HALMSTAD UNIVERSITY

Master's Programme in Embedded and Intelligent Systems, 120 credits

MASTER THESIS

Robotic First Aid:

Using a mobile robot to localise and visualise points of interest for first aid

Wolfgang Hotze

Embedded and Intelligent Systems, 30 credits

School of Information Technology, 24th September 2016

SUPERVISORS:
Dr. Martin Cooney
Dr. Anita Sant'Anna

EXAMINERS:
Prof. Antanas Verikas
Dr. Sławomir Nowaczyk

LOCATION:
Kristian IV:s väg 3, 301 18 Halmstad, Sweden

TIME FRAME:
January-September 2016

ABSTRACT

Domestic robots developed to support human beings by performing daily tasks such as cleaning should also be able to help in emergencies by finding, analysing, and assisting persons in need of first aid. Here such a robot capable of performing some useful task related to first aid is referred to as a First Aid Mobile Robot (FAMR). One challenge which to the author's knowledge has not been solved is how such a FAMR can find a fallen person's pose within an environment, recognising locations of points of interest for first aid such as the mouth, nose, chin, chest and hands on a map. To overcome the challenge, a new approach is introduced based on leveraging a robot's capabilities (multiple sensors and mobility), called AHBL. AHBL comprises four steps: Anomaly detection, Human detection, Body part recognition, and Localisation on a map. It was broken down into four steps for modularity (e.g., a different way of detecting anomalies can be slipped in without changing the other modules) and because it was not clear which step is hardest to implement. As a result of evaluating AHBL, a FAMR developed for this work was able to find the pose of a fallen person (a mannequin) in a known environment with an average success rate of 83%, and an average localisation discrepancy of 1.47cm between estimated body part locations and ground truth. The presented approach can be adapted for use in other robots and contexts, and can act as a starting point toward designing systems for autonomous robotic first aid.

ACKNOWLEDGEMENTS

I would like to thank my thesis supervisors Dr. Anita Sant'Anna and Dr. Martin Cooney of the Embedded and Intelligent Systems Department of the School of Information Technology at Halmstad University for their competent advice over the entire course of the project.

I would also like to thank the expert who gave insight in healthcare: Carina Göransson. Without this input towards healthcare the important aspect about localisation could not have been successfully included.

Particular thanks to Nikolas Kratschmann who let me use his images for the mannequin.

In addition, a thank you to my colleague Majid Karimi, BSc who helped me with a couple of experiments.

I would also like to thank my parents, Hedwig and Wolfgang Hotze, who supported me through my study and by sending me lovely food parcels.

Thank you.

Wolfgang Hotze

CONTENTS

List of Figures viii
List of Tables ix
Acronyms x
1 INTRODUCTION 1
1.1 Goal and Approach . . . 2
1.2 Contribution . . . 4
1.3 Thesis Outline . . . 4
2 RELATED WORK 5
2.1 Anomaly Detection . . . 5
2.2 Human Detection . . . 6
2.3 Body Part Detection . . . 7
2.4 Localisation and Visualisation . . . 8
2.5 Summary . . . 8
3 ALGORITHMS 9
3.1 Theory . . . 9
3.1.1 Localisation . . . 9
3.1.2 Image processing . . . 10
3.2 Tools . . . 12
4 DESIGN AND IMPLEMENTATION 15
4.1 Setup . . . 15
4.2 Design . . . 20
4.3 Navigation . . . 23
4.4 Anomaly Detection . . . 25
4.5 Human Detection . . . 28
4.6 Body Part Detection . . . 30
4.7 Localisation and Visualisation . . . 34
5 EVALUATION 37
5.1 Anomaly Detection . . . 39
5.2 Human Detection . . . 41
5.3 Body Part Detection . . . 43
5.4 Localisation and Visualisation . . . 46
6 CONCLUSION 49
6.1 Limitations . . . 50
6.2 Future Work . . . 51

BIBLIOGRAPHY 53

LIST OF FIGURES

Figure 3.1 Rectangle filter types 11
Figure 3.2 Example of a cascade of classifiers 11
Figure 4.1 Component diagram of the modified Turtlebot 16
Figure 4.2 Modified Turtlebot 2 17
Figure 4.3 Intelligent environment as PGM 18
Figure 4.4 Architectural overview 20
Figure 4.5 Software component diagram of the robot . . . 21
Figure 4.6 Software component diagram of the workstation 22
Figure 4.7 RVIZ view of the patrolling path 23
Figure 4.8 Flowchart of the *Navigation Node* 24
Figure 4.9 Flowchart of the *Anomaly Detection Node* 25
Figure 4.10 Flowchart of the *LaserScan-Callback function* . . 26
Figure 4.11 Detected anomaly within the environment . . 28
Figure 4.12 Flowchart of the *Human Detection Node* 30
Figure 4.13 Flowchart of the *Body Part Detection Programme* 31
Figure 4.14 Body part detection frame 34
Figure 4.15 Flowchart of the *Localisation Programme* 35
Figure 4.16 RVIZ view after process 36
Figure 5.1 Standard posture for evaluation 37
Figure 5.2 Advanced posture for evaluation 38
Figure 5.3 Anomaly detection evaluation overview 39
Figure 5.4 False detected anomaly 41
Figure 5.5 Human detection evaluation overview 42
Figure 5.6 Face with different illumination and occlusion 43
Figure 5.7 Body part detection evaluation overview . . . 44
Figure 5.8 Examples of true/false positive face detections 45
Figure 5.9 Localisation evaluation overview 46

LIST OF TABLES

Table 4.1 Threshold measured with a thermometer . . . 29
Table 4.2 Breath sensor measurements 32
Table 5.1 Evaluation of anomaly detection 40
Table 5.2 Evaluation of human detection 42
Table 5.3 Evaluation of body part detection 44
Table 5.4 Evaluation of body part detection II 46
Table 5.5 Evaluation of localisation accuracy 47

ACRONYMS

AHBL Anomaly detection, Human detection, Body part detection, and Localisation and visualisation

AMCL Adaptive Monte Carlo Localisation

API Application Programming Interface

CPR Cardiopulmonary Resuscitation

DOF Degrees of Freedom

EKF Extended Kalman Filter

FAMR First Aid Mobile Robot

FLIR Forward Looking Infra Red

HOG Histogram of Oriented Gradients

HRI Human-Robot Interaction

IMU Inertial Measurement Unit

L-ABCDE Localisation, Airway, Breathing, Circulation, Disability and Exposure

MUX Multiplexer

OpenCV Open Source Computer Vision

OpenNI Open Natural Interaction

PCA Principal Component Analysis

PDF Probability Density Function

PGM Portable Gray Map

ROS Robot Operating System

RVIZ ROS Visualisation

SLAM Simultaneous Localisation and Mapping

SIFT Scale-invariant Feature Transform

UAV Unmanned Aerial Vehicle

URDF Unified Robot Description Format

XML-RPC eXtensible Markup Language - Remote Procedure Call protocol

1 INTRODUCTION

More and more mobile robots are finding their way into human society. Home robots already assist human beings by performing daily tasks at home like Roomba and Looj[1] which clean inside and outside a house. Personal robots can interact helpfully with people, such as Care-O-bot[2] or the Personal Robot[3] that can bring you a cup of coffee or tell you your schedule. Companion robots have been built for the elderly. Additionally, robots could assist with first aid in emergencies in people's homes and public locations when medical personnel are far away or not available, e. g., by monitoring vital signs as Gross et al. described [1]. Robotic assistance in first aid would be important because lives could be saved and damage from emergencies reduced.

In the last years, progress in the fields of robotics and intelligent systems has led to new technologies and methods. Robots are able to overcome more and more tasks like navigating through a known/unknown environment, interacting with a human or analysing their surroundings. Image processing offers several algorithms such as Histogram of Oriented Gradients (HOG) by Dalal and Triggs [2] or Viola and Jones [3] to detect humans and to segment individual body parts such as head, torso and limbs which are required for first aid analysis. Such knowledge can be applied to make progress toward realising robotic first aid.

First aid focuses on critical lifesaving actions. The goal of first aid is described by the 3 P's, to preserve life, prevent injuries and promote recovery. First aid can and should be immediately initiated by anyone, in any situation [4]. The first important step is to understand what happened: where is the person and how is the human being positioned. Blood circulation is checked by detecting the pulse and treatment can involve chest compressions. Also, attention is brought towards the airway to ensure that the person is breathing and nothing obstructs the airways. If that is not the case, the first aider has to provide rescue breathing and remove the blockage. Other steps are to check how responsive the victim is, and to expose wounds and stop any deadly bleeding, although much other knowledge is also important for nurses, such as knowing about previous medical conditions. The steps above form an important base for first aid, also described by the acronym Localisation, Airway, Breathing, Circulation, Disability and Exposure (L-ABCDE) [5, 6].

1 See: https://www.irobot.com
2 See: http://www.care-o-bot-4.de/
3 See: https://www.autonomous.ai/robot

1.1 GOAL AND APPROACH

The aim of this project is to engineer a First Aid Mobile Robot (FAMR). Various kinds of such robots could be built. Some robots have been built which seek to facilitate CPR such as a remote-controlled snake robot built to help soldiers in a battleground [7]. Another example is a flying robot[4] which can be sent to a location of an emergency to provide a defibrillator. But in some cases a human medical expert might not be available to help; therefore we investigate how to build a robot which can operate semi-autonomously.

Such a robot could try to detect a fallen person's location and pose, detect biosignals, or facilitate or perform first aid such as Cardio-pulmonary Resuscitation (CPR). Some previous research has tackled biosignals like Chang et al. developed a system to measure and transmit biosignals of interest for first aid such as electrocardiogram, pulse or blood oxygen content [8] but localisation and CPR require more work (details are provided in Chapter 2). The current project focuses on the first point localisation, because CPR requires the exact location of the human being and it is not clear how a robot at home can find a fallen person's location and pose.

Therefore in the current study, the goal is to create a robot which can detect a person's pose on a map. We investigate where a fallen person is (e. g., in a kitchen) and detect points of interest for first aid, which a robot must detect in order to conduct first aid procedures such as CPR. By points of interest, we mean the mouth, nose, and chin for the airway, and the chest and hands for circulation (it is common to check the pulse on the wrist and a bluing of the hand (also known as cyanosis) which can indicate circulatory problems).

Also, various sensors could be used. Robots usually have multiple sensors, each of them with various strengths and weaknesses and may complement one another. For example, vision sensors are commonly used for recognition but are highly affected by illumination and occlusions. This can mix up objects in homes such as chairs and tables with humans, which can be improved by using other sensors. Therefore, in the current project the following sensors were used: a laser-scan from a depth sensor, a non-contact temperature sensor, a RGB camera (with a laser pointer), and a breath sensor.

Additionally, an intelligent environment which has several sensors installed like cameras and microphones could send a command to a robot to go towards the site of an emergency or a robot could check occasionally itself to see if it detects a fallen person. Currently most human environments are not intelligent; moreover, environmental

4 See: http://www.io.tudelft.nl/onderzoek/delft-design-labs/applied-labs/ambulance-drone/

sensors can be occluded and wearable sensors might not be worn when an emergency occurs. Thus we focused on the basic case of a semi-autonomous robot which can occasionally patrol through a home and find fallen persons, either at certain times or when it is not currently performing a task for the user. CPR is highly difficult even for trained humans, and can result in problems like broken ribs.

By visualising a person's location and pose, a robot could send close-up pictures to a medical expert without the expert having to manually control the robot to find body part locations. Or the robot could help a novice first aider to help the fallen person. For example, the robot could point out where the novice should place their hands and indicate the rhythm for chest compressions. We investigate both possibilities as either could be highly important.

Thus, our goal is to create a robot with various sensors which patrols, finds fallen persons, and detects location and pose, in terms of points of interest which are important for first aid: mouth, nose, chin, chest and hands, and visualises its knowledge in a way which is easy for humans to understand.

To achieve the goal, we designed and implemented an approach which we call AHBL (pronounced like "able") consisting of four steps: anomaly detection, human detection, body part detection, and localisation and visualisation. The mobile robot first navigates through and analyses an environment. The robot requires the capability to check its sensor readings in reference to a static model of the living space to detect anomalous regions or objects which might be detected due to a fallen person. Next the robot checks if a recognised anomaly is a human. Then the robot detects points of interest for first aid. Finally, the detected points are localised on a map. It was broken down into four steps for modularity (e.g., a different way of detecting anomalies can be slipped in without changing the other modules) and because it was not clear which step is hardest to implement. Also, anomaly detection was separated from human detection because non-human anomalies such as fallen objects, water pools, or fires might also be useful for a home robot to detect.

AHBL can be described in terms of first aid as corresponding to the L (Localisation) of L-ABCDE. AHBL allows to localise and visualise the points of interest for first aid (mouth, nose, chin, chest and hands) on the robot's map and also by pointing using a laser carried by the robot. This system brings together previous work by Lazzaro, Zhang and Zhao, and Mayr as will be described in Chapter 2 [9, 10, 11].

1.2 CONTRIBUTION

The field of contribution is a junction between mobile robotics, applied machine intelligence and image processing which combines advantages from robotics such as the ability to look from different positions and angles with various sensors as well as machine intelligence/image processing like software libraries for recognising humans and human body parts to solve a new problem. Those technologies were used to form an approach which was called Anomaly detection, Human detection, Body part detection, and Localisation and visualisation (AHBL).

The novelty is finding a fallen person's pose within an environment, recognising locations of points of interest for first aid on a map, based on levering a robot's capabilities (multiple sensors and mobility). More specifically, the problem seems to be novel (e.g., body parts in a standing human have been localised using Kinect, but body parts on a lying human have not been localised on a map, to the author's knowledge). Points of interest refers to the mouth, nose, chin, chest, and hands on a fallen person. Also the proposed approach AHBL using a mobile robot with a mounted robot arm equipped with a camera, remote thermometer, breath sensor and indicating those with a laser transmitter module seems novel. (The current work will be positioned relative to the literature in Chapter 2.)

1.3 THESIS OUTLINE

The next part of the thesis introduces Chapter 2 which briefly describes the related work regarding anomaly detection, human detection, body part detection and localisation and visualisation which are necessary for L of L-ABCDE. Chapter 3 covers used methods in the areas of navigation and image processing in Section 3.1, followed by Section 3.2 which introduces the utilised frameworks, libraries as well as terminology.

Chapter 4 describes the hardware setup in Section 4.1, followed by the system design in Section 4.2. Next, the implementation, comprising all four AHBL steps: anomaly detection (Section 4.4), human detection (Section 4.5), body part detection (Section 4.6), and localisation and visualisation (Section 4.7).

The system is evaluated by measuring all AHBL steps as shown in Chapter 5. Chapter 6 discusses results and clarifies the limitations and explains the future work.

2 RELATED WORK

In order to clarify the novelty of the current project, this chapter describes the formulated problem and positions the current study in regard to some previous studies which relate to each component of the system.

This project addresses a new problem of robotic localisation of human body parts important for first aid on a detected fallen person. The localisation requires various techniques in the fields of robotics and computer vision which have to be combined; in the recent past, it has been that inexpensive mobile robot platforms and free software tools such as ROS and OpenCV with functions for navigating and detecting objects have made it increasingly feasible to tackle this kind of problem. For example, to perform chest compressions a robot must be able to find a person's chest. Even though separate techniques exist to find body parts like face, nose or chest, this problem has not been addressed with the junction of robotics, computer vision and healthcare for first aid in real human environments.

The approach in the current project AHBL combines some previous techniques with heuristics. Although the fourth component (Localisation) seems to have been dealt with in the least by previous studies, we feel the novelty is not in one part of the system, but in the system as a whole. Below some studies related to each system component are described.

2.1 ANOMALY DETECTION

Some studies have described powerful approaches to anomaly detection. In order to detect an anomaly, a robot can compare what it senses with a known model to detect unknown information.

Hornung and his colleagues used simulations to show a model-free approach for detecting anomalous events [12]. Suzuki et al. reported on a mobile news finding robot which used a bag of various features and cameras to detect anomalies at human height [13]. Ma used a mobile robot with a camera to detect anomalies on the top of the robot [14]. Also, an approach was described for using multiple robots with cameras to investigate an anomaly projected onto a floor in a lab setting and simulation by Saldana et al. [15].

These studies are capable of recognising various kinds of anomalies but may have some shortcomings for the current application: they use RGB cameras which may have problems with illumination, some use simulations which may not represent the real world, some

detect anomalies at the height of a standing robot and are not typically capable of detecting anomalies on a floor. In the current project, a simple approach using a low-height laser scan and a priori knowledge (a map of the environment) is used which is robust to changes in illumination and can detect anomalies near a floor.

2.2 HUMAN DETECTION

Some previous studies have sought to classify if an object is a human being. It can be a challenging task to distinguish between a chair and a human being, or a pillow and a human body. It has to be clarified that in the last years, research in the field of deep neural networks showed, e. g., procedures for object detection by Szegedy and Toshev [16] or human detection [17] by Kim and Moon that are more likely capable of distinguishing between a pillow and a human body. Human detection is an ongoing research area which uses different technologies like radar, images, video, or thermal sensors.

Shingu et al. conducted a simulation proposing how humans could be detected with Ultra-wideband radars in a home [18]; absorption characteristics could discriminate between a human and a desk, but in the simulation, a number of radars were used in the environment, which is not currently common in human homes. Other studies have investigated pedestrian detection from images like Stoble and Sreeraj [19]; Zheng and Chen provide a review [20]. Lazzaro described a way to detect a human with captured images from a Kinect device by combining several methods like thresholding, blob analysis or chamfer distance matching [9].

A drawback is that visual recognition can be difficult because objects in human environments such as pillows or posters may have complex designs and even pictures of humans on them and problems can occur due to illumination. Using videos, common techniques like background subtraction can be used to detect moving humans. A short-coming is that motion is not feasible when a human is unconscious, and possibly not breathing.

Two studies have used combinations including thermal sensors. One example is to find and detect injured humans outdoors in a certain area to deliver medical supplies by an Unmanned Aerial Vehicle (UAV) using boosted classifiers with Haar-like features with thermal and colour video sources by Rudol and Doherty [21]. A similar idea to our project was presented by Correa and his colleagues to detect a human by visual and thermal skin detection, human body detection, face detection and recognition as well as decision-making for the robot's navigation [22]; the work focused on detecting and identifying humans in an environment, but did not focus on first aid.

In the current project, a similar but simplified approach is used using an inexpensive thermal range sensor and experimentally determined thresholds.

2.3 BODY PART DETECTION

The current work aims to detect facial parts such as mouth, nose, and chin, as well as chest and hands with the use of a mobile robot. Some previous studies have used environmental or on-robot sensors to detect human body parts or full-body poses from different types of data: single images, multiple images, depth, breath, and microphones.

Viola and Jones developed a real-time detection algorithm which can detect faces and body parts in single images [3]. Additional work was done to recognise face parts like nose and mouth by Maghraby et al. [23] as well as to detect the chin by Hu et al. [24]. Another method is capable of detecting extremities from single images as Wang, Zabir and Leibe illustrated [25]. Srivastava and his colleagues dealt with the rotation of faces on multiple images [26]. Ibrahim et al. used the skin tone colour of the face for skin detection [27].

A study used a robot's navigation system to obtain multiple views of a human, which allowed a body part based detection by Zivkovic and Krose [28]. A study by Shotton and his colleagues used depth images as an input signal that can be used for real-time detection methods for human body parts [29]. Also, body parts can be inferred from the actual pose of a human as a proposed technique introduces which estimates the pose of a human being with high precision by Toshev and Szegedy [30].

Breath sensors could be highly useful as they can operate even in noisy environments and have even been used to detect difficult health problems such as colorectal cancer as Meij et al. described [31]. Avalu worked on human breath detection using a microphone [32]. Lagido et al. worked on a camera-based monitoring of heart rate and rhythm [33].

In previous studies, subjects had to stand in front of the Kinect to detect body parts. The face, extremities or torso are parts that an algorithm can recognise from a video stream, but there are factors that can make the process more difficult like the body shape, various clothing or the brightness level in the captured frame. Environmental sensors cannot deal with occlusions. Also, it is difficult to detect body parts robustly with only one sensor.

Thus in the current project, we use a robot with an assortment of different sensors such as camera, thermal, and breath to detect body parts from a low position. We are not aware of any system which has combined these sensors to perform this task.

2.4 LOCALISATION AND VISUALISATION

Some previous studies have located people on a map. One example is finding injured humans outdoors in a certain area to deliver medical supplies by an Unmanned Aerial Vehicle (UAV) using Haar-like features with thermal and colour video sources as Rudol and Doherty described [21]. Different approaches have been developed to localise a human body such as an omnidirectional visual system in combination with a shape-based human body detector as described by Liu et al. [34] or to use an Extended Kalman Filter (EKF) to localise landmarks and distinguish between a human and an object via HOG as described by Luo et al. [35].

Pointing can also be used to localise. Holladay et al. suggested how a robot can point in a clear, legible manner to one of two objects [36]. A recent study showed a human tracking method with an interactive robot which can also perform pointing gestures [37]. Chadalavada and his colleagues developed a robot which uses a projector to show humans who wish to cross a robot's path that the robot has seen them by visualising a zebra crossing on the floor [38]. LED displays have been mounted on flying robots to show an intention to turn by Szafir et al. [39].

These studies used different methods to localise but did not focus on first aid. Knowing a position on a map or silhouette is not enough for a robot to know where a body part is, toward being able to perform first aid. And, pointing by itself would not be sufficient to show where a point of interest is, such as where a human should place their hands to perform chest compressions (e. g., two fingers above the sternum).

Therefore in the current study, the robot produces a map with the locations of body parts important for first aid marked on the map. And the robot uses a laser pointer to produce a small dot which can be used to clearly indicate a specific location.

2.5 SUMMARY

Maybe the closest work to the current one is the one conducted by Mayr, that describes a robot which could be commanded to move to a location, detect the face of a sitting person, and ask if the person was alright [11]. Some large differences exist with the current project: we report on a robot which can by itself detect fallen persons via size and heat, and from multimodal data gained while navigating, localise important points for first aid.

The novelty in the current project is the combination and adjustment of previous techniques in applied robotics, computer vision, and healthcare, to create a system capable of robotic localisation of first aid points of interest.

3 ALGORITHMS

This chapter describes the theory by illustrating the utilised methods in the fields of robotic localisation and image processing. Additionally, several tools are introduced that were used in the current project. Furthermore, necessary terminology is clarified in this chapter.

3.1 THEORY

This section covers the utilised navigation techniques to localise the robot in a known environment. Moreover, image processing methods are described such as Viola-Jones for face detection, Principal Component Analysis (PCA) for orientation of an object, and Histogram Back Projection for skin detection.

3.1.1 *Localisation*

Kalman filters represent the state of a system by using a single modal hypothesis tracking algorithm. Such a filter can provide optimal solutions for problems which can be modelled simply with single Gaussians. The Kalman filter describes the state of the systems with single Gaussians. Some models might require a high computing power or a simplified model that does not depict the problem; rather it generates significant errors. For example, if a Kalman filter represents a localisation system for a robot and has to choose between two different locations. Such a system could choose the wrong state and could hardly recover from the mistake. Another filter for localisation is the particle filter, which could yield a better solution, in the situation described before, than the Kalman filter.

PARTICLE FILTERS like Kalman filters combine different information from different sources and track the state of a dynamic system from sensor measurements. These filters have similar applications than Kalman filters, but can be used for more dimensional problems/hypotheses. However, the solution is an approximation of the problem and, therefore, not as accurate as the Kalman filter. Particle filters are represented as a Probability Density Function (PDF). A single particle is a predicted value. The algorithm starts with an initial PDF which is e. g., represented by stored values from the last iteration or by using random particles like gaussian distribution. After the initialisation, the algorithm iterates through the prediction, update and resample phase for each time interval.

The equation below calculates the belief of the different states, e. g., for a robot localisation:

$$\underbrace{p(x_t \mid d_{0\ldots t})}_{\substack{\text{Resample:}\\ \text{p(x) - the}\\ \text{posterior}\\ \text{probability}}} = \underbrace{\eta\, p(z_t \mid x_t)}_{\substack{\text{Update:}\\ \text{w(x) - the}\\ \text{importance}\\ \text{weights}}} \underbrace{\int (p(x_t \mid u_{t-1}, x_{t-1}) p(x_{t-1} \mid d_{0\ldots t-1}))\, dx_{t-1}}_{\substack{\text{Prediction: q(x) -}\\ \text{the prior probability}}} \tag{3.1}$$

where $p(x_t \mid d_{0\ldots t})$ represents the PDF of the current state, x is the state variable, d represents the data consisting of inputs and observations, η is a normalisation constant, $p(z_t \mid x_t)$ is the perception model, z represents the observations, $p(x_t \mid u_{t-1}, x_{t-1})$ illustrates the motion model, u represent the inputs, $p(x_{t-1} \mid d_{0\ldots t-1})$ represents the PDF from the last time step.

The particle filter calculates the equation from right to left. Starting with the prediction step which iterates through each particle and adds a random sample from the motion model. This procedure results in cloud of particles based on how the robot is thought to have moved. Next, the algorithm adjusts weights of each particle based on sensor measurements like a laserscan. After which resampling occurs which creates a new set of particles by choosing the particles in proportion to their weights. Therefore, highly unlikely particles are removed [40].

3.1.2 *Image processing*

VIOLA-JONES describes an object detection algorithm which as one application also has been used to detect faces. The algorithm utilises rectangle filters that are used to detect objects based on patterns like in a human dark eyebrows or different intensities such as above and below the upper and lower lip. Three different filter types of rectangles are shown in Figure 3.1. These three rectangles are used to build a library of filters by changing the position, scale, and type. The scaling factor represents a scalar with which the filter can be stretched or shrunk in every iteration. These filters are shifted over the image and the sum of the white rectangles are subtracted from the sum of the grey rectangles.

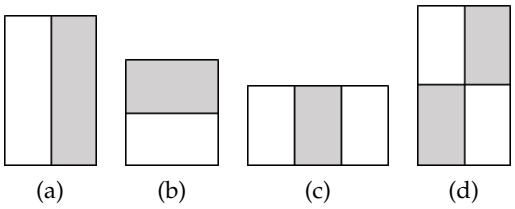


Figure 3.1: Rectangle filter types; (a) and (b) are two-rectangle filter, (c) a three-rectangle filter and (d) a four-rectangle filter; also called Rectangular Haar-like features used in object recognition

The rectangle filters can be computed by using an integral image. An integral image is used to speed up calculations and is a representation of the sum of the pixels above and to the left of the original image as followed:

$$ii(x, y) = \sum_{x' \leqslant x, y' \leqslant y} i(x', y'), \tag{3.2}$$

where $ii(x, y)$ is the resulting integral image, the location at x, y represent the sum of the upper and left pixels of x, y and $i(x', y')$ is the original image.

Adaboost finds weak classifiers, one by one. At each iteration a weak classifier is selected which has an accuracy of at least 50%. Then the misclassified samples are given higher weights, and another different weak classifier is found, until a halting criterion is reached. Multiple small adaboost classifiers with only one or two features are arranged in a cascade model for speed. A positive detection triggers the evaluation of a second classifier. A negative outcome results in a non-face detection. This leads to chain classifiers that are more complex, but have a lower false positive rate. Figure 3.2 illustrates this process.

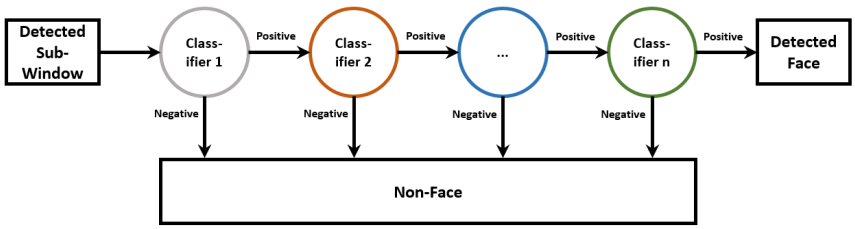


Figure 3.2: Cascade of classifiers, which are constructed by training classifiers using *Adaboost*.

In summary, the algorithm uses rectangle filters, integral images for fast computation, Boosting for selecting the rectangle filters by minimizing the error and cascade classifiers for fast rejection of negative windows [3].

PRINCIPAL COMPONENT ANALYSIS is a statistical analysis, which converts a set of observations into a set of vectors, called principle components. The procedure simplifies the model by using a linear transformation. The first principal component has the the largest variance value. Further components are only kept, if they are orthogonal to the previous one and the greatest possible variance. PCA is a technique that, e. g., can be used for image processing or pattern recognition [41].

HISTOGRAM BACK PROJECTION is a technique that calculates the histogram model of a given Region of Interest (ROI) and uses this model to find similar regions within an image. Such a region could be the skin colour of a human being. The method uses a window which only contains the region that should be extracted. The model is now normalised, therefore each pixel of the original window correlates to a bin in the histogram model. The value of the bin is saved in a new image, which represents the probability that a pixel in the other image is associated with the ROI [42].

3.2 TOOLS

This section introduces the utilised software frameworks ROS and OpenCV. In addition, the Arduino tool is described, followed by the establishment of ROS terminology.

ROBOTIC OPERATING SYSTEM is a development framework for robotic applications and therefore offers an assortment of tools, libraries, and conventions. The framework supports the Linux distributions Ubuntu and Debian and offers an experimental version for Mac OS X and Gentoo. The ROS distribution is in its current tenth development cycle. The latest release is called Kinetic Kame. All versions provide functionalities and code examples for common tasks like localisation and navigation. The framework allows access to sensors and actuators for creating software which can be shared with the community. The framework is written in C/C++ and provides interfaces for C, C++, Python, and Lisp [43].

OPEN SOURCE COMPUTER VISION is an open source computer vision framework which runs on Windows, Linux, Android as well as Mac OS and is written in C/C++ [44]. The framework is used together with ROS and Boost to develop a real-time image processing application for mobile robots in the current project.

ARDUINO is an open-source electronics platform based on hardware and software which is accessible and modifiable, e. g. adding additional sensors or adapters [45]. The platform is used to access and control the robot arm as well as the equipped electronics.

ROS TERMINOLOGY

ROS Node: is an executable that utilises ROS to communicate with other nodes via ROS topics.

ROS Topic: is used to publish or subscribe to specific channels, therefore information can be exchanged via messages.

ROS Messages: contains information in the form of a data type that can be sent or received via ROS topics.

ROS Launch File: is a valid XML file to launch multiple ROS nodes. Such a file can be used to start certain nodes with specific parameters.

ROS Master: offers a naming and registration service which is utilised by all ROS nodes to locate one another via publishers and subscribers. The ROS Master also provides the Parameter Server.

Parameter Server: is a shared, multi-variate dictionary to store and enquire parameters at runtime. For example, Kinect parameters or private parameters for naming conventions.

Rosout: is a console log that consists of nodes and topics.

RVIZ: ROS Visualization is a 3D tool for displaying inter alia: the environment, robot models, sensor data, state information or path planning [46].

4 DESIGN AND IMPLEMENTATION

In order to illustrate the system of the current project, this chapter initially describes the hardware setup, followed by the overall design of the system. Then the implementation is described for each software component.

4.1 SETUP

This section covers the used hardware such as the mobile robot, robot arm, electronics, and the environment the robot used in this project. Furthermore, the firmware for the robot arm and the Turtlebot launch process are described.

HARDWARE: This project's main component is a modified Turtlebot 2 composed of a Kobuki base[1], a Kinect device[2] and a netbook. The first component consists of a differential drive system with one set of wheels, a gyroscope, and an accelerometer. The Kinect is used to get real-time data of its current surroundings and is used in conjunction with the Open Natural Interaction (OpenNI) framework as a ROS package. In the current project, the Kinect device was lowered for a better detection of fallen persons. The netbook operates on Ubuntu 12.04 with the framework ROS Hydro and the libraries Boost with version 1.48 and OpenCV 2.4.9.

Additionally, a 5-DOFs robot arm[3] was mounted on top of the Turtlebot as well as components to control the arm by utilising two Adafruit Motor shields[4] and an Arduino board[5]. The joint angles were measured via potentiometers. The robot arm consists of a base, an elbow, wrist and a gripper.

Further electronics were attached to the arm like a temperature and breath sensor as well as a laser pointer module. The first one is an infrared thermometer for non-contact ambient and object measurements in the sight of the sensor [47]. The gas sensor is capable of detecting different combustible gas by measuring the conductivity between the sensor material SnO_2 and the target [48]. The laser module uses an intense red laser beam as an output. Additionally, a webcam is attached at the top of the robot's gripper. The previ-

1 See: http://kobuki.yujinrobot.com/documentation/
2 See: https://msdn.microsoft.com/en-us/library/jj131033.aspx
3 See: https://www.adafruit.com/products/548
4 See: https://learn.adafruit.com/adafruit-motor-shield
5 See: https://www.arduino.cc/

ously described components are illustrated by a component diagram as shown in the following figure:

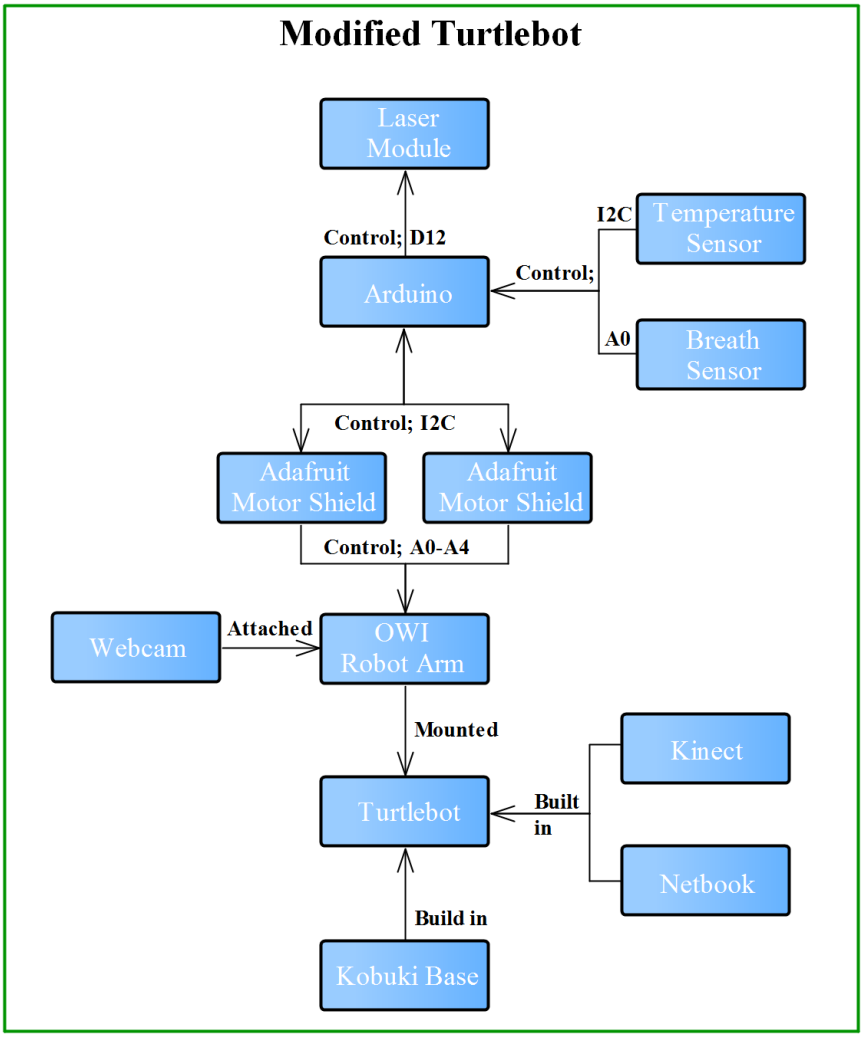


Figure 4.1: Component diagram of the modified Turtlebot.

The image above also illustrates the communication layout between the components. Starting at the top, the Arduino is responsible for all electronics. The laser module is turned on via the digital pin D12. The temperature sensor uses the I^2C connection. The breath sensor is accessed by the analogue pin A0 via the Multiplexer (MUX). Additionally, the Arduino board communicates with two Adafruit Motor Shields by using I^2C. The potentiometers which are used to control the motors are accessed via the analogue pins $A0 - A4$ of the Arduino board.

To gain a better understanding of the real hardware, the following figure shows the modified Turtlebot with all previously described components.

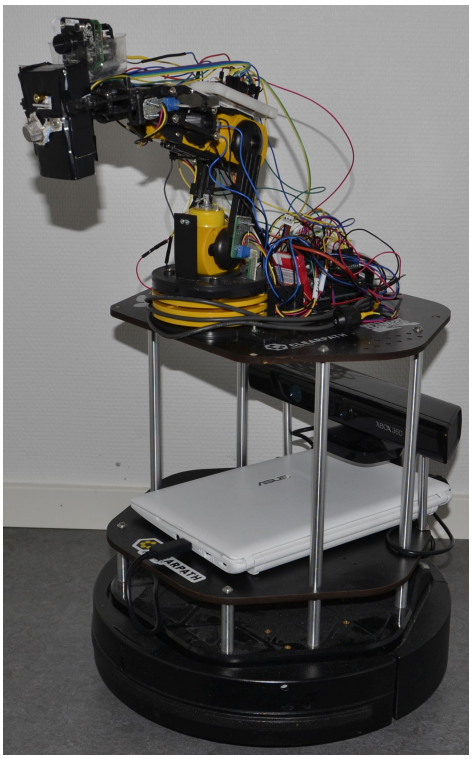


Figure 4.2: Modified Turtlebot 2.

ENVIRONMENT: The mobile robot is part of an intelligent environment that was built to resemble an actual apartment with a bedroom, living area, kitchen and workspace as shown in Figure 4.3. The map was built using the ROS package *gmapping* that created a Portable Gray Map (PGM) with the Simultaneous Localisation and Mapping (SLAM) method.

Figure 4.3: Intelligent environment as PGM.

ARDUINO CONTROL FIRMWARE: The software includes the *wire*, *Adafruit_MotorShield* and *SparkFunMLX90614* library. The first is for I^2C communication, Adafruit is used to control the motors and the last library provides access to the non-contact temperature sensor. The firmware sets up a serial communication with a baud rate of 115200bps and initialises all four motors, potentiometers, thermometer and pin modes for the laser module and a Multiplexer (MUX).

All components of the robot arm can be separately moved and measured. While the mobile robot analyses an object, the arm can be moved with several actions like horizontal, vertical or rectangular movement. The robot arm can be retracted to reduce harm, while the Turtlebot navigates through the environment. Additional procedures were implemented for a better accuracy of the robot arm, whereby all motors react to certain angles by changing their position with a given speed. The function checks all motor positions with a jitter threshold and changes all motor angles simultaneously, while checking the minimum and maximum range of each motor and also adjusts the speed in relation to the current distance.

Several commands were implemented to interact with the electronics for reading sensor values or to toggle the laser module. The MUX is used to add more digital/analogue pins to the Arduino board, that way, the breath sensor can be accessed and controlled.

All electronics can be controlled separately via byte commands. The most important ones are 'l' to enable the laser, 't' to read the temperature sensor, 'g' to read the breath sensor, and 'r' to read the current motor position as well as 'c' to move the robot arm in the safe position.

TURTLEBOT LAUNCH PROCESS: The pre-defined launch process of the Turtlebot starts by running the *roscore* executable, which starts several nodes such as ROS Master, Parameter Server, and rosout as previously described in Section 3.2. Additionally, the *turtlebot_bringup* launch file is loaded which runs basic nodes like battery_monitor, robot_state_publisher, diagnostic_aggregator, robot_pose_ekf, and boots the Turtlebot with all peripherals. The file loads the default configuration by setting up the Unified Robot Description Format (URDF) and calibration of the Kobuki base within a single ROS Master environment. For more detailed information, see the *rosmaster* package as well as the open source code Application Programming Interface (API)[6].

The robot_state_publisher node transmits the current robot state to the tf node. Thus, it is possible to keep track of all necessary coordinate frames over time. Furthermore, the robot_pose_ekf package provides the estimated 3D pose of the robot by measuring different sources such as the wheel odometry and the Inertial Measurement Unit (IMU) sensor. The package uses an extended Kalman filter to combine these sources. The resulting pose is passed to the odom_combined frame and published by the robot_pose_ekf topic [46].

Afterwards, the Adaptive Monte Carlo Localisation (AMCL) launch file is loaded for approximating the position of the Turtlebot in conjunction with a known environment by using a particle filter. The AMCL uses the result from the robot_pose_ekf topic and combines the information with the laser scans and map data. The file has been modified so that the correct map (see Figure 4.3) is loaded.

6 See: http://wiki.ros.org/rosmaster

4.2 DESIGN

First of all, this section illustrates the architectural overview, followed by the software component diagram of the mobile robot and workstation.

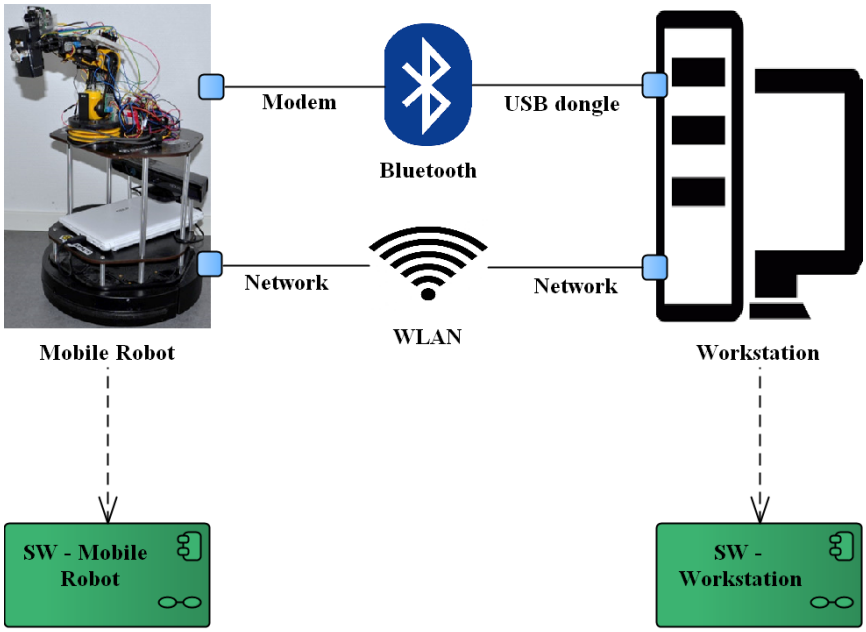


Figure 4.4: Architectural overview.

The prior overview illustrated in Figure 4.4 shows the intercommunication between the mobile robot and the workstation via WLAN by using the ROS framework. The workstation is used in conjunction with the netbook for tasks which require heavy computational power like image processing. The netbook is the central system of the current project and the "master" system in ROS. Additionally, the workstation receives continuously frames from the webcam by using a WLAN adapter. Bluetooth is used to establish a serial connection between the Arduino and the workstation. It is also to mention that the netbook and workstation utilise the same versions of the operating system, frameworks, and libraries. The system uses several software programmes for each device.

The bottom left side of the image reflects the component diagram of the mobile robot. The software package contains the *Arduino Control Firmware* previously described at the last paragraph of Section 4.1. The navigation component represents the ROS *Navigation Node*, which is written in C++ and works in conjunction with the pre-existing launch files for the Turtlebot as presented in Section 4.1. The navigation makes use of the ROS and Boost library and utilises the Particle Filter for localisation of the mobile robot. The *Anomaly Detection Node*, written in C++, is deployed on the netbook and communicates with the *Human Detection Node*. The node is responsible for finding anomalies in a known environment. The *Anomaly Detection Node* uses *Stop* and *Continue Patrolling* messages, that are sent via ROS topics to the *Navigation Node*. The software component uses the ROS library and algorithms such as Thresholding and PCA for the anomaly detection. Figure 4.5 shows each component of the mobile robot.

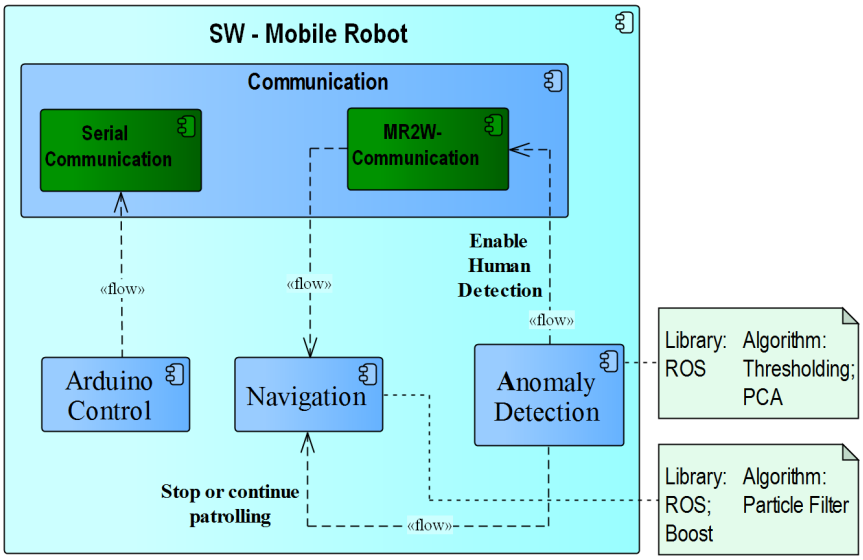


Figure 4.5: Software component diagram of the mobile robot.

The software component diagram illustrated on the bottom right side of Figure 4.4 consists of the *Human Detection Node*, the *Body Part Detection Programme* and the *Localisation and Visualisation Node*. The first component interacts with the Arduino board to measure the temperature and uses thresholding to recognise a human being.

The node uses the ROS library to interact with *Navigation Node*. The *Body Part Detection Programme* utilises the OpenCV and the Arduino library to analyse each captured frame via the webcam, to identify body parts, to control the robot arm, and to use the breath sensor. The programme communicates with the *Navigation Node*, if the robot has to change the position. The application uses the Viola-Jones for face, mouth, nose detection and the Histogram Back-Projection algorithm for hand detection.

The *Localisation and Visualisation Node* utilises the ROS library to visualise the fallen person in RVIZ. The OpenCV framework is used to estimate the position of the hands. The Intra-Communication component, at the bottom left side, represents the communication between the AHBL programmes within the components of the workstation.The following figure shows all these components in Figure 4.6.

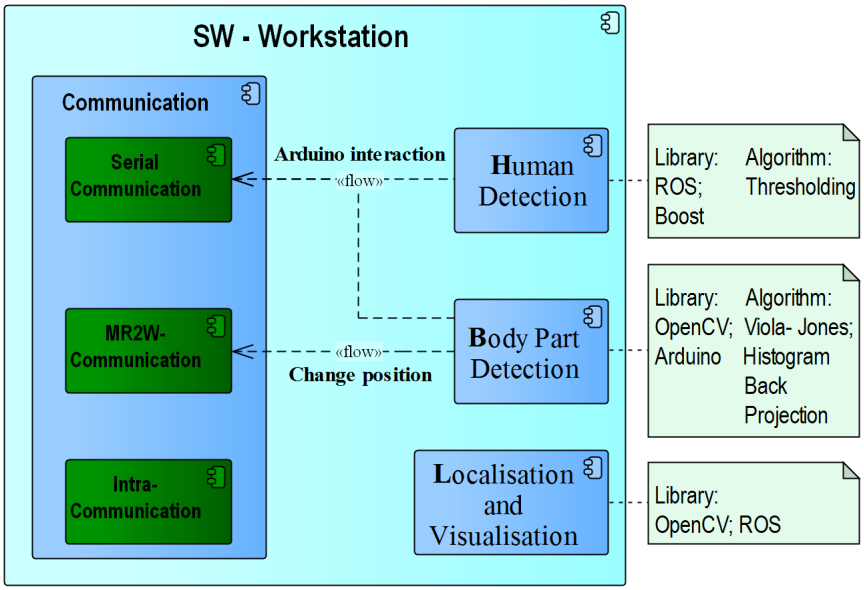


Figure 4.6: Software component diagram of the workstation.

4.3 NAVIGATION

After the launch process as described in Section 4.1, the Turtlebot is able to patrol within the room by using the *Navigation Node*. The node is based on the ROS actionlib interface for preemptable assignments such as pathfinding commands. The action client sends the goal with the *MoveBaseGoal* message to the actionlib server, that executes the task and sends a feedback depending whether the action was successful or not. The *MoveBaseGoal* message type contains the target pose that the robot should reach by checking the current position and orientation of the mobile robot [46]. Figure 4.7 shows the environment in RVIZ with the self-chosen patrolling path of the robot. This path can be set to any arbitrary path.

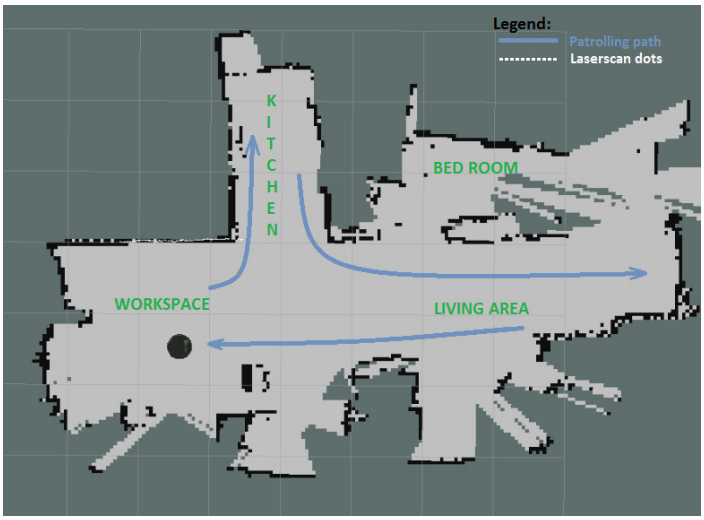


Figure 4.7: RVIZ view of the patrolling path and laser scan data.

After the initialisation phase of the Turtlebot, the programme starts and listens to the ROS topic */navigation_state*, which is used for an immediate navigation change. The actionlib server reacts on three different states. First of all, the commonly used *patrolling state*. If this state is active, the mobile robot patrols within the environment as described in Figure 4.7.

The *change_position state* is used to navigate to a specific position within the environment, e. g., to circle around the fallen person in order to change the view of the robot. The *analyse state* immediately cancels all ongoing goals and sends a stop request to the robot. Then the *MoveBaseClient* starts and waits 5 seconds till the action server is ready to use. The robot has to pass every waypoint within the environment, which is part of the *MoveBaseGoal* information in a configuration file. The data of all goals are stored in a vector. *ros::spin* starts a thread which executes every second the *Navigation State Callback function* to check the *navigation_state*.

While the *patrolling state* is active, the programme continuously iterates through the vector and sends each goal to the actionlib sever. The node waits until the *MoveBaseClient* sends a result of the outcome. If the goal is reached, the Turtlebot drives to the next position. If the result is negative, the robot has additional 10 seconds to reach the target; otherwise, the next goal is loaded. If the loop fails, the programme terminates. Figure 4.8 illustrates the previously described workflow of the *Navigation Node*.

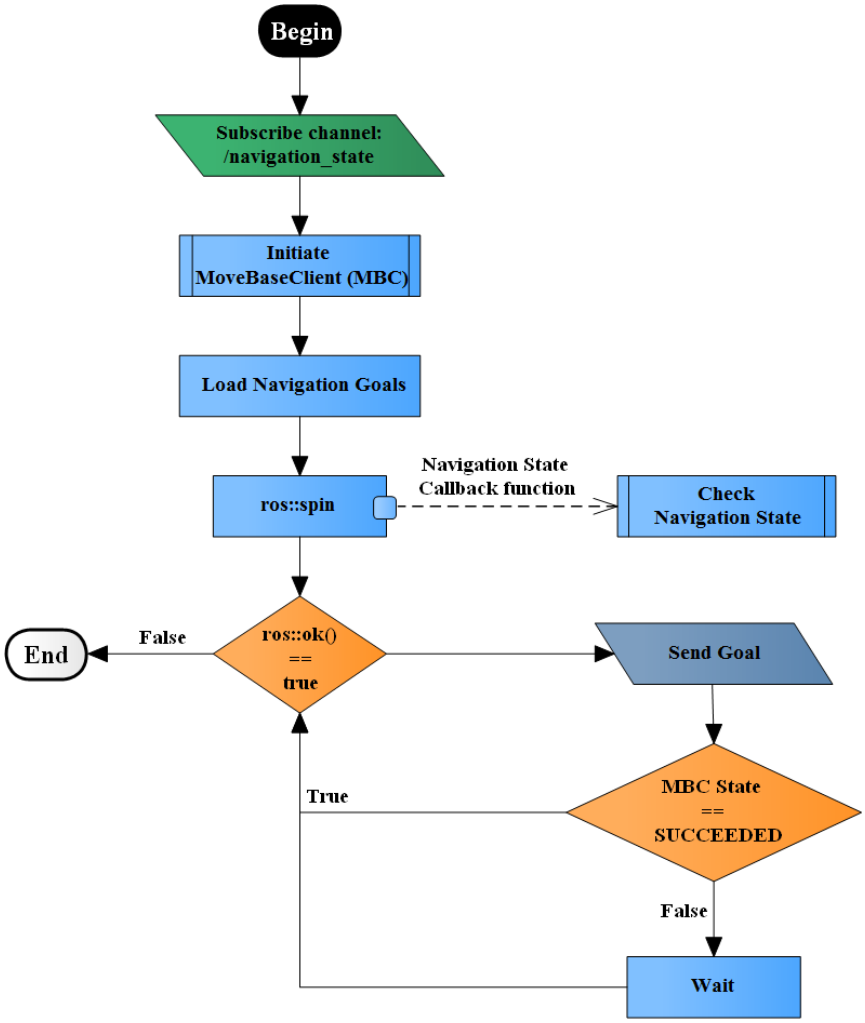


Figure 4.8: Flowchart of the *Navigation Node*.

4.4 ANOMALY DETECTION

The anomaly detection uses the laser scan data of the mobile robot to find human sized anomalies within the environment. Therefore, the *Anomaly Detection Node* utilises the following ROS message types: sensor_msgs to access the LaserScan data, nav_msgs to load the PGM as an OccupancyGrid, and the visualization_msgs to be able to set markers. Additionally, the tf package is used to access the transform_listener. Figure 4.9 illustrates the workflow of the *Anomaly Detection Node*.

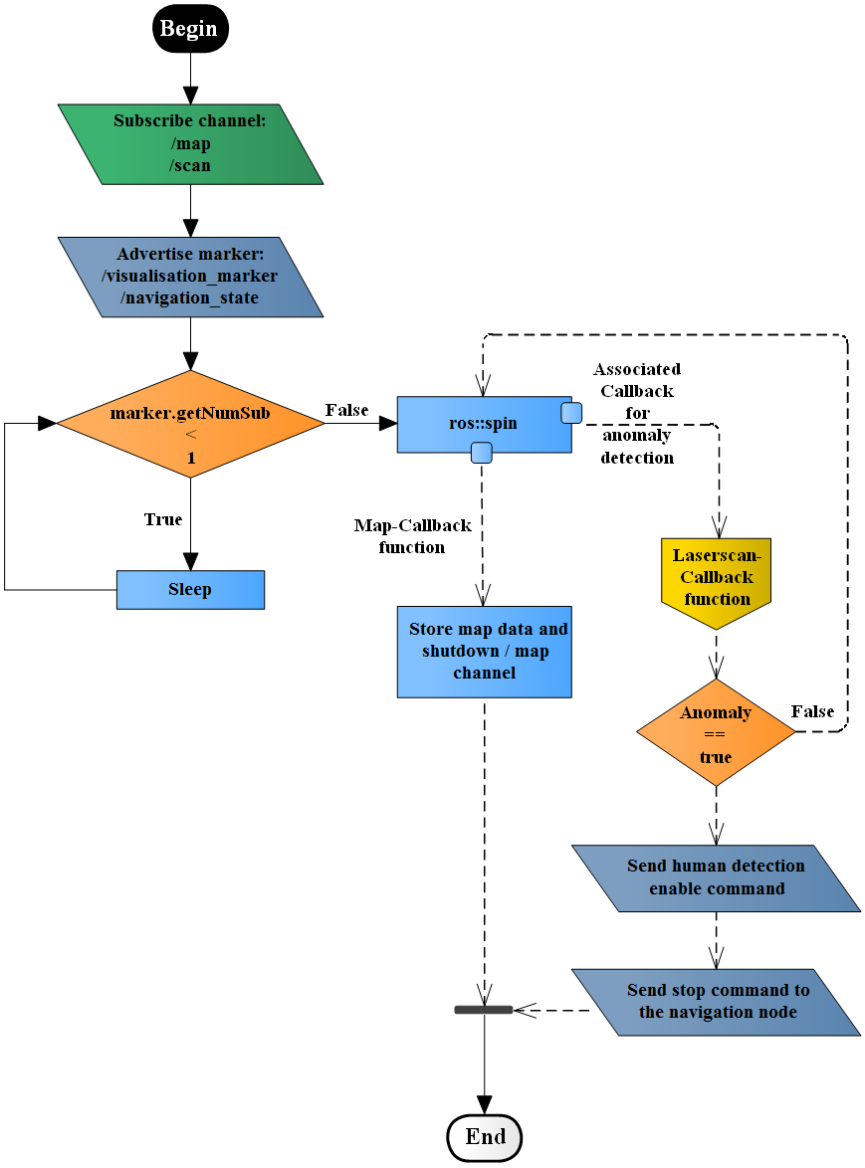


Figure 4.9: Flowchart of the *Anomaly Detection Node*.

The node starts to subscribe to the /map and /scan topic. In addition, the /visualisation_marker and /navigation_state topic are published. Then the transform_listener is initialised.

The programme waits until another node subscribes to the /visualisation_marker channel. This occurs by manually adding the topic to the RVIZ tool. For a detailed description of the laser scan function see Figure 4.10.

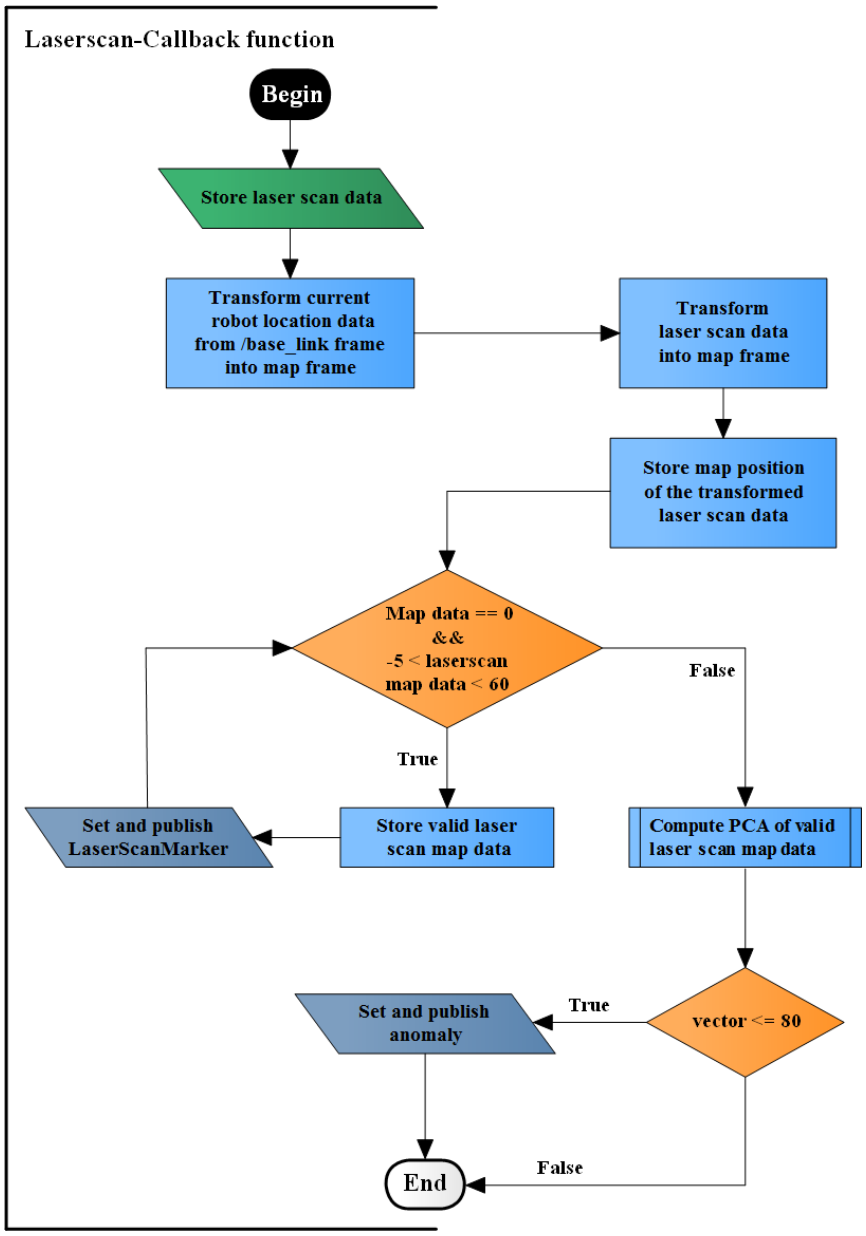


Figure 4.10: Flowchart of the *LaserScan-Callback function*.

Afterwards, a ros::spin starts the threads for two callback functions. The *LaserScan-Callback function* is terminated until the *Map-Callback function* is executed. This function receives the PGM and stores the OccupancyGrid data, which contains the map values as well as metadata information such as the map size and the resolution.

The return value of this function provides a boolean expression. If it is true, two messages are sent. The first is sent to start the *Human Detection Node* and the second to stop the robot by sending an *Analyse* message to the *Navigation Node*. If the analysed object is not an anomaly, the mobile robot continuous patrolling after sending a *Patrolling* message to the *Navigation Node*. The *LaserScan-Callback function* is successfully executed in a 10Hz interval, after the programme received the map data. First of all, the function receives the *LaserScan* message which consists of 640 laser scan dots. Each entry contains the detected range in [m]. The *LaserScan* message also contains metadata such as the angular increment in [rad] between all measurements as well as additional properties. The listener is used to transform the current pose of the robot into the /map frame. The information is evaluated by calculating the laser scan map position by using the laser scan distance and angle in conjunction with the current pose of the robot. The function also terminates if the scanned ranges are equal to the maximum range of the laser scan.

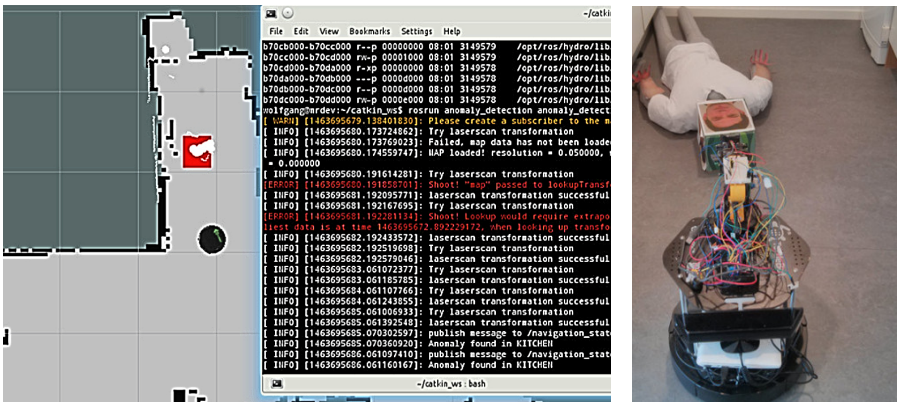
$$\begin{aligned} x_{ls} &= x_m + r[i]_{ls} \cos(\varphi_r + (i\varphi_{inc} + \varphi_{min})), \\ y_{ls} &= y_m + r[i]_{ls} \sin(\varphi_r + (i\varphi_{inc} + \varphi_{min})), \end{aligned} \tag{4.1}$$

where x_{ls}, y_{ls} represent the x and y position of the laser scan in the /robot frame, x_m, y_m are equal the x and y position of the map origin, $r[i]_{ls}$ is the distance for each entry, φ_r is the current yaw rotation of the robot, i is the counter variable for each entry, φ_{inc} the angular increment for each measurement and φ_{min} represents the start angle of the scan in [rad].

Afterwards, the points are converted into the map frame by using the map metadata. Then the laser scan can be compared with the static map. The first check compares each laser scan position within the static map. This comparison evaluates the cell value of the PGM, e. g., a value of 0 represents an empty cell, 100 means fully occupied and -1 refers to unknown. Thus, the laser scan position has to be compared with the current map data. Therefore, the laser dot is checked by using an 5x5 matrix. If an occupied cell is within the neighbourhood, the laser scan position is stored in a vector and published by the *visualisation_marker* topic.

Once all dots were checked, the PCA algorithm is used to check whether all valid laser scan dots represent an anomaly or not. The boolean expression is positive if the threshold is greater than or equal to 80 laser dots. This value approximately represents the width of a male chest (39cm). The final threshold was converted into the number of laser dots in relation to the maximum field of view of the Kinect device (3m) and experimentally adjusted. Additionally, the location of the anomaly is saved and displayed as a console output.

Figure 4.11a illustrates a positive detection of an anomaly in RVIZ. The red block represents the anomaly. The small white dots are the laser scan dots, published by the Turtlebot. The big white dots on top of the anomaly are the valid laser scan dots, that do not correlate with the known environment. The console output contains the location of the detected anomaly. The Figure 4.11b shows the represented pose of the mannequin.



(a) RVIZ view of the anomaly with console output

(b) Posture of the mannequin

Figure 4.11: Test case of a detected anomaly within the environment.

4.5 HUMAN DETECTION

The Human detection node is called when an anomaly is detected and checks whether it is a human being or not. The *Human Detection Node* receives an *Enable* message from the *Anomaly Detection Node*. An *Enable* message is referred as a pipe message that updates a log file which is checked by the *Human Detection Node*. Then the temperature sensor is used to measure the anomaly to distinguish between an object and a human being. Thresholds for objects and human beings were empirically obtained.

To classify if an anomaly is a human being or an object, the distance between the sensor and several test cases such as a pillow and a human being were calculated and compared. Additionally, both cases were measured with two different distances: a close distance (0.5m) and a far distance (1.5m). All four test cases were measured over a time period of one minute. Then the mean value of each case was calculated as shown below in Table 4.1.

TEST CASES	DISTANCE IN [m]	TEMPERATURE IN [°C]
Pillow close	0.5	23.25
Pillow far	1.5	21.88
Human close	0.5	24.66
Human far	1.5	22.49

Table 4.1: Threshold measured with an non-contact infrared thermometer; The first column describes the measured test cases, next the distance between the object an the measure followed by the determined temperature.

The measurements resulted in a noticeable difference between an object and a human being. Thus, the *Human Detection Node* utilises the *Enable* message from the *Anomaly Detection Node* by adding the range of the found anomaly. Therefore, the algorithm is able to compare the measurements with two thresholds. The average object temperature represents the lower limit to classify cool objects. The upper limit is used to exclude hot objects like an oven. The thresholds are defined between 23°C and 27°C.

The programme establishes a serial connection with the Arduino board by utilising the *Boost* library. Further, the /navigation_state and /visualisation_marker topic are published. After the serial connection is active, the temperature sensor is accessed by sending a 't' byte command in order to read the non-contact temperature sensor. This action is executed over a period of 10 seconds and each value is stored in a vector variable.

Afterwards, the median value is calculated and compared with the prior defined thresholds in conjunction with the current distance to the anomaly. If the comparison is true, the prior published anomaly marker is replaced by a human marker and an *Enable* message is sent to the *Body Part Detection Programme*. The mobile robot continuous patrolling if the analysed object is not a human by sending a *Patrolling* message to the *Navigation Node*.

Figure 4.12 illustrates the previously described workflow of the *Human Detection Node*.

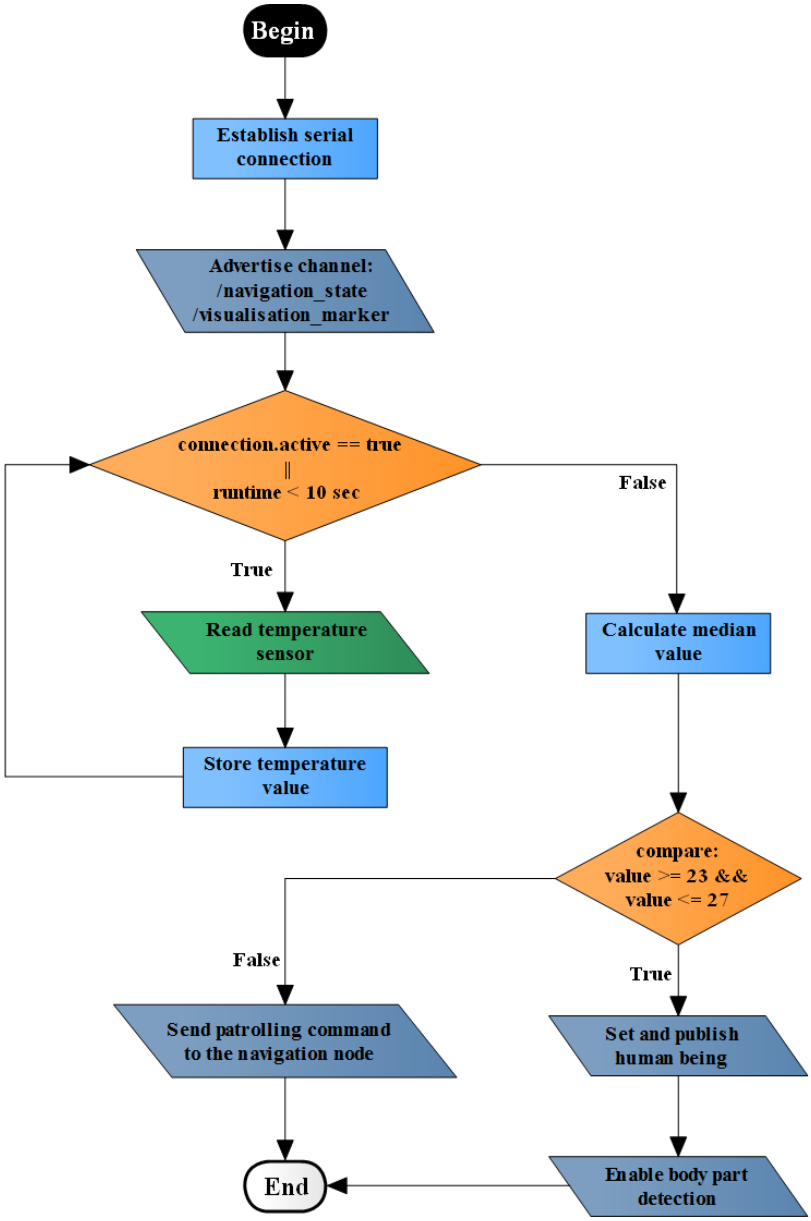


Figure 4.12: Flowchart of the *Human Detection Node*.

4.6 BODY PART DETECTION

The body part detection is responsible to find the face, nose and mouth of a fallen person which are used to calculate the other points of interest. To find these parts, the laser module, webcam and breath sensor are utilised in conjunction with the robot arm.

The workflow of the *Body Part Detection Programme* is illustrated in Figure 4.13.

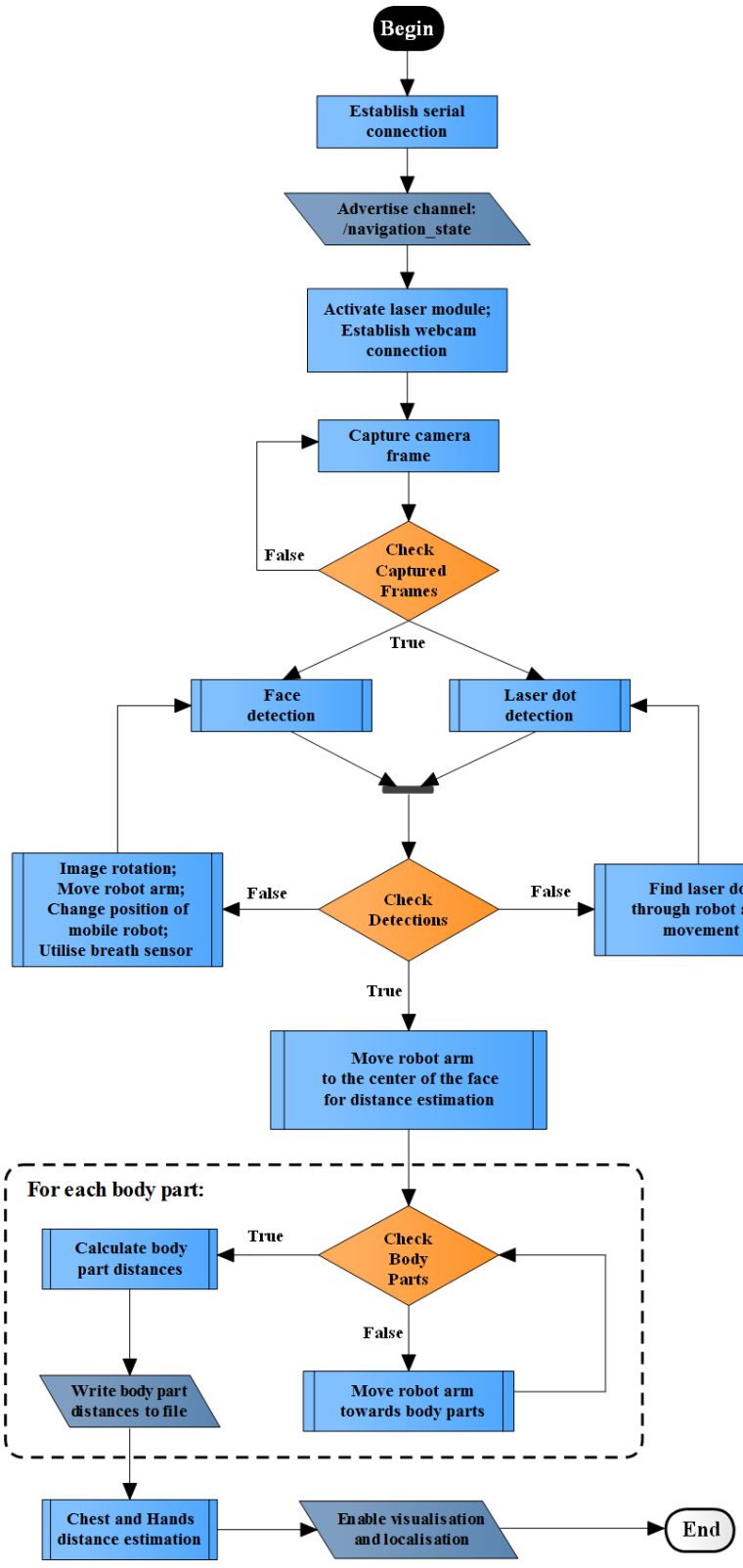


Figure 4.13: Flowchart of the *Body Part Detection Programme*.

The *Body Part Detection Programme* receives an *Enable* message from the *Human Detection Node* in order to activate the programme flow. First of all, the serial connection is established between the Arduino board and the body part software, followed by the advertisement of the /navigation_state topic. Then the laser module is switched on, followed by enabling the webcam.

Each captured frame of the camera is verified if it was received without an error. Afterwards, the frame is analysed by two defined processes. The face detection and laser dot detection are both executed within different threads. The programme flow is interrupted until each thread delivers the return value. The face detection uses Viola-Jones. If a face is detected, a rectangle is drawn in the frame to visualise the result at the workstation. Moreover, the average skin colour within the recognised face is stored in the body part logfile which is utilised by the localisation programme later on. Next, the centre of the face within the frame is returned.

The laser dot detection extracts all three HSV (hue, saturation, and lightness) channels from the frame. Then a binary threshold and bitwise operations are used to extract red colour from the HSV space. Furthermore, a 1x1 matrix is used in conjunction with Dilation to widen the range of the red colour in the image. Next, the frame is smoothed with a Gaussian operator, followed by the extraction of the circle by using the OpenCV method HoughCircle with the Hough transformation. Additionally, the location of the laser is drawn in the frame to track the dot movement at the workstation. The position of the dot is returned.

Then both results are checked. If the face detection fails, the frame is rotated in 15°steps. The application is able to use the robot arm. After a full rotation without a success, the arm is moved to the upper right limits of the motors to get a different view of the captured frames. Then the rotation process repeats. After another failure, the arm is moved to the lower left side. If the algorithm still is not able to find a face, the breath sensor is utilised. A threshold has been defined by assessing three cases: the environment noise, a plate full of alcohol, and a human breathing as shown in Table 4.2.

TEST CASE	ANALOGUE VALUE
Environment noise	239.8
Alcohol; 30cm away	571.7
Human Breathing; 10cm away	246.2

Table 4.2: Breath sensor measurements; The first column describes the cases, next to the calculated analogue median value.

Each case was measured for one minute and afterwards the median value was calculated. The table Table 4.2 illustrates that the breath sensor reacts strongly with the plate full of alcohol and yielded a small difference between a human breathing and the environment noise. After the arm movement was not a success, the breath sensor is read and compared with the defined threshold of an analogue value of 280. The last option leads the *Body Part Detection Programme* to send a *Change Position* message to the *Navigation Node*. Then the robot moves to the opposite site of the fallen person in relation to the current distance along with the angle between the robot and the human being.

If the laser dot cannot be recognised within the captured frame the programme utilises the robot arm by simple random movements.

After a successful face and laser detection, the robot arm is moved into the centre of the face by utilising the position of the laser dot. This procedure encapsulates a jitter process by saving the face and laser positions for each iteration. The method is used if the arm is not able to exactly point to the center of the face by allowing a margin of 10 pixels. Next, the distance between the face and the robot is computed by using the triangle similarity.

$$F = \frac{P \cdot D}{W}, \qquad D' = \frac{W \cdot F}{P}, \tag{4.2}$$

where F represent the focal length of the camera, P is the measured width of the body part in pixels, D is the distance from the camera in m, W is the known width of the body part in m and D' is the final calculated distance of the body part in m. The first equation is used to determine the focal length of the camera by using known information about the body part and distances. The second equation provides the distance calculation between the body part and the camera that is used for the distance calculation.

The distance is stored in the logfile. The same procedure is used for nose and mouth. Moreover, the location of the chest is estimated by calculating a vector between the centre of the nose and the middle of the bottom rectangle of the detected face. This vector is increased by 25cm. The distance and angle of the chest are stored in the logfile.

Afterwards, the robot is rotated towards the position of the chest. Next, the location of the hands are estimated by using the Histogram Back Projection. The algorithm uses the skin tone of the face to find the hands. If the method identifies more than 20% of the captured frame as skin colour, the location of the hands (within a 30cm radius of the chest (left and right side)) are stored in the log file. At the end, the programme sends an *Enable* message to the *Localisation and Visualisation Programme*.

Figure 4.14 illustrates the view of a captured frame after a successful body part detection.

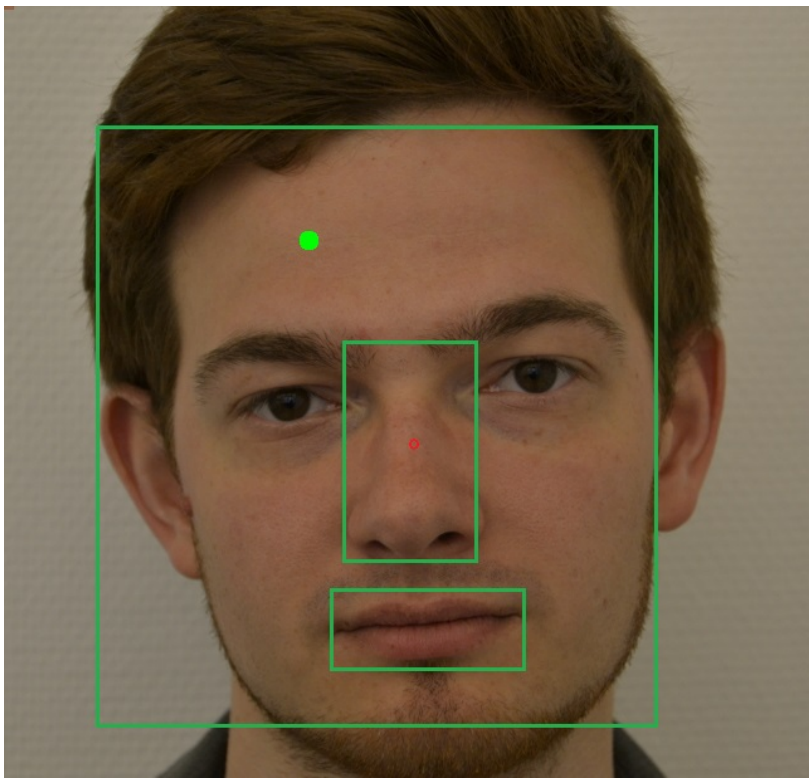


Figure 4.14: View of a captured frame after the body part detection with a recognised face, nose, mouth and laser dot.

4.7 LOCALISATION AND VISUALISATION

The *Localisation and Visualisation Node* publishes the body parts on the map by using the gathered data of the *Body Part Detection Programme*. The node is executed after an *Enable* message is received. The /visualisation_marker topic is advertised and the workflow is interrupted until the first subscriber. Then a static listener transforms the current pose of the robot from the /base_link frame into the /map frame to get the current pose within the map. Next, the body part logfile is loaded. The content comprises the distances of the face, mouth, nose, chest, and hands as well as the skin colour. Each part is published via the /visualisation_marker topic with the corresponding colour.

Figure 4.15 illustrates the prior described workflow of the *Localisation and Visualisation Node*.

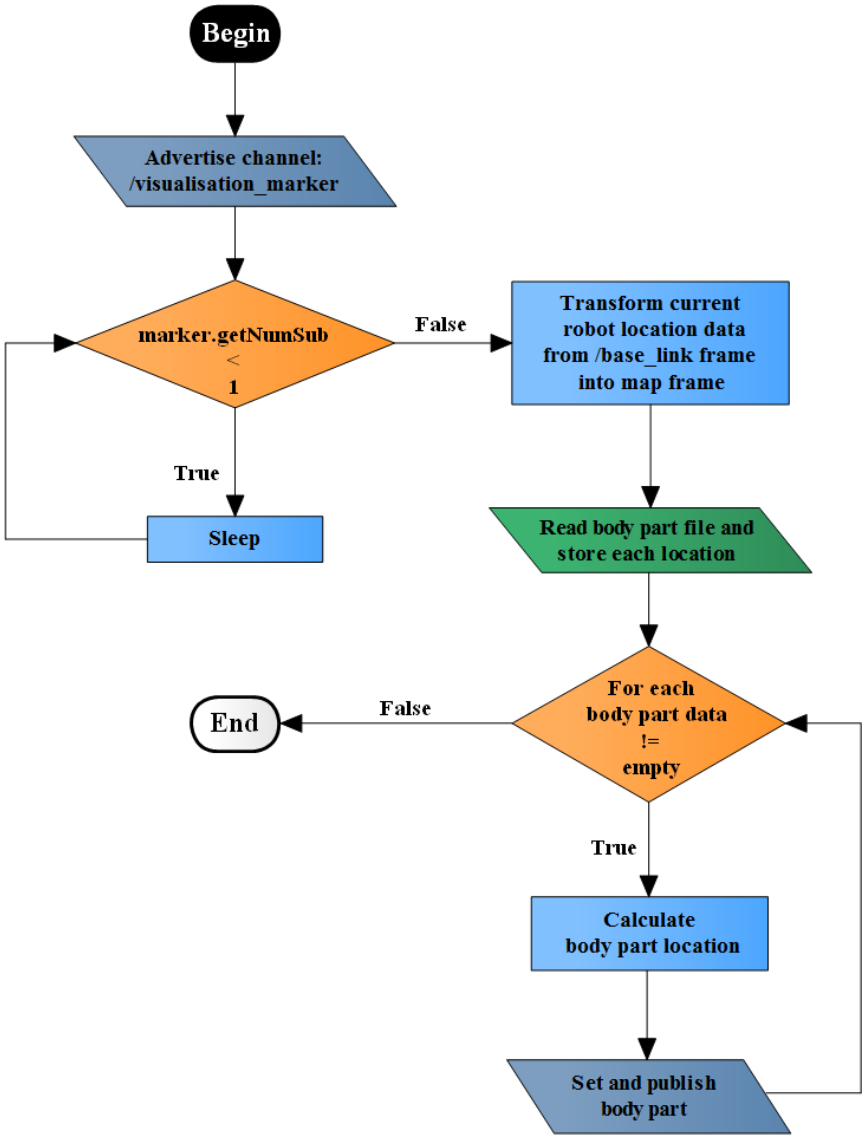


Figure 4.15: Flowchart of the *Localisation and Visualisation Programme*.

Figure 4.16 illustrates the RVIZ view after the visualisation is complete.

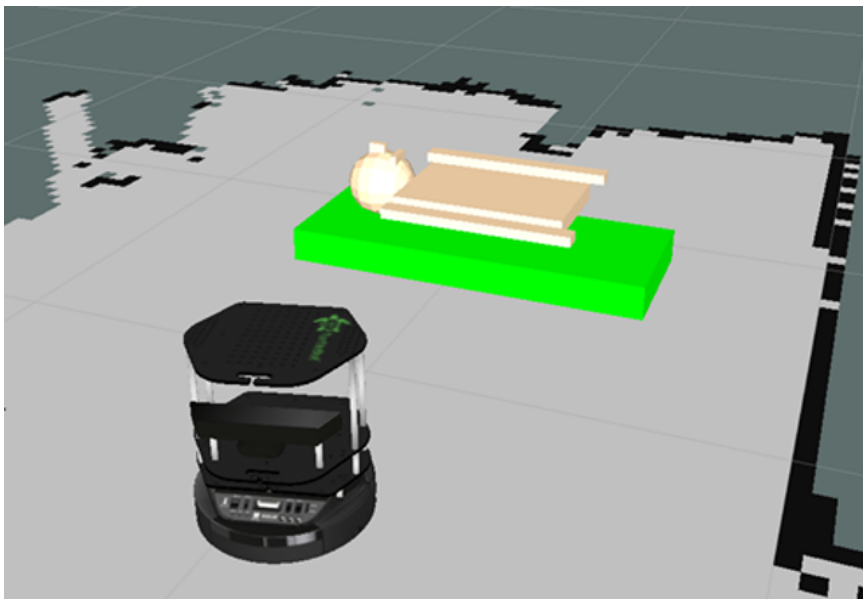
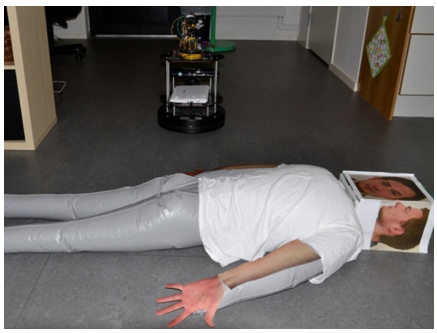


Figure 4.16: RVIZ view after localisation and visualisation process.

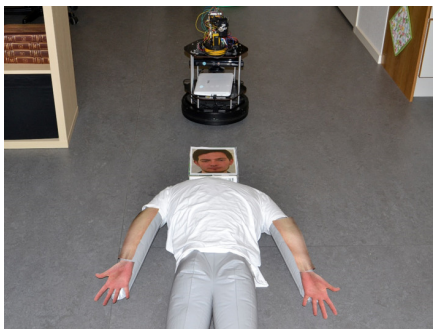
5 EVALUATION

The following sections describe the evaluation process of the proposed approach (AHBL). All four parts were evaluated: Anomaly detection, Human detection, Body part detection, and Localisation and visualisation. Each part was tested 20 times. Test cases for each part included some typical poses which a person might adopt, as well as some base cases tailored to each part which seemed like they could present difficulty.

In addition, some test cases used objects like a pillow or chair. It should also be briefly mentioned that the mannequin was placed in a random location in a well-lit environment. The following figure shows all poses which were used to evaluate each AHBL step illustrated by a mannequin.



(a) Mannequin in supine position; horizontally positioned towards the robot's sensors

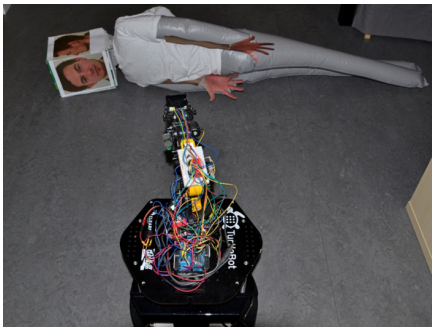


(b) Mannequin in supine position; vertically positioned towards the robot's sensors

(c) Mannequin in lean position

Figure 5.1: Standard mannequin poses for all AHBL evaluation steps.

Figure 5.1a shows a mannequin as a fallen person lying horizontally towards the robot's sensors. Next to the right illustrates the mannequin vertically aligned to the robot. The last Figure 5.1c displays the mannequin sitting against a wall. Further positions were needed to evaluate additional mechanics as shown in Figure 5.2.



(a) Side-lying mannequin; towards the robot

(b) Side-lying mannequin; face away from the robot

(c) Mannequin in prone position

Figure 5.2: Advanced poses for the evaluation process.

The upper left side of Figure 5.2 illustrates a side-lying mannequin facing the robot. Figure 5.2b shows the mannequin facing in the opposite direction of the robot. The lower image represents a person in the prone position.

5.1 ANOMALY DETECTION

The goal of the anomaly detection is that large (human-sized) anomalies on the floor which are unlikely to be noise can be found. Therefore, we evaluated the system by running the robot with the designated route, as described in Section 4.1 with different objects in between the patrolling path as illustrated in Figure 5.3.

Numerous problems could cause a system malfunction. One issue could be that the localisation of the robot in the 2D map is not accurate and therefore, the laser scan data is incorrect. Another issue could be that the environment was not mapped correctly or some furnishing components were moved without updating the 2D map.

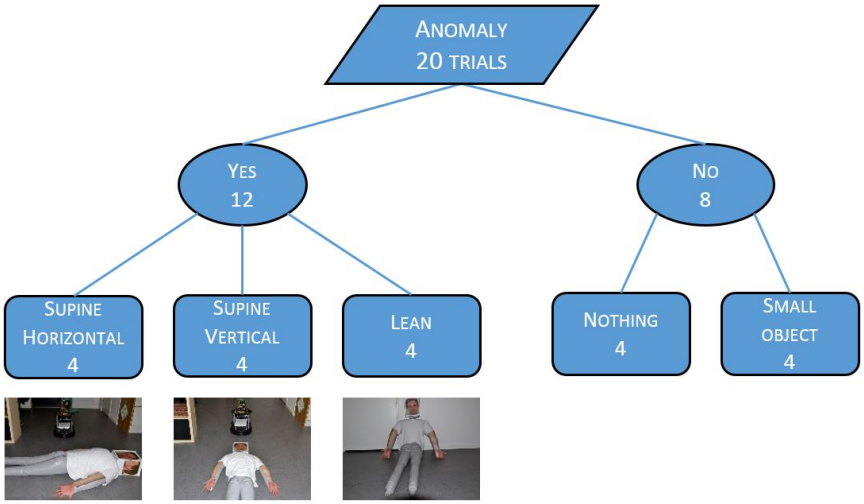


Figure 5.3: Anomaly detection evaluation overview; the number below each case refers to the test runs.

The evaluation of the anomaly detection distinguishes between two main cases. The algorithm should detect large (human-sized) anomalies which refer to the "Yes" path in the Tree diagram by placing a fallen person in the form of a mannequin in the environment. The mannequin was placed with three different poses as described in Figure 5.1.

Each pose was evaluated four times. The other two cases were evaluated by placing no obstacles in between the patrolling path. Additionally, a small object like a pillow was placed somewhere along the path that should not be detected as an anomaly. Each scenario was tested with four consequent iterations as illustrated in the "No" path. The algorithm was evaluated by the amount of successfully recognised anomalies as true positives ("Yes" path). False positives are represented in the "No" path. The detection rate was calculated by the correctness of all iterations using the equation below.

$$r = \frac{1}{N_{it}} \sum_{i=1}^{N_{it}} d_i, \tag{5.1}$$

where r is detection rate, $N_{it} = 20$ total number of iterations and d_i is the detection outcome (0 or 1).

ANOMALY	POSITIVE	NEGATIVE
Supine Horizontal	3	1
Supine Vertical	4	0
Lean	4	0
Nothing	3	1
Small Object	3	1

Table 5.1: Evaluation of anomaly detection; The first column describes the cases, next to the positive detections of a anomaly followed by the negative detections.

While the algorithm was most of the time able to detect the anomaly, 3 out of 20 cases were falsely detected. Errors arose when, for example, the mannequin was laid very close to a wall or due to a inconsistency of the model. The calculated detection rate yielded 85%.

Figure 5.4 shows that even though no object was within the environment, the algorithm detected an anomaly because the wall between the living area and the bedroom was not mapped due to a mirror that reflected the laser scan.

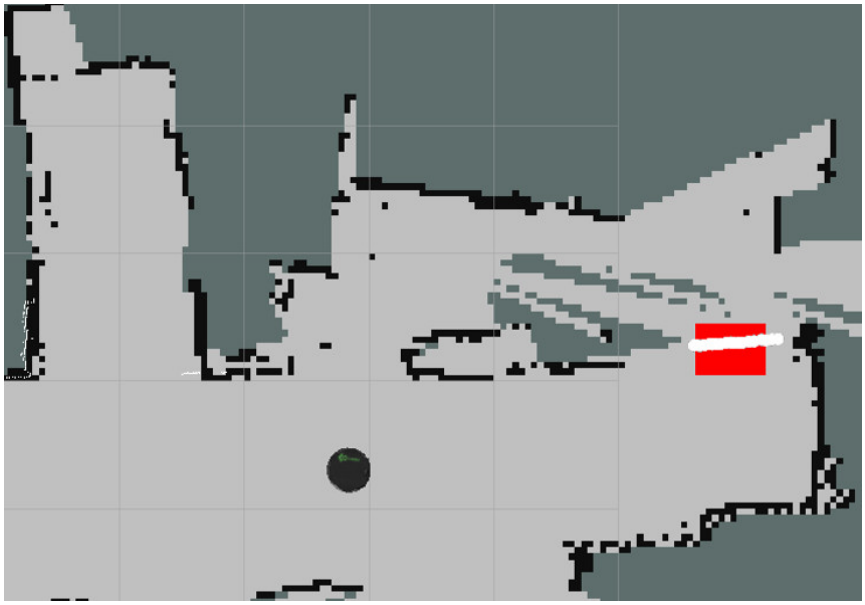


Figure 5.4: False detected anomaly.

5.2 HUMAN DETECTION

Various anomalies can be important for a home robot to detect, such as large objects or pools of water which could cause a human to trip. In the current work, the robot especially should detect fallen persons. The algorithm measures the temperature of the found anomaly and compares it to a model of temperatures at different distances for humans and objects.

A false human detection could be caused by a malfunction of the robot or the temperature sensor is not aligned with the anomaly. Another problem could be that a hot object would be classified as a human being.

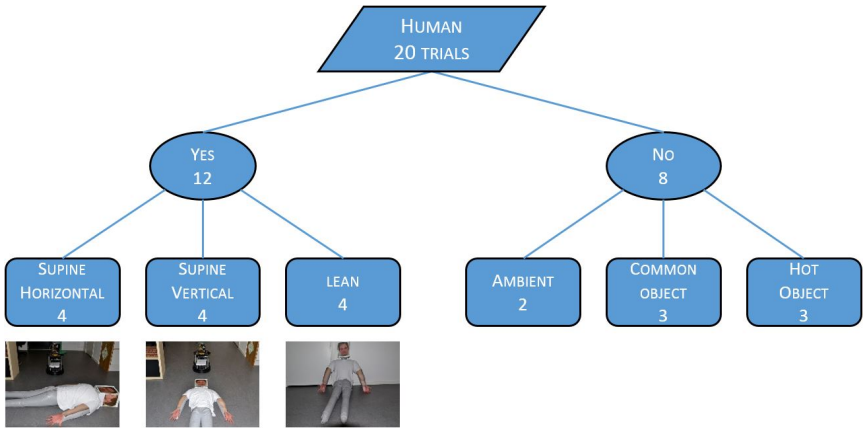


Figure 5.5: Human detection evaluation overview; the number below each case refers to the test runs.

The human detection evaluation started with the same base cases as described in Figure 5.1 and is referred as a true positive by the "Yes" path in Figure 5.5. Additionally, the room temperature, followed by common objects like a pillow as well as a hot object like a radiant heater were measured. Each test case was measured immediately after the anomaly was found to simulate a system demonstration test run. We decided to run each test case for 10 seconds. Afterwards, the mean value for each dataset was calculated and compared with the defined thresholds (see Section 4.5). The evaluation was calculated with the detection rate as the anomaly detection evaluation.

The results in Table 5.2 show that almost all iterations were able to be correctly rated, although one case was incorrectly classified as a human being because the temperature sensor did not point towards the measured object, that resulted in a detection rate of 95%.

OBJECT	POSITIVE	NEGATIVE
Supine Horizontal	4	0
Supine Vertical	4	0
Lean	4	0
Ambient	2	0
Common Object	3	0
Hot Object	2	1

Table 5.2: Evaluation of human detection; The first column describes the cases, next to the positive detections of a human being followed by the negative detections.

5.3 BODY PART DETECTION

The system detects human body parts such as nose, mouth, chin, chest and hands. Detecting these parts depends on detecting the face. Therefore, we evaluate the systems capabilities to detect the face with the pre-defined standard positions as shown in Figure 5.1. Figure 5.2 shows further cases, e. g., if the human being is in a semi-prone or prone position. The latter required the evaluation of the breath sensor to estimate the location of the head. Thirdly, the pixel accuracy of the recognised system was evaluated.

Various issues could affect body part detection, e. g., if the front face is significantly shifted since Viola-Jones is typically used for front faces. False positives could lead to incorrect detections of nose, mouth and chin, followed by wrong estimations of chest and hands. A hardware-based issue could be caused by the robot arm or webcam which could lead to a misconduct of the body part detection. A defect of one of those would simply mean a misbehaviour of the recognition system. The scenario as shown in Figure 5.2b evaluates the additional navigation pattern. Therefore the robot has to drive around the fallen person, assuming that there is enough free space. Another issue could be that the face is completely invisible as illustrated in Figure 5.2c and the breath sensor is not able to recognise any breathing.

(a) Face with occlusion like sun glasses (b) Image with low illumination

Figure 5.6: Face with different illumination and occlusion.

In addition, two different cases were tested. Occlusions or illumination could be a common problem as shown in Figure 5.6. Occlusions like sunglasses could interfere with the skin detection. Illumination could cause a false negative of the face detection.

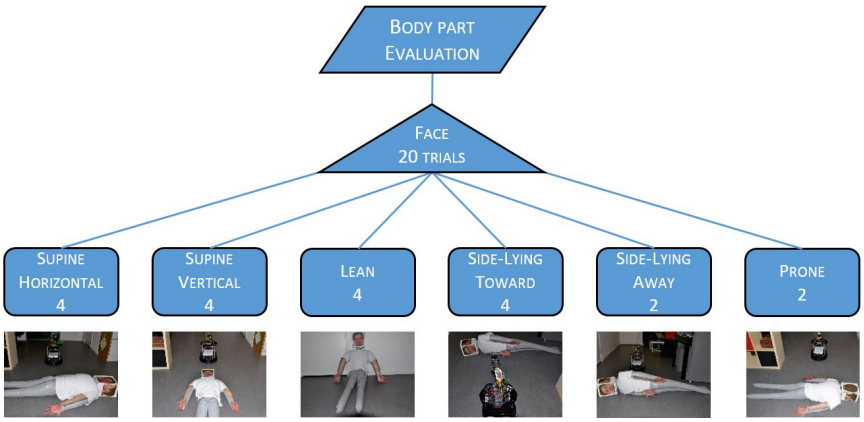


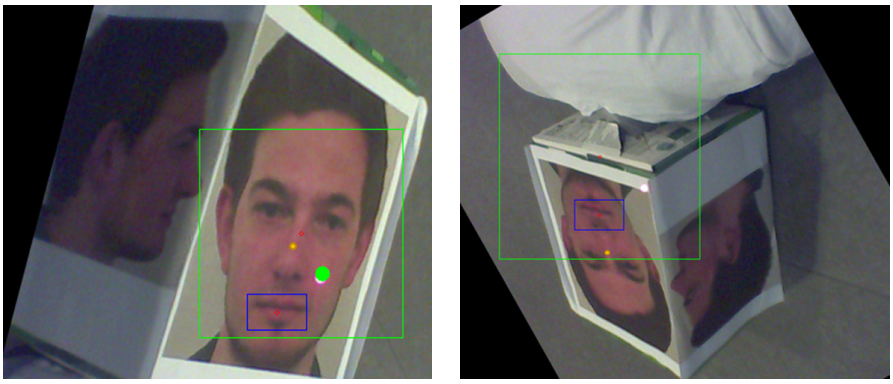
Figure 5.7: Body part detection evaluation overview; the number below each case refers to the test runs.

The evaluation of the body part detection was carried out with eight cases comprising six different positions as shown in Figure 5.7. Each case was evaluated four times, except the last two. The semi-lying facing against the robot and the prone position were tested twice, due to the additional navigation. This tree was used for the body part detection and distance evaluation.

MANNEQUIN POSE	FACE POSITIVE	FACE NEGATIVE
Supine Horizontal	3	1
Supine Vertical	3	1
Lean	3	1
Side-Lying Toward	4	0
Side-Lying Away	1	1
Prone	0	2

Table 5.3: Evaluation of body part detection; The first column describes the cases, next to the positive detections of a face followed by the negative detections.

Table 5.3 shows that the algorithm had problems to detect a face while the mannequin was in the horizontal and vertical supine position, due to the fact that more than one or no faces were found. The algorithm had also a problem with the additional navigation, which was used to drive to the opposite side, because of lack of space. The body part detection had additional issues with the breath sensor because the simulated breathing (bowl filled with alcohol) was not recognised. The system had no troubles with occlusions or illumination. The overall detection rate resulted in 70%.



(a) True positive of face detection (b) False positive of face detection

Figure 5.8: Examples of true and false positive face detections.

Figure 5.8 above shows the difference between a true positive tested face and to the right a false positive tested face. The green rectangle represents the detected face, the blue the detected mouth, the red circle corresponds to the center of the face and mouth, the green dot is the detected laser pointer and the orange dot is the real centre of the face.

We also stored each frame after the face was detected and defined ground truth by clicking within the image. We also defined success. If the euclidean distance between ground truth and the predicted centre of the face was within 50 pixels, we regarded the results as a true positive. If greater, then true negative. If there was no face, but our system detected something, it was a false positive. If there was a face, and our system detected nothing, it was a false negative. Based on these numbers, we computed a measure of system performance in terms of precision, recall and f-score.

$$\text{Precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive},} \tag{5.2}$$

where Precision is the fraction of positive detections that are relevant for the system.

$$\text{Recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative},} \tag{5.3}$$

where Recall is the fraction of correct detections that are relevant for the system.

$$\text{F} - \text{Score} = 2\frac{\text{Precision Recall}}{\text{Precision} + \text{Recall},} \tag{5.4}$$

where F-Score combines precision and recall and calculates the average to measure the test accuracy.

Table 5.4 shows the results of the evaluated body part detection. Most of the detected body part were in the range of 50 pixels. The calculated precision value resulted in 72.7%, the Recall resulted in 88.9% as well as the f-score with 0.8. Those values reveal that the system is able to detect most of the faces.

MANNEQUIN POSE	TP	TN	FP	FN
Supine Horizontal	2	1	0	1
Supine Vertical	2	1	0	1
Lean	2	1	0	1
Side-Lying Toward	3	1	0	0
Side-Lying Away	1	0	0	1
Prone	0	1	0	1

Table 5.4: Evaluation of body part detection II; The first column describes the cases, next to the true positives of a face in range followed by true negative, a false positive as well as as false negative.

5.4 LOCALISATION AND VISUALISATION

The evaluation of the localisation and visualisation was done by calculating the distance between the recognised laser dot at each body part and ground truth. Ground truth was pre-defined for each body part. In addition, the distance between the camera position and ground truth were measured. Afterwards, both distances were compared to each other.

A malfunction of the robot could cause a misbehaviour of the laser pointer. The human being could wear red clothes and therefore the image processing algorithm would not recognise the red dot. Another issue could be that the body part recognition system registers a false positive which would lead to a wrong distance estimation.

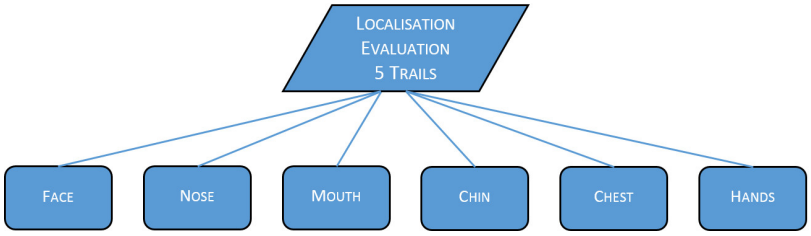


Figure 5.9: Localisation and visualisation evaluation overview; the number below each case refers to the test runs.

Figure 5.9 shows each evaluated body part for the localisation and visualisation. The mannequin was placed five times once with each pose except the prone position. Each scenario measured and estimated the distance for all body parts in cm. The mean value for all distances regarding each body part was calculated. These values were used for comparison between the estimated and measured distances as shown in Table 5.5.

BODY PARTS	MEASURED DIST. [m]	ESTIMATED DIST. [m]
Face	0.922	0.927
Nose	0.904	0.909
Mouth	0.885	0.891
Chin	0.849	0.853
Chest	0.164	0.2
Hand	0.283	0.32

Table 5.5: Evaluation of localisation accuracy; The first column describes the cases, next to the measured distance of each body part, followed by the estimated distance of each case.

The table above shows the average distances of the estimation and measurement between all five trials. As we can see, the average error distance calculation among face, nose, mouth and chin resulted in 0.5cm. The estimated distance of chest and hands yielded a greater error of 3.65cm which lead to an overall difference of 1.47cm.

6 CONCLUSION

The current project reported on the design of a FAMR which can localise a fallen person on a map using the proposed approach (AHBL), which consists of Anomaly detection, Human detection, Body part detection, and Localisation and visualisation.

For the first step, we introduced an algorithm to find human sized anomalies followed by a defined threshold to find human beings with a temperature sensor. The body part recognition system algorithm is able to find faces with different angles via the Viola-Jones method as well as nose and mouth. In addition, a vector between nose and chin is calculated to estimate the position of chest and hands. The detection of the laser pointer which is used for the robot arm interaction was implemented. Furthermore, the algorithm calculates the distance between the robot and the detected person via the triangle similarity. This information is used by the last step localisation and visualisation to calculate the 2D position with the map and visualise the detected body parts in RVIZ.

The main contribution of this project was to localise and visualise important parts of a human body for first aid with a mobile robot using different kinds of sensors. Thus, with AHBL the mobile robot can detect fallen persons as well as points of interest and map the location of those within a 2D map.

The development of a first aid mobile robot in a domestic environment is a challenging task. The evaluation results of the prototype were promising. The evaluation yielded that the system is capable of detecting anomalies with a detection rate of 85%. The system is also able to distinguish between an object and a human being with a rate of 95%. The body part detection rate resulted in 70%. Thereby it was possible to deduce the chin as well as to estimate the chest and hands. The localisation of all body parts yielded an average difference of 1.47cm and was visualised in RVIZ. The system is a step in the right direction, but the mobile robot should not perform first aid tasks like CPR on the human being with such an accuracy.

Robustness was one of the key factors as well as one of the main issues in our development of a FAMR. For example, how to identify a fallen person or to detect/segment human body parts, because image processing can be highly affected by illumination. Therefore we had to introduce some limitations to our system.

6.1 LIMITATIONS

At the beginning of the project certain choices regarding the overall project were made, i.e. the problem definition was chosen broadly with a specific hardware design. This resulted in a couple of limitations. All AHBL steps are restricted based on the used electronics. The current project had only a limited amount of positions to access as described in Chapter 5. The anomaly detection was restricted, which is restricted due to the depth data of the Kinect device and therefore depends on the knowledge of the position of the robot within the environment. It is also not possible to detect a fallen person in the bathtub or a person having health problems in a chair. The human detection depends solely on the non-contact temperature sensor, therefore if the fallen person wears a lot of clothing the algorithm could yield a false positive and identify the anomaly as an object, different body temperature may be an issue as well. The system may get distracted by analysing a corpse, a cooling oven or a computer.

Even though several considerations factor into the body part detection algorithm, the inevitable factor is to detect the face as well as nose and mouth, because all other body parts such as chin, chest and hands are deduced by calculating each relative distance. The face detection algorithm uses the standard Viola-Jones detection for front faces and therefore may not be able to detect faces with very different angles (in regard to the camera). It also relies on a well-lit room. Additionally, the skin colour calculation uses the average colour of the face rectangle which could result in a false positive by an occlusion, e. g., black sunglasses. The robot arm is using simple to use kinematics that could be heavily improved to analyse body parts such as the nose for breathing or the localisation to help novice first aiders with an improved HRI. Moreover, the laser point detection has issues with certain surfaces and edges. When the laser beam splits and consequently reduces the detection rate and therefore the mobile robot is not capable of estimating the distance between the body parts and the mobile base for localisation and visualisation.

First aid needs real world information in a 3D environment, but our system uses only 2D data to visualise the human body in a virtual environment. The following future work section will tackle those limitations by introducing several possibilities.

6.2 FUTURE WORK

Some limitations can be reduced by upgrading the hardware like switching from a Kinect (infrared laser) to a laser system like the Hokuyo device that may yield to a broader and more accurate environment to improve localisation of the robot and consequently the anomaly detection. Instead of the temperature sensor a heat camera like a Forward Looking Infra Red (FLIR)[1] system could be used in conjunction with the anomaly detection to identify a human.

The body part detection algorithm could be improved by detecting several parts independently by using the depth of the laser scan data as rough segmentation of extremities, additional classifiers for side faces or even an own face database with different views of a face such as top and bottom angles could yield a higher face detection rate.

The laser point detection rate could be improved by an additional laser with a different colour to increase robustness (to detect a laser point even if the victim wears red). The depth camera of the Kinect device could also be used for the distance estimation. The robot arm could be improved by using forward and inverse kinematics which would also improve the accuracy of the system and could also be designed as a ROS component which would yield easier communication. Additionally, all body parts should be detected and the 3D information should be used for further assessment.

The mentioned improvements could be implemented as a second iteration of the FAMR. Even though performance wasn't a factor we focused on (the goal was to build a prototype), it could be improved by upgrading the software frameworks and libraries that have been used such as ROS, OpenCV and Boost.

Further research should tackle complex problems like how to handle multiple targets, e. g., a triage by assessing all victims and to determine which one of them to help first. The robot could signal the first aider by illuminating the most critically wounded victim. In the distant future, additional studies should work on the robots' capabilities to perform first aid like how to do CPR.

1 See: http://www.flir.com/home/news/details/?ID=75067

BIBLIOGRAPHY

[1] H. M. Gross, C. Schroeter, S. Mueller, M. Volkhardt, E. Einhorn, A. Bley, T. Langner, C. Martin, and M. Merten, "I'll keep an eye on you: Home robot companion for elderly people with cognitive impairment," in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pp. 2481–2488, Oct 2011.

[2] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, (Washington, DC, USA), pp. 886–893, IEEE Computer Society, 2005.

[3] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154.

[4] A. H. Travers, T. D. Rea, and B. J. Bobrow, "CPR overview 2010 American Heart Association guidelines for cardiopulmonary resuscitation and emergency cardiovascular care," pp. 676–684, November 2010.

[5] p. Jevon, B. Ewens, J. Pooni, J. Breakwell, and M. Dhanda, *Att övervaka patienter med livshotande sjukdom*. Studentlitteratur, 2014.

[6] G. Jones, R. Endacott, and R. Crouch, *Emergency Nursing Care: Principles and Practice*. Cambridge University Press, 2002.

[7] D. Rollinson, Y. Bilgen, B. Brown, F. Enner, S. Ford, C. Layton, J. Rembisz, M. Schwerin, A. Willig, P. Velagapudi, and H. Choset, "Design and architecture of a series elastic snake robot," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4630–4636, Sept 2014.

[8] C.-S. Chang, T.-H. Tan, Y.-F. Chen, Y.-F. Huang, M.-H. Lee, J.-C. Hsu, and H.-C. Chen, *"Development of a Ubiquitous Emergency Medical Service System Based on Zigbee and 3.5G Wireless Communication Technologies"*, pp. 201–208. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[9] G. Lazzaro, "First Response to Emergency Situation in a Smart Environment using a Mobile Robot," Master's thesis, Halmstad University, School of Information Science, Computer and Electrical Engineering, 2014.

[10] T. Zhang and Y. Zhao, "Recognition for Robot First Aid," Master's thesis, Halmstad University, School of Information Science, Computer and Electrical Engineering, 2015.

[11] M. Mayr, "A Mobile Robot System for Ambient Intelligence," in *Bachelor thesis, Halmstad University, School of Information Science, Computer and Electrical Engineering*.

[12] R. Hornung, H. Urbanek, J. Klodmann, C. Osendorfer, and P. van der Smagt, "Model-free robot anomaly detection," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3676–3683, Sept 2014.

[13] T. Suzuki, F. Bessho, T. Harada, and Y. Kuniyoshi, "Visual anomaly detection under temporal and spatial non-uniformity for news finding robot.," in *IROS*, pp. 1214–1220, IEEE, 2011.

[14] Y. Ma, "TROLL: a regenerating robot," Master's thesis, Halmstad University, School of Information Science, Computer and Electrical Engineering, 2015.

[15] D. Saldana, R. Assuncao, and M. F. M. Campos, "A distributed multi-robot approach for the detection and tracking of multiple dynamic anomalies," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1262–1267, May 2015.

[16] C. Szegedy, A. Toshev, and D. Erhan, "Deep Neural Networks for Object Detection," in *Advances in Neural Information Processing Systems 26* (C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds.), pp. 2553–2561, Curran Associates, Inc., 2013.

[17] Y. Kim and T. Moon, "Human Detection and Activity Classification Based on Micro-Doppler Signatures Using Deep Convolutional Neural Networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, pp. 8–12, Jan 2016.

[18] G. Shingu, K. Takizawa, and T. Ikegami, "Human body detection using UWB radar in an indoor environment," in *Communications and Information Technologies, 2008. ISCIT 2008. International Symposium on*, pp. 283–288, Oct 2008.

[19] J. S. B. and S. M., "Multi-posture Human Detection Based on Hybrid HOG-BO Feature," in *2015 Fifth International Conference on Advances in Computing and Communications (ICACC)*, pp. 37–40, Sept 2015.

[20] G. Zheng and Y. Chen, "A review on vision-based pedestrian detection," in *Global High Tech Congress on Electronics (GHTCE), 2012 IEEE*, pp. 49–54, Nov 2012.

[21] P. Rudol and P. Doherty, "Human Body Detection and Geolocalization for UAV Search and Rescue Missions Using Color and Thermal Imagery," in *Aerospace Conference, 2008 IEEE*, pp. 1–8, March 2008.

[22] M. Correa, G. Hermosilla, R. Verschae, and J. Ruiz-del Solar, "'"human detection and identification by robots using thermal and visual information in domestic environments"," *Journal of Intelligent & Robotic Systems*, vol. 66, no. 1, pp. 223–243, 2012.

[23] A. E. Maghraby, M. Abdalla, O. Enany, and M. Y. E. Nahas, "Article: Detect and Analyze Face Parts Information using Viola-Jones and Geometric Approaches," *International Journal of Computer Applications*, vol. 101, pp. 23–28, September 2014. Full text available.

[24] A. Hu, S. Worrall, A. Sadka, and A. Kondoz, "A fast and efficient chin detection method for 2d scalable face model design," in *Visual Information Engineering, 2003. VIE 2003. International Conference on*, pp. 121–124, July 2003.

[25] H. Durrant-Whyte, N. Roy, and P. Abbeel, *Lying Pose Recognition for Elderly Fall Detection*, pp. 345–353. MIT Press, 2012.

[26] R. Srivastava, S. Roy, and T. Sim, "Rotation invariant Facial Expression Recognition in image sequences," in *Multimedia and Expo (ICME), 2010 IEEE International Conference on*, pp. 179–184, July 2010.

[27] N. B. Ibrahim, M. M. Selim, and H. H. Zayed, "A dynamic skin detector based on face skin tone color," in *Informatics and Systems (INFOS), 2012 8th International Conference on*, pp. MM–1–MM–5, May 2012.

[28] Z. Zivkovic and B. Krose, "Part based people detection using 2d range data and images," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 214–219, Oct 2007.

[29] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1297–1304, June 2011.

[30] A. Toshev and C. Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1653–1660, June 2014.

[31] T. G. de Meij, I. B. Larbi, M. P. van der Schee, Y. E. Lentferink, T. Paff, J. S. Terhaar Sive Droste, C. J. Mulder, A. A. van Bodegraven, and N. K. de Boer, "Electronic nose can discriminate colorectal carcinoma and advanced adenomas by fecal volatile biomarker analysis: proof of principle study," *Int. J. Cancer*, vol. 134, pp. 1132–1138, Mar 2014.

[32] D. S. Avalu, "Human Breath Detection using a Microphone," Master's thesis, University of Groningen, 2013.

[33] R. B. Lagido, J. Lobo, S. Leite, C. Sousa, L. Ferreira, and J. Silva-Cardoso, "Using the smartphone camera to monitor heart rate and rhythm in heart failure patients," in *Biomedical and Health Informatics (BHI), 2014 IEEE-EMBS International Conference on*, pp. 556–559, June 2014.

[34] H. Liu, Z. Huo, and G. Yang, "Omnidirectional vision for mobile robot human body detection and localization," in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, pp. 2186–2191, Oct 2010.

[35] R. C. Luo, K. Y. Chen, and M. Hsiao, "Visual simultaneous localization and mapping using stereo vision with human body elimination for service robotics," in *2012 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 244–249, July 2012.

[36] R. M. Holladay, A. D. Dragan, and S. S. Srinivasa, "Legible robot pointing," in *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, pp. 217–223, Aug 2014.

[37] P. Liu, D. F. Glas, T. Kanda, H. Ishiguro, and N. Hagita, "A Model for Generating Socially-Appropriate Deictic Behaviors Towards People," *International Journal of Social Robotics*, vol. TBA, no. TBA, p. TBA, 2016.

[38] R. T. Chadalavada, H. Andreasson, R. Krug, and A. J. Lilienthal, "That's on my mind! robot to human intention communication through on-board projection on shared floor space," in *Mobile Robots (ECMR), 2015 European Conference on*, pp. 1–6, Sept 2015.

[39] D. Szafir, B. Mutlu, and T. Fong, "Communicating directionality in flying robots," in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, pp. 19–26, ACM, 2015.

[40] K. Hsiao, H. de Plinval-Salgues, and J. Miller, *"Particle Filters and Their Applications"*. Cognitive Robotics, 4 2005.

[41] I. Jolliffe, *Principal Component Analysis*. Springer Series in Statistics, Springer, 2002.

[42] M. J. Swain and D. H. Ballard, "Indexing via color histograms," in *Computer Vision, 1990. Proceedings, Third International Conference on*, pp. 390–393, Dec 1990.

[43] A. Martinez and E. Fernández, *Learning ROS for Robotics Programming*. Packt Publishing, 2013.

[44] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 2008.

[45] S. Monk, *Programming Arduino Next Steps: Going Further with Sketches*. McGraw-Hill Education, 2013.

[46] M. Quigley, B. Gerkey, and W. Smart, *Programming Robots with ROS: A Practical Introduction to the Robot Operating System*. O'Reilly Media, 2015.

[47] Melexis Microelectronic Integrated Systems, *"Single and Dual Zone Infra Red Thermometer in TO-39"*, 5 2015. Rev. 009.

[48] Hanwei Sensors, *"Flammable Gas Sensor"*, 5 2014. Rev. 1.3.

PO Box 823, SE-301 18 Halmstad
Phone: +35 46 16 71 00
E-mail: registrator@hh.se
www.hh.se