Master's thesis in Computer Vision

DETECTING AND TRACKING PLAYERS IN FOOTBALL USING STEREO VISION

Department of Electrical Engineering
Linköping University, Sweden.

by

JOHAN BORG

LiTH−ISY−EX−−07/3535−−SE
Linköping 2007

Master's thesis in Computer Vision

DETECTING AND TRACKING PLAYERS IN FOOTBALL USING STEREO VISION

Department of Electrical Engineering
Linköping University, Sweden.

by

JOHAN BORG

LiTH−ISY−EX−−07/3535−−SE
Linköping 2007

INSTITUTE OF TECHNOLOGY
LINKÖPING UNIVERSITY

Supervisor: **FOLKE ISAKSSON**
Examiner: **KLAS NORDBERG**

Linköping February 7, 2007.

Abstract

The objective of this thesis is to investigate if it is possible to use stereo vision to find and track the players and the ball during a football game.

The thesis shows that it is possible to detect all players that isn't too occluded by another player. Situations when a player is occluded by another player is solved by tracking the players from frame to frame.

The ball is also detected in most frames by looking for ball-like features. As with the players the ball is tracked from frame to frame so that when the ball is occluded, the positions is estimated by the tracker.

Keywords: Stereo vision, Football, disparity estimation, camera calibration.

Acknowledgment

I would like to take this opportunity to thank my advisor Folke Isaksson for all his contributions to this thesis. I would also like to thank Wilhelm Isoz for his opposition and review of this thesis. A special thanks goes to Robert Hernadi and all the people at Tracab AB and Hego AB who helped to put this system into practice.

Notation

This section describes special symbols, variables and common abbreviations used throughout this thesis.

Symbols

$\bar{x}$	Vectors.
$\hat{x}$	Identity vectors.
$\hat{\theta}$	Parameter vector.

Variables

s_x, s_y	Width and height of image plane.
s_u, s_v	Width and height of image.
$\mathrm{fov_s}$, $\mathrm{fov_h}$	Horizontal and vertical field of view.
f	Focal length.

Abbreviations

Baseline	The line between the left and the right camera in a stereo pair.
Epipolar line	The line connecting the centers of projection in two cameras.
Quadrature filter	A filter with no DC component and no support in the negative frequency domain.

Contents

Part I

Introduction

Chapter 1

Introduction

1.1 Background

This thesis has been performed at the department Sensor Systems at SAAB Bofors Dynamics in Linköping, Sweden. The thesis is the last year's work for a master's degree in Computer Science and Engineering at Linköping University, Sweden.

In the late 2003, a former board member of the Swedish football association, Robert Hernadi, had a vision of an image processing system to track football players with the goal to detect offside situations. Robert H. together with a Swedish broadcasting company, Hego AB, contacted Saab Bofors Dynamics AB (SBD) with a wish of an evaluation of such a system. From negotiations between SBD and Robert H./Hego AB it was decided that a thesis should be written in the subject.

1.2 Objective

The objective of this thesis is to investigate if it is possible to implement a system for tracking all the players and the ball on a football pitch. Since the pitch is so large, several cameras have to be used in a real system. In this thesis a limitation is set to only cover a part of the pitch. The cameras that will be used is standard digital CCD cameras. During calculations every other line in the images will be ignored due to the interlace.

1.2.1 Goal

Three reference scenes was selected. These scenes was to be used to show that the system was able to track the players and the ball in this configuration.

The first scene contains two players running with the ball towards the goal. This scene is quite simple and contains no occlusion. The purpose of the first scene is to show that the players can be tracked as well as to evaluate the tracking of the ball.

The second scene also contains two players, the players are occluded during some frames. The purpose of the second scene is to show that the players can be tracked during occlusion.

The third scene is a model scene where the position of all the players is known (with some degree of correctness). The purpose of the third scene is to show the error in position measurement.

The following goals has been defined for this thesis in terms with the commission.

- Find a working camera setup that can cover the entire pitch.
- Locate and track the players in the reference scenes.
- Locate and track the ball in the refrence scenes.

Limitations

The following limitations has been set for this thesis:

- The cameras only covers a part of the pitch (since only two cameras was available for testing).

1.3 Applications

There are a huge number of applications for a system like this. The intention is as said, that the system should be used by broadcast companies to be able to automatically monitor a football game. However the system is much more usable than that. It can for example be a great tool for the coaches for post-game analysis, or to gather information for game statistics. Since every player can be tracked, all sorts of information can be collected, total running distance, maximum speed, average speed, the time a player has the ball etc. The possibilities are almost endless.

1.4 Similar work

Systems that aim to solve the same problem as this thesis has recently started to appear on the market, some of these system uses transponders attached to the players to triangulate the position of each player (and ball) during the game. These systems have proved to be very expensive and requires that antennas are installed around the arena. A German company called Cairos has developed just such a system.

In 2003 Daniel Setterwall (KTH) did a masters thesis with roughly the same goal as this thesis ([9]). The goal for Setterwall in his thesis was to find a method to use computer vision to find and track football players. The solution for tracking the players here was to use template tracking (the tracking is done by searching for each player with a template or model of that particular player). This methods biggest problem is occlusion. Since the cameras are often placed quite low, players tend to occlude each other very often.

There are a number of commercial companies that are offering football player tracking using computer vision. Two of the biggest companies is ProZone ([8]) and Sport Universal ([11]). It is not clear exactly how the tracking is done in these systems. ProZone uses several cameras placed around the pitch and delivers statistics of ball posession, pass maps and distance for each player and much more. The information is not however in real time. Sport Universals system is very similar to ProZone but delivers statistics in real time. Both these systems have in common that they are very expensive thus they are targeting the major leagues and larger football clubs.

1.5 Document overview

The thesis is divided into 4 parts. Part 1 is the introduction to this thesis. Part 2 describes the basic theory that this thesis is based on. Part 3 describes the implementation and the decisions made to realize the system. Part 4 contains an evaluation which describes the performance of the system and at last a summary.

Part II

Theory

Chapter 2

The camera model

In order to measure distances between objects in a image you need a model of the camera that accurately mimics the physical properties of the real camera. Several types of camera models exists, two of them are described in this section. Figure 2.1 shows the coordinate systems that are of interest for these camera models.

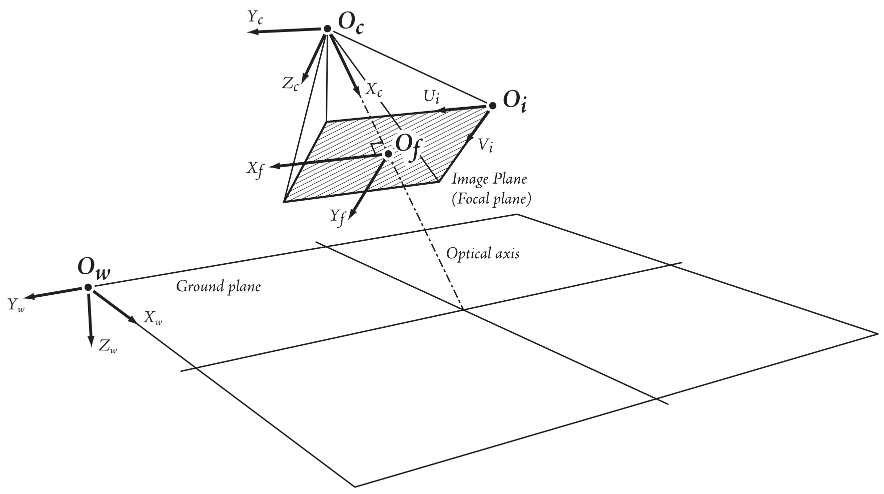


Figure 2.1. Camera model

- $\mathcal{O}_w$, World coordinates $[x_w, y_w, z_w]$.
- $\mathcal{O}_c$, Camera coordinates $[x_c, y_c, z_c]$.
- $\mathcal{O}_f$, Image plane coordinates (Also known as *focal plane*) $[x_f, y_f]$.
- $\mathcal{O}_i$, Image coordinates $[u_i, v_i]$.

The *world coordinates* is the most basic coordinate system, all other system will be defined in relation to this one. The world coordinate system is situated somewhere on the football pitch which causes the pitch to have zero height ($z = 0$), this is convenient when selecting points for calibration as will be described in a later chapter.

Camera coordinates is unique for every camera (since each camera has a unique orientation), it is described in relation to world coordinate system.

Image plane coordinates is also unique for every camera, it is essentially the same coordinate system as camera coordinates but in two dimensions. The scaling of the image plane coordinates is always the same as for the camera coordinates.

At last there is *Image coordinates*. Image coordinates is, like image plane coordinates, two-dimensional. Image coordinates is essentially the same coordinate system as image plane coordinate system with the exception of different scaling and translation.

2.1 Parameters

A camera model is defined by a set of parameters that is used to model the camera. The parameters is usually divided into two categories, intrinsic and extrinsic parameters.

2.1.1 Intrinsic parameters

Intrinsic parameters are those that controls the physical properties of the camera. These are usually one or more of the following (others exists as well).

- Focal length.
- Image plane size.
- Lens distortion, measured as a radial distortion.
- Image plane displacement, measured as displacement from the optical axis.

2.1.2 Extrinsic parameters

Extrinsic parameters are those that controls the physical location of the camera. These can also be represented in many ways, but one of the more simple ways is to use a location vector and Euler angles. The parameters is given in world coordinates.

- Camera rotation.
- Camera location.

2.2 Pinhole camera

The pinhole camera is one of the most basic camera models. This camera model is often used in computer graphics since it is so simple to use. The pinhole camera does not use sophisticated intrinsic camera parameters such as lens distortion and image plane offset.

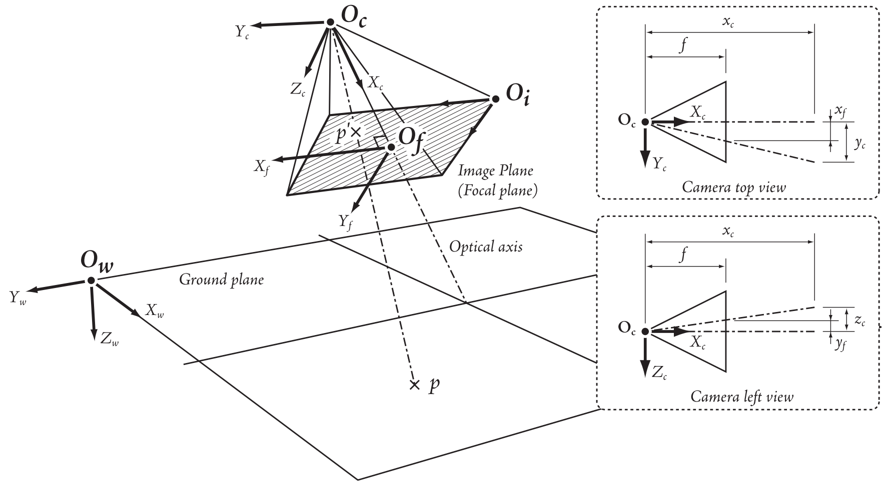


Figure 2.2. Model of the pinhole camera

The geometry of the camera model gives the following basic relation between a point in camera coordinates, $\hat{p} = (x_c, y_c, z_c)^T$, and the image plane coordinates, $\hat{p}' = (x_f, y_f)^T$

$$x_f = \frac{f \cdot y_c}{x_c} \tag{2.1}$$

$$y_f = \frac{f \cdot z_c}{x_c} \tag{2.2}$$

This gives the common pinhole camera equation, which describes the projection of a point in camera coordinates to the image plane.

2.3 Lens distortion

Real images usually suffer from lens distortion effects. This is especially noticeable when a lens with a short focal length is used. The effect is often referred as the fisheye effect. The picture below shows how a image of a grid is distorted by the lens.

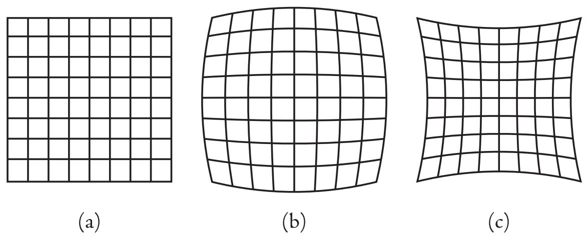


Figure 2.3. Different types of lens distortion. (a) No distortion. (b) Barrel distortion, common on regular cameras. (c) Pillow distortion.

The pinhole camera model does not model this effect. Thus some modifications has to be done to the model. The parameter for lens distortion in the intrinsic parameters is used to describe this effect. Lens distortion is usually modeled as a radial displacement, that is, a displacement that only depends on the distance from the image centre. This radial distortion can have many mathematical definitions, many researchers ([13] for example) use a polynomial function to model the displacement. The position in the image $[r, \theta]$ acquired by the pinhole camera model, is corrected by the radial function to $[r', \theta']$ according to equation 2.3 and 2.4

$$r' = r \sum_{i=0}^{N} k_i r^{i+1} \tag{2.3}$$

$$\theta' = \theta \tag{2.4}$$

2.4 Other distortion factors

During the manufacturing of the cameras it is almost impossible to exactly line the image plane (the CCD chip) with the centre of the camera. This causes the centre of the image plane to be out of alignment. To counter this effect two more intrinsic parameters are introduced, the offset of the image plane (c_x and c_y).

$$x'' = c_x + x' \tag{2.5}$$

$$y'' = c_y + y' \tag{2.6}$$

Where x' and y' are the coordinates after correction for radial distortion. The local coordinate system is shifted according to the offset giving the final corrected coordinates, x'' and y''.

2.5 The complete camera model

The camera model used in this thesis uses a combination of the pinhole camera model and the lens distortion model. This gives a quite accurate model of a real camera.

Both the pinhole model and the lens distortion can be combined which leads to the following relation between a camera coordinate (x_c, y_c, z_c) and the resulting image coordinate (x_i, y_i).

$$x_0 = \frac{fy_c}{x_c} \qquad y_0 = \frac{fz_c}{x_c} \tag{2.7}$$

$$r_0 = \sqrt{x_0^2 + y_0^2} \qquad \theta_0 = \tan^{-1}(y_0/x_0) \tag{2.8}$$

$$r_1 = r_0 + k_2' r_0^3 + k_3' r_0^4 \qquad \theta_1 = \theta_0 \tag{2.9}$$

$$x_2 = c_x' + r_1 \cdot cos(\theta_1) \qquad y_2 = c_y' + r_1 \cdot sin(\theta_1) \tag{2.10}$$

$$x_i = x_2 \cdot \frac{w_i/2}{s_x/2} + w_i/2 \qquad y_i = y_2 \cdot \frac{h_i/2}{s_y/2} + h_i/2 \tag{2.11}$$

In this thesis these equations is rewritten with the following substitutions.

$$\frac{s_x}{2f} = \tan(fov_s/2) \qquad \frac{s_x}{2f} = \tan(fov_s/2) \tag{2.12}$$

$$k_2 = k_2' f^2 \qquad k_3 = k_3' f^3 \tag{2.13}$$

$$c_x = c_x'/f \qquad c_y = c_y'/f \tag{2.14}$$

Redefining the camera parameters produces the final model as defined below. The reason for these substitutions is that the camera model is then valid even if the images are rescaled.

$$x_0 = \frac{y_c}{x_c} \qquad y_0 = \frac{z_c}{x_c} \tag{2.15}$$

$$r_0 = \sqrt{x_0^2 + y_0^2} \qquad \theta_0 = \tan^{-1}(y_0/x_0) \tag{2.16}$$

$$r_1 = r_0 + k_2 r_0^3 + k_3 r_0^4 \qquad \theta_1 = \theta_0 \tag{2.17}$$

$$x_2 = c_x + r_1 \cdot cos(\theta_1) \qquad y_2 = c_y + r_1 \cdot sin(\theta_1) \tag{2.18}$$

$$x_i = x_2 \cdot \frac{w_i/2}{\tan(\text{fov}_s/2)} + w_i/2 \qquad y_i = y_2 \cdot \frac{h_i/2}{\tan(\text{fov}_h/2)} + h_i/2 \tag{2.19}$$

The polynom used in the lens distortion is here defined as $r_1 = r_0 + k_2 r_0^3 + k_3 r_0^4$. The reason for choosing this definition is that it has been used previously in other projects at *SAAB Bofors Dynamics* so much of the code for implementation was already done. Experience with the older projects also shows that this function models the camera very accurately.

Chapter 3

Stereo vision

Stereo vision is a technique to calculate a depth image from two or more camera views. Stereo vision has several applications and one of the more obvious is to be able to get an object's distance from the camera.

Most stereo algorithms rely on specific rules that define how the cameras have been mounted, this is usually called *canonical configuration* and is defined in the next section.

3.1 Stereo geometry

A stereo geometry is formed by placing two pin-hole cameras with the same focal length in a so called *canonical configuration*. This means that the two cameras are shifted sideways by a distance called *base distance*. The camera image planes and optical axes are parallel.

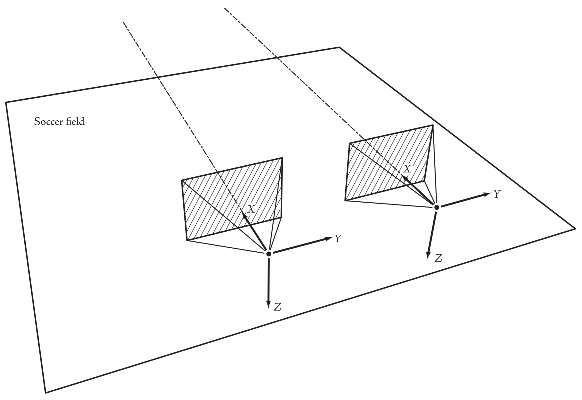


Figure 3.1. Two cameras in a canonical configuration.

If a point in space is considered at the coordinate $\overline{p} = (X, Y, Z)^T$. The following relation comes directly from the geometry (b is the camera base, the distance between the two cameras).

$$\frac{b/2 + X}{Z} = \frac{x_l}{f} \tag{3.1}$$

$$\frac{b/2 - X}{Z} = \frac{x_r}{f} \tag{3.2}$$

Adding these equations together and defining $d = x_l - x_r$ produces the following equation.

$$Z = \frac{bf}{x_l - x_r} = \frac{bf}{d} \tag{3.3}$$

Thus if it is possible to detect how much one pixel in the left image has moved sideways to the right image, it's simple to calculate the depth of the corresponding point.

There are however a few problems with this definition. First, according to the definition, both image planes must be parallel. This is usually not practical nor possible to achieve. Secondly, the definition builds on the pinhole camera model, since a real camera is subjected to lens distortion and centre of projection displacement this will not be accurate. One solution to both these problems is called image rectification.

3.2 Image rectification

As mentioned earlier, image rectification is a process to transform the two images in a stereo pair so that the epipolar lines becomes collinear and parallel to the horizontal image axis (see figure 3.2).

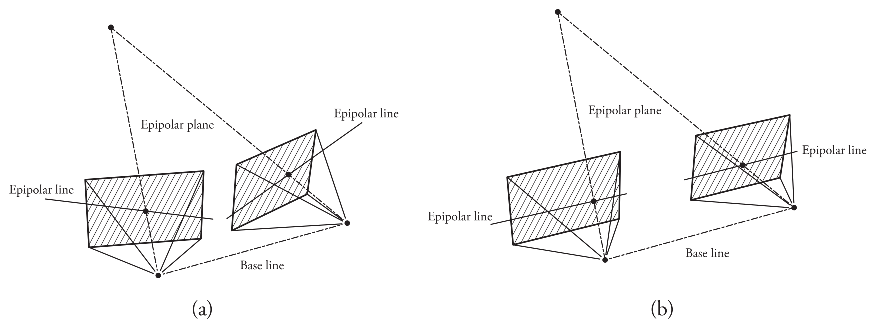


Figure 3.2. a) Non-collinear epipolar geometry. b) Collinear epipolar geometry.

These images can be thought of as being acquired by a new stereo rig with pinhole cameras that are already parallel as defined in the previous section. This rectification process also deals with the fact that lens distortion is present.

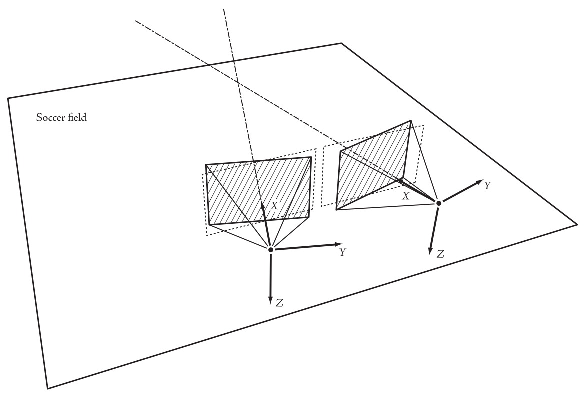


Figure 3.3. Two cameras in a more common configuration.

Many researchers ([5] and [7] for example) use the image rectification process as figure 3.3 shows. A rotational matrix is created that rotates the image planes so that they become collinear. From this matrix a warp function is produced that transform the old image plane to the new one. The resulting rectified image is similar to the original image but a little distorted. This method does not however deal with lens distortion.

3.2.1 Ground plane rectification

For this thesis the new image planes are not chosen in this way. As figure 3.4 shows, the image planes are chosen to be the same for both left and right camera and that it is located in the ground plane. There are two reasons for chosing this image plane. The first reason is simply that the transformation from the original camera view to the new image plane is very simple and can be performed very fast. The second reason is that the disparity becomes very limited for objects that are of interest. Objects that are located on the ground plane has no disparity, objects that are above the ground plane gets more disparity the higher the object is in relation to the ground plane. Since the players on the pitch always is quite near ($<$ 2.5m above) the ground plane, the disparity is very limited.

The new image plane is also rotated around the z-axis to align the x-axis with the camera base. This ensures that the disparity only will occur along the image u-axis.

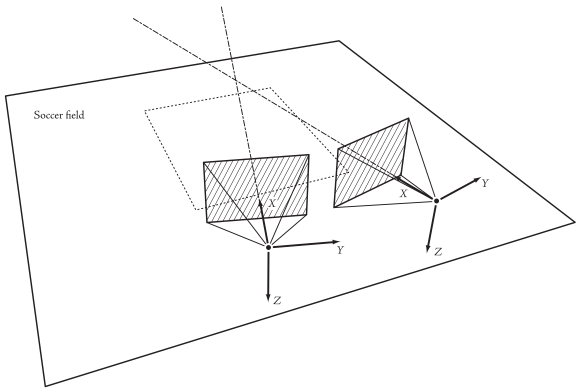


Figure 3.4. Rectification as done in this thesis.

The resulting rectified images for the left and right camera are shown in figure 3.5. Note that the lines that lie in the ground plane are identical in both images. The images also shows that the goal which does not lie in the ground plane is shifted sideways between the two images.

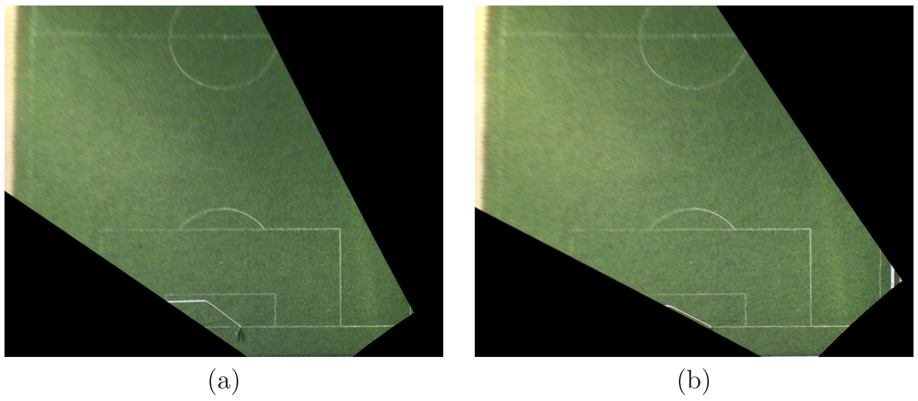


Figure 3.5. (a) Left rectified camera view. (b) Right rectified camera view.

3.3 Disparity estimation

There exists many ways to estimate depth from stereo images, most of them depend on the restrictions defined in section 3.1.

Correlation based stereo is probably the most simple way of calculating disparity, it is therefore also the most widely used method. As the name foretells correlation based stereo uses correlation to estimate how much each pixel has moved from one image to the other. The most intuitive way of implementing this is by using area correlation (see [10], [14] and [15]), this method is however extremely slow in comparison to other available methods.

An approach using local polynomial expansion was developed by [4]. This method expands every N×N region in both images with a polynom of degree 2. The disparity is then estimated by calculating the change in the polynomial coefficients. This method is faster than the correlation based method, but the result was not as good (on our test scenes, this is not however the general case).

Phase based disparity is a technique based on the Fourier shift theorem, which basically says that a small shift in spatial domain results in a relative small change in the signals phase. The phase of the signal is estimated using a quadrature filter. More information about different quadrature filters can be found in [12] which also has an extensive view over how phase based stereo works. Phase based disparity with a Non-ringing quadrature filter (defined in [12]) will be used in this thesis for all disparity estimations.There are several different quadrature filters to chose from but the Non-ringing filter is good since it has small spatial support, thus leads to fast calculations.

3.3.1 Phase based disparity estimation

Phase based disparity estimation is a method to calculate the shift between two stereo pair (that are rectified).

The method is based on the fact that a sideways spatial shift in the image results in a relative shift in the local phase. Local phase describes local edge symmetry independent of absolute gray value. By calculating the local phase in both stereo pairs the difference in local phase at each point results in an estimation of the disparity between the two pairs.

The method can not estimate a disparity that is too large. Therefore it is common to do the calculations in a resolution pyramid. This means that the estimations is first done in the coarsest resolution, the result is then used to shift the input the the next resoltion level thus reduces the disparity with each level. The method is defined in detail in [12].

Part III

Implementation

Chapter 4

Camera calibration

In order to perform any measurements with a camera, all parameters in the camera model has to be known. Camera calibration is a procedure to extract these parameters. This procedure is performed for all cameras and all lenses because the model can (and usualy do) vary from camera to camera and lens to lens. The parameters are divided into intrinsic and extrinsic parameters as mentioned earlier, all of which have to be estimated.

The intrinsic parameters that the model contains are:

- Field of view (vertically and horizontally, fov_h and fov_s).
- Lens distortion coefficients (k_2 and k_3).
- Image plane offset (vertical and horisontal offset of the CCD sensor, c_y and c_x).

The extrinsic parameters are:

- Camera rotation (ϕ_{yaw}, ϕ_{pitch}, ϕ_{roll}).
- Camera location (x_cam, y_cam, z_cam).

As described in the first section the calibration procedure is separated into two procedures, offline and online calibration. Offline calibration is performed once for every camera (usually in a lab) while online calibration is performed continously while the system is running.

4.1 Offline calibration

The offline calibration is used to calibrate parameters in the camera that won't change while the camera is in operation. These parameters are field of view, lens distortion and image plane offset. Since these parameters just have to be calibrated once it is possible to use a quite extensive procedure for them.

At SAAB an inhouse tool has been developed that is used to create panorama images. This tool can also be used to perform offline calibration. The calibration starts with an operator taking several pictures while rotating the camera and holding the camera focal point in the center at all times (see figure 4.1 and 4.2).

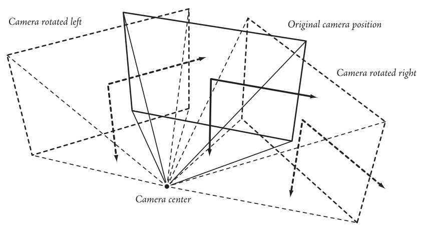


Figure 4.1. Three camera positions, each rotated around the local z axis.

The goal is now to estimate the intrinsic parameters as well as the rotation about the z-axis (the rest are not relevant). The parameters are initialized to a rough estimate set by the operator.

For every adjecent pair of images the program warps one image onto the other using the current camera model.

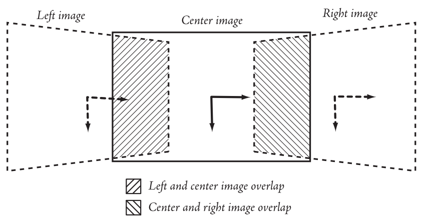


Figure 4.2. The overlap between three images.

If the camera parameters were perfect, the common part of the two images would be the same. Since this is usually not the case the two images will be spatially shifted and differently distorted. The total absolute differance between the two images is a good estimate of how correct the parameters are. The common area is divided into several sub areas (3x3 for example). Using area-correlation the program calculates how much each of these sub areas have to move in order to minimize the total absolute differance. The program tries to minimize the distance that the sub areas have to move, and if they don't have to move at all, the parameters are correct.

When taking the pictures the operator takes a full 360° panorama view. Since the images are taken in all directions, the horizontal field of view is uniquely defined and is estimated with very high precision.

At last an image taken from a view rotated 90° about the cameras x-axis is included. This causes the width-to-height ratio to be uniquely defined.

When the calculations are done the intrinsic parameters should be very accurate. The rotation about the z-axis is ignored since it's only valid in the panorama setup.

4.2 Online calibration

The offline calibration estimated the intrinsic parameters. When the cameras have been placed around the football pitch the extrinsic parameters (location and rotation) have to be estimated as well. The trivial solution would be to physically measure these parameters. To do this with sufficient accuracy is impractical. Also, when the system is in operation these parameters could change due to movement in the camera rig (caused by the wind or other external force). The online calibration is performed in three steps. Step 1 is used to calculate a rough estimate of the extrinsic parameters. Step 2 is used to improve the estimation of each camera model. Step 3 is used to improve the relationship between the left and right camera model.

4.2.1 Step 1

The online calibration starts with an operator selecting a few points in the left and right image that have a known coordinate in the real world. These poins should be placed where the image has an intrinsic dimensionality higher than one (all points must not lie in a straight line), thus where lines intersect is a good place to select points. This is done once during the startup of the system. The points selected is shown in figure 4.3.

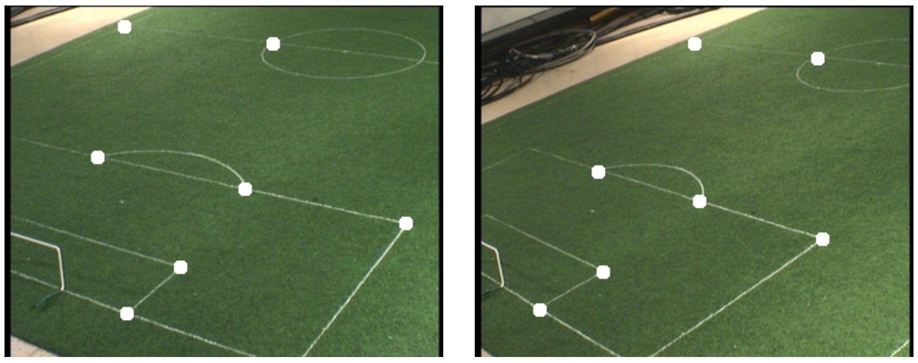

(a) (b)

Figure 4.3. (a) Football pitch from left camera view. (b) Football pitch from right camera view.

The world coordinates are projected to the image plane, the distance between the projected point and the selected points is minimized by adjusting the camera parameters. The parameters can in this way be estimated.

4.2.2 Step 2

A problem with step 1 is that the operator manually selects the points in the image. It's not possible to select these points with a better precision than ±1 pixels so an error is introduced to the estimation. Step 2 is used to improve the calibration by comparing the left and right image with a model of the pitch.

The model of the pitch is created from a plan-view image of the pitch. This image is warped to the left and right image plane producing the images seen in figure 4.4.

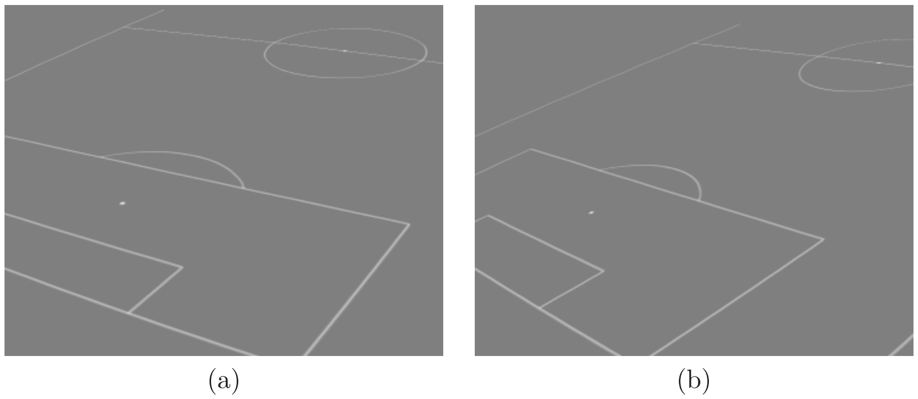


Figure 4.4. (a) Model of football pitch transformed to left camera view. (b) Model of football pitch transformed to right camera.

The lines in the pitch model can be compared with the lines in the original image, thus calculating the error in the model. For each corner or line intersection area correlation is used to calcualate how much difference there is between the model and the real image. This difference is minimized by adjusting the camera parameters using a multidimensional Newton-Rahpson search scheme.

4.2.3 Step 3

As a last adjustment on the camera models, step 3 is used to improve the joint calibration by comparing the left and right image. The left and right image contains many features that occur in both images (lines etc). By warping the rectified right image to the original left image plane figure 4.5 is produced. As seen this procedure creates an image quite similar to the left image. The difference between the two images is that features that do not lie in the ground plane are shifted sideways (note the goal).

By comparing features on the ground (with intrinsic dimensionallity higher than one) it is possible to calculate the relative error in the models. Figure 4.6a shows the absolute difference between the two figures in figure 4.5. If the two models were perfect the features in the ground would be completeley grey, this is not the case in figure 4.6a.

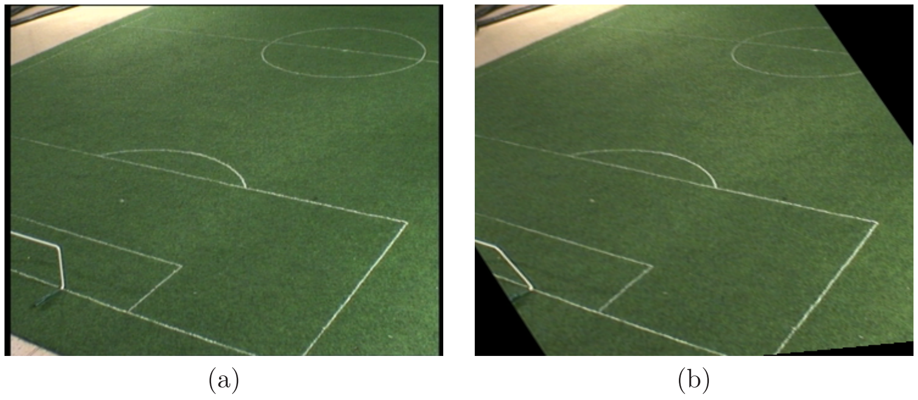


Figure 4.5. (a) Left camera view. (b) Rectefied right camera view warped to left.

Again using area correlation as in step 2 it is possible to adjust the left camera parameters so that the error is minimized. Since only the left camera is adjusted this procedure does not correct the absolute location and orientation of the camera. But rather the relation between the two camera models. Figure 4.6b shows the difference image after this calculation. The features in the ground are now much less apperant.

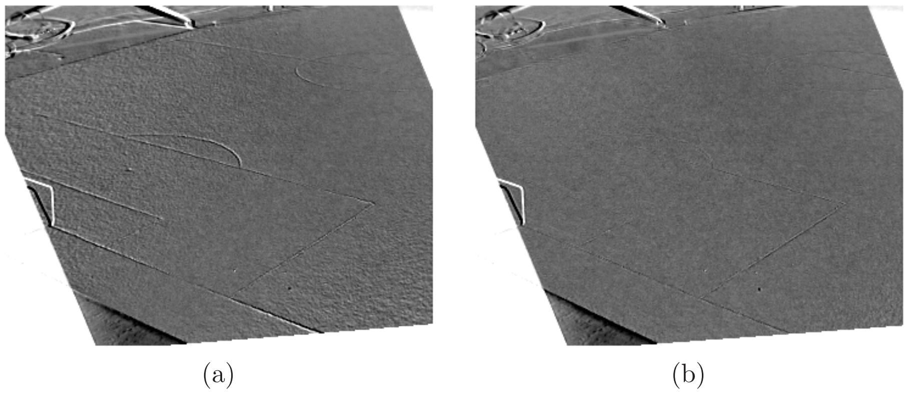


Figure 4.6. (a) Difference between left and the right projected image after the first calibration step. (b) Difference after the second calibration step.

Chapter 5

Camera setup

If the system should cover the entire football pitch, several rigs (a rig consists of two cameras) must be used.

5.1 The pitch

During the first stages of the thesis there where no real football images to work on. Because of this a scale model of a football pitch was built. The scale of the model is 1 : 35 and the dimensions have been chosen to comply with international specifications ([3]).

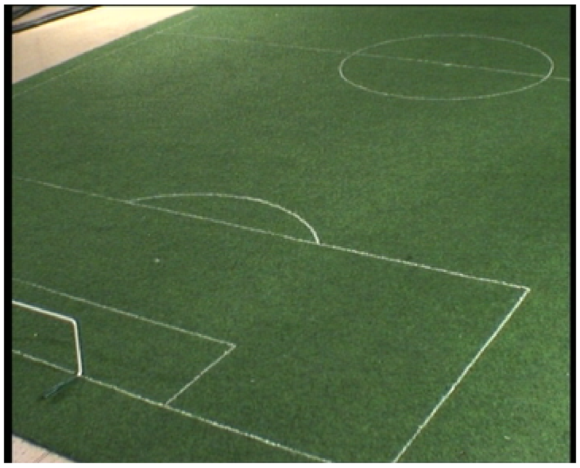


Figure 5.1. A photo of the scale model.

5.2 Possible setups

There is a huge number of possible ways to place the camera rigs, all depending on how many rigs that are used, what focal length that are chosen and how much overlap that is desirable.
There are numerous factors to take into the account when deciding these questions. Below follows a few of these factors that I found most important.

1. How many cameras can be used?
2. Should the player be covered from more than one direction? (Should each area of the pitch be covered by more than one stereo pair).
3. Which area should each camera rig cover?
4. Where (most important, how far away from the pitch) can the cameras be placed?
5. What pixel resolution must a player have at a certain distance?
6. What range resolution must a player have?

Two examples of how four cameras can be placed to cover the football pitch is shown in figure 5.2.

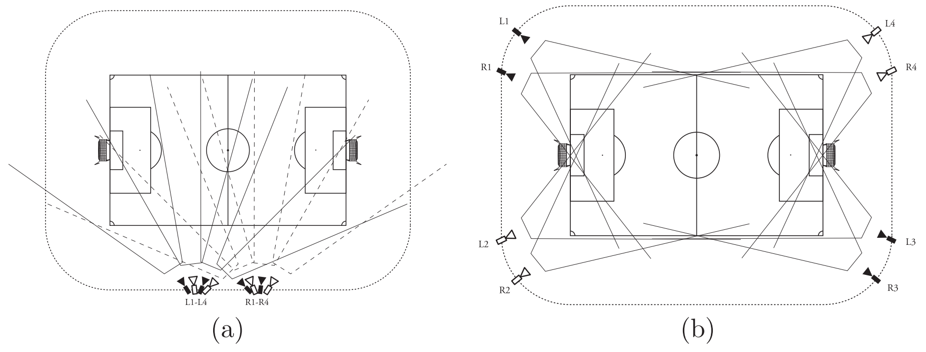


Figure 5.2. Two possible camera setups with 8 cameras.

5.2.1 Coverage (point 1-4)

Deciding on how many cameras that can be used is mainly an economical question. The cameras themselves are not so expensive (between \$400 – \$1000). But all the things around the cameras, cable's, rigs etc. significantly increase the price. Tests have shown that it's very difficult to use less than 8 cameras, otherwise the resolution would be very poor since the cameras have to be equipped with very wide-angle lenses.
Covering the player from more than one direction would imply to double the amount of cameras. If this is economically justified then this would be a great way to increase the performance of the system. However, this question is somewhat out of the scope for this thesis.

Figure 5.2 shows two ways of covering the pitch with 8 cameras.

Figure 5.2a places all the cameras on one side of the pitch and divides the pitch into four pieces.

Figure 5.2b shows another way of cover the pitch, this time from two different corners. When covering the pitch in this manner the players is covered from two directions thus improving the estimation of the position in this area. This way requires a wider FOV thus reducing the resolution in the middle of the pitch.

5.2.2 Placement (point 5)

The placement of the cameras is crucial to the choice of camera lens. If the cameras is placed far away from the pitch ($> 50m$) a lens with large focal length can be used. If the cameras must be a lot closer a lens with shorter focal length has to be chosen. For this thesis it has been assumed that the cameras can be placed $30m$ from the pitch, at a height if about $25m$. This location was chosen after studying larger arenas in Europe.

5.2.3 Resolution (point 6-7)

By the resolution of the player, two things are referred to. First there is the actual *pixel resolution*. This means how tall (in image pixels) a player is at a specific distance. The second resolution measure is *depth resolution*. Depth resolution is a measure of how well the distance between the object (the player) and the camera can be estimated. This depends mostly on the camera base, focal length and the distance from the camera.

Pixel resolution

Pixel resolution is the size of an object in the image (in pixels). The pixel resolution for objects at various distances and sizes is shown in figure 5.4.

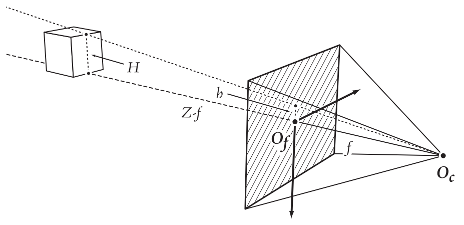


Figure 5.3. An objects height in the image in relation to the size in the real world.

The geometry of figure 5.3 gives the following basic relation.

$$\frac{H}{Z} = \frac{h}{f} \rightarrow h = \frac{fH}{Z} \tag{5.1}$$

The pixel resolution, here referred to as v, is then defined as below.

$$v = \frac{fH}{Z} \cdot \frac{s_v}{s_y} = \frac{fH}{Z} \cdot \frac{s_v}{2f \cdot \tan(\text{fov}_\text{h}/2)} = \frac{Hs_v}{2Z \cdot \tan(\text{fov}_\text{h}/2)} \tag{5.2}$$

A plot of this resolution measure with respect to the distance from the camera is shown in figure 5.4.

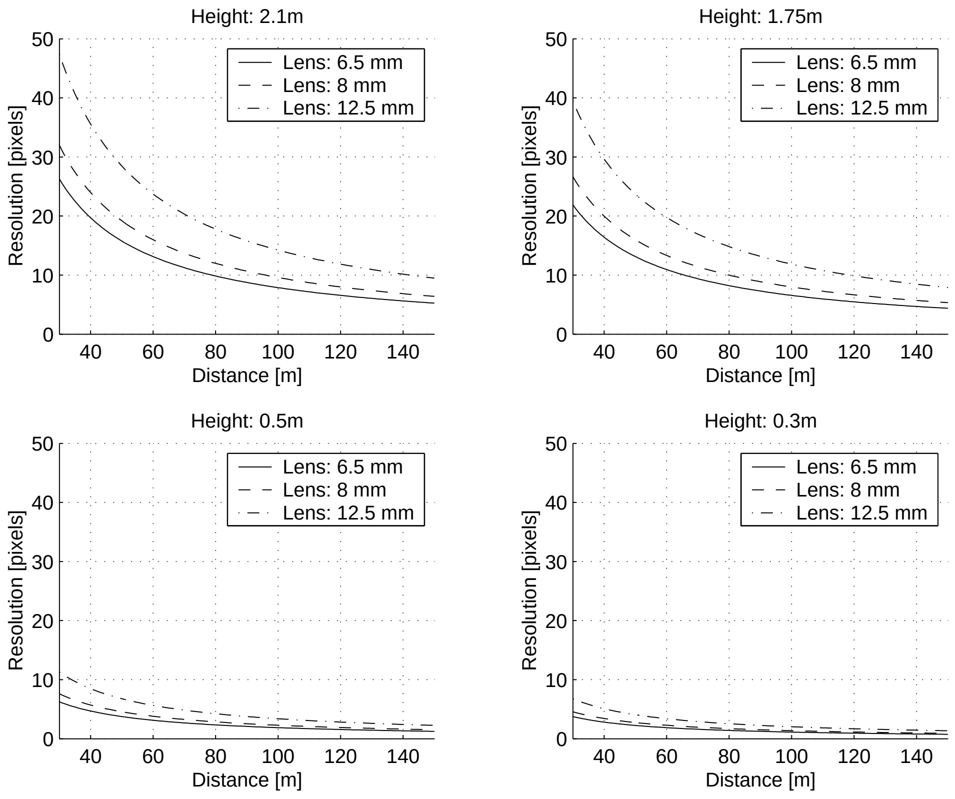


Figure 5.4. The size of an object in pixels with respect to the distance from the camera.

Depth resolution

Depth resolution is a measure of how well the distance between an object in the scene and the camera is estimated. The depth resolution is defined as the difference in range when the stereo algorithm fails it's search with ± 0.5 pixels. This is the best estimate that can be calculated without sub-pixel accuracy.

From the stereo geometry the following basic relation can be found.

$$x_1 = \frac{-fb}{y_1} \tag{5.3}$$

$$x_2 = \frac{-fb}{y_2} \tag{5.4}$$

Depth resolution is here defined as $\triangle x = x_2 - x_1$. The error made by the stereo estimation is a positive shift of y_1 and is defined as $\triangle y = \frac{s_x}{s_u}$ (a shift of one pixel).

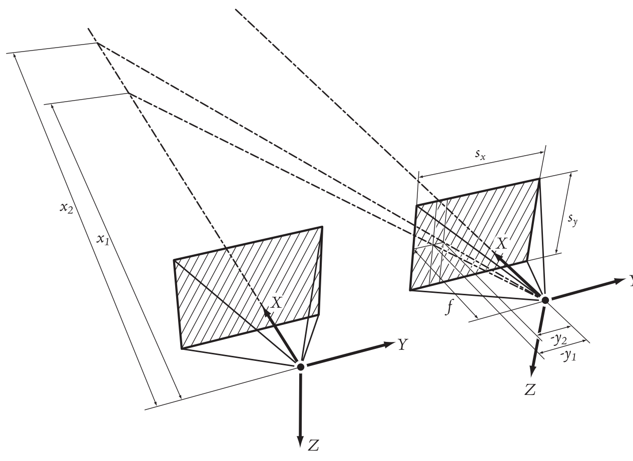


Figure 5.5. Two cameras in an epipolar configuration (stereo geometry).

$$\begin{aligned}
\triangle x &= x_2 - x_1 = \frac{-fb}{y_2} - x_1 = \frac{-fb}{y_1 + \triangle y} - x_1 = \\
&= \frac{-fb}{\left(\frac{-fb}{x_1}\right) + \triangle y} - x_1 = \\
&= x_1 \cdot \left(\frac{fb}{fb - x_1 \triangle y} - 1 \right) = \frac{\triangle y \cdot x_1^2}{fb - x_1 \triangle y} = \\
&= \frac{\frac{s_x}{s_u} \cdot x_1^2}{fb - x_1 \frac{s_x}{s_u}} = \frac{x_1^2}{\frac{bs_u}{2\tan(\mathrm{fov_s}/2)} - x_1} \qquad (5.5)
\end{aligned}$$

A plot of this resolution measure with respect to the distance from the camera is shown in figure 5.6.

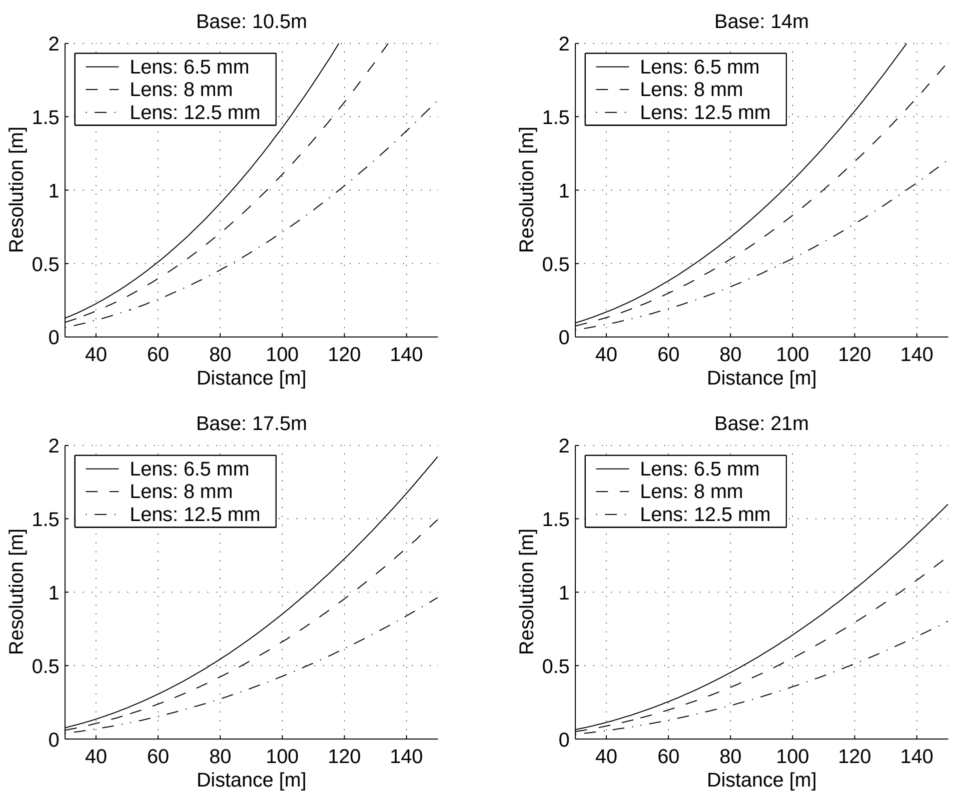


Figure 5.6. Error with respect to distance from the camera.

5.3 Final setup

During my work with this thesis it has become clear that it isn't possible to select a "best" camera setup. It all depends more or less on the following questions.

- How large is the arena? Larger arena means more cameras or wider lenses.
- How many cameras can be afforded? With more cameras smaller lenses can be used, which leads to better performance (although more processing power).
- What precision is desirable? Good precision implies using many cameras.
- What should the system be used for? 3-d reconstructions implies using cameras all around the arena so that players can be covered from more than one direction.

There are some advantages with placing all the cameras at the same location (in the middle of the long side at the same side as the main broadcast camera).

First there is the cost issue, long cables is very expensive and is inconvenient to install. Second, if the system has an operator he/she would probably sit at this locations since it has the best overview of the pitch, and it is convenient that the operator sees the same view from the cameras as in real life.

As long as all the camera pairs are working independently, there is no problem with adding and reducing the number of cameras used. Thus there should be no problem leaving these questions unanswered until the system is about to be installed at a specific arena.

Chapter 6

Detecting the players

6.1 Related work

In recent years several papers have appeard that deals with tracking players in different sports. Many of these papers ([2] for example) uses template matching to track the players from one frame to the next. This usually requires manual initialization of the player position. Also these papers do not really concentrate on the actual measurement of the players positions but rather the tracking of the players.

6.2 Foreground extraction

The job of finding the players gets significantly easier if the system knows what part of the image that is background (pitch, spectators and other static parts of the arena) and what is foreground (the players and the ball). The simplest way to determine this is to use a model of the background and then extract this model from the image acquired each frame.

6.2.1 Background model

The background model is essentially an image built by Gaussian distributions. The system is initialized by collecting N frames of an empty pitch which represent the background, the mean and standard deviation of each pixel for these N frames is calculated and this builds the first background model.

A problem with this background model is that it is only valid for a brief moment. As soon as the lighting changes or something else happens in the background, this model will become invalid. Thus, in order to have a valid background model it has to be updated continuously. This is however not that hard of a task if the system successfully can detect all players and the ball. Since the system then knows where the players are, it is trivial to guess that the rest of the image depicts the

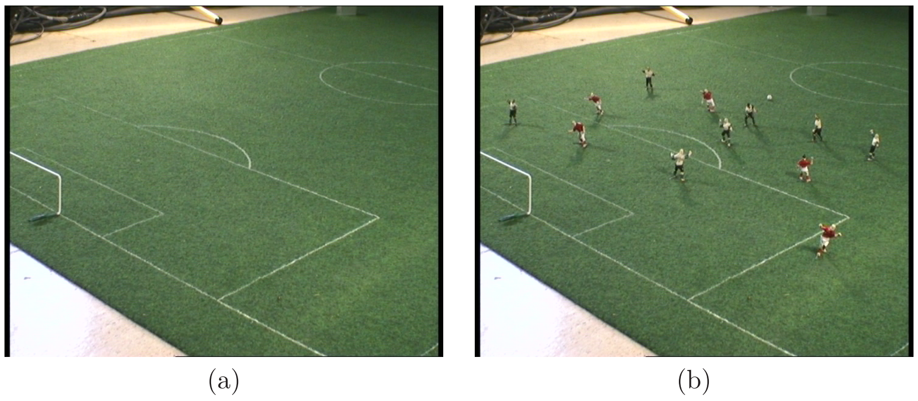


Figure 6.1. (a) Left background model. (b) Left input image.

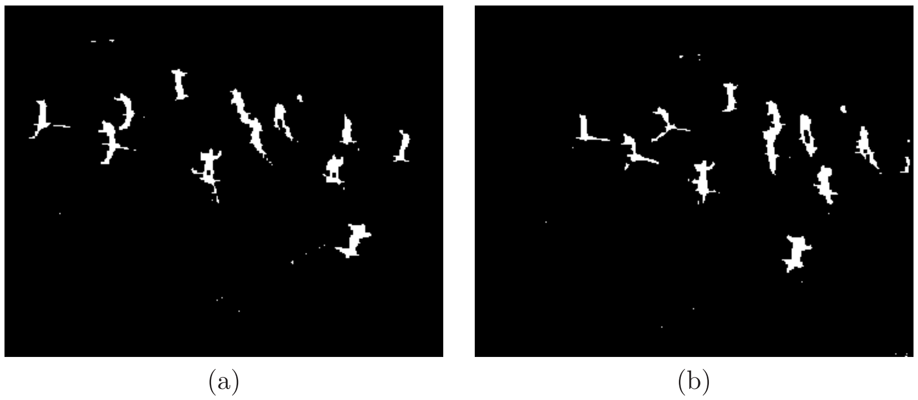


Figure 6.2. (a) Left extracted foreground. (b) Right extracted foreground.

background, and this can then be used to update the background model. Since the players usually don't stand still for more than a few seconds most of the background will be updated successfully.

6.2.2 Right to left model

A problem with the background model is that the shadows that the players cast are not in the background model, this causes these shadows to appear in the foreground. If tracking with one camera this is a big problem since it is hard to determine the center of the players when there is a shadow near the player all the time that changes direction with time. With stereo vision it is usually possible to see that the shadow lies in the ground plane, this it does no harm. However in some conditions the shadow can interfere with the disparity estimation, thus it is desirable to remove the shadow from the foreground anyway. A way to get rid of these shadows is to warp the right camera image to the left through the ground plane. This result in an image very similar to the left image but with the difference that things that are not in the ground plane will be tilted. Since the shadows are all in the ground plane it is possible to see the difference between the players and their shadows.

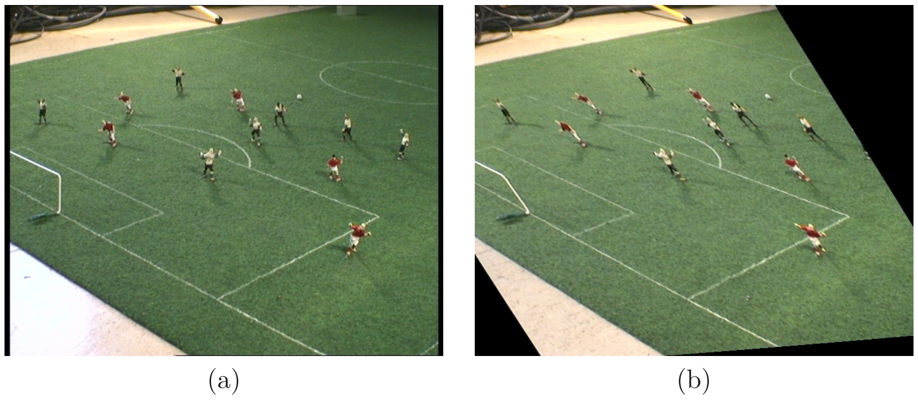


Figure 6.3. (a) Left input image. (b) Right input image transformed to left view.

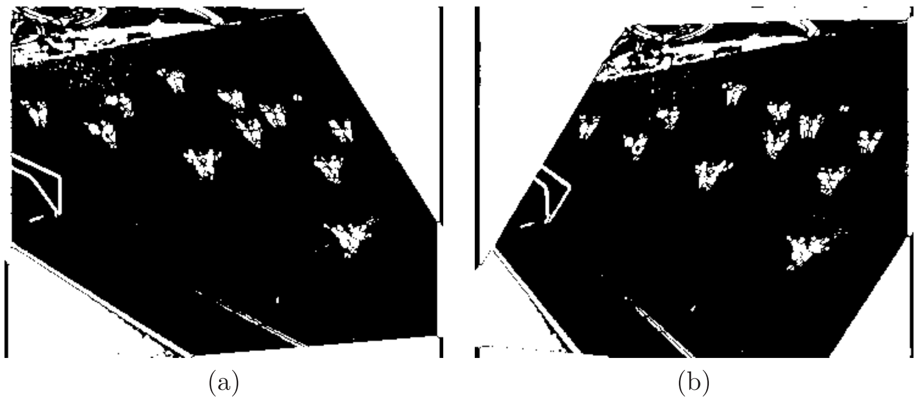


Figure 6.4. (a) Left extracted foreground. (b) Right extracted foreground.

Combining these two techniques gives a quite good foreground extraction that neither has any problem with shadows nor any problem with light changes.

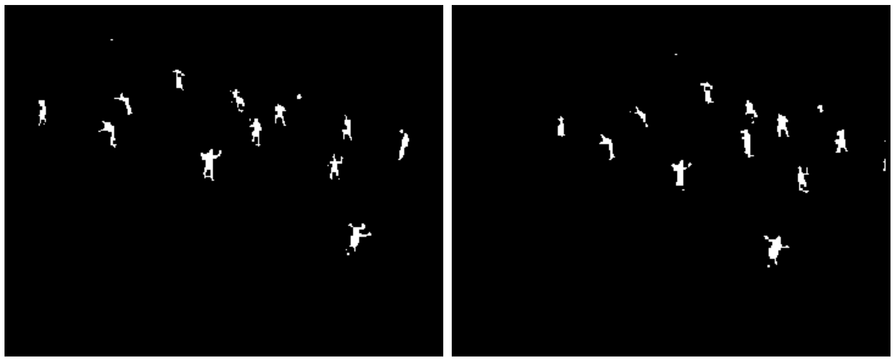


Figure 6.5. Final foreground extractions for left and right image

6.3 Disparity estimation

As mentioned earlier, the disparity estimation is performed on the rectified images. The left and right rectified images of a scene are shown in figure 6.6.

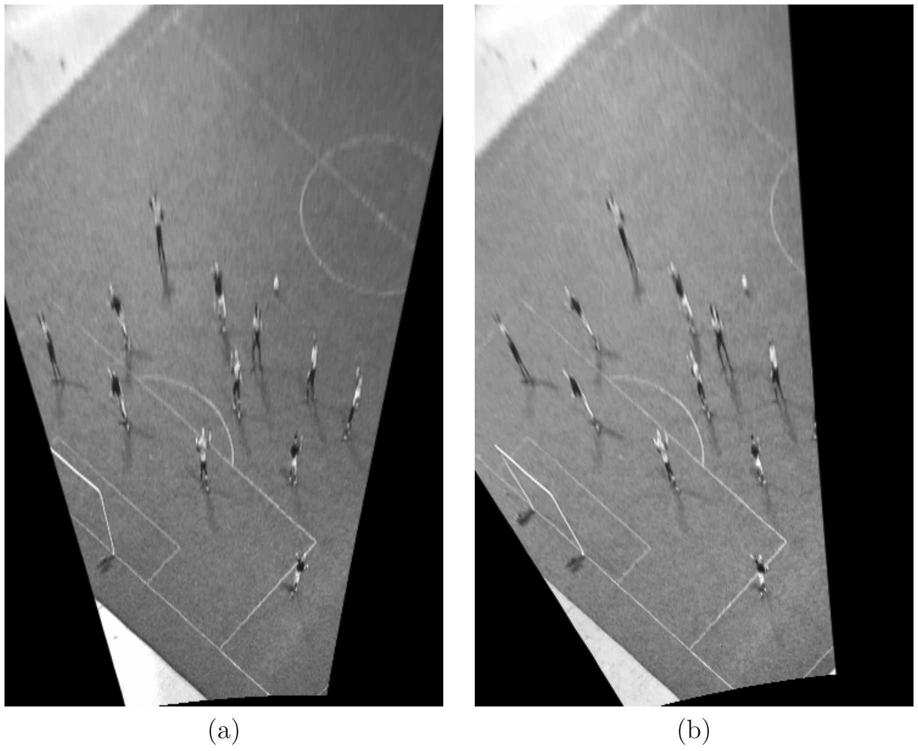
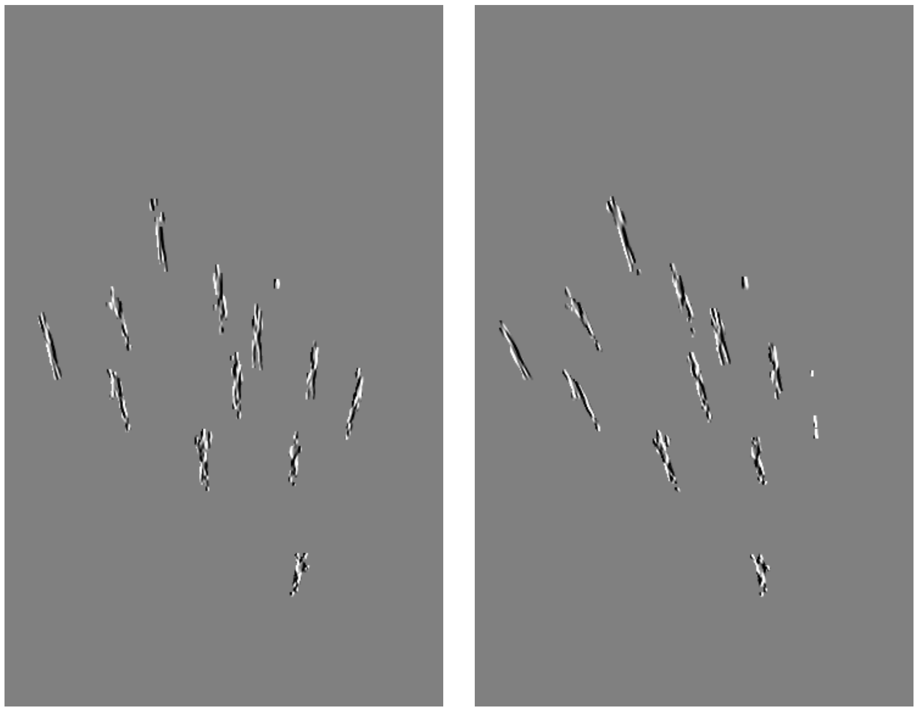


Figure 6.6. (a) Left rectified image. (b) Right rectified image.

The phase based stereo was chosen since it was the fastest method available and seemed to give a quite good estimation of the disparity in the reference scenes.

Calculating disparity directly from the two images will not work very well as seen in figure 6.8a. The reason for this is that the black borders are too dominant in the images (and not equal in the left and right image). These black borders affect the disparity image in a very unsatisfactory way. Another problem in these images is that the lines in the pitch can affect the disparity estimation as well. As shown with the bottom most player in figure 6.6a and 6.6b, in the left image the head of the player is on one side of a line in the pitch, and in the right image it is on the other side of that line. This can seriously affect the disparity estimation.

To deal with these problems an edge image is calcualted from the rectified images (sobel-x). This edge image is then masked with the mask created previously (figure 6.5). The mask is rectified in the same way as the original images. The resulting edge image is shown in figure 6.7.



(a) (b)

Figure 6.7. (a) Left edge image masked with left mask. (b) Right edge image masked with right mask.

It's clear that the resulting edge image only contains foreground objects. The resulting stereo estimation is shown in figure 6.8b.

Figure 6.8b shows the final disparity image. The difference between the two disparity images is quite big. The procedure of creating an edge image and then mask this really improves the result.

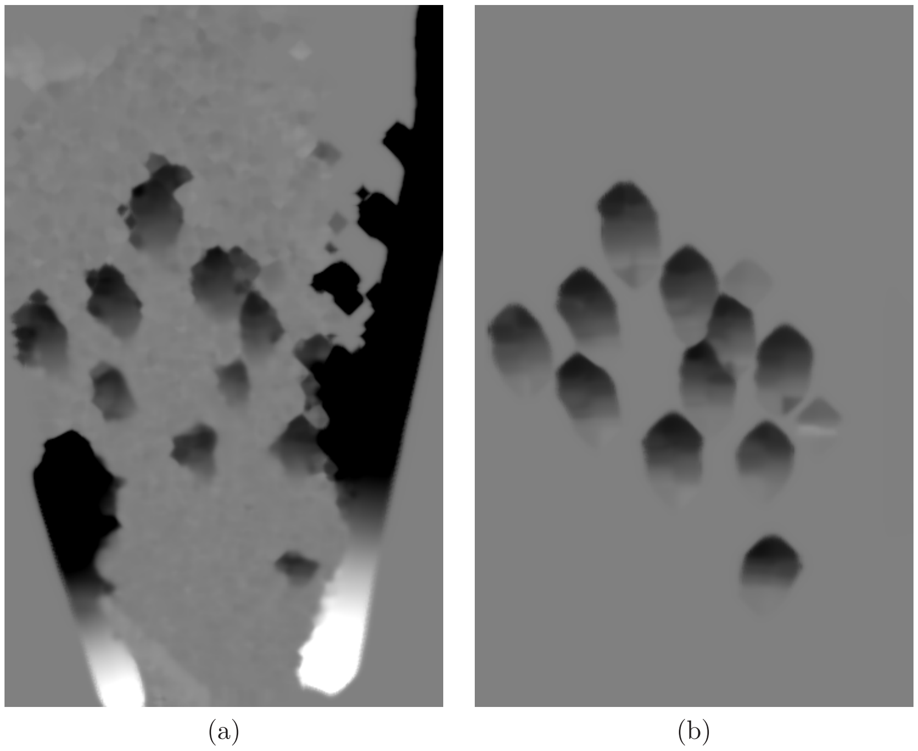


Figure 6.8. (a) Disparity calculated from the rectified images. (b) Disparity calculated from the edge images.

Figure 6.9a shows the same image as figure 6.8 but now masked with the left mask. This stereo calculation is performed both from left to right and from right to left as seen in figure 6.9b. The reason for this is that the disparity estimation assumes that there is no discontinuities in the images, but at the edge of the players this is obviously the case. By calculating from left to right and from right to left it is possible to detect where the discontinuities are, and remove them from the disparity image.

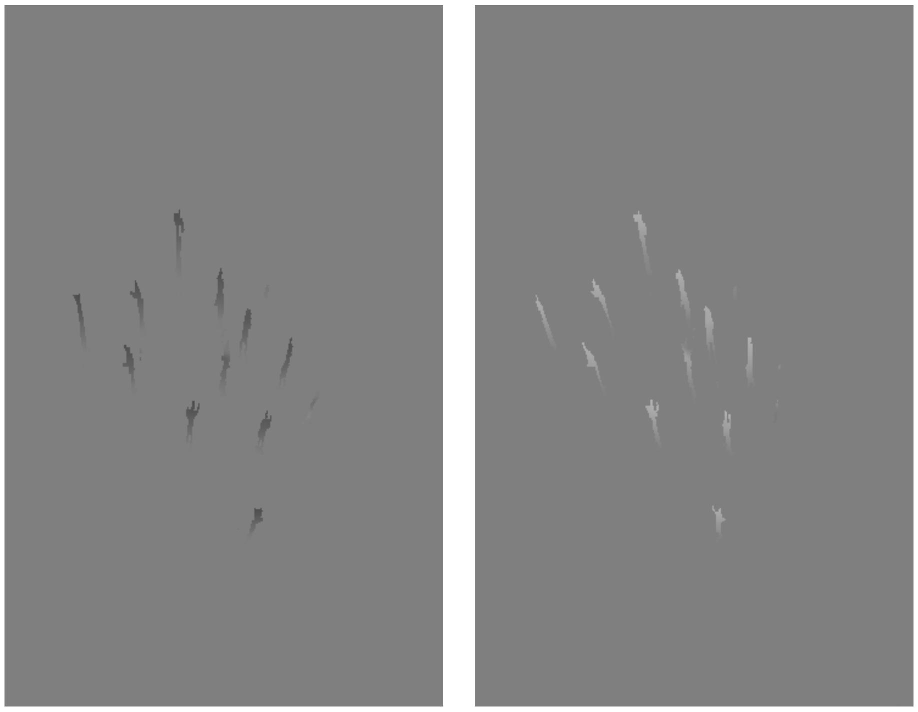


Figure 6.9. (a) Disparity calculated from left to right then masked with left mask. (b) Disparity calculated from right to left then masked with right mask.

6.4 Player localisation

The player's position is extracted from the disparity image. A quite simple way to do this is to project the information in the disparity image down to the ground plane to what is here called a *density map*.

6.4.1 Density map

For each pixel in the disparity image (figure 6.11a) a real 3-d space coordinate is estimated. This calculation is done according to figure 6.10.

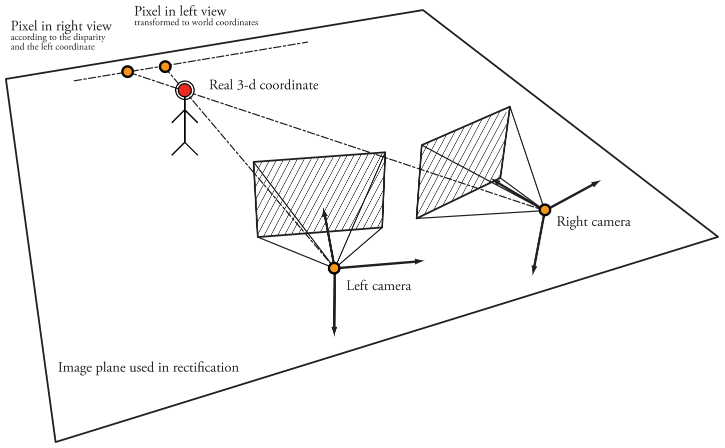

Pixel in right view
according to the disparity and the left coordinate
Pixel in left view
transformed to world coordinates
Real 3-d coordinate
Right camera
Left camera
Image plane used in rectification

Figure 6.10. Triangulation to retrive real 3-d position.

For each pixel in the left image its coordinate is calculated in 3-d space ($\bar{p}_b$) assuming that the coordinate is located in the ground plane. Using the information in the disparity image it is also possible to calculate the coordinates for the right image ($\bar{p}_d$).

$$\begin{aligned} \bar{p}_b &= f_{t2w}(\bar{u}) && (6.1) \\ \bar{p}_d &= f_{t2w}(\bar{u} + [d(\bar{u}) \quad 0]^T) && (6.2) \end{aligned}$$

f_{t2w}	transformation function from transformed image coordinates to plane coordinates.
$\bar{u}$	2-d image coordinate.
$d(\bar{u})$	disparity value at coordinate $\bar{u}$.

By defining the left and right camera coordinate as $\bar{p}_a$ and $\bar{p}_c$ respectivelly, the following two lines in 3-d space can be defined.

$$\begin{aligned} \mathcal{L}_1 : &\quad \bar{p}_a + \mu_1(\bar{p}_b - \bar{p}_a) && (6.3) \\ \mathcal{L}_2 : &\quad \bar{p}_c + \mu_2(\bar{p}_d - \bar{p}_c) && (6.4) \end{aligned}$$

Line 1 describes all the possible world coordinates for the image coordinate $\bar{u}$. While line 2 describes the same but for the right camera coordinates. The world coordinate is chosen at the midpoint of the shortest line between the two lines $\mathcal{L}_1$ and $\mathcal{L}_2$.

When all coordinates have been calculated, the 3-d space is divided into a evenly distributed 3-d grid. Intergration along the z-axis produces the density map shown in figure 6.11b.

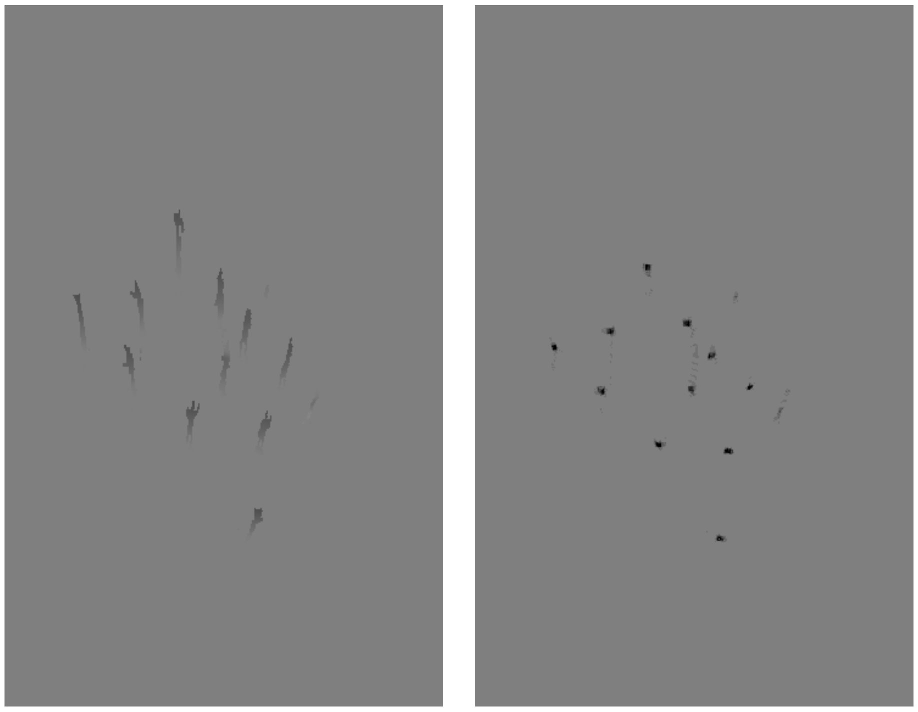

(a) (b)

Figure 6.11. (a) Disparity image. (b) Resulting density map.

The displacement of one pixel in an image measured in meters on the ground plane is different depending on where in the image the displacement is. This is easily seen in the image since the players have different heights depending on how far away they are from the camera. The effect is not as apparent sideways. Since all the players should form spots of the same size, this effect has to be accounted for.

The figure below shows the area on the ground k that is covered by one pixel in the camera view.

The area k is described by equation 6.5. It is assumed that the area is constant sideways but changing with the distance from the camera.

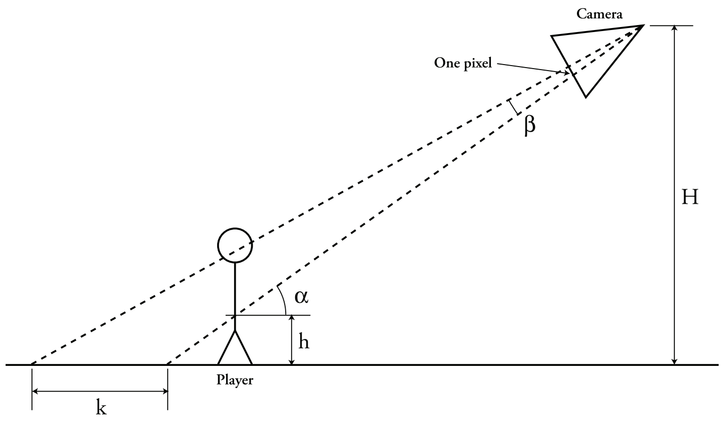


Figure 6.12. Estimation of area behind one pixel in the camera image.

$$k = \frac{H}{tan(\beta + \frac{\pi}{2} - \alpha)} - \frac{H}{tan(\frac{\pi}{2} - \alpha)} \tag{6.5}$$

$$\alpha = atan(\frac{H-h}{d}) \tag{6.6}$$

$$\beta = \frac{fov_h}{s_v} \tag{6.7}$$

k	size of the voxel.
d	the horizontal distance between the voxel and the camera.
h	the height of the voxel.
H	the height of the camera.

When transfering a pixel from the disparity map to the density map each pixel is thus scaled by a factor of $1/k$.

6.4.2 Position extraction

The task of locating the players is now a task of finding the spots in the density map that are large enough to represent a player. Also, if a spot is too large, one might assume that there are two or more players standing very close to each other.

Finding the player among these spots can be done in several ways, for example using some sort of hill-climbing or mean-shift analysis. However, a very simple tool can also be applied, a max-filter (a filter that takes the maximum value in a NxN region).

A max-filter with size 5x5 is used. This size has been chosen since it is about the expected size of a human standing upright in this plane. The density image is first lowpass filtered, to lower the sensitivity to singular points (which usually are not correct anyway). The values in the density map are an indication of how much of a player occupies that particular space. If this value is small one can conclude that it is probably noise that isn't interesting. So all values under a specific limit are removed, this threshold has been chosen empirically and depends on what resolution the players have in the images. The resulting image is then subjected to the max-filter. Extreme points (maxima) should occur where the output of the max-filter and the lowpass-filter are equal. The size of the lowpass kernel has also been chosen empirically and is in this case a standard $[1\ 2\ 1] \star [1\ 2\ 1]^T$ kernel.

When a person isn't standing upright, this method may result in several maxima within one person. Since the players are tracked trough several frames there should be no problem to remove the false maxima since one can assume that the players moves quite slow (when $25fps$ is used anyway). And it's known that new players can't just appear out of the blue, so if maxima like this occur, it is possible to conclude that they are false.

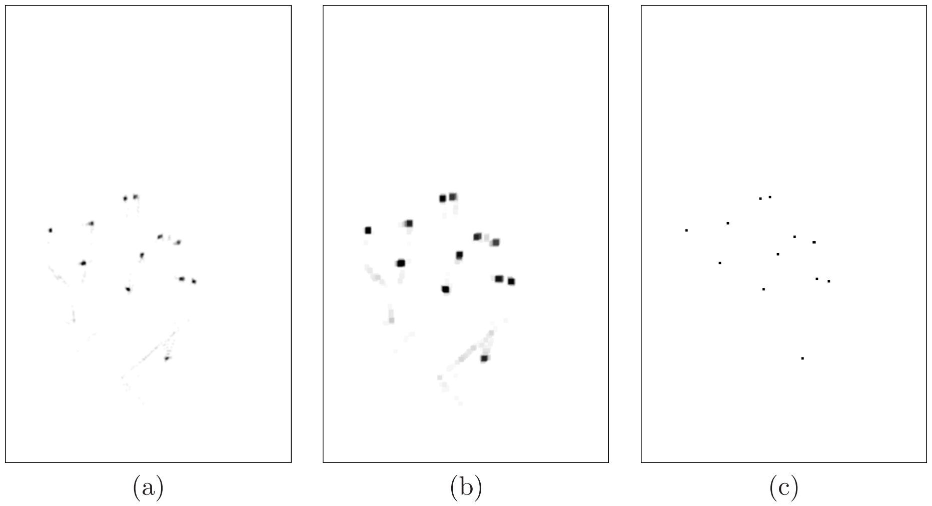


Figure 6.13. (a) Low-pass filtered density map. (b) Max-filtered density map. (c) Extracted extreme points (position image).

6.4.3 Position evaluation

The stereo estimation results in two different disparity images, one with the left image as reference and one with the right image as reference. Because of this the procedure described earlier with density map and position extraction is performed on both disparity images. The result is shown in figure 6.14.

As seen in the figure 6.14, they are almost identical, however not exactly. Two player indications are shown in the right image surrounded by circles. These are not present in the left image. By comparing these two results it's possible to eliminate false players, i.e. points that have no match in both images. We try to match the points in both images in a least-square fashion, that is, the difference in positions between the points in the left image to it's companion in the right image is minimized. Points that have no match are eliminated.

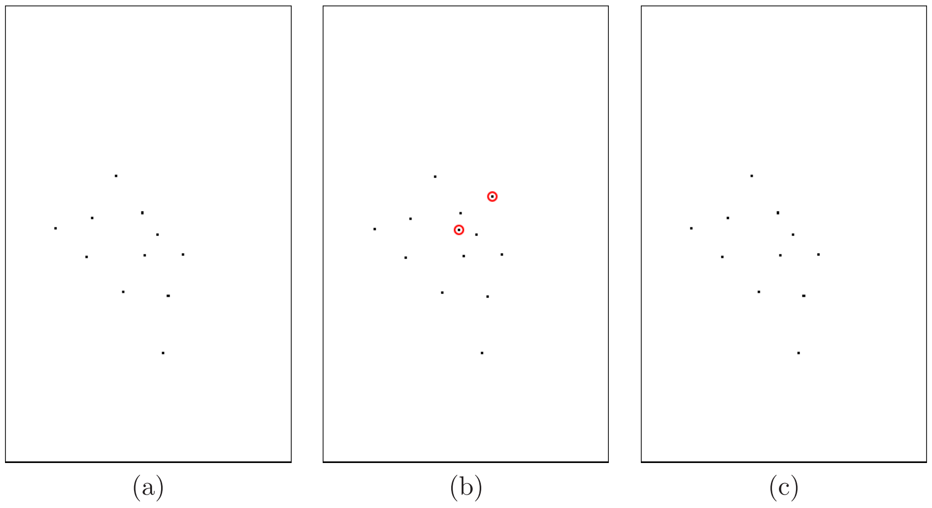


Figure 6.14. (a) Position image from left-to-right disparity image. (b) Position image from right-to-left disparity image. (c) Resulting position image (equal to the first image in this case).

6.5 Player segmentation

After the player's positions has been established in the plan-view image it's desireable to establish what area that the player occupy in the original image. The first step to do this is to label the position image that was created in the position evaluation step, the result is shown in figure 6.15a. The labeling procedure tags each isolated segment in the position image with a unique number (color). The resulting labeled image is then expanded so that every position occupies a quite large area. The size of this area has been chosen to be the largest area a human can occupy in this view, the result from this operation is shown in figure 6.15b. Two regions that are very close do not grow over each other but stops when they meet, as seen in the figure.

The target for the player segmentation step is to project this labeled information to the mask created in an earlier chapter. This is done by projecting the left and right mask using one of the disparity images as done in the density map procedure. The mask itself isn't projected but rather it's coordinates, from the projected coordinate the color from the expanded position image can be extracted. The result is shown in figure 6.16.

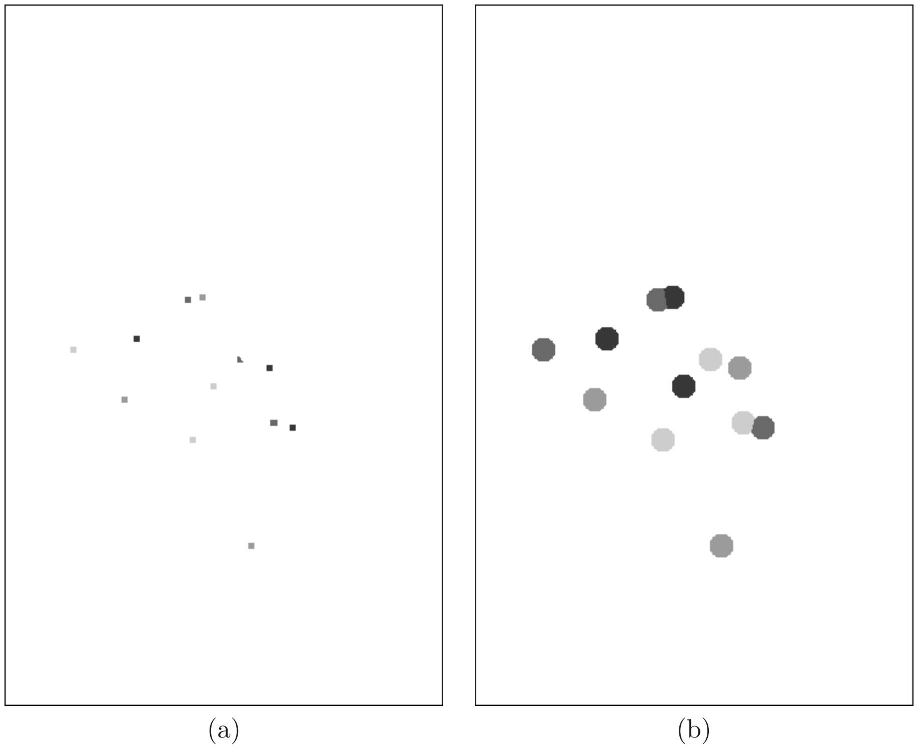


Figure 6.15. (a) Labeled position image. (b) Expanded position image.

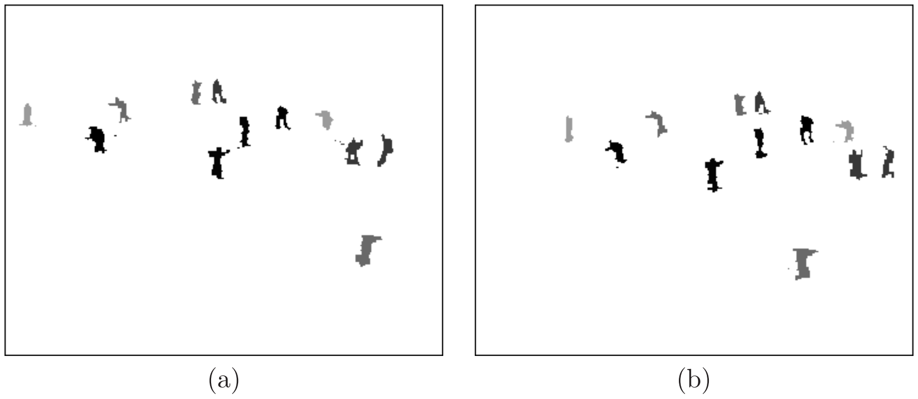


Figure 6.16. (a) Left mask, labeled. (b) Right mask, labeled.

6.6 Player position refinement

The first position extraction is quite rough. The position is estimated with the same resolution as the plan-view map and thus there is room for improvementes. There are two main reasons to why the first stereo estimation isn't good enough. First there is a resolution problem, the image of the transformed image plane is 360x576 pixels. The entire pitch that is covered is squeezed into this image, thus the players become wery thin in this view. The stereo estimation uses a quite large quadrature filter (11x1), this means that the estimation can never be very accurate since the player's are so thin, thus there are no details in the players. One could of course increase the image size to improve this but the computatonal overhead would make this undesirable. The second reason is that the disparities of the players are still quite large (even though the ground plane is used for the transformation). The stereo algorithm works at it's best when the disparities are small.

A solution for this is to select a new image plane and do the stereo estimation again. One smaller plane is chosen for each player, this plane is located at the roughly estimated position of the player and has been rotated 90° around the y-axis. An example of this is shown in figure 6.18a. When transforming the left and right view to this plane, everything that lies behind the plane is shifted to the left, and everything in front of the plane is shifted to the right. The transformed images are shown in figure 6.17a - 6.17f (all players are not listed here). The high resolution and small disparity of this image pair gives a good position estimation. Since the mask is segmented it is easy to only include the pixels in the disparity image that belongs to each player in the position calculation.

The stereo is estimated in the same way as before, i.e by calculating an edge image from the transformed images and then mask the images with the foreground mask (figure 6.17g and 6.17h). The stereo estimation (figure 6.17k) is then masked with the labeled image created earlier (figure 6.17i and 6.17j). As before the stereo estimation is projected to a plane in the ground (figure 6.17l) and the new position is estimated by calculating the center of mass in this image.

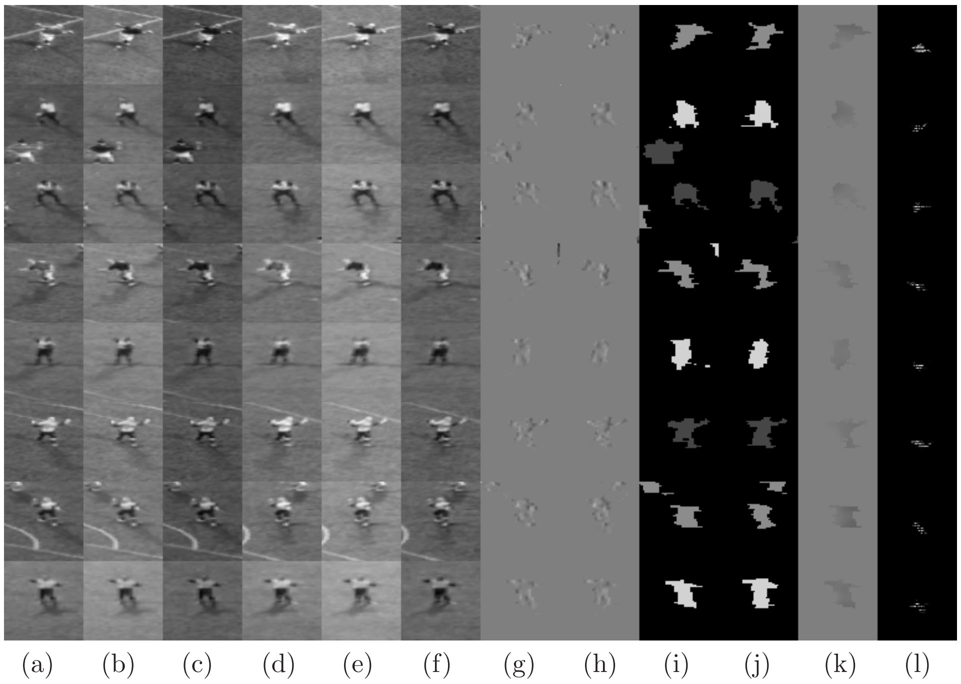


Figure 6.17. (a)-(f) Zoomed images. (g)-(h) Zoomed edge images. (i)-(j) Zoomed labeled mask images. (k) Stereo estimation. (l) Plan-view density map.

Figure 6.18 shows a larger version of a zoomed player and the resulting disparity estimate.

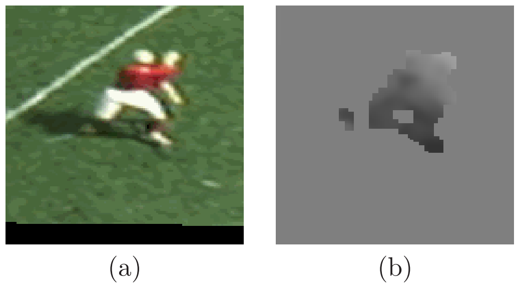


Figure 6.18. A larger view of a zoomed player and his disparity estimate.

Using the disparity estimate and the texture from the zoomed player, a simple 3-d reconstruction can be viewed as shown in figure 6.19.

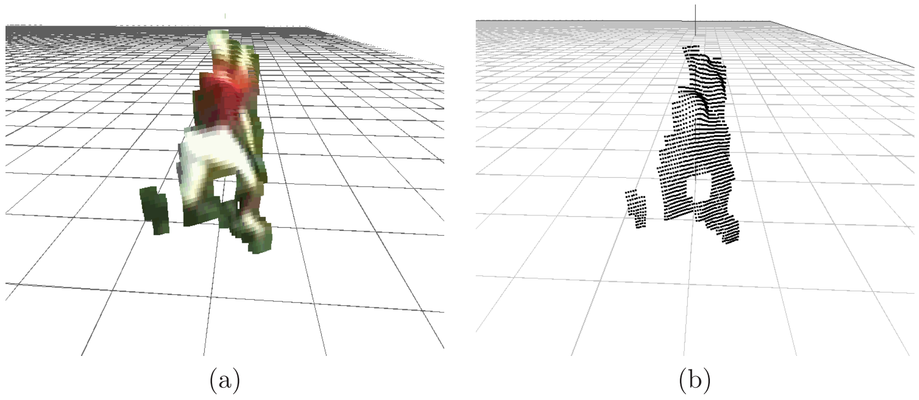


Figure 6.19. Disparity image projected onto 3-d space.

Chapter 7

Detecting the ball

During the process of finding the players it became clear that it wasn't easy to find the ball in the same manner. Much of this was because of the way that the first stereo estimation was performed. During that step the images are transformed to a plane that lies in the football pitch, thus making the disparity only to move along the x-axis. The disparity decreases with the height above this plane. A player has a limited height above the plane, thus it has a limited disparity range. The ball however can reach great heights and thus the disparity will be large as well. This makes it hard for the disparity estimation to correctly estimate the ball position.

Because of these problems a separate process of locating the ball was developed.

7.1 Related work

Some papers ([6], [1]) propose a post processing algorithm to find the ball. The algorithm tries to find ball like features and tracks these with a kalman filter. When enough frames have been processed the ball is located by looking at the trajectory of the tracked objects. Since the ball is a small rigid object with a behaviour that could be anticipated this works quite well. However since the goal is do to the calculations in real time, this is not really a solution. Other simpler solutions exists as well, [2] uses a manual initialization of the location of the ball, i.e an operator marks where the ball is each time the tracker has lost the ball. This solution could be used since an operator is probably needed for a system like this. However an automatic way of locating the ball would be desirable.

7.2 Ball candidates

The process of locating the ball starts with locating all the parts of the image that can be a ball, that is locating all the ball candidates. This is done with a filter that takes the maximum value of a NxN region and substract the maximum of a (N+1)x(N+1) region (just the border). This filter gives high values as response to small white objects on a dark background. This produces a list of 2-d ball candidates in the left and right image. The ball candidates are located in the images using the left and right mask. A ball is supposed to be a small round object in the mask, thus we extract all objects in the masks that fits this description.

For all the candidates in the left view a 3-d line is calculated. The line emanates from the left camera and passing through the candidate in the image plane. The real position of the candidate is somewhere along this line. For every candidate in the right view the same calculations is performed.

Every line in the left view is then compared with every line in the right view, if they intersect (or closely intersect) it can be a match, thus their intersect point (or closest point if intersection) is considered to be a 3-d ball candidate. When all the lines has been compared, a list of new ball candidates with 3-d coordinates has formed. These coordinates are projected to the left camera view and using a correlation algorithm with a ball model an estimate of how “ball-like” the candidate is, is created.

Chapter 8

Tracking

8.1 Tracking the players

A very simple tracking algorithm has been implemented for the players as described below. This can of course be done in a much more elaborate way, using some sort of filter (kalman etc.). But this is beyond the scope of this thesis.

8.1.1 Initialization procedure

For the first frame it's assumed that all the players are standing upright (so that they are not producing any false positions) and that they are not too heavily occluded. The players velocity and accelerations is initialized to zero.

8.1.2 Tracking procedure

In consequtive frames the localization procedure is perform again, and then the data is compared with the data that has been stored from previous frames. From previous frames there also exist a list of players, with position, velocity and acceleration stored. During the player localization step a new list is created that contains all the players found in the current frame, these only have a position. For all the saved players an estimated position is calculated (based on old data). This position is our estimate on where the player should be in the current frame. This position is compared with the new position information. In the same manner as with the position evaluation, the saved players position is matched with the new position data in a least-square fashion, where the total matching score is minimized. The matching score is calculated based on the difference between the estimated position and the new position, and the difference between the old velocity vector and the velocity vector required to get to the specific position. If a saved player can't find a match among the new positions, it is considered to be occluded. If a new position can't find a match among the saved players it is considered to be a false position unless it is on the edge of the image. It is then considered to be a new player, entering the scene.

8.2 Tracking the ball

As with the tracking of the player, the tracking algorithm used for tracking the ball is very simple as described below. This can be done in a much more elaborate way, using some sort of filter (kalman etc.). But this is beyond the scope of this thesis.

8.2.1 Initialization procedure

For the first frame the ball candidate that has the best score is selected, i.e. the candidate that is most “ball-like”. The velocity and acceleration for the ball is initialized to zero.

8.2.2 Tracking procedure

In concurrent frames all the ball candidates are compared with the current ball. One would expect that the ball hasn’t changed its speed, acceleration and position too much from the previous frame. A new “ball-like” score estimate is created, based on the score for each candidate and the difference in speed. Also the distance between the estimated position of the ball and the position of the candidate is taken into account, this distance should in normal cases be quite small. When a candidate with the lowest total score has been found it is considered to be the ball, the ball model is then updated (position, speed and acceleration) with the information from the candidate. If the total lowest score is higher than a certain limit, it’s concluded that a ball couldn’t be found in the current frame (it is probably occluded), when this happens the balls position is updated with saved information (acceleration and speed), thus the current position is estimated.

Part IV

Evaluation

Chapter 9

Evaluation

Looking at the overall performance of this system it is clear that the players are successfully found if they are not too heavily occluded.

The result of the system developed looks very promising. The system has been tested on three reference scenes. The two first scenes is real live scenes that include two players and a football. The third is a static model of a pitch that was chosen since it allowed for a controlled environment.

9.1 Reliability

The reliability of the system seems to be quite good. The players are located using the plan-view density map, since players seldom occludes each other from above it is quite easy to distinguish different players through this view. Problems occur when two players are too close to each other, those situations have been resolved using tracking over several frames.

The first two reference scenes were used to define the reliability of the implementation. The reliability is defined as a percentage of the frames in the scene that the implemention was unable to find the player.

9.1.1 Scene 1 (real)

One frame for the first test scene is shown in figure 9.1. The scene consists of two players and one ball. The players are running towards the goal and one of the players shoots the ball into the goal.

Table 9.1 shows the reliability value when detecting the players and the ball. As seen in the table the reliability value for the ball is much higher than for the players. This is natural since the ball is occluded by the players much more often than the players themselfs.

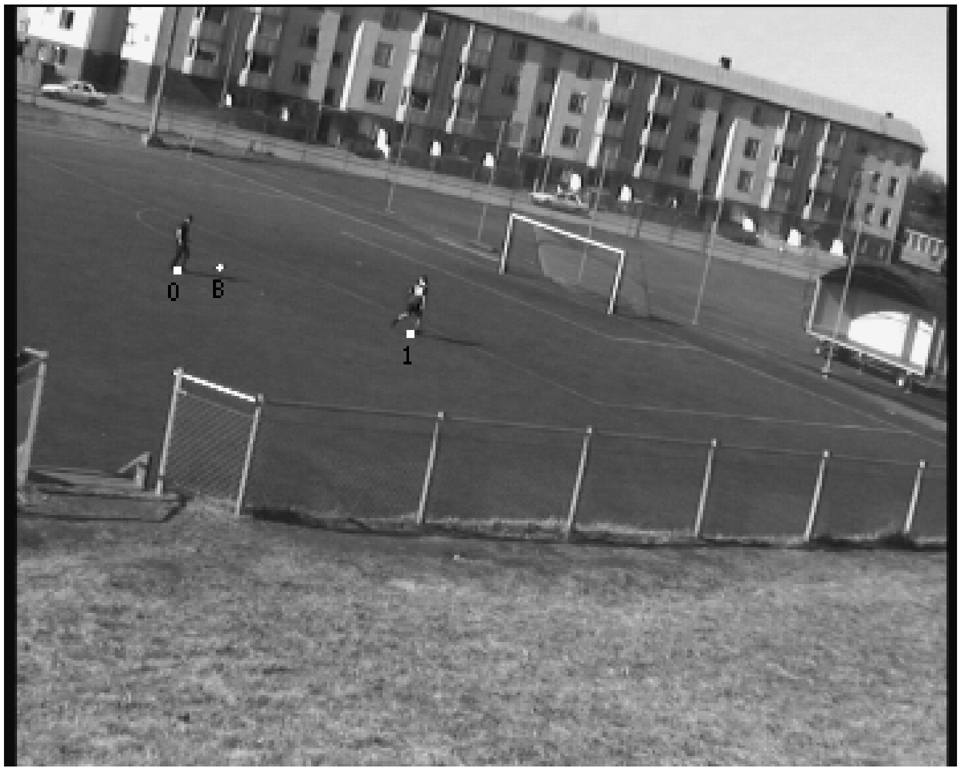


Figure 9.1. First test scene (149 frames).

	# frame	# not found	# found	reliability
player 1	149	0	149	100%
player 2	149	7	142	95%
ball	149	54	95	64%

Table 9.1. Detection result for scene 1.

It should be clearly noted that even if a player or the ball isn't found in one frame, he is still tracked correctly in most cases since the new position is predicted using saved data (old position and velocity).
Figure 9.2 shows the tracked paths for scene 1.

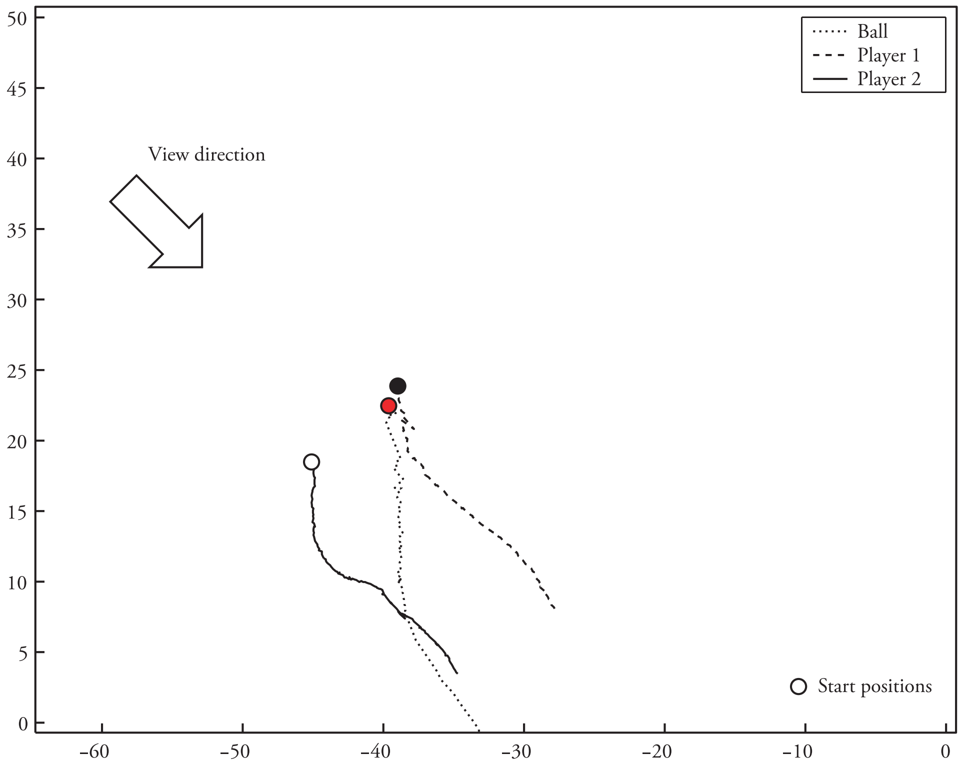


Figure 9.2. First test scene.

9.1.2 Scene 2 (real)

One frame for the second test scene is shown in figure 9.3. This scene also consists of two players and one ball. The players starts by running away from each other. The players then turn and run toward each other and smash together at the end of the scene.

Table 9.2 shows the reliability value when detecting the players and the ball. As with scene 1 the the reliability value is much higher with the ball. The reliability values for the players are also higher in this this scene than in scene 1. One reason for this is that the players run into each other, therefore only one player is seen in those frames.

	# frame	# not found	# found	reliability
player 1	269	49	220	82%
player 2	269	18	251	93%
ball	269	64	205	76%

Table 9.2. Detection result for scene 2.

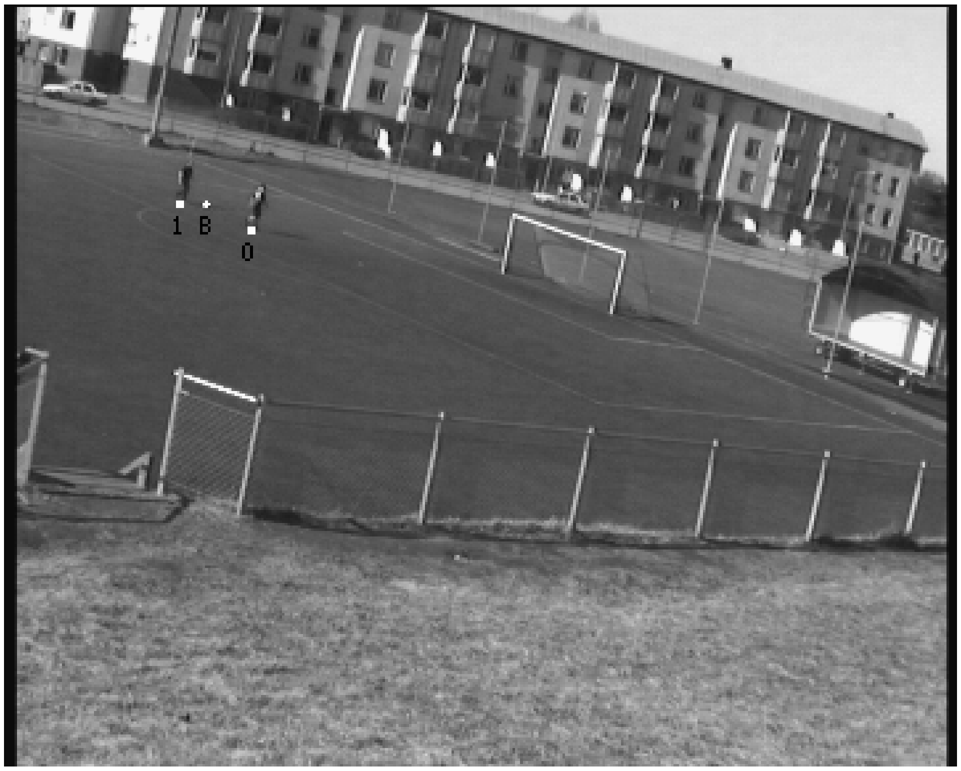


Figure 9.3. Second test scene (269 frames).

Figure 9.4 shows the tracked paths for scene 2. Note that there is a discontinuity in the path for player 1 at the point where the two players meet. This is not correct but rather a consequence of player 1 being occluded by player 2, thus his position has to be predicted. When player 1 is visible again the tracker recognizes this and the tracking is correct again. By estimating the velocity and acceleration in a better way, this fault in estimation would not be so apparent.

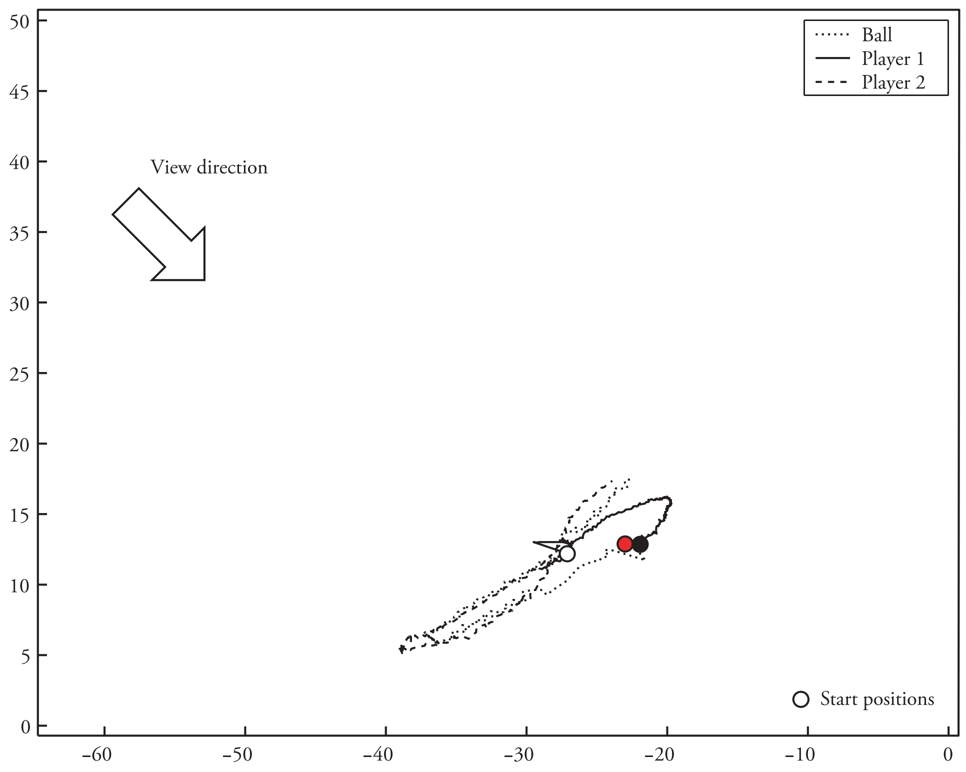


Figure 9.4. Second test scene.

9.2 Precision

One of the more important aspects of the system is to know how well a players position is estimated. For now the position is estimated directly from the disparity image. The disparity image is projected to the ground and creates a pile of voxels who's centre of mass describes the position of the player. This means that the position estimated is measured on the side of the player that is facing the camera, thus not really the centre of the player as one might expect. Depending on what application the system is to be used for this might or might not be adequate.

The third reference scene was to be used as an indication how how well the position was estimated.

9.2.1 Scene 3 (model)

A simple test was carried out to give an illustration as to how well the position is estimated. A player (in the pitch model) walks along a known path and the resulting measurements are shown in figure 9.6. The scene was animated by hand and the player was rotated to simulate the movement of a real player while still holding the center of the player along the line.

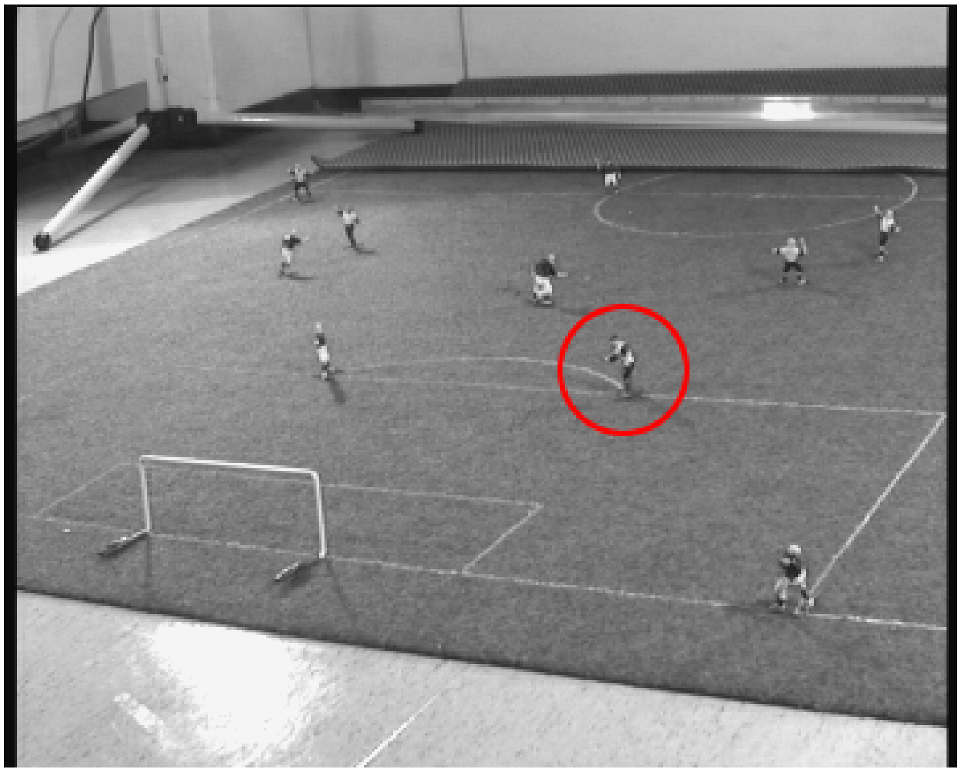


Figure 9.5. Third test scene (15 frames).

As seen in figure 9.6, the positions seems to be estimated with a small offset from the ideal route (along the straight line). This offset is the cause of an error in the camera calibration, since this can be adjusted with a little more work this is ignored. According to the graph, the maximal estimated error is somewhere in the region of $\pm 0.2m$ with a standard deviation of $0.142m$. However it should be noted that the scenario where this measurements was taken does not include occlusion or other harder situations.

It is important to note that this measurement is only valid in this resolution and at this particular distance from the camera.

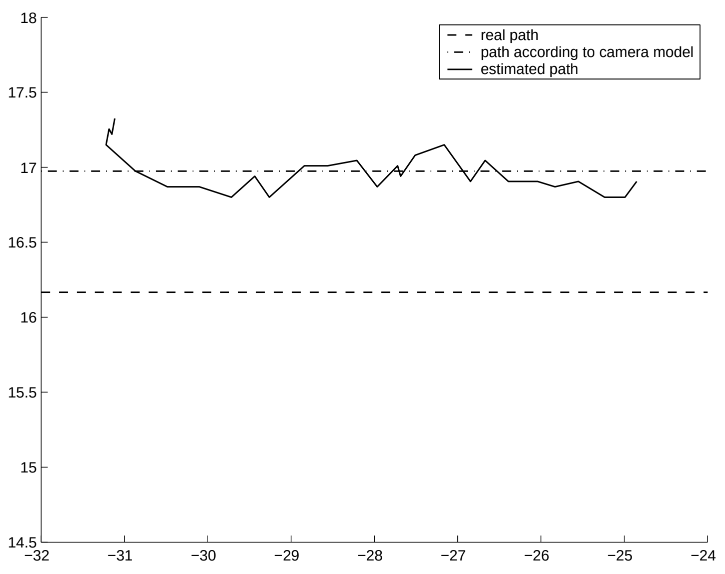


Figure 9.6. Path for the marked player.

9.3 Camera placement

As mentioned earlier there are no real limit on how many cameras that can be used. More cameras will improve the overall performance of the system but will also make the price higher as well as demanding more computational power.
It is unlikely that the system can work properly with less than 8 cameras since the resolution of the input images the would be too low. The system was designed with 8 cameras in mind, but no changes are necessary if more are used.

9.4 Computational power

Development of the system was performed on a Pentium 3, 2x700Mhz running Slackware Linux. The total time used to calculate one frame is about $2s$ (when using one thread). With opimizations and newer hardware (Pentium 4, 3.8Ghz) it has been possible to reduce this time to about $30ms$ (for at most 25 tracked players), thus ensuring the calculations to run in real-time (25 times per second with PAL cameras).

Chapter 10

Summary

10.1 Conclusion

The goal of the thesis was to investigate if it was possible to implement a system used to find and track the players on a football pitch. This thesis has shown that this is indeed possible with an aceptable precision and reliability in the reference scenes. To be able to show that this will acually work in a larger scale during a real football game would require much more work. However I think that this thesis has shown that stereo vision has huge potential and it is very possible that this technique can be used in a real tracking system.

In the beginning of this thesis the following goal was defined:

1. Find a working camera setup that can cover the entire pitch.
2. Locate and track the players in the reference scenes.
3. Locate and track the ball in the refrence scenes.

Below is a short summary on how these goals was reached.

1. Two possible camera setups was defined. As described these are not the only two possible setups but rather a suggestion.
2. The players are located and tracked by extracting information from the disparity between two rectified camera views. The disparity information is projected to a ground plane thus creating collections of white pixels where the players are located. This image is segmented and each player is assigned a segment of pixels. The position of each player is calculated as a weighted position of the segment.
3. The ball is located and tracked in the images. The ball is first located with a simple filter that extracts all small white dots in the image. The correct ball is then selected based on the foreground mask (the ball is supposed to be small and round in the mask as well).

10.2 Future work

In order to have a complete system for tracking the players much work is still to be done. The tracking of the players can be done in a more elaborate way, for example with a correlation-based search from one frame to the next. Also much more tests have to be performed with real football images with many players on the pitch.

The position measurement can probably be significantly improved, this is especially true for the ball. The biggest problem with the estimation as it is done now, is that there is no real control as to where on the player the position is estimated. The position is estimated where the players have their largest bodymass. This position is probably not the same for a player that is standing upright and a player that lies down on the ground. When the players is “zoomed” (during the position refinement step) and stereo has been calculated for the second time we have a lot of good measurements on the player. If the program can make use of these measurements in a better way and from these measurements locate the chest for example, the precision would improve a lot.

Since the ball is a small rigid body it should be possible to quite accurately calculate the ball trajectory when it is occluded. As said earlier, the location of the ball could be estimated with a much better precision if it was subjected to a stereo estimation after the first location has been extracted.

Bibliography

[1] Xinguo Yu (and others). Trajectory-based ball detection and tracking with applications to semantic analysis of broadcast soccer video. Technical report, Institute for Infocomm Research., Singapore, 2003.

[2] Sunghoon Choi, Seo Youngduek, Kim Hyunwoo, and Hong Ki-Sang. Where are the ball and players? : Soccer game analysis with color-based tracking and image mosaick. Technical report, Department of Electrical Engineering, University of Pohang, San 31 Hyoja Dong, Pohang, Republic of Korea.

[3] Federation Internationale de Football Association (FIFA). Laws of the game, July 2003.

[4] Gunnar Farnebäck. Disparity esimation from local polynomial expansions. Technical report, Department of Electrical Engineering, Linköping University, Linköping, Sweden, 2001.

[5] Andrea Fusiello, Emanuele Trucco, and Alessandro Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, (12):16–22, December 2000.

[6] James Orwell Jinchang Ren and Graeme A. Jones. Estimating the position of a football from multiple image sequences. Technical report, Digital Imaging Research Center, Kingston University, Surrey, UK, 2003.

[7] Charles Loop and Zhengyou Zhang. Computing rectifying homographies for stereo vision. In *Proceedings of IEEE conference on Computer Vision and Pattern Recognition*, volume 1, pages 125–131, Fort Collins, Colorado, USA, June 1999. IEEE.

[8] ProZone. http://www.pzfootball.co.uk.

[9] Daniel Setterwall. Computerized video analysis of football - technical and commercial possibilities for football coaching. Master's thesis ISSN 1403-0721, Department of Numerical Analysis and Computer Science, KTH, Stockholm, Sweden, 2004.

[10] Changming Sun. A fast stereo matching method. *Digital Image Computing: Techniques and Applications*, pages 95–100, December 1997.

[11] Sport Universal. http://www.sport-universal.com.

[12] Carl-Johan Westelius. *Focus of Attention and Gaze Control for Robot Vision.* Phd thesis 379, Department of Electrical Engineering, Linköping University, Linköping, Sweden, 1995.

[13] Chan Yuan and Guangyou Xu. Correction of radial distorsion of fish-eye cameras. Technical Report 100084, Department of Computer Science, Tsinghua University, Beijing, China, 2000.

[14] Ye Zhang and Chandra Kambhamettu. Stereo matching with segmentation-based cooperation. Technical report, Video/Image Modeling and Synthesis Lab, University of Delware, Newark, USA, 2002.

[15] Lawrence Zitnick and Takeo Kanade. A cooperative algorithm for stereo matching and occulsion detection. Technical Report CMU-RI-TR-99-35, The Robotics Institute, Carnegie Mellon University, Pittsburgh, USA, October 1999.