Improving the Accuracy of a 3D Motion Capture System

By Neil Robinson

MSc Artificial Intelligence
University of Edinburgh
1999

This project explores the possibility of improving the accuracy of a motion capture system. The motion capture system in question uses specially designed spherical markers to analyse the motion of test subjects. There are two questions this project is asking on this topic. The first is whether or not grey level image information can improve the accuracy of marker location compared with just binary level information. The second is to find if there are techniques for dealing with the situation when markers are partially occluded.

The project will entail the implementation and testing of algorithms which perform marker location in order to answer these questions.

I would like to thank my supervisors for their assistance with this project :-

Bob Fisher

Mark Wright

Overview

Motion capture system

The Edinburgh Virtual Environments Centre (EdVEC) is working on a system for motion capture. This systems purpose is to record and analysis data that represents complex objects in motion. Potential uses of such a system include analysis of sports men and women, and medical applications for helping people with walking difficulties.

The system works by attaching various spherical markers to the object of interest, generally a person. The markers are placed such that their motion provides a large amount of information about the overall motion of the object. The data capture is done is a specially set up studio. There is a central workspace where the motion events occur, this space is surrounded by 8 cameras which record the motion. The cameras each produce their own light source using an array of LED's that emit light of a specific frequency. This frequency corresponds with the optimal reflectivity of the markers; thus the images produced by the cameras are hyper sensitive to the location of the markers. An example of the system in use might be recording motion data form a dancer. The dancer would have a number of the markers attached to him/her and then made to dance in the studio workspace. The movements of the markers would be picked up by the system from the cameras, and from this data the system would create a representation of the motion events occurring.

Data is sent from the cameras to a computer running software for analysing the images. The software uses the images from the 8 cameras to recreate in 3D the positions of each of the markers in the scene at each time interval. This evolves various image processing techniques such as thresholding the image to remove background noise, locating the markers within the image, and converting image co-

The Problem

The overall aim of the project is to improve the accuracy of the marker detection used by the system. Improved accuracy at the level of marker detection should translate to improved accuracy at the higher levels of 3-D motion capture, thus there are clear potential benefits for this investigation.

In order to locate the true 3D location of the markers, it is first necessary to locate their position within the image. Once this has been done the true world location of the markers can be calculated (this is explained in the next chapter).
The image acquired from the cameras can be thought of as 2 dimensional array of values representing image intensity. The current system thresholds this image so that only the markers should show up. Since the resulting image is binary a binary centre of mass algorithm can be used to determine the image centre position of each of the markers present in the image.

There are two areas of investigation that will be explored in the course of the project.

The second is to develop techniques for dealing with the possibility that markers are partially occluded by an object. When this is the case the existing algorithm will miscalculate the centre of the marker since its appearance in the image is incomplete.

The details of these will be discussed in the relevant chapters of this report. Exploring these ideas will require both the design and implementation of new algorithms for marker location, and the proper scientific analysis of their performance compared with the existing algorithms.

Background

This section covers important issues related to computer vision and image processing and how they are relevant to this project.

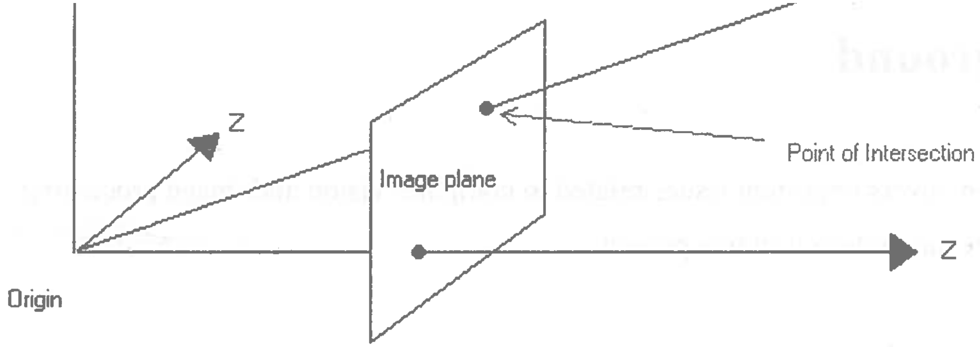
Image formation

The process by which images are formed by a camera can be modelled by two separate processes. The first is that of perspective projection, this maps points in a scene onto the image. In terms of this project the process of perspective projection thus provides a relationship between a markers position relative to the camera and its position in the image. Since we are interested in finding the location of a marker from an image containing it, this is clearly of importance.
The perspective projection equation indicates where a line that connects a point in the centre and the aperture of the camera intersects the image plane (the part of the camera where the image is formed).
It is generally simpler if the world co-ordinates used in the equation are such that the image plane is **z = constant** and the focal point is at the origin. In this case the equation is: -

x intersect = Px * constant / Pz

y intersect = Py * constant / Pz

where P is the co-ordinates of a point in the scene



For a real camera the image plane would clearly lie behind the origin, placing it in front here is done to make the diagram simpler.

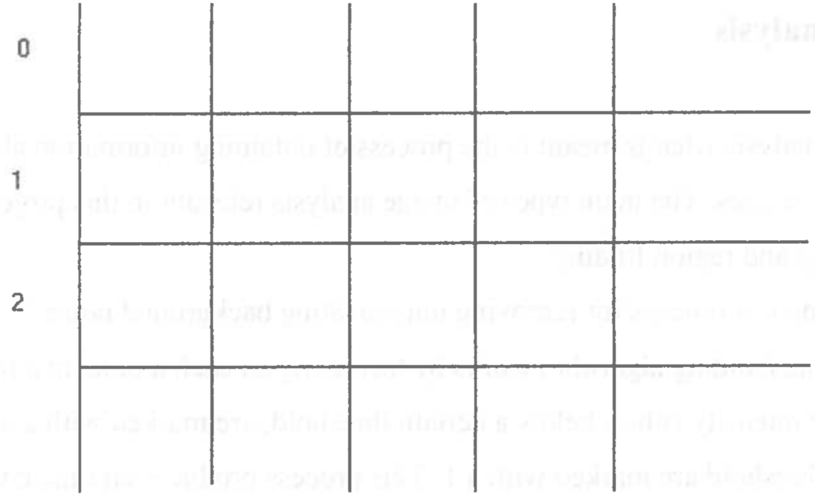
The position of the intersect point in the imaging plane indicates the image co-ordinates of the point P. The imaging plane co-ordinates consist of an **x** and **y** component with origin in the centre of the image plane. The image itself consists of individual pixels, rather than being a continuum of intensity values. Pixels are referred to by real number co-ordinates with the origin at the top left corner of the image. The pixel co-ordinates are **i** (which points down), and **j** (which points right). The relationship between image co-ordinates and image pixel co-ordinates is described by the following equations: -

$$x = j - (m - 1) / 2$$
$$y = -(i - (n - 1) / 2)$$

where

m is the number of pixels in the horizontal

n is the number of pixels in the vertical



The top left hand corner of an image

The second process is that of image irradiance. This is an equation that relates the intensity of the image at various points to the light incident on the surface point that is mapped to this image point and the reflectance function of the surface.

Ideally, a mathematical model of the set-up used for the motion capture system could be constructed based on these processes. Such a model would provide a functional mapping between any given scene (including camera position) and the resultant image. This model could then form the basis of algorithms for locating markers. Unfortunately, this is not feasible for a number of reasons. The exact reflectance function of the markers is unknown. Each camera uses a large number of LED's to generate illumination, this makes the resultant equations for image irradiance very complex (a large number of simultaneous equations). The position of the imaging plane in the camera and the optical axis are also unknown.

However, it is still useful to gather some aid from the field of image formation analysis. Looking at the images of markers it seems safe to assume that, although the exact reflectance properties are unknown, they contain a large lambertian component.

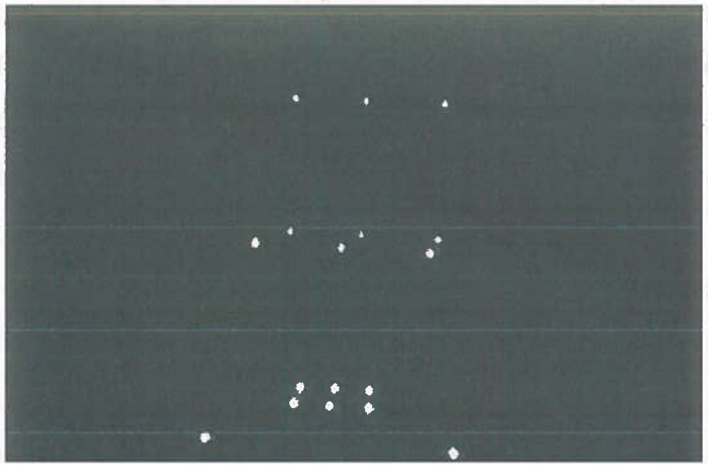
Image Analysis

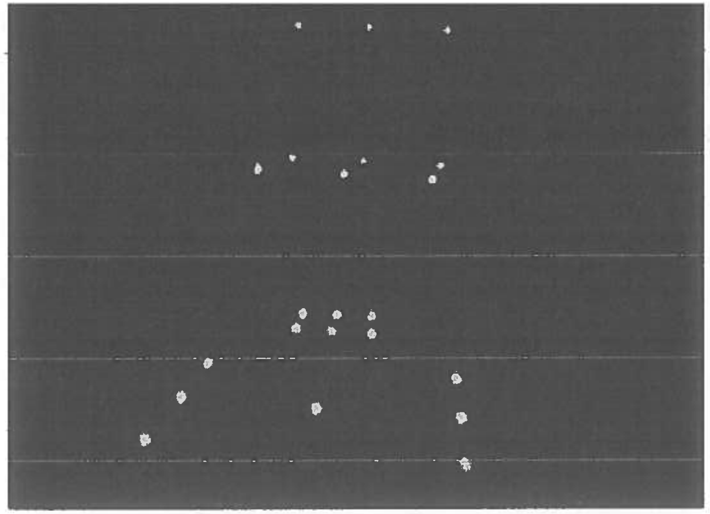
By image analysis what is meant is the process of obtaining information about the scene from images. The main types of image analysis relevant to this project are thresholding and region finding.

Thresholding is a process for removing uninteresting background noise from an image. A thresholding algorithm works by traversing an each a pixel at a time. Pixels, which have intensity values below a certain threshold, are marked with a 0. Pixels above the threshold are marked with a 1. This process produces an image where each pixel has the value 1 or 0. The 1s should indicate the location of interesting objects and the 0s the background. This obviously assumes that the background has lower intensity than the foreground. For this project that should be the case since the markers are specially designed to reflect the light produced by the cameras.

The threshold constant can be chosen by hand or an algorithm can compute it from the image. There also exist adaptive thresholding algorithms that have variable thresholds, but they are used to deal with variable illumination, which should not be present in this system.

Below is an image of a number of markers followed by an image after thresholding.

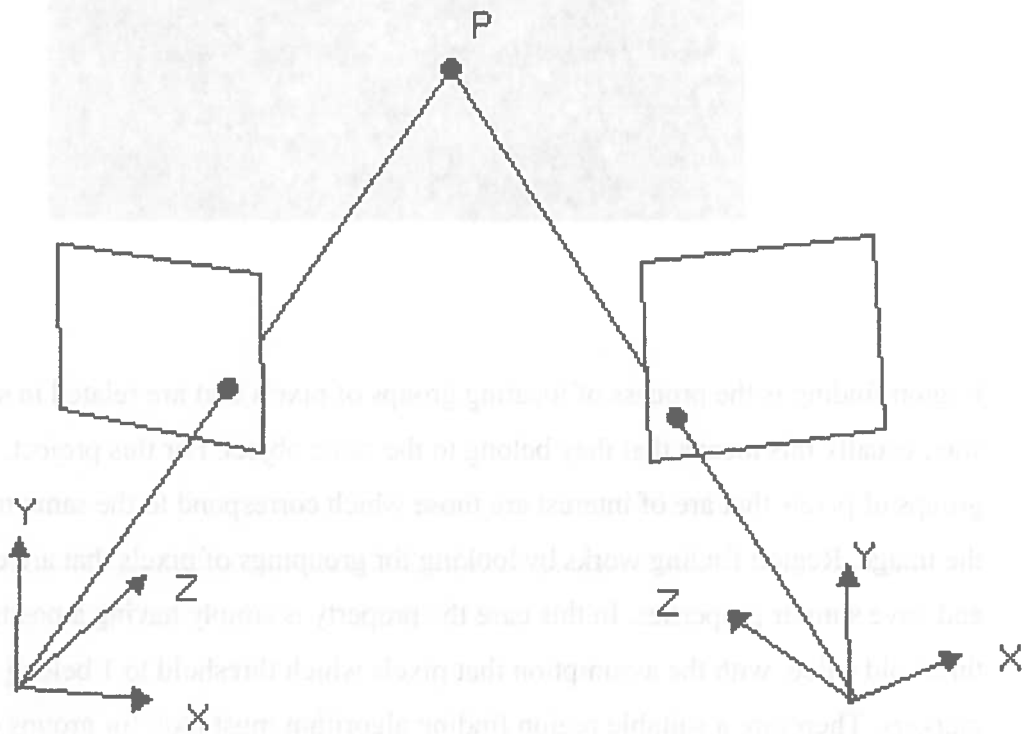




Region finding is the process of locating groups of pixels that are related in some way; usually this means that they belong to the same object. For this project, the groups of pixels that are of interest are those which correspond to the same marker in the image. Region finding works by looking for groupings of pixels that are connected and have similar properties. In this case the property is simply having a positive threshold value, with the assumption that pixels which threshold to 1 belong to markers. Therefore a suitable region finding algorithm must look for groups of connected 1s in the image. In the resulting image of this process, pixels belonging to the same group are given the same labelling. Thus by choosing a particular label it is pick out all pixels that correspond to a particular marker.

The combination of thresholding and region finding produces the information needed for the centre location algorithms that this project is investigating. These take a list of pixels belonging to a particular marker and calculate a centre position for this marker. It is important to note that the centre position being referred to here is the centre position of the image of the marker, it is a point in the image plane.

in the scene that maps to this point in the image. Using two cameras results in two lines. The intersection of these two lines will be the true position of the centre of the marker in the scene (relative to the camera co-ordinates).



Testing the performance of marker location algorithms

In order to evaluate whether or not new techniques for locating the position of the markers in the image could improve the accuracy of the system it is first necessary to devise a test procedure that the different algorithms can be put through and compared. It is important that the methods used to test the algorithms are in some way representative of the situations that would occur when the system is in actual use, thus good performance on the tests should equate to good performance in a real situation. Any testing method would clearly need to make use of the new algorithms in evaluating data from the cameras of the markers.

The most obvious test would be to incorporate the new algorithms into the existing system, replacing the existing software for performing marker location. The system could then be used in an experiment resembling the motion capture for which it is intended and its performance with the different algorithms assessed. Unfortunately it was not possible to incorporate any new code into the existing system. Hence other methods had to be found to test the new algorithms.

The first method that was considered was the design and implementation of a system for locating the position of markers. The system would use two cameras from the motion capture system. These would be connected to a computer running software, which would use information from the two cameras to calculate the position of any markers in front of the cameras. By comparing the computer calculated position of a given marker can be calculated using the equations previously discussed.

There were, however, serious problems with this approach. These problems all related to the fact that this method is dependant on a number of facts to be known about the world and that these facts must be known to a high degree of precision. Facts that need to be known are: -

The location of the camera.
This includes both the position and orientation, since the direction of the viewing axis must be known. These are needed for the calculations that determine the true position of markers in the image.

The location of the markers.
In order to compare the accuracy of the different algorithms in determining the position of markers it is necessary to be able to place markers in known positions in front of the cameras. This is essentially providing a ground truth against which the calculations can be compared.

The physics of the imaging process.
This concerns the processes that lead to the production of the image on the imaging plane. What is needed is the mathematical model of the system of the image irradiance type discussed previously.

These quantities must be known to a high degree of precision due to the nature of the project. The project is attempting to test potential improvements in accuracy, it is therefore necessary that all factors influencing the tests are controlled precisely. If this

What was needed was a method of testing the new algorithms that did not depend on knowing precisely the optics of the system. Making use of the cross ratio invariance property made this possible. The cross ratio property is a mathematical formula that can represent certain arrangements of objects in a scene, and is invariant on the location of the viewing camera. The cross ratio property applies to arrangements of four or more objects that lie in a straight line. When this is the case a cross ratio value can be calculated and this value can be recovered from any image containing this arrangement of objects.

The cross ratio for four objects is calculated using the following formula: -

Cross-ratio = (Distance13 * Distance24) / (Distance14 * Distance23)

Where

DistanceXY = The distance between points X and Y

The distance between any two objects can be either their physical distance or the distance they are apart on the image (although clearly the decision must be kept for all calculations in any given evaluation of the cross ratio).

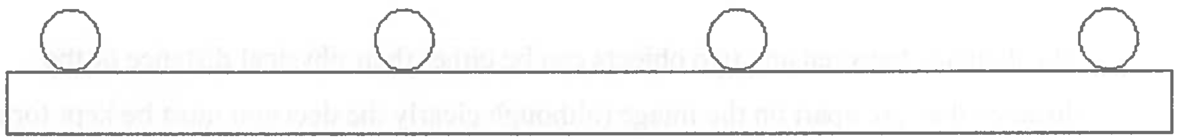
Since the cross-ratio can be calculated from the distance markers are apart in the image, it is not necessary for the system that is using the cross ratio to perform the calculations to convert image position into global positions.

Using the cross ratio as the basis of the tests also means that an absolute position of the markers does not need to be measured. Only the positions of the markers relative

To use the cross ratio property as a basis for testing the accuracy of different algorithms would require the construction of a system that could calculate the cross ratio value for a bar placed in front of a camera. The system would use the various new algorithms for the calculation of the image marker positions, hence allowing an assessment of their accuracy to be made.

The bar was a metal spirit metre with 4 markers attached to it. The markers were placed in a line at equal distance intervals. A brief computer simulation indicated that the cross ratio was least susceptible to errors when the 4 points were equidistant, hence the choosing of this arrangement. This simulation evolved testing a large number of different configurations of marker positions and for each making a variety of small random changes to the marker positions and measuring the error produced by these small changes to the cross-ratio. The bar was machined at the university workshops to provide as high an accuracy of the marker positions as possible.

Cross-ratio bar with markers at equal distances



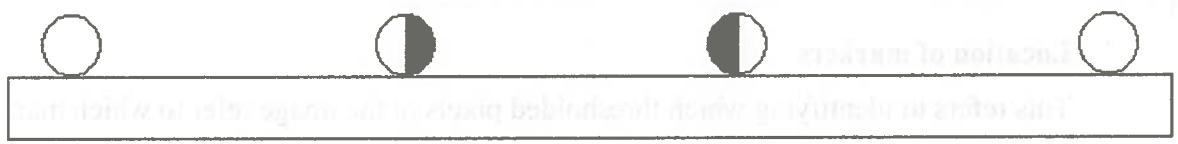
Testing the accuracy of an algorithm involves comparing the cross ratio calculations the system makes from images of the bar with the calculations made for different algorithms. Clearly a large number of images of the bar would be needed to provide a statistically meaningful assessment.

This set of images shall be referred to as test data. Test data was obtained by recording images of the bar at different positions and orientations relative to the

the camera was measured as the distance between the lens and the centre point of the bar, this is clearly an approximation. For each individual test image the bar was placed at a position such that the centre of the bar was either 3m or 6m from the lens of the camera. The bar was held in position by a tripod stand the height of the centre point of the bar being equal to the height of the camera lens. The bar was then rotated to some arbitrary orientation and an image captured.

As well as different distances, test data was obtained for both non-occluded and occluded markers. Occlusion was simulated by having black tape wrapped around half of the two centre markers. In total there were four separate sets of data obtained for the experiment. These sets derive from the combinations of the two distances and whether or not the markers were occluded. For each of the sets 60 test images were obtained.

Cross-ratio bar simulating occlusion



The test images were given as input to the software for computing cross-ratio. The software then calculated the cross-ratio for each image using each of the separate algorithms for marker position location. For each of the four test sets this led to a set of cross-ratio values for each of the algorithms. With this data statistical analysis of the effectiveness of the different algorithms under the four test situations could be carried out.

A brief description of the software written for the tests follows; implementation issues not directly relevant to the project are not covered.

3. **calculate centre positions of each marker**
4. **calculate cross-ratio**

The third step is the job of the new algorithms that this project is attempting to investigate. These are covered in the next chapter.

The other 3 steps are detailed as follows: -

Thresholding

Thresholding is not a serious problem for these images because the markers are designed to reflect the light given out by the cameras they are significantly brighter than anything else in the image.

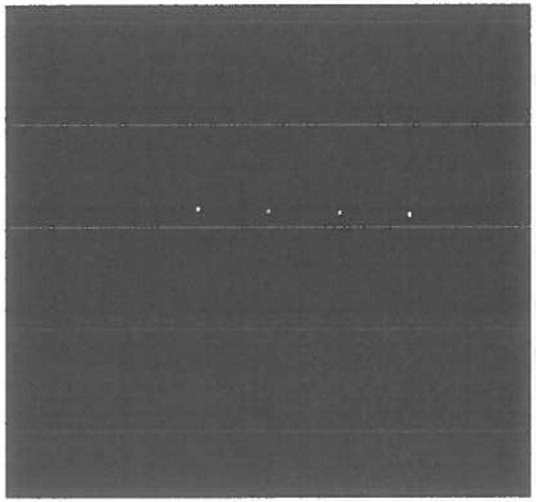
The thresholding algorithm used was taken from the book **Machine Vision, Jain Kasturi and Schunck** (see appendix).

Location of markers

This refers to identifying which thresholded pixels in the image refer to which marker. Since the markers are spatially separate from each other and the pixels belonging to them form a connected structure, this problem amounts to a region finding problem. The algorithm used was taken from a web site of computer vision lectures (see appendix).

Calculation of cross-ratio

The equation previously discussed, although the marker positions must first be sorted so that the correct marker location is used for the equivalent term. Although it is purely arbitrary which marker is used for which term in the equation, it is important that the same mapping is used for each calculation. Otherwise different values will be



The above image is of the cross-ratio bar in a horizontal position. The markers shall be numbered one to four, going from left to right.

The image positions of the four markers are calculated as follows: -

Point 1

i = 187 j = 198

Point 2

i = 255 j = 200

Point 3

i = 325 j = 202

Point 4

i = 394 j = 204

From these four positions the cross-ratio can be calculated from the formula.

Distance23 = √((255 – 325)^2 + (200 – 202)^2) = 70

Cross-ratio = (138 * 139) / (207 * 70) = 1.32

This can be compared with the true cross-ratio of the bar. If the length of the bar is said to cover an arbitrary distance between 0 and 1 then, since the markers are all equidistant apart the distances area as follows: -

Distance13 = 0.6667
Distance24 = 0.6667
Distance14 = 1.0000
Distance23 = 0.3333

Cross-ratio = (0.6667 * 0.6667) / (1.0000 * 0.3333) = 1.33

Algorithms for marker location

This chapter discusses the algorithms used for locating markers. These algorithms are integrated into the system discussed in the previous chapter. The design and testing of these algorithms is the core purpose of the project. From a computational perspective these algorithms can be thought of as follows: -

Input

An image of a scene along with the information telling it which pixels correspond to the marker it is supposed to be locating

Output

The image position corresponding to the centre of the marker.

There are four algorithms that will be examined, each adhering to the above definition.

Binary centre of mass

This algorithm considers binary information about the image. Pixels that cover the marker are considered to have a value of 1. All other pixels have the value of 0. The grey level information representing the intensity values of different pixels is ignored. The output values for the x and y locations of the marker centre are calculated from the below formulae: -

x = Average for all pixels of (pixel value * x reference of pixel)
y = Average for all pixels of (pixel value * y reference of pixel)

This is the algorithm used by the current system.

In pseudo code the algorithm used is: -

```
Variables x_total, y_total, pixel_count      (all starting at zero)

For each pixel in the image
    If pixel is part of the marker
        Add x reference of pixel to x_total
        Add y reference of pixel to y_total
        Increment pixel_count

x return value is x_total / pixel_count
y return value is y_total / pixel_count
```

Grey level centre of mass

This algorithm is a variation of the binary centre of mass. Like the binary centre of mass in computes an average of the x and y values of pixels which correspond to the marker. However, instead of ignoring grey level information, the grey level centre of mass algorithm uses the intensity values of pixels such that pixels with higher intensity values contribute proportionally more than those of lower intensity values.

Mathematically the centre of mass is calculated according to the equations: -

x = (Sum for all marker pixels of (pixel intensity * x reference of pixel)) /
(Sum for all marker pixels of (pixel intensity))

mass.

The algorithm used is: -

```
Variables x_total, y_total, intensity_sum          (all starting at zero)

For each pixel in the image
    If pixel is part of the marker
        Add x reference of pixel * intensity value to x_total
        Add y reference of pixel * intensity value to y_total
        Add intensity value of pixel to intensity_sum

x return value is x_total / intensity_sum
y return value is y_total / intensity_sum
```

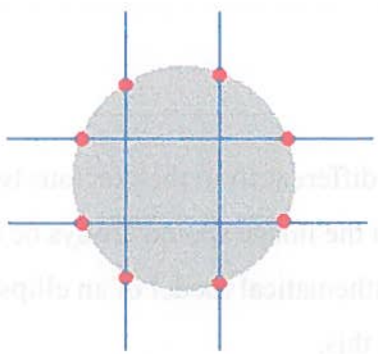
Ellipse fitting

This algorithm is considerably different from the previous two. It is based on the fact that the shape of the markers in the image should always be an ellipse, therefore it should be possible to fit the mathematical model of an ellipse to the image and then derive the centre position from this.

The algorithm consists of three parts. The first part calculates an large number of points on the edge of the marker in the image. The second uses these points as the input for an ellipse fitting algorithm. The third is the calculation of the centre point from the equation of the ellipse. These three parts will be discussed individually

Points are calculated from the image positions where the intensity crosses a certain value. It is assumed that the image of the markers consists of a set of concentric ellipses of decreasing intensity, therefore any 'threshold' intensity value can be used to derive the edge points and still yield an ellipse with centre at the marker centre. An intensity value around half the maximum was chosen as this was considered to be more error tolerant than picking a very low value.

The edge points were calculated by scanning across each horizontal and vertical scan line in the image and adding the x and y values of any point were the intensity value crossed the threshold.



The above diagram shows the process by which scan lines (the blue lines) produce edge points by their intersections with the edge of the marker (the red blobs).

The exact position of the point is calculated using a linear intersection technique. This

Consider a horizontal scan where a crossing point has been located

The threshold value is 20

The pixels under consideration are

x = 6, y = 4, intensity = 10

x = 7, y = 4, intensity = 40

The change of intensity is 30 and this occurs linearly between x = 6 and 7

The threshold will thus be crossed a third of the way between x and y

(20 – 10) / 30 = 1 / 3

The x point of crossing is thus the x value of the first pixel plus a third

x = 6.33

The y value is just the y value of the horizontal scan

y = 4

The algorithm used: -

```
For each horizontal row of pixels
    For each pair of adjacent pixels
        If threshold lies between pixel intensity values
            Add point to edge_point_list
                Where
                    x value of point is the x value of linear
                                intersection
                    y value of point is y reference of current row

For each vertical column of pixels
    For each pair of adjacent pixels
        If threshold lies between pixel intensity values
            Add point to edge_point_list
```

The ellipse-fitting algorithm used is taken from the book **Introductory Techniques for 3-D Computer Vision by Trucco and Verri.**
It works by building a matrix formula that incorporates the points to be fitted. This formula matches the generalised eigenvalue problem. Finding the eigenvalues allows the parameters of the ellipse to be obtained. This process produces the ellipse parameters that minimise the error of the data points used.

From the ellipse parameters the centre point of the ellipse and consequently the centre point of the marker can calculated: -

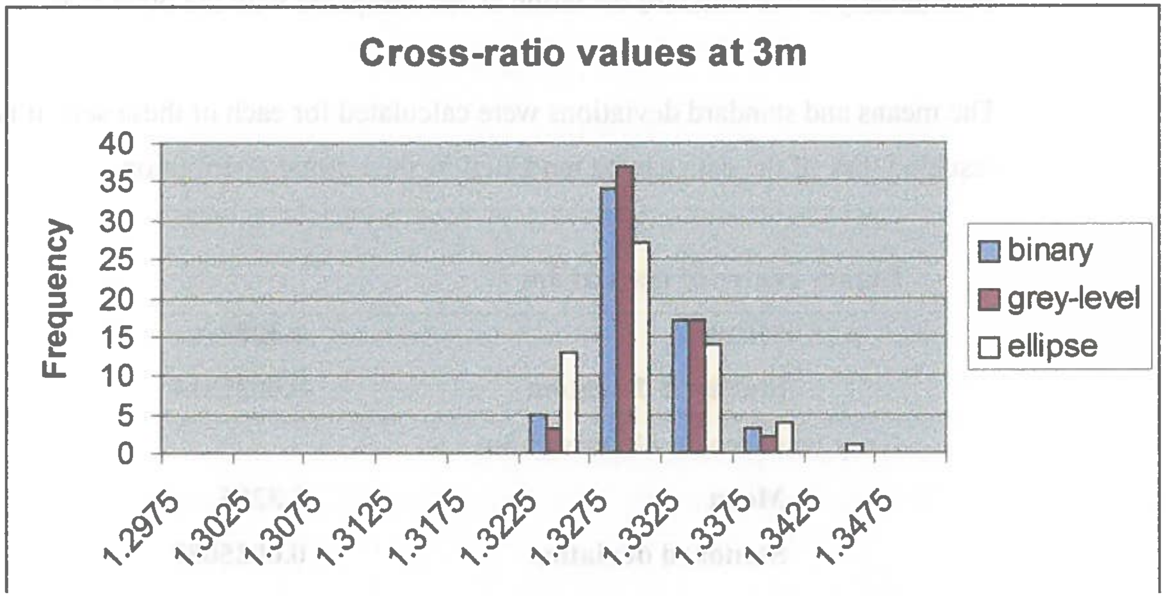
Ellipse fitting with occlusion filtering

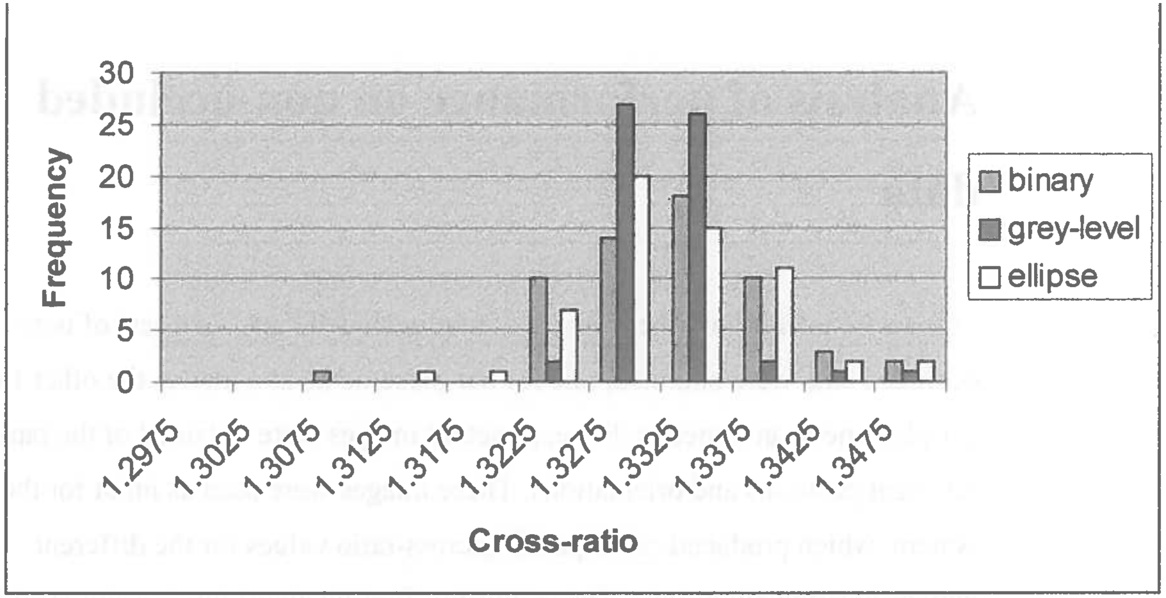
This algorithm is intended to be able to deal with markers that are partially occluded. The image centre of an occluded marker as calculated by one of the previous algorithms will not correspond with the true centre of the marker since these algorithms do not take into account the hidden parts of the marker.
This algorithm is similar to the standard ellipse fitting algorithm, the difference being that it filters out edge points which it believes to be part of an occluded edge of a marker. Assuming that the markers are approximately lambertian, the changes in intensity in the image along a straight line through a marker should be gradual. If a marker is occluded, however, there should be a sudden change in intensity. It should therefore be possible to detect edges that are caused by occlusion, and ignore them.
The algorithm finds a set of edge points, as with the standard case, but rejects any that matches the occlusion criterion. After this the algorithm is identical to the ellipse-fitting algorithm above.

```
                difference between pixel intensity values is less than
                occlusion filtering limit

            Add point to edge_point_list
                Where
                    x value of point is the x value of linear
                                            intersection
                    y value of point is y reference of current row

For each vertical column of pixels
    For each pair of adjacent pixels
        If threshold lies between pixel intensity values and
                difference between pixel intensity values is less than
                occlusion filtering limit

            Add point to edge_point_list
                Where
                    y value of point is the y value of linear
                                            intersection
                    x value of point is x reference of current row
```

Analysis of performance on non-occluded data

This section deals with the analysis of non-occluded data. Two sets of non-occluded data were obtained, one for bar placements at 3 metres the other for bar placements at 6 metres. For each set 60 images were obtained of the bar at different positions and orientations. These images were used as input for the system, which produced corresponding cross-ratio values for the different marker location algorithms. The occlusion-filtering algorithm was not used for this set of data. Consequently six sets of output were obtained from this experiment, each of the first three algorithms with both sets of input data. The output consisted of sets of cross-ratio values, 60 in each set. The data obtained was plotted in histogram graphs.





Initial observations from these graphs are that there is significant overlap of the distributions (this is important because it makes differences in standard deviation easier to analyse), and the grey-level centre of mass algorithm has considerably lower standard deviation at 6m compared with the other two.

The means and standard deviations were calculated for each of these sets. It is assumed that all the data can be modelled by the normal distribution.

Binary centre of mass at 3m

Mean	**1.3289**
Standard deviation	**0.0031514**

Grey level centre of mass at 3m

Mean	**1.3286**
Standard deviation	**0.0025689**

Standard deviation	**0.0077261**
Grey level centre of mass at 6m	
Mean	**1.3304**
Standard deviation	**0.0036122**
Ellipse fitting at 6m	
Mean	**1.3310**
Standard deviation	**0.0061077**

At three metres the standard deviation for the ellipse-fitting algorithm is higher than for the other two. The binary standard deviation is slightly higher than the grey-level. At 6m the standard deviation for all the algorithms is higher. This makes sense, essentially the calculation of the cross-ratio is less accurate at longer distances. Of most significance here is the performance of the binary algorithm compared with the other two. The ellipse fitting and grey level centre of mass algorithms have lower standard deviations than the binary level centre of mass. The grey level centre of mass algorithm has a standard deviation a quarter that of the binary algorithm. The ellipse-fitting algorithm has gone from having a higher standard deviation at 3m to having a standard deviation substantially lower than the binary algorithm. The fact that the algorithms using grey level information are more stable at the longer range could be explained by the fact that the grey-level information is more important at further distances. This may be because there are less pixels for each marker and consequently each pixel covers more of the marker, thus the actual intensity values of the pixels gives more information about the marker. Also, at close ranges there was a noticeable saturation effect that caused the centre of the markers to have a uniform intensity value. Due to this grey-level information would have been lost, this may also have a reflection on the relative improved performance of the intervals for the mean and standard deviation were computed for each distribution at 95 percent level of certainty.

Binary centre of mass at 3m	
Lower bound for mean	**1.3281**
Upper bound for mean	**1.3297**
Lower bound for Std	**0.0026988**
Upper bound for Std	**0.0036520**
Grey level centre of mass at 3m	
Lower bound for mean	**1.3280**
Upper bound for mean	**1.3293**
Lower bound for Std	**0.0022000**
Upper bound for Std	**0.0029770**
Ellipse fitting at 3m	
Lower bound for mean	**1.3276**
Upper bound for mean	**1.3298**
Lower bound for Std	**0.0035993**
Upper bound for Std	**0.0048706**
Binary centre of mass at 6m	
Lower bound for mean	**1.3296**
Upper bound for mean	**1.3335**
Lower bound for Std	**0.0066166**
Upper bound for Std	**0.0089535**
Grey level centre of mass at 6m	
Lower bound for mean	**1.3295**
Upper bound for mean	**1.3314**
Lower bound for Std	**0.0030935**
Upper bound for Std	**0.0041860**

These tests show that the lower standard deviation for the grey-level centre of mass at 6m compared with the binary centre of mass is significant with 95 percent confidence. This can be said because the lower bound of the binary level is still higher than the grey-level upper bound. It is not possible to form, however, any evidence that the ellipse fitting has significantly lower standard deviation since there is overlap between the confidence ranges.

An unexpected result is the fact that there is significant difference in the means at 3m and 6m. This is the case for the two centre of mass algorithms. The mean value for cross-ratio is higher at 6m than at 3m. This indicates that these algorithms have some bias towards estimating higher cross-ratios at further distances.

Since we are interested in how effective the algorithms are, it would be valuable to compare the cross-ratio values obtained from the calculations with the true cross-ratio value of the bar. To allow this, the data is converted such that the new data for each set becomes the difference between the cross-ratio value calculated and the true cross-ratio value.

Once this has been done the mean and standard deviations can be found as before.

Binary centre of mass at 3m	
Mean	**0.0047547**
Lower bound for mean	**0.0040999**
Upper bound for mean	**0.0054095**
Standard deviation	**0.0025661**
Lower bound for Std	**0.0021976**
Upper bound for Std	**0.0029738**

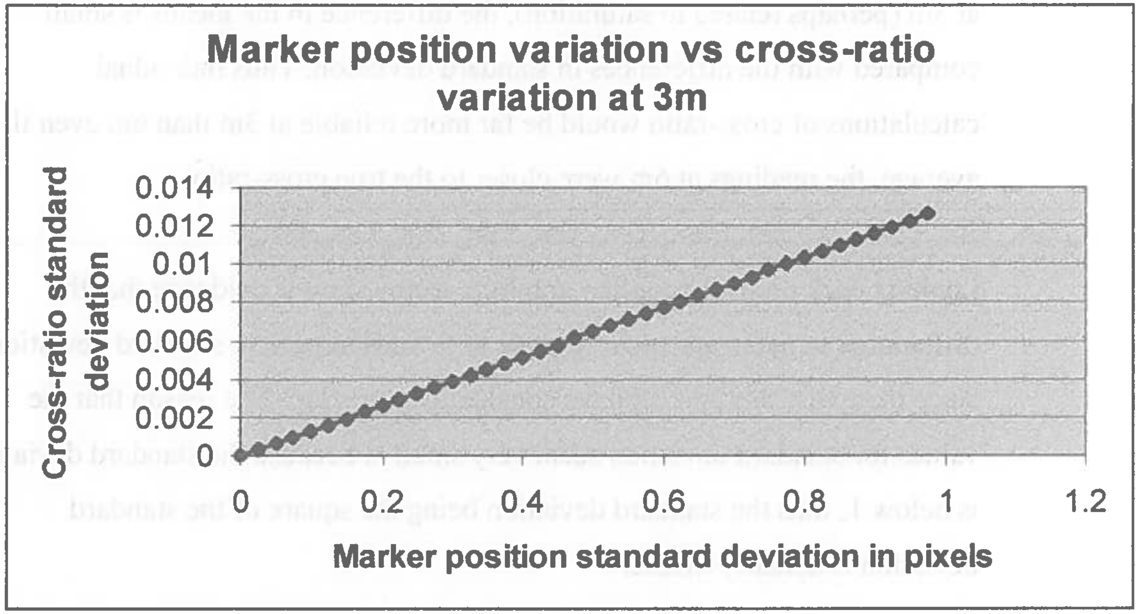
Upper bound for Std	0.0027047
Ellipse fitting at 3m	
Mean	0.0053579
Lower bound for mean	0.0045385
Upper bound for mean	0.0061774
Standard deviation	0.0032114
Lower bound for Std	0.0027503
Upper bound for Std	0.0037216
Binary centre of mass at 6m	
Mean	0.0059658
Lower bound for mean	0.0046448
Upper bound for mean	0.0072867
Standard deviation	0.0051768
Lower bound for Std	0.0044334
Upper bound for Std	0.0059992
Grey level centre of mass at 6m	
Mean	0.0038376
Lower bound for mean	0.0031870
Upper bound for mean	0.0044881
Standard deviation	0.0025937
Lower bound for Std	0.0021833
Upper bound for Std	0.0029544
Ellipse fitting at 6m	
Mean	0.0051632
Lower bound for mean	0.0041563
Upper bound for mean	0.0061701
Standard deviation	0.0039460
Lower bound for Std	0.0033793

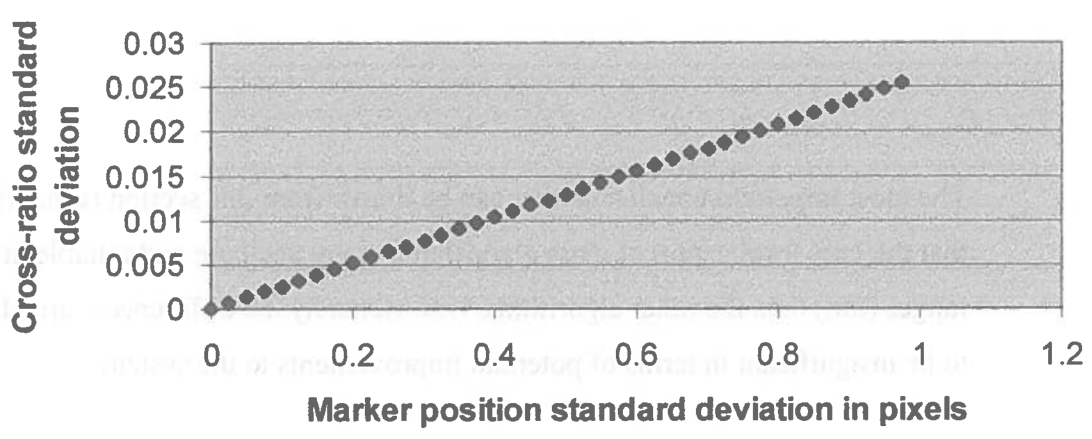
difference at this range is lower than the lower bounds for the other two. The grey-level centre of mass also has lower standard deviation at 6m in these tests, this indicates that it is more consistent at longer ranges than the other two.

A strange result here is that the mean difference between estimate and true cross-ratio is significantly lower at 6m than at 3m for the grey-level centre of mass algorithm. The algorithm is more accurate at longer ranges based on this measure. The opposite is true for the standard deviation, the algorithm has lower standard deviation at 3m than at 6m. A lot depends on how important differences in mean are compared with differences in standard deviation for this data. For instance it could be that, whilst the algorithm has a greater mean error at 3m (perhaps related to saturation), the difference in the means is small compared with the differences in standard deviation. Thus individual calculations of cross-ratio would be far more reliable at 3m than 6m even if, on average, the readings at 6m were closer to the true cross-ratio.

Looking back on the histogram graphs it seems there is evidence that the differences in mean are small relative to the differences in standard deviation since there distributions all have considerable overlap. The reason that the values for standard deviation seem very small is because the standard deviation is below 1; thus the standard deviation being the square of the standard deviation is actually smaller.

Although the differences in the standard deviation between the grey-level and binary image algorithms at 6m are statistically meaningful, this gives no indication as to how significant the differences are to functioning of the motion capture system. The important issue for the motion capture system is how accurately the marker positions are found, it is thus necessary to find the generated a number of cross-ratios from the marker positions obtained using a random number generator based on normal distributions of the appropriate parameters. For each value of standard deviation of the positions of the markers a corresponding value of standard deviation for the cross-ratios can be obtained. The results of this are displayed in two graphs. The graphs represent the standard deviation of the marker positions in terms of number of pixels in the image, thus the need for a graph for 3m and a graph for 6m. Changes in marker position in the image at 6m will have a greater effect on the cross-ratio than at 3m, because the bar is fewer pixels in length.





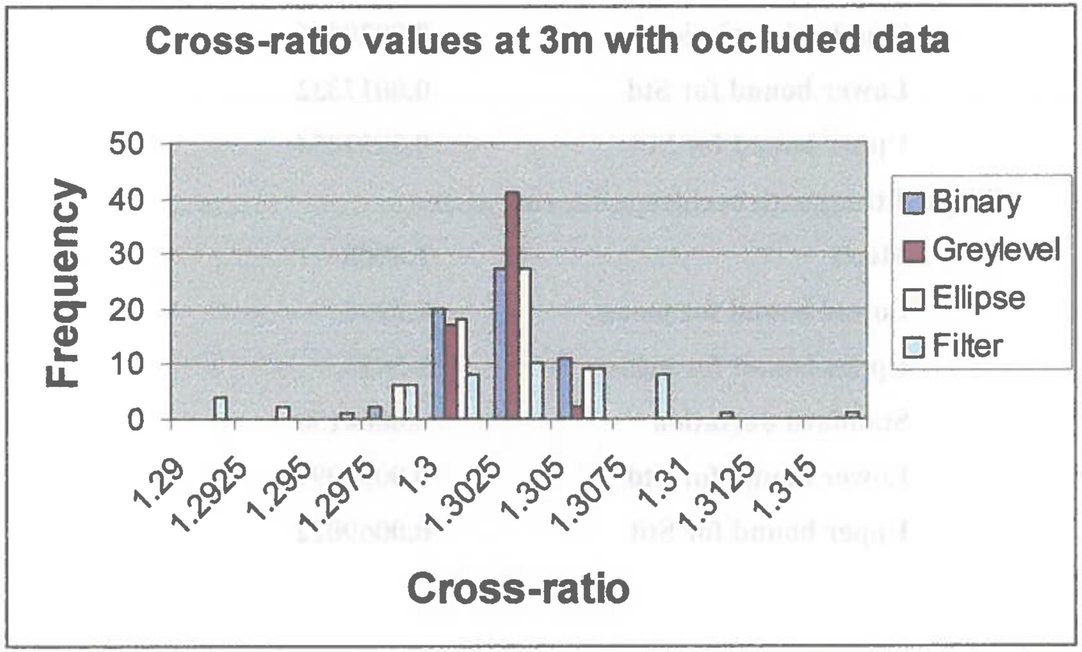
The graphs show that at variations in the marker position of standard deviation less than one pixel the relationship between cross-ratio standard deviation and marker position standard deviation is fairly linear. The results of the tests can be assessed in light of this information. The grey-level centre of mass algorithm at 6m has a standard deviation in cross-ratio of **0.00361.** This equates to a marker position standard deviation of about **0.14** pixels by extrapolating from the graph. This can be compared with the binary algorithm. The binary centre of mass algorithm has cross-ratio standard deviation of **0.00773**. This works out as a marker position standard deviation of about **0.3** pixels. Thus the difference between the binary and grey level centre of mass in terms of standard deviation of marker positions at 6m is about **0.16** pixels. Measuring the images, the cross-ratio bar traverses around **108** pixels across its length. Since the bar is 1m long there is approximately 1 pixel per centimetre. The difference in standard deviation between the two algorithms is thus equivalent to a real difference in the marker positions on the bar of about **1.5** millimetres. Looking at the binary and grey level.

The most important conclusion that can be drawn from this section is that there that the grey level centre of mass algorithm is more accurate and reliable at long ranges (6m) than the other algorithms. Unfortunately the differences turned out to be insignificant in terms of potential improvements to the system.

Analysis of performance on occluded data

This section deals with the analysis of the algorithm on the occluded data sets. The 3m occluded data shall be considered first, the reason for this being that it should be the easier set of data to work on. With the 3m data set the markers are larger in the image, hence there are more pixels with which to identify and correct for occlusion. The 3m images were processed with the system using all four algorithms for marker position estimation.

The results obtained were plotted in a histogram.



The means and standard deviations were calculated

Standard deviation	0.0016963
Lower bound for Std	0.0014652
Upper bound for Std	0.0019826
Grey level centre of mass at 3m	
Mean	1.3004
Lower bound for mean	1.3001
Upper bound for mean	1.3007
Standard deviation	0.0011327
Lower bound for Std	0.0009784
Upper bound for Std	0.0013239
Ellipse fitting at 3m	
Mean	1.3003
Lower bound for mean	1.2998
Upper bound for mean	1.3008
Standard deviation	0.0020066
Lower bound for Std	0.0017332
Upper bound for Std	0.0023454
Ellipse fitting with occlusion filtering at 3m	
Mean	1.3008
Lower bound for mean	1.2990
Upper bound for mean	1.3025
Standard deviation	0.0064138
Lower bound for Std	0.0050999
Upper bound for Std	0.0069012

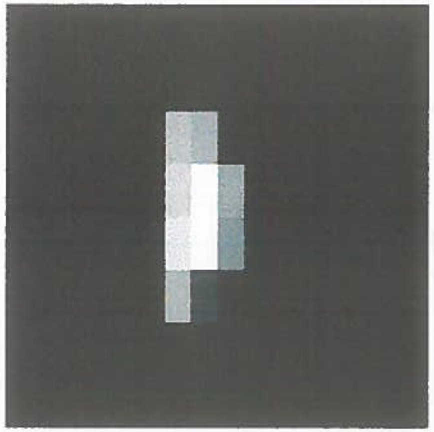
Also the differences of the cross-ratio values form the true cross-ratio.

Upper bound for Std	**0.0019826**
Grey level centre of mass at 3m	
Mean	**0.032903**
Lower bound for mean	**0.032616**
Upper bound for mean	**0.033189**
Standard deviation	**0.0011327**
Lower bound for Std	**0.0009784**
Upper bound for Std	**0.0013239**
Ellipse fitting at 3m	
Mean	**0.033056**
Lower bound for mean	**0.032549**
Upper bound for mean	**0.033564**
Standard deviation	**0.0020066**
Lower bound for Std	**0.0017332**
Upper bound for Std	**0.0023454**
Ellipse fitting with occlusion filtering at 3m	
Mean	**0.032576**
Lower bound for mean	**0.030816**
Upper bound for mean	**0.034336**
Standard deviation	**0.0064138**
Lower bound for Std	**0.0050999**
Upper bound for Std	**0.0069012**

All the algorithms appear to underestimate the cross-ratio by around **0.03**. This compares with the non-occluded data where the average estimate was far closer to the true value. The fact that the error in estimation seems to be fairly ordered makes sense since all the data images should be effected in roughly the same way by the changes

The grey-level centre of mass algorithm again seems to perform the best.

Looking at the occlusion-filtering algorithm in action, it seems that a false assumption is the cause of its poor performance. The idea behind the occlusion filtering was that occluded edges would be indicated by sudden drops in intensity in the image. This was justified by inspecting images containing occluded markers. Unfortunately, the images inspected happened to be images where the occluded edges were either directly vertical or horizontal. In these special cases the assumption hold true, however when the occlusion edge is at a diagonal this is not the case as can be seen from the images below: -



Even when the markers are occluded along a vertical or horizontal line, the algorithm does not function correctly. In these cases, although the occlusion filtering removes many of the points associated with occlusion edges, a number are still left behind after the process. When the ellipse-fitting algorithm is applied to this data it still matches the ellipse which passes through the occlusion edge since it is trying to minimise the error and these points will produce a large error if measured against the true ellipse of greater the effect of variation in individual points.

Summary

The overall conclusions that can be drawn from these experiments are disappointing. In terms of whether or not using grey-level information can provide greater accuracy for marker location than binary, neither of the algorithms used produced any significant difference. This does not mean that improvements cannot be made, but it certainly counts against any argument that they can.

The attempts to improve the accuracy of the system when dealing with occluded markers likewise met with failure, however, looking at the causes of the failure a possible method for dealing with occlusions became apparent.

Although the images of the markers did not contain the necessary information detect the occlusion edge, they were squashed in the direction of the occlusion. Therefore the angle of the occluded edge could be determined from an image of an occluded marker. The line going through the centre of mass of the marker at this angle would bisect the marker into two halves. The occlusion edge must clearly lie, in its entirety, on one side of this line. Therefore the line could act as the filter during the edge detection stage of the ellipse algorithm. All points found on one side of the ellipse could be ignored. The remaining points could be guaranteed not to be part of the occluded edge. The problem, of course, is that it is not known which side of the line the occlusion is on. There is no obvious solution to this problem at the image level, however it is possible to obtain two estimates of marker location from the algorithm. One estimate could be found assuming that the occlusion lay on one side of the line, the other assuming it lay on the other side. These two estimates could be passed on to the higher levels of the system which could, hopefully, make use of information from other cameras to decide which one was correct.

References

Machine Vision

Jain, Kasturi, Schunck

Robot Vision

Horn

Introductory Techniques for Computer Vision

Trucco, Verri

Elementary Statistics

Triola

Numerical Recipes in C

Press, Teukolsky, Vetterling, Flannery

Cvonline

A. Copy of web page from which algorithm for identifying connected components for labeling of markers was obtained.

B. Algorithm used for thresholding, taken from the book Machine vision

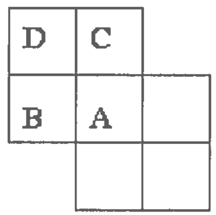
Subsections

- A sequential scan labelling algorithm

Identifying connected components

Given a binary image we wish to scan through it, identify distinct 'blobs' and label each one uniquely. Connectivity will be described using a left-skewed 6-connectedness neighbourhood scheme, as shown in figure 1.

Figure 1: The 6-connected neighbourhood.



A sequential scan labelling algorithm

We scan the image using a typical raster scan, row by row, top to bottom, left to right. Then, when we examine a particular cell *A*, we know that the cell to its left, *B*, has already been labelled, as has the cell *C* directly above *A*. Moreover, the cell *D* directly above *B* is also considered connected to *A* so its labelling must also be taken into account.

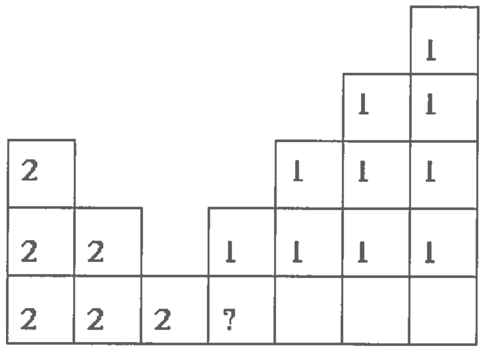
The sequential scan labelling algorithm is described as follows:

```
if A = 0          do nothing

else if D labeled
     copy label to A

else if (not B labeled) and (not C labeled)
     increment label numbering and label A
```

```
    else
        copy either B label or C label to A
        record equivalence of labels
```

After running this algorithm to label all the pixels, a second scan through the image is required to clean up the label equivalences, giving each connected component in the image a unique label.

Figure 2: Two regions previously thought to be disconnected are later discovered to be connected. Equivalent labels must be identified by a second scan.



Next | Up | Previous

Next: Mathematical Morphology **Up:** Computer Vision IT412 **Previous:** Lecture 3

Robyn Owens
10/29/1997

Taken from the book Machine Vision by Jain, Kasturi, Schunck

1. **Select an initial estimate of the threshold, T. A good initial value is the average intensity of the image.**
2. **Partition the image into two groups, R1 and R2, using the threshold T.**
3. **Calculate the mean gray values u1 and u2 of the partitions R1 and R2.**
4. **Select a new threshold:**

   $T = \frac{1}{2}(u1 + u2)$
5. **Repeat steps 2-4 until mean values u1 and u2 in successive iterations do not change.**