Estimating the Height of an Unknown Object in a 2D Image

JOHAN VESTER

KTH Computer Science and Communication

Master of Science Thesis
Stockholm, Sweden 2012

Estimating the Height of an Unknown Object in a 2D Image

J O H A N   V E S T E R

DD221X, Master's Thesis in Computer Science (30 ECTS credits)
Degree Progr. in Computer Science and Engineering 300 credits
Royal Institute of Technology year 2012
Supervisor at CSC was Josephine Sullivan
Examiner was Stefan Carlsson

TRITA-CSC-E 2012:040
ISRN-KTH/CSC/E--12/040--SE
ISSN-1653-5715

Royal Institute of Technology
School of Computer Science and Communication

KTH CSC
SE-100 44 Stockholm, Sweden

URL: www.kth.se/csc

Abstract

In this thesis three different algorithms to estimate the height of an object are investigated. The investigation mainly focuses on how accurate the height estimation is and how user friendly it is to calibrate. In order to determine that, a survey was conducted in which individuals had to calibrate each algorithm and then answer to some questions about how they experienced the process.
The first algorithm was a re-implementation of a method available in literature, which measures distance between two parallel planes by finding the ground plane's vanishing line and the vanishing point in the reference direction. The second algorithm estimates the object's world scene's 3D pose by reversing the projection made by the camera when the image was captured. The third and last algorithm investigated, estimates the height of the object by using information about how many pixels a reference object, of know height, occupies when it appears at different locations in the image.
The company, on behalf which this thesis was written, wants to use the information about an object's height in order to build a classifier capable of distinguishing pets from burglars. The algorithm most suitable for that task was simulated to test how sensitive it is to noisy input. Finally that algorithm was recommended to the company.

Referat

Uppskattning av ett okänt objekts höjd i en 2D-bild

I det här examensarbetet undersöktes tre olika algoritmer för att estimera höjden av ett objekt. Undersökningen var i huvudsak inriktad mot att avgöra hur bra estimeringar algoritmerna kan ge, samt hur enkelt det är för en ovan person att kalibrera dem. För att avgöra det genomfördes ett användbarhetsartest med tio försökspersoner som fick testa att kalibrera de olika algoritmerna och svara på några frågor om hur de upplevde den processen.
Den första algoritmen i den här rapporten är en återimplementation av en befintlig metod som räknar ut avståndet mellan två parallella plan genom att hitta markplanets flyktlinje (vanishing line) och referensriktningens flyktpunkt (vanishing point). Den andra algoritmen estimerar ett objekts höjd genom att omvända projektionen som kameran gjorde när bilden togs och på så sätt få fram objektets 3D position i det koordinatsystemet som används för att beskriva verkligheten. Den sista av de tre algoritmerna som undersökts uppskattar objektets höjd genom att använda information om hur många pixlar ett referens objekt, vars höjd är känd, tar upp på olika ställen i bilden.
Företaget för vars räkning detta arbetet gjordes vill använda information om ett objekts höjd för att ta fram en klassificerare som kan klassificera inbrottstjuvar och husdjur. Slutligen så gavs en rekommendation till företaget om vilken algoritm som ansågs mest lämplig för den tillämpningen. Rekommendation gavs efter det att simuleringar hade testat hur känslig algoritmen är för brus i indatan.

Contents

Chapter 1

Introduction

The following sections will briefly introduce the reader to the subject and describe the problem and limitations of this thesis.

1.1 Camera-based surveillance alarm systems

This master thesis took place at a surveillance company located in Stockholm, Sweden. The company is developing a camera-based surveillance alarm system with the purpose to protect apartments and houses from burglary. The customer's local system consists of an alarm unit, a couple of cameras and a control panel which is used to configure the whole system. That system is connected through the internet to the company's servers and in that way a security company is contacted after the alarm is triggered in the case of a burglary. The alarm is triggered if a camera detects a movement after processing it's images in real time.

Nowadays, there are many companies which offer camera alarm surveillance. Nevertheless, the alarms currently available on the market rely additionally on detectors such as IR (Infrared) temperature sensors whereas the alarm system considered within the context of this thesis will only use information from camera images as its input. This allows the company keep their hardware expenses low, on the other hand, the image processing algorithms running on the cameras have to be more robust which makes the development of the appropriate algorithms a more difficult and time-consuming process.

At present, the cameras are able to filter out some kind of movements like shadows and sunlight which should not trigger the alarm. The alarm should neither be triggered by the movement of pets. This is an important problem which has to be solved in order to be able to attract customers living with indoor pets. The company can accept a small amount of false positive triggered alarms but the opposite case i.e. burglars detected as pets, is very critical and has to be avoided.

1.2 Constraints & limitations

The company's main goal is to be able to determine if a moving object is a family pet or a burglar by processing a single 2D (two-dimensional) image taken with a camera as part of a surveillance alarm system. That is a complex classification problem, therefore it was necessary to introduce some assumptions in order to obtain the solutions presented within the present work.

First of all, it is assumed that pets are always distinctly shorter than burglars, i.e. a house with a Saint Bernard dog will not be safe against a remarkably short burglar. It is also assumed that the actual size of the pet is provided by the user.

An additional limitation is that the object is assumed to be standing on the ground. This might not always reflect the reality very well, it is not uncommon that pets like dogs and cats walk around on tables and other furniture. Another case where this assumption is not true is when a burglar is not standing up but is laying or crawling on the floor.

The company has already the appropriate tools to get some information about an object moving in front of the camera such as its speed, direction and location (the two end points of the diagonal of the object's bounding box i.e. the smallest rectangular which can fit the object). Therefore, the way in which such kind of data are extracted from the images will not be considered in this thesis.

In the experiments performed for the scope of the current project, the object's location will be inserted manually to the algorithms.

Since the experiments were done in people's homes, it was not possible to mount the camera on the ceiling. Instead it was placed on top of bookshelves or similar to get as close as possible to the ceiling.

1.3 Problem statement

In general, it is not possible to provide a good estimation of the height of an unknown object in a single 2D image. However, if one has some prior knowledge about the camera and its relation to the environment it is monitoring (for example distance to the floor, the size of its sensor, etc) it is possible to give a better estimation of an object's true height.

Three different approaches of performing such a camera calibration will be investigated and presented in the current thesis in order to examine which one is the most suitable for the alarm system considered here. For a decision to be made, three properties will be compared, namely the quality of estimation, the usability and the time complexity of the estimating process.

In order to assess for the quality of the calibration, an object's estimated height will be compared with its true height. Since the three algorithms uses totally different approaches they need different input from the user in order to be calibrated. The degree of usability of each calibration case will be based on how easy or difficult it is to provide the algorithm with the necessary input. Finally, the time complexity of the algorithms will be calculated theoretically.

Bearing in mind the above remarks, the questions this report aims to answer are which algorithm is most suited to estimate the height of an object standing on the floor from a single 2D image and if any of them is good enough to base an alarm system upon which will keep a house safe from burglars without mistaking them for pets present in the house.

Chapter 2

Theory

The following sections provide to the reader a short introduction to the concepts of projective geometry and focus on the methods and terminology which will be used later on in this report.

2.1 Definitions

Principal axis is the axis in the camera's viewing direction. In this report the principal axis is equal to the camera's y-axis. The principal axis is also called the principal ray.

Principal point is the intersection point between the image plane and the principal axis.

Angle of view is the angle which limits what the camera is capable of projecting. Everything inside the camera's angle of view will be projected if it is not hidden behind another object. A camera has two angles of views, one horizontal and one vertical.

Radial distortion means that straight lines in the world scene occur like bent lines when they are projected onto the image plane. The problem with this phenomenon increases when the camera's focal length decreases. Pinhole cameras do not suffer from this problem.

2.2 Basic single view geometry

2.2.1 Homogeneous points and lines

A point $\mathbf{x}$ in a xy-plane is normally written as $\mathbf{x} = (x, y)$. A *homogeneous* vector representation of a point in the plane is written as $\mathbf{x}' = (x_1, x_2, x_3)$. The relation between $\mathbf{x}$ and $\mathbf{x}'$ is $\mathbf{x} = (x_1/x_3, x_2/x_3)$. It is worth noticing that by using the homogeneous vector notation[1] all vectors (kx, ky, k), $k \neq 0$, correspond to the same point in the plane.

The example above is in 2D but the concept of homogeneous points can be generalized to higher dimensions. In the general case a homogeneous point in N spaces has $N+1$ elements and the last element in the vector is dividing the other elements, and then it is removed, when the vector is transformed back to the Cartesian coordinate system.

Lines can also be expressed as homogeneous vectors. The vector $\mathbf{l} = (a, b, c)$ then corresponds to the line $ax + by + c = 0$. In the same way as $k \cdot \mathbf{x}'$ will result in the same point as $\mathbf{x}'$, the multiplication of $k \cdot \mathbf{l}$ will result in the same line as $\mathbf{l}$. The scalar product $\mathbf{x} \cdot \mathbf{l}$ is equal to zero if $\mathbf{x}$ is a point on $\mathbf{l}$.

As will be shown later in this report homogeneous representations of lines and points tend to be convenient to use and from here on all lines and points in this document will be assumed to be of this form if nothing else is mentioned.

2.2.2 The projection matrix

When a photo is captured by means of a camera, points corresponding to the real world are projected onto an image plane. The image plane is located at distance $-f$ from the camera centre but for convenience it is in literature often thought of as being located at f, with f being the camera's *focal length* (Hartley & Zisserman, 2000). Mathematically, the projection from the real world onto the image plane can be expressed with a matrix-vector multiplication. The vector is the 3D representation of the point in the real world and the matrix is called the *camera projection matrix* and can differ depending on the camera's location and orientation i.e how the coordinate system is defined in relation to the camera.

The $\mathbf{P_1}$ matrix below assumes that the origin of the world coordinate system is located at the camera centre and that the principal axis coincides with the y-axis. It also assumes that the origin of the coordinate system used in the image coincides with the principal point:

$$\mathbf{P_1} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{2.1}$$

Under those assumptions the projection of a point $\mathbf{X}$ onto the image plane has the form: $\mathbf{x} = \mathbf{P_1}\mathbf{X}$. The company's customers will have their cameras mounted on the ceiling with an angle θ (the camera's *tilt angle*) between the ceiling and the camera's principal axis. The floor is defined to coincide with the world's xy-plane for $Z = 0$. The camera's location in the world frame coordinate system will be $(0, 0, H, 1)$, where H is the distance in height between the camera and the floor. In figure 2.1 it can be seen how the coordinate system used in this report is defined. When a picture is taken, points in the physical space are projected in the camera's

[1]The homogeneous coordinates were introduced by August Ferdinand Möbius (1790–1868) in 1827 and are a system of coordinates used in projective geometry. They have the advantage that

coordinate frame and if $\mathbf{X}$ is expressed in world frame coordinates it first has to be transformed into the camera's frame. Such a transformation can be expressed like this:

$$\mathbf{X_{cam}} = \begin{bmatrix} R & -R\tilde{C} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.2}$$

where $\mathbf{R}$ is a 3×3 rotation matrix and $\tilde{\mathbf{C}}$ is an inhomogeneous vector representing the camera's location in the world frame.

Since the only rotation needed for the projections in the experiments performed here is around the x-axis it can be assumed that $\mathbf{R} = \mathbf{R_x}$ and therefore the following matrix is obtained:

$$\mathbf{R_x}(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\ \theta & -\sin\ \theta \\ 0 & \sin\theta & \cos\ \theta \end{bmatrix} \tag{2.3}$$

the coordinates of points at infinity can be represented using finite coordinates.

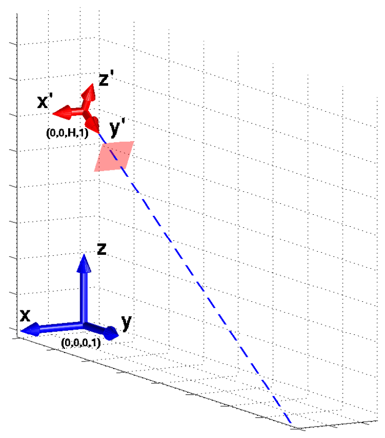

Figure 2.1: *Definition of the world scene's and the camera's coordinate systems. Axis x, y, and z are the world scene's and axis x', y' and z' are the camera's. The rectangle in front of the camera is the image plane.*

Therefore a projection of the world scene coordinate $\mathbf{X}$ onto the image plane is of the form:

$$\lambda \mathbf{x} = \lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathbf{KR}[\mathbf{I}| - \tilde{\mathbf{C}}]\mathbf{X} = \mathbf{PX} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\,\theta & -\sin\,\theta \\ 0 & \sin\,\theta & \cos\,\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -H \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{2.4}$$

where λ is a scaling factor. Notice that the $\mathbf{K}$ matrix is the same as the $\mathbf{P_1}$ matrix but without the last column, which needs to be removed since the projected point will be an inhomogeneous 3 components vector after the transformation to the camera's coordinate frame (Hartley & Zisserman, 2000). Figure 2.2 shows a projection similar to the environment which will be used in this report.
Even though $\mathbf{R_x}$ is the only rotation matrix needed for projections in this report, $\mathbf{R_z}$ is used for another purpose in one of the methods described in section 3.2. $\mathbf{R_z}$ performs a rotation around the z-axis and it is defined as:

$$\mathbf{R_z}(\theta) = \begin{bmatrix} \cos\,\theta & -\sin\,\theta & 0 \\ \sin\theta & \cos\,\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.5}$$

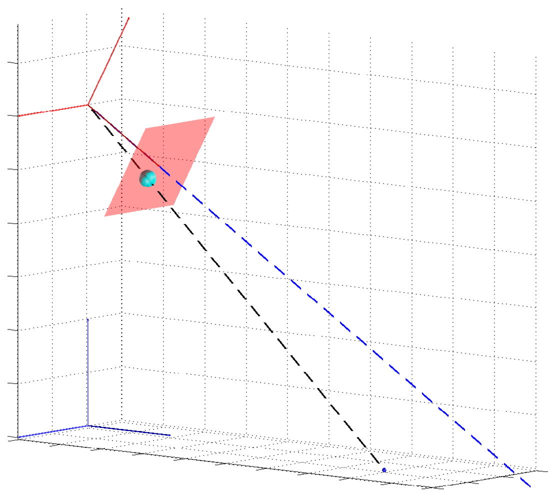

Figure 2.2: *Projection of the point (0.5, 1, 0, 1) in 3D space onto the image plane of a camera mounted at $H = 3$ and with an angle $\theta = 30°$. The upper coordinate frame is the camera's and the lower is the world's. The ray which coincides with the camera's y-axis is the principal axis.*

2.2.3 The concept of vanishing points and lines

Two lines $\mathbf{l}$ and $\mathbf{l}'$ intersect at the point $\mathbf{x} = \mathbf{l} \times \mathbf{l}'$ (the cross product between $\mathbf{l}$ and $\mathbf{l}'$). Two parallel lines always intersect at a point where $x_3 = 0$. If $\mathbf{x} = (x_1, x_2, x_3 = 0)$ then the corresponding point in the plane is $\mathbf{x} = (x_1/0, x_2/0)$. Such a point could be interpreted as being at infinity and is called an *ideal point* or simply a *point at infinity.*

Ideal points and lines are located in the real world and when they are projected onto the image plane they are instead called *vanishing points* and *vanishing lines.*

Since a line can be determined from two known points on the line it is possible to find the vanishing line by finding the intersection points (two vanishing points) of two pair of lines in the image. The two lines in each pair need to be parallel to each other in the real world. All four lines can not be parallel since that would result in only one vanishing point and the line could therefore not be determined uniquely.

To summarize the concept, the vanishing line is the line in which all pairs of parallel lines lying on the same plane in the real world will intersect each other in the image plane (Hartley & Zisserman, 2000). In theory, a set of parallel lines, $\mathbf{L}$, will all intersect at the same point. But if the data are noisy (which could not be avoided during the experiments in this report), different combinations of lines in $\mathbf{L}$ can intersect at different points. In Liebowitz & Zisserman (1998) a successful approach which uses the *Maximum Likelihood Estimation* (MLE) in order to reduce the error from the noisy data was proposed. The method moves the lines in $\mathbf{L}$ with the constraint that all of them have to intersect at a common point $\hat{\mathbf{v}}$ i.e. the estimated vanishing point. The function which is minimized by the MLE is illustrated in figure 2.3 and is defined as follows:

$$\mathcal{C} = \sum_i d_\perp^2(\hat{\mathbf{l}}_\mathbf{i}, \mathbf{x}_\mathbf{i}^\mathbf{a}) + d_\perp^2(\hat{\mathbf{l}}_\mathbf{i}, \mathbf{x}_\mathbf{i}^\mathbf{b}) \tag{2.6}$$

where the $d_\perp^2$ function is the squared perpendicular distances between a point and a

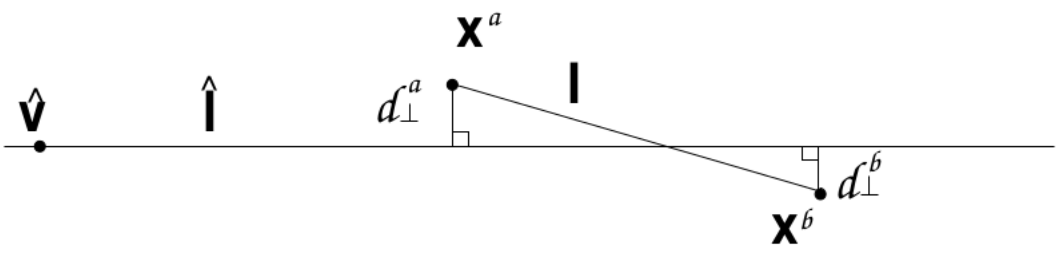

Figure 2.3: *An illustration of the cost function used in equation 2.6. Reprinted from Liebowitz & Zisserman (1998).*

line, $\mathbf{x_i^a}$ and $\mathbf{x_i^b}$ are the two end points of a line $\mathbf{l_i} \in \mathbf{L}$. Minimization of equation 2.6 makes sure that the distance between $\mathbf{L}$ and $\hat{\mathbf{L}}$ (the set of moved lines) is as short as possible.

2.2.4 Measuring the distance between two parallel planes

In Criminisi & Zisserman (1999,2000) it is shown how to calculate the metric distance between parallel planes in a 2D perspective image without any information about the camera. The main idea is to use the vanishing line $\mathbf{l}$ of a reference plane (e.g. a floor in an indoor environment) together with a vanishing point, $\mathbf{v}$, for any direction not parallel to that plane in order to define a cross-ratio (see figure 2.4) between a point $\mathbf{i}$ on $\mathbf{l}$, $\mathbf{v}$ and the two points $\mathbf{b}$ and $\mathbf{t}$. Those points are all located on the image plane and the right hand side of the cross-ratio are the corresponding world scene points. $\mathbf{b}$ is projected from the reference plane and $\mathbf{t}$ is projected from a plane parallel to that reference plane. These points have to be chosen in such a way which makes them all correspond to each other, otherwise it would not be

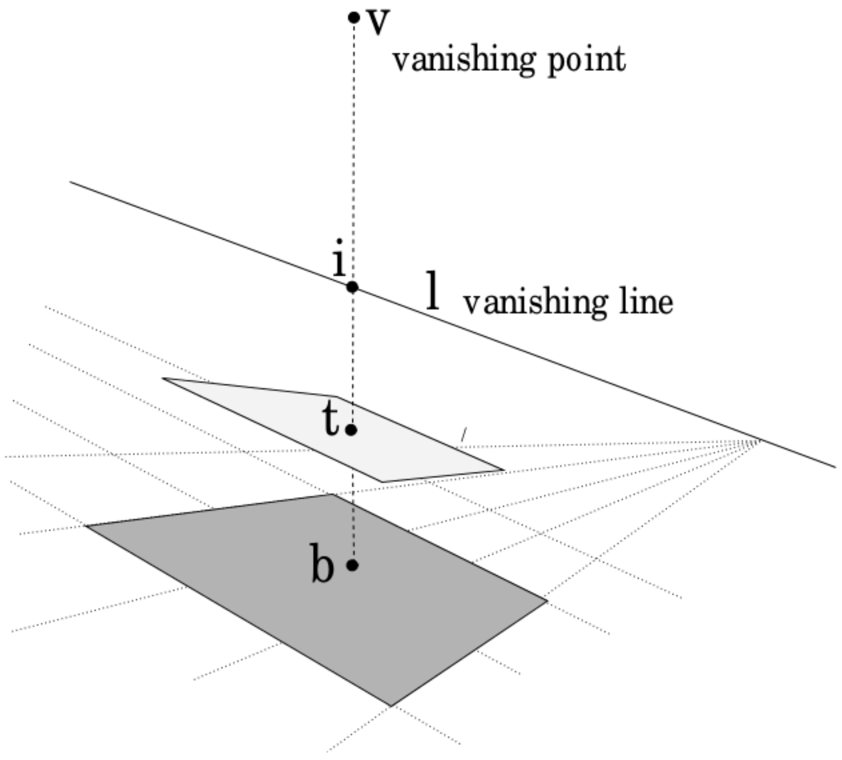

Figure 2.4: *The points that define the cross-ratio. Reprinted from Criminisi & Zisserman (1999,2000).*

possible to define a cross-ratio. The cross-ratio equation is defined as:

$$\frac{d(\mathbf{b},\mathbf{i})\,d(\mathbf{t},\mathbf{v})}{d(\mathbf{t},\mathbf{i})\,d(\mathbf{b},\mathbf{v})} = \frac{d(\mathbf{B},\mathbf{I})\,d(\mathbf{T},\mathbf{V})}{d(\mathbf{T},\mathbf{I})\,d(\mathbf{B},\mathbf{V})} \tag{2.7}$$

where the d-function is the distance between two points and the uppercase letters used in the equation 2.7 are the world scene points of the corresponding lowercase image point.

In section 2.2 it was shown that a point at the vanishing line lies at infinity in the real world, therefore $d(\mathbf{T},\mathbf{V})/d(\mathbf{B},\mathbf{V}) \approx 1$. As can be seen in figure 2.5 the point $\mathbf{I}$ will have the same $\mathbf{Z}$ coordinate as the camera centre. Hence equation 2.7 can be rewritten as:

$$\begin{gathered} \frac{d(\mathbf{b},\mathbf{i})\,d(\mathbf{t},\mathbf{v})}{d(\mathbf{t},\mathbf{i})\,d(\mathbf{b},\mathbf{v})} = \frac{Z_C}{Z_C - Z_T} \Leftrightarrow \\ \frac{Z_T}{Z_C} = 1 - \frac{d(\mathbf{b},\mathbf{i})\,d(\mathbf{t},\mathbf{v})}{d(\mathbf{t},\mathbf{i})\,d(\mathbf{b},\mathbf{v})} \end{gathered} \tag{2.8}$$

where Z_T is the metric distance between the reference plane and the point $\mathbf{T}$ and Z_C is the distance to the camera centre from the same plane. In addition to the cross-ration approach, Criminisi & Zisserman (1999,2000) derived also the algebraic equation 2.9, which calculates the distance between two parallel planes as follows:

$$\alpha Z = \frac{-||\mathbf{b} \times \mathbf{t}||}{(\hat{\mathbf{l}} \cdot \mathbf{b})||\mathbf{v} \times \mathbf{t}||} \tag{2.9}$$

where α is a scale factor, Z is the world $\mathbf{Z}$-coordinate of the plane parallel to the reference plane (the reference plane's $\mathbf{Z}$-coordinate is set to 0), $\mathbf{b}$ is the projection of the point $\mathbf{B}$ from the reference plane onto the image plane and $\mathbf{t}$ is the projection of the point $\mathbf{T}$ from the parallel plane onto the image plane. $\mathbf{B}$ and $\mathbf{T}$ have the same coordinates in the xy-plane. $\hat{\mathbf{l}}$ is a normalized vector representing the reference plane's vanishing line and $\mathbf{v}$ is the vanishing point of the (in this case) $\mathbf{Z}$ direction. Therefore, by using equation 2.9 in order to calculate Z it is possible to determine the real world distance between the two planes (Criminisi & Zisserman, 1999,2000).

However, if α is unknown it is only possible to obtain affine structures of the scene such as parallelism, ratio of areas, ratio of lengths on collinear or parallel lines etc (Hartley & Zisserman, 2000). Consequently, without any knowledge about α it is only possible to measure how Z is related to Z', the distance between the reference plane and another parallel plane, and not the real metric distance between the two planes.
By placing a parallel plane at a point in the image for which the distance to the reference plane is known it is easy to calculate α. That value can later on be used to measure the distance to any parallel plane.

Another approach to determine α is to manually measure the distance between the camera centre and the reference plane (Criminisi & Zisserman, 1999,2000).

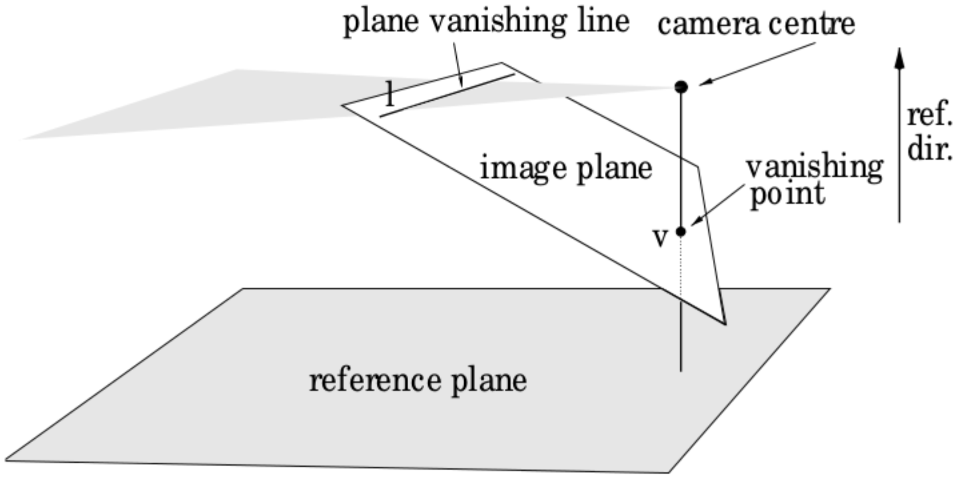

Figure 2.5: *Two geometrical facts.* ***1)*** *The vanishing line is equal to the intersection line between the image plane and the plane which is parallel to the reference plane and is located at the same height as the camera.* ***2)*** *The reference direction's vanishing point* **v** *is the point on the image plane which lies on the line which is parallel to the reference direction and is passing through the camera centre. Reprinted from Criminisi & Zisserman (1999,2000).*

Hence, which method to use depends merely on what knowledge is accessible. The remaining variables in equation 2.9, such as the vanishing line, is in most cases possible to extract from the image itself. Different methods of extracting such information from the images are explained in Hartley & Zisserman (2000).

As mentioned above, equation 2.9 was derived algebraically in Criminisi & Zisserman (1999,2000). A summary of that derivation follows in the coming paragraphs.

The algebraic approach is based on how a projection from the world scene onto the image plane can be done by using a 3×4 projection matrix $\mathbf{P}$, see section 2.2.2. Criminisi & Zisserman (1999,2000) define the world scene's coordinate system in a way which makes the reference plane equal to $Z = 0$ and the origin $\mathbf{o}$ equal to the last column of $\mathbf{P}$. With those constraints they define $\mathbf{P}$ as:

$$\mathbf{P} = [\mathbf{v_x} \; \mathbf{v_y} \; \alpha\mathbf{v} \; \hat{\mathbf{l}}] \tag{2.10}$$

where $\mathbf{v_x}$ and $\mathbf{v_y}$ are the vanishing points in the X and Y direction, respectively.

Therefore if $\mathbf{B} = (X, Y, 0, 1)$ and $\mathbf{T} = (X, Y, Z, 1)$, then:

$$\mathbf{b} = \mathbf{P}\begin{pmatrix} X \\ Y \\ 0 \\ 1 \end{pmatrix} \text{ and } \mathbf{t} = \mathbf{P}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

or equivalently:

$$\mathbf{b} = \rho(X\mathbf{v_x} + Y\mathbf{v_y} + \hat{\mathbf{l}}) \tag{2.11}$$

$$\mathbf{t} = \mu(X\mathbf{v_x} + Y\mathbf{v_y} + Z\alpha\mathbf{v} + \hat{\mathbf{l}}) \tag{2.12}$$

where ρ and μ are two scale factors, μ is unknown, but ρ can be calculated by multiplying equation 2.11 with $\hat{\mathbf{l}}$ which yields:

$$\mathbf{b}\hat{\mathbf{l}} = \rho(X\mathbf{v_x} + Y\mathbf{v_y} + \hat{\mathbf{l}})\hat{\mathbf{l}} = \rho \tag{2.13}$$

In order to understand why only ρ is left after the multiplication, recall that $\mathbf{v_x}$ and $\mathbf{v_y}$ lay on $\hat{\mathbf{l}}$ and hence those two scalar products are equal to zero. The last multiplication $(\hat{\mathbf{l}} \cdot \hat{\mathbf{l}})$ is equal to one since $\hat{\mathbf{l}}$ is normalized. Now equation 2.9 can be obtained by combining equations 2.11, 2.12 and 2.13 as:

$$\begin{aligned}
\mathbf{t} = \mu(X\mathbf{v_x} + Y\mathbf{v_y} + \alpha Z\mathbf{v} + \hat{\mathbf{l}}) &\Leftrightarrow \\
\mathbf{t} = \mu(\alpha Z\mathbf{v} + \frac{\mathbf{b}}{\rho}) &\Leftrightarrow \\
\mathbf{t} \times \mathbf{t} = \mu(\alpha Z\mathbf{v} + \frac{\mathbf{b}}{\rho}) \times \mathbf{t} &\Leftrightarrow \\
\varnothing = \mu\alpha Z(\mathbf{v} \times \mathbf{t}) + \frac{\mu}{\rho}(\mathbf{b} \times \mathbf{t}) &\Leftrightarrow \\
\mu\alpha Z(\mathbf{v} \times \mathbf{t}) = -\frac{\mu}{(\mathbf{b} \cdot \hat{\mathbf{l}})}(\mathbf{b} \times \mathbf{t}) &\Leftrightarrow \\
\alpha Z(\mathbf{v} \times \mathbf{t}) = -\frac{1}{(\mathbf{b} \cdot \hat{\mathbf{l}})}(\mathbf{b} \times \mathbf{t}) &\Leftrightarrow \\
\alpha Z = \frac{-\mathbf{b} \times \mathbf{t}}{(\mathbf{b} \cdot \hat{\mathbf{l}})(\mathbf{v} \times \mathbf{t})} &\Leftrightarrow \\
||\alpha Z|| = -||\frac{\mathbf{b} \times \mathbf{t}}{(\mathbf{b} \cdot \hat{\mathbf{l}})(\mathbf{v} \times \mathbf{t})}|| &\Leftrightarrow \\
\alpha Z = \frac{-||\mathbf{b} \times \mathbf{t}||}{(\mathbf{b} \cdot \hat{\mathbf{l}})||\mathbf{v} \times \mathbf{t}||} &
\end{aligned} \tag{2.14}$$

By using the technique described above, Criminisi & Zisserman (1999,2000) showed that it is possible to estimate the height of a 190 cm tall person with an uncertainty of ±4.1 cm. The uncertainty was reduced to ±2.9 cm by combining three different reference points to obtain a more exact value of α. The authors themselves describe their method as reversing the ancient rules by Leon Battista Alberti on how to draw perspective images. That shows also that some of the concepts used can be traced back to the 15th century.

Given α, it is also possible to determine where the camera centre is located in the real world when the image is taken (Criminisi & Zisserman, 1999,2000). This algorithm is explained further in section 3.1, which focuses more on the implementation details.

2.2.5 Estimating distance to an unknown object

In Yamachika & Hata (2004) a method to generate images which could be viewed from an arbitrary viewpoint is presented. The proposed method differs from other available methods, since it only uses one camera while the most common approaches use stereo vision. The method is divided into tree steps, namely detecting objects which are not part of the original environment, determine the object's 3D pose and finally generate the images.

First of all, the method has some limitations such as the distance from the camera center to the floor needs to be known as well as the camera's focal length and the size of the camera's image sensor (which in digital photography corresponds to the camera film). Additionally, it is assumed that all objects are located on the floor and that they are (or can be approximated as) rectangular parallelepipeds. The last assumption is not necessary for the purpose of this thesis.

Yamachika & Hata (2004) performed experiments to determine the quality of the depth estimation. In the worst documented case the depth error was measured to 22.9%, but in most cases presented in the report, the error was less than 10%.

2.2.6 Improving depth estimation by using semantic classes

In Liu & Koller (2010) a method to estimate the depth of each pixel in a 2D image is proposed. Each pixel is labeled as one of the following eight semantic classes: grass, sky, tree, road, foreground object, water, building and mountain. Then one learner for each class is trained to estimate the depth for the pixels within that class. In the last step, the knowledge from the relation between the different classes and result from the learners is used together with a *Markov Random Field* (MRF) in order to obtain a better estimation. As an example of knowledge about the relation between classes could be that the sky is in most pictures more far away than the grass. Using this method, Liu & Koller (2010) achieved state-of-the-art performance. The method was only tested on outdoor images and is an extension of Saxena & Ng (2005) and Saxena & Ng (2008) in which semantic classes were not used.

Chapter 3

Method

In this thesis three different approaches to measure the height of an unknown object were investigated. In the following sections the three approaches will be explained in detail.

3.1 Algorithm 1 - Distance between parallel planes

The first algorithm in this thesis was first presented in Criminisi & Zisserman (1999,2000) and is also briefly explained in section 2.2.4 of the thesis. Compared to section 2.2.4, which focuses on the theory behind the algorithm, this section will focus more on the implementation details. When specific implementation details about this algorithm are mentioned in the following sections, they all refer to how the algorithm was implemented in order to perform the investigations presented in the present thesis. For example, Criminisi & Zisserman (1999,2000) most likely had another implementation approach which might have some impact on the result.

3.1.1 Finding the vanishing line

A plane's vanishing line is the line where all parallel lines on the plane in the real world will intersect in the image plane, as mentioned in section 2.2.3. The floor of a room can be thought of as a plane and in order to calibrate this algorithm the floor's vanishing line needs to be known.

The vanishing line can be extracted from the image itself if two pairs of lines which are both located on the floor and are parallel to each other in the real world can be found. If no such lines can be found, they can easily be introduced to the camera scene during the calibration process, for example a rectangular paper or magazine can be placed on the floor in front of the camera.

At the beginning of the calibration process, an image is shown to the user with the option to click on two points from each of the four parallel lines. The obtained eight points are located in the image plane's coordinate frame, where the origin $(0, 0)$ is the upper left corner of the image. Two points, (x_1, y_1) and (x_2, y_2), are enough to uniquely determine a line which can be represented with a homogeneous vector as:

$$\mathbf{l} = (y_1 - y_2, x_2 - x_1, x_1 y_2 - x_2 y_1) \tag{3.1}$$

When all four lines are known the next step is to find two points on the vanishing line. From the way we extracted the lines it is known that each pair of lines are parallel in the real world, so there only remains to find the intersection point, **c**, between those lines on the image plane. The intersection point **c** of two lines is equal to the cross product of the two lines and is hence defined as:

$$\mathbf{c} = \mathbf{l} \times \mathbf{l}' = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ l_1 & l_2 & l_3 \\ l'_1 & l'_2 & l'_3 \end{vmatrix} \tag{3.2}$$

Therefore by using equation 3.2 to find the two intersection points from the two pair of parallel lines two points on the vanishing lines are known. When those two points are known equation 3.1 can be used once again and then the vanishing line is found. Notice that the intersection point from equation 3.2 is a homogeneous vector while the points used in equation 3.1 are inhomogeneous vectors. How those two notations are related is discussed in section 2.2.

3.1.2 Finding the vanishing point in the reference direction

The next step involves finding a vanishing point and it is similar to the previous step described in section 3.1.1. The major difference is that this step spreads over both the calibrating part and the measuring part of the algorithm. The reason for this is that the two lines used to determine the vanishing points are the reference object (from the calibrating part) and the object one wants to measure the height of (from the measuring part). In theory, the same vanishing point will be derived between different measurements even if different objects are measured. Since though the lines are obtained from user input, they will contain some degree of noise and therefore the vanishing point will differ slightly from one measurement to another even if the same calibration procedure is used.

3.1.3 Obtaining the metric height

In Criminisi & Zisserman (1999,2000) two approaches to get the metric height are presented, the cross-ratio (equation 2.7) and the algebraic (equation 3.3) one. Both approaches are similar in the sense that they need the same known quantities as inputs. The implementation used in this report is the algebraic one. By using the algebraic approach the height can easily be calculated at this stage by putting the information from the above steps (section 3.1.1-3.1.2) into the equation below:

$$\alpha Z = \frac{-||\mathbf{b} \times \mathbf{t}||}{(\hat{\mathbf{l}} \cdot \mathbf{b})||\mathbf{v} \times \mathbf{t}||} \tag{3.3}$$

During a measurement the equation above is used twice, first to calculate α by substituting Z for the metric height of the reference object and later when α is known to calculate Z. $\hat{\mathbf{l}}$ is equal to the reference plane's vanishing line $\mathbf{l}$ but normalized, i.e. $\hat{\mathbf{l}} = \mathbf{l}/||\mathbf{l}||$. The image coordinates of $\mathbf{b}$ and $\mathbf{t}$ are both known from user input. The theory and notation used in equation 3.3 are explained in section 2.2.4.

3.2 Algorithm 2 - Back projection

The second algorithm in this thesis uses information about the camera and its relation to the world scene. The information needed is the distance between the camera center and the floor, the camera's tilt angle, the camera's focal length and the size of its CMOS[1] sensor. If one knows the height of a reference object in a picture taken from the same setup, then one of the other parameters can be calculated and hence be unknown from the beginning.

The main idea with this method is to reverse the projection made by the camera from the world scene onto the image plane. From the back projection it is possible to retrieve the projected points original 3D position. This is feasible from the assumption that the object is standing on the floor. Without that assumption it would only be possible to find the line, $\mathbf{l_p}$, which intersects both with the camera center and the original position and not the original position itself. Even with the standing on the floor assumption it is essential to find that line. Note that this method is not using homogeneous notations for points and lines. A more detailed explanation of the method follows.

3.2.1 Finding the line $\mathbf{l_p}$

A line in 3D space can be expressed using the scalar parametric equations below:

$$\begin{cases} x = x_0 + a \cdot t \\ y = y_0 + b \cdot t \\ z = z_0 + c \cdot t \end{cases} \qquad (-\infty < t < \infty) \tag{3.4}$$

where (x_0, y_0, z_0) is a point on the line, t is the parameter and $\mathbf{r} = (a, b, c)$ is a vector with the same direction as the line (Adams, 2006).

As mentioned in section 2.2 the camera is mounted in such a way that the camera centre is located at $\mathbf{T} = (0, 0, H)$ with a tilt angle θ. Therefore, $\mathbf{T}$ is a point on $\mathbf{l_p}$ and hence only $\mathbf{r}$ remains to be found and by adding the limitation $||\mathbf{r}|| = 1$, the solution obtained will be unique.
It is worth noticing that if $\theta = 0$ and the projection of the original 3D point coincides with the principal point, then $\mathbf{r} = (0, 1, 0)$. Let us label that solution $\mathbf{r_s}$ (see also figure 3.1) and use it as a starting point to find $\mathbf{r}$ in the general case. In the general case, $\mathbf{r_s}$ is rotated with an angle α around the x-axis and with an angle β around the z-axis. If the projected point is equal to the principal point, then $\alpha = \theta$ and

[1]Complementary metal oxide semiconductor

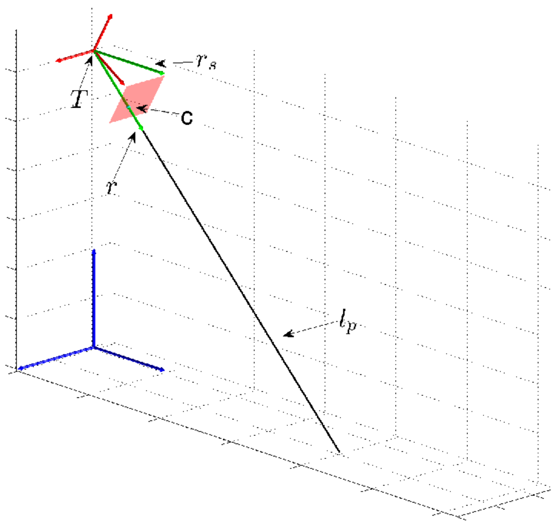

Figure 3.1: *Some of the quantities used in algorithm 2.* **T** *is the camera's location expressed in the world scene's coordinate frame,* **c** *is the coordinates of the point to be back projected,* $\mathbf{r_s}$ *is the direction used when all angles is equal to* 0*,* **r** *is the direction* **c** *from* **T** *and* $\mathbf{l_p}$ *is the line passing through* **T** *with the direction* **r**.

$\beta = 0$. If the principal point is thought of as the origin of the image's coordinate system and the projected point as the coordinates $\mathbf{c} = (x_p, y_p)$ in that frame, then it is possible to estimate the two angles by using trigonometry. The distance between the camera and the image plane is equal to the focal length f and the distance from the principal point to an edge on the CMOS sensor is $S/2$. Where S is either the width S_x or the height S_y of the sensor.

Using these facts one can derive the following formula which calculates the camera's angle of view as:

$$\gamma = 2 \cdot \arctan(\frac{S}{2f}) \tag{3.5}$$

By using $S = S_x$ one gets the horizontal angle of view γ_h and by using $S = S_y$ the vertical angle of view γ_v is calculated. The geometry behind the horizontal case is shown in figure 3.2. At the principal point $\beta = 0$ and in the general case it will be: $-\gamma_h/2 \leq \beta \leq \gamma_h/2$.

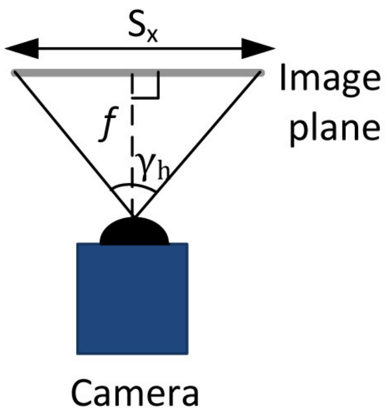

Figure 3.2: *Quantities used to calculate γ_h (*top view*).*

If the image has the resolution $R_x \times R_y$, then β can be estimated from the equation below:

$$\beta = -\frac{x_p}{\frac{R_x}{2}} \cdot \frac{\gamma_h}{2} = -\frac{x_p}{R_x}\gamma_h \tag{3.6}$$

The ratio $-x_p/R_x/2$ estimates how many percentages of half the horizontal angle of view β corresponds to.

The angle α can be estimated in the same way but since the camera also has a tilt angle θ it needs to be added to the formula as:

$$\alpha = -\frac{y_p}{\frac{R_y}{2}} \cdot \frac{\gamma_v}{2} + \theta = -\frac{y_p}{R_y}\gamma_v + \theta \tag{3.7}$$

In the case where α and β are known, the last thing needed in order to estimate $\mathbf{l_p}$ is to rotate $\mathbf{r_s}$. The rotation of a vector is shown in section 2.2.2 therefore only the equation on how to obtain $\mathbf{r}$ is presented here:

$$\mathbf{r} = \mathbf{R_x}(\alpha)\mathbf{R_z}(\beta)\mathbf{r_s} \tag{3.8}$$

3.2.2 Retrieve the 3D position

If $\mathbf{r}$ is known one can easily substitute the variables derived above into equation 3.4 and thereby obtain $\mathbf{l_p}$. It is now the assumption that the object is standing on the floor comes in handy, since that means that the point in the real world has its Z coordinate equal to zero. That, in turn, means that it is possible to find the value of the parameter t in equation 3.4 and then also the X and Y coordinates. That is one way of obtaining the original 3D position of a point projected onto the image plane if at least one of its original coordinates (in this case $Z = 0$) is known from the beginning.

The purpose of this method is not to determine the 3D position, but to measure the height of an object even though this method can also been used to obtain that. By repeating the whole procedure, but instead of using the point on the floor the point for which we want to know the height of is used, the height of an object can be calculated. That means that the Z coordinate is unknown but on the other hand the X and Y coordinates are the same as for the point on the floor and hence one of them can be used to obtain Z.

If one of the variables needed to calculate Z is unknown it is possible to perform the above method for an object of known height. In the end, Z can be substituted for the known value and equation 3.4 can then be solved for the unknown variable which can be used to calibrate the system for later usage. Such an unknown variable would most likely be the tilt angle or the camera's height above the floor.

3.2.3 An alternative way of finding the line $\mathbf{l_p}$

It should be mentioned at this point that there exists a method to determine the line $\mathbf{l_p}$ exactly from the known quantities described in section 3.2.1. This method is standard in literature, Hartley & Zisserman (2000), and is expressed in an equation as:

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \mathbf{T} + \lambda \mathbf{R_x^{-1}} \mathbf{P^{-1}} \mathbf{c} \tag{3.9}$$

where $\mathbf{T}$ is the camera's location, λ is the line's scaling factor (similar to t in section 3.2.1), $\mathbf{R_x^{-1}}$ and $\mathbf{P^{-1}}$ is the inverse of the rotation and the projection matrix respectively and $\mathbf{c}$ is the coordinates of the point to be back projected. This approach to obtain $\mathbf{l_p}$ is also described in Morvan (2009) and is from now on referred to as *exact back projection.*

3.3 Algorithm 3 - Weighted reference objects

The basic idea of the third algorithm in this thesis is to use five reference objects which are positioned like the letter **X**, see figure 3.3. The corner objects should be placed as close to the corners of the picture as possible. By using two corners and the middle reference object one can construct four different triangles. Those triangles are hereby referred to as the upper, lower, left and right triangle. When one wants to know the height of an unknown object in the image the first step is to determine in which triangle the object is located. The next step is to calculate how much weight each reference object in that specific triangle should have. The weighting is based on the pixel distance between the unknown object and a reference object, the object furthest away from the unknown object will get the smallest weight. The third and last step in this method is to use the information obtained above to approximate the metric height of the unknown object. The approximation is, except

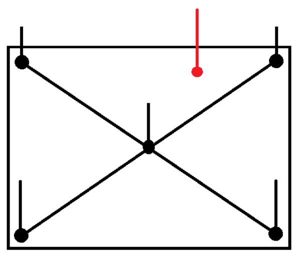

Figure 3.3: *The dot in the middle and the four dots in the corners are locations of the reference objects. The remaining dot is an unknown object. The lines raising from the dots indicates how many pixel the object occupies.*

from the weights, based on the vertical number of pixels the objects occupies. The only calibration needed for this method is the metric height and the locations of the reference objects. In this case location means two points of the object, one at the bottom and one at the top. Below follows a more detailed explanation of the three steps.

3.3.1 Determining in which triangle the unknown object is located

The problem of determining in which of the four triangle an object is located can be solved by drawing two lines in the picture. One line from the upper left corner to the lower right corner and one line from the lower left corner to the upper right corner. By thinking of the pixels in the picture as points in a coordinate system it is easy to write those lines on the form of the standard equation of a straight line as:

$$y = k_i x + m_i \tag{3.10}$$

where i is the number of the line. When k_i and m_i are calculated it is possible to determine if the point $\mathbf{x} = (x_0, y_0)$ is positioned above or below the i-th line. To do that, one only has to insert the x and y coordinate of $\mathbf{x}$ into the equation of the i-th line. If the $\mathbf{x}$ is not on the line the equation is not correct and has to be rewritten as an inequality. One can divide the inequality into two cases, the first of the form: $y \geq k_i x + m_i$ and the second of the form: $y < k_i x + m_i$. In the former case $\mathbf{x}$ is located on or above the i-th line while in the later case it is located below. An example on how to determine if $\mathbf{x}$ is in the left triangle follows. If $\mathbf{x}$ is in the left triangle it means that the point is below the first line and above the second one. Notice that in order to assure that the point lies truly inside the triangle, it is also necessary to check on what side of the line between the two left corners $\mathbf{x}$ the point is located. In this method, the corners are located close to the corner of the picture and it is therefore assumed that x is located on the right side of that line.

3.3.2 Calculation of the weights

From the previous step described in section 3.3.1 it is known which three references that consist the corners of the surrounding triangle. Those three objects will all be used in the approximation step and hence need to be weighted. As mentioned before the weighting is based on the pixel distance between the unknown object and the corners. The pixel distance D between two points (x_0, y_0) and (x_1, y_1) is calculated by using Pythagoras's theorem:

$$D = \sqrt{(x_0 - x_1)^2 + (y_0 - y_1)^2} \tag{3.11}$$

In order to achieve the desired result the weighting process has to satisfy two conditions, the total sum of the weights has to be equal to 1 (normalized) and $w_i > w_j$ if $D_i < D_j$. The last condition means that if the distance to reference object i from the unknown object is less than the distance to object j, then i's weight has to be bigger than j's weight. One approach to satisfy those two conditions is to first calculate the total distance as:

$$D_{total} = \sum_{i=1}^{3} D_i \tag{3.12}$$

Then each weight can be calculated as:

$$w_i = \frac{D_{total} - D_i}{2D_{total}} \tag{3.13}$$

which makes sure that the second condition is true.

Also the first condition holds since:

$$\sum_{i=1}^{3} w_i = \frac{D_{total} - D_1 + D_{total} - D_2 + D_{total} - D_3}{2D_{total}} = \frac{2D_{total}}{2D_{total}} = 1 \tag{3.14}$$

3.3.3 The metric approximation

The last step of this method starts by approximating how many pixels in the vertical direction the reference object would occupy if it was located at the same position as the unknown object. To do that each reference object's pixel height P_i is multiplied with its weight.

The sum of those multiplications is then the estimated pixel height P_{ref} as:

$$P_{ref} = w_1 P_1 + w_2 P_2 + w_3 P_3 \tag{3.15}$$

The last estimation, which will result in this method's output, uses information about the relation between P_{ref}, $P_{unknown}$ and the reference object's real metric height H_{ref}. One way of expressing such a relation is as follows:

$$H_{unknown} = \frac{P_{unknown}}{P_{ref}} H_{ref} \tag{3.16}$$

3.4 Radial distortion

As can be seen in figure 3.4, the camera provided by the company suffers from radial distortion. The algorithms described in this chapter do not correct that distortion and it can therefore have impact on the results.

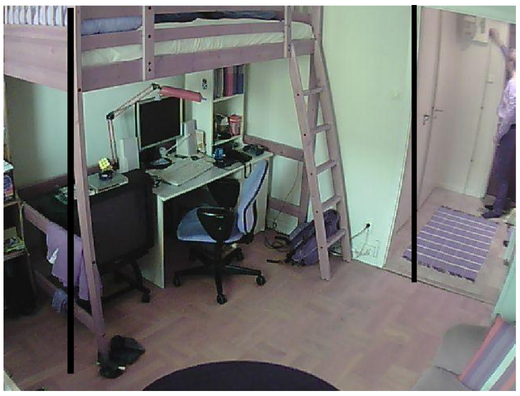

Figure 3.4: *The camera used for those experiments has some degree of radial distortion. The distortion is most apparent from the left side of the door in the middle of the room and from the bed's left leg closest to the camera. The two black lines indicates where the straight lines should be.*

3.5 Measurement points

The described algorithms all takes two points on the image as input, one where the object is touching the floor and one on top of the object. The two points are assumed to have the same X and Y coordinates in the real world. If one of the algorithms will be integrated into the company's system those points will be taken from the object's bounding box. The shape of the object's bounding box can change depending on how the object is placed in relation to the camera (as can be seen in figure 3.5), that will result in some error. For some breed of dogs, like a dachshund, the shape of the bounding box can change even more.

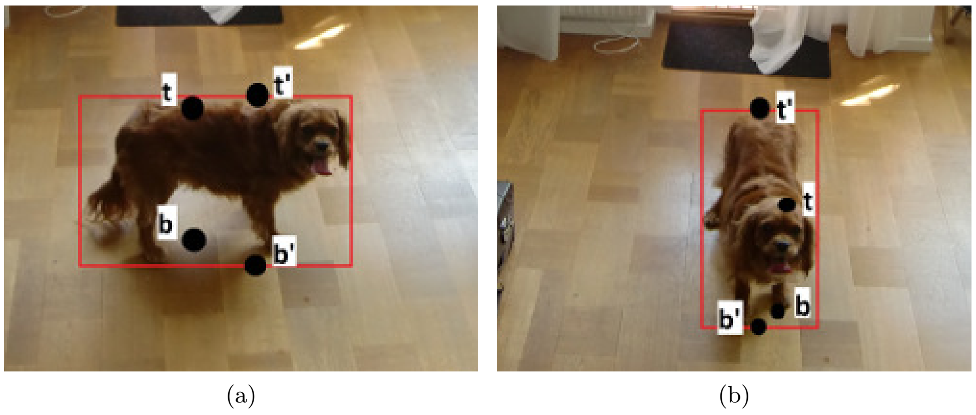

Figure 3.5: ***a)*** *A bounding box for a dog at a specific moment.* ***b)*** *The same dog but in another position and orientation which changes the shape of the bounding box. Points b and t should be used as input, but by using the bounding box as input, the points b' and t' will be used instead.*

Chapter 4

Results

In the following sections a detailed explanation on how the investigation was performed and what it resulted in will be presented.

4.1 Evaluation

The following section considering the evaluation of the algorithms, is divided into four parts, namely quality of measurement, usability and time complexity. At the end of the section a recommendation based on the evaluation procedure will be made on which algorithm should be preferred for real life applications. The evaluation of the algorithms in the current report was performed by means of a survey. The survey involved ten people (referred to as *test persons* for the remaining parts of the thesis) who calibrated the system three times, one for each algorithm. The test persons were guided on what to do during the calibration process but they had to do all steps by themselves. See appendix A for more details on what the survey contained. The results of the evaluations are in section 4.2. In addition to the survey, simulations were made to investigate how the accuracy of algorithm 2 was affected from that the line $\mathbf{l_p}$ was estimated instead of being calculated exactly. The results from the simulations are presented in section 4.4.

4.1.1 Quality of measurement

In order to assess for the accuracy of each algorithm, ten set ups were calibrated by the test persons. The reason why the calibration was done by others and not by the author of this thesis is simply because if one of the algorithms will be used by the company in real life applications, the calibration will be done by inexperienced people and not by the developer of the algorithms. Therefore the result of this report reflects the outcome the company should expect if the algorithms presented in this thesis are integrated in the company's alarm systems which will have to be calibrated by the customers themselves.

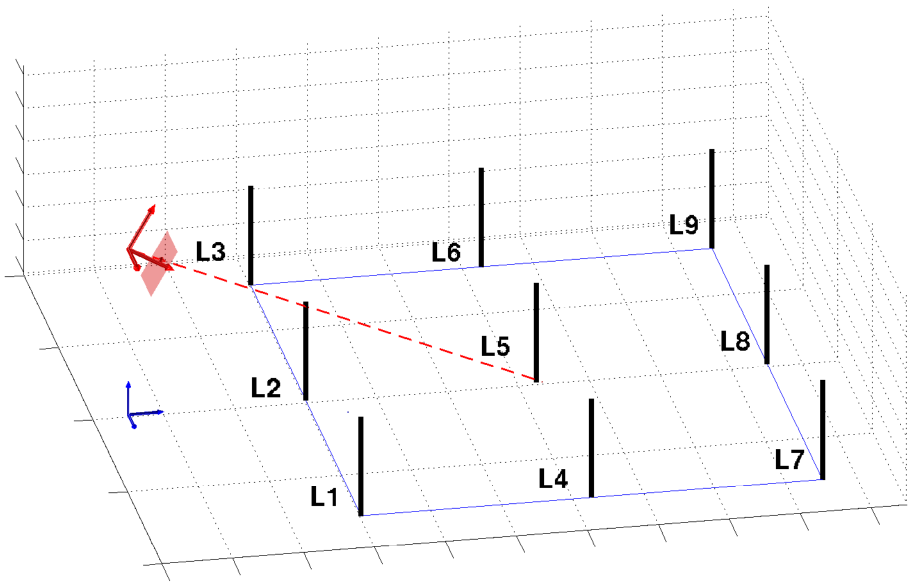

Figure 4.1: *The result in this report is based on measurements from nine locations as shown here. The blue square is the part of the floor which the camera is capable of projecting. In those cases the environment did not allow some of the locations, there might be furnitures or something similar which blocks the view, they were ignored.*

For each set up, three different objects of significantly different heights (40 cm, 100 cm and 180 cm) were placed at nine different locations where the camera could see both the bottom and the top side of them. As long as the available space in front of the camera allows it, the locations were picked according to figure 4.1. A total number of 642 measurements were obtained during the survey. All errors presented in the sections below are absolute percent errors on the form:

$$\textbf{error} = \left|\frac{h - \hat{h}}{h}\right| \cdot 100 \tag{4.1}$$

where h is the true height and $\hat{h}$ is the estimated height.
The same pictures were used for all three algorithms. All pictures used to obtain the result in this thesis were taken in a way similarly to figure 4.2.

4.1.2 Usability

One of the main aims of the current thesis was to evaluate the calibration usability of each one of the proposed algorithms. All the test persons graded each algorithm on a scale from one to five, where one corresponds to "almost impossible" to calibrate

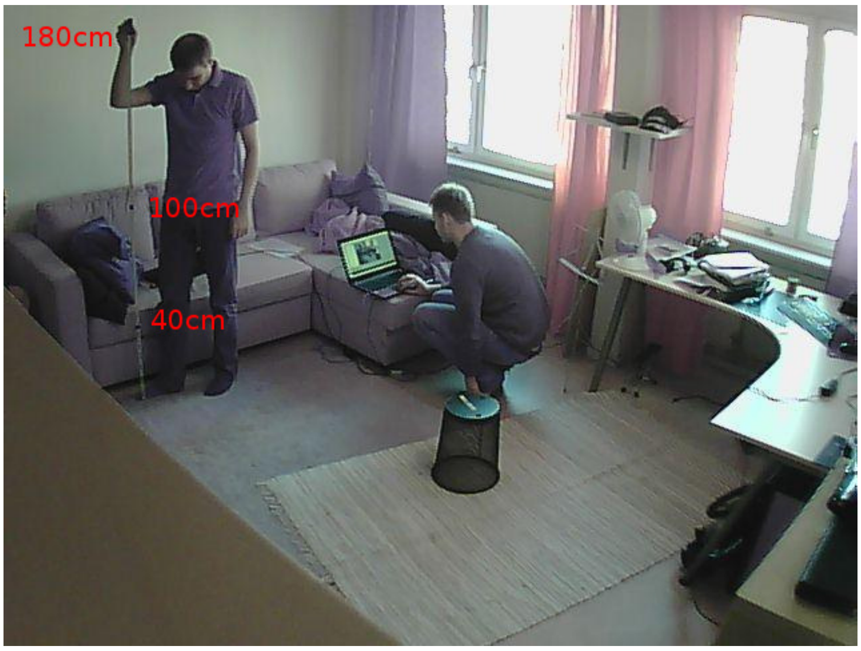

Figure 4.2: *One of the pictures used to investigate the quality of the algorithms. As can be seen three different heights were measured, 40 cm, 100 cm and 180 cm. The first two heights were marked in black on a measuring tape and the third height is equal to the upper end point of the tape.*

while five means "very easy" to calibrate. In addition to that, questions were asked about which part of the calibration for a specific algorithm was most difficult and so on. Since algorithm 3.2 can be calibrated in three different ways the survey investigated also which was the most preferable way of those available.

Note that the usability investigation was focused on how easy or how difficult it was to calibrate the algorithms in terms of measuring the distance between the camera and the floor, draw parallel lines, etc. That means that no attention was paid on how well the users interacted with the GUI[1].

4.1.3 Time complexity

Another factor which was investigated in this report was the time complexity of the three presented algorithms. The time complexity investigations were not considering the calibrating part of the algorithms since that procedure only has to be done once and not in real time. The test persons from the survey were not involved in this part of the evaluation procedure.

[1]Graphical user interface

4.1.4 Simulations

The simulations were mainly performed in order to investigate how the quality of the algorithm's measurements were affected by estimating the line $\mathbf{l_p}$ instead of calculating it exactly like in Hartley & Zisserman (2000). The simulations also investigate how the accuracy is affected by noise in the input. Three variants of algorithm 2 were simulated. The first one is estimating $\mathbf{l_p}$ similarly to section 3.2.1. The second one is the same as the first one, but instead of using the simulated environment's exact tilt angle it obtains that angle by using a reference object of known height. The third one is calculating $\mathbf{l_p}$ as in Hartley & Zisserman (2000).

The parameters for the simulations were set to emulate the environments in the survey as well as possible. Therefore the tilt angle and the camera's distance to the floor were set to average values of those environments to 26.4° and 206.1 cm. The camera's internal parameters such as the focal length and the size of the CMOS sensor are the same as for the company's cameras.

The three different simulated variants of algorithm 2 are all measured at nine different locations and on three objects of the same height as in the survey. The nine locations were at 300 cm, 500 cm and 700 cm world scene's y-axis and at 180 cm, 0 cm and -180 cm on the x-axis.

In order to investigate how sensitive to noise algorithm 2 is, after it is calibrated, Gaussian noise was added to the pixels used as the measured points. The Gaussian distributions used in the simulations have the true pixels as their mean μ and standard deviation σ. Simulations were made for different values of σ. All simulated data presented in this report are based on the average of 100000 measurements where noise from the same distribution was added.

4.2 Survey

In this section the data from the survey will be presented.

4.2.1 Quality

Figures 4.3-4.6 illustrate how well the three algorithms were performed at different locations and for different heights. The y-axis shows the estimated height while the x-axis corresponds to the nine different locations shown in figure 4.1. To increase the readability of the figures the results from the test persons are grouped around their location with some small space between each data point.

An estimated height equal to zero means that it was not possible to measure at that location for a specific setup. For some measurements with large errors it is deem difficult to observe which height the measurement is supposed to estimate, e.g. if the measured object's true height is 40 cm or 100 cm. But since for all measurements the estimated value for 40 cm is always less then the estimated value for 100 cm which, in turn, always is less then the estimated value for 180 cm, that is not a problem in our case. From the setup used and calibrated by person 8 it was

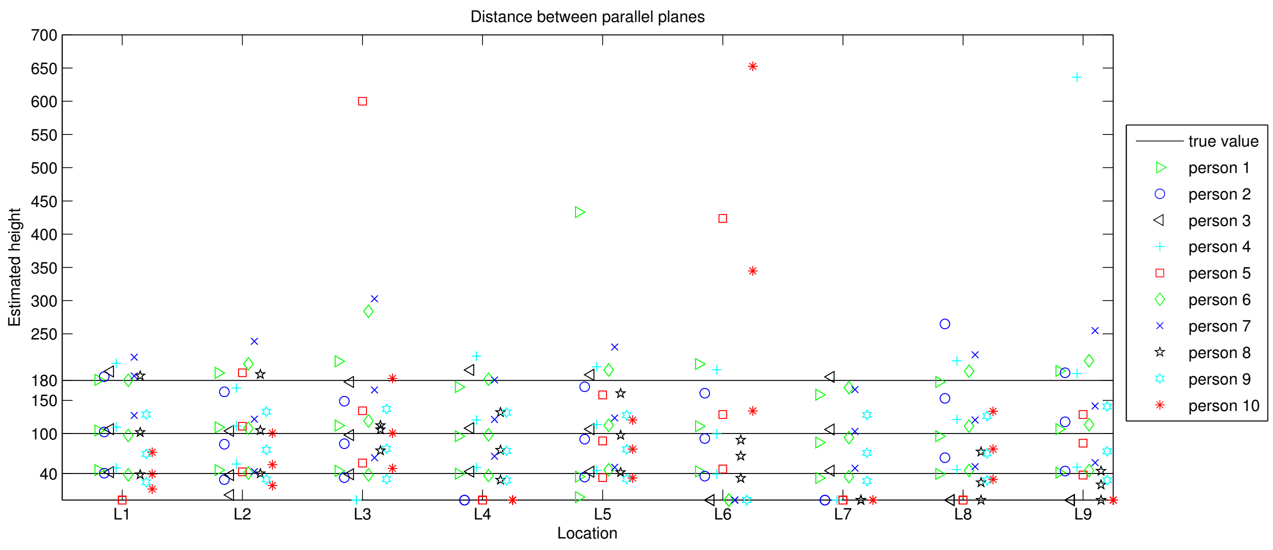

Figure 4.3: *This figure shows the result from the survey when algorithm 1 was used.*

not possible to do any measurements for objects of 180 cm height at locations $L7$, $L8$ and $L9$ and they are therefore all set to zero in that case.

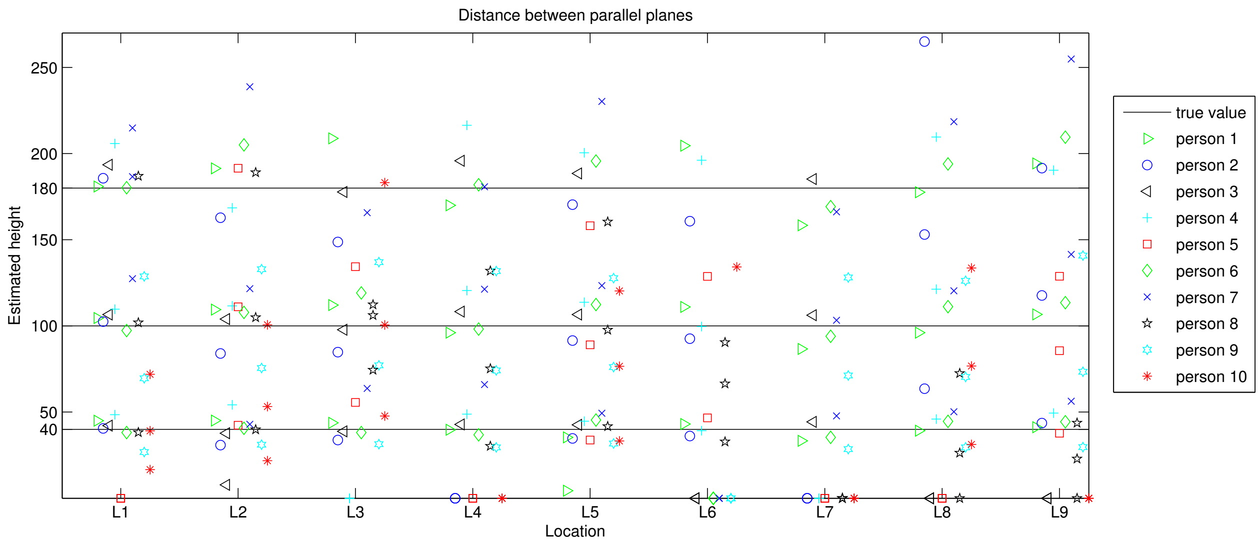

Figure 4.4: *This figure shows the same results as figure 4.3, but in this one some of the outliers were removed in order to show the rest of the data in more detail.*

In figure 4.3 it can be observed that algorithm 1 sometimes gives inaccurate estimations, e.g. the estimation at location 6 for person 10 returns a value greater than

Calibration	**180 cm**	**100 cm**	**40 cm**
Person 1	22.7	17.9	8.5
Person 2	14.3	17.2	18.7
Person 3	20.1	5.6	6.4
Person 4	47.3	23.9	18.7
Person 5	83.1	19.9	16.6
Person 6	14.0	9.3	8.3
Person 7	27.4	35.5	57.4
Person 8	22.3	19.7	26.8
Person 9	26.9	26.6	25.5
Person 10	71.2	66.6	66.3

Table 4.1: *The average percent error for each test person's calibration of algorithm 1. Each calibration were tested on three different objects of heights 180 cm, 100 cm and 40 cm.*

650 cm while the true height is 180 cm. On the other hand, one can note that the same person has a spot on result at location 3 for the same true height. Therefore, even within the same calibration the result can fluctuate a lot. The outliers in figure 4.3 make it hard to see how well the more accurate measurements correspond to their true values, therefore they were removed in figure 4.4 where it is observed that none of the test persons managed to calibrate the system accurately at all locations. For example, person 1 obtained good results at all locations except from $L5$, where the algorithm fails and returns a value greater than 400 cm (see figure 4.3) when it was supposed to return 180 cm. Table 4.1 shows that test person 10 has an average percent error around 70 % for all of the three different heights, that is the worst result among all the calibrations and all the algorithms investigated in this survey. From table 4.2 it seems like the algorithm cannot provide an accurate estimation at location $L6$. In figure 4.3 it is evident that the spurious results from person 10 have their greatest impact at location $L6$. If the results from person 10 are removed then the obtained average error for $L6$'s row would be: 43.7 %, 16.2 % and 10.8 %, which is not remarkable.

Location	**180 cm**	**100 cm**	**40 cm**
L1	15.3	22.9	39.9
L2	24.6	15.8	16.0
L3	50.8	19.8	28.6
L4	12.8	15.2	21.9
L5	28.4	22.1	13.7
L6	80.1	54.3	48.2
L7	11.5	11.5	17.5
L8	21.4	23.8	24.4
L9	53.7	33.4	20.2

Table 4.2: *The average percent error for algorithm 1 at each location. Each calibration were tested on three different objects of heights 180 cm, 100 cm and 40 cm.*

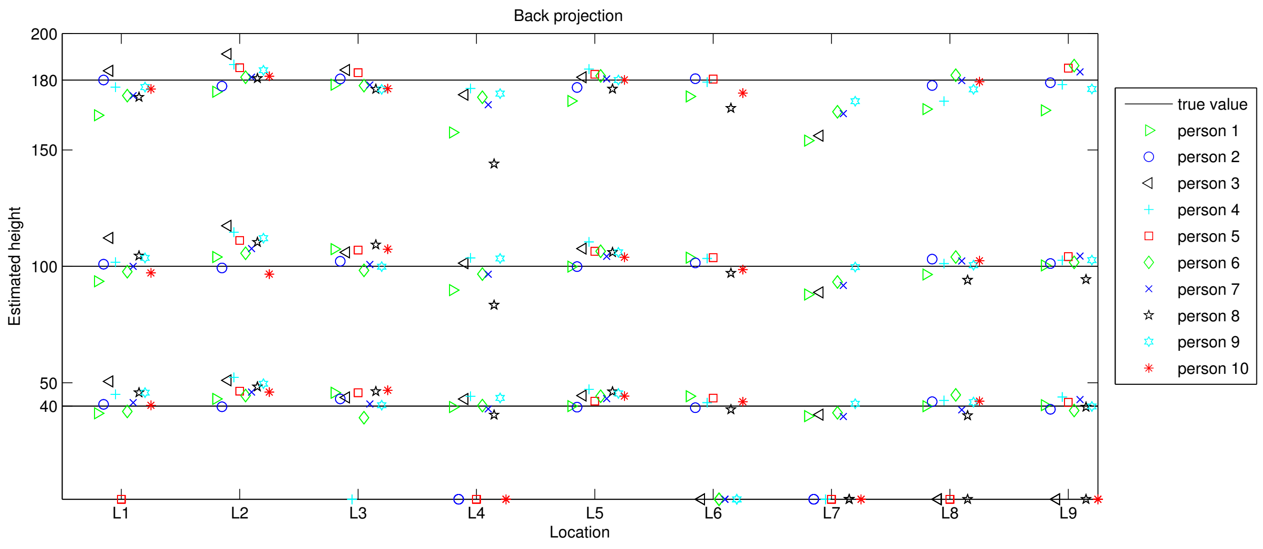

Figure 4.5: *The result from the survey when algorithm 2 was used.*

In figure 4.5 and table 4.4 it is observed that algorithm 2 in general gives accurate results. However, it can also be seen that at locations *L*4 (middle right) and *L*7 (back right) the algorithm returns a value slightly below the desired one. The same locations on the left hand side are more accurate. In table 4.3 the data shows that person 2 has an average percent error less than 4 % for all three heights, that is the best result achieved in this report.

Calibration	**180 cm**	**100 cm**	**40 cm**
Person 1	6.9	5.3	5.9
Person 2	0.8	1.4	3.1
Person 3	4.7	9.3	15.2
Person 4	2.4	5.4	13.0
Person 5	1.8	6.5	9.6
Person 6	2.9	4.0	8.1
Person 7	2.7	3.9	6.8
Person 8	5.9	7.6	11.4
Person 9	2.4	3.6	8.6
Person 10	1.4	3.5	8.9

Table 4.3: *The average percent error for each test person's calibration of algorithm 2. Each calibration were tested on three different objects of heights 180 cm, 100 cm and 40 cm.*

Location	**180 cm**	**100 cm**	**40 cm**
L1	3.1	3.8	9.7
L2	2.2	8.7	16.9
L3	1.6	4.7	10.4
L4	7.3	6.0	5.8
L5	1.5	5.1	9.4
L6	2.5	2.8	5.5
L7	9.7	7.7	8.2
L8	2.5	2.9	5.8
L9	2.8	2.9	3.9

Table 4.4: *The average percent error for algorithm 2 at each location. Each calibration were tested on three different objects of heights 180 cm, 100 cm and 40 cm.*

Table 4.6 and table 4.7 both indicates that the relative error decreases the bigger the measured object is when using algorithm 2. Figure 4.6 and table 4.7 clearly shows that the quality of the estimated height from algorithm 3 is highly dependent on the object's location. At the front row (locations $L1$ - $L3$) almost all estimated measures are higher than the true values. In almost all cases the height of the object decreases the further away from the camera it is located. The data presented in table 4.6 shows that person 7 and person 10 have the best results, especially from the front row's measurements. From table 4.5 it is seen that those two calibrations have one thing in common which distinguishes them from the other calibration methods, i.e. they both used a tall reference object.

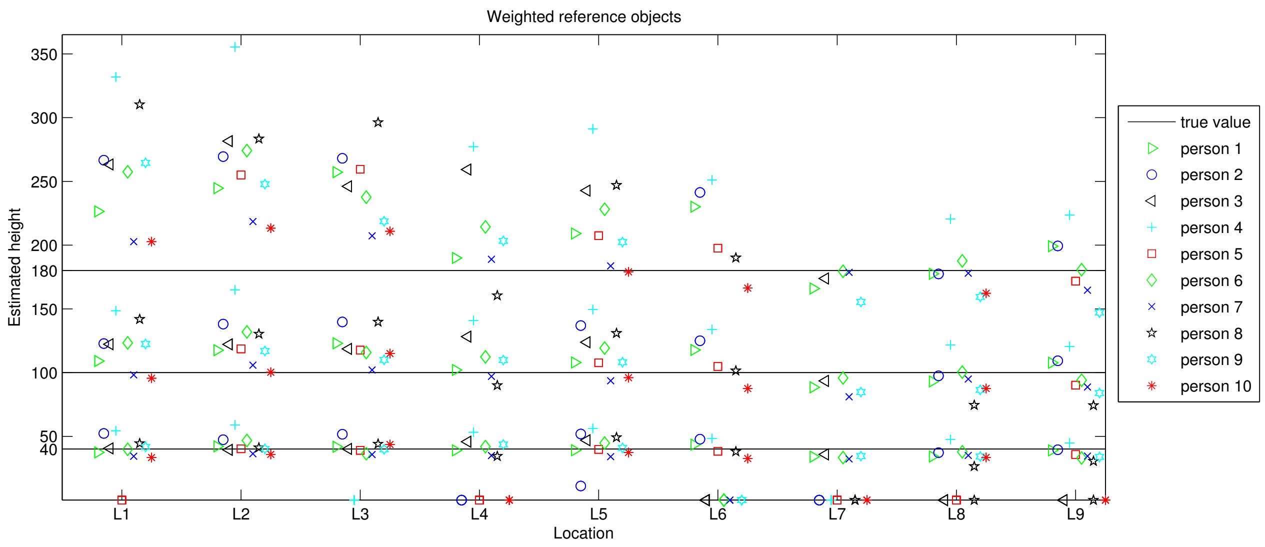

Figure 4.6: *The result from the survey when algorithm 3 was used.*

4.2.2 Usability

From table 4.8 it is clear that the test persons did not prefer algorithm 3 since none of them picked it as their first choice. From the questionnaires they filled in, the reason for that was that they found it hard and time consuming to find five good locations to place the reference object at. The table also shows that almost everyone calibrated algorithm 2 faster than the other two. The only person disagreeing to that spend a lot of time and effort in order to derive the tilt angle, in the end he

Calibration	Ref. object's height(cm)
Person 1	36
Person 2	28
Person 3	32
Person 4	45
Person 5	32
Person 6	30
Person 7	186
Person 8	40
Person 9	84
Person 10	186

Table 4.5: *The height of the object each test person used as their reference object when they calibrated algorithm 3.*

Calibration	**180 cm**	**100 cm**	**40 cm**
Person 1	19.3	11.5	6.9
Person 2	41.0	24.9	19.3
Person 3	37.0	20.2	7.5
Person 4	54.8	40.0	29.7
Person 5	23.1	11.7	3.8
Person 6	22.3	14.1	10.3
Person 7	8.3	6.8	13.3
Person 8	41.3	25.7	15.4
Person 9	21.9	14.0	7.6
Person 10	11.1	8.1	13.0

Table 4.6: *The average error for each test person's calibration of algorithm 3. Each calibration were tested on three different objects of heights 180 cm, 100 cm and 40 cm.*

Location	**180 cm**	**100 cm**	**40 cm**
L1	43.6	21.8	13.4
L2	46.8	24.6	11.2
L3	35.9	20.2	8.4
L4	21.6	15.2	12.9
L5	30.1	19.4	14.9
L6	20.7	15.9	12.9
L7	5.2	11.3	15.1
L8	7.4	11.0	15.4
L9	11.1	13.3	12.1

Table 4.7: *The average percent error for algorithm 3 at each location. Each calibration were tested on three different objects of heights 180 cm, 100 cm and 40 cm.*

Question	**Alg. 1(%)**	**Alg. 2(%)**	**Alg. 3(%)**
Which algorithm was the easiest one to calibrate?	40	60	0
Which algorithm was the quickest one to calibrate?	10	90	0
Which algorithm do you prefer?	30	70	0

Table 4.8: *The result from the test persons comparison of the three algorithms.*

skipped that approach and obtained the angle from a reference object instead.

The majority found algorithm 1 and 2 “easy” or “very easy” to calibrate. When the same question was asked about algorithm 3, most people answered “neutral” (they did not find it very easy, neither very hard) and one person found it “hard”.

Ten out of ten persons preferred to calibrate algorithm 2 by measuring the distance between floor and the camera center and using a reference object of known height in the picture. They also found it “hard” or “impossible” to give a good estimation of the camera’s tilt angle. One person believed that it could be troublesome to measure the distance to the camera from the floor in some setups, although he had no problem in his own specific case.

Ten out of ten persons felt that the parallel lines they gave as input to the calibration process of algorithm corresponded “very well” with the lines drawn by the calibration algorithm as feedback to the user.

4.3 Time complexity

From the way the three algorithms are designed (see chapter 3) there is no particular interest in discussing what time complexity they run in. This is simply because there is not something in the input that can grow and hence the time complexity for all three algorithm has to be constant, $\mathcal{O}(1)$. The time complexity would have been interesting to investigate if the running time was dependent on the input, e.g. the resolution of the image.

4.4 Simulations

In this section the data from the simulations of algorithm 2 will be presented. The reason for only simulate algorith 2 and not the others were mainly because algorithm performed best and because it was interesting to see what impact it would have on the result if exact back project were used instead of the estimation. Tables 4.9-4.13 show the data from the simulations corresponding to $\sigma \in \{0, 1, 5, 10, 25\}$. Algorithm 2 using the ground truth tilt angle is denoted with Alg.2, algorithm 2 with the tilt angle obtained from measurements on a reference object is denoted with Alg.2$'$ and EBP is algorithm 2 when $\mathbf{l_p}$ is calculated using exact back projection.

It is evident that the percent error increases the closer to the camera the object is located. The tables also show that as the noise increases it becomes less important which algorithm is used. Table 4.9 shows that when no noise is introduced, EBP is

		Height / Algorithm								
		180 cm			**100 cm**			**40 cm**		
		Alg.2	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**
Location	L1	10.5	2.1	0.0	14.1	7.8	0.0	14.4	9.4	0.0
	L2	8.8	1.0	0.0	10.4	4.4	0.0	9.8	4.9	0.0
	L3	10.5	2.1	0.0	14.1	7.8	0.0	14.4	9.4	0.0
	L4	13.3	0.2	0.0	16.3	4.8	0.0	17.6	6.9	0.0
	L5	12.9	0.0	0.0	15.3	3.9	0.0	16.3	5.7	0.0
	L6	13.3	0.2	0.0	16.3	4.8	0.0	17.6	6.9	0.0
	L7	16.5	0.9	0.0	19.0	2.9	0.0	20.3	5.0	0.0
	L8	16.3	1.0	0.0	18.6	2.5	0.0	19.7	4.5	0.0
	L9	16.5	0.9	0.0	19.0	2.9	0.0	20.3	5.0	0.0

Table 4.9: *The average percent error for* $\sigma = 0$

the only algorithm with no error. It also shows that the error decreases significantly when the tilt angle is obtained by using a reference object.

		Height / Algorithm								
		180 cm			**100 cm**			**40 cm**		
		Alg.2	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**
Location	L1	10.5	2.1	0.3	14.1	7.8	0.7	14.4	9.4	2.3
	L2	8.8	1.0	0.3	10.4	4.4	0.7	9.8	5.0	2.3
	L3	10.5	2.1	0.3	14.1	7.8	0.7	14.4	9.4	2.3
	L4	13.3	0.5	0.4	16.3	4.8	1.0	17.6	7.0	3.2
	L5	12.9	0.5	0.4	15.3	3.9	1.0	16.3	5.9	3.2
	L6	13.3	0.5	0.4	16.3	4.8	1.0	17.6	7.1	3.2
	L7	16.5	1.0	0.6	19.0	2.9	1.3	20.3	5.8	4.0
	L8	16.3	1.0	0.6	18.6	2.6	1.3	19.8	5.4	4.1
	L9	16.5	1.0	0.6	19.0	2.9	1.3	20.3	5.8	4.0

Table 4.10: *The average percent error for* $\sigma = 1$

Tables 4.4 and 4.12 shows that the noise from the measurements done in the survey is close to be Gaussian with a standard deviation of $\sigma \approx 10$.

		Height / Algorithm								
		180 cm			**100 cm**			**40 cm**		
		Alg.2	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**
Location	L1	10.5	2.4	1.3	14.1	8.0	3.5	16.5	13.8	11.5
	L2	8.8	1.7	1.3	10.5	5.1	3.5	13.3	11.7	11.5
	L3	10.5	2.4	1.3	14.2	8.0	3.5	16.5	13.8	11.5
	L4	13.3	2.4	2.1	16.4	6.4	5.1	20.4	16.1	15.8
	L5	12.9	2.3	2.1	15.4	5.9	5.1	19.4	15.7	15.8
	L6	13.3	2.4	2.1	16.4	6.4	5.0	20.3	16.2	15.8
	L7	16.5	3.3	2.9	19.1	7.0	6.6	24.0	19.9	20.3
	L8	16.3	3.2	2.9	18.7	6.9	6.7	23.6	19.8	20.3
	L9	16.5	3.3	2.9	19.1	7.1	6.6	24.0	19.9	20.3

Table 4.11: *The average percent error for* $\sigma = 5$

		Height / Algorithm								
		180 cm			**100 cm**			**40 cm**		
		Alg.2	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**
Location	L1	10.5	3.6	2.6	14.6	9.8	7.1	24.7	23.9	23.1
	L2	8.9	3.0	2.6	11.3	7.8	7.1	22.5	22.5	23.1
	L3	10.5	3.6	2.6	14.6	9.8	7.0	24.8	24.0	23.2
	L4	13.4	4.7	4.2	17.2	10.9	10.1	30.8	30.9	31.7
	L5	12.9	4.6	4.2	16.4	10.6	10.2	30.2	30.5	31.8
	L6	13.4	4.7	4.2	17.2	10.8	10.1	30.8	31.0	31.8
	L7	16.7	6.5	5.8	20.4	13.6	13.3	37.3	39.6	40.8
	L8	16.4	6.4	5.8	20.1	13.5	13.3	37.0	39.4	40.7
	L9	16.7	6.5	5.8	20.4	13.6	13.3	37.3	39.4	40.9

Table 4.12: *The average percent error for* $\sigma = 10$

		Height / Algorithm								
		180 cm			**100 cm**			**40 cm**		
		Alg.2	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**	**Alg.2**	**Alg.2′**	**EBP**
Location	L1	11.6	8.0	6.6	21.0	19.4	17.8	56.0	58.2	58.5
	L2	10.1	7.4	6.6	18.6	17.8	17.8	54.1	56.3	58.4
	L3	11.7	8.1	6.6	20.8	19.4	17.8	56.2	58.0	58.4
	L4	15.6	12.1	10.7	26.9	26.3	25.8	70.5	78.4	81.3
	L5	15.1	11.8	10.7	26.3	25.8	25.9	69.8	77.7	80.8
	L6	15.6	12.0	10.7	26.8	26.1	25.9	70.7	78.1	81.5
	L7	20.0	16.5	15.1	33.6	34.7	34.6	87.8	102.9	105.9
	L8	19.7	16.4	15.0	33.5	34.7	34.5	87.4	102.3	106.1
	L9	19.9	16.5	15.0	33.6	34.7	34.6	87.4	102.2	106.2

Table 4.13: *The average percent error for* $\sigma = 25$

Chapter 5

Conclusions

In the following sections the conclusions from the results in chapter 4 are given.

5.1 Algorithm 1

From the test persons perspective, algorithm 1 was easy to calibrate since they only had to click at ten points on the image. On the other hand, fluctuating results indicate that the whole method is very sensitive to noise and that is probably why no one had accurate results at all nine locations. As mentioned in section 3.4 the camera used in this thesis has some degree of radial distortion. Since the whole idea behind this algorithm is based on properties of straight lines this is probably the algorithm that was most affected from not correcting the distortion. The reason why this method is so sensitive to noise is most likely because of the vanishing points are calculated by finding the intersection point of two lines. If the angle between those two lines has an error the calculated coordinates of the resulting vanishing point could deviate a lot from its true coordinates.

Since the quality of this implementation of the algorithm can differ a lot between measurements from different locations, it has to be considered to be unreliable and is hence not a desirable choice to base an alarm system up on. The first step to improve this algorithm (except from removing radial distortion) and make it less sensitive to noise would be to use more parallel lines in order to get more accurate vanishing points like they did in Liebowitz & Zisserman (1998). Using several reference objects like in Criminisi & Zisserman (1999,2000) would be the second step to improve it.

5.2 Algorithm 2

This method was the most popular among the majority of the test persons, it was the easiest and quickest one to calibrate and most people got the best overall impression from it. From the survey it is obvious that letting the company's customer estimate the camera's tilt angle by themselves would not be the optimal procedure. However, the cameras are equipped with a motor to change the tilt angle and in the future the company might add support to accurately set the tilt angle automatically. In that case this technique will become even easier to calibrate since the customer only has to measure the distance between the camera and the floor or provide the height of an reference object. The simulations show that this algorithm itself is not very accurate, but that the inaccuracy can be compensated by using a reference object to obtain the tilt angle. The simulations also indicate that when noise similar to that in the survey occurs, the performance of algorithm 2 with an estimation and an exact representation of $\mathbf{l_p}$ can be equally good.

5.3 Algorithm 3

The overall result from this algorithm was not satisfactory, but the result obtained when using reference objects of an height around 190 cm shows that it has potential. The calibrating process is this algorithm's weakest point, since almost all test persons had some problem to find good locations to place the reference object. Both the usability and quality could probably be improved with some small changes. Instead of placing the reference object at only five locations, N arbitrary locations could be used. Then the four triangles can not be used anymore, but other techniques like for example weighting the three closest reference locations could be used. Such a change would change the time complexity of the algorithm to $\mathcal{O}(N)$, where N is the number of reference locations. Since this method tends to overestimate the object's height when it is located close to the camera, another improvement could be to introduce a scaling factor based on the object's location.

5.4 Recommendation

As mentioned in section 4.1 a recommendation based on the result from the survey will be given in this part of the thesis. The first condition was the quality of the estimated height and from section 4.2.1 it is clear that algorithm 2 performed much better than the other two, it had in general more accurate estimations and not a single extreme outlier.

The second condition was the usability of the calibrating process. Since none of the ten test persons found it easy to calibrate algorithm 3 it can not compete against the other two algorithms when it comes to usability. Among the remaining two algorithms most people found algorithm 2 slightly easier to calibrate than algorithm 1.

The third and last condition to be investigated was the time complexity, but as mention in 4.3 it does not make sense to talk about the time complexity from the way the three algorithms were designed. Instead, one could measure the running time of each algorithm and compare them against each other. But since the company can accept a running time up to approximately 1 minute and all the three algorithms runs in much less than 1 second that result should not have any impact on the final recommendation and hence it was not measured.

From the outcome of the survey there is in my opinion only one algorithm which is good enough to be recommended, algorithm 2. The major reason for not recommending algorithm 1 is because it is too unreliable and the reason for not recommending algorithm 3 is that it is both unreliable and not very user friendly to calibrate.

Even if the simulated results indicate that noise neglects the importance of calculating $\mathbf{l_p}$ exactly, it is needless to introduce more errors than necessary. Therefore it is also recommended to use exact back projection to find $\mathbf{l_p}$.

5.5 Applications

As mentioned in the introduction the company's main goal with this thesis project was to get closer to an algorithm which can avoid their alarm system to be triggered by pets. The three algorithms presented in the previous chapters have all been chosen and developed with this in mind. During the whole process the assumptions that the object one wants to measure is standing up and that it is located on the floor has been used.

By using algorithm 2 under those assumptions it is believed that a pet safe algorithm could be constructed. However, if those assumptions are removed, the algorithm as it is constructed today will fail. It would be enough for a cat to walk on a table for the algorithm to output a result similar to if a tall object, far away from the camera, was moving. Another kind of problem would occur if a burglar is not standing up, but is crawling on the floor. The output in such a case will differ a lot depending on how the burglar is oriented in relation to the camera, but in any orientation the estimated height will be significantly shorter than the burglar's true height. Algorithm 2 could without any major modifications be used to measure the height of a person laying on the floor, the problem would then be to determine if the person is laying or standing up.

Other applications could be automatically measuring people's height at a funfair to see if they are within the allowed range to ride a specific roller-coaster, measure objects which are hard to reach, counting items of known height, etc.

Chapter 6

Summary and outlook

In the following sections a short summary and suggestions for future works are given.

6.1 Summary

Three different approaches to measure the height of an object standing on the floor have been investigated. The result was based on a survey in which ten test persons took part by calibrating and giving feedback on three different algorithms I implemented. The data collected from the survey showed that algorithm 2 (Back Projection) was the most accurate, reliable and user friendly algorithm among the three. Simulations showed that algorithm 2 would be more accurate if the line $\mathbf{l_p}$ is calculated exactly, but when there is noise in the input an estimation of that line results in similar accuracy. Under the assumptions mentioned above, algorithm 2 was recommended to a security company to base an algorithm upon to distinguish pets from burglars in a camera based surveillance system.

6.2 Future Works

This thesis has mainly focused on different techniques to estimate an object's height as a first step to build a classifier which is capable of classify pets and burglars. The next step would be to integrate algorithm 2 to run on the company's hardware and thereby be able to start experimenting to find a suitable threshold for objects of what height should trigger the alarm. That threshold should be adjustable to what kind of pets there is in a specific alarm system's environment. Such experiments will also find out how the quality of the measurements is affected when the measure points is taken from a bounding box.

In order for the company to get a stable product out on the market, they need to find a way around the assumptions made in this thesis. To get rid of the assumption that the object has to be located on the floor it might be possible for the customer to mark areas like tables and beds and provide their Z-coordinate to the algorithm.

If that Z value is subtracted from the cameras distance to the floor for such an area, algorithm 2 should work without any major changes. In case of such an improvement of the algorithm the usability of the algorithm might change and hence a new usability test should be made.

Avoiding the assumption that the burglar has to be standing up is more complicated and will probably be very hard to solve for a classifier which only relies on the object's height as input. A learning algorithm might be the solution to that problem and then the object's hight could be an important input parameter to the learner.

It would also be interesting to implement and evaluate the improvements of algorithm 1 and algorithm 3 suggested in section 5.1 and section 5.2 respectively, to see if they can do better than algorithm 2.

Bibliography

Adams, R. 2006 *Calculus - A Complete Course, Sixth Edition*, Pearson - Addison-Wesley, Toronto.

Criminisi, A., Reid, I. & Zisserman, A. 1999 Single view metrology. *Tech. Rep.*, Dep. Eng. Sci., Oxford Univ.

Hartley, R. & Zisserman, A. 2000 *Multiple View Geometry in computer vision*, Cambridge Univ. Press.

Liebowitz, D. & Zisserman, A. 1998 Metric rectification for perspective images of planes. *Tech. Rep.*, Dep. Eng. Sci., Oxford Univ.

Morvan, Y. 2009 *Acquisition, Compression and Rendering of Depth and Texture for Multi-View Video*, Eindhoven.
http://www.epixea.com/research/multi-view-coding-thesisse8.html (2012-06-04)

Liu, B., Gould, S. & Koller, D. 2010 Single image depth estimation from predicted semantic labels. *Tech. Rep.*, Dep. Comp. Sci. and Elec. Eng., Stanford Univ.

Saxena, A., Chung,S.H. & Ng, A. Y. 2005 Learning depth from single monocular images. *Tech. Rep.*. Comp. Sci. Dep., Stanford Univ.

Saxena, A., Sun, M. & Ng, A. 2008 Make3d: Learning 3-d scene structure from a single still image. *Tech. Rep.*, Comp. Sci. Dep., Stanford Univ.

Yamachika, A., Kondo,K., Kobashi, S. & Hata, Y. 2004 Arbitrary viewpoint image generation method of unknown objects in known environment using a single camera. *Tech. Rep.*, Univ. Hyogo.

Acknowledgements

First of all I would like to thank my supervisor Dr. Josephine Sullivan for the guidance throughout the writing of this thesis.

My colleagues at CSC are acknowledged for creating a nice environment at school and for all the discussions (technical and non-) during my 5-years of studies.

I would also like to devote a small part of this section and thank the people who have supported me outside KTH and are very important to me. My mother, Karin and my father Rickard for supporting me always by any means. My siblings, Annie and Axel for always being there for me. *Tack ska ni ha!*

Last but definitely not least, all my love goes to my dear Sissy for all the good times spent together.

Appendix A

Survey

Camera calibration questionnaire

This survey is part of a Master degree thesis project at the Royal Institute of Technology (KTH), Sweden. The purpose is to investigate three different algorithms, which all try to give a good estimation of an object's height in a 2D image. The survey contains both questions and practical steps. The practical steps consists of calibrating some parameters needed by the algorithms. The result will mainly be used for two purposes: measure the quality of the algorithms and to see how user friendly it is to calibrate them for a specific algorithm.
Please note that this suvery can not be filled in alone. The second part of it needs access to specific hardware/software and guidance from the author. Make sure to understand all questions, if somethings needs to be clarified do not hesitate to ask for a clarification!

About you

1.	**Your name**: ____________________________
2.	**How old are you?** I am ______ years old.
3a.	**Do you have any previous experience with camera calibrations?** ○ Yes ○ No
3b.	**If you answered yes on the question above, please describe briefly what kind of experience you have. Otherwise skip to the next question.** ____________________________ ____________________________ ____________________________ ____________________________

Evaluating of algorithms

This section is supposed to be filled in at the same time as the calibration of the three different algorithms is done. Instructions on how to do the calibrations will be given orally. In order to get a good result out of this survey it is important that you ask questions if something is unclear.

Algorithm 1 - Distance between parallel planes

4. After you had marked the parallel lines they were shown to you. How well did the lines shown to you correspond to what you intended to mark?
very bad ○—○—○—○—○ very well

5. What was the most difficult part of this calibration?
____
____
____
____

6. How difficult was this method to calibrate?
almost impossible ○—○—○—○—○ very easy

Algorithm 2 - Reverse projection

7a. This algorithm could be calibrated in three different ways, which way did you prefer?
- ○ Height of camera / Camera tilt angle
- ○ Height of camera / Reference object
- ○ Camera tilt angle / Reference object

7b. Explain your answer in the question above:
____
____
____
____

8. What was the most difficult part of this calibration?
____
____
____
____

9. How difficult was this method to calibrate?
almost impossible ○—○—○—○—○ very easy

Algorithm 3 - Weighted reference objects

10. What kind of object did you use as reference object?
____

11. What was the height of that object? _______

12. What was the most difficult part of this calibration?
____
____
____
____

13. How difficult was this method to calibrate?
almost impossible ○—○—○—○—○ very easy

Comparison of algorithms

14. Which algorithm was the easiest one to calibrate?
- ○ Algorithm 1
- ○ Algorithm 2
- ○ Algorithm 3
- ○ None of them were easy

15. Which algorithm was the quickest one to calibrate?
- ○ Algorithm 1
- ○ Algorithm 2
- ○ Algorithm 3
- ○ None of them were quick

16. Which algorithm do you prefer?
- ○ Algorithm 1
- ○ Algorithm 2
- ○ Algorithm 3
- ○ None of them

17. If you have any other comments, please write them down here:

Thank you for your time!

TRITA-CSC-E 2012:040
ISRN-KTH/CSC/E--12/040-SE
ISSN-1653-5715