Vision-based Control of an Encoderless Lightweight Robot Arm

Ho Seok Ahn[1], Forest Fraser[1], Jonathan Lee[1], Kazuki Nomura[2], Hye-Jong Kim[2], Sadao Kawamura[2], Bruce A. MacDonald[1]
[1]Department of Electrical and Computer Engineering, CARES, University of Auckland
Auckland, New Zealand
{hs.ahn, b.macdonald}@auckland.ac.nz, {ffra821, jlee611}@aucklanduni.ac.nz
[2]Ritsumeikan University, Japan
{rr0016vp, rr008083}@ed.ritsumei.ac.jp, kawamura@se.ritsumei.ac.jp

Abstract

This paper presents a vision-based control method for a novel low-cost encoderless flexible lightweight robot arm system, designed to assist people who need help manipulating objects in everyday life. The vision system detects the positions of the joints, using two cameras in an orthogonal configuration that is shown to provide accurate 3D detection after calibration. The lightweight robot arm is a 3-joint non-planar robot, consisting of a soft plastic inflatable arm, pneumatic air valves, a custom PCB with multiple DAC channels, two cameras, a Raspberry Pi and a PC. The lightweight robot arm is centered in the view of two orthogonal cameras to provide measurements of each joint in 3 dimensions. Inverse kinematics are used to calculate the angles required to reach a target and PI controllers are used to control the angle of each joint. A fuzzy controller is then used to remove the steady state error caused by unintended flexion and rotation in the joints. Using the PI controllers and fuzzy based position controller, the arm has been tested to precisely navigate to and track stationary and moving objects within a close proximity.

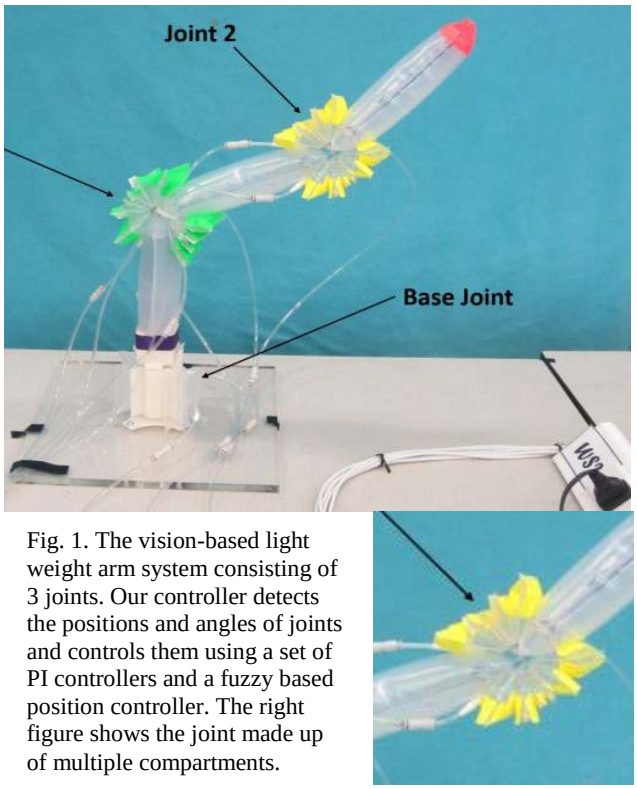

Fig. 1. The vision-based light weight arm system consisting of 3 joints. Our controller detects the positions and angles of joints and controls them using a set of PI controllers and a fuzzy based position controller. The right figure shows the joint made up of multiple compartments.

1 Introduction

The proportion of older people is increasing in many parts of the world [United Nations, 2002], and this increasingly aging population is resulting in a larger number of elderly people who require assistance completing everyday household tasks. Difficulty in grasping and lifting objects is a common physical disability in this population. In the US, 23.8 percent of the population over the age of 65 suffer from such disabilities [U.S. Census Bureau, 2013].

Assistive robotic technologies have been investigated as a solution to these problems. Some studies show that robots are acceptable to older adults [Kuo et al., 2009], robots can be deployed in a real environment [Stafford et al., 2010], and robots can be helpful to humans [Ahn et al., 2014]. It is expected that 12,400 elderly assistance robots will be sold in the period 2014-2017 [International Federation of Robotics, 2014].

However the current state of manipulator technology has several limitations, which inhibit use in personal living environments, such as cost and safety. This inspired research into the construction of low-cost robot arms that promote safe human-robot interaction. One potential solution is to use soft, flexible and lightweight inexpensive materials, along with accurate sensing and control that compensates for the inaccuracy of soft flexible materials. As well as using low cost materials, the costs of sensors and actuators should be minimized. But, it is difficult to control and sense the movements of the arm accurately due to the flexible and lightweight nature such a novel arm design. These arms are also unable to

support large or heavy mechanical sensors nor typical encoders for electrical motors.

In this paper, we propose vision-based sensing and control of the arm movement. Also we investigate the validity and effectiveness of the vision-based controller. We used a lightweight low-cost assistive arm, shown in Fig. 1, developed by Ritsumeikan University in Japan, which can reach and grasp small objects, thus allowing the consumer to regain some dexterity at an affordable budget.

This paper is organized as follows. In Section 2, we report previous visual servoing methods. In Section 3, we describe a design based on the requirements of the system. We present the development of our system in Section 4. The experimental results are provided in Section 5. Finally, we conclude this paper in Section 6.

2 Related Work

2.1 Camera Setup

Camera configurations in visual servoing can be classified in two categories; eye-in-hand and eye-to-hand. For the eye-in-hand method, the camera is attached to the manipulator, allowing calculation of arm position error by comparing with the sensed environment, and the arm acts as a reference. Such setups were used in visual servoing applications [Ma et al., 2013; Attolico et al., 2005]. The alternative setup is the eye-to-hand method, which comprises an externally fixed camera that observes the manipulator from a distance.

The eye-to-hand method often uses a parallel stereo-camera setup, where two cameras are mounted together and the depth of an object is calculated via triangulation, much like human vision. In [Huang et al., 2011], a parallel two camera setup is used to calculate the 3D positions of a manipulator's end-effector and target point.

On the other hand, an orthogonal setup can be used where the cameras are positioned with a 90-degree rotation between their axes. It is often used in velocimetry experiments [Bethea et al., 1997; McDowell et al., 2004], where 3D positions of fluids and particles are being measured, hence requiring a high level of accuracy. The orthogonal setup can provide better accuracy than a parallel setup as the depth information of one camera is directly related to the horizontal axis of the second camera. However due to the nature of the required setup, it is more limited in terms of applicability; the separated camera positions may allow more occlusion, for example.

2.2 Vision Processing Methods

Feature extraction and object detection are some of the major on-going topics of research in computer vision, which involves distinguishing and locating objects of interest from the background. For detecting features in an image, we usually use well-known detection methods, such as corner detection, edge detection, and blob detection.

Corner and edge detection have been a long-standing problem in object detection. The most popular approach to corner detection, the Harris corner detector [Harris et al., 1988], dates back over 25 years. Corner detection is also used in camera calibration techniques, to detect points of interest, such as the corners on a chessboard, to speed up the calibration process.

Edge detection may be used as a part of other detection algorithms, as edges are insufficient to act as unique features by themselves. It has been widely developed, the most common algorithm being the Canny edge detector [Canny, 1986]. Often, edge detection is used to find contours in an image, to aid other detection algorithms.

Blob detection is also a popular method, for its simplicity and efficiency. A blob is a group of connected pixels that share a certain property within an image, usually pixel intensity or colour. These pixels can be labelled together as a region using connected-component labelling algorithms such as grassfire or wave propagation methods. Different colour spaces can be used for colour blob detection. The HSV colour space is often preferred over RGB for colour detection due to its ability to separate the tint of colour from the saturation or intensity of a pixel, allowing the detection algorithm to be more robust to varying lighting conditions.

2.3 Visual Servoing

Visual servoing or vision-based robot control is a popular research topic, and often involves the navigation and tracking of a stationary or moving object with the end-effector [Hutchinson et al, 1996]. It is particularly relevant and useful for lightweight and/or flexible manipulators due to the bending displacements in the links, which make it hard to calculate the position of the end-effector with just encoders. In [Jiang et al., 2007] a vision feedback based end-effector motion controller is presented for a flexible robot arm, where a CCD camera is used to calculate the trajectory of an end-effector using visual feedback and inverse kinematics.

In [Moreno-Armendariz et al., 2005], a stereo camera setup for visual servoing is proposed with the implementation of a fuzzy controller for controlling the joint angles with only visual information. The controller generates angular change commands using an inverse kinematics model of the robot, based on the joint angle error detected from visual feedback. Similarly in [Huang et al., 2011], two CCD cameras are used to compensate for the error in the robot's end-effector, using a fuzzy position error compensator based on a set of fuzzy rules. Together with the inverse kinematic model of the arm, the target is eventually reached over several compensation iterations.

3 System Design

3.1 Requirements

The initial hypothesis was that once the vision-based control system could detect the individual joints and end-effector, it would then be possible to find the current joint angles, thus allowing it to control the lightweight robot arm without the need for any encoders. By tracking each joint we expect to understand the arm behavior and evaluate the ability to achieve a fast and accurate response. The next stage will be to to control by measuring only the end-effector. A set of requirements was chosen so that if the arm had a gripper, they would allow the system to control the arm to perform a simple task for a human user, such as picking up a small object. Hence the requirements of the system were as follows:

- Detect the 3D position of an object within 1 cm accuracy with the vision system,

- Navigate the tip of the arm to a stationary target within 2cm accuracy,
- Track a target moving at 5cm/s and maintain a distance of 10cm between the tip and target,
- Detection should be robust enough to work under a typical domestic environment.

3.2 Overall System

The vision-based controller comprises many different components, which all have to be selected and/or designed as a part of this system. Each individual component is chosen after a careful analysis of the available design options, to set up accurate inputs for the control system. Fig. 2 shows the overall system architecture with all the components. The lightweight arm requires an air pressure supply in order to inflate its links and actuators. The Raspberry Pi sets the actuator pressures over the interfacing PCB, which then controls the pressure regulators by outputting a voltage to each of its digital to analogue converters. The visual feedback for the lightweight arm is supplied by two webcams connected over USB to a PC where all the processing is done. The PC sends actuator values to the Raspberry Pi via Wi-Fi.

3.3 Arm System

3.3.1 Lightweignt Arm

The lightweight arm, developed by Ritsumeikan University, processes information for 3 degrees of freedom including a 3D printed rotating base. The arm is constructed of many polyethylene and polypropylene sheets welded together to form air compartments. These compartments form the basis for the links and actuators of the arm. Tubes are inserted to the links and actuators to provide an airway to allow pressurization of the compartments.

Each joint consist of two opposing actuators that form a pair. The actuators are made of many small plastic bags linked together that expand when air pressure is applied. This expansion causes the joint to rotate in one direction. Then the rotation of each joint can be controlled by the balance of pressure between the actuator pair of a joint. The joints are able to rotate approximately 1.8 radians in either direction while the base can rotate approximately 1.5 radians in either direction.

For all tests the links have been run at a pressure of 30kPa. The actuator's pressures range from 0 to 40kPa depending on the desired rotation of the joint. The pressure for each joint is given by

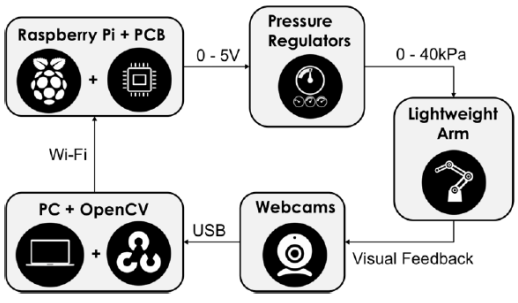

Fig. 2. Overall system architecture of the vision-based lightweight arm system.

$$P_{dU} = P_{sU} - \frac{1}{2}P_r, \quad (1)$$

$$P_{dL} = P_{sL} + \frac{1}{2}P_r, \quad (2)$$

where P_{dU} is desired pressure of the upper actuator, P_{dL} is desired pressure of the lower actuator, P_{sU} is set pressure of the upper actuator, P_{sL} is set pressure of the lower actuator, and P_r is differential pressure of the upper and lower actuator.

3.3.2 Pneumatic System

The pneumatic system utilises SMC Corporation Compact Electro regulators to control the pressure. They are chosen for their accuracy as they feature an internal control loop. The regulators are connected to an air compressor, which supplies the required air pressure based on the voltage received, between 0 and 5V. There are 7 regulators required in total, as 2 regulators are required per each joint (one per actuator), and 1 regulator is required to inflate the links of the arm, which all share a common pressure.

3.4 Vision System

3.4.1 Camera Setup

The visual feedback is provided by 2 Logitech webcams via USB, both with a resolution of 640 x 480 pixels. The cameras are positioned in an orthogonal eye-to-hand setup as shown in Fig. 3, as it can provide more accuracy than a typical parallel setup used in most computer vision applications. The depth of one camera's view is related to the other camera's horizontal axis, and they share a common vertical axis. Accurate calibration is required in order to ensure that the images from the camera feeds are perfectly orthogonal.

3.4.2 Joint Detection

Of the many detection algorithms, we use colour blob detection due to its simplicity and efficiency. Also, to meet the last requirement, we use HSV colour space to ensure more robustness in the detection process, under varying lighting conditions. To detect the joints with blob

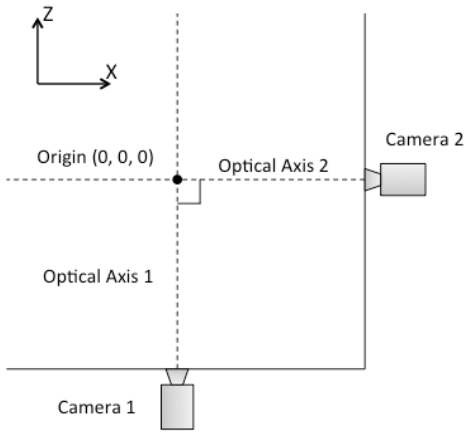

Fig. 3. Orthogonal eye-to-hand camera setup of the vision-based lightweight arm system.

detection, distinct coloured markers were placed at each joint. However, due to the structure of the arm, it was difficult to place a marker at the centre of each joint and maximise visibility from both cameras. The main problem was the actuators' structure, which changed in size and shape due to inflation and deflation.

Initially it was decided that it would be less difficult to place the coloured bands on the links of the arm and detect the centre of the bands to calculate the link vector. The joint positions and angles could then be calculated by finding the angle between the link vectors. However the bands were often obstructed from the cameras' views and the results were inaccurate and unstable. The final solution was to cover the actuator bags with the coloured markers and calculate the centre point of the cluster of detected blobs as shown in Fig 4. This method provided more stable and reliable.

3.5 Control System

3.5.1 Hardware Design

To control the pressure regulators, multiple digital to analogue converters are needed. A custom PCB with 8 DAC channels was designed using Altium to interface between a Raspberry Pi and the pressure regulators. The DACs are controlled from the Raspberry Pi using the SPI bus. The PCB also features headers for easy connection to the regulators. It also contains a 5V mini USB port for powering the Raspberry Pi.

Initially it was proposed to run all image processing and control algorithms on the Raspberry Pi to keep the vision control system constrained to a small, embedded system. However it was found that this approach lacked the necessary processing power to be able to run the image processing for two 640x480 pixel video streams. Installing a different distribution of Linux was attempted to fully utilize the A7 CPU on the Raspberry Pi. This involved enabling the NEON (single instruction multiple data) and VFP4 (hardware support for floating point operations) flags in OpenCV however this was still insufficient as the average frames processed per second was only 0.6 to 0.7. TBB (Threading Building Blocks) support in OpenCV and writing multithreaded code were not examined because even with a maximum speedup of 4 (number of CPU cores) the processing time would still be too long.

As a result it was necessary to run all of the image processing and control code on an external PC and transfer the actuator values to the Raspberry Pi over Wi-Fi. Due to its simplicity, a Linux package Socat was used as a relay for data transfer between the PC and Raspberry. Data was transferred by piping the standard output from the PC program to the Raspberry Pi using UDP packets. A built-in Client Server could be setup between the Raspberry Pi and PC if this distributed system continues to be used in the future. The average latency of the network on which the arm was tested was around 3ms and was considered negligible compared to the processing time. Dropped packets and unexpected delays in the Wi-Fi connection were not considered for the purposes of this project.

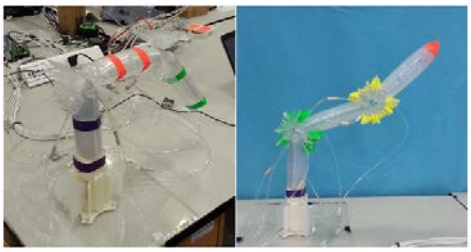

Fig. 4. Initial strip marker placement (left). Final colour marker placement (right).

3.5.2 Control Algorithms

As mentioned in the requirements, to accurately control the arm without the use of any encoders, it is necessary to be able to detect and set the joint angles accurately. To achieve this, the joints require a controller to correct their angles and converge to a target. Implementation of a PI controller was considered, to allow each joint to be set quickly and accurately. The integral term allows for the removal of steady state errors. It can also account for the deflections in the links caused by the flexible nature of the lightweight arm. The PI controller was also chosen for its simplicity in implementation and tuning.

However the PI controllers do not account for the error due to unintended rotations and twists in the joints and links, that can arise from the non-rigid structure of the arm, as rotations are not strictly fixed to the intended axis of rotation. While the PI controllers allow for rapid and easy convergence to the desired angles, the implementation of an additional controller at an upper-level was investigated.

After reviewing the visual servoing related literature, it was found that fuzzy logic based controllers have been used for vision control of manipulators [Huang et al., 2011]. As the joints of the lightweight robot arm were already equipped with PI controllers for accurate joint angle setting, a fuzzy position error compensator based on the algorithm in [Huang et al., 2011] was chosen as an upper layer controller, for its simplicity and the ease of integrating it into the control system. In the future direct control of the arm end point should alleviate this control requirement.

4 Vision-based Lightweight Arm System

4.1 Camera Calibration

To calculate the 3D positions of the joints the intrinsic parameters of each camera must be known and the cameras must be as close as possible to 90 degrees apart. Therefore the vision control system must be calibrated before use. Calibration is completed in three steps. First the cameras must be calibrated independently to discover their intrinsic parameters, which includes the camera matrix and distortion parameters. Next both cameras must be calibrated together to find their poses relative to each other. Finally this information is used to create a mapping of x and y pixel values to undistort and rectify rotation offsets in the images received from both cameras.

4.1.1 Calibration

OpenCV provides many useful features for calibrating a single camera. It supports calibration using multiple views of a chessboard a symmetrical circle pattern or an

asymmetrical circle pattern. Once the camera has been calibrated its co-ordinates can be remapped to reflect the real world position of points on the imaging plane. However the scale of a position is unknown as only one perspective is presented therefore 2 cameras must be used to determine the position of an object in 3D space. To retrieve 3 dimensional information from two cameras, they must be calibrated relative to each other, therefore the pose between cameras must be found. Once the rotation matrix and intrinsic parameters of both cameras have been found, we create a mapping of *x* and *y* pixel locations for each camera to remove any distortion in the cameras and rotation offsets between them.

4.1.2 3D Reconstruction

After the system has been calibrated, any 3D point in the view of both cameras can be calculated as shown in Fig. 5. Since the camera matrixes and the distance between the cameras are known, the following equations for the *X* and *Y* position relative to the origin point can be found.

$$X = \left(\frac{x_1 - c_{x1}}{f_{x1}}\right) \times \left(Y + D_y\right), \tag{3}$$

$$Y = \left(\frac{x_2 - c_{x2}}{f_{x2}}\right) \times \left(-X + D_x\right), \tag{4}$$

$$Z = \frac{1}{2}\left[\left(\frac{y_1 - c_{y1}}{f_{y1}}\right) \times D_y + \left(\frac{y_2 - c_{y2}}{f_{y2}}\right) \times D_x - D_y\right], \tag{5}$$

where D_x is distance in the x axis of camera 1 to camera 2, D_y is distance in the y axis of camera 1 to camera 2, D_z is distance in the z axis of camera 1 to camera 2, x_i is horizontal pixel location in camera i, y_i is vertical pixel location in camera i, c_{xi} is displacement of the center coordinates in the x axis for camera i, c_{yi} is displacement of center coordinates in the y axis for camera i, f_{xi} is the horizontal focal length in camera i, and f_{yi} is the vertical focal length in camera i.

Therefore using simultaneous equations *X* and *Y* can be solved as

$$X = \frac{b \times D_x + D_y}{\frac{1}{a} + b}, \tag{6}$$

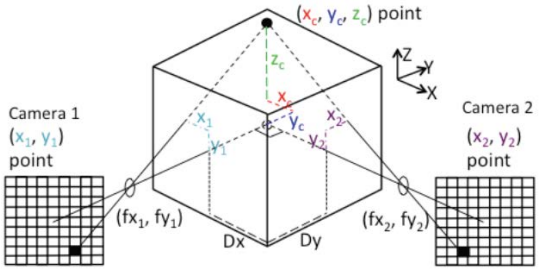

Fig. 5. Reconstruction of 3D point from two 2D points.

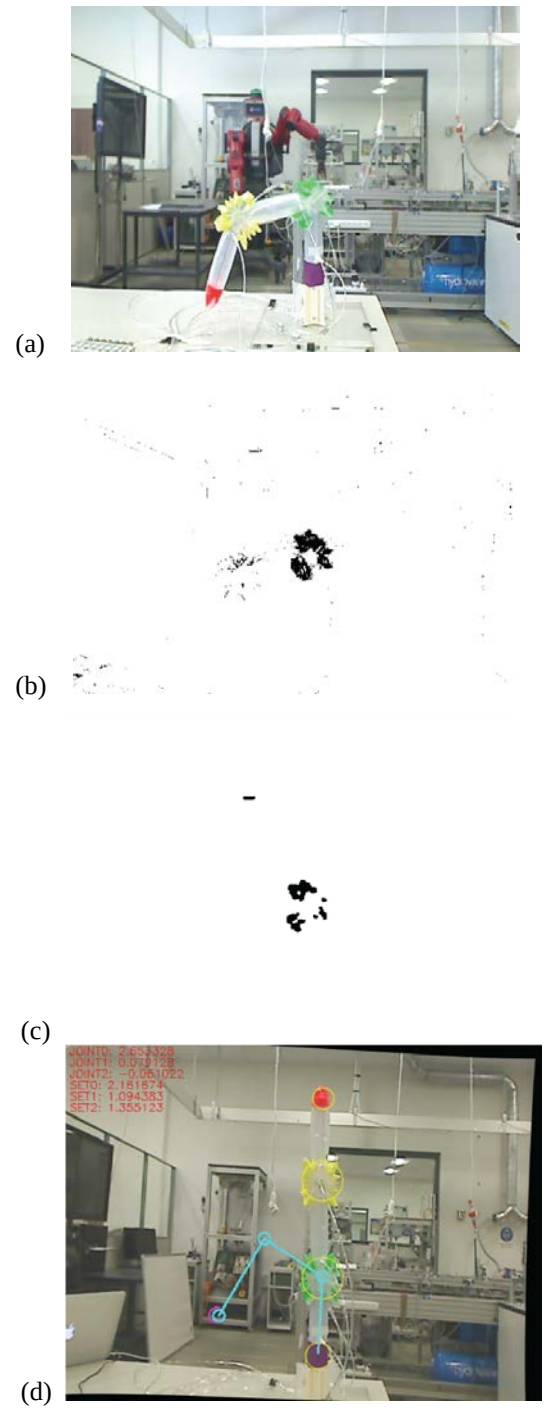

Fig. 6. Vision processing steps; (a) original image, (b) Joint detection using binary threshold filter, (c) Joint detection using erosion and dilation, (d) Visualisation of inverse kinematics on camera image.

$$Y = \frac{D_x - a \times D_y}{a + \frac{1}{b}}, \tag{7}$$

where

$$a = \frac{x_1 + c_{x1}}{f_{x1}}, \tag{8}$$

$$b = \frac{x_2 + c_{x2}}{f_{x2}}. \tag{9}$$

Once the positions of the joints, tip and base of the arm are found an angle can be found between the two vectors using the following equation.

$$\theta = \cos^{-1}\left(\frac{u \cdot v}{\|u\| \times \|v\|}\right), \tag{10}$$

where u and v are link vectors.

However this only provides the magnitude of rotation. Hence the cross product of both vectors is calculated. The resulting x value can be used to determine the direction of rotation for the joint since the base joint is fixed and can rotate a maximum of 1.5 radians in either direction. For the Base Joint only the arc tangent of the position of Joint 1 less the base position is found, to calculate the base rotation.

4.2 Target Position Calculation

4.2.1 Joint Detection

Prior to running the detection, the HSV ranges for each joint are gathered and stored in an XML file. The function *SetBlobColour* is run to manually select and configure the each joint's HSV ranges and save them to the file. The detection algorithm will then be run for each set of HSV range saved in the XML file for images from both cameras. An iteration of the detection algorithm involves the following steps. The image is retrieved from the camera and a binary threshold is applied over the image between the HSV ranges for the particular joint as shown in Fig. 6(b).

However the image can contain noise or unintended detection of background objects with a similar hue. Hence, a series of morphological operations are performed over the image as shown in Fig. 6(c). Erosion fills the gaps of the dark areas to strengthen the detection, while dilation eliminates small pixels to reduce noise (the binary image has been inverted).

Next, using a modified version of the OpenCV's *SimpleBlobDetector* class, blobs are formed and detected by connected component labelling and filtered based on the blob area. The modified class extends *SimpleBlobDetector*, whilst allowing the private variables and characteristics of each blob to be accessible, such as blob inertia, circularity and area, to aid with tuning and debugging the blob detection process.

As each joint consists of multiple actuator bags on either side of the arm, they will most likely be detected as a cluster of blobs in a condensed region. Therefore, each detected blob is measured in terms of distance to other nearby blobs and grouped as a single cluster. The centre point of these blobs is then calculated by weighing the position of each blob based on its size.

The calculated joint centre is fed into a moving average filter, in order to stabilise the joint position over a few frames and reduce the effect of spikes from false detections.

Lastly to optimise performance, the detection is only performed over a small region of interest where the joint was last detected. The detection is performed over the whole image only if it fails to detect in the region.

4.2.2 Joint Angle Calculation

Once the base joint, two planar link joints and the end-effector have been detected, the joint angles between them can be calculated simply by finding the vectors and calculating the angle between them, using 3D geometry. However a method to realise the direction of the angle is required as the above method can only calculate the magnitude of the angle between the vectors. Hence, the cross product of the vectors is calculated and compared to the base angle vector.

4.2.3 Inverse Kinematics

For a given target within the reachable workspace of the arm, it is possible to find the required joint angles that will navigate the arm to the target, by using inverse kinematics. The inverse kinematic model for a typical 2 link planar manipulator on a rotating base can be easily calculated as shown in Fig. 7.

For an arbitrary target location $P_t(x_t, y_t, z_t)$, the base joint angle θ_0, joint angles θ_1 and θ_2 can be calculated with the following equations.

$$\theta_0 = \begin{cases} 0 & where\ x_t = 0 \\ \tan^{-1}\left(\frac{y_t}{x_t}\right) & where\ x_t \neq 0 \end{cases}, \tag{11}$$

$$\theta_1 = \frac{\pi}{2} - \tan^{-1}\left(\frac{z_t - l_0}{\sqrt{x_t^2 + y_t^2}}\right) + \tan^{-1}\left(\frac{l_2 \sin(-\theta_2)}{l_t + l_2 \cos\theta_2}\right), \tag{12}$$

$$\theta_2 = -\tan^{-1}\left(\frac{-\sqrt{1 - D^2}}{D}\right), \tag{13}$$

where

$$D = \frac{x_t^2 + y_t^2 + \left(z_t - l_0\right)^2 - l_1^2 + l_2^2}{2 l_1 l_2}, \tag{14}$$

link length l_0 is a distance between base of arm to joint 1, l_1 is a distance between joint 1 and 2, and l_2 is a distance between joint 2 and end-effector. Each of the link lengths are calculated based on the observed position of the joints. Using calculated desired joint angles (θ_0, θ_1, θ_2) and the measured link lengths (l_0, l_1, l_2), the expected pose of the arm is drawn to the screen to visualise how the target will be reached by the arm, as shown in Fig. 6(d).

4.3 Arm Controlling

4.3.1 PI Controller

Each joint is equipped with its own PI controller to allow the joint angle to converge to the desired angle calculated by inverse kinematics. The integral term allows the angle to converge with no steady state error. The PI controller takes the error between the current measured joint angle from detection and desired joint angle calculated by inverse kinematics as input, and outputs a double, representing the pressure value for the actuators between the ranges -164 and 164. The controllers were tuned with Kp value of 400 and Ki value of 25.

4.3.2 Fuzzy Based Position Controller

The fuzzy based position controller, based on the fuzzy position error compensator in [Huang et al., 2011] was implemented. For an arbitrary target position P_t (x_t, y_t, z_t) and end-effector position P_c (x_c, y_c, z_c), the error e is equal to

$$\begin{aligned} e &= P_t - P_c \\ &= \left(x_t - x_c,\ y_t - y_c,\ z_t - z_c\right), \\ &= \left(e_x,\ e_y,\ e_z\right) \end{aligned} \tag{15}$$

where P_t is the target position in x,y,z coordinates, P_c is the end effector position in x, y, z coordinates, e_x is error in the x direction, e_y is error in the y direction, and e_z is error in the z direction.

Ideally if the model of the arm was perfect and there were no deflections or twists in the links, e would equal to zero and $P_t = P_c$. However most of the time this would not be the case, hence for an error, e_k in the k-axis where $k = (x, y, z)$, the compensation step Δk is calculated based on the set of fuzzy rules as shown in the equation (16). The fuzzy position controller is a fuzzy rule base (FRB) where the antecedent is e_k and the consequent is the compensation step Δk, which adjusts the target position P_t. The fuzzy rule base is as follows.

$$FRB: \begin{cases} Rule\ 1:\ If\ e_k\ is\ HN\ then\ \Delta k\ is\ HN \\ Rule\ 2:\ If\ e_k\ is\ LN\ then\ \Delta k\ is\ LN \\ Rule\ 3:\ If\ e_k\ is\ ZO\ then\ \Delta k\ is\ ZO \\ Rule\ 4:\ If\ e_k\ is\ LP\ then\ \Delta k\ is\ LP \\ Rule\ 5:\ If\ e_k\ is\ HP\ then\ \Delta k\ is\ HP \end{cases}, \tag{16}$$

where *HN* is High Negative, *LN* is Low Negative, *ZO* is Zero, *LP* is Low Positive, and *HP* is High Positive. Fig. 8 shows the membership functions of e_k and Δk, used for the fuzzification process. The defuzzification of the compensation step is implemented as the weighted average method [Li et al., 2004].,

$$\Delta k = \sum_{r=1}^{5} \mu^r\left(e_k\right) \cdot e_k \cdot u_k^r, \quad (k = x, y, z), \tag{17}$$

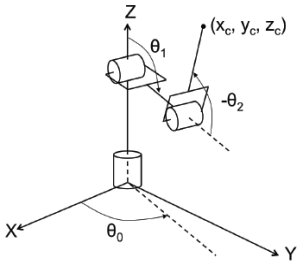

Fig. 7. Inverse kinematic model of lightweight arm.

where $\mu^r(e_k)$ is the weight of the membership function for the rth rule and u_k^r is the support of the rth consequent fuzzy set.

After adjusting the target by the compensation step, the joint angles are recalculated using inverse kinematics. Once the PI controllers have brought the joint angles to a steady state, the error is evaluated again. If the absolute error is within the threshold of 10mm, no changes are made, otherwise the compensation step is calculated again and the process repeats until the absolute error is below the threshold.

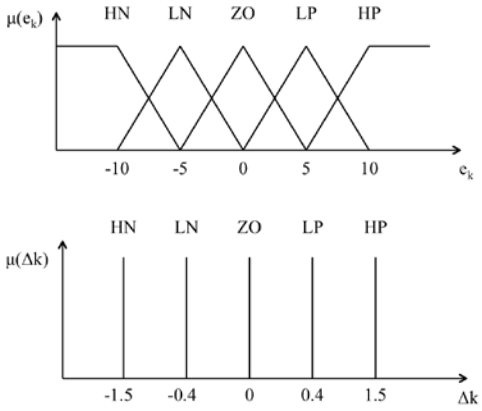

Fig. 8. Membership functions of error and compensation step for k = x, y, z.

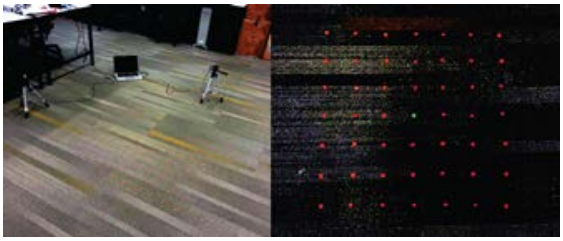

Fig. 9. Reprojection test setup (left). Reprojected grid of X and Y ranging from -300 to 300mm (right).

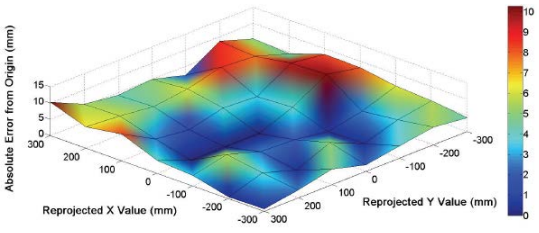

Fig. 10. Surface plot of absolute error from reprojection test.

5 Experimental Results

5.1 3D Space Reprojection Test

5.1.1 Experimental Setup

After calibration, the 3D space was reprojected to the real world in order to measure its accuracy. This was done by first reprojecting the origin and then the individual points with 100mm increments in *X*, *Y* and *Z*, from the camera's image feeds. The reprojected points ranged in the *X* and *Y* axes from -300mm to 300mm, and from -100mm to 100mm in the *Z*-axis. After reprojection, the points were marked on the ground by markers and their respective heights were recorded. The distance of each point from the origin was marked as green in Fig. 9, and compared with the expected distance from the origin.

5.1.2 Results

The absolute error between the measured distance and the expected distance from the origin was recorded. The error in the *Z*-axis only varied by 2mm or less throughout the different points. Hence error in the *Z*-axis was considered to be a result of measurement error. The reprojected points of varying *X* and *Y* values, and fixed *Z* value at 0, was tested and drawn as a surface plot as shown in Fig. 10. Overall the error appeared to be small and random with no distinct pattern, with an average error of 4.38mm and a maximum error of 10.26mm. However, the measurement method added a lot of inaccuracy and did not account for errors in different dimensions, hence a better test should be adopted in the future.

5.2 Joint Angle Calculation Test

5.2.1 Experimental Setup

For the purpose of testing the accuracy of the detection method and ensuring an accurate input was fed into the PI controller, the joint angle was measured on a mechanical KUKA YouBot manipulator. The YouBot has encoders built into its joints, thus the encoder's reading and the vision system's measurement could be compared. The YouBot was fixed at 45 degrees from both cameras for easy detection, and the test was conducted at the 4 quadrants of the 3D space as shown in Fig. 11.

5.2.2 Results

The results of the test were plotted as shown in Fig. 12, as a graph of error between measured and expected angle against expected angle ranging from 0 to 2.5 radians. The results showed that the average angle error was only 0.0126 radians (0.722 deg) and the maximum error was 0.04 radians (2.292 deg). Both the average and maximum errors were very low overall, thus the measured angles were considered to be sufficiently accurate to be used as input for the PI controllers.

5.3 PI Controller Test

5.3.1 Experimental Setup

The PI controller for each joint on the infaltable robot arm was tested against a step and ramp input of half pi radians. Only the joint under testing was moved for each test and no load was applied to the arm. The response was recorded against time.

5.3.2 Step Input Result

Fig. 13 shows the response of each joint when given a step input of half pi radians. The joint angles converged to the desired angles with no steady state error and settling times (within 5% of final value) of 7.34, 10.46 and 9.45 seconds for the base joint, joint 1 and joint 2 respectively.

Joint 1 showed more oscillations and a less smooth convergence trajectory than joint 2. This is most likely due to joint 1 having more load than joint 2, and the upper link could sway due to the arm's lightweight and non-rigid nature.

5.3.3 Ramp Input Result

Fig. 14 shows the response of each joint when given a ramp input of 6 deg/s over 15 seconds. The joint angle converged to the desired angle quickly and accurately with no steady state error. However joint 1, like in the step input results, showed more oscillations than the other joints when given a ramp input, for similar reasons.

5.4 Vision-based Control System Test

5.4.1 Experimental Setup

The overall vision control system was tested with and without the fuzzy position controller enabled against a step input, to show the effectiveness of the fuzzy position controller and test if the system has met the requirement for tracking a stationary target. The step input consisted of a change in virtual target position from (200mm, -200mm, -200mm) to (-200mm, -200mm, -200mm), a shift of 400mm in the *X*-axis.

Finally to test if the system has met the requirement for tracking a moving object, it was tested against a ramp

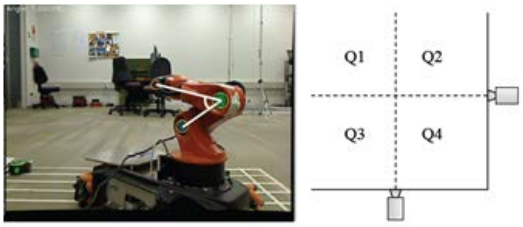

Fig. 11. Joint angle calculation on the KUKA YouBot (left). 3D space quadrants (right).

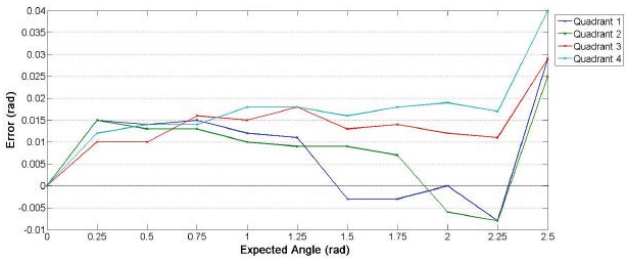

Fig. 12. Joint angle calculation test results.

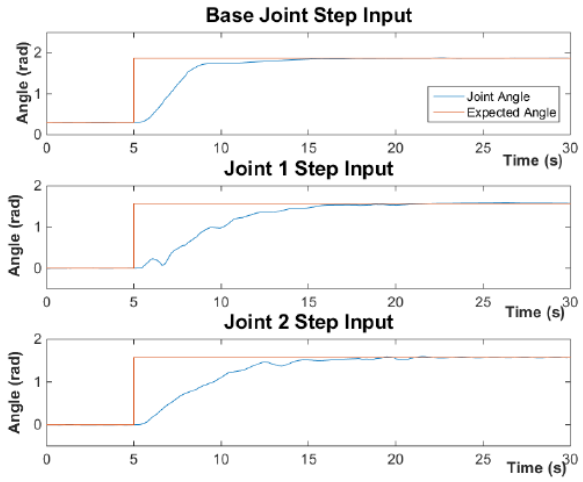

Fig. 13. PI controller step input test results.

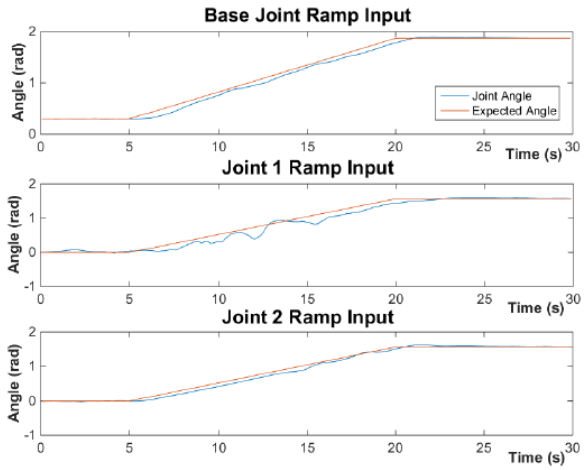

Fig. 14. PI controller ramp input test results.

input, consisting of a virtual target changing in position over 8 seconds in the X-axis direction at 50 mm/s, from (-200mm, -200mm, -200mm) to (200mm, -200mm, -200mm). The absolute distance the end-effector maintained from the target was recorded against time.

5.4.2 Step Input Result

The results of the step input test were as shown in Fig. 15. The vision control system without the fuzzy position controller displayed an absolute steady state error of approximately 31mm. However, when equipped with a fuzzy position controller, the end-effector managed to navigate and converge to the target without any steady state error, but with minor oscillations within 7mm. This is due to the fact that without the fuzzy controller, the system cannot compensate for the errors caused by the unintended rotations and twists in the links. The vision control system overall, appeared to follow a stationary target within the accuracy specified in the requirements, with a settling time of 7.5 seconds.

5.4.3 Ramp Input Result

The results of the ramp input test was as shown in Figure 16, where the red line represents the target position and the blue line represents the end-effector position. It can be observed that the moving target was tracked while

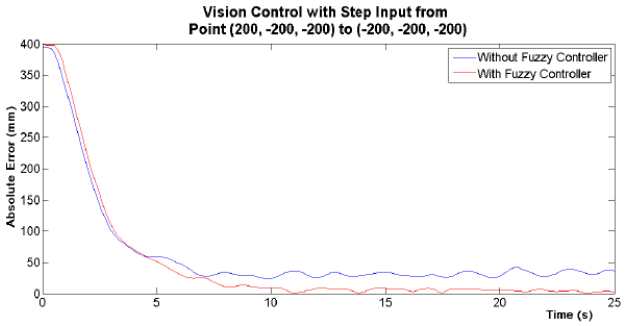

Fig. 15. Vision control system step input test results.

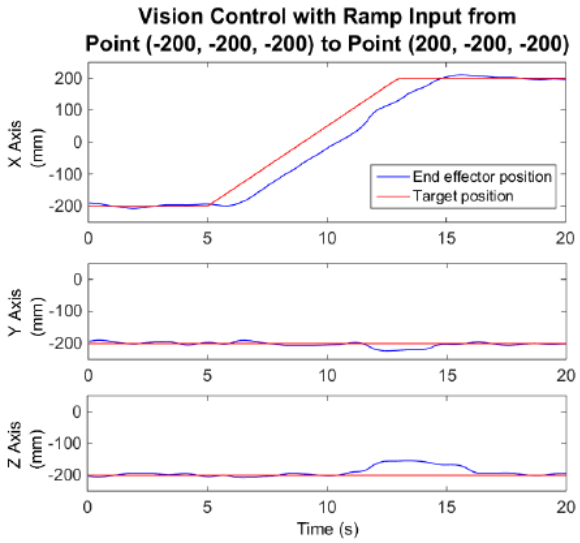

Fig. 16. Vision control system ramp input test results.

maintaining sufficient accuracy. Once the target was stopped, the error was quickly reduced and steady state error was removed. The maximum absolute distance that was recorded between the target and the end-effector throughout the test was 8.48cm, thus meeting the third requirement.

6 Conclusions

In this paper, we presented an implementation of a vision-based control system to investigate the validity and effectiveness of controlling a lightweight robot arm. A 3D space was calibrated and tested, to provide accurate inputs to the vision control system. The error from the results of the 3D reprojection test however slightly exceeded the specified requirement, most likely due to an inaccurate measurement method. We designed a hybrid controller using PI joint controllers and a fuzzy controller to tune the errors to compensate for link deflections. The proposed control methods have proven to be able to control the lightweight robot arm accurately. Also the proposed control system has met the requirements in tracking a stationary and moving target within the specified accuracy. Overall, the proposed system has been demonstrated to be a viable method for controlling the lightweight robot arm.

For future work, the addition of a gripper should be investigated in order to test the vision-based control

system in its ability to perform simple tasks related to the original motivation of the project, such as picking up small objects. In terms of testing, a more accurate method of measuring the 3D calibrated space should be investigated, as a fair portion of the error from the 3D space reprojection results appeared to be measurement error. Control methods could be further refined and tuned, to reduce settling time. The addition of the derivative term could be investigated for the PI controllers. The fuzzy based controller also requires more tuning. Lastly the performance of the overall system could be optimised so that it can run on a lower-end processor as a self-contained embedded system.

References

[Ahn et al., 2014] Ho Seok Ahn, I-Han Kuo, Chandan Datta, Rebecca Stafford, Ngaire Kerse, Kathy Peri, Elizabeth Broadbent, and Bruce MacDonald. Design of a Kiosk Type Healthcare Robot System for Older People in Private and Public Places. *Proceedings of the 2014 International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, page 578-589, 2014.

[Attolico et al., 2005] M. Attolico, O. Cargnel, R. Cazzoli, A. Davighi, and F. Bernelli-Zazzera. A visual servoing control system for lightweight robotic manipulator. *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, page 1005-1010, 2005.

[Bethea et al., 1997] M. D. Bethea, J. A. Lock, F. Merat, P. Crouser. Three-dimensional camera calibration Technique for stereo imaging velocimetry experiments. *Optical Engineering*, 32(12): 3445-3454, 2014.

[Canny, 1986] John Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679-698, 1986.

[Hutchinson et al, 1996] S. Hutchinson, G. Hager, and P. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(6), pp. 651-670. October 1996.

[Harris et al., 1988] C. Harris and M. Stephens. A combined corner and edge detector. *Proceedings of the 4th Alvey Vision Conference*, page 147-151, 1988.

[Huang et al., 2011] Cheng-Hao Huang, Chi-Sheng Hsu, Po-Chien Tsai, Rong-Jyue Wang, and Wen-June Wang. Vision based 3-D position control for a robot arm. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, page 1699-1703, 2011.

[International Federation of Robotic, 2014] International Federation of Robotic, World Robotics Service Robotics. *http://www.ifr.org/service-robots/statistics*, 2014.

[Jiang et al., 2007] Zhao-Hui Jiang, T. Eguchi. Vision feedback based end-effector motion control of a flexible robot arm. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, page 2413-2419, 2007.

[Kim et al., 2015] H. Kim, Y. Tanaka, A. Kawamura, S. Kawamura, Y. Nishioka. Improvement of position accuracy for inflatable arm using visual feedback control method. *Proceedings of the IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, page 767-772, 2015.

[Kuo et al., 2009] I. H. Kuo, J. M. Rabindran, E. Broadbent, Y. I. Lee, N. Kerse, R. M. Q. Stafford and Bruce. A. MacDonald. Age and gender factors in user acceptance of healthcare robots. *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication*, page 214-219, 2009.

[Li et al., 2004] T. H. S. Li, S. J. Chang, and W. Tong. Fuzzy target tracking control of autonomous mobile robots by using infrared sensors. *IEEE Transactions on Fuzzy Systems*, 12(4):491-501, 2004.

[Ma et al., 2013] G. Ma, Q. Huang, Z. Yu, X. Chen, L. Meng, M. S. Sultan, W. Zhang, Y. Liu. Hand-eye servo and flexible control of an anthropomorphic arm. *Proceedings of the IEEE International on Robotics and Biomimetics (ROBIO)*, page 1432-1437, 2013.

[McDowell et al., 2004] M. McDowell, E. Gray. Stereo imaging velocimetry technique using standard off-the-shelf CCD cameras. *Tech. report, NASA Centre for AeroSpace Information*, Hanover, Maryland, USA, 2004.

[Moreno-Armendariz et al., 2005] M. A. Moreno-Armendariz and W. Yu. A new fuzzy visual servoing with application to robot manipulator. *Proceedings of the American Control Conference*, page 3688-3693, 2005.

[Stafford et al., 2010] R. Q. Stafford, E. Broadbent, C. Jayawardena, U. Unger, I. H. Kuo, A. Igic, R. Wong, N. Kerse, C. Watson, and B. A. MacDonald. Improved robot attitudes and emotions at a retirement home after meeting a robot. *Proceedings of the 2010 IEEE International Symposium on Robot and Human Interactive Communication*, page 82-87, 2010.

[United Nations, 2002] United Nations, "World population aging," *United Nations Publications*, New York, 2002.

[U.S. Census Bureau, 2013] U.S. Census Bureau. Disabled World. *http://www.disabled-world.com/disability/statis tics/info.php*. 2013.