A Lens-Calibrated Active Marker Metrology System

Richard E. Gunstone and Mark. H. Lee
Department of Computer Science
University of Wales,
Aberystwyth,
Ceredigion,
Wales, UK SY23 3DB
{rcg,mhl}@aber.ac.uk

Abstract

This paper presents a prototypical marker tracking system, MT, which is capable of recording multiple mobile robot trajectories in parallel for offline analysis. The system is also capable of providing trajectory data in realtime to agents (such as robots in an arena) and implements several multi-agent operators to simplify agent-based perception. The latter characteristic provides an ability to minimise the normally expensive process of implementing agent-centric perceptual mechanisms and provides a means for multi-agent *global knowledge* (Parker 1993).

1. Introduction

A frequent hurdle in mobile robotics is the challenge of analysing the behaviour of robots in a systematic and accurate way. Marker tracking systems fulfills this need by recording the trajectories of markers in a scene over time, providing positional information from which the path and position of a collection of markers can be derived. By arranging markers in suitable patterns for recognition, the position and optionally the orientation of a robot can be ascertained.

A *marker tracking* or *metrology system* is a mechanism that records the movement of markers in a scene. Markers can be attached to objects of interest, which in this case are mobile robots. Over time a body of data can be generated that shows the path the marker takes through movement. With additional processing and pre-defined arrangements of markers, high-level analysis is possible of one or more robots in an arena.

As well as providing offline datasets for later analysis, *realtime marker tracking systems* can also provide on-line feedback on marker positions during experimentation. This is advantageous when implementing complex experiments where many factors are secondary to the principal investigation; for example when a learning system is being developed or when perceptual mechanisms are costly or scarce, significant development time can be expended in developing agent-based perceptual mechanisms that are not critically important. Further, the prospect of introducing virtual elements into visual space is a distinct possibility, and perhaps a desirable one: a handful of hand-built objects (such as food pucks in foraging experiments) can be complemented by a dearth of objects simulated in visual space to further assess effectiveness. Similarly, the definition of particular areas or regions can be performed with minimal adjustment of the environment. Realtime tracking systems can also provide an effective alternative to other more costly methods of implementing complex operators for multi-agent collective robotics, such as radio receivers or agent-based camera units.

The metrology system (MT) presented in this paper was developed to fulfill both realtime and offline data capture needs while investigating a learning architecture, exploiting a priori marker arrangements attached to robots to derive position and direction. The approach taken can also track the position of objects such as pucks (also possessing markers) in realtime. The most recent version of the system uses standard PC equipment, which provides significant speed improvements over earlier versions that were developed using a Sun Ultra-10 SPARC with a Sun capture board and XIL library for image processing. These earlier versions also investigated optic flow as a means for tracking, however this was dropped in favour of predefined marker arrays in the image attached to agents and objects of interest, on the grounds of efficiency.

2. Capture Equipment

MT is a visually-based marker tracking system that uses a video camera and capture board to observe markers in a scene. A digital JVC camera was used in the tracking system, with various enhancements, including automatic digital image processing, color balancing and a number of other image-enhancing functions.

The camera was mounted on a fixed bar attached to a ceiling in the laboratory, and outfitted with a 92-degree wide angle lens. Facing downwards, this particular lens was chosen for its ability to capture the necessary area on the laboratory floor in which the mobile robots operated, thus allowing markers to be observed throughout a trial of the tracking system. An unfortunate side-effect however of using such a lens

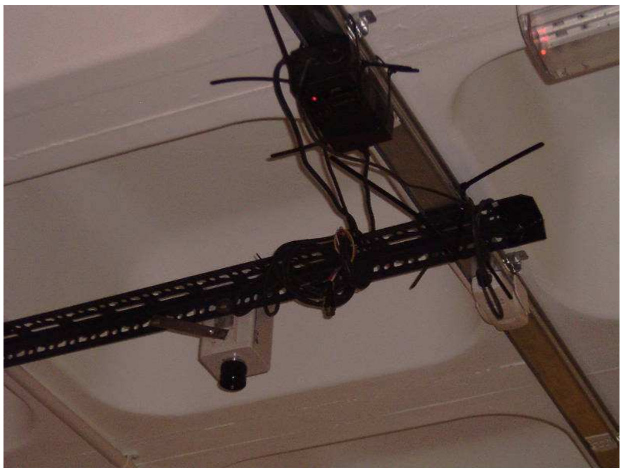
Figure 1: The tracking camera orthogonally mounted on the ceiling.

is the introduction of a significant barrel distortion in the captured image, which is examined in section 8.

A key issue in performing effective extraction of the markers was found to be the removal of irrelevant information in the scene, and it was found a combination of camera-based modifications and software extraction stages were necessary. Since the markers used in the eventual system were active high-intensity bulbs, much of the irrelevant information was less brighter than the markers themselves. A minimal aperture[1] setting on the camera lens was used to remove a significant amount of irrelevant information in the image, yielding a set of isolated high-intensity markers. Later software processing on the captured image was used to isolate the markers from the captured images more thoroughly.

The image was carried over a composite video signal lead to an analogue PCI Hauppauge Win TV capture board (based on the BTTV chipset and of the kind used for television viewing on a personal computer) resident in a Linux-based PC. These hardware and software components are widely-available and cost-effective.

3. Types of Markers Considered

In order for a marker extraction algorithm to be effective, it must exploit a pixel type and structure in the image as the marker. Several marker types were investigated during development. Broadly two main types of marker (or "fiducials") are possible: *reflective* or *emissive*. Reflective (passive) markers reflect ambient light in the environment, and ideally are diffuse reflectors with a low specular reflectance. Since they reflect rather than emit light, such markers are dependent on a sufficient level of ambient light to allows the marker extraction algorithm to work effectively on captured camera images. Reflective markers can also possess a degree of luminescence to further help the extraction process. Emissive (active) markers radiate light into the environment. As they are emitters and not reflectors, they are not dependent on reflected light and thus ambient light in the surroundings can vary. Better results are typically obtained with reduced lighting.

Reflective markers are the easiest and most straightforward kind of marker to implement. An early version of the software made use of reflective luminescent and patterned paper markers for tracking, in normal lighting conditions. We found that the majority of the time the marker was identified correctly, however a noticeable number of iterations of the algorithm showed identification of other objects in the scene in error.

Emissive markers are more difficult to implement, and for best results tend to require reduced lighting conditions. They are typically implemented using either Light Emitting Diodes (LEDs) or bulbs. LEDs are usually of a specific colour and can emit light at extremely high intensities. An LED can also avoid saturation of the captured image, allowing for accurate colour isolation (for marker classes). A typical Super-Bright AlGaA LED with a diffuser lens can produce an intensity of 160mcd, but with a small viewing angle. A non-diffuser lens version has a much higher intensity of 530mcd but a much narrower viewing angle[2]. *Incandescent bulbs* offer the highest brightness, emit light in all directions, but can saturate camera capture systems.

Markers can also be of the same or differing colours. Markers of the same colour are quicker to identify, but need other constraints in order to extract structural information such as robot pose—for example through the use of positional arrangements. Colour markers reduce the number of markers required in the scene, but introduce further costs in image processing.

The properties of the laboratory were found to be a major factor in conjunction with the cost of physically building the markers. A low ceiling height and wide arena meant that the ceiling camera, mounted orthogonally to the floor, needed to be augmented with a 92 degree lens so as to ensure a capture of the entire mobile robot arena. This lens also introduces a ~45 degree angle with the floor at the extremities, meaning any marker must have at least such an angle (or ideally more) of visibility in all directions for effective capture. Wide angle lenses also present a reduced number of pixels for the same marker in an extreme position. This lead to the use of incandescent bulbs for the markers, although LEDs with diffuser lenses would be a further possibility.

For reasons of simplicity, identically colour markers were found to be the most effective when computing within the RGB space[3]. Although these bulbs tended to saturate the image at the centre of the marker, relying on the pixel intensities allowed this problem to be side-stepped. We found it to be the most successful of all methods considered.

[1]The aperture of a camera lens determines the amount of light entering the lens and eventually falling onto the CCD chip that generates the video image.

[2]Ultra-Bright LEDs are also quite expensive.

[3]Better results for coloured markers could be obtained through the use of the HSV (Hue, Saturation, Variance) colour space, since the hue component can be separated regardless of other factors.

To calculate the robot orientation an a priori structural pattern was introduced using three markers: two on the rear (*lateral*) and one on the tip (front or *ventral*). The lateral inter-marker distance was significantly lower than the ventral axis-tip distance, which allowed the tracking algorithm to determine the pose of the lighting array effectively by calculating the shorter lateral inter-marker distances.

4. General Tracking Process

The marker tracking top-level process is a repetitive loop in which images containing the thresholded marker pixels are repeatedly acquired and processed. The output data from these two steps is used to generate the offline trajectory records and update the resources used in realtime information processes. Robots in the environment that wish to find their own positions can do so by querying the tracking system to obtain this information. The data generated by the tracker is recorded and output to file, facilitating the construction of post-experimental metrological plots of robot activity.

The main functional stages in sequence are as follows:

Acquisition—The frame capture boards are instructed to acquire the data from the attached camera system. This involves the conversion of the raw analogue signal from the camera (in this case a PAL signal) to appropriate representations for memory, which in this case is an RGB representation. A convolution is applied during this stage. Image acquisition is performed using a Video4Linux memory map.

Marker Identification—At this point the markers in the image are identified. This is a multi-part process involving the isolation of appropriate parts of the image (via thresholding), the identification of marker clusters, and the reconstruction of disjoint pixels belonging to markers. The centres of markers are then identified using a centre-of-mass operator. An IMLIB window is updated at this point to show the positions of the markers. To remedy the curvilinear distortion exhibited by the wide-angle lens, a calibration model is applied to the image-space coordinates.

Record Update—At this point the markers in a given image have been identified but are not yet "localised" to the existing set of markers currently being tracked by the software. The update phase involves finding existing marker tracker records for a given marker and updating the attached position with the data returned from the previous stages. Higher level tracker structures relating to the robot and objects are also updated. Robots use a special triangular lighting array that encodes both position and direction, and are represented in memory using a high level structure that references individual markers that constitute the array.

Datagram Request Processing—Robots can request from the metrology system information associated with their individual marker arrays, thus acquiring their own position and direction for various behavioural routines. This is achieved by requesting a User Datagram Protocol (UDP) datagram from the tracking system containing a variety of *global knowledge* for the task at hand[4]. Using wireless Ethernet cards and cable-based Ethernet, the robots submit requests for information that are then processed in this phase of the loop. All of the information pertinent to a particular robot is encoded in a string in the UDP packet, which is transmitted back to the originating robot.

Recording Functions—As well as vending realtime live information over the network, the marker tracker also performs recording of marker data for later offline analysis after an experiment. A series of records are deposited in log files containing marker robot and object positional information entries, each associated with a precise μ-second time stamp from the system clock, which is in turn synchronised along with robot internal clocks using a Network Time Protocol (NTP) server.

5. Tracking Technique Between Frames

Several assumptions are made by the tracker to permit effective tracking, so that it has some confidence that a given marker vector X corresponds to a tracked marker vector Y contained in memory. A Euclidean distance threshold is used to determine this: $X = Y$ iff $|XY| < \alpha$ where $\alpha = 15px$.

The use of a Euclidean distance threshold imposes a constraint on the amount of motion permitted in the image for a correct capture, between two frames, and as a consequence imposes a maximum speed on tracked robots operating in the arena. The amount of motion allowed is a function of the speed at which the capture system can fetch from the board and perform the necessary computations. On the 2.4GHz Fedora Core computer used (with an optimised kernel) this is typically around 2.4Hz, meaning a maximum angular velocity of 20-25 degrees/sec was imposed on the robot motors for effective capture, determined empirically[5].

5.1 In-Memory Structures

There are several distinct tracking structures that are maintained in memory for the purposes of tracking, some of which are only applicable after a particular stage of the tracking process while others are persistent throughout the entire tracking operation:

The **list of centres-of-mass** or **COM list** contains the list of coordinate pairs in image space that represent the centres of the markers observed on each iteration of the tracking algorithm. Each entry in the COM list is a coordinate pair in image space, (I_x, I_y).

The **marker list** is the persistent list of markers, maintained throughout the tracking process. Each of the entries in the marker list is a *marker record* containing a unique iden-

[4]In the experiments using MT, this information contained a variety of operators relating to foraging, such as the centroid of the cluster or the homing position. More information can be found in Mataric (1994).

[5]This is a relatively low speed as is far lower than other tracking systems (such as Mezzanine), but further refinements including the removal of the IMLIB stage and a detailed profiling of the program would result in significant improvements and a higher capture rate.

tifier, a coordinate pair from image space (I_x, I_y), a coordinate pair containing the estimated real position in the arena (A_x, A_y), and the recency value r.

The **robot list** is the persistent list of robot displays, maintained throughout the tracking process. Each entry in this list is a *robot record* containing a unique identifier, a reference to the identifiers for the rear markers $R1$ and $R2$, a reference to the identifier for the tip marker (IT_x, IT_y), an estimated central position of the robot display in image space $C = (I_x, I_y)$, estimated real arena position of the centre of the display $T = (A_x, A_y)$, and the estimated orientation angle of the display in image space computed from the coordinates of the lateral rear axis rr_x, rr_y where $rr_x = \frac{R1_x + R2_x}{2}$ and $rr_y = \frac{R1_y + R_y}{2}$, and the tip marker $(IT_x, IT_y):\ \theta = \frac{atan2(rr_y - IT_y, rr_x - IT_x)}{180 \cdot \pi^{-1}}$.

The **object (puck) list** is responsible for maintaining a list of all non-robot markers, which are assumed to that of objects. Each entry in the object list contains a unique identifier and a position pair. This list is used to generate directional headings for the various multi-agent operators.

5.2 Detecting and Extracting Markers: Stages

The most important stage in the overall marker tracking process is the extraction of marker pixels from the original image and the determination of marker centrepoints, the latter being a non-trivial process that involves the analysis of disjoint pixels. The process used is as follows:

Convolution—This step, more commonly referred to as *image smoothing*, convolves the image by applying a Gaussian kernel iteratively over the entire image. This is done with the aim of removing steep changes in the colour between pixels and so ensuring a smooth continuation and more successful extraction of pixels associated with a marker.

Threshold—The threshold stage removes the background in the aim of leaving the pixels associated with markers left in the image. Each pixel is tested against a set of equality values, whose thresholds were determined empirically.

Coalesce—The coalescing function scans over the image and attempts to locate similar regions or markers in the image, grouping disjoint pixels into pixel clusters (markers).

Centre of Gravity/Mass—The COM/COG step examines the detected markers and finds the centre of mass for each. The centre of mass is the coordinate given by averaging each dimension, producing a centre point $x = \Sigma x_i/n, y = \Sigma y_i/n$. The final step involves the update of the marker list. This examines the list of values produced by the COM stage and updates the persistent marker list state using a Euclidean distance threshold.

5.2.1 Applying a Gaussian Convolution

The camera used for the tracking process is an analogue camera, which requires the conversion from analogue to digital form using a frame grabber. This process is prone to digitisation and quantisation errors into the image, producing noise effects in the form of minor fluctuations in individual pixels that affect the isolation of the pixels for thresholding and later marker extraction. It is therefore desirable to remove such variations for a given pixel with a weighted average of its neighbours. The tracking system incorporates a convolution operation to achieve this.

Convolution is a process in which the original image is smoothed by applying a small matrix of values (K) to each pixel in the image iteratively. This matrix —the *convolution kernel*—is initialised at the start of the marker tracking process and remains static or unchanged throughout the entire run. The pixel at the centre of the kernel is redefined as the weighted sum of the pixels that underly all of the other elements contained in the kernel *K*. The whole of the original image $\mathcal{F}$ is convolved by sliding the kernel over the image, starting from the top left and passing through all achievable positions in x and y implementable using two iterative loops. For edge pixels (which are undefined by the convolution process) the software adopts a zero-value policy: the target image $\mathcal{H}$ is initialised to zero values at the start of the convolution process, with edge pixels in $\mathcal{H}$ being left zero if not examined during the application of the kernel.

The smoothing process is performed using a Gaussian kernel. The formal definition of the Gaussian kernel is: $K_\sigma(x, y) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{(x^2+y^2)}{2\sigma^2}\right)$. The smoothing effect produced by the Gaussian assigns greater weighting to pixels at the centre than to those at the periphery of the kernel. This can be justified qualitatively: smoothing suppresses noise by enforcing the requirement that pixels must look more like their neighbours (Forsyth and Ponce 2003). Conversely, uniform smoothing kernels assigns equal weighting to all pixels under the mask; the pixel at the centre has the same significance as pixels at the periphery. The variation in the parameter σ to the Gaussian model affects the weighting assigned to pixels underlying the centre of the mask (Forsyth and Ponce 2003), overcoming ringing[6] effects typical with unweighted convolution (Forsyth and Ponce 2003).

A discrete smoothing kernel in a two-dimensional $2k + 1 \times 2k + 1$ array is used in the tracking system, whose i, jth value is determined according to $H_{ij} = \frac{1}{2\pi\sigma^2} exp\left(-\frac{[(i-k-1)^2+(j-k-1)^2]}{2\sigma^2}\right)$ (Forsyth and Ponce 2003).

5.2.2 Isolating Marker Pixels using Thresholding

Thresholding removes pixels that do not exceed a given numerical value. All pixels in the original image $\mathcal{F}$ are passed through a thresholding algorithm that sets the target image pixel based on whether the red, green and blue components of the original pixel H_{ij} exceed a pre specified threshold, lev.

[6]Ringing is usually found with unweighted convolution where each component of the kernel is 1.0, and is manifested as a series of narrow vertical and horizontal bars in the image (Forsyth and Ponce 2003).

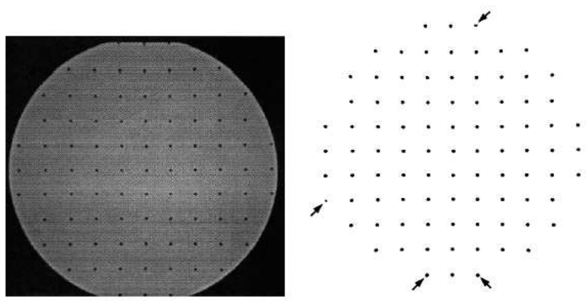

Figure 2: Left—An XRII image of the BB phantom produced by a Thompson fluoroscopy unit, right—the segmented image after background subtraction (Cho and Johnson 1998)

5.2.3 *Coalescing Disjoint Pixels to Find Markers*

The coalescing stage takes the raw information produced by the earlier stages, and attempts to determine the constituent marker that a particular pixel belongs to. The algorithm used for this stage is derived from an earlier algorithm presented by Cho and Johnson (1998) that was designed to detect ball-bearings in images produced by x-ray image intensifiers.

Cho and Johnson's original aim was to find ball bearings shown in Ball-Bearing (BB) 'phantoms' obtained by a variety of x-ray image intensifiers (XRII) systems used in diagnostic radiology and other applications, whose positions once detected could be used to correct for gravitational effects. Figure 2 shows a BB phantom produced by a Thomson fluoroscopy laboratory unit used for high spatial resolution applications. The images produced from the XRII process show similarities with the images produced through the marker tracking process, in particular the image dimensions are similar as are the areas occupied by the BB phantoms. In common with their process, the tracking system produces an image that has the background removed with the remaining logically disjoint pixels assumed to be the visible parts of a marker. These similarities together with the common need for marker coalescing were the main reason for adopting Cho and Johnson's algorithm for the metrology system.

In the Cho and Johnson algorithm clusters are assumed to be clusters of contiguous, thresholded pixels. A row-wise operation is used to examine each pixel and the surrounding neighbours. If a pixel is identified as a cluster member, it is added to the cluster. Otherwise the pixel becomes the "seed" of a new cluster. After this decision step, the algorithm continues to analyse further adjacent pixels that exist after thresholding, but do not possess a cluster ID.

5.2.4 *Finding the Marker Centre of Mass*

The Cho and Johnson process examined earlier also introduced the notion of the centre of mass or COM of a given cluster as a means to finding the central location. It is a straightforward calculation that computes the mean average for all values along a particular dimension, the final values of which can be recombined to form the Cartesian position of the centre of mass in the image $\mathcal{F}$. Using the notation from the original algorithm, the centre coordinates (x_c, y_c) can be computed as:$x_c = \frac{\sum_{i=1}^{N} x_i}{N}$ and $y_c = \frac{\sum_{i=1}^{N} y_i}{N}$.

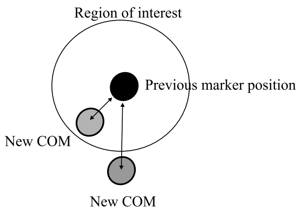

Figure 3: The region of interest around an existing marker.

The result of this processing is a list of marker identifiers and calculated position pairs in image space. The distortion the lens presents to the tracking system is accommodated through the application of a calibration model in which the coordinates are converted into a radial system from the centrepoint of the image at the same angle of incidence (see section 8.). With the new corrected distance from the centrepoint, the coordinates of the image are recomputed into Cartesian floor coordinates in centimetres.

5.2.5 *Updating in-Memory Marker Records*

After the pixel coalescing process, the metrology system attempts to update the lists of markers that are currently being tracked. This is maintained as a list in memory, and is initialised on the first iteration of the tracking system. Each newly identified COM position in the image is checked against the current list of markers being tracked, and whose values are used to update entries on the list where the observed COM position is thought to correspond to a marker under tracking.

The first iteration of the metrology system is perhaps the most important out of all of the iterations. During this first run, the marker lists are populated with an initial set of records one per marker identified in the actual scene. Robots in the arena use a special lighting array so that they can be identified and analysed for directional information, as shown in figure 4. In the initial iteration these displays must be placed in a number of specially designated regions in the arena —*robot regions*— which can be arbitrarily defined beforehand as rectangular regions between two points[7]. Each of the regions are examined for markers, which are constrained to contain the three markers per display. This restriction is necessary to enable the automatic labelling of the robot marker arrays, and hence the initialisation of the robot record

[7]An arbitrary number of these regions can be defined, hence permitting an arbitrary number of robots to be tracked at the same time. There are limitations both in the processing time needed to track multiple robots and the physical space needed for the robot regions (and thus how many robots can be physically placed in the arena at the same time). We have attempting tracking with two to four robots using this tracking system.

list. To label a robot display, MT must know which markers correspond to which robot.

Each of the markers constituting each display is exhaustively examined against its counterparts, with the two markers having the minimal Euclidean distance being designated as the rear markers and the remaining marker being designed as the tip. Each robot array identified in the image is allocated a robot record, which in turn references the constituent markers that compose the array, both the rear and front markers.

Thereafter the update process is as follows: (1) A COM position C is extracted from the newly acquired list of coordinates, (c_x, c_y); (2) This position is compared in a pairwise fashion with each of the markers currently being tracked in the marker list, M_i with positions (m_x, m_y); (3) The Euclidean distance e is evaluated $e = \sqrt{(c_x - m_x)^2 + (c_y - m_y)^2}$. Depending on the value of e, the algorithm either updates an existing record in memory or creates a new one to express a new marker: (1) If the distance value is less than a predefined threshold, then C is considered the new position of the previous marker list entry, which is updated; (2) If the distance value is greater, then a new entry in the marker list is created and the positional values for C are copied to the new record. See fig 3.

Each marker tracking record incorporates a *recency value*: when a successful update takes place, the recency value is incremented to record that the marker was recently updated and accounted for. At every iteration, the distribution of recency values is computed for all of the markers contained in the tracking lists; records that have not been updated by corresponding COMs garnered from the marker identification process —reflected by low recency values— are removed automatically in a periodic pruning process. This is defined as the values below the lower quartile of the distribution.

At each iteration after the initial iteration the robot records are updated. The markers identified in the first stage as being part of the robot display, which have been updated continuously to reflect new positions, are used to calculate the new centre point of the robot record and the orientation. This information is also stored within the robot record. A straightforward set of geometric formulae can thus be used to compute the position (using $r_x = \frac{\Sigma_{i=0}^{i=3}(m_x)}{3}$ $r_y = \frac{\Sigma_{i=0}^{i=3}(m_y)}{3}$) and orientation of the robot displays (using $\theta = 180.0 - arctan2(r_x - tip_x, r_y - tip_y) * \frac{180}{\pi}$). The centrepoint is the centre of mass of the triangular region represented by the markers. The orientation value represents the rotation of the display from the horizontal (converted from the internal radial encoding to degrees largely for interpretability).

Any markers that are not found to be part of a robot light display are assumed to be those of objects, with one marker being attached to each object. This kind of extension is useful for tasks where the collection of objects from the environment using a gripper is needed. In the case of the task used with this tracking system (foraging), the proximity of the centrepoint of each robot record is repeatedly compared with the position of an object record, together with an analysis of the

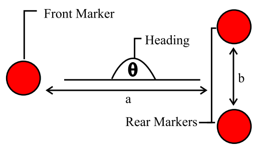

Figure 4: The marker arrangement used on the robot displays.

directional heading of the robot compared to the direction of the object from the centrepoint of the robot. These two comparisons are then used to determine if proximity is seen and whether a grasping situation is possible. Such information can then be used by the robot in the selection of suitable behaviours. Alternative possibilities include the use of sonar sensors to detect suitable objects.

6. Providing Realtime Data over UDP

The mobile robots are also able to interact with the tracking system to obtain what Parker (1993) terms *global knowledge*: information about the task from a global reference frame. The request for information and the response are both encapsulated in datagram packets that are sent via the mobile robot wireless Ethernet access point to and from the tracking computer, which is connected by wire to the laboratory Ethernet.

The server resides on a high UDP port number, listening for incoming UDP request packets. Upon receiving a request, the metrology system interrogates the robot list record for the particular robot, and computes a number of different pieces of information and sends it back out to the querying agent. Some notable elements in the packets include the current angle to the centroid needed to perform clustering with the other agents in the arena, $\delta\theta_c$, the angle away from the centroid needed to disperse away from the other agents in the arena, $\delta\theta_d$, whether the robot is in a good grasping situation, whether the robot is in the home region, and whether the arena is in a simulated "night time". These operators were derived from earlier work by Mataric (1994), whose task description was employed in our underlying investigations.

Information such as homing headings are used to manipulate the behaviour of the robot over time, however their appearance some time after the original request will inevitably mean that the information will have applied to time points in the recent past. This is a kind of temporal error, and we recognise that it will occur in the realtime aspects of the approach. Low-level avoidance routines in the behavioural architecture will utilise current sonar data to perform avoidance, allowing the tracking system-dependent aspects to be corrected as information can be obtained. The use of UDP packets attempts to correct for this error partially, by minimising the network latency and hence improving the speed of data acquisition, however some predictive mechanism would be ideally needed to correct for temporal error.

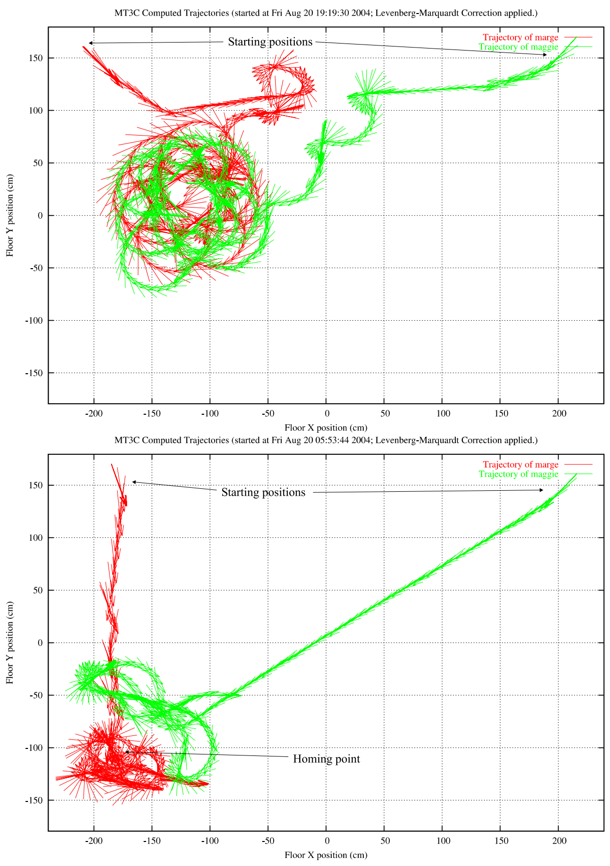

Figure 5: Top - a clustering multi-agent operator being used, and bottom - the homing operator being used by two Pioneers.

7. Metrological Plots

From the recorded data, a graphical presentation of the marker trajectories over time both possible and is a useful means of qualitatively analysing the effectiveness both of individual behaviours and the performance of the learning system in general.

We define a *metrological plot* to be a 2D vector plot generated from the perspective of the tracking camera. Some example plots are shown in figure 5. Using the recorded stream of data, a series of vectors can be computed from the positional data, and plotted for a portion of time or for the entire tracking session[8]. These plots were generated using GNUPLOT, a freely-available plotting system for UNIX. By using the vectorial plotting functions in GNUPLOT, the path for the robot can be shown as a series of directed arrows through the arena. Additional benefits include a variety of export formats using either GNUPLOT or other freely-available image conversion programs, including but not limited to vector-type Postscript outputs suitable for scaling and publications.

8. Calibrating the Camera Lens

The use of a wide-angle lens offers significant advantages in terms of arena coverage, but presents new problems in the form of a significant curvilinear distortion in the captured image. The presence of this distortion hinders effective tracking and several operations involving the calculation of angular data between markers are less accurate as a result. Performing a calibration procedure and forming a calibration model of the distortion accommodates both the distortion and allows for the conversion between image and floor space, that coordinates in image space can be resolved to positions on the arena floor in terms of actual physical measurements (cm).

8.1 Calibration Model

The lens exhibits a *radial distortion* caused by a position in the image moving from the expected position in a positive or negative direction from the centre of the image plane. A positive shift causes a pin-cushioning effect, whilst a negative shift causes the observed barrel distortion (Nakamura et al. 2002).

The method used is to calibrate the length from the image centre C_{img} to a point in image space $I_p = (I_x, I_y)$ against the length to the point in the arena $A_p = (A_x, A_y)$ from the centre of the floor, C_{floor}. Under a *ground plane constraint*[9], taking pairs of measurements in this way produces a set of data points that captures the deviation from the expected position on both the arena floor and the corresponding point position in image space. The floor point underlying the centre of the camera point is used for C_{floor}. This technique assumes that the centres of the image, lens and floor are at the same point, and the lens has a "uniform" fall off (such as parabolic). Using a fourth-order polynomial $y = f(x) = Ax^4 + Bx^3 + Cx^2 + Dx + E$, the distortion exhibited by the lens can be modelled effectively using a polar coordinate system where the radius from the centre of the image is recorded against a similar distance from the centre of the arena floor. Calibration thus reduces to estimating the approximate values for A, B, C,D, and E—the *distortion coefficients* (Ma et al. 2003).

An effective method of calculating these values, which effectively amounts to a curve fitting procedure, can be found in the nonlinear least-squares (NLLS) Marquardt-Levenberg algorithm[10]. The calibration procedure amounts to the repeated acquisition of pairs of real-world and image space centerpoint distances. The measurement pairs are passed to the ML procedure, which fits the polynomial to the data. The resultant polynomial can then be used to convert radial distances in image space to corresponding radial distances on the actual workspace. The polar coordinate system can be converted

[8]We found the reliability of the approach described in this paper to be fairly good, with a majority of trials being successfully tracked from start to finish. One limitation arises in the dependence of agent-based perception on the tracking system, which consequently requires continuous successful tracking.

[9]That the points for the markers lie on a level plane.

[10]The coefficients were determined using the GNUFIT component of GNUPLOT, which implements the M-L procedure.

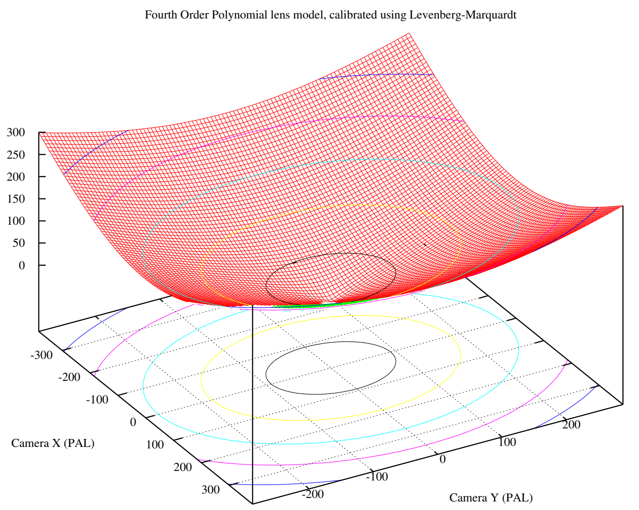

Figure 6: The calibration surface.

back into a Cartesian space location using the standard formulae: $x = Ir * cos(\theta)$ and $y = Ir * sin(\theta)$. This gives the floor position in the units used during calibration (in this case centimetres).

The estimated LM parameters can be inserted into the tracking program and used to convert image space coordinates into reasonable estimates of the real world position of the marker in the arena[11]. Substituting into the polynomial gives $f(x) = -2.9797e^{-6}x + 4.25033e^{-9}x^3 + 0.000868696x^2 + 0.455563x + 1.39055$, which is maintained in memory and used by the tracking system.

A 3D plot of the calibration surface that results is shown in figure 6[12]. The vertical axis shows the estimated displacement between the floor centrepoint and the corresponding point in image space, indicating the necessary radial correction shift to obtain the true floor position. The curvature in the visualisation confirms the significant distortion that the wide angle lens possesses.

9. Conclusions

This paper has introduced a novel metrology approach that can be used for gathering offline data concerning the trajectories of multiple active marker displays attached to mobile robots, and can optionally provide information to agents concerning their position and relative heading to various areas of interest, such as the home area, nearest puck or a number of more complex centroid operators such as the centre of the group (convergence or flocking), heading away from the centre of the group (divergence) amongst others. Although prototypical in nature, it is presented it in the hope of encouraging a much-needed literature on mobile robot tracking techniques.

9.1 *Similar Methods*

The tracking system described in this paper is similar to several other freely-available systems that have been devised for tracking markers. Lund et al. (1996), in their tracker developed at Edinburgh, describe a simpler marker tracker system that employs two LEDs mounted on a Khepera mobile robot. *Mezzanine*[13] is another freely-available tracking system that offers similar functionality including a much higher capture rate of 30Hz, and uses more accurate color space processing methods. Other techniques can be found in face recognition (Scassellati 2001).

Several commercial tracking systems are available, offering considerably higher levels of performance and full 3D tracking capabilities. For example, the VICON 512 system by Oxford Metrics Ltd. offers support for precise measurement and fast acquisition suitable for biomechanical problems, incorporating multiple tripod or ceiling-mounted cameras. Such systems differ considerably from the tracking system described in this paper. MT relies on a ground plane constraint, and is thus restricted to the identification of markers on a two-dimensional plane in the arena.

9.2 *Acknowledgements*

The authors would like to thank Ian Izett for technical support and Fred Labrosse for assistance in developing the Video4Linux interface.

References

Cho, P. S. and Johnson, R. H. (1998). Automated Detection of BB Pixel Clusters in Digital Fluroscopic Images. *Physics in Medicine and Biology*, 43:2677–2683.

Forsyth, D. A. and Ponce, J. (2003). *Computer Vision: A Modern Approach*. Upper Sadle River, NJ: Prentice Hall.

Lund, H. H., de Ves Cuenca, E., and Hallam, J. (1996). A Simple Real-Time Mobile Robot Tracking System. Technical Report 41, Department of Artificial Intelligence, University of Edinburgh.

Ma, L., Chen, Y., and Moore, K. L. (2003). A New Analytical Radial Distortion Model for Camera Calibration. Unpublished Paper.

Mataric, M. J. (1994). *Interaction and Intelligent Behavior*. PhD thesis, Massachusetts Institute of Technology, MA: Boston.

Nakamura, S., Aoki, Y., Hasimoto, S., and Hata, K. (2002). Image Distortion Correction for Accurate Image Measurement. In *The Eighth Korea-Japan Joint Workshop on Computer Vision (FCV2002)*, pages 163–170.

Parker, L. E. (1993). Designing Control Laws for Cooperative Agent Teams. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, pages 582–587. Incomplete.

Scassellati, B. M. (2001). *Foundations for a Theory of Mind for a Humanoid Robot*. PhD thesis, Department of Computer Science and Electrical Engineering, MA: Boston.

[11]The markers were calibrated from a level of several centimetres from the arena floor. In practice the markers on the tracking arrays were marginally higher than the height of the robot.

[12]Although the 3D plot is useful for visualising the lens distortion, the actual calibration function is two-dimensional.

[13]`http://playerstage.sourceforge.net/mezzanine/`