Multi camera pose aided docking of a non-holonomic robot

Anthony Tompkins (309 187 893) and Giovanni D'urso (310 187 230)

School of Aerospace, Mechanical and Mechatronic Engineering
Email: atom7176@uni.sydney.edu.au, gdur6497@uni.sydney.edu.au

Abstract

Docking is an increasingly desired ability for robots as autonomous operation becomes more prevalent. Accurate feedback is essential for providing the grounds for a robot to dock autonomously and so we explore the design process for developing a reliable and robust identifiable encoded marker (fiducial) system, a method for integrating two independent sets of pose data consisting of both topographical global information and localized pose information with respect to a custom made docking panel, and pathing of a non holonomic rover with the fusion of the multiple data streams to consistently dock accurately. Our system operates in various adverse lighting conditions and provides significant pose accuracy at close range to the designated docking panel through solving the perspective-n-point given appropriate experimentally considered boundary constraints such as extreme angle and distance limitations.

1 Introduction

Docking is a critical problem for mobile robots specifically ones that need to operate for extended periods of time or move to a particular position in space to deliver a payload because if it does not dock accurately and reliably it will not be able to correctly deliver a payload or recharge. In response to the above, we require a higher level of precision and accuracy in a system robust and invariant to perturbations.

This paper specifically focuses on the docking problem in the realm of non-holonomic rovers using multiple cameras for closed loop feedback in order to achieve the goal of docking. Not all robotic designs allow or have a holonomic mobility system which means that a large portion require a more tailored approach in order to dock under the constraint of being non-holonomic. Therefore, the system displayed in this paper may be applied to a large variety of robots as visual systems do not require significant infrastructure. Thus it can even be applied to non terrestrial robots where some conventional positioning systems such as GPS will fail due to the lack of appropriate infrastructure.

In the following sections this paper will cover the design of the fiducials used in the visual pose estimation system, the design and application of the PID controller, the visual pose estimation system, and the data fusion used to allow multiple cameras to increase the accuracy of the system.

2 Background

2.1 Fiducial markers and visual pose estimation

Fiducial markers are identifiers that allow a system to associate observed data with other known information. In terms of visual pose fiducial makers these often take the form of optically identifiable structures such as patterns.

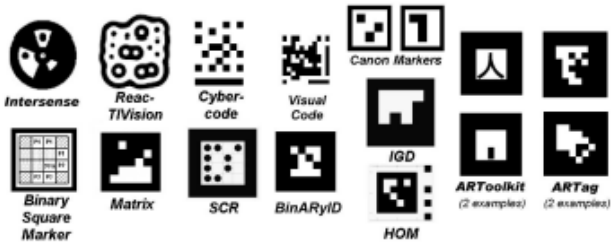

Figure 1. Various types of fiducials [1]

Common uses of fiducial markers are for indoor augmented reality, and pose determination [1]. Fiducial markers are often rated on detection reliability and tolerance in poor lighting conditions. Existing systems exist in forms such as bar-codes (Data matrix, Maxicode, Quick Response) [1] but are not designed specifically for dealing with large fields of view and significant perspective distortion.

There are various methods for solving the visual pose estimation problem such as Posit[2], perspective-n-point[3-5], ellipse back projection and pose estimation[6]; all of which are computationally fast and operate favorably for most visual fiducial based solutions.

2.2 Vision based docking

Docking allows robots to advance their level of self-sustainability by removing the requirement for an external operator to accurately position a robot in a required way. A common requirement for docking robots is to dock at a recharging station in order to renew energy supply and thus decrease external reliance and increase autonomous capability. Common commercially available solutions include a recharging station for the ActivMedia Pioneer robot[7], the Helpmate [8] which has been utilized in

hospitals, and curious creative solutions such as the PR2 beer grabbing robot [9]. All of which implement visual feedback in order to align themselves to their appropriate fine tuned docking actions. Other motivations for using vision based docking is the low power, cost, and size requirements for successful docking; a novel example of this is a biologically inspired method which utilizes only visual image data with no apriori knowledge of the docking scene [10], thereby exemplifying benefits of visual feedback.

2.3 PID control scheme

$$\boldsymbol{PID} = \boldsymbol{K_p} \times \boldsymbol{e(t)} + \boldsymbol{K_i} \int_0^t \boldsymbol{e(t)dt} + \boldsymbol{K_d} \frac{\boldsymbol{d}}{\boldsymbol{dt}} \boldsymbol{e(t)}$$

A wide variety of applications employ PID controllers to regulate or control their systems, the benefit of a PID controller is that you do not need to entirely understand the underlying system dynamics which may be extremely complex. Instead you tune the system output to what you require by tuning the amount of feedback provided by adjusting the tuning parameters of the controller. The system tracks a desired output based on how the error function is defined because the PID controller applies feedback based on the amount of error. [11]
[12] used a PID controller on their robot arm to control the amount of force overtime and analyzed the response of the system based on different tuning parameters. Using PID for a speed control using only speed feedback was demonstrated to be successful [13] and by carefully tuning the PID gains they were able to make PID controller track the desired speed with minimum oscillations.
From these examples and their widespread application throughout industry it can be seen that a PID controller when applied correctly can be a very simple but effective way of controlling a system.

2.4 Nonholonomic control

A robot does not need to know if it is holonomic in order to plan its actions to reach some goal. But if you use traditional methods of robot path planning they will not be applicable or able to solve problems faced by nonholonomic systems. These problems arise from the fact that the system has fewer controls than configuration variables[14]. For example a car has two controllable variables (linear velocity and turning angle thus angular velocity) ,but it operates in a three dimensional configuration space (X,Y,Theta). Because of this not every path in the configuration space does not correspond to a robot feasible path in the work space. Thus in order to control nonholonomic robots accurately the approach towards motion planning needs to consider the system constraints such as turning angle and position at points in time in order to achieve moving to a desired position and goal. Due to these constraints the method of which we path plan and follow in order to dock our robot needs to allow for these constraints.

3 Experimental design and setup

3.1 Overall system architecture

Figure 2. Overview of the whole system consisting of: Topview camera mounted on tripod, 3x workspace fiducials, 1x iRobot with docking camera, docking fiducials

This experiment's purpose is to develop a closed loop visual feedback docking system. The system needs to be as robust as possible, hence we decided on a multiple camera system to allow noise in the global pose whilst still allowing docking to happen in the expected position in the docking frame of reference. The hardware used in this system is an iRobot, two laptops and two cameras as well as visual fiducials.

Hardware

A fiducial marker is attached to the top of the mobile base so the global pose estimation system can identify the robot and calculate its position in global space. An iRobot was used as the mobile base because it allowed us to quickly prototype our system and is controlled by serial commands sent over a USB TTL serial link with MAX232. This allows the computer to transmit serial byte commands to the iRobot which has a nicely defined API. Another advantage of the iRobot over building our own system is it already includes encoders, an internal speed controller and has a nice gearbox and wheels thus allowing us to utilize a reliable and easy to control base with little to no slip on the test linoleum. F-100 wide angle cameras were used because they have a 120 degree FOV, and 1080p at 30 FPS which allows the camera to work at a sharp angle between the fiducial and the camera as well as at a range of 6-10 M that is reliable with the current design of fiducial; as well as allowing a reasonably high update rate for a visual system.

A laptop with an i5 processor is attached to the iRobot in order to process the docking cameras and run the PID controller and is connected via wifi to a ground station computer that processes the global camera and transmits values to the iRobot computer

Software

The software consists of several subsystems, the communications system talks to the iRobot and to the ground control laptop, the visual pose system calculates the robots position relative to known positions of fiducial markers and the PID controller calculates the set of drive commands required to move the robot to the requested position.

The iRobot is controlled via a TTL serial link using serial byte commands which are defined in the Create open interface. So for example in order to drive the robot forward an opcode of 137 is transmitted followed by 4 data bytes which define the speed and radius of circle the robot should drive.

The laptops communicate via wifi link using the UDP protocol. This is preferred because the latency of packets is important, whilst the reliability is less so because if a packet is lost it is better to get a new one rather than lock the system down waiting for a new packet as would happen in TCP. Also given that the system was designed to work in proximity to the cameras the distance between the wireless router and robot is low and hence the packet loss is negligible. The computers interact via a set of encoded string commands and if a command isn't recognized it is rejected and has no affect. The iRobot waits for localization data to determine movement direction, which is encoded as a 3 tuple of position x, position y and bearing [x,y,theta] (in the global frame) before it starts moving in order to know where to move to otherwise there is no point moving. If at any time the iRobot receives a shutdown command it will cease moving and stop the current running code, this was added as a safety precaution as a wireless emergency stop in case it was needed.

The visual pose system uses a set of two cameras to detect the position and orientation (pose) of the markers in order to both define the workspace in terms of pixel and equivalent real world coordinates. This visual pose system is subdivided into two subsystems: an global topographical pose system, and a docking specific camera system.

In order to allow the robot to follow paths we implemented the PID control loop controller which calculates the error between the robots current position and angle and its ideal position and angle then combines the proportional , integral and derivative controllers to help it track the input and increase robustness in input/output error

3.2 Fiducial visual tracking and identification

Overall system design considerations

There are various considerations which define qualities of a fiducial which when optimized can increase robustness and effectiveness. Some of these have been outlined in [1] and have been used as primary factors in our system design. We considered the following:

False positive rate – return of a valid fiducial when none exists.

False negative rate – lack of return of a valid fiducial when one exists.

Immunity to lighting conditions – ability to detect and identify fiducials in adverse lighting scenarios such as large differences in brightness.

Speed performance – the ability of the system to operate at fast speeds applicable to real time application.

Marker size – maximization of detection and identification ability.

In addition to the above there were extra factors that needed to be considered due to the time constrained nature of the project as well as limited resource available:

Ease of manufacture – ability to fabricate accurately, cheaply, and efficiently.

Rotation identification – The relative rotation of the fiducial must be able to be determined for: Start and stop bit selection, and fiducial relative rotation.

Fiducial marker design

In our fiducial marker design we have implemented a circular marker with a 4-bit encode. This gives $2^4 = 16$ identities to work with. The rationale with choosing a relatively low set of unique identities was based a minimal clutter area (lab floor) of our workspace and since we only required a total of 8 unique fiducials to complete the system we have a maximum separation of 2. A more noise resistant system would have incorporated significantly higher quantities of unique identities to reduce the false positive rate because of higher value separation (distinctions of > 100 between fiducial identities as opposed to just 2)

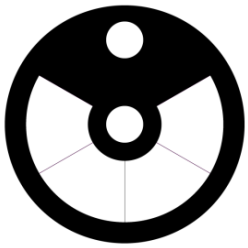

Figure 3. Our final 4-bit fiducial design

Another requirement of our fiducial design was that it must incorporate the ability to detect relative rotation of the fiducial itself as one was mounted on the robot and was value critical to the robot pose. Thus, the two circular bits in the middle and edge provide the ability to distinguish rotation of the fiducial.

Detection, tracking and decoding

The detection and tracking stage for fiducial recognition is depicted visually as follows.

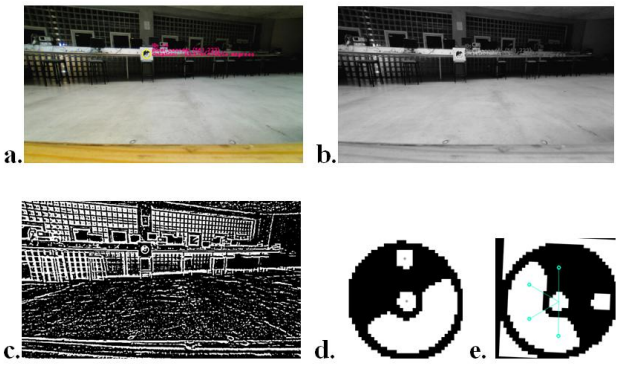

Figure 4. Steps for detecting and decoding fiducials: a. Raw frame, b. Grayscale conversion, c. Adaptive thresholding, d. Various metric based conditional filtering, e. Radial binary

sampling to obtain 4-bit value

Thus a detail process is as follows:

1. *Grayscale conversion*: To speed up processing and prepare the image for adaptive thresholding.

2. *Adaptive thresholding*: In order to reduce effects of inconsistent lighting scenarios and remove the necessity of global thresholding, adaptive thresholding on a per pixel basis was implemented.
3. *Morphological erosion*: To remove small outliers and speed up processing for the following contour detection.

4. *Connected neighbour contour detection:* On the resulting image contours of all the blobs in the image are obtained for further filtering. Experimentally obtained sizes of contours (> 100 px^2 and < 60000 px^2) were obtained to define those that represent expected fiducials. Contour areas outside of these thresholds are ignored.

5. *Least squares rectangle fitting:* To create an approximation bounding rectangle.

6. *Area ratio metric* [15]:

$$\left(\frac{\pi}{4}\right)(l_1 l_2)$$

Where:
l_1 = length of bounding rect side 1
l_2 = length of bounding rect side 2

6. *Contour convexity check*: To eliminate those with convexity defects.

7. *Roundness ratio check*: A metric to determine roundness.

$$\frac{4\pi a}{p^2}$$

Where :
a = area of contour
p = perimeter of contour
A result of 1 refers to a perfect circle, ~0.8 refers to a smooth ellipse, < 0.8 suggests an imperfect ellipse.

8. *Region of interest (ROI) crop and scale to n x n resolution*: In order to verify the contour is actually a fiducial we scale to a specific size.

9. *Sub contour area check*: For each ROI we detect the contours and check the area of the two smallest and verify they are similar in area equality and between a threshold of 1500 and 2500 px.

10. *Find signed angle and correct for error*: We obtain the relative angle between the line created by the centroids of the sub centroid contours, and the line created by workspace fiducials F_0 and F_1.

11. *Rotate ROI crop by angle found in 10*: To align all the fiducials consistently.

12. *Check the ROI crop* at an offset of 45 degrees to check if there exist any '1' bits. Store these in an array.

13. *Check if binary value exists:* Compare and verify the binary array found by the loop in 12. with a hash table which contains all the valid fiducial identities.

3.3 Docking panel design

The docking panel was created to allow a more accurate ground based perspective to allow a greater accuracy for smoother docking methods. Existing fiducials were used to identify real world coordinates which were then used to fulfill requirements to solve the perspective-n-point localisation model [3-5].

A docking cam view which depicts the docking fiducial panel is as follows

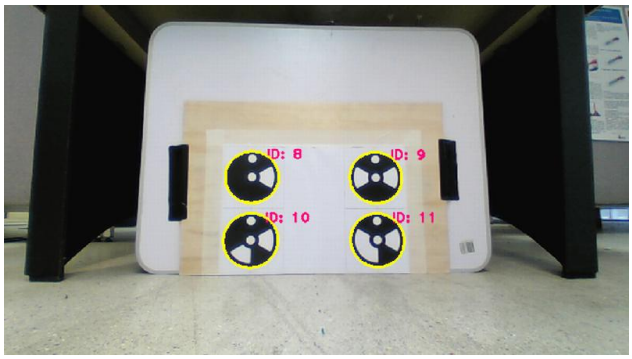

Figure 5. Docking camera viewpoint of docking fiducial panel with four identified fiducials representing known real world coordinates

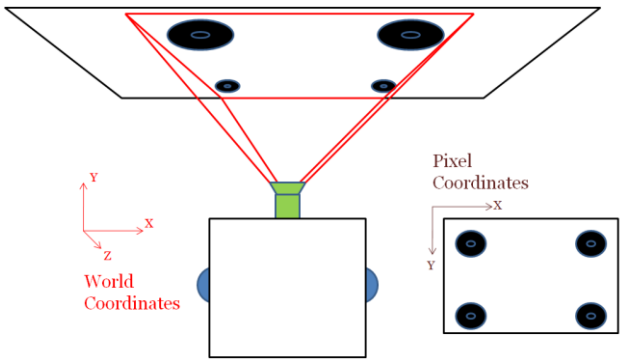

Figure 6. Comparison of global frame of reference with pixel frame of reference of the docking panel

For each observed fiducial, with identities *8, 9, 10, 11*, there exist corresponding real world locations (x,y,z) which allow the pnp model to be solved.

3.4 Visual localization

Visual localization takes on two forms in this project:

1. Topographical localization – Using a birdseye camera to define the pixel and real world workspace relationship for global pose
2. Docking localization – Using the perspective-n-point method to obtain a 3D pose of the robot. Only the Z orientation, and X and Y translations are used in our case.

Before any of the localization solutions were implemented, intrinsic camera calibration as per common camera calibration techniques were acquired with the toolbox by [16]. The camera intrinsics are used initially

for removing the visual distortion from the camera image to allow linear observation of the workspace. The secondary use is for the pinhole camera model to implement the perspective-n-point solution to find object pose relative to the camera.

Topographical localization

In order to find the real world locations of the robot based on pixel coordinates a simple scaling scheme was implemented. A topographical view of our scenario from a real world and diagrammatic perspective are as follows:

Figure 7. Top view real world view of workspace

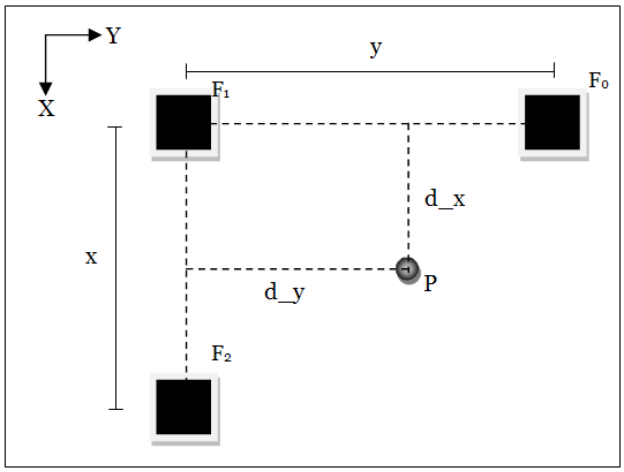

Figure 8. Top view diagrammatic representation of workspace

The boundaries of the workplace are defined by 3 known fiducials F_0, F_1, F_2 and the pose of the robot is defined by pose P which contains an array consisting of x and y translation, and rotation around the z axis. The orientation of the robot fiducial is obtained by finding the angle offset of the line defined by the rotation identifier bits and workspace fiducials F_0 and F_1.

Since we have known fiducial pixel and real world locations that create a right angle, this information can be used to define a scale factor in the X and Y directions.

XScale = RealWorldDist_X / PixelDist_X
YScale = RealWorldDist_Y / PixelDist_Y

Where pixel distances d_x and d_y, and the real world distances D_x and D_y are defined as followed:

d_x = (perpendicular distance of point P with the line defined by F1 and F2)
d_y = (perpendicular distance of point P with the line define by F0 and F1)
D_x = d_x * YScale
D_y = d_y * XScale

Docking panel localisation

The problem of pnp and the pinhole camera model mentioned previously may be observed in the following:

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Figure 9. 3D point projection into image plane using camera intrinsic perspective transformation [17]

Where [17]:
(X,Y,Z) are the coordinates of a 3D point in the world coordinate space
(u,v) are the coordinates of the projection point in pixels
(A) is a camera matrix, or a matrix of intrinsic parameters
(cx,cy) is a principal point that is usually at the image center
f_x and f_y are the focal lengths expressed in pixel units.

In order to solve the problem of perspective transformation we must have known real world coordinates for the docking panel. These values were measured relative to the centre of the docking panel and thus all localisation results would be relative to the local origin on the docking panel.

This results in the issue of having to rely on the physical docking location to be accurate with respect to true global coordinates – even a small inaccuracy between the docking original and the global coordinate system may skew fused pose results when transitioning from global pose to docking based pose. In order to deal with this, the primary assumption is made that once the docking panel is found, and if and only if the docking panel is within a range threshold of < 1m (experimentally determined for accuracy validity), and if and only if the docking pose angle $45^o < \theta < 45^o$ then do we accept the reading. Thus it can be seen we treat the docking panel with a 100% weighting if it is in range to mitigate issues with conflicting pose data with the global (topographic) pose and relative global (docking) pose.

After the pnp solution has converged, we invert the values to obtain the camera pose frame with respect to the object frame. This is insufficient to provide a desired (centre) body pose as the camera is offset from the body pose centre. Thus a standard Euclidean transformation is applied to obtain the true body pose.

3.5 PID control scheme

In our mobility system we have a vehicle with a fast steering model since it is similar to the tricycle model but it implies a minimum turning circle where the change in theta is a function of the turning circle.

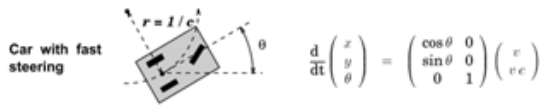

Figure 10. car with fast steering model [18] pg.4 fig 2.

The PID controller can still be used for this model because it can be approximately linearised and can be controlled via PID[19]

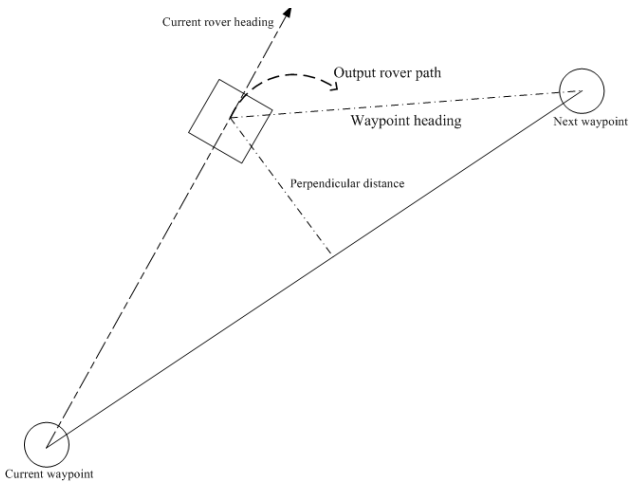

Figure 11. Error definition between waypoints

The error used in the calculation for the PID is a linear combination of the perpendicular distance and error in heading.

$$heading\ error = current\ heading - heading\ to\ waypoint$$

$$distance\ error = current\ position - \ projected\ position$$

$$u(t) = \ distanceWeighting \times (PID_1(distance\ error)) + angleWeighting \ \times (PID_2(heading\ error))$$

These factors are used in calculating the output from the PID controller which is then fed into the wheels to cause a speed difference between the wheels hence the robot will turn in an arc towards the correct position. The controller output has an upper bound and is scaled in order to make the robot adhere to the minimum turn radius constraint that we applied to mimic a non-holonomic system that cannot rotate around its base with the iRobot, this allows it to emulate controlling a system such as a car with Ackerman steering which has a minimum turning circle. Once these systems had been setup it was a matter of doing multiple tests and tuning the several weights until the system achieved acceptable results.

3.6 Nonholonomic control

More complex non-holonomic control was considered, but due to the time constraints of the project we were only able to implement maneuvers which allow a non-holonomic system to imitate a holonomic system. For example, for a non-holonomic system to move parallel it needs to rotate 90^o then drive forward by $\boldsymbol{\delta}$and rotate 90^o back in order to be in the same heading, this can also be done by driving in a set of small arcs as shown below in order to achieve the parallel displacement $\boldsymbol{\delta}$ while maintaining the same heading.

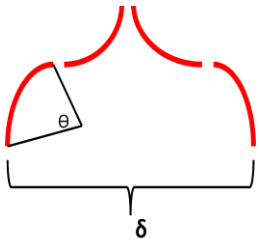

Figure 12. Example of a parallel movement maneuver

These sets of hard coded maneuvers were done in order to allow our non-holonomic system to follow a holonomic path. A better system would be to implement a more extensive non-holonomic planner using a reachability graph of Dubins curves[20] or replacing a non feasible holonomic path with a feasible nonholonomic path[18]. These were considered and attempted, but the implementation did not work due to insufficient time for implementation. However, since path following and planning were not in the original scope of the project they are considered extensions and their incompleteness is not detrimental to the robots ability to dock.

4 Results

This section details the results of the whole system tests

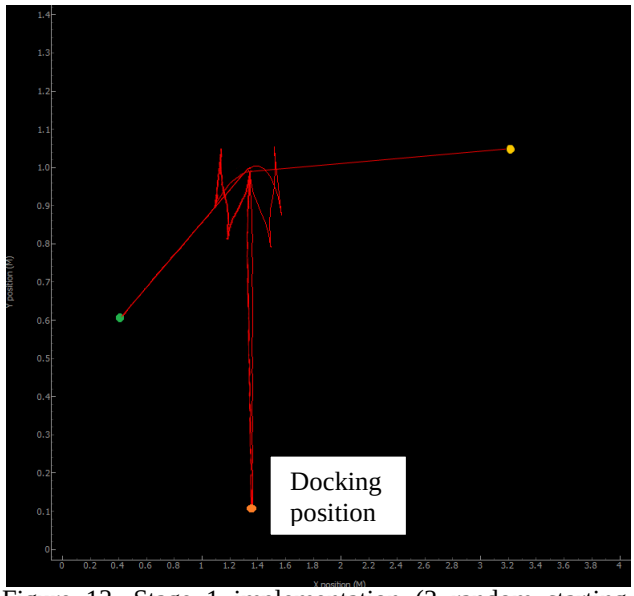

Figure 13. Stage 1 implementation (2 random starting positions showing 2 separate paths followed)

The Stage 1 implementation did not use the docking camera and relied only on the global pose and the pre programmed maneuvers. In this test the robot had to traverse from a random starting position to (1.35, 1) and then proceed to dock at (1.35, 0.15), this was done to simulate the robot following a more complex path before docking.

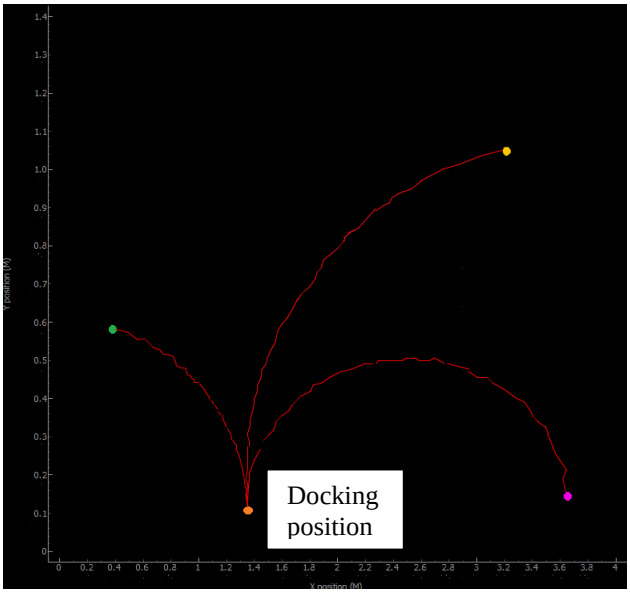

Figure 14. stage 2 implementation (3 random starting positions showing 3 separate paths followed)

The Stage 2 implementation removes redundant maneuvers to increase docking speed and uses docking cameras

The larger oscillations in movement can be attributed to testing with a different computer (slower frame rate on cameras) and then trying to follow a discretised circular path instead of a straight one. In this test the robot went from a random starting position directly to docking mode and proceeded to dock by following a single path.

Comparing results of accuracy with and without docking cam we have:

	Docking camera	No docking camera
Average positional error	0.01m	0.05m
Average angular error	1 degree	0.5 degree

Figure 15. Comparison of average error between topographic and docking localisation systems

These results were found by running the system multiple times with different positions but the same for each set of tests. The measurements were taken by carefully measuring out a grid on the floor and then measuring to the centroid of the robot and the angle between the front and the expected angle position.
The error on the measurements is about 1 degree and 1cm.

Performance of the contour/blob detection method was highly variable dependent on computing setup with used. Two separate platforms were used to run the visual localisation systems and their performance is outline as follows:

Processor:	2.5 GHz i5	1.8GHz Centrino
FPS:	9-12	1-2

Figure 16. Comparison of performance between different CPUs

Similarly, resolution changes were significant in terms of performance.

With the i5 processor:

Resolution:	640x480px	1280x720px	1920x1080px
FPS:	13-18	9-12	2-7

Figure 17. Comparison of performance between different raw acquisition resolutions

5 Discussion

The increase in accuracy down to 1cm resolution shows that the docking camera addition is functional and improves system performance. It also increases robustness because it allows the docking position to be relative to the docking board so if there is noise in the position of the docking board or noise in the global pose system the robot is still able to dock. The limitations are that the system requires a camera with a clear view of all bounding fiducials at all times and a wireless connection between ground station and robot else it will not be able to dock successfully.

A potential source for pose error and noise in measurements lies with the observed centre of the fiducial placed on the robot. Our method of placing the robot fiducial was a visual side view alignment and approximation to the centre of the wheel turning centre. A better system would be to find the geometric centre and position a physical marker on the robot itself. Similarly, when measuring the offset of the camera from the geometric robot centre hand measurements with a ruler were taken and this visual approximation used as the final offset.

With respect to the blob detection, although performance on the i5 processor was good, there is room for significant improvement. Due to the amount of contours created by the adaptive thresholding system, especially at higher resolutions, this single process takes up significant amounts of time for finding ellipses. By considering the performance changes based on the resolution performance logs, it can be inferred a better approach would be to scale down the original raw image, apply stronger morphological processing to reduce noise and outliers, apply fiducial searching, then for each potential valid fiducial blob, scale the encoder check ROI back to the full resolution image.

The repeated successful tests show that we have developed a robust and reliable system, but there is a lot of room for improvement

The PID system can be improved by using an adaptive tuning method instead of a manually tuned one. By doing so, if the system parameters change it will be able function rather than having to run all the tests again and retune the system.

The initial implementation of the nonholonomic maneuvers to decrease error in path following, although functional, added unnecessary complexity and encouraged us to adopt a simpler solution which is the stage 2 implementation. This was found to be successful thus allowing removal of the maneuvers from the final system. By better tuning and implementing the error function for the PID controller we increased the capability of the robot to follow a path to the desired position, reduced complexity and decreased the time it takes to dock.

6 Conclusions and future work

For this setup the system was capable of driving, controlling and docking a mobile robot platform to a specified position and angle within 1cm and 0.5 degrees fairly robustly and reliably. The system can still have errors and is sensitive to changes within the workspace such as a blocked fiducial or obstacles, but there is room for improvement by making the system more robust to those changes. Improvements to make the system a more complete package would be to

- Implement obstacle detection using the onboard/global cameras or a lidar
- Implement a nonholonomic path planner such as exploring a Dubins reachability graph.
- Implement a mapping algorithm on the overhead camera in order to allow the system to plan paths
- Use a more sophisticated fusing algorithm such as an EKF or particle filter to increase reliability and decrease noise in measurements

Other improvements to the vision system for detecting fiducials may also be investigated. In particular, *least squares ellipse fit for centroids approximation* as opposed to raw contour centroids will allow robustness to noise at longer distances due to pixilation which occurs as a function of camera distance from the image. Furthermore it would be appropriate to design a fiducial system with greater than 4 bits of information as well as a redundancy check when decoding the fiducial such that instead of sampling a single point for a particular bit, multiple points are sample and a weighted average is used to determine the bit value. Thus, this extra redundancy will aid in reducing the chance of false positive for the fiducial decoding.

Once these improvements have been made the system will be more robust and able to cope with changes within the workspace such as obstacles and noise on the visual pose system and will be more accurate.

References

[1] M. Fiala, "Designing Highly Reliable Fiducial Markers," *Journal of pattern analysis and machine intelligence*, 2007.

[2] D. F. DeMentho and L. S. Davis, "Model-Based Object Pose in 25 lines of Code," 1995.

[3] J. Fabrizo and J. Devars, "An analytical solution to the perspective-n-point problem for common planar camera and for catadioptric sensor," *International Journal of Image and Graphics*, vol. 8, 2007.

[4] X.-s. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng, "Complete solution classification for the perspective-three-point problem " *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, pp. 930-943, 2003.

[5] F. Moreno-Noguer, V. Lepetit, and P. Fua, "Accurate Non-IterativeO(n)Solution to the PnP Problem," presented at the IEEE International Conference on Computer Vision, Rio de Janeiro, Brazil, 2007.

[6] D. L. d. Ipina, P. R. S. Mendonca, and A. Hopper, "TRIP: a Low-Cost Vision-Based Location System for Ubiquitous Computing."

[7] F. Michaud, J. Audet, D. Letourneau, L. Lussier, C. Theberge-Turmel, and S. Caron, "Autonomous robot that uses symbol recognition and artifical emotionn to attend the AAAI conference," 2000.

[8] S. King and C. Weiman, "Helpmate autonomous mobile navigation system," in *Proceedings of SPIE Conference on Mobile Robots*, Boston, MA, 1990, pp. 190-198.

[9] Willowgarage. (2010). *Beer Me, Robot*. Available: http://www.willowgarage.com/blog/2010/07/06/beer-me-robot

[10] E. M. P. Low, I. R. Manchester, and A. V. Savkin, "A biologically inspired method for vision-based docking of wheeled mobile robots," *Robotics and Autonomous Systems*, vol. 55, pp. 769-784, 2007.

[11] K. J. Åström and R. M. Murray. (2012, 12 Febuary 2013). *Feedback Systems: An Introduction for Scientists and Engineers*. Available: http://www.cds.caltech.edu/~murray/amwiki/index.php/Main_Page

[12] G. I. Selim, N. H. E.-. Amary, and D. M. A. Dahab, "Force Signal Tuning for a Surgical Robotic Arm Using PID Controller," *International Journal of Computer Theory and Engineering*, vol. 4, 2012.

[13] B.-H. Nguyen, H.-B. Ngo, and J.-H. Ryu, "Novel Robust Control Algorithm of DC Motors," presented at the URAI 2009, Korea, 2009.

[14] J. P. Laumond, S. Sekhavat, and F. Lamiraux, "Guidelines in Nonholonomic Motion Planning for Mobile Robots," in *Robot Motion Planning and Control*, J.-P. Laumond, Ed., ed: Springer, 1998.

[15] L. Naimark and E. Foxlin, "Circular Data Matrix Fiducial System and Robust Image Processing for a Wearable Vision-Inertial Self-Tracker," presented at the IEEE International Symposium on Mixed and Augmented Reality, Darmstadt, Germany, 2002.

[16] J.-Y. Bouguet. (2010). *Camera Calibration Toolbox for Matlab*. Available: http://www.vision.caltech.edu/bouguetj/calib_doc/

[17] o. d. team. (2013). *Camera Calibration and 3D Reconstruction*. Available: http://docs.opencv.org/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html

[18] B. Triggs, "Motion Planning for Nonholonomic Vehicles: An Introduction," 2010.

[19] A. D. Luca, G. Oriolo, and C. Samson, "Feedback Control of a Nonholonomic Car-like Robot," in *Robot Motion Planning and Control*, J.-P. Laumond, Ed., ed: Springer, 1999.

[20] S. M. LaValle, *planning algorithms*: Cambridge University Press, 2006.