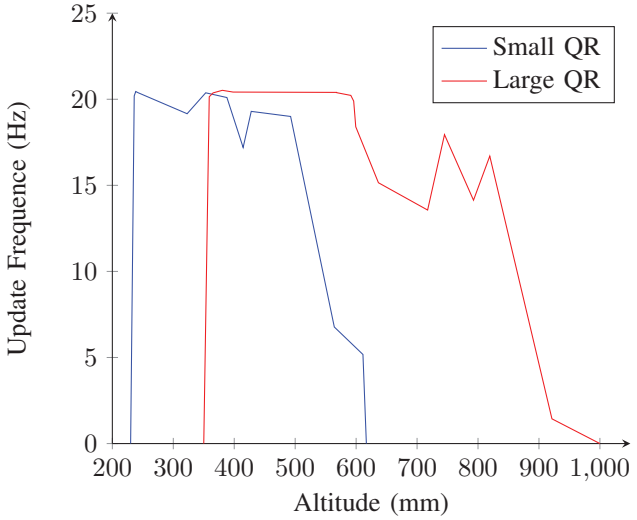

Fig. 3. Altitude update frequency from QR codes

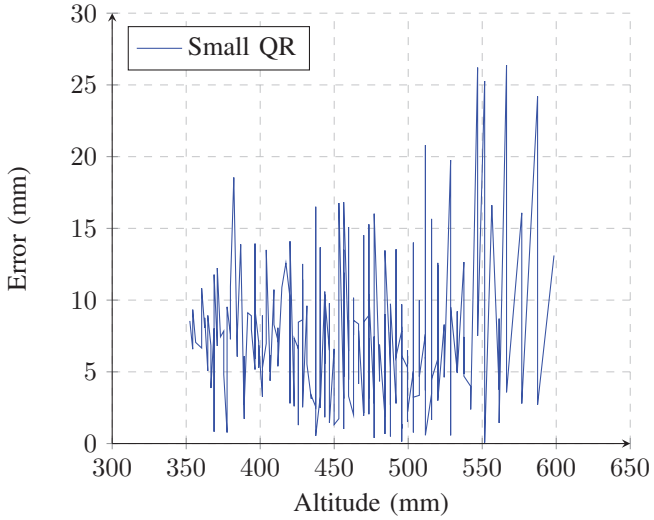

Fig. 5. Small QR measured error after calibration

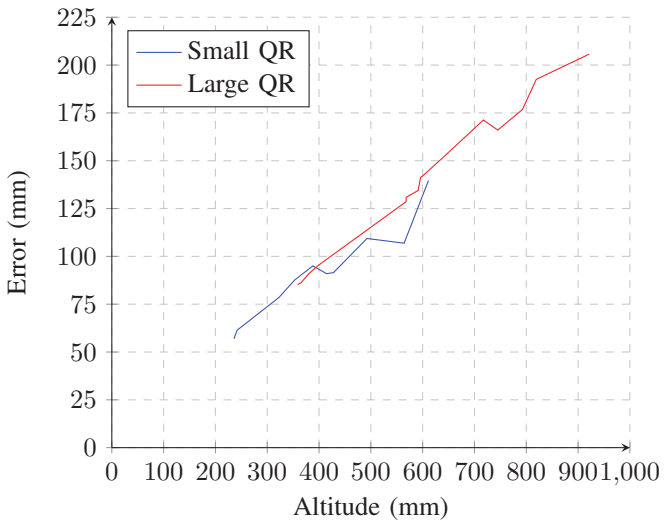

Fig. 4. Altitude Estimation Error

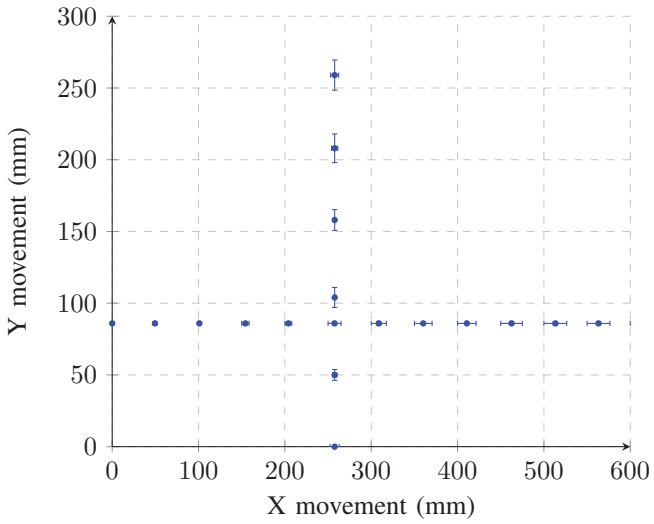

Fig. 6. QR Tracking from level aircraft

at 235 mm, and at 360 mm for the larger one. Once the system is calibrated (using 100 frames captured whilst flying) the results improve significantly. Figure 5 shows that after calibration (and distortion correction), the error reduces to a peak of 25 mm and is fairly consistent throughout the range. Unfortunately calibration does not enable reading to be achieved at a greater altitude.

To evaluate the accuracy of measuring ground movements, the aircraft was fixed at an altitude of 870 mm, just inside the maximum altitude to offer the widest range of movement. This gives a field of view on the ground of 600 mm $\times$ 300 mm, and the QR codes are 170 mm wide. Figure 6 shows that the error is fairly consistent across the range of movement; however, the results in figure 7 demonstrate that accurate estimations require the aircraft to be level with the target. Figure 7 shows that the error increases with the roll of the aircraft. The aircraft was tilted approximately 5° to the right.

The results of these experiments with a static aircraft demonstrate that, providing that the aircraft can maintain a stable altitude of up to 1 metre, a QR code can provide an altitude measurement with an accuracy of ± 200 mm in Z and relative position information with an accuracy of ± 20 mm in X and Y. Figures 8 and 9 demonstrate that the system becomes more accurate when working with undistorted images. The error reduces to approximately ± 5 mm error in the x plane of the image and an mean of ± 2 mm in y. However, in the real world, UAVs are unstable and tend to vibrate when hovering. In the next experiment, the UAV is flown above the large QR code without any stabilisation or support to assess the worst case performance. Figure 10 evaluates the accuracy of the altitude measurement when flying in realistic conditions.

The aircraft is naturally unstable and has a tenancy to drift

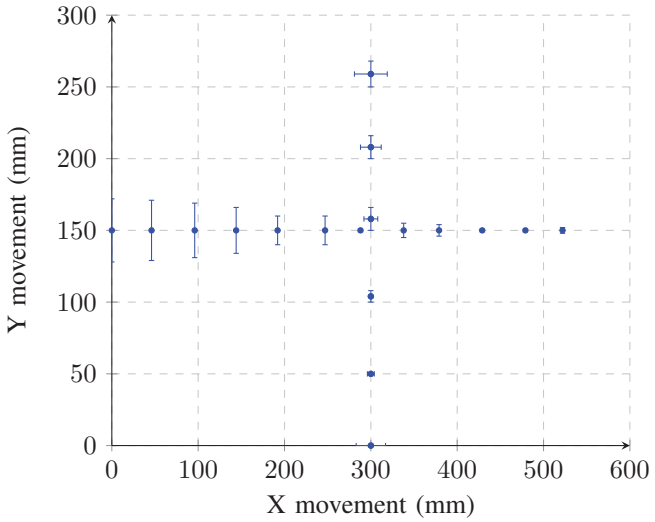

Fig. 7. QR tacking with aircraft at 5° roll

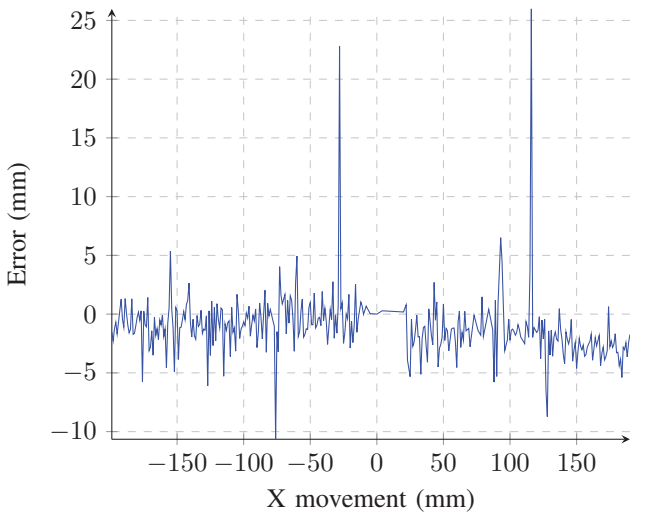

Fig. 8. QR moving in the image x plane with a calibrated camera source

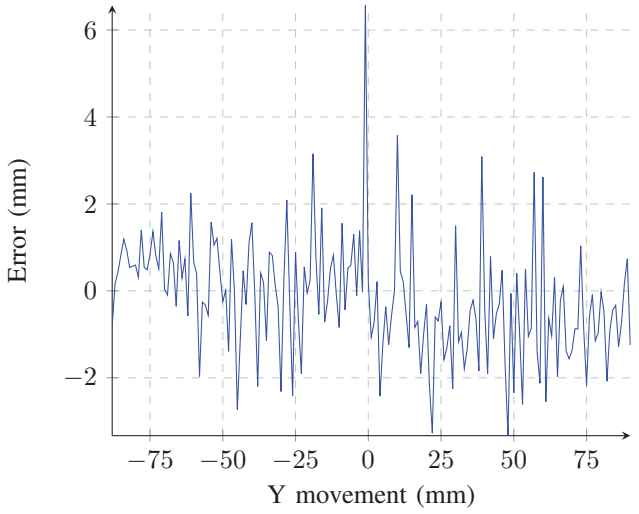

Fig. 9. QR moving in the image x plane with a calibrated camera source

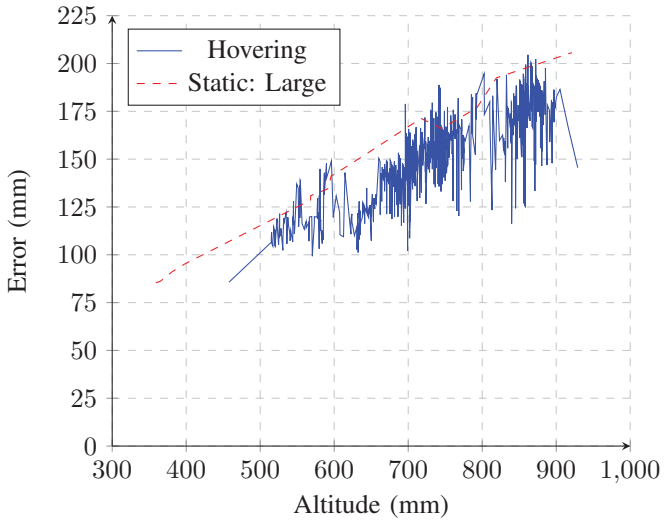

Fig. 10. Error in Reading Altitude During Flight (Uncalibrated)

when left to hover unassisted. The aircraft was held in position over the QR with pilot assistance to reduce the 'wandering' behaviour. Even with a pilot, the aircraft wandered across the target multiple times. The results shown in figure 10 show that it was possible to both decode and track QR codes whilst airborne. Using the live feedback from the QR position, the aircraft is able to hover unaided and maintain pose and altitude above the target with the aid of control loops using the QR as feedback. The error in the altitude remains consistent with the linear error trend from the static tests, as shown by the red dashed line. The results are noisy as the raw measurements from a test flight have been plotted. The aircraft pitches and rolls to maintain position over the target which introduces error as seen earlier in figure 7. The minimum read height decreases to 457 mm due to the unstable hovering and needing to see the whole QR target in frame to decode.

VI. Conclusions

The principal aim of this work was to establish whether a series of steps, each of which involves the motion of one robot relative to others, is able to yield a reasonably accurate determination of the distance travelled.

The results have shown that, given a fairly stable aerial platform, QR codes can be used to provide relative position information with an accuracy up to 20 mm in the 2D X-Y plane. These experiments have identified the series of steps needed to be able to achieve collaborative ground vehicle tracking and management without external sensors. This approach is limited by the quality and resolution of the camera available; however, the results have shown that even a low-resolution sensor is able to achieve an accuracy of 20 mm from a height of one metre. Camera systems with longer focal lengths will enable control to be achieved from higher altitudes; this is desirable because

the aircraft needs to be able to operate above head height and cover a wider ground area.

ACKNOWLEDGEMENTS

The first author acknowledges the receipt of EPSRC doctoral training funding to support his work.

REFERENCES

[1] P. Ritsos and A. F. Clark, "Engineering an augmented reality tour guide," in *Proceedings of Eurowearables '03*, Sep. 2003, pp. 119–124.

[2] J. Campbell, R. Sukthankar, I. Nourbakhsh, and A. Pahwa, "A robust visual odometry and precipice detection system using consumer-grade monocular vision," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, Apr. 2005.

[3] M. Labbé and F. Michaud, "Online global loop closure detection for large-scale multi-session graph-based slam," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014.

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[5] "Cyark," http://www.cyark.org/, accessed: 2015-06-25.

[6] "Vicon motion capture systems," http://www.vicon.com/products/camera-systems, accessed: 2015-06-25.

[7] D. J. Johnston and A. F. Clark, "A vision-based location system using fiducials," in *Proceedings of the First International Conference on Vision, Video and Graphics*, Jul. 2003, pp. 159–166.

[8] K. Häming and G. Peters, ""the structure-from-motion reconstruction pipeline — a survey with focus on short image sequences," *Kybernetika*, 2010.

[9] "QR code features," http://archive.is/20120915040047/http://www.qrcode.com/en/qrfeature.html, accessed: 2015-06-25.

[10] G. L. Squires, *Practical Physics*, 4th ed. Cambridge University Press, 2001.

[11] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, Mar. 2004.

[12] J. Y. Bouguet, "Camera calibration toolbox for Matlab," 2008. [Online]. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/.

[13] "Zbar bar code reader," http://zbar.sourceforge.net/index.html, accessed: 2015-06-25.

[14] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. B. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, 2009.