CHALMERS

Active Vision System with Human Detection

Using RGB-D images and machine learning algorithms

Master's thesis in Applied Physics and in Biomedical Engineering

ANDREAS BERGGREN
ERIC BJÖRKLUND

Department of Applied Mechanics
Division of Vehicle Engineering and Autonomous Systems
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2012

Master's thesis 2012:28

MASTER'S THESIS IN APPLIED PHYSICS AND IN BIOMEDICAL ENGINEERING

Active Vision System with Human Detection

Using RGB-D images and machine learning algorithms

ANDREAS BERGGREN
ERIC BJÖRKLUND

Department of Applied Mechanics
Division of Vehicle Engineering and Autonomous Systems
CHALMERS UNIVERSITY OF TECHNOLOGY

Gothenburg, Sweden 2012

Active Vision System with Human Detection
Using RGB-D images and machine learning algorithms
ANDREAS BERGGREN
ERIC BJÖRKLUND

Master's thesis 2012:28
ISSN 1652-8557
Department of Applied Mechanics
Division of Vehicle Engineering and Autonomous Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone: +46 (0)31-772 1000

Cover:
The figure shows a possible solution for the *proof of concept* display. The images in the display are shown and explained in Figure 4.3.1.

Chalmers Reproservice
Gothenburg, Sweden 2012

Active Vision System with Human Detection
Using RGB-D images and machine learning algorithms
Master's thesis in Applied Physics and in Biomedical Engineering
ANDREAS BERGGREN
ERIC BJÖRKLUND
Department of Applied Mechanics
Division of Vehicle Engineering and Autonomous Systems
Chalmers University of Technology

Abstract

This master's thesis will focus on an active safety system for the protection of humans close to commercial construction equipments. The purpose is therefore to propose sensors and algorithms suitable for human detection and furthermore to demonstrate a proof of concept.

Early on in the project it was decided to use RGB-D images, which is a conventional color image together with a depth map. This report analyzes both a Kinect sensor and a stereo vision system in order to generate a depth map. Machine learning algorithms were used to classify humans where an artificial neural network was found to be the best performing classifier. Finding informative features is important to facilitate classification. Several imaging features were tested and the six most interesting are presented in this report. The feature called *fourier descriptor* showed the best performance.

Keywords: Human detection, object recognition, computer vision, RGB-D, depth map, Kinect, stereo vision, feature extraction, fourier descriptors, Haar-like features, image moments, machine learning, k-nearest neighbors, support vector machines, decision tree, artificial neural network

Preface

This report is the result of our master's thesis project carried out at CPAC Systems AB in Gothenburg and the final part of our M.Sc. degree at Chalmers University of Technology.

CPAC mainly focuses on safety-critical electronic control systems, but CPAC is also interested in further exploring the area of computer vision in order to evaluate its applicability in active safety. These tools could for instance be used to detect humans close to construction equipment to reduce the number of accidents. Computer vision can also be a powerful tool when it comes to creating autonomous vehicles which is an interesting area of development in, for example, the construction equipment industry.

Acknowledgements

First of all we would like to thank Anders Ek who initiated and were responsible for this project at CPAC Systems. We would also like to thank Dr. Peter Forsberg, our supervisor at CPAC Systems and Dr. Krister Wolff, our supervisor and examiner at Chalmers University of Technology.

Furthermore we would like to thank Oskar Talcoth for some, much appreciated, discussions about image analysis and feature extraction.

And finally, we want to thank other thesis workers and employees at CPAC who helped us in building our data set.

Göteborg July 5, 2012
Andreas Berggren
Eric Björklund

Nomenclature

RGB-D	Red Green Blue - Depth
HOG	Histogram of Oriented Gradients
ANN	Artificial Neural Network
SVM	Support Vector Machine
Pixel	Picture Element
CMOS	Complementary Metal Oxide Semiconductor
IR	Infrared
DFT	Discrete Fourier Transform
FD	Fourier descriptors
kNN	k-Nearest Neighbor
DT	Decision Tree
ROC	Receiver Operating Characteristics
AUC	Area Under the Curve
FPS	Frames Per Second

Contents

1 Introduction

Work in the vicinity of construction machines is a dangerous job. Accidents in this industry are common and the consequences are usually severe. There are a lot of research on developing intelligent systems for detecting humans in proximity to vehicles, also referred to as pedestrians in this thesis. The purpose is often to warn the driver or in worst case automatically initiate braking. The long-term scenario can in some cases even be to totally remove the influence of a driver and create fully automated vehicles. In either case a huge step toward succeeding is to create a robust reliable vision system that enables the vehicles to interact with their environment.

1.1 Background

A common problem with construction machines are that they often are large and bulky. The operator have to keep track of his surrounding which can be troublesome when the machine can block a significant part of the vision field. To aid the operator in avoiding accidents, a vision system could be developed that can help the operator to keep track of the surroundings.

1.2 Objective

The purpose of this master thesis is to implement a *proof of concept* vision system for the detection of humans in proximity to construction vehicles.

1.3 Problem Definition

The task is to implement a fast and accurate human detection system that works mounted on a moving vehicle. The task is translated into a few questions to be answered in order to make the task more manageable. The treatment of these questions is described in the thesis.

- Which sensors are best suitable for the application?
- Which features extraction methods should be used?
- What classifiers should be used?

The thesis work is divided into 3 main areas. These areas are, first of all, to investigate and propose sensor technology suitable for the application. When an appropriate sensor technology is chosen the project moves on to the main focus of this thesis work which is to investigate and propose algorithms for object detection. This includes features and classifiers. The last step is to implement a proof of concept system which can be used to detect humans.

1.4 Limitations

To detect humans and fulfill the objective is a huge undertaking. Some limitations have to be set to make the goal possible to reach. All conceivable different types of image producing sensors[1] are considered but there is a limitation to the price where extremely expensive[2] sensors are not considered. There are also some limitations to the classification part. A human should be detected if having an upright pose, full body is visible and in an indoor environment. Humans in other poses, partially occluded or in other environment is not considered in this project. The detection is limited to individual images, which means that no motion tracking over image sequences is implemented. All these limitation would be possible and interesting to suspend, this is further discussed in chapter 5.

[1] Examples of image producing sensors could be cameras that produce conventional RGB images, heat images or distance map.
[2] It is interesting to evaluate low cost technology in order to develop a cost-efficient system.

1.5 Related Work

There are a wide range of different methods and projects to automatically detect humans. All projects have their different approach and main focus. Some are focusing on finding better sensors, some are focusing on creating better features, and others on creating better classifiers. Inputs from other works done on the subject are considered when choosing the appropriate method to be used.

A commonly used way to detect humans is to use some kind of conventional camera combined with a feature called *histogram of oriented gradients* (HOG)[1, 2, 3] or *Haar-like features*[4, 5]. More information about the environment can be obtained using other sensors like thermal imaging[6, 7], IR-flash[8] or depth sensing technology such as LADAR[9], Kinect[10, 11] or stereo vision[12, 13].

A common factor in the reviewed reports is the use of some kind of heuristic classifier. Commonly used classifiers are *artificial neural networks* (ANN)[14] and *support vector machines* (SVM)[1, 9].

Comparison of different detection methods are difficult to perform due to different data sets and varying evaluation methods but an attempt is made in [15, 16].

A lot of research has also been done on using motion information to facilitate identification of moving objects. Kamijo et al. presents an algorithm for on-board monocular cameras that tracks foreground objects like pedestrians with the use of motion difference [17].

A common method used with stationary cameras is to subtract the background to find a moving human [18], however this method is not possible to use in this project due to the fact that the camera will be placed on a moving vehicle.

1.6 Reading Guidance

The next chapter explains the theory behind the sensor technology, image processing, image analysis and machine learning. The theory chapter can be seen as an encyclopedia and will be referred to from the other chapters. Chapter 3 contains the method and implementation and a description of the setup and circumstances for this project. It is recommended to start reading this chapter and go back to the theory chapter when a better understanding is required. The results and the motivation of choices will be presented and discussed in chapter 4. The last chapter is where the different conclusions are drawn and suggestions for further research are proposed.

Some basic background knowledge in machine learning and image processing could be of good use to fully understand this report, although it should not be necessary.

2 Theory

The Theory chapter will cover the needed theory to understand the report. The Theory chapter is meant to be used as an encyclopedia and will be referred to from the the Method and Implementation chapter. It is recommended to start reading the Method and Implementation and check the theory chapter if needed.

2.1 Image Setup and RGB-D Image

Gray scale digital images are constructed as matrices where each element in the matrix represents a pixel. In case of 8-bit images, each pixel can take a value from 0 to 255, where 0 is a black pixel and 255 a white. A RGB image can be constructed with the use of three such 8-bit matrices where each 8-bit channel represents one of the three colors; red, green and blue [19].

The RGB-D image is an extension of the RGB image where another channel, usually called depth map, is added which contain depth information. This channel is not restricted to be an 8-bit image. In this project a 16-bit image is used, which means that it can handle values from 0 to 65535, where the value represents the distance from the camera in millimeter.

The use of depth map is a way to represent 3D-images. This method to present 3D-images is also called 2D+Z where 2D is either a color or gray scale image and Z is the depth map. This means that each pixel in the color or gray scale image has a corresponding position in the depth map with the information of the distance from the camera [20].

Examples of depth maps can be seen in Figure 4.1.1. Since the depth map contains values from 0 to 65535 and a normal gray scale image can only contain 255 different type of gray, it is hard to visualize the depth map with a gray scale image. For such images color mapping is often used to enhance the contrast and make the image more easy to interpret [21].

2.2 Sensors

Two technologies for acquiring RGB-D images are analyzed in the report, these are the Kinect sensor [22] and a stereo vision system. Both techniques have their advantages and disadvantages which are further analyzed in section 4.1.1.

2.2.1 Kinect

The Kinect sensor consists of three parts, a CMOS camera that captures color images, an IR laser emitter and another CMOS camera that is used as an IR receiver. The IR sensors is used to create a depth map with a technique called light coding [22].

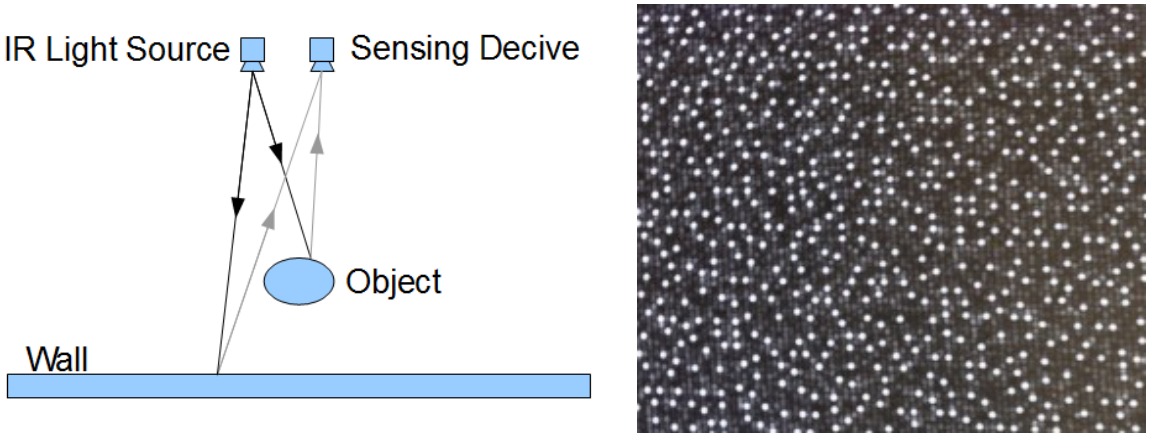
The IR emitter projects a static pseudo-random grid of points on the surroundings and the reflections of these points is captured by the receiver, see Figure 2.2.1. This creates two different views of the point grid, the transmitted light pattern and the deformed received light pattern, and by comparing these patterns a depth map can be created [22].

The Kinect sensor is an active sensor that uses IR. This makes it ineffective in environments with large amounts of radiation in that frequency, for instance outdoors in direct sunlight. This problem is further discussed in section 4.1.1. The quality of the depth map also depends on how the environment reflects IR [22].

2.2.2 Stereo Vision

The stereo vision method is actually quite similar to the light coding method. The difference is that stereo vision uses two conventional cameras and tries to identify the position of identical points in both images. When the position of a point is identified in both images the distance to that point can be determined by triangulation, see Figure 2.2.2.

The challenge with stereo vision is the *correspondence problem*, that is to identify the same points with both cameras. This is difficult for a number of reasons. The matching is a two-dimensional problem but by



(a) An illustration of how the Kinect sensor works

(b) A pseudo-random grid

Figure 2.2.1: (a) The black arrows represents the borders of the static pseudo-random grid of points and the gray arrows are supposed to illustrate how the other camera is seeing the reflection of this grid. (b) An image of a part of the pseudo-random grid emitted by the Kinect.

performing an epipolar rectification[1] the problem can be simplified to one dimension.

The images can lack consistency due to intensity and color differences in images from viewpoint or camera differences and camera noise, these effects are usually not a major problem.

It is harder to deal with the fact that the environment can consist of untextured regions or repetitive patterns which complicates unique matching of two points. There is also a possibility that some regions are occluded in one of the cameras in which case it is totally impossible to get any depth information in that region [24]. These problems will not be further discussed here as they are outside the scope of this thesis.

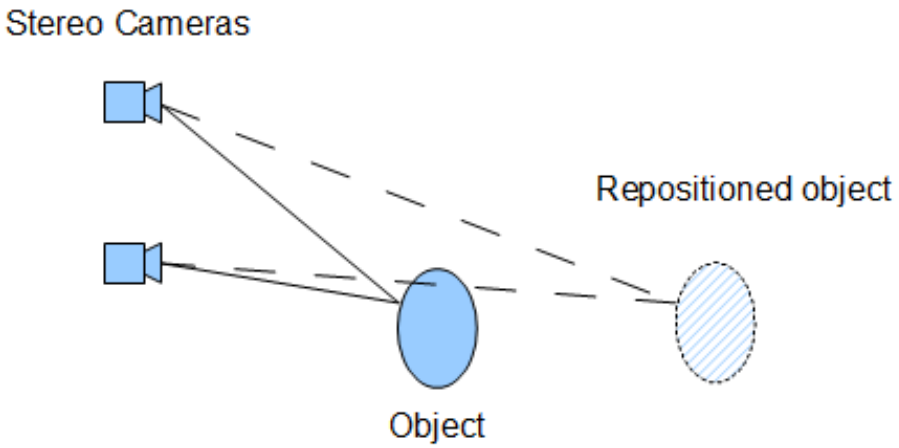


Figure 2.2.2: An illustration of stereo vision. The figure shows how the angle between a point and the cameras changes when an object moves away from the camera system.

[1]Epipolar rectification is the procedure of transforming a pair of images so that epipolar lines are parallel and horizontal in each image [23].

2.3 Classification

In contrast to deterministic classification, the heuristical method uses "trial and error" to solve a given problem and uses training data in order to give the classifier the information it needs. This makes the classifier highly dependent on how well the training data represents the actual classification problem.

The detection of a human is a complex problem. If deterministic methods were to be used the problem would probably need to be simplified to be able to prove that the problem is solved. The simplification means that the solved problem is not the actual problem but a simplified version of it.

The simplification of the problem is not needed when using a heuristic method. The negative aspect of the heuristic method is that it is hard, if not impossible, to actually prove that an optimal solution is found. It is also often hard to know which information the classifier is using when the classification is done [25].

2.3.1 Training a Classifier

This report will only describe the very basics of what is needed for a computer to be able to learn and to classify images. It will not cover the mathematics behind the learning theory or the used learning algorithms.

The purpose of the classifier is to determine which class an image belongs to, in this case: human or not human. To be able to do this, the classifier is trained with a data set. A commonly used method is called supervised learning which is implemented by training the classifier with a set of images and the classes they belong to. Pre-processing and features help the classifier to interpret the images. Figure 2.3.1 shows the training procedure.

The classifier uses the inputs and the class labels of the training data to optimize classifier-dependent internal parameters to correctly classify the training data.

A common problem with supervised learning is overfitting, that is when the classifier becomes to specialized towards the training data. Overfitting is dealt with differently depending on the classifier but usually crossvalidation is used. Crossvalidation means that some of the labeled data are used for validation instead of training. If the classification accuracy is improved for the training set but reduced for the validation set, the classifier is becoming to specialized and training should be stopped [26, 27].

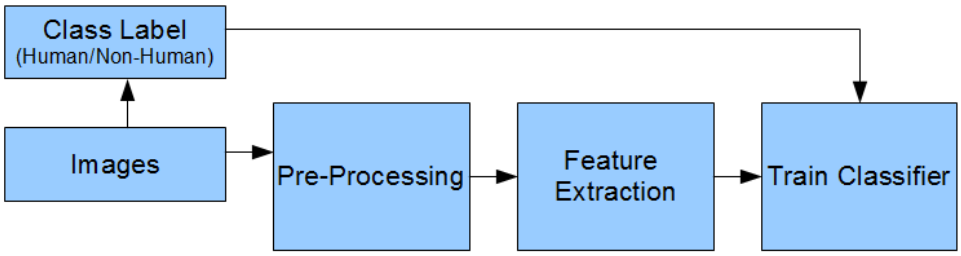


Figure 2.3.1: This figure illustrates the procedure when training a heuristic classifier. The training set is preprocessed, features are extracted and presented to the classifier along with a class label that tells the classifier whether an image contains a human or not.

2.3.2 Pre-processing Algorithms

This section contains the theory behind three of the images created in the pre-processor, see Figure 3.2.4. The purpose of having the pre-processor, instead of doing these computations as a part of the feature, is to make sure that these computations are only executed once.

Binary Image and Thresholding

The binary image is acquired from the depth map by thresholding as shown in the first step in Figure 2.3.2. Thresholding is a imaging operation which means that a certain threshold is chosen and all pixels with values greater or equal to the threshold is set to white and all pixels below is set to black. In the example of Figure 2.3.2 the purpose is to keep all pixels that have information about the depth and therefore the chosen threshold

is set to 1. The binary images are used in many situations, primary to find interesting areas and extract contours but also in the pre-processor to understand how much information a certain segment contains [19].

Contour Image

When working with object detection or object recognition it is often preferred to find the contour of the object of interest. The first step in finding the contour is to create a binary image. The binary image is then systematically scanned until a component is found. When a component is found its edge is followed and stored before the systematic scanning continues. This procedure continues until the whole image is scanned and all contours are stored [28]. In this project, the largest contour is kept and the rest discarded. The procedure of finding the contour is shown in Figure 2.3.2.

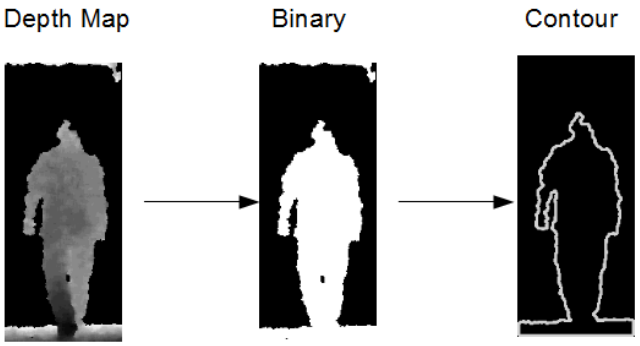


Figure 2.3.2: Illustration of the steps from depth map to contour image of a human.

Integral image

The purpose of the integral image is to speed up calculations of features that uses rectangular filters, that is the sum of rectangular areas of an image.

The integral image is used as a lookup table. It is constructed so that the element with index (i, j) in the integral image corresponds to the sum of all elements inside a rectangle with the corners (0, 0), (i, 0), (0, j) and (i, j) in the original image, see Figure 2.3.5.

The use of integral images is only beneficial if the total area of the used rectangles are greater than the the area of the image, or more precise, if the computations used to sum the total area of the used rectangles is less than the computations used to construct the integral image and perform the look ups [29].

2.3.3 Features

This section explains some background theory of the used methods to extract features. In this report, a feature is defined as one value. Each method generates a feature vector that consists of several features. The word *feature* is sometimes also referring to a feature extraction method.

Fourier Transform

The fourier transform is an important tool that can be used in image processing to decompose an image into sine and cosine components. The output of the fourier transform will be the image representation in the frequency domain from which both the amplitude and the phase can be extracted as shown in Figure 2.3.3. The *discrete fourier transform* (DFT) is the sampled version of the fourier transform and is used for digital images. Using the frequency domain representation of the image, numerous operations can be done with different kind of image processing purpose [19]. In this report the phase and amplitude are used by the classifier without further processing.

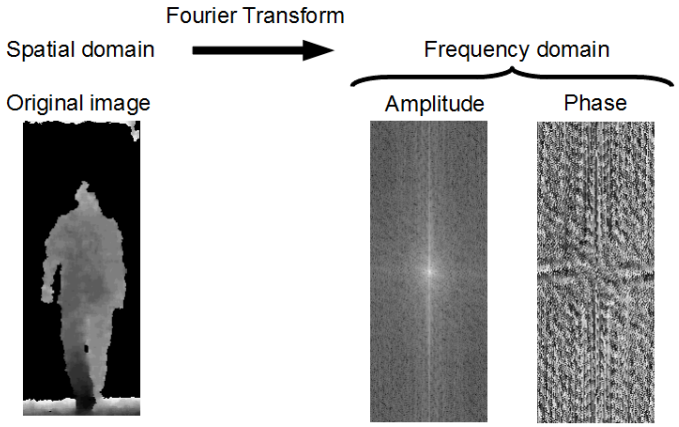


Figure 2.3.3: Illustration of the fourier transform. The amplitude and the phase are the output from the fourier transform. The shown amplitude image is in logarithmic scale. Both the amplitude and phase image are normalized.

Fourier Descriptors

Fourier descriptors (FD) is a shape based feature that utilizes that the shape can be roughly described by the low frequency components of the contour.

The contour is given by N points in a two-dimensional space. To be able to use the fourier transform on these points they are represented as complex number with the x-coordinate encoded as the real part and the y-component as the imaginary part. The resulting frequency spectra (F0, F1 ... F(N-1)) contains information about the shape, low frequency contains information about the general shape and high frequency describes finer details. For classification general information is desirable and for some classification methods, like neural networks, it is important to have the same number of inputs so a fixed number of the lowest frequencies are used to construct the feature vector. Figure 2.3.4 illustrates how much information that is contained in the coefficients.

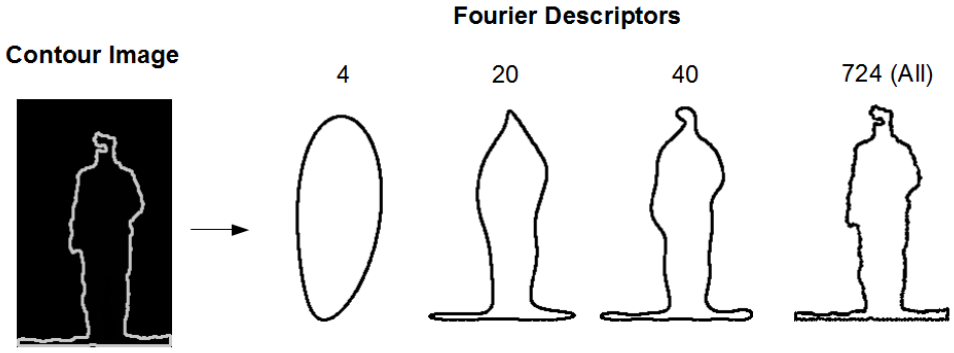


Figure 2.3.4: Illustration of the contained information in the coefficients. Fourier transform is applied on the contour image to the left. Then a number of low frequency coefficients are used to create the images to the right via inverse fourier transform. The number above the images tells how many coefficients that are used.

An advantage of FD is that they are rotation invariant. It is also possible to make them translation and scale invariant. All translation information is contained in F0, the constant component. Thus FD is made translation invariant by setting F0 to zero. The scale invariance can be realized by dividing each component by F1 [30]. In this project 40 of the low frequencies are used to create the feature vector.

Image Moments

The image moments are shape descriptors that often plays an important role in object recognition. There are three kind of moments that are used in this project; spatial moments, central moments and central normalized moments. These are calculated as shown in Equation 2.3.1 - 2.3.3 and uses a filled version of the contour image from the pre-processor [19].

$$M_{ji} = \sum_x \sum_y x^i y^j f(x, y) \tag{2.3.1}$$

$$\mu_{ji} = \sum_x \sum_y (x - \bar{x})^i (y - \bar{y})^j f(x, y) \tag{2.3.2}$$

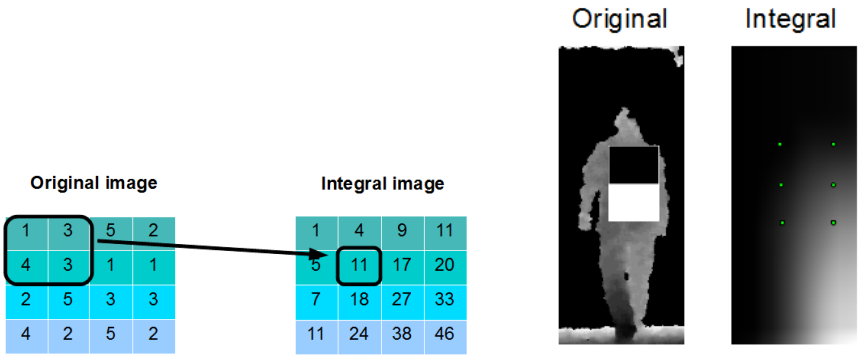
$$\eta_{ji} = \frac{\mu_{ij}}{\mu_{00}^{(1+\frac{i+j}{2})}} \tag{2.3.3}$$

Here, f(x,y) represents the intensity of an image and x and y is the pixel's position in the image. From these equations 24 moments have been proven to be useful and are shown in [31]. The central normalized moments calculated by Equation 2.3.3 are shown to be both translation and scale invariant. Another interesting moment feature that is often being used but not in this report are the *Hu moments*. These moments are invariant under translation, changes in scale and also rotation [19]. These are not used due to that they did not perform well.

Haar-Like features

Haar-like features compare the pixel intensities at different parts of an image usually using rectangular filters. A rectangular filter is just the sum of all pixels inside a rectangular area of the image. The features are derived by adding or subtracting one or several of these sums [32]. An example of a Haar-like feature can be seen in Figure 2.3.5.

In this project, several rectangular filters of random size are randomly placed. Then a random number between 2 and 6 of theses are either added or subtracted from the final feature value. The features are created at random but the random generator is initialized with the same seed for all images, that is, the features used on the training set is the same as the ones used on the images that are classified.



(a) How to create integral image. (b) How to use integral image.

Figure 2.3.5: (a) How the integral image is created. Each number is the value of a pixel. Each element in the integral image is the sum of all elements inside a rectangle with upper left corner in origin and the down right corner at the corresponding element from the original image. (b) How the integral image could be used. In this case the wanted feature is the value of sum of pixels inside the white rectangle subtracted by the sum of all pixels inside the black rectangle. If you have the integral image it is possible to calculate this value by using the points shown in the integral image. These points corresponds to the corners of the rectangles in the original image.

Histogram of Oriented Gradients

HOG is a widely used feature for pedestrian detection in color/grayscale images. This method uses intensity gradients for each pixel to represent shape information.

A typical setup is to have an image size of 128x64 pixels and to divide the images into 8x8 pixel cells. The gradient is computed for each pixel in the cell and is used to build a histogram where the gradient orientation is divided into 9 bins in the range 0-180°. The cells are divided into larger blocks and their corresponding histograms are normalized within the block to take into account local variation of lighting. Each cell typically belongs to four blocks and will hence contribute with four versions of its histogram to the final feature vector, normalized in four different ways [2]. Figure 2.3.6 shows the procedure.

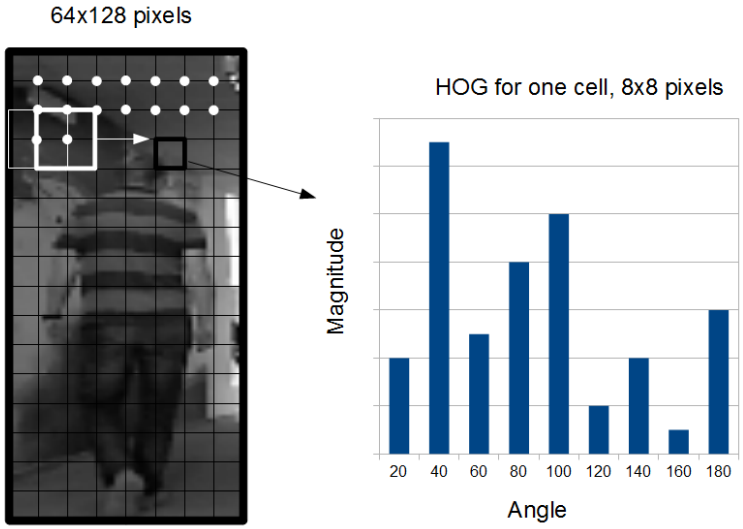


Figure 2.3.6: An illustration of how HOG works. The whole picture is 64x128 pixels and is divided into many 8x8 pixel cells. Each cell has a small corresponding HOG. The white square, called block, of size 16x16 pixels, covers four cells for which the histograms are normalized and sent to the final feature vector.

Overview Histogram Feature

The overview histogram feature was created with some inspiration from [33]. The purpose is to make a 2D - histogram of the depth map image by removing the Y-axis and projecting all points to the XZ-plane as shown in Figure 2.3.7. This means that if a point would be at (1,2) with depth value (Z) 3 it would increase the 2D histogram at point (1,3) by one.

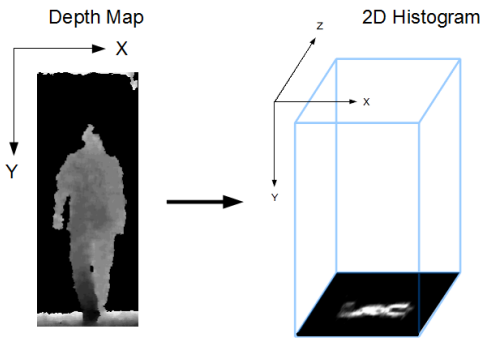


Figure 2.3.7: How the overview histogram is created. The histogram is the gray scale image in the bottom of the box. The whiter the pixel the more corresponding positions from the depth map exist.

After the histogram is created it is smoothed, thresholded, contours are found and finally a rectangle and a ellipse are fitted to the contour as good as possible. The values from the width and length of the rectangle and the ellipse are used to create the feature vector. Figure 2.3.8 shows this procedure.

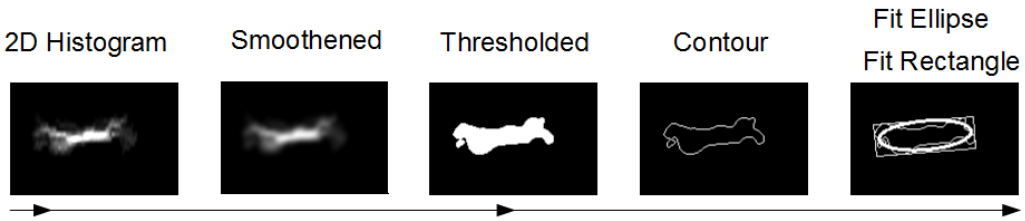


Figure 2.3.8: How the overview histogram is used. The final feature vector consist of the length and width of the ellipse and rectangle.

Unfortunately this feature did never work since it was discovered that the resolution of the depth, the Z axis, decreased the further away an object was. The feature was included in the report anyway because it seems promising if the problem with the resolution could be fixed, this is further discussed in section 4.2.1.

2.3.4 Classifiers

This section contains a brief description of the analyzed classifiers. These classifiers are machine learning classifiers that all are trained using supervised learning.

There are two different types of supervised learning models, classification classifiers and regression classifiers. Classification classifiers map the input space to a finite set of predefined classes, whilst regression classifiers map the input space to a continuous real valued output space [34].

k-Nearest Neighbor

The *k-nearest neighbor* (kNN) is a very simple classifier but is in many cases effective. The training procedure only consists of storing the features from the training data. When a new instance is to be classified it is placed in the feature space and is then given the label of the most common labels of the k nearest neighbors [35]. Figure 2.3.9 illustrates a simple example.

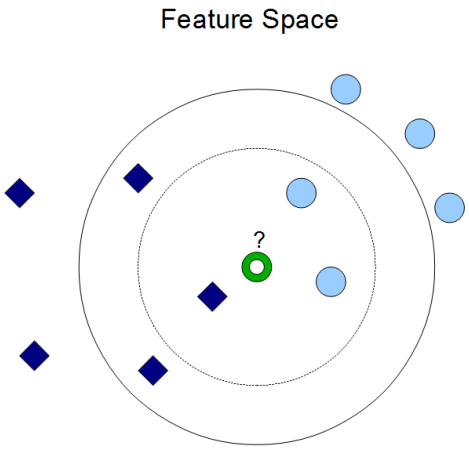


Figure 2.3.9: Example of how kNN-classifier works in a 2D feature space. The ring in the middle is to be classified as either a circle or a square. The already existing circles and squares are the used training data. How should it classify the ring in the middle? If k = 3 it would classify the ring as a circle but if k = 5 it would classify the ring as a square.

Support Vector Machine

SVM are primarily intended for two-class classification and uses a quite straight forward method. During the training phase, the SVM tries to find a (N-1)-dimensional hyperplane that separates the N-dimensional training data into the two classes with maximal margin, that is, with maximal distance to all nearby samples. A sample can then be classified depending on where the sample is located in relation to the hyperplane, see Figure 2.3.10.

When the problem is non-linear it is impossible to separate the classes with a linear hyperplane. The solution is to map the problem into a higher dimension space[2] which often makes the problem easier to separate in that space.

The training samples that lie closest to the hyperplane are the only ones that affect the position of the hyperplane, hence they are called support vectors as they "support" the hyperplane. The other training samples do not contribute to the training. This makes the SVM somewhat vulnerable to noise which motivates the introduction of a soft margin that allows some of the training samples to lie inside the margin or even be misclassified [36].

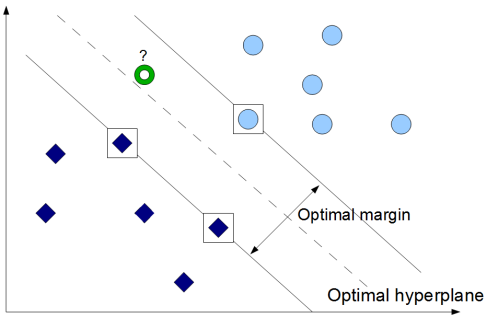


Figure 2.3.10: Example of how SVM classifier works in a 2D feature space. The already existing circles and squares are the used training data. The support vectors are marked with a square and lying on the line. The ring is in this case classified as a circle.

Decision Tree

A *decision tree* (DT) consists of a tree of linked decision nodes and can be either a classification tree or a regression tree. The output of a classification tree is which class the input belongs to and the output of a regression tree is a floating point number which gives an indication of the probability that the input belong to some class.

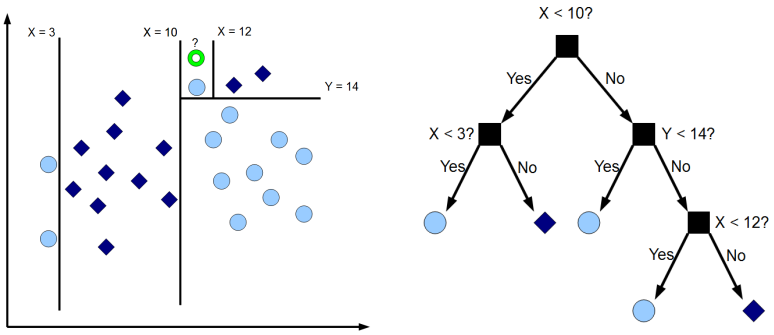
The input vector can consist of both continuous valued variables and discreet case variables. In the case of continuous valued variables the node checks which range a sample belongs to.

The tree structure is created at the training stage. A set of labeled input vectors is recursively split into subsets until some stopping criteria is met, for instance if all samples in a subset belongs to the same class. Each split corresponds to a node in the tree, the first split is done in the root node, the second in one of its child nodes and so on. The split is based on the input parameter that can order the subsets the most which means that the depth of the node indicates how influential the parameter used by that node is. The root node is most important and the importance decreases with each generation [34]. A simple example of a DT can be seen in Figure 2.3.11

Artificial Neural Network

An ANN consists of neurons and connection between the neurons which are supposed to imitate a human brain. A neuron can take an arbitrary number of inputs, which are weighted individually and summed. The summed value is compared with a threshold function and the threshold value is weighted and then sent as an output. This output could either be the final output or used as input to one or several other neurons [37]. The neurons can be connected in many variations but the most common is that they are placed in layers and this is the used type of network in this project. Figure 2.3.12 illustrates a neuron and an example of a network. This type

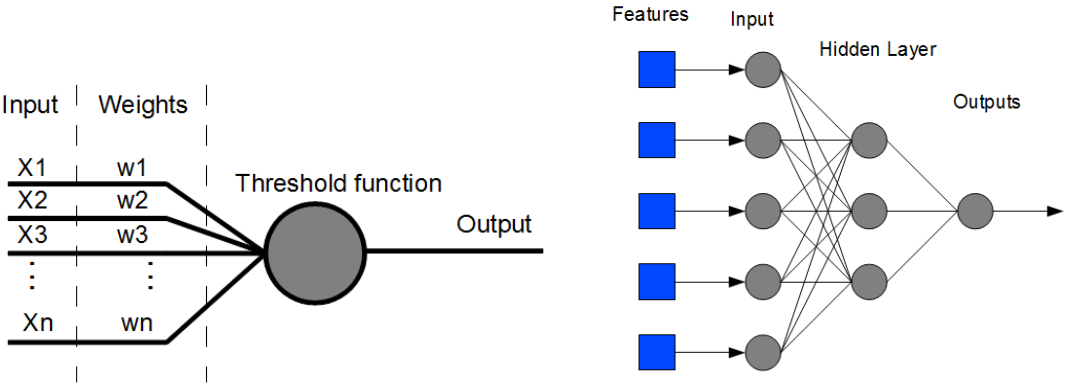
[2]This is accomplished with the use of kernel functions, more about this can be read in [36].



(a) (b)

Figure 2.3.11: Example of how DT classifier works in a 2D feature space. The already existing circles and squares in (a) are the used training data. The lines in (a) shows how the tree in (b) split the features. The method to classify the ring is just to follow the tree in (b) from the top until a class is found, in this case a circle.

of network is called a feed forward network which is arranged so that the output of the neurons in one layer are used as input in the next layer [37].



(a) A neuron (b) A neural network

Figure 2.3.12: Example of how an ANN-classifier works. An illustration of a neuron can be seen in (a). In (b) an example is shown of how input and output can be connected together with one hidden layer. The numbers of hidden layers and neuron in the hidden layer can be changed.

2.3.5 Evaluation

Confusion matrix and *receiver operating characteristic* (ROC) are the two methods used to be able to compare the features or classifiers and present the result in the report.

Confusion Matrix

To evaluate the different classifiers and features a confusion matrix is created. This is a method of analyzing how often the classifier is correct and how often it gives false classification. The correct classifications are divided into true positive and true negative and the same thing is done for false classification [38]. Figure 2.3.13 a confusion matrix. The accuracy of a classifier is defined as *correctly classified samples* divided by *all samples*.

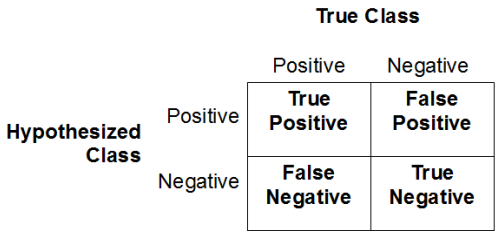


Figure 2.3.13: Illustration of how a confusion matrix works.

Receiver Operating Characteristic

The ROC curve is a method to evaluate the classification. This method analyze the true positive rate and the false positive rate as a function of the classification threshold. Points are generated by sweeping the threshold range and computing both the true positive rate and the false positive rate. The ROC curve is generated by connecting these points, an example can be seen in Figure 4.2.1. The *area under the curve* (AUC) can be calculated and used to compare performance with other classifiers [38]. A good classifier is characterized by having a high true positive rate and a low false positive rate.

3 Method and Implementation

This chapter describes the approach used to achieve the results of the report. Starting with the choice of sensors and then moving on to how the algorithms are built and at the end, a working prototype.

3.1 Setup

This section presents under which circumstances the project where performed.

3.1.1 Environment

The training data is collected in an indoor environment due to restrictions in the Kinect sensor which does not work outdoors. This indoor environment contains more regular shapes, like walls, floor and furniture, than an outdoor environment or a construction site where the final product is supposed to be used.

3.1.2 Sensors

To start with, a comparative study is conducted to find suitable sensors. The analyzed sensors in this study are LADAR, thermal imaging, Kinect and stereo vision. The idea to use LADAR and thermal imaging were discarded but are discussed as a future improvements in chapter 5. The two remaining sensors; a Kinect sensor and a stereo vision system, are described in section 2.2. The advantages and disadvantages of the sensor is analyzed and presented in section 4.1.1. The stereo vision system is created with two standard webcameras with fixed position relative to each other. This setup is shown in Figure 4.1.2.

3.1.3 Software

The software used during this project are listed in Table 3.1.1. The C++ environment from MS Visual studio combined with OpenCV have been the main tools to solve the problem. MATLAB was added as a means of presenting data graphically and OpenNI is used to receive the Kinect image from the sensor.

Table 3.1.1: The software used in this project.

Software	**Purpose**	**Comment**
MS Visual Studio 2010 [39]	C++ environment	
MATLAB R2010b [40]	Plotting graphs	
OpenCV V 2.3.0 [41]	Image function	An open computer vision library.
OpenNI V 1.5 [42]	Kinect image grabbing	An open source device interface library.

3.2 Classification

Theory behind heuristic classification is described in section 2.3. This section contains the implementation of segmentation, pre-processing, feature extraction and classification. Figure 3.2.1 shows how these parts are connected. To get a working classifier it has to be trained before it can be used for classification, this is described in section 2.3.1.

3.2.1 Segmentation

The sensor provide both a color image and a depth map image to the segmentation.

The segmentation is performed by first dividing the 3D space into depth layers with a thickness of 1000 mm and an offset of 250 mm, meaning that the layers overlap.

Equation 3.2.1 is used as an approximate way of determining the pixel height corresponding to 2100 mm at different depths. The constant C is calculated by using a measured value of the pixel height (442 px) at a depth of 2500 mm. The calculated height and an aspect ratio of 2.5 are used to construct a rectangle at each depth

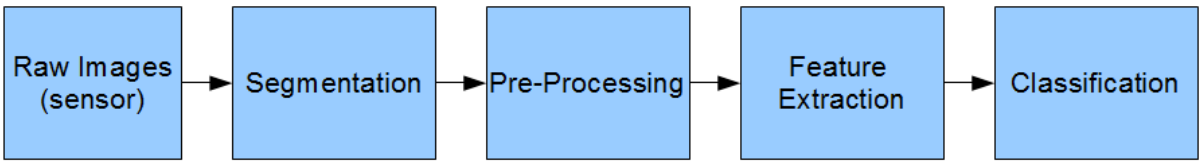


Figure 3.2.1: Overview of the procedure to classify a human.

for which the pixel dimensions corresponds to the real dimensions: height = 2100 mm and width = 840 mm. The rectangle is used as a window which is swept, sometimes called sliding window, over the depth layer with some overlap, see Figure 3.2.2, and each resulting segment corresponds to a 3D box which is further processed.

$$f(x) = C/x$$
$$C = 442\ [\text{px}] \cdot 2500\ [\text{mm}] \approx 1.1e6[\text{px} \cdot \text{mm}] \tag{3.2.1}$$

As the approximate size of a human is known at each depth, the percentage of depth pixels that should lie inside the box when the box contains a human is also known. Boxes that contain little or no depth pixels at all can therefore be discarded without further processing which saves a lot of computations.

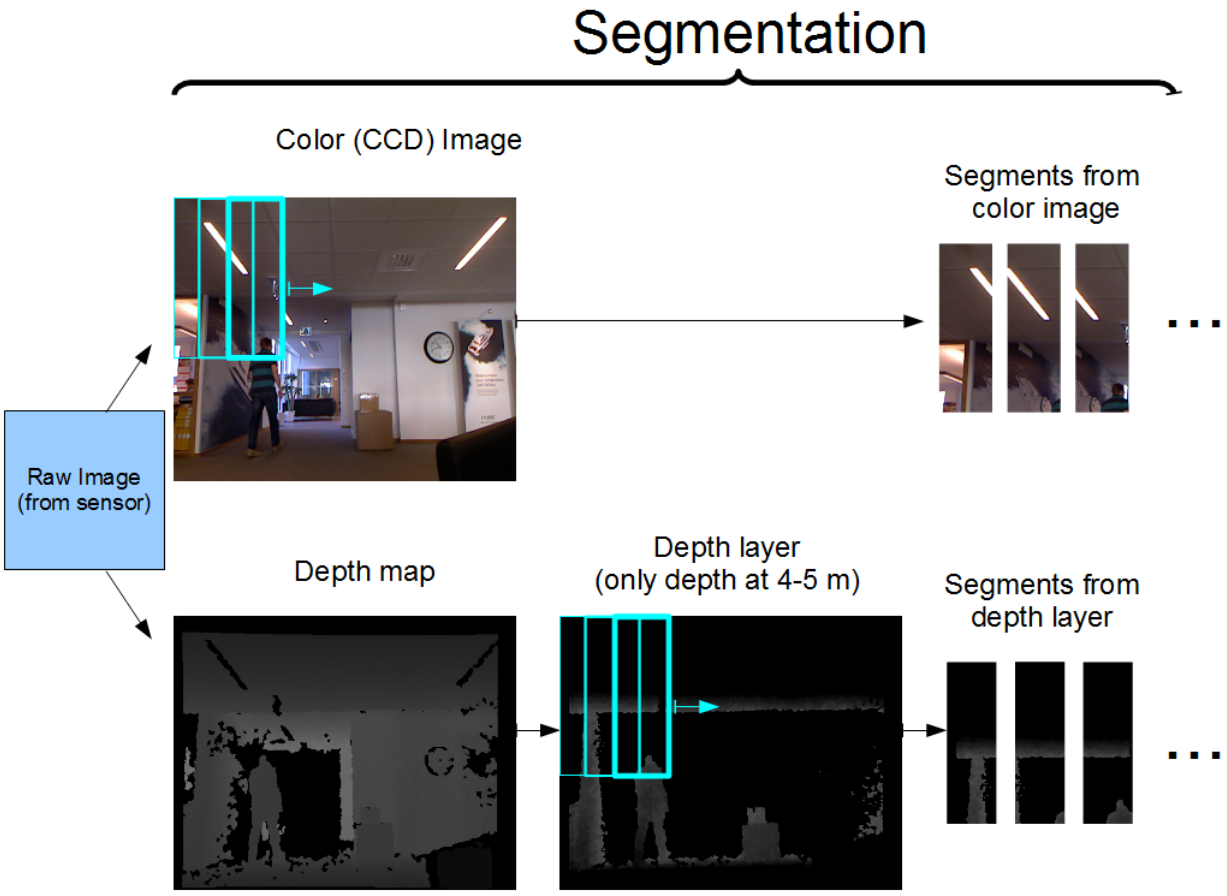


Figure 3.2.2: Overview of the segmentation procedure. First two images, the color image and depth map, are received from the sensor. A depth layer is extracted and a rectangle is swept over the images in order to obtain the segments from both the depth map and the color image. Then a new depth layer is extracted and the procedure continues until the whole space is segmented.

The classifier assumes that all segments have the same size. The solution used is that the change of window size is actually implemented by changing the size of the original image and keeping the window size fixed.

In this way if the distance is swept from 2500 mm to 7500 mm about 2000 segments are created of which as many as 1800 segments contain little or no depth information and can be discarded[1]. This means that about 200 images are further sent to the pre-processor.

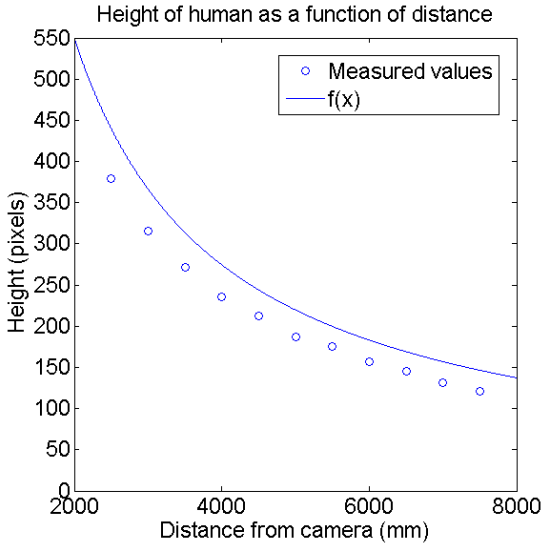


Figure 3.2.3: The height of a human as a function of the distance to the camera. The function in Equation 3.2.1, used to convert the real height of 210 cm at a certain distance to pixel height, is compared to measured values of a 180 cm tall person.

3.2.2 Pre-processing

The pre-processor receives two types of images from the segmentation part. These are the segments from the color image and the segments from the depth map. During the pre-processor step the color segments and depth segments are transformed to other types of images to make it easier for the feature extractor to extract the feature. Another aspect is that multiple feature extractors can use the same image type without having to recompute that image. The types of images created and sent forward from the pre-processor step are shown in Figure 3.2.4.

Two of the image types are a gray scale image, which is derived from the color segment, and the color segment itself.

The depth layer segment is converted to three other types of images; binary image, contour image and an integral image, which are described in section 2.3.2. These three, along with the original segment is sent forward to the feature extractor. Altogether there are six types of images sent from the pre-processor to the feature extractor, all this is illustrated in Figure 3.2.4.

3.2.3 Feature Extraction

This section will describe how the features are used. A more detailed description on how these features work can be found in section 2.3.3. Every feature uses one or more images from the pre-processing part. The implemented features along with their used image type can be seen in Table 3.2.1.

3.2.4 Classification

Four different heuristic classifiers are being analyzed, these are kNN, SVM, DT and ANN. How these work are described in section 2.3.4. There are several parameters that can be customized for each method and the

[1]The number of segments that can be discarded is of course highly dependent on the environment at the moment. The indoor environment shown in Figure 3.2.2 is considered in this case.

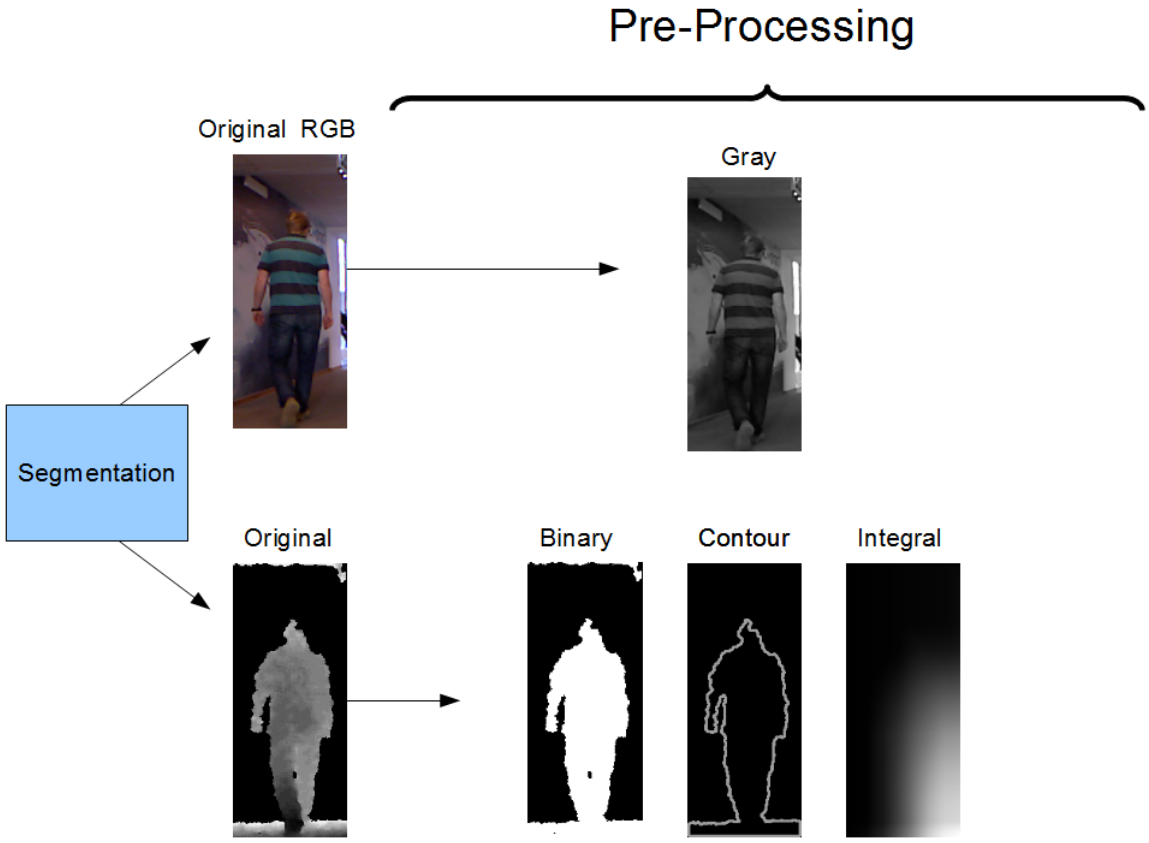


Figure 3.2.4: Overview of the pre-processing procedure. The binary image is not used in any feature itself but is used to determine whether a segment contains enough information to be a human. It is also used to derive the contour image.

Table 3.2.1: The most interesting feature extraction methods.

Feature	**Used Image**	**Comment**
Depth Map	Depth Map	Depth map is downsampled
DFT	Depth Map	
FD	Contour Image	
Moments	Contour Image	
Haar-like features	Integral Image	
HOG	Gray Scale Image	
Overview Histogram	Depth Map	Does not work yet

purpose is to make them as good as possible for the chosen features. The problem here is that it is impossible to know if the parameters are optimized for a certain problem. This mean that it can be hard to compare classifiers if the parameters are badly chosen. The goal is to make the conditions for the classifiers as similar as possible. The base for all of these classifiers were already implemented in OpenCV. This did save a lot of time when creating the classifiers and also made it possible to use some classifiers even when the experience in those classifiers were low.

Combination of Features

Each feature has its own classifier in a "pre-classification" stage. When two or more features are used a new classifier of the same kind is created and uses the output from the "pre-classification" stage as input. This procedure is shown in Figure 3.2.5. This method makes it very easy to add and analyze additional features.

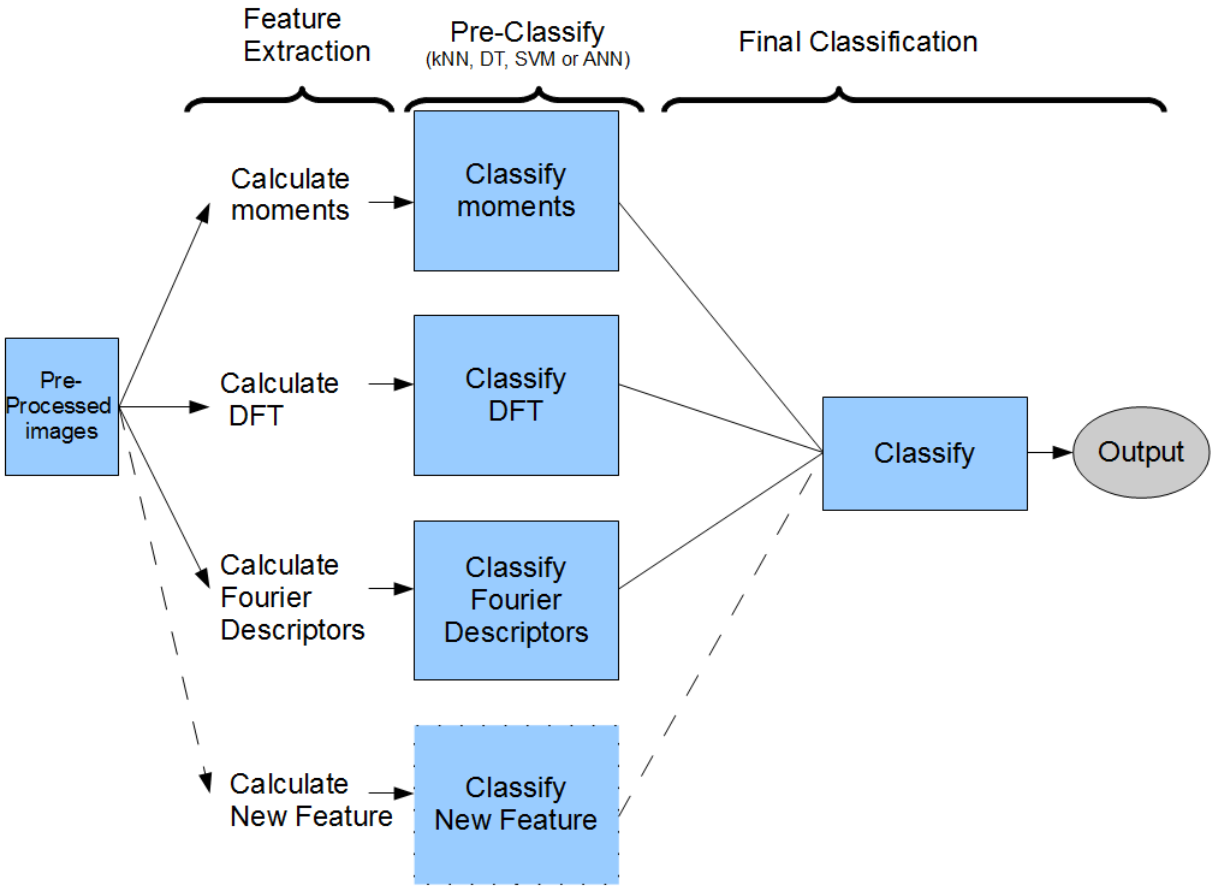


Figure 3.2.5: Shows the implemented setup which makes it easy to add and analyze new features.

3.2.5 Data Mining

The data set is divided into two parts, a training set and a test set. As the name indicates the purpose for the training set is to train the classifier and the purpose of the test set is to test the performance of the classifier. The positive training set contains 1100 images of four persons and the positive test set contains 800 images of four other persons. The negative test set were collected by capturing 1500 images from an environment that was not used for the training set. The method to collect the negative training set follows a different procedure. First 3500 negative training images were captured at four different places. The classifiers were then trained and used in an environment without humans. Whenever an object in this environment were classified as a human it was added to the negative training set. After several false classifications the classifier was retrained with the new training set and the procedure was repeated until a satisfying classifier was trained. At the end there were 5000 negative images.

3.3 Proof of Concept

The purpose of this project is to implement an active safety system that should detect pedestrians from a moving vehicle. The implication of this is that the detector must be able to detect a pedestrian before a collision is inevitable or the consequences might be catastrophic.

Image processing is often very suitable for parallelization, including this case. The same computations are done on several parts of the same image making it possible to run them concurrently. This will not make any difference if they are run on a single processing core. Running computations concurrently on a multicore processor however, can increase the computation speed considerably.

Multi-threading is implemented in the application to enable parallel classification of the different segments. A threadpool with a work queue is used to avoid unnecessary spawning of threads. When the image is segmented each segment is added to the queue and is processed as a thread becomes available.

The *proof of concept* prototype consists of a Kinect sensor connected to 12 V battery and mounted on mobile robot. An ordinary laptop is used as processor and display. The setup can be seen in Figure 4.3.2.

4 Result and Discussion

This chapter presents the results from the evaluation. The first part is a comparison between the Kinect sensor and a stereo vision system. The rest of the result chapter is focused on the image processing and algorithms. Only the Kinect sensor is used for the evaluation of the algorithms. This is due to the lack of robustness in the implemented stereo vision setup which led to a decrease in image quality some time after calibration. However, the algorithms are developed for RGB-D images obtained with any sensor technology.

4.1 Sensor

After a lot of research it was decided to use a RGB-D image because it provides a lot of information and can be obtained with cost-efficient technology. The depth map is considered suitable for classification as it is invariant to color and different lighting conditions, thus providing general information.

There are some different technologies available that generates depth maps. Two technologies were chosen for further evaluation, Kinect and stereo vision. LADAR was discarded as it was considered too expensive.

Other technologies that were researched includes thermal imaging, IR-flash, conventional RGB, radar and ultrasound.

A major flaw with several of the other researched methods are that they are not invariant to variations in conditions that are very common. Thermal imaging, for instance, will provide very different contrast depending on the temperature and a conventional RGB-camera will provide different contrast under different lighting conditions.

The reviewed articles that used the IR-flash method all assumed that pedestrians wears reflex vests. This was considered a too strict restriction.

Radar and ultrasound would have been very challenging and a lot of time would have been spent on just getting the sensors to work, and did not seem particularly promising for this application.

4.1.1 Sensor Comparison

The results from the comparison between the Kinect sensor and the stereo vision system is shown in Figure 4.1.1 and Table 4.1.1. The Kinect sensor was much easier to use and well suited for the indoor environment used when evaluating algorithms and is therefore used in the rest of the comparison between features and classifiers.

The stereo vision system setup, shown in Figure 4.1.2, was a fast and cheap setup which can be improved. It would probably be an interesting way of proceeding when the application is developed for outdoor use. There are some embedded solutions for stereo vision which could possible provide a high quality depth map.

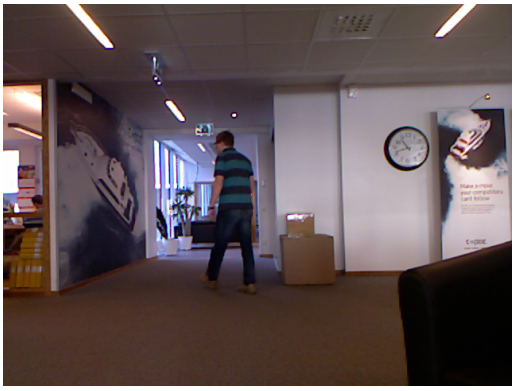
It might seem contradictory that the Kinect was chosen even though it does not work outdoors and the long-term objective is to use the application outdoors. This was mainly to be able to start with the classification software development in an early stage. It was also argued that it is easy to switch to another depth map-providing technology when the *proof of concept* application works satisfactory indoors.

Table 4.1.1: A short list of advantages and disadvantages with Kinect and the implemented stereo vision system.

Kinect		Stereo Vision	
Advantages	**Disadvantages**	**Advantages**	**Disadvantages**
Night vision	Does not work in daylight	Works in daylight	Light dependent
Stable quality	Limited range, up to 9 meters	Works at long distance	Unstable quality of image
Sharp image	Problems with reflecting materials	Can see through glass	Needs calibration
			Texture dependant
			Computational heavy

(a) One of the two cameras from the stereo vision setup.

(b) The depth map created from the stereo vision.



(c) Color image from Kinect.

(d) The depth map created from the Kinect.

Figure 4.1.1: A comparison between depth maps from Kinect and stereo vision. The depth maps are color coded to make them more observable and the black regions mark areas for which the depth is undefined. The advantages and disadvantages are very visible here. The stereo vision can create a depth map for environments that are far away, consists of reflecting materials or are occluded by glass, these parts will be problematic with Kinect. The lack of texture on the floor makes the stereo vision lose much important information. The biggest difference is the sharpness, where the Kinect provides a much sharper depth map.

(a) The stereo vision setup.

(b) The Kinect setup.

Figure 4.1.2: The two analyzed sensor setups.

4.2 Evaluation of Algorithms

Early in the project the use of heuristical classification algorithms was chosen. This makes it easy to change the classifiers intent by just changing the training data, for instance if classification of humans from a different angle or even classification of other objects would be wanted.

A comparison between this project and other projects is difficult due to differences in circumstances. The following results are therefore mostly interesting to be compared with each other. A comparison with other projects might not yield a fair result.

The results are presented as ROC curves and AUC values which are briefly explained in section 2.3.5. The

results from the evaluation of algorithms are shown in Table 4.2.1. The AUC value should not be mixed up with the accuracy.

Table 4.2.1: Performance of the different features using different classifiers. The result is presented as *mean* ± *standard deviation*. The classifiers are first trained and tested 10 times on 50 % of the test set. These 50 % are randomly chosen each time. *The final classification stage in Figure 3.2.5 is a simple neural network and not a DT. A DT at the final classification stage could not create enough nodes to get a decent AUC.

Feature	**AUC for ANN**	**AUC for kNN**	**AUC for SVM**	**AUC for DT**
Depth Map	0.905 ± 0.0079	0.971 ± 0.0038	0.882 ± 0.0105	0.965 ± 0.0047
DFT	0.922 ± 0.0065	0.944 ± 0.0028	0.940 ± 0.0035	0.939 ± 0.0028
FD	0.994 ± 0.0018	0.991 ± 0.0017	0.981 ± 0.0033	0.994 ± 0.0012
Moments	0.965 ± 0.0024	0.759 ± 0.0099	0.927 ± 0.0059	0.945 ± 0.0043
Haar-like features	0.962 ± 0.0057	0.916 ± 0.0065	0.937 ± 0.0044	0.972 ± 0.0040
Combination of Features	0.997 ± 0.0007	0.996 ± 0.0016	0.997 ± 0.0004	0.995* ± 0.0009

The values in Table 4.2.1 are highly dependent on how the training set and test set are formed. The method to create those are described in section 3.2.5.

4.2.1 Feature Extraction

The performance of the features can be seen in Table 4.2.1. For most features there are at least some parameter that can be tuned to change the performance. The features are explained in section 2.3.3 and some choices of parameters are also stated there.

The overview histogram is not included in Table 4.2.1 because the performance differed greatly depending on distance. The performance was very promising for distances shorter than about 5 m, but got increasingly worse with longer distances. Upon investigation it was realized that the depth resolution for the Kinect is depth dependent which probably is the cause of the depth dependance in performance. The reason why the overview histogram is kept in the report is because it is considered a very interesting feature that could perform very well if it is implemented correctly and the distance resolution problem is solved.

HOG is also not included because it was implemented as an already trained classifier. The performance of HOG on the test set gave an AUC of 0.925 which is a good value to compare with the other features. One should keep in mind that the HOG classifier is trained on a more general data set than the other used classifiers, which are only trained on the data that was generated in this project. The features in the list were the features that performed well and the most successful feature would be the fourier descriptor.

Figure 4.2.1 shows the ROC curves for four different features evaluated with an ANN classifier. It can be seen that the FD feature is the best performing feature of those four.

4.2.2 Classification Method

The four different classification methods used are ANN, SVM, kNN and DT which are explained in section 2.3.4. The kNN classifier is a simple classifier that actually performs quite well. Both kNN and DT were easy to configure. The ANN and SVM classifiers are more advanced than kNN and DT and performs slightly better but is harder to configure.

The result for the different classifiers shown in Table 4.2.1 differs a bit for each feature but is surprisingly similar for the combined version. All classifiers but kNN have several parameters that can be configured. The configurations that generated the results are optimized to some extent but can not be proved to be the optimal configurations. However it seems to be, as Dollár mentions in [16], that the choice of classification method plays very little role.

One of the best performing classifier was the ANN classifier with an AUC of 0.997. To get a better picture of how the classifier actually performs, one can look at the following example: Consider the ROC curve of the classifier as seen in Figure 4.2.2. The marked point of the curve have three values; X = 0.002, Y = 0.94 and Z = -0.4 which are false positive rate, true positive rate and the threshold. This means that 0.2 % of the false images are classified as true and 94 % of the true images are classified as true when the threshold is set to -0.4. The confusion matrix for the this threshold can be seen in Figure 4.2.3.

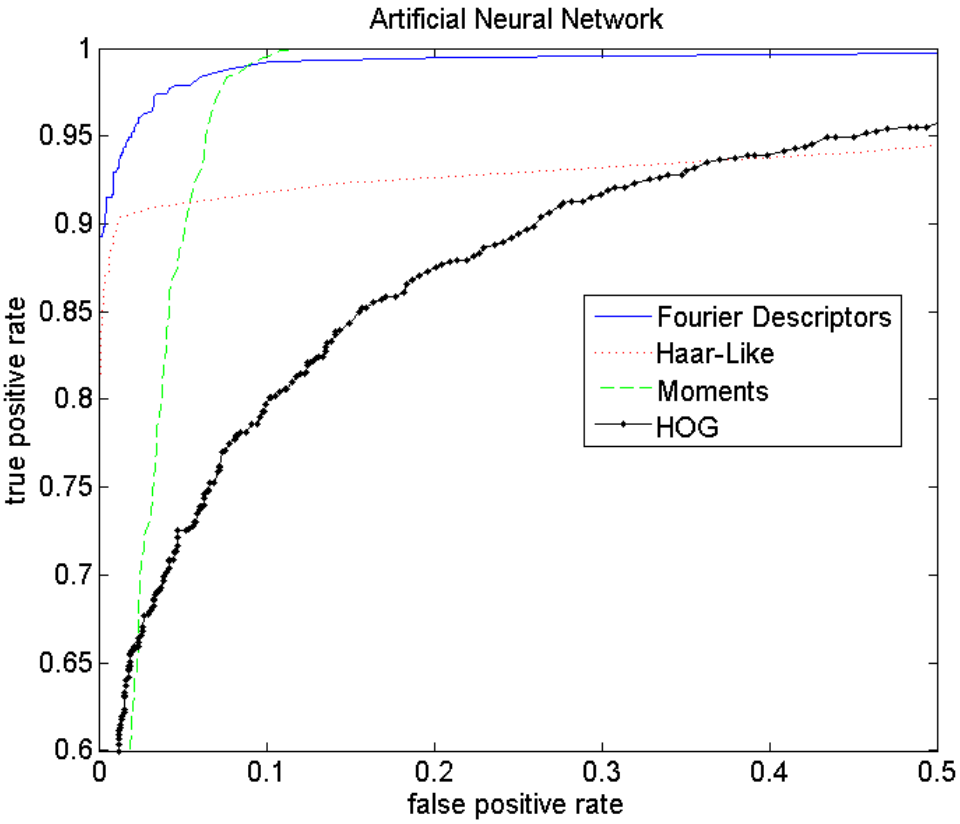


Figure 4.2.1: Four ROC curves for HOG and the three best performing features for the ANN. These features are FD, moments and Haar-Like features. The graph is zoomed in to make it easier to observe the differences in the curves.

Consider that the segmentation will find about 200 negative segments each frame and the frame rate is 10 *frames per second* (FPS). This would be a total of 2000 negative images each second which would lead to about four missclassified images each second. If a human appear on in the environment the segmentation would probably have at least four segments of the human due to the overlap. This means that the probability to find a human where four segments are found would be $1 - (1 - 0.94)^4 = 0.9999870$ if these four segments would be independent. Of course many segments of the same human taken at the same time are not independent but the chance to find at least one of human in several segments is greatly increased. The conclusion of this is that the threshold could be increased to decrease the amount of false positives while still have a very high detection rate.

4.3 Proof of Concept

The display for the *proof of concept* is shown in Figure 4.3.1 and the setup can be seen in Figure 4.3.2.

An important aspect when it comes to the proof of concept is the speed of the classifier. All classifiers where analyzed with the combined feature setup and the FPS differed. DT was the fastest and achieved 13 FPS, ANN had 12 FPS, SVM had 4 FPS and the kNN had 0.5 FPS for k = 30 when running on a 2.2 GHZ Intel i7 quad core processor. Only minor optimization, such as multi-threading, has been done since a speed over 10 FPS was satisfying and the main focus was the classification performance.

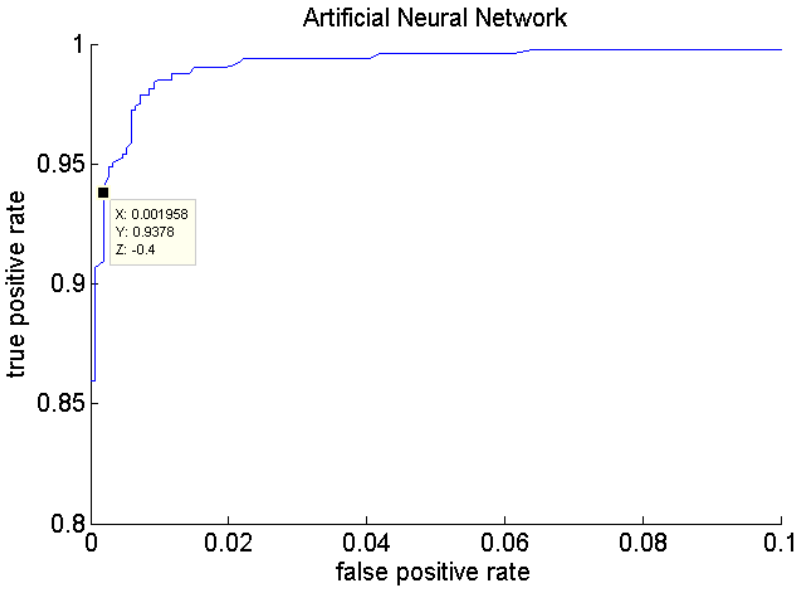


Figure 4.2.2: The figure shows a portion of the ROC curve for the ANN classifier when combining all features in Table 4.2.1. The X and Y value are the corresponding axes and the Z value represents the threshold at the given point.

		True Human: Positive	True Human: Negative
Hypothesized Human	Positive	**754**	**3**
	Negative	**50**	**1529**

Figure 4.2.3: The confusion matrix for the marked threshold of the ROC curve in Figure 4.2.2. The accuracy would be $(1529 + 754)/(1532 + 804) = 97,7\%$. It may be more interesting to decrease the false positive than to decrease the false negative due to several possible hits on each human.

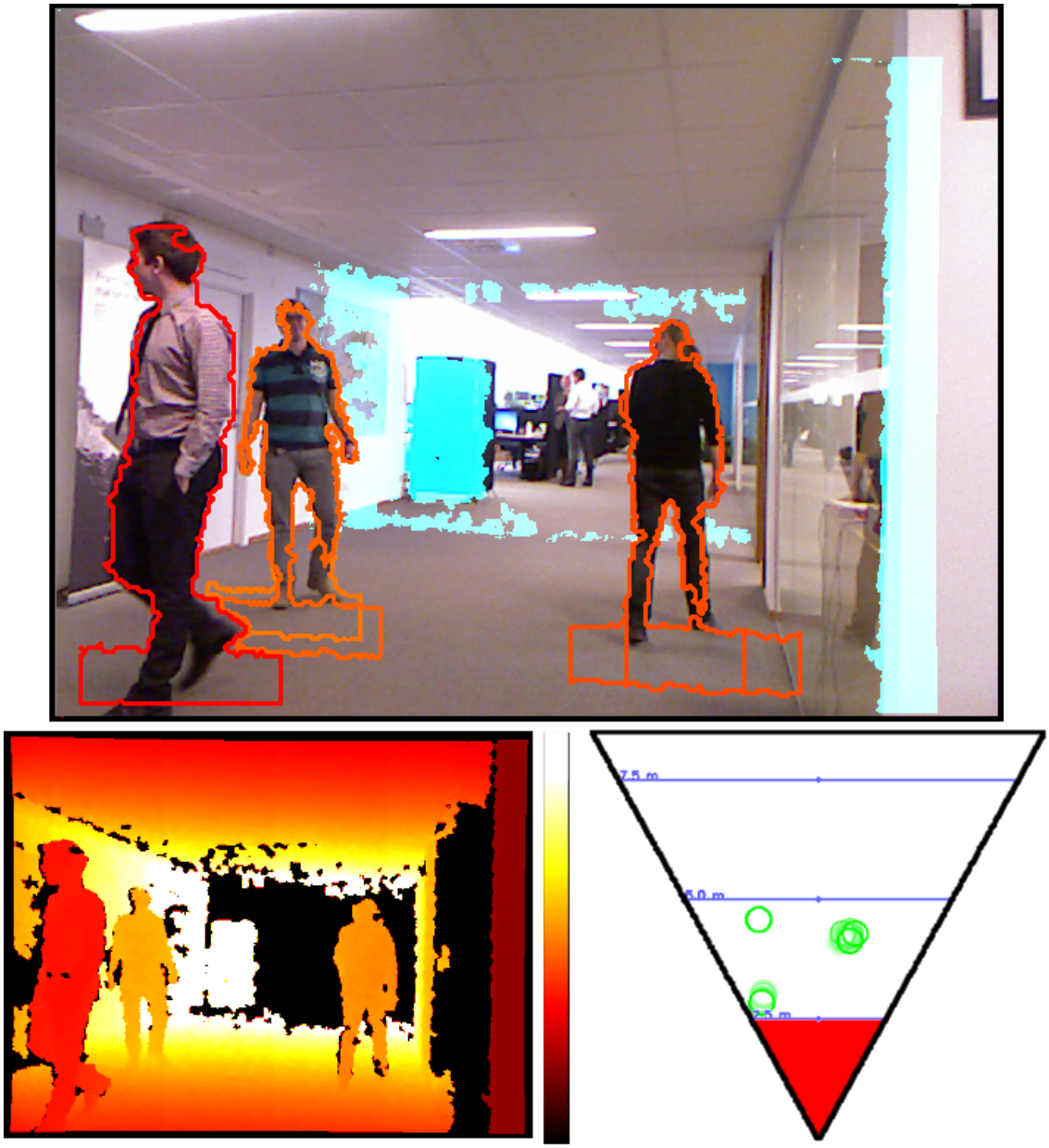
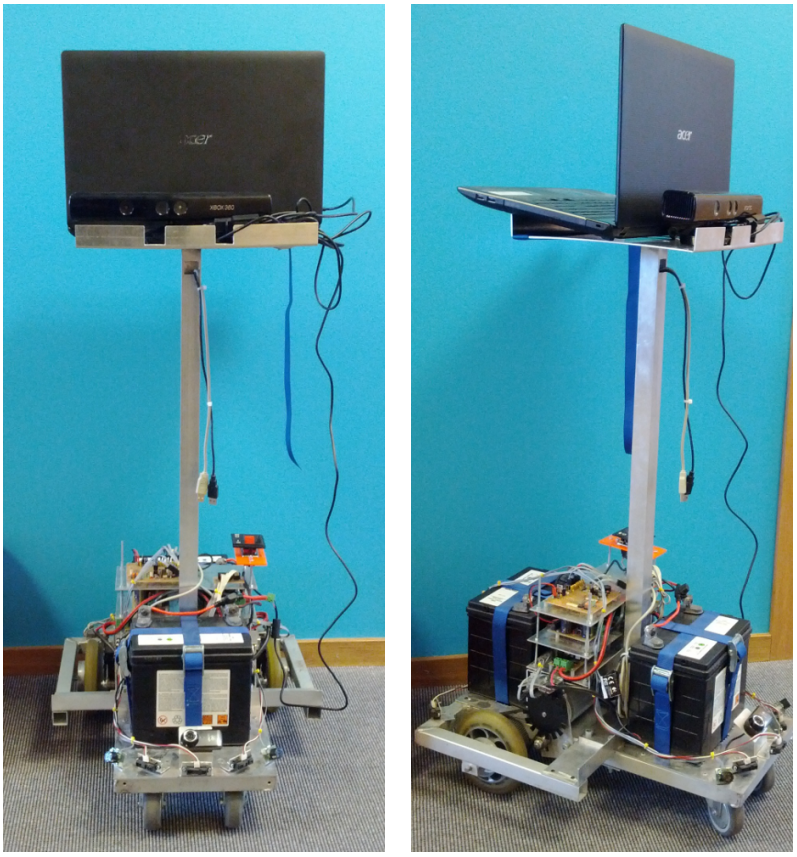


Figure 4.3.1: The figure shows a screen capture of the *proof of concept* display. In the top image, found humans are marked with their contour in a color corresponding to their distance from the camera. The color coded depth map is shown in the bottom left image and an overview position display is shown in the bottom right image. The color bar in the middle corresponds to the distances shown in the position overview. Marked distances are 2.5 m, 5 m and 7.5 m respectively. The red region in the overview and teal regions in the top image are outside of the detection area.



(a) (b)

Figure 4.3.2: The setup for a *proof of concept* system. The robot was constructed in another project but was used to make the detection system mobile.

5 Conclusions and Future Work

This chapter contains the drawn conclusions and answers to the problem definition. It also contains suggestions for future research.

5.1 Sensor

The conclusion from analyzing the Kinect and stereo vision was that a sensor that provides depth information is very suitable for human detection as it gives general color-independent shape information.

It was found that the Kinect was a more appropriate technology to use in this report. But a deeper investigation in a stereo vision system would be very interesting since it would enable the system to work outdoors where the final product is supposed to be used. The stereo vision system used in this report were created with two web cameras, these could be exchanged with a dedicated stereo vision system that could use hardware accelerated algorithms to generate the depth map. That would probably provide a stable enough depth map for the algorithm used in this report and would also give the advantages of a longer range than the Kinect can offer. Another way of creating the depth map would be to use a sweeping laser method. This is probably a more stable way to create the depth map but also a much more expensive solution.

It would be interesting to extend the detection system with an IR-camera and use heat signatures to further sort out false positives.

5.2 Algorithm

The quality of the features have a large impact on detection accuracy. The feature that showed the best individual result was fourier descriptors. Of the used classifiers ANN had the best classification performance while using all features. ANN was also the best classifier when considering classification speed.

There are a lot of possible improvements that could be implemented. Of course there could be even more efficient features waiting to be discovered, it could for instance be interesting to include the distance in some of the features that performs differently for different distances.

The fact that humans are unlikely to move very far between two frames can be used for motion tracking and concentrate the detection around interesting areas and probably prevent a lot of sporadic false positives. Another interesting improvement could be to group several positive hits, in close distance to each other, to one human. As for now, each human can give many hits because of the overlap during segmentation.

5.3 Proof of Concept

A proof of concept prototype have been implemented using the Kinect sensor technology. This prototype shows good results at detecting humans with regard to the limitations. Some modifications would have to be implemented to reach the goal of detecting humans in proximity to construction vehicles. The sensor, for example, would have to be modified to work outdoors or switched to another sensor that works outdoors. Stereo vision could be a very interesting alternative as previously mentioned.

The pose limitation is not realistic at a construction site and the next natural step would therefore be to improve detection in other poses. One possible way of achieving that could be to use multiple detectors that are trained on different poses.

Using multiple detectors for different distances could also be interesting. Experience shows that it is easier to train a detector for a narrow distance interval compared to a general detector.

References

[1] Z.-R. Wang et al. "Pedestrian Detection Using Boosted HOG Features". In: *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*. 2008, pp. 1155 –1160. DOI: 10.1109/ITSC.2008.4732553.

[2] N. Dalal and B. Triggs. "Histograms of Oriented Gradients for Human Detection". In: *In CVPR*. 2005, pp. 886–893.

[3] F. Suard et al. "Pedestrian Detection using Infrared images and Histograms of Oriented Gradients". In: *Intelligent Vehicles Symposium, 2006 IEEE*. 2006, pp. 206 –212. DOI: 10.1109/IVS.2006.1689629.

[4] V. Vaidehi et al. "Multiclass object detection system in imaging sensor network using Haar-like features and Joint-Boosting algorithm". In: *Recent Trends in Information Technology (ICRTIT), 2011 International Conference on*. 2011, pp. 1011 –1015. DOI: 10.1109/ICRTIT.2011.5972251.

[5] W. Yongzhi et al. "Pedestrian Detection Using Coarse-to-Fine Method with Haar-Like and Shapelet Features". In: *Multimedia Technology (ICMT), 2010 International Conference on*. 2010, pp. 1 –4. DOI: 10.1109/ICMULT.2010.5630446.

[6] E.-J. Choi and D.-J. Park. "Human detection using image fusion of thermal and visible image with new joint bilateral filter". In: *Computer Sciences and Convergence Information Technology (ICCIT), 2010 5th International Conference on*. 2010, pp. 882 –885. DOI: 10.1109/ICCIT.2010.5711182.

[7] H. Sun, C. Wang, and B. Wang. "Night Vision Pedestrian Detection Using a Forward-Looking Infrared Camera". In: *Multi-Platform/Multi-Sensor Remote Sensing and Mapping (M2RSM), 2011 International Workshop on*. 2011, pp. 1 –4. DOI: 10.1109/M2RSM.2011.5697384.

[8] H. Andreasson, R. Triebel, and A. J. Lilienthal. "Vision-based People Detection Utilizing Reflective Vests for Autonomous Transportation Applications". In: *IROS Workshop on Metrics and Methodologies for Autonomous Robot Teams in Logistics (MMART-LOG)*. 2011.

[9] L. E. Navarro-Serment, C. Mertz, and M. Hebert. "Pedestrian Detection and Tracking Using Three-dimensional LADAR Data". In: *The International Journal of Robotics Research* 29.12 (Oct. 2010), pp. 1516–1528. DOI: 10.1177/0278364910370216. URL: http://dx.doi.org/10.1177/0278364910370216.

[10] L. Spinello and K. O. Arras. "Leveraging RGB-D Data: Adaptive Fusion and Domain Adaptation for Object Detection." In: *Proc. of The International Conference in Robotics and Automation (ICRA)*. 2012.

[11] M. Luber, L. Spinello, and K. O. Arras. "People tracking in RGB-D data with on-line boosted target models". In: *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*. 2011, pp. 3844 –3849. DOI: 10.1109/IROS.2011.6095075.

[12] A Howard et al. "Detecting Pedestrians with Stereo Vision : Safe Operation of Autonomous Ground Vehicles in Dynamic Environments". In: *System* (2007). URL: http://www-robotics.jpl.nasa.gov/publications/Andrew_Howard/howard_isrr07.pdf.

[13] M. Bertozzi et al. "Stereo Vision-based approaches for Pedestrian Detection". In: *PROCEEDINGS OF IEEE INT. CONF. ON COMPUTER VISION AND PATTERN RECOGNITION*. 2005, pp. 23–28.

[14] L. Zhao and C. Thorpe. "Stereo and Neural Network-based Pedestrian Detection". In: *IEEE Transactions on Intelligent Transportation Systems* 1.3 (2000), pp. 148 –154.

[15] P. Dollár et al. "Pedestrian Detection: An Evaluation of the State of the Art". In: vol. 99. PrePrints. 2011. DOI: http://doi.ieeecomputersociety.org/10.1109/TPAMI.2011.155.

[16] P. Dollár. *Pedestrian Detection: The State of the art*. Video seminar (http://research.microsoft.com/apps/video/?id=135046). [Online; accessed 5-June-2012]. 2010.

[17] S. Kamijo, K. Fujimura, and Y. Shibayama. "Pedestrian detection algorithm for on-board cameras of multi view angles". In: *Intelligent Vehicles Symposium (IV), 2010 IEEE*. 2010, pp. 973 –980. DOI: 10.1109/IVS.2010.5548113.

[18] L. Zhang and Y. Liang. "Motion Human Detection Based on Background Subtraction". In: *Education Technology and Computer Science (ETCS), 2010 Second International Workshop on*. Vol. 1. 2010, pp. 284 –287. DOI: 10.1109/ETCS.2010.440.

[19] R. C. Gonzales and R. E. Woods. *Digital Image Processing*. Pearson International, 2008, pp. 115, 205–255, 394–456, 839–840. ISBN: 978-0-13-505267-9.

[20] Dimenco. http://www.dimenco.eu/2dz/. [Online; accessed 23-April-2012]. 2011.

[21] H. Levkowitz and G. T. Herman. "Color Scales for Image Data". In: *IEEE Comput. Graph. Appl.* 12.1 (Jan. 1992), pp. 72–80. ISSN: 0272-1716. DOI: 10.1109/38.135886. URL: http://dx.doi.org/10.1109/38.135886.

[22] PrimeSense. http://www.primesense.com/en/technology/115-the-primesense-3d-sensing-solution. [Online; accessed 16-April-2012]. 2011.
[23] A. Fusiello et al. *A Compact Algorithm for Rectification of Stereo Pairs*. 1999.
[24] M. Gosta and M. Grgic. "Accomplishments and challenges of computer stereo vision". In: *ELMAR, 2010 PROCEEDINGS*. 2010, pp. 57 –64.
[25] N. Kokash. *An introduction to heuristic algorithms*. 2005.
[26] S. B. Kotsiantis. *Supervised Machine Learning: A Review of Classification Techniques. Informatica 31:249–268*. 2007.
[27] M. Wahde. *Biologically Inspired Optimization Methods*. WIT Press, 2008. ISBN: 9781845641481.
[28] R. Laganiere. *OpenCV 2 Computer Vision Application Programming Cookbook*. Packt Publishing, 2011, pp. 182–185, ISBN: 978-1-849513-24-1.
[29] K. G. Derpanis. *Integral image-based representations*. Tech. rep. Department of Computer Science and Engineering York University, 2007.
[30] M. Jie et al. "Fast Fourier Descriptor Method of the Shape Feature in Low Resolution Images". In: *Wireless Communications Networking and Mobile Computing (WiCOM), 2010 6th International Conference on*. 2010, pp. 1 –4. DOI: 10.1109/WICOM.2010.5601317.
[31] Willows Garage. http://opencv.willowgarage.com/documentation/cpp/structural_analysis_and_shape_descriptors.html. [Online; accessed 27-April-2012]. 2012.
[32] R. Lienhart. *An Extended Set of Haar-like Features for Rapid Object Detection*. Tech. rep. Intel Labs Intel Corporation Santa Clara USA, 2002.
[33] T. Kim et al. *Pose Robust Human Detection in Depth Image Using Four Directional 2D Elliptical Filters*. Tech. rep. Department of Computer Science, Engineering Pohang University of Science, and Technology, 2008.
[34] L. Rokach and O. Maimon. "Top-down induction of decision trees classifiers - a survey". In: *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 35.4 (2005), pp. 476 –487. ISSN: 1094-6977. DOI: 10.1109/TSMCC.2004.843247.
[35] S. Thirumuruganathan. http://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/. [Online; accessed 26-April-2012]. 2010.
[36] C. J. Burges. "A Tutorial on Support Vector Machines for Pattern Recognition". In: *Data Mining and Knowledge Discovery* 2 (1998), pp. 121–167.
[37] S. Haykin. *Neural Network and Learning Machines*. Pearson International, 2009, pp. 40–42,51–52. ISBN: 978-0-13-129376-2.
[38] T. Fawcett. *ROC Graphs: Notes and Practical Considerations for Data Mining Researchers*. 2003.
[39] Microsoft. http://emea.microsoftstore.com/UK/en-GB/Microsoft/Design-+-Developer/Visual-Studio-2010. [Online; accessed 27-April-2012]. 2012.
[40] The MathWorks, Inc. http://www.mathworks.se/. [Online; accessed 27-April-2012]. 2012.
[41] Willows Garage. http://www.willowgarage.com/pages/software/opencv. [Online; accessed 27-April-2012]. 2012.
[42] DotNetNuke Corporation. http://75.98.78.94/default.aspx. [Online; accessed 27-April-2012]. 2012.