DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING, SECOND CYCLE, 30 CREDITS

STOCKHOLM, SWEDEN 2017

Autonomous Identification of Human Activity Regions

LIN QI

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF COMPUTER SCIENCE AND COMMUNICATION

KTH Computer Science
and Communication

Autonomous Identification of Human Activity Regions

LIN QI

Master's Thesis at NADA
Supervisor: Patric Jensfelt
Examiner: Joakim Gustafsson

2017-07-29

Abstract

Human activity regions (HARs) are human-centric semantic partitions where observing and/or interacting with humans is likely in indoor environments. HARs are useful for achieving successful human-robot interaction, such as in safe navigation around a building or to know where to be able to assist humans in their activities.

In this thesis, a system is designed for generating HARs automatically based on data recorded by robots. This approach to generating HARs is to cluster the areas that are commonly associated with frequent human presence. In order to detect human positions, we employ state-of-the-art perception techniques. The environment that the robot patrols is assumed to be an indoor environment such as an office.

We show how we can generate HARs in correct regions by clustering human position data. The experimental evaluations show that we can do so in different indoor environments, with data acquired from different sensors and that the system can handle noise.

Referat

Mänskliga aktivitetsregioner, HARs (Human Activity Regions) är människocentrerade regioner som ger en semantisk partitionering av inomhusmiljöer. HARs är användbara för att uppnå väl fungerande människa-robot-interaktioner. I denna avhandling utformas ett system för att generera HARs automatiskt baserat på data från robotar. Detta görs genom att klustra observationer av människor för att på så vis få fram de områden som är associerade med frekvent mänsklig närvaro. Experiment visar att systemet kan hantera data som registrerats av olika sensorer i olika inomhusmiljöer och att det är robust. Framförallt genererar systemet en pålitlig partitionering av miljön.

Contents

Chapter 1

Introduction

As time goes on, robots will be expected to be able to carry out more and more tasks effectively. In the concept of a future factory, robots are envisioned to interact with humans, for example, to detect the positions of humans for safe navigation or to detect human activities and offer help. Thus, how to detect people and classify spatial regions is critical.

This thesis makes use of the concept of human activity region (HAR). A HAR is a human-centric semantic partition of space to facilitate observing and/or interacting with humans in an indoor environment. The basic role of a HAR is to partition space into regions. A HAR should be a region in which humans show frequent presence during a time interval. This characteristic makes HARs different from crossing regions, such as hallways. A robot can make use of HAR information to analyze human activities and/or interactions and modify its task routine accordingly.

One of the fundamental goals of future robots will be to assist humans, and generating HARs will allow robots to locate humans and analyze human activities which is helpful to achieve successful human-robot interaction. The proposed method of this project aims to enable robots to detect humans, track their positions and use the position data to generate HARs automatically. With these regions, the robot itself can "decide" what kind of movement to perform, such as determining a safe trajectory.

Note that, in this work, we are primarily concerned with determining HARs automatically and clustering the human positions to generate HARs by using human presence densities. Activity recognition within HARs is not a part of this thesis.

Based on previous work in this field, the primary contribution of our method is that it will use only a set of perceptual data of the environment to generate the HARs automatically. In our system, we use datasets contain data recorded during one to three months. Furthermore, we validate our approach using two datasets, one of them is collected over the period of three months, another is collected over one month. The datasets are further discussed in Chapter 5. The generated HARs can be used to plan optimal positions for activity recognition or for social mapping approaches [1].

1.1 Research Question and System Requirements

The research question of this thesis is *How can a set of human position data be used to generate human-centric partitions in an indoor environment?*

The thesis aims to design a system which can generate HARs automatically. The system will be evaluated on two different datasets in order to determine its generalization and precision on various data acquired in different environments. There are several requirements the system should satisfy, including the following:

- The system can generate clusters of human positions in the correct regions.
- The system can be used with data acquired from different indoor environments.
- The system can use the position data acquired by different sensors, such as RGB-D cameras and laser scanners.
- The system can handle data with system noise which is generated by errors of the system and random noise which is generated by some environment reasons.

1.2 Report Outline

The structure of this report is as follows. Related work about human detection and HAR generation is introduced in Chapter 2. Chapter 3 and 4 introduce the basic methods for designing the system. The experiment implementation is given in Chapter 5. Chapter 6 shows the results of the experiment. Finally, at the end of the report, there are some conclusions, a brief discussion and some suggestions for future work.

Chapter 2

Related Work

2.1 Human Detection

There are many methods of detecting and tracking humans with different kinds of sensors. Two of the most widely-used sensors are RGB-D cameras and laser scanners. RGB-D cameras are used to record color images and depth images, laser scanners are used to detect human legs. Both kinds of sensors are used for tracking humans and recording data for the position calculation.

For the laser sensor, many researchers have focused on the problem of tracking people based on detecting legs. They detected moving blobs that appear as a local minimum in the range image. In range imaging, we use the laser to produce a 2D pixel value showing the distance between points in a scene and a specific point. The result of the range imaging is a range image. A range image has pixel values that correspond to the distance [2][3]. When the laser scanner detects objects, the detected object shows a local minimum in the range image. For example, a laser scanner is set in the center of a room with four walls around. And one pillar is set in front of the scanner. The range image is shown as four lines (consist of many points) which represent walls and one half circle (consist of many points) which represents the pillar. The distances between points in this half circle and the position of the laser scanner show a local minimum compared with the distances between points in four lines and the laser scanner. As a result, this kind of local minimum in range image is used for representing detected objects. In this situation, scientists have defined two kinds of features for leg detection: motion features and geometric features.

Geometric features are features of objects constituted by a set of geometric elements such as points, lines, curves or surfaces. These features such as corner features, edge features, blobs, ridges and so on can be detected by feature detection methods.

A motion feature can typically be identified by subtracting two subsequent scans in range data. Arras et al. [1] give an example scan in range data from a typical office in their paper as Figure 2.1. The blue points are objects that detected by

the laser scanner. Topp et al. [4] and Schulz et al. [5] introduced two similar methods for extracting motion features. In these methods, robot's scans will firstly be aligned in order to ensure the differences of motion features are only caused by moving humans. Then the changing scans could represent humans. Topp et al. had good results in typical scenarios, but had problems in cluttered environments, because the lack of advanced pattern detection of people.

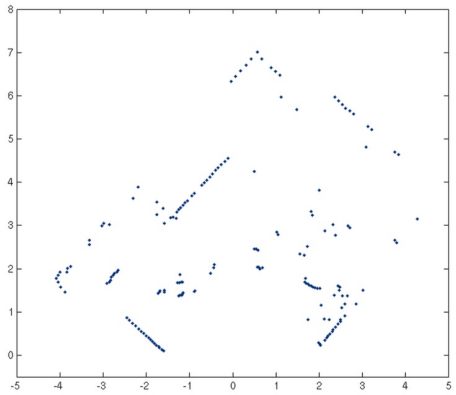

Figure 2.1: Example scan from a typical office.[1]

Hähnel et al. [6] considered the problems of identifying beams in range scans that are reflected by moving objects. A beam is a brunch of laser generated by the scanner and will reflect when reaches objects. They used the Expectation Maximization Algorithm (EM algorithm) [7] in their method for determining whether the beam was reflected by a moving human or not. Arras et al. [8] introduced an opposite method. They considered groups of beams and classified entire groups according to their properties. They mention 14 leg features in their paper, e.g. the number of beams, the circularity, the radius, mean curvature, and the mean speed, to only name a few. They used the AdaBoost algorithm [9] to classify human legs. For classification, we want to use features to represent objects and make features as targets for classifiers. Some features can only represent a part of the object, compared with features that can represent the whole object, we call these partial features weak features. AdaBoost is a kind of boosting function which uses weak features to train robust classifiers. A boosting function uses weak features to train weak classifiers, then uses some specific ways to combine weak classifiers to a strong classifier. The strong classifier is more robust than all the weak classifiers and can be used for classifying.

Using the RGB-D camera is another kind of effective method for detecting humans. Liu et al. [10] introduced a method for using a single RGB-D camera to detect and track humans. They transformed the original RGB-D data into a "point ensemble image". Each RGB-D data is recorded as a cell in a point ensemble image

(denoted by E), each cell of the image represents an ensemble of the points (a set of the points) which are projected into that cell, so it can be formulated as:

$$E_{i,j} = \{pi | p \in P \wedge (p_x, p_y) \in g_{i,j}\}$$

where P is the point cloud in the new coordinate system, $pi = (p_x, p_y, p_z)$ is a 3D point of p, and $g_{i,j}$ is a cell of the ensemble image. Then they used "physically plausible candidate localization", which means separating 3D human positions into 2D grids in the ensemble image, and "learning-based refinement" [9], which means using boosting function with weak features to refine the classifiers and methods for detecting humans. We give each weak features a weight and multiply weights with features to calculate the values of classifiers, then use voting method to refine the weak classifiers into a strong and robust classifier.

Dondrup et al. [11] also introduced a so-called upper detector in their project. They used a template and the depth information of an RGB-D sensor to identify upper bodies (shoulders and head), designed to work for close range human detection using head-mounted cameras. Dondrup et al. [11] used the data with the Bayesian tracker [12] to calculate human positions. A Bayesian tracker is also known as a Bayesian filter, it is a general probabilistic approach. A Bayesian tracker uses incoming measurements and a mathematical process model to estimate an unknown probability density function [12].

For the multisensor-based detection, Bellotto et al. [13] introduced a combination method for different sensors. They used a sequential Unscented Kalman Filter to fuse the two different sensor data in order to implement a robust human tracking method. The Unscented Kalman Filter (UKF) is an algorithm that uses a series of measurements observed over time and produces estimates of unknown variables [14]. An UKF is more precise than other filters based on a single measurement. In their paper, they firstly tracked humans with a laser used to detect legs. If they found the leg features, they would update the state by the UKF, then ask for a face detection result. If not, they would try to locate the face detection without updating the state, then use the same updating rule for face detection. They decided that robust results would come from the fusion of the results from both kinds of sensor.

Dai et al. [15] introduced an object detection method with "Region-based Fully Convolutional Networks"(R-FCN). The details of R-FCN can be found in the paper [15], Figure 2.2 is a brief overview of the architecture copied form the paper. The convolutional network is especially well developed for usage on images. They extracted the image data by local features in hidden levels and convoluted them into feature maps, a feature map is a function which maps a data vector to feature space. Then convoluted the feature maps into Region Proposal Network (RPN). Given the Regions of Interest (RoIs) in RPN, the R-FCN architecture is designed to classify the RoIs into object categories and backgrounds. The result of the whole network is generating a "bounding box" to localize the target feature's position in the image. A bounding box is a rectangle added by the network and used for representing detected features. This method can be used for human detection by changing the

target features to human features, and it can generate a bounding box to represent the position of the detected human.

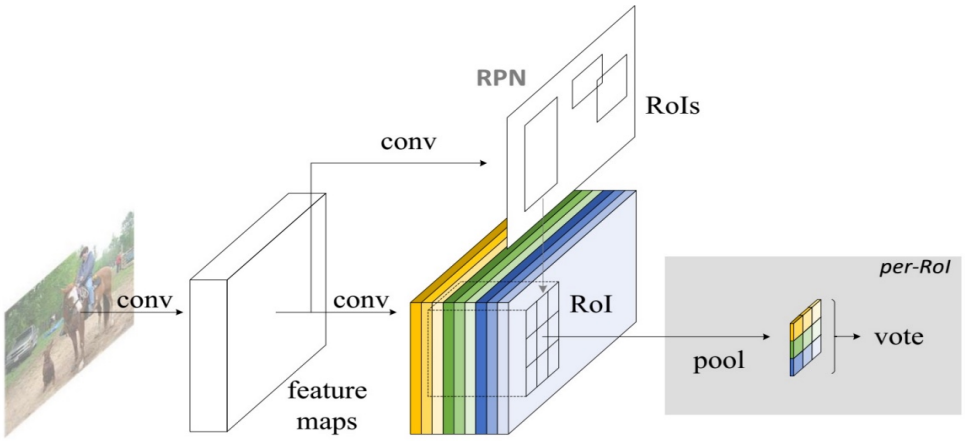

Figure 2.2: Key idea of R-FCN for object detection [15].

2.2 Data Handling and HAR Generation

HAR generation, in general, is a process of spatial partitioning of indoor environments. There are many studies about spatial partitioning. Karaoguz et al. [16] partitioned the space by detecting objects connected to activities. For example, they detected chairs and desks to represent a meeting room in their experiment. These generated regions were compared against human position data to see that many observations are gathered in these regions. This is the starting point for this thesis, where we instead focus on how to best process the human position data and extract regions for this data. In [16], spatial partitioning studies was separated into two categories based on the scale of the partition scope. The first group is local methods, which mostly analyzed individually based on single scenes, such as [17][18][19]. In these papers, they used the local features acquired by sensors such as an RGB-D camera with geometric relationship between local features to segment and semantically label scenes with predetermined labels, then used these labels to separate small scenes in order to partition the whole environment. Kunze et al. [20] combined 3D object recognition with qualitative spatial relations between objects to segment and semantically label a table-top scene.

The second group aims to partition a complete floor plan rather than single scenes based on global approaches [21][22][23]. For the global approaches, various methods exist, such as feature based methods [15] and segmentation based methods [17]. For example, based on semantically labeling in an indoor environment, the author in [23] offered a maximum likelihood semantic map of the global spatial which had topological and semantic representations.

Considering the randomness of human movements and system errors, the results

of a tracking system will contain noise. The noise is defined as incorrect positions in our system and the resources of the noise will be described later in Section 3.1. A natural method to remove most of the noise is density estimation [24]. Density estimation is a non-parametric method to estimate the probability density of a random variable. This method is used to fit data into different models, and uses a threshold function to remove irrelevant data.

With the position data collected by the sensors, HARs should be the areas in which humans are most frequently present or with a high density of human position data in a particular time period. One of efficient methods for generating HARs based on the position data is clustering. One of efficient clustering algorithms is K-means. Huang [25] presented two variations of the K-means algorithm (k-modes and k-prototypes) for clustering categorical data. Ferreira et al. [26] used an extended K-means for daily clustering patterns of human activities. In their paper, they gave the daily patterns nine labels (i.e., work, shopping etc) and separated the timestamps into a 2592-dimensional binary vector. Then they used K-means combined with the Principal Component Analysis (PCA) algorithm [27] to partition individuals in a metropolitan area into clusters based on their daily activity similarity. Here, PCA was used for reducing the high dimension to remove similarities. It uses linear transforms to project the original features in a low dimensional space which can save time and consumption compared with doing the feature extraction in a high dimensional space.

Considering the optimal cluster number problem of the K-means algorithm, which means the K-means algorithm needs to initialize the number of clusters by humans, we can use another clustering algorithm called Self-organizing Feature Map (SOM). In the SOM algorithm, it is not necessary to initialize the number of clusters. Everingham et al. [28] provided a "novel SOM network" that dynamically recognized separate clusters of data in the input levels. By using an iterative approach, the category data can be separated into several clusters. However, SOM not only updates the weight of the current center point, but also updates the weight of the neighbor of the current center point, so it is significantly influenced by the noise. For example, if a noise is far away from all the other points, the distance between the noise and other points is bigger, the hidden level will generate a larger weight for the noise. This larger weight will influence the feedback values of the neighbor and result in a incorrect classification.

The density-based clustering algorithm, Density-Based Spatial Clustering of Applications with Noise (DBScan), introduced by Ester et al. [29] is different from the classifying cluster algorithms such as K-means, it does not need to initialize the number of clusters. Ester et al. introduced the concept of "Density Neighbor" which means the area around a center point with a given radius. The basic concept of this algorithm is each center point P, and all points O in P's density neighbor comprise a complete cluster. The algorithm has the advantage of being able to cluster data into different shapes, and generating clusters automatically and correctly even when influenced by noise. However, it is sensitive to the parameters that define the radius and the minimum number of the cluster.

Chapter 3

Data Handling Methods

The whole system consists of two small systems, the data handling system and the human tracking system. The data handling system is the main part of this project. The method for data handling mainly concentrates on the position of humans and is aimed at generating HARs based on the density of the position data. The input data for the system can be a set of human positions generated by the human tracking system or some independent sets of human positions. The basic components of the data handling system can be divided into three parts: data pre-processing, density estimation based filtering, and a clustering algorithm. The flow chart of the system can be seen in Figure 3.1.

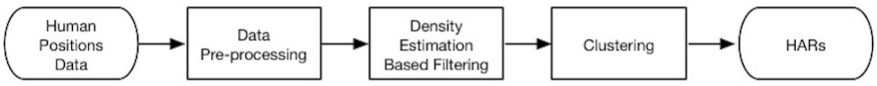

Figure 3.1: Flow chart for the data handling system.

In this system, the pre-processing part is used for removing system noise from the dataset. The density estimation based filtering part is used for calculating the posterior probability of each position as a part of an HAR. In the clustering part, the system uses a density based algorithm to generate region partitions. Each part of the system is described in detail in Sections 3.1, 3.2, and 3.3. For each part we also describe what to consider when evaluating that part. Details about the evaluation is provided in later chapters.

3.1 Data Pre-processing

The data pre-processing part is used for removing system errors from the dataset. System errors can arise for the following reasons:

- Incorrect localization. This kind of error is generated by the incorrect localization of the robot. The human position calculation part is based on the distance between the image center and the robot position. If the robot position is incorrect, then the result of the human position will be biased.
- Incorrect distance calculation. This kind of error could be generated by the wrong depth value. The distance calculation is based on the depth image, but sometimes the depth image values are not a number (NaN). In this situation, the distance results will become system errors.
- The human position is out of range. This kind of error is generated in some specific situations. For example, a human passes the robot lab outside the window, and the robot captures him as an image and uses this to generate a position. This position is outside of the map and constitutes system errors influencing the final result. Thus, we use data pre-processing to remove this kind of error.

When generating HARs, the noise positions (for example, some positions are inside the wall or outside the map) will influence the accuracy of the result. In order to remove noise, the system can set a threshold on the distance between the human and the robot. The threshold function could be the furthest distance based on the real environment. For example, in our system, we assume that the robot only patrols in small environments such as offices and corridors. Kittler et al. [30] mentioned that in a small environment, the threshold can be set to be the width of the smallest room empirically. If we choose the width of a larger room as the threshold, when the robot patrols in a small room and detects a human outside the window. The distance is smaller than the threshold, the robot will record this human and generate a position. This position, in this thesis, should be a system error. As a result, we choose the width of the smallest room as the threshold. If the distance between the robot and the detected human is bigger than the threshold, the resulting position will be considered as noise. Moreover, the system can use the map as a limitation; if the resulting position is outside of the map, it will be removed as noise. The system will use both a threshold function and the map as limitations to remove system noise.

The pre-processing can remove most of the system noise inside the input, but there might be some other kinds of system noise, such as a human passing the robot in a hallway, resulting in the robot generating a position. This kind of data is random noise because it appears randomly with respect to the positions of the HARs. It only appears a few times and is only maintained for a short time, so it is not recognized as a part of the HAR based on the definition. The function for handling this kind of random noise is introduced in Section 3.2.

3.2 Density Estimation Based Filtering

3.2.1 Method Description and Motivation

As mentioned in the previous Section 3.1, the data set might contain position data which shows up randomly, such as when a human walks in the hallway and is captured by the robot's sensors. In our project, we assume the HARs should be regions that contain a large amount of human positions. In these positions data should appear concentrated and plentiful. The density of the human positions in a HAR should be large, therefore the small density parts can be removed as noise. Notice that in some situations, such as a human always working at a desk in an office, the detection results of his appearance show up in one position many times or in a small area frequently. This kind of area or set of points still has a large density and should be recognized as a HAR.

The motivation for choosing density estimation [24] is mainly based on the fact that it is a natural method for handling this kind of problem. Our system recognizes HARs based on the density of the region, density estimation is a method that is used for estimating the density of a point or a region contains sets of points, thus it is the first choice of our system. Density estimation is a non-parametric method which is used to estimate the density function for unknown distribution, and can be used for calculating the posterior probability for each event. The method is chosen for calculating the posterior probability for each position into the input and uses a threshold function to classify them as part of a HAR or not.

3.2.2 Algorithm and Metrics

For independent points $x_1, x_2, ..x_n$ which have the same distribution, we assume their density function to be f, then their density estimation function is shown as:

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\frac{x - x_i}{h}$$

where K is a Kernel that is positive with integration equal to 1, h is a smooth function also known as a bandwidth, and $K_h(x)$ is a scaled Kernel. The smooth function here is used for removing the noise and distortion.

The system uses the Gaussian function as a Kernel in the density estimation because of its mathematical convenience such as we can get a Gaussian distribution after summing up two Gaussian distributions. For each position in the input, it calculates the 2D Gaussian value between it and all the other positions in the input. The 2D Gaussian value for two points, P1 and P2, is calculated as:

$$Gaussian(P1, P2) = e^{-\frac{(x_1 - x_2)^2 + (y_1 - y_2)^2}{2 * \sigma^2}}$$

where x and y are coordinates of the position, σ is the variance of the Gaussian function.

For variance, we need to choose the one that makes the minimum error. The method, mean integrated squared error (MISE) can be used for evaluating the variance. The method is calculated as:

$$MISE(\sigma) = E[\int (\hat{f}_\sigma(x) - f(x))^2 dx.]$$

where f is the unknown density, f_n is f's estimation based on a set of independent and identically distributed random variable samples. E denotes the expected value with respect to the set of samples.

Using MISE method can transfer the question about finding a variance to get minimum errors to finding a limit value to make the MISE of σ to be the minimum. The variance is related to the inputs; different inputs should have different variances.

After calculating the Gaussian value for every position, we need to find the largest value. Using the Gaussian function to estimate each point, the sum of the Gaussian distributions should still be a Gaussian distribution. The largest number is the highest peak of all the Gaussian distributions. Then it uses a threshold to remove the positions which have small peak values. A small peak value means that the posterior probability of this position as part of the HAR is small, as a result, it could be random noise inside the input, such as a human passing the robot in a hallway. The threshold should be a percentage of the largest value.

The density estimation based filtering algorithm of the system can be seen in Algorithm 1.

Algorithm 1: Framework of density estimation based filtering.

Input: Human position dataset P_n points $p_1, p_2 ... p_n$, variance σ and threshold θ_r

Output: New human position dataset P'_n

for *point* p_i *in* P_n **do**
  $GaussianP_i = 0.0$;
  for *Other points* p_j *in dataset* **do**
    $GaussianP_{ij} = e^{-\frac{(x_1-x_2)^2+(y_1-y_2)^2}{2*\sigma^2}}$;
    $GaussianP_i += GaussianP_{ij}$;
  end
end
Initialize a new empty set P'_n;
Find the biggest Gaussian value in dataset as $Peak$;
for *point* p_i *in* P_n **do**
  if $GaussianP_i > \theta_r * Peak$ **then**
    Save P_i in a new dataset P'_n;
end
return New dataset P'_n;

3.2.3 Considerations for the Evaluation

The evaluation method of the density estimation based filtering part aims at evaluating how well the method works in removing the random noise. The parameter that influences the performance of the method is the threshold θ. The threshold will decide how large a density the system needs to remove noise. Thus, the performance of this part can be transferred to evaluate how well the method removes noise with different threshold values.

3.3 Clustering Algorithm

3.3.1 Method Description and Motivation

The ultimate goal of this system is generating HARs automatically. A HAR in general is a clustered region for human positions, so the basic method is clustering a set of human positions automatically. The density estimation based filtering part will generate clean position data with a large density. Our system uses a density-based algorithm for discovering clusters: Density-Based Spatial Clustering of Applications with Noise (DBScan) [29]. It groups together points that are close together in some space (points with many nearby neighbors), marking as outliers points that lie alone in low-density regions and whose nearest neighbors are too far away.

DBScan was chosen firstly because it does not need to initialize the number of clusters, which reduces the need for human input and this increases the level of autonomy of the system. Secondly, it can handle data with noise efficiently; even though we have pre-processing and density estimation based filter for removing noise, there may still be noise inside the input, and DBScan can handle that. Thirdly, it can generate different shapes of clusters; that means it is not influenced by the correlation between data significantly. Finally, it is not quite sensitive to the parameters, it is easier to get suitable parameters than other kinds of clustering algorithm.

3.3.2 Algorithm and Concepts

DBScan [29] is a density based spatial clustering algorithm. It separates the environment into several clusters which have large density. The algorithm needs to initialize two parameters: the radius Eps, which is the circle range for a given point, and the minimum point number $MinPts$, which is the minimum number of points inside a given point's neighbor domain. There are some basic concepts in this algorithm

- Neighbor domain: The area defined by a center point P and its radius Eps.
- Direct density reachable: If a point Q is inside the neighbor domain of the center point P, then P and Q are direct density reachable.

- Indirect density reachable: If there are three points O, P and Q. O and P are direct density reachable, and O and Q are direct density reachable, then P and Q are indirect density reachable.

- Center point: The point P is called a center point if the number of points from which P is direct or indirect density reachable equals $MinPts$.

The basic algorithm of the system was introduced by [29] and can be seen in Algorithm 2.

Algorithm 2: Framework of DBScan for our system.

Input: Human position dataset P_n points $p_1, p_2 ... p_n$, domain range Eps and minimum number of points in domain $MinPts$
Output: Points with their cluster number

```
function DBScan(D, eps, MinPt):
for Un-visited point P in dataset D do
    mark P as visited; NeighborPoints = regionSearch(P, Eps);
    if sizeof(NeighborPoints) < MinPts then
        Mark P as NOISE;
    else
        C = Next cluster;
        expandCluster(P, NeighborPoints, C, Eps, MinPts)
end
function ExpandCluster(P, NeighborPoints, C, Eps, MinPts):
add P to cluster C;
for Point P' in NeighborPoints do
    if P' is not visited then
        Mark P' as visited; NeighborPoints' = regionSearch(P', Eps);
        if sizeof(NeighborPoints') >= MinPts then
            NeighborPoints = NeighborPoints joined with NeighborPoints';
        if P' is not yet member of any cluster then
            Add P' to cluster C;
end
function regionSearch(P, Eps):
return all points within P's eps-neighborhood;
```

3.3.3 Considerations for the Evaluation

To evaluate the performance of the clustering part, we should focus on how well the clusters correspond to the real world environment. Considering the algorithm, the performance of the method is influenced by two parameters, radius Eps and minimum number $MinPts$. The radius and minimum number of points influence the size and the shape of each cluster. Different parameters result in a different number of clusters. Thus, the evaluation of the clustering part will be transferred

to evaluate how well the clusters correspond to the real world environment based on different parameters.

Chapter 4

Human Detection Methods

In this chapter, the methods that are used for detecting humans are presented and motivated. What to consider for the evaluation is also described.

4.1 Upper Body Detector

4.1.1 Motivation and Method Description

The state-of-the art algorithm R-FCN [15] is used to detect target features in image data. The method for human detection in this system is chosen as a deep neural network which employs by R-FCN [15] with some fine-tuned models [31].

The motivation for choosing R-FCN method is that firstly, the CNNs are powerful in handling image data; secondly, the R-FCN [15] is encapsulated methods, and is easy to use with fixed models; thirdly, because the method will generate a bounding box in the image, it is easy to show and save the images with their bounding boxes, if they have them. This is convenient for checking the results of detection. For example, we can get the robot positions in the world coordinate frame by the localization method, then transfer the human position into the world frame and compare the position of the robot to check the precision of detection.

The network uses both RGB images and depth images which are captured by an RGB-D camera to find humans and generates bounding boxes inside the image to localize the human positions. For an example, see Figure 4.1.

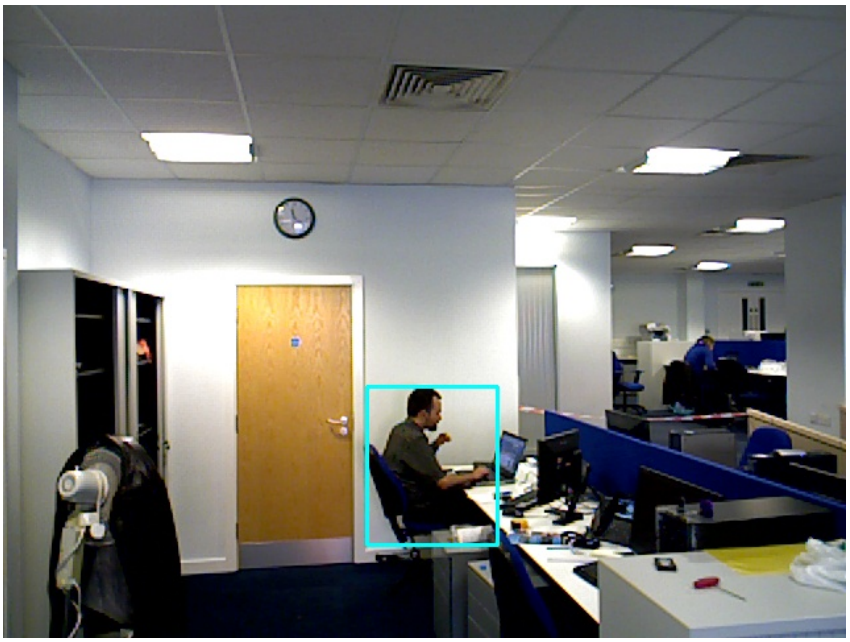

Figure 4.1: Bounding box example image.

Next, the distance between the origin and the center of the bounding box is calculated to get the coordinates the center of the bounding box in the camera coordinate system. Based on the transformation between the camera coordinate system and the real world coordinate system, the position of the bounding box's center can be transferred into the real-world frame; the system can store it as a human position with respect to the real-world frame. We use this human position to represent a human on the map.

4.1.2 Distance Calculation Method

We use fine-tuning models which consist of human features as targets of the network. The results of R-FCN [15] with these models are the positions of the bounding boxes which are used to localize humans. We still need to calculate the coordinates of the center of the bounding box with respect to the world frame and assume it to be the position of the human. We assume that the method mentioned previously will generate the 2D coordinates for the image's center as centerX and centerY, and the 2D coordinates of the bounding box's center as bboxX and bboxY, with respect to the image frame. The position of the human with respect to the camera frame should be a 3D coordinate (X,Y,Z). The Z value of the position should be related to the depth image value. Considering the random noise and detection error, the Z value is calculated as an average depth value around the bounding box center position in order to remove the NaN value. Then Z value equals to the average depth value.

After obtaining the Z value, we still need the angleX in Figure 4.2. Because of

the law of tangents, we can calculate angleX as:

$$angleX = \arctan \frac{differenceX}{focallength}$$

The difference X is the difference in x-axis between the center of the image and the center of the bounding box, i.e. differenceX = bboxX − centerX as illustrated in Figure 4.3.

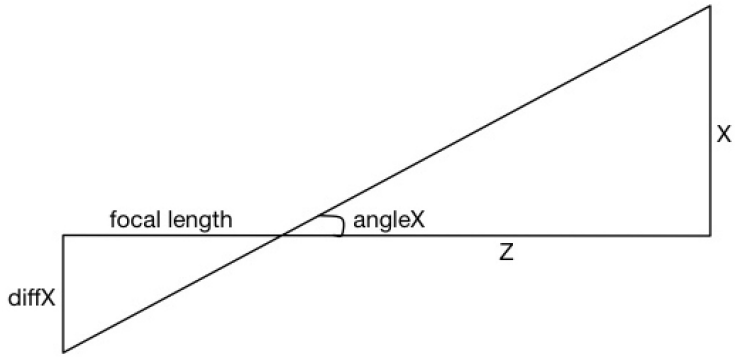

Figure 4.2: Projection rule between camera frame and image frame.

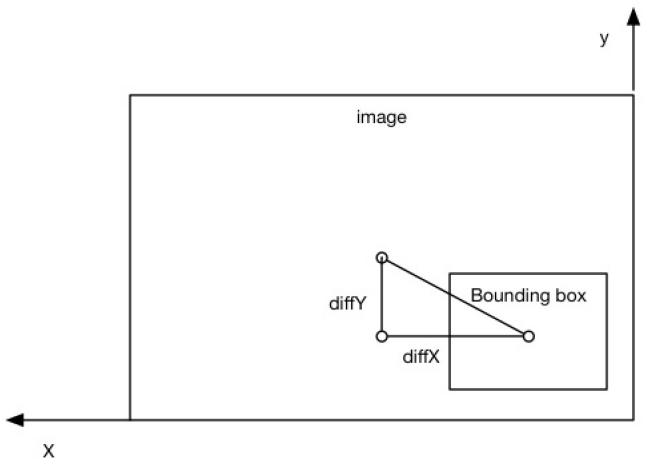

Figure 4.3: Difference calculation between image center and bounding box center.

The X value is calculated as:

$$X = Z * \tan(angleX)$$

Calculating the Y value the same way as the X value, there will be 3D coordinates (X,Y,Z) with respect to the camera coordinate system. We transform the human positions in the camera frame to the world frame using the position of the robot[1].

4.1.3 Considerations for the Evaluation

The performance of the upper body detector is evaluated by the accuracy of the human detection. This method needs image data as input and generates positions as results. The method can output images with bounding boxes containing detected humans, which are the criteria to decide the correctness of the detection. We also investigate images where is a human but no humans were detected or where detected a human with no human to find out what situations the detector has problems with.

4.2 Leg Detector

4.2.1 Motivation and Method Description

The leg detector mainly extracts features which represent legs from the laser scanner data. The method here is based on the work described in [13] and features are chosen from some of the 14 features described in [8]. After obtaining features that represent legs, the system uses a half of them as a training set to train weak classifiers. Then we use AdaBoost [9] to combine weak classifiers into a robust classifier for classifying human legs. The other half of features will be used as a testing set to test the performance of the robust classifier. Furthermore, we can get the coordinates of the human legs by the laser scanner with respect to the laser scanner's coordinate system. Based on the transforms between different coordinate systems, we can get the positions of human legs in the real world environment which represent the human positions.

The motivation for choosing the method introduced before is mainly based on the fact that these features are more representative than the rest of the others in an indoor environment; they are much easier to classify. For example, Figure 4.4 shows one set of laser data segmentation with its features. The marked black points correspond to the segmentation, and the crosses depict other features that could be read in the scan. The circle and line are fitted for the linearity and circularity features.

[1]We make use of the transform (TF) function in ROS [32] to accomplish this transformation.

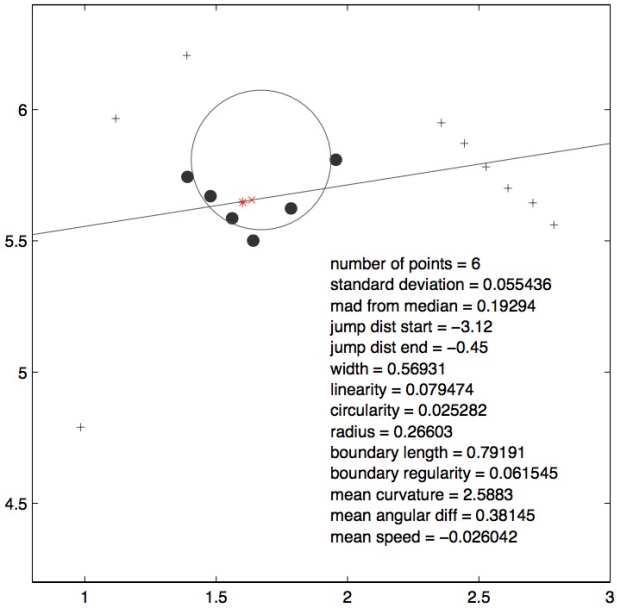

Figure 4.4: Laser segmentation with its features [13].

In our system, we choose 7 of the 14 features mentioned in [13], they are:

- Jump distance from preceding segmentation
- Jump distance to succeeding segmentation
- Standard deviation
- Mean average
- Width
- Radius
- Circularity

The reason for choosing these 7 features is because of the conclusions introduced in [13], the most informative features in a small environment such as an office are the features mentioned before. In this system, we assume that the robot will patrol indoor environments such as corridors, offices etc., so our system only chooses features which are most effective at separating human legs from other leg-like objects.

4.2.2 Considerations for the Evaluation

The leg detector uses laser data as input and generate human positions as results. The evaluation of the leg detector mainly focuses on the precision of the detector. The precision is decided by the positions it generate represent a human or not.

Chapter 5

Experimental Setup

5.1 Dataset Description

The dataset used as input for our system should contain the following data.

- RGB color images
- Depth images or depth register images (depth image registered with RGB data)
- Depth point cloud
- Map information
- Transformation between frames
- Robot localization data

We used two datasets to evaluate the performance of the whole system. The first one is the *Witham Wharf RGB-D dataset* [33] which is collected by the Lincoln Center for Autonomous Systems. The second one is the *KTH Longterm Dataset* [34] which is collected by the KTH Robotics, Perception and Learning department.

5.1.1 Dataset1: Witham Wharf RGB-D Dataset

The *Witham Wharf RGB-D dataset* is a part of the larger LCAS-STRANDS long-term dataset collection. This dataset uses benchmark visual and RGB-D localization to analyze environments. It was collected by a Scitos G5 robot with an RGB-D camera and Sick300 laser scanner over three months. The robot patrolled 8 places in the robot laboratory, stayed in each place for 10 minutes, and saved all the data in these places. It generates the dataset in a rosbag form (a bag contains the information generated in ROS system). Each rosbag contains a depth/color image, camera information, robot position, transform data, laser scan captured by the robot at a location and time that is encoded in the rosbag name, which contains day, month, year, hour, minute and location id [33].

5.1.2 Dataset2: KTH Longterm Dataset

Considering the author of the *Witham Wharf RGB-D dataset* only made the robot collect data in 8 places, some of the results could be influenced by human intervention. In order to remove this probable influence on the results, we introduce another dataset, the *KTH Longterm Dataset*, which contains data that the robot collects from its whole trajectory.

The *KTH Longterm Dataset* is part of the Strands EU FP7 project, and was collected by a Scitos G5 robot with an RGB-D camera. See Figure 5.1.

Figure 5.1: KTH Scitos G5 robot - Rosie.

The robot navigated through the KTH office environment over a period of approximately 30 days. Each observation consisted of a set of 51 RGB-D images obtained by moving the pan-tilt in a pattern in increments of 20 degrees horizontally and 25 degrees vertically [34]. The dataset was stored in point clouds form, with each point cloud corresponding to an RGB and depth image acquired by the camera while conducting the sweep, and transforms between the RGB-D sensor frame and the map frame.

5.2 Evaluations and Experiments

This section adds more details to the evaluation methods and introduces the experiments for each part of the system in detail.

5.2.1 Upper Body Detector

We will only focus on the images detected by the upper body detector and ignore other non-detected images. We define the images the detector detects as positive detection results, others are negative detection results. For positive images, the

method aims to find a bounding box in the image data. For each image which has found a human, we use an OpenCV function to draw the bounding box inside the image and save it. Because the results list which contains the human positions has the same order and timestamp as the saved images, it can evaluate the result of human detection by checking each result is a human with a bounding box or not. The performance of the upper body detector is decided by the precision and the recall. In order to calculate the precision and recall, in the experiment, we analyze every image in the dataset to find the number of

- true positive (TP): Detector detects a human where there is a human.
- false positive (FP): Detector detects a human where there is no human.
- true negative (TN): Detector detects no human in an image without humans.
- false negative (FN): Detector does not detect a human where there is a human.

Based on these four types of results, we introduce the definitions of recall and precision as formulas 5.1 and 5.2.

$$Recall = \frac{TP}{TP + FN} \tag{5.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{5.2}$$

In the experiment, the system will use the *Witham Wharf RGB-D dataset* as an input to generate a human positions set and evaluate the performance of the human detector.

5.2.2 Leg Detector

The leg detector uses laser data to generate human positions. Considering the positions generated by the upper body detector has high accuracy of human detection, when the robot records the data, the image data and the laser data will have the same timestamp. Based on the timestamp, we use the results generated by the upper body detector as criteria to evaluate the leg detector's results. In this part we use the same method as Section 5.2.1 to calculate the precision and the recall based on four types of results to evaluate the performance.

In the experiment, we will use the *Witham Wharf RGB-D dataset* as an input to generate a human positions set based on the laser data, then evaluate the detection by the classifier precision and recall, and compare the performance with the upper body detector.

5.2.3 Density Estimation Based Filtering

After calculating the suitable variance for Gaussian kernel based on the dataset, the performance of the density estimation based filtering part only relies on the value

of the threshold. We introduce two parameters to evaluate the performance of the method with different thresholds. The first one is the precision of removing true noise. The second one is the percentage of the true noise remaining after applying the density threshold. A different threshold will result in different parameters. We choose the threshold with high removing precision and low remaining percentage to be the suitable one. Then we use this threshold value to evaluate the noise removing precision of the density estimation based filtering. In order to evaluate the performance, we introduce four types of results as Section 5.2.1, they are defined as:

- true positive (TP): TP from the detector is not filtered away.
- false positive (FP): FP from detector is not filtered away.
- true negative (TN): FP from detector is filtered away.
- false negative (FN): TP from detector is filtered away.

In our experiment, we want to tune the filter to get a good trade off between keeping true positives and removing false positives. The true positive rate is known as the recall which can be calculated by formula 5.1, the false positive rate is known as the fall-out and is defined as formula 5.3.

$$fall\ out = \frac{FP}{FP + TN} \tag{5.3}$$

In the experiment, we will use the *Witham Wharf RGB-D dataset* to calculate precision, recall and fall-out values with a suitable threshold. Based on these values, we can evaluate the method by different kinds of input in different environments.

5.2.4 Clustering Algorithm

Considering the size of the dataset and the density of the data is large, the minimum number of points $MinPts$ inside a cluster does not influence the result significantly. In [29], $MinPts$ is always chosen between 1 and 10 empirically. The result of the clustering algorithm is mainly influenced by the radius Eps. The system chooses a radius Eps value based on the K-nearest distance method. For a center point P, we randomly choose a point Q and draw a 2D curve whose x-axis is Q and whose y-axis is the K-nearest distance to the center point P. The plot of our system will be shown in Chapter 6. There will be a big decrease or increase in the curve. This is because we assume that the point Q is in the neighbor domain of the center point P, thus the distance between Q and P should not be remote. If the distance of the current point Q is very big, it should be far away from the center point which means it does not belong to the neighbor domain of the center point P. As a result, we can choose the Eps value where shows this kind of change. After fixing Eps, we choose a suitable $MinPts$ value based on the clustering results, and evaluate the clustering results by the real world environment.

The clustering results can be evaluated using the map. The datasets the system uses are based on the known maps of Lincoln University and KTH. Comparing the partitions generated by the system to the real environments, the system can find the radius showing the most reasonable partition results as a best choice. For evaluation, our method is that we manually mark some regions to represent real human activity regions, such as a kitchen. Then we compare clustering results with marked regions and find four types of results as Section 5.2.1 to evaluate the similarity between clustering results with the real world environment. These results are defined as:

- true positive (TP): The system clusters a position when it belongs to a represented region.
- false positive (FP): The system clusters a position when it belongs to an unrepresented region.
- true negative (TN): The system does not cluster a position when it belongs to an unrepresented region.
- false negative (FN): The system does not cluster a position when it belongs to a represented region.

In the experiment, we will use the *Witham Wharf RGB-D dataset* and as inputs to evaluate the performance of the system. Based on these results we can calculate the precision and the recall to evaluate the clustering part. The recall and the precision are defined as formulas 5.1 and 5.2.

After evaluating the performance of every part of the system, we will use the *KTH Longterm dataset* to evaluate the generalization of the system. Considering the two environments are totally different, the correlation between the positions and the environments will not influence the results. If the system has good performance on *KTH Longterm dataset* with the same parameters when handling the *Witham Wharf RGB-D dataset*, we can assume that the generalization of the system is verified.

Chapter 6

Experimental Results and Analysis

In this chapter we present the results from the experiments described in the previous chapter. Along with the results we also provide an analysis of the results. We keep the results and analysis separated, to make it clear what are objective results and what are subjective analysis, but close, for easier reading.

6.1 Upper Body Detector

6.1.1 Results

In the *Witham Wharf RGB-D dataset*, 1943 humans were found by the upper body detector in all 3871 images. Their positions are shown in Figure 6.1.

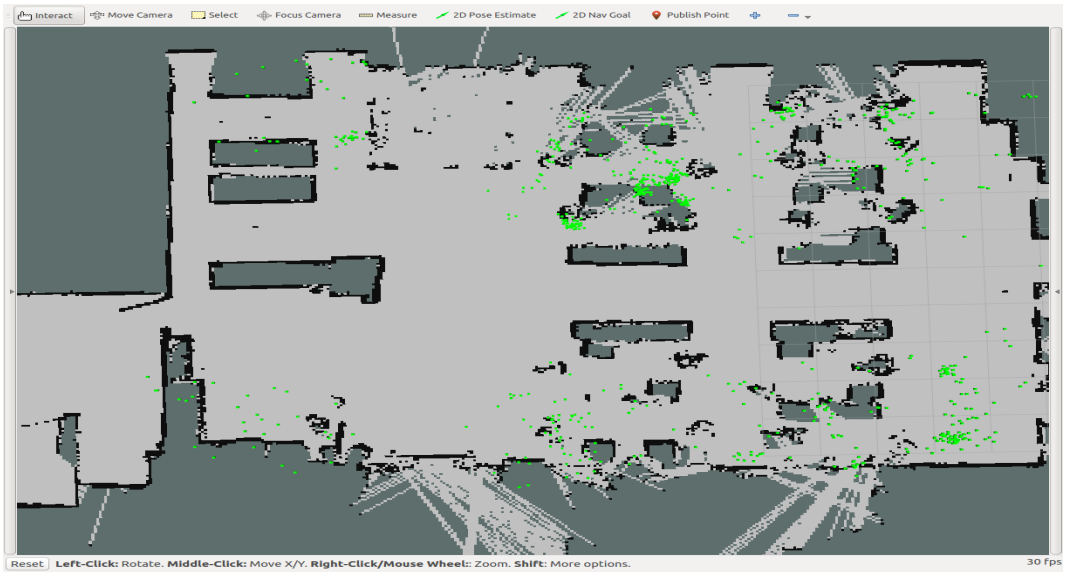

Figure 6.1: Dataset1 human detection using the upper body detector.

In our experiment, the map uses dark green grids to represent the positions the robot can not reach, and uses black lines to represent the edges of the objects. Figure 6.3-6.6 show some example human detection results obtained using the upper body detector. As it can be seen from the results, the upper body detector can correctly localize people also when they are sitting and working.

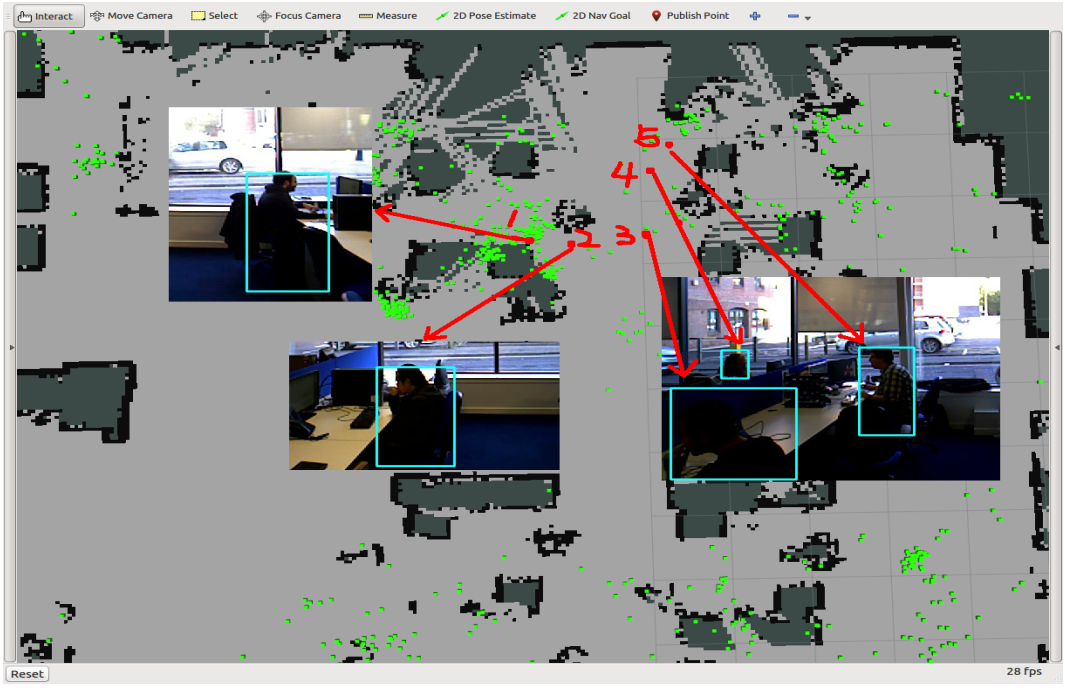

Figure 6.2: Right upper corner of Witham Wharf RGB-D dataset position results .

Figure 6.3 and 6.4 represent the left 2 red points (point 1 and 2) in Figure 6.2. Figure 6.5 and 6.6 represent the 3 red points (point 3, 4 and 5) on the right side in Figure 6.2.

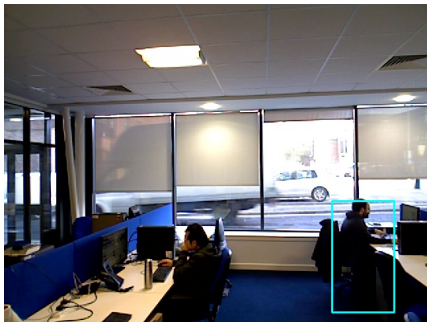

Figure 6.3: Human detection example 1.

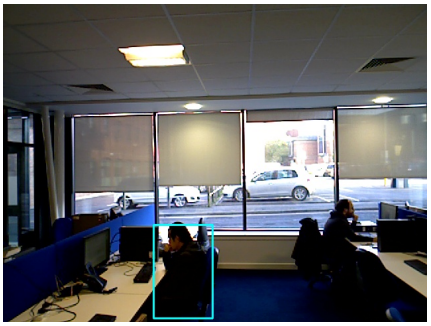

Figure 6.4: Human detection example 2.

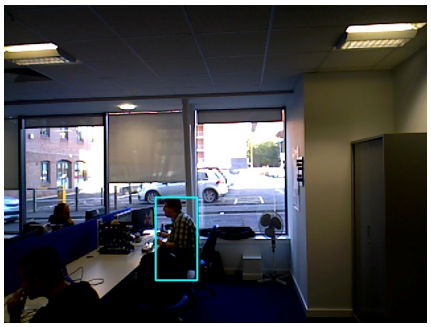

Figure 6.5: Human detection example 3.

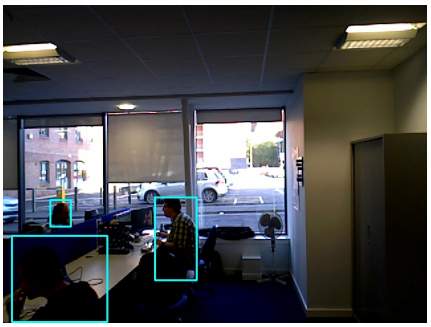

Figure 6.6: Human detection example 4.

Based on the evaluation method mentioned in Section 5.2.1, we analyze every image in the dataset to find the number of four types of results. There are 3871 images in the Witham Wharf RGB-D dataset. 1943 have been labeled by the upper body detector as containing at least one human, and 1928 as not containing any human. Example images for a false positive result and a false negative result are shown as Figure 6.7 and 6.8. The performance of the detector as defined by the four parameters in Section 5.2.1 is shown in Table 6.1.

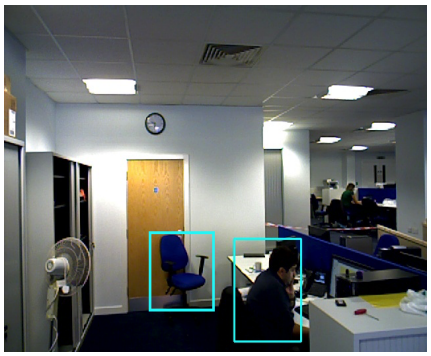

Figure 6.7: False positive human detection example.

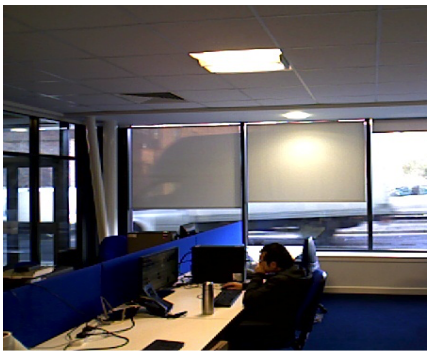

Figure 6.8: False negative human detection example.

Table 6.1: Accuracy of the upper body detector

	Positive	Negative
True	1793 (92.2%)	1791(92.8%)
False	150(7.8%)	137(7.2%)

Based on the results in Table 6.1 and formulas for recall and precision 5.1 and 5.2, we can calculate the recall and the precision to evaluate the performance of the upper body detector. The recall is calculated by $\frac{1793}{1793+137}$ which is approximately

equal to 92.9%. The precision is calculated by $\frac{1793}{1793+150}$ which is approximately equal to 92.2%.

6.1.2 Analysis

There is some incorrect detection generated by the detector, false negative results and false positive results. The reason for having false negative results can be because of incorrect depth measurements. When the detected human is wearing a dark coat, the light reflection may be very low, then the depth camera may not capture this depth value. Another reason is the distance. If the distance between the human and the camera is too great, the detector may not get enough depth and RGB data to extract features. The percentage of false negative results is only about 7.2%. The reason for generating false positive results can be that sometimes the classifier misclassifies objects because of its appearance or other conditions such as light, distance, etc. For example, in Figure 6.7, a chair is misclassified as a human.

To conclude, our analysis shows that the upper body detector should be able to provide our system with sufficient data, with high recall (92.9%) as well as high precision (92.2%).

6.2 Leg Detector

6.2.1 Results

When the method described in Section 4.2 was used, 2107 positions were generated in *Witham Wharf RGB-D dataset*. These positions are shown in Figure 6.9. Based on the evaluation method in Section 5.2.2, to evaluate the leg detector we use the results from evaluating the upper body detector where we have identified scenes with and without humans, at least as seen by the camera. As mentioned in Section 6.1, the total number of images in the dataset is 3871. 1930 have been labeled as containing at least one human and 1941 as not containing any human. Based on the definitions of different results, we use the labeled images to check if the laser detector detects a human or not through timestamps. The performance of the detector defined by the four parameters above is shown in Table 6.2.

Table 6.2: Accuracy of the leg detector

	Positive	Negative
True	1660(83.0%)	1603(85.5%)
False	338(17.0%)	270(15.5%)

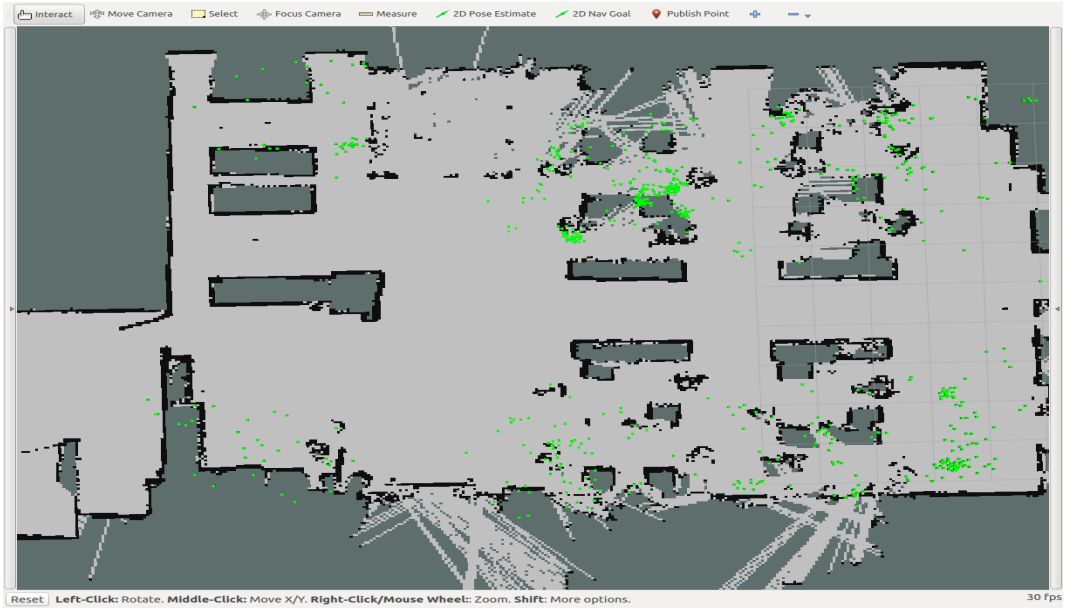

Figure 6.9: Witham Wharf RGB-D dataset human detection using the leg detector.

Based on formulas for precision and recall 5.1 and 5.2 and the results in Table 6.2, the recall of the leg detector is calculated by $\frac{1660}{1660+270}$ which is approximately equal to 86%, and the precision is calculated by $\frac{1660}{1660+338}$ which is approximately equal to 83%.

6.2.2 Analysis

By comparing the leg detector's results with the upper body detector's results through timestamps, there are three possible outcomes: i) both of the detectors find a human at the same position, ii) only one of the detector finds a human and iii) none of the detectors find a human.

Situation i), both of the detectors find a human at the same position, happens when both detectors generate positive results (detect a human) with the same timestamps. Based on the positive results in Table 6.2, the leg detector has high accuracy (83.0%) in classifying the positive results. However, the other situations below will also influence the integral accuracy.

In situation ii), one of the detector detects a human and generates a position, but the other does not. Here we analyze this in two parts. In the first part, we assume that the leg detector can detect a person while the upper body detector can not. We make a hypothesis that this is because the person is not in the view of the camera. In our experiment, we use a SICK 300 laser scanner whose field of view is about 270 degrees, and a Kinetic RGB-D camera whose field of view is about 60 degrees. Thus, this hypothesis is highly likely. For example, as in Figure 6.10, at this timestamp, the image contains three humans, and the leg detector detects two

humans, but the upper body detector only detects one. Comparing the coordinates of the positions, both detectors detect the human with a bounding box in the image, but the leg detector detects the human in the lower left corner. We can assume that the difference is caused by the different field of view of the different sensors.

The second part is that the upper body detector detects a human but the leg detector does not. We can make a hypothesis that the human is close to some objects or the legs can not be detected. For example, as in Figure 6.11, the upper body detector generates two positions with the same timestamp. However, the leg detector only generates one. This could be because the man is sitting close to the desk, he may put his leg under that table, thus a position will not be generated to represent him. Another possible reason could be that his legs are near the desk, and the desk is a leg-like object and may influence the result of the detection.

Situation iii), both of the detectors do not detect a human, happens when both detectors generate negative results; both detectors generate true negative results when there is actually no human. The leg detector generates false negative results when the leg is converged or too close to other objects. The upper body detector generates false negative results when the depth value of the human is NaN or not sufficient. We can assume a scenario for this situation as a man with a black coat sitting under a dusky light with his legs covered by the sides of the desk.

Based on the precision and recall we calculate in Section 6.2.1, we can conclude that the leg detector should be able to provide our system with sufficient data, with a high recall rate (86.0%) as well as a high precision (83.0%). If we compare to the upper body detector we see that the precision is almost the same but the recall is a bit lower for the leg detector. Results of the leg detector and the human detector show that the system can handle different input data such as the image data and the laser data. This contributes one of the requirements of the system; the system can handle data acquired from different sensors.

Figure 6.10: Situation ii part 1 example.

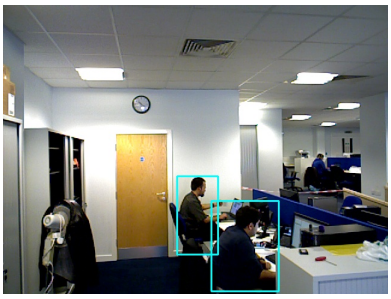

Figure 6.11: Situation ii part 2 example

6.3 Density Estimation Based Filtering

The results generated by the upper body detector based on the *Witham Wharf RGB-D dataset* will be used as inputs for evaluating the data handling part.

6.3.1 Results

Before we can evaluate the result of the density estimation based filtering part we need to determine the parameters θ, the density threshold. According to results in Section 6.1, the input only contains true positive results and false positive results. Based on the evaluation method mentioned in Section 5.2.3, we want a suitable threshold to tune the filter to get a good trade off between keeping true positives (high recall) and removing false positives (low fall-out).

For choosing a suitable threshold, the approach has been evaluated several times by us. We calculate the precision, recall and fall-out based on different thresholds and choose the threshold which has high precision, high recall and low fall-out. In our experiment, we found that the best range for the threshold is 0.2 and 0.4. If it is higher than 0.4, the method will remove most of the true values which means a low recall. If it is lower than 0.2, the method fails to remove most of the obvious noise which means a high fall-out. The performance of the density estimation based filtering part as defined by four parameters in Section 5.2.3 is shown in Table 6.3.

Table 6.3: Accuracy of density estimation based filtering of the image dataset

Threshold	TP	FP	TN	FN	Recall	Fall-Out
0.1	640	858	237	119	84.3%	78.3%
0.2	752	512	583	196	79.3%	46.7%
0.3	818	209	886	30	96.4%	24.6%
0.4	807	28	1067	41	95.1%	2.6%
0.5	417	70	1025	431	49.2%	6.3%

Based on the results in Table 6.3, we choose the threshold value with a high recall and a low fall-out, which means this threshold can remove most of false positives and keep most of true positives. Notice that when the threshold equals 0.5 it has a low fall-out (6.3%), however, it has a low recall (49.2%) compared to the threshold values in the range 0.2 to 0.4. Therefore, the threshold equals to 0.5 can not keep most of true positives, it is not a suitable value. On the contrary, when the threshold is 0.1, the system can keep most of true positives, but it will also keep most of false positives. As a conclusion, we choose 0.4 as the threshold value whose recall is high (95.1%) and fall-out is low (2.6%), which means the density estimation based filtering method with this threshold, θ equals to 0.4, can keep most of true positives and remove most of false positives for the input. With the threshold θ= 0.4 we use the set of human positions data generated by image data as an input. The results after using data pre-processing and density estimation based filtering

are shown in Figure 6.12.

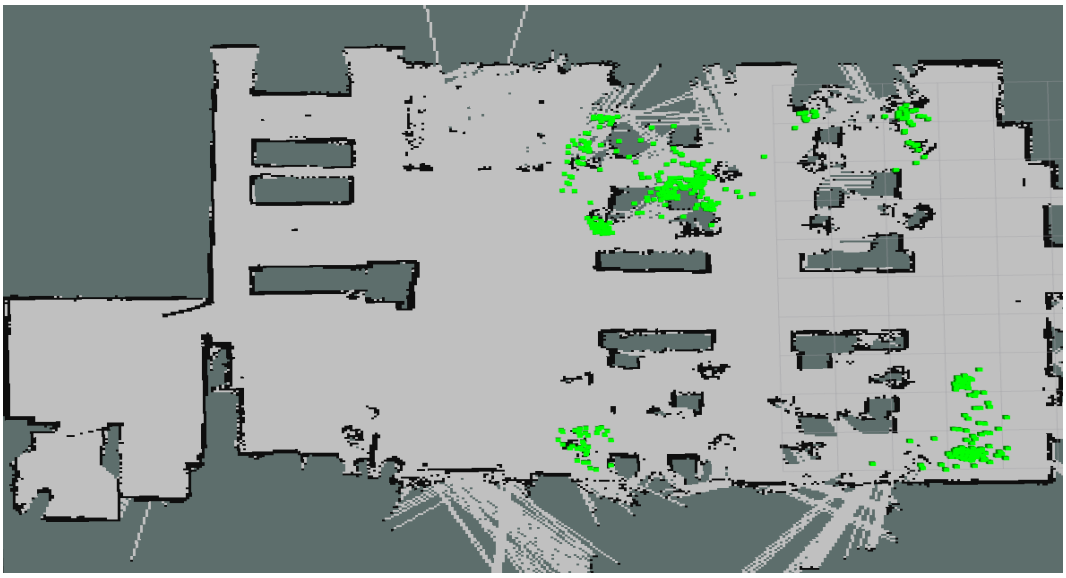

Figure 6.12: Witham Wharf RGB-D dataset density estimation based filtering results extracted from image data.

According to Table 6.3 and formula for precision 5.2, when the threshold equals 0.4, the precision of the density estimation based filter is calculated by $\frac{807}{807+28}$ which is equal to approximately 96.6%.

6.3.2 Analysis

If we compare the results generated by the upper body detector (Figure 6.1) with the results generated by the density estimation based filter (Figure 6.12), the system removes the positions that are outside the map. Also it removes a lot of small density positions such as the upper left corner. We need to evaluate whether this is a correct removal or not by checking the images for the upper left corner firstly, see Figure 6.13 and 6.14. This region contains a sofa with humans. Based on the image, they are two true positive results and should not be removed as noise. Then we need to check the Gaussian value (density) for the positions in these regions. The biggest value in this region is equal to 96.92, and the biggest value is equal to 237.82. The result of dividing the Gaussian value by the biggest value is equal to 0.398, smaller than the threshold, which means that the density of these areas is not sufficient. Therefore, positions in these areas should be removed. Green points in Figure 6.12 are positions filtered by the density estimation based filtering method. All these positions are detected human positions and have large density which can be used to generate HARs.

To conclude, based on Table 6.3 our analysis shows that the density estimation

based filtering part has high precision (96.6%), high recall (95.1%) and low fall-out (2.6%) in handling the set of positions data generated by the detector. Moreover, the density estimation based filtering part can generate a set of positions with low noise and high density which can be used for clustering.

Figure 6.13: Density estimation based filtering result example 1.

Figure 6.14: Density estimation based filtering result example 2.

6.4 Clustering

After removing most of the noise inside the two sets of human position data, the clustering method can use the sets as inputs to generate HARs. Based on the evaluation method in Section 5.2.4, the performance of the clustering part is decided by the precision and the recall compared with the real world environment.

6.4.1 Map description

The environment is described based on the Witham Wharf RGB-D dataset description in [33], it mentioned that the robot only patrolled in 8 places in a laboratory. There is one kitchen, one resting area, five offices, and a manager's office. The localizations of each region are shown in the regions map in Figure 6.15.

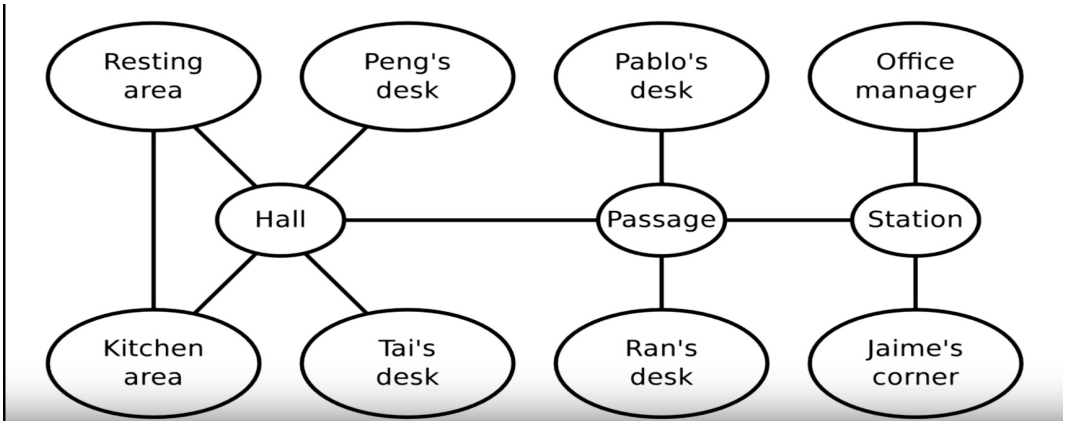

Figure 6.15: Regions map for Witham Wharf RGB-D dataset [33].

6.4.2 Parameter Selection

As mentioned in Section 3.3, there are two parameters influencing the performance of the algorithm, radius *Eps* and minimum number points $MinPts$. The radius *Eps* is chosen based on the K-nearest distance method mentioned in Section 5.2.4. It randomly chooses point Q and calculates the distance between every other point in the input sets and Q. The distances are sorted into an ascending order and a figure is created. The figure for a random point Q and distance between Q and some of the other points is shown in Figure 6.16. Based on the zoom of the curve, the *Eps* could be equal to approximately 1.4 meters for the whole Witham Wharf RGB-D dataset.

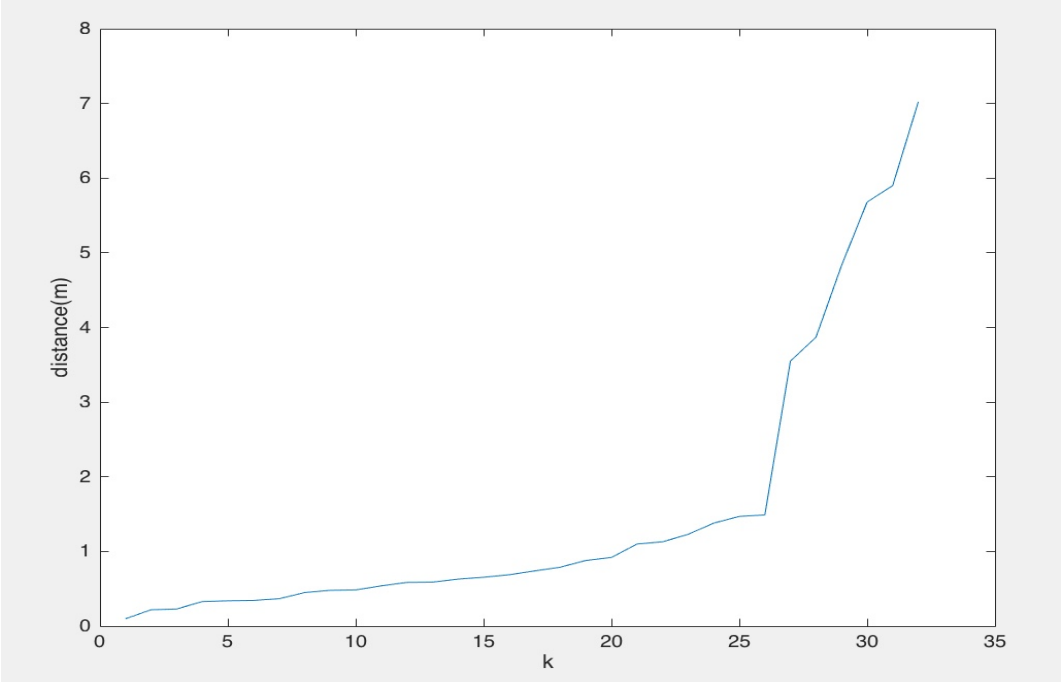

Figure 6.16: K-nearest distance figure for random point Q in Witham Wharf RGB-D dataset.

The parameter for the minimum point $MinPts$ is chosen empirically. The minimum point means that at least $MinPts$ points in a cluster. Considering the distance between two points in a HAR should be close, and the number of the points is sufficient, $MinPts$ will only influence the results slightly. Ester et al. [29] mentioned the $MinPts$ is chosen empirically in a range of 1 to 10. In our experiment, by generating the results and comparing to the real world environment several times with different parameters, the $MinPts$ equal to 3 and the radius equals to 1.4 make the clustering results most similar to the real world environment.

6.4.3 Results

Figures 6.17 shows the clustering results for each input set based on a radius equal to 1.4 and a minimum points equal to 3. Notice that the region of the cluster is not a rectangle, we only use a rectangle to represent a cluster more obviously.

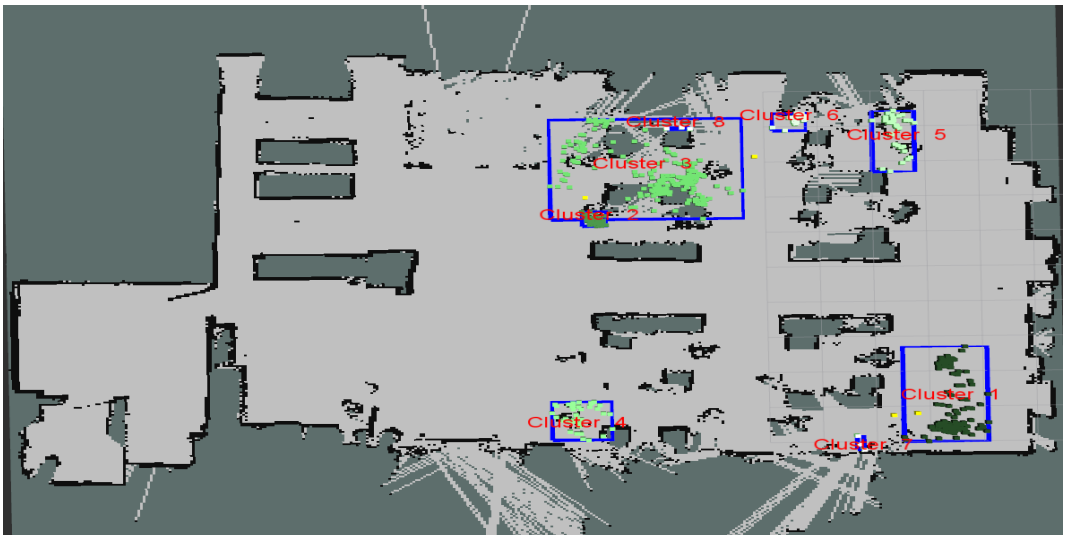

Figure 6.17: Witham Wharf RGB-D dataset clustering results generated from image data.

After generating clusters, it is necessary to check whether all of points inside a cluster are in the same region or not. For example, in Figure 6.17, considering cluster 1 at the lower right corner, there are some images (Figure 6.18 and 6.19) that represent the points inside cluster 1.

Figure 6.18: Cluster 1 example 1.

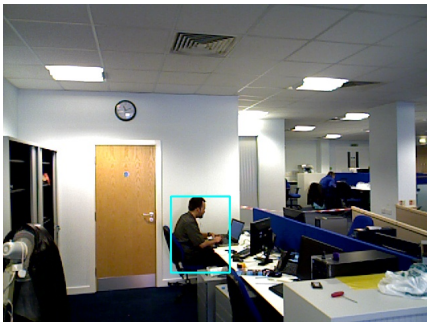

Figure 6.19: Cluster 1 example 2.

In Figure 6.18 and 6.19, there are two humans working at a long desk, these two positions should belong to the same region. After checking all points inside this cluster, it was found that in the real map, all of the points in one cluster belong

to the same area. Based on the density estimation based filtering results in Section 6.3, the density of these points should be large, thus, this region is recognized as a HAR by our system. By doing the same check on the other clusters, we reach the conclusion that each cluster represent one, or a part of a HAR. We give each region in Figure 6.15 an example image to represent it. This is shown in Figure 6.20.

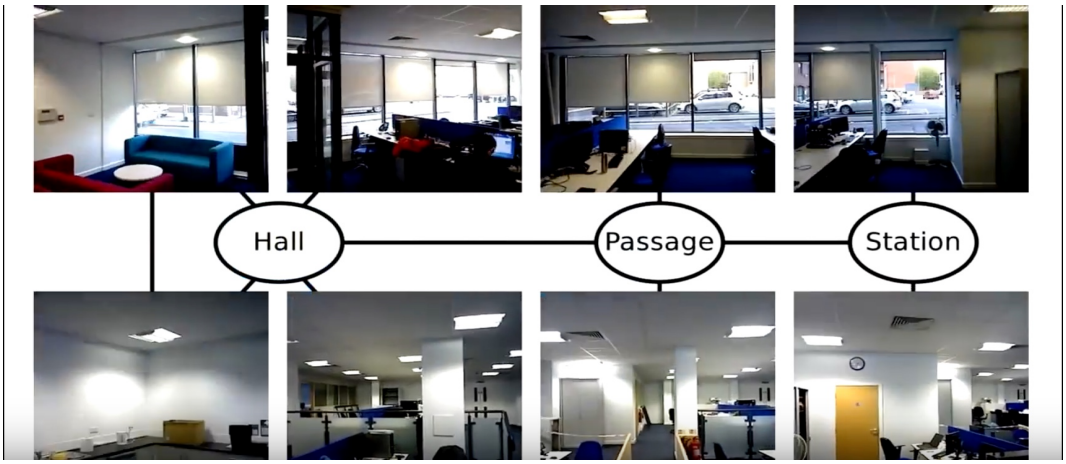

Figure 6.20: Example images for each region.

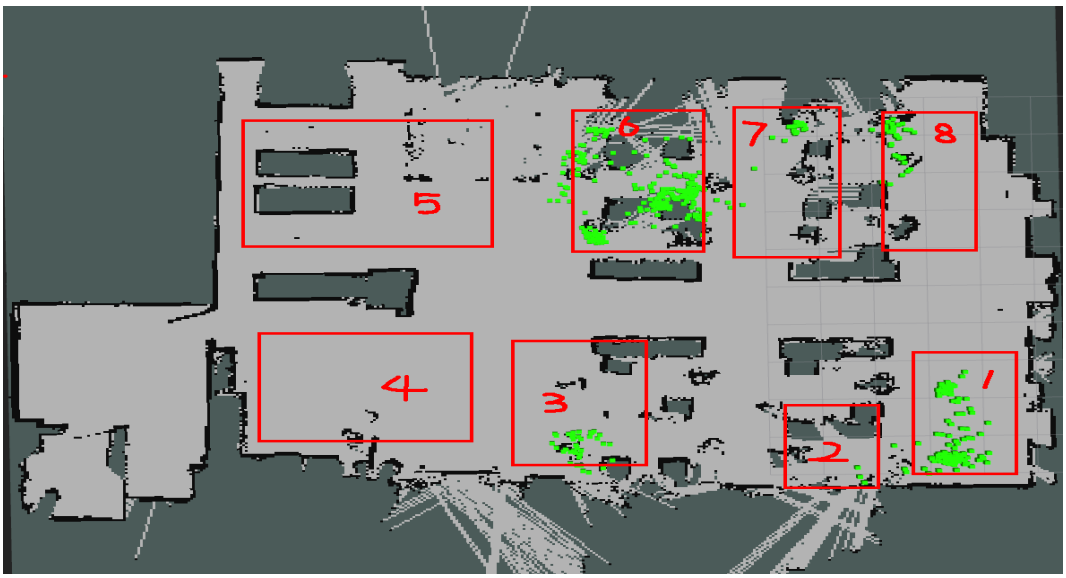

Figure 6.21: Manually regions marking results.

Based on the example images in Figure 6.20, we manually mark some regions to represent each place. The marked regions are related to the real environment,

as region 1 represents Jaime's corner, region 2 represents Ran's desk, region 3 represents Tai's desk, region 4 represents the kitchen, region 5 represents the resting room, region 6 represents Peng's desk, region 7 represents Pablo's desk, and region 8 represents the office manager's desk. The marked regions with red boundaries are shown in Figure 6.21.

We use the map containing density estimation based filtering results in order to decide if a position belongs to a region or not. Comparing Figure 6.17, clustering results of the Witham Wharf RGB-D dataset, with Figure 6.21 we can assume that cluster 1 represents region 1, cluster 7 represents region 2, cluster 4 represents region 3, cluster 2, 3, 8 represent region 6, cluster 6 represents region 7, and cluster 5 represents region 8. Based on the evaluation method described in Section 5.2.4, we can evaluate the performance of the clustering part by comparing the partitions generated by our system with the real world environment. Based on the accuracy of the density estimation based filter in Table 6.3, there are 835 positions remaining (TP+FP). The performance of the clustering part as defined by four parameters in Section 5.2.4 is shown in Table 6.4.

Table 6.4: Accuracy of the clustering part

	Positive	Negative
True	758 (90.7%)	763(91.3%)
False	77(9.3%)	72(8.7%)

The recall and the precision of the clustering part is calculated based on formulas for recall and precision 5.1 and 5.2. The recall is calculated by $\frac{758}{758+72}$ which is equal to 97.1%. The precision is calculated by $\frac{758}{758+77}$ which is equal to 90.7%.

6.4.4 Analysis

Base on the precision and the recall we calculate in Section 6.4.3, the clustering part has high recall (97.1%) and high precision (90.7%) in partitioning. And clustering results fit the real world environment to a large extent. In conclusion, the system can generate correct partitions of real world environments based on the position data.

Besides, manually, we assume the resting area and the kitchen should be HARs. However, the system ignores them and removes the points inside these two areas. In this situation, the ignorance is because the density of these regions is not sufficient compared to the other HARs. This could have a systemic reason or an environmental reason. The systemic reason is that based on the results in Section 6.1, there are 137 false negative results whose images contain humans but the system does not detect them. Also, as mentioned in Section 6.2, the field of view of the camera is small, so the robot may not capture the human even though they are present in this region. The environmental reason may because of the robot itself. The definition of an HAR is a region in which humans have frequent presence for long periods. The

robot patrols only one place during one time interval. Thus, it may not patrol in these two regions when they are filled with humans. For example, humans always take lunch at noon. During this time interval, the kitchen is filled with humans, but the robot is not patrolling in this area. Then these humans are not recorded by the robot. Sometimes, the robot patrols in the kitchen when it is filled with humans, the robot records the images and generates some positions such as the result of our experiment. No matter what the reason is, the points in these regions are significantly fewer than in HARs, and the density of these regions is significantly lower than in HARs. As a result, the system removes them and does not generate clusters in these regions.

However, the *Witham Wharf RGB-D dataset* the system uses does not contain the whole trajectory of the robot. The robot only recorded the data in these 8 places, thus, the clustering results may be influenced by the dataset itself. Considering this kind of possibility, we use the *KTH Longterm dataset* to verify the performance and the generalization of the system.

6.5 KTH Longterm Dataset

6.5.1 Map Description

Figure 6.22 shows a map of a corridor in the KTH Robotics, Perception and Learning department.

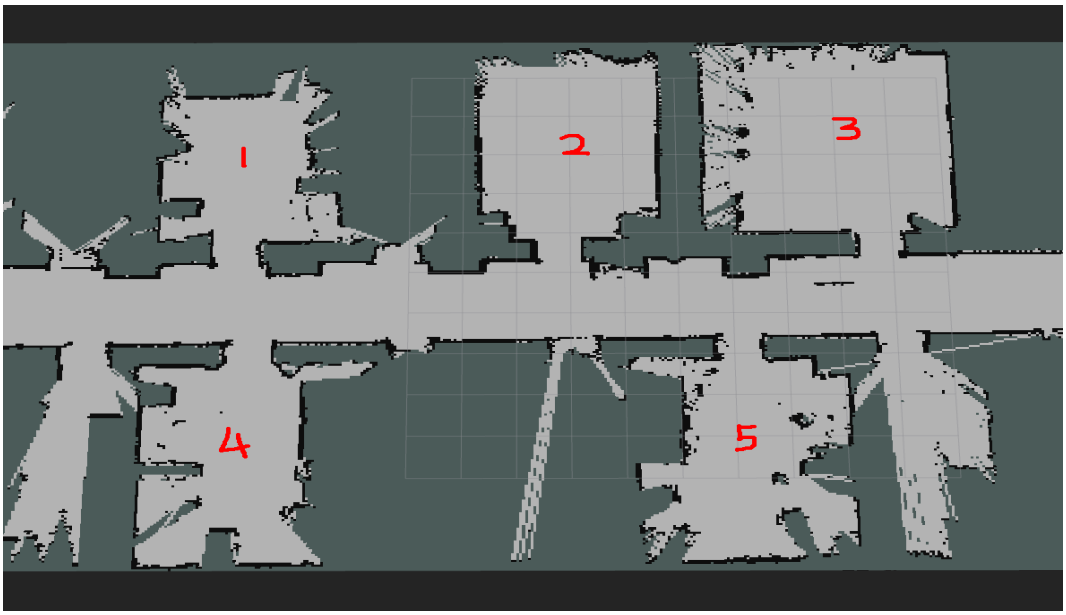

Figure 6.22: Map for KTH longterm dataset

We manually mark five regions on the map based on the real world situation:

1, 4 and 5 are offices, 2 is a meeting room and 3 is a kitchen. The marked regions are used to make comparisons with the results of the system in order to evaluate its performance. Examples of robot observations of the different rooms are shown in Figures 6.23 to 6.26.

Figure 6.23: Human in the office.

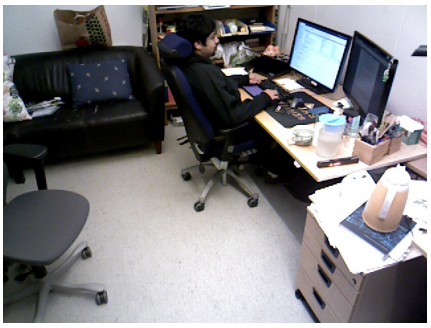

Figure 6.24: Human in the office 2.

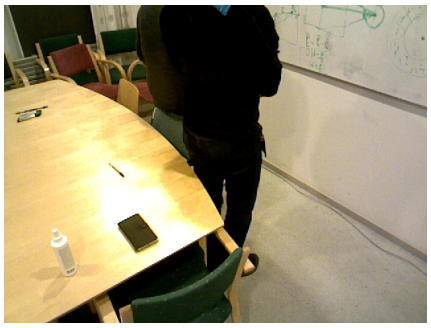

Figure 6.25: Human in the meeting room.

Figure 6.26: Human in the kitchen.

6.5.2 Results

For the *KTH Longterm dataset* we used the same parameters as the *Witham Wharf RGB-D dataset* and checked the results. The *KTH Longterm dataset* only contains image data.

The results for using the upper body detector to generate human position data are shown in Figure 6.27.

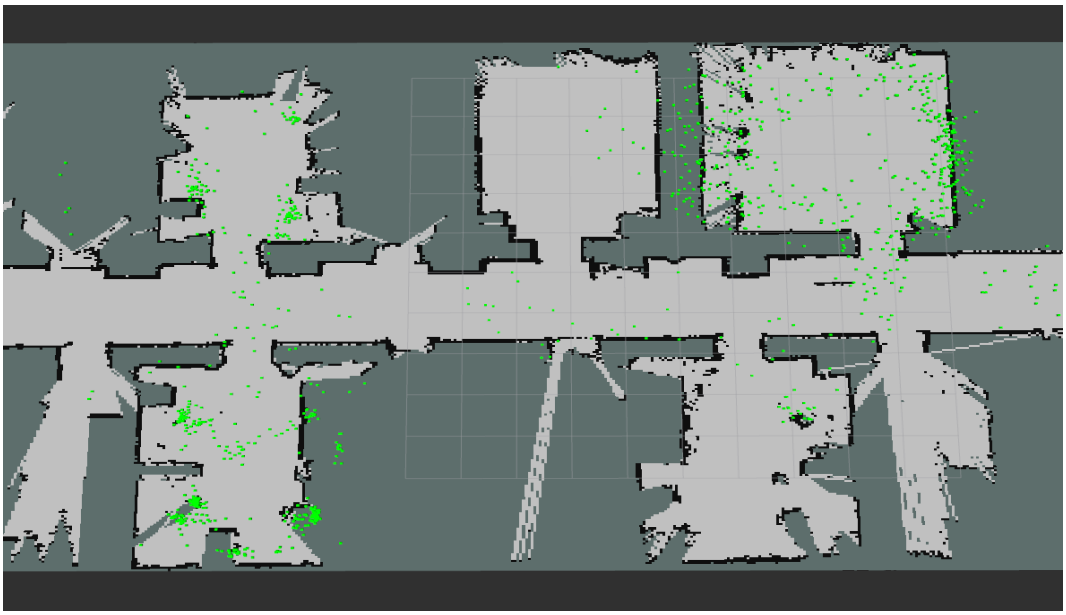

Figure 6.27: KTH Longterm dataset position results extracted from image data.

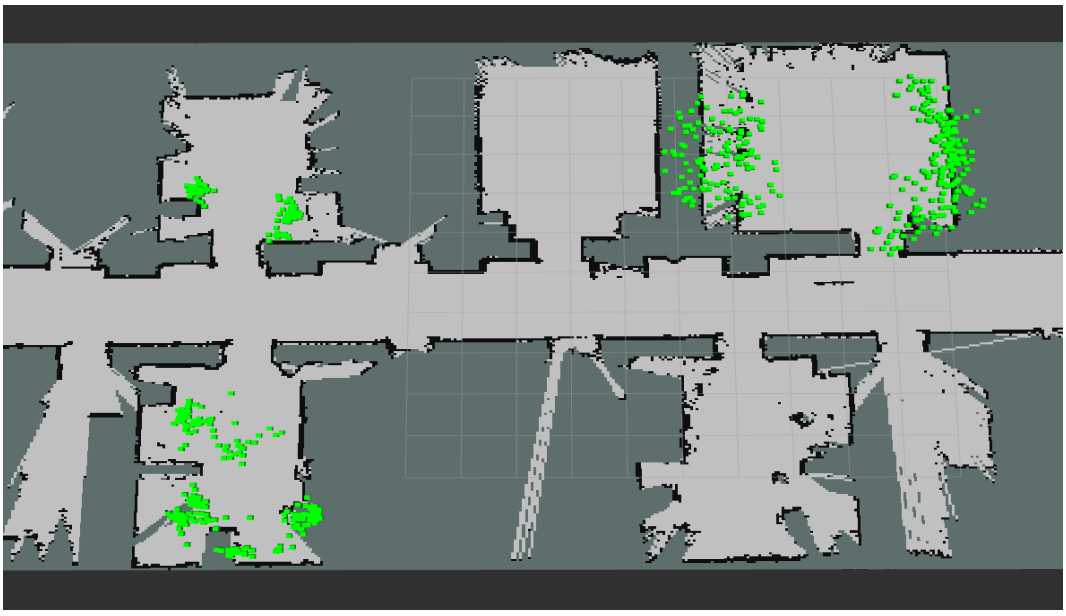

Figure 6.28: KTH Longterm dataset density estimation based filtering results.

Notice that there is a defect in this dataset. In some places, the map ends where there is furniture. For example, see Figure 6.26, the map for the kitchen is limited by the sofa not the wall. That means the size of the map in this kitchen is smaller than the real world environment. Once the pre-processing method removes

the points outside the map, it will remove plenty of positions which are valuable in the processes of the density estimation based filtering and the clustering. Using the kitchen as an example again, considering the real environment, humans are always sitting on the sofa and eating their lunch. When the robot patrols in this kitchen, it detects human and generates positions. However, in the map frame, these positions may be outside the map. The data pre-processing will remove them as noise. But manually, we should recognize these kinds of positions as parts of HARs. For this specific situation, in the experiment for the *KTH Longterm dataset*, we do not use the map as a limitation in the data pre-processing. The results after applying the density estimation based filter are shown in Figure 6.28. The results for clustering are shown in Figure 6.29.

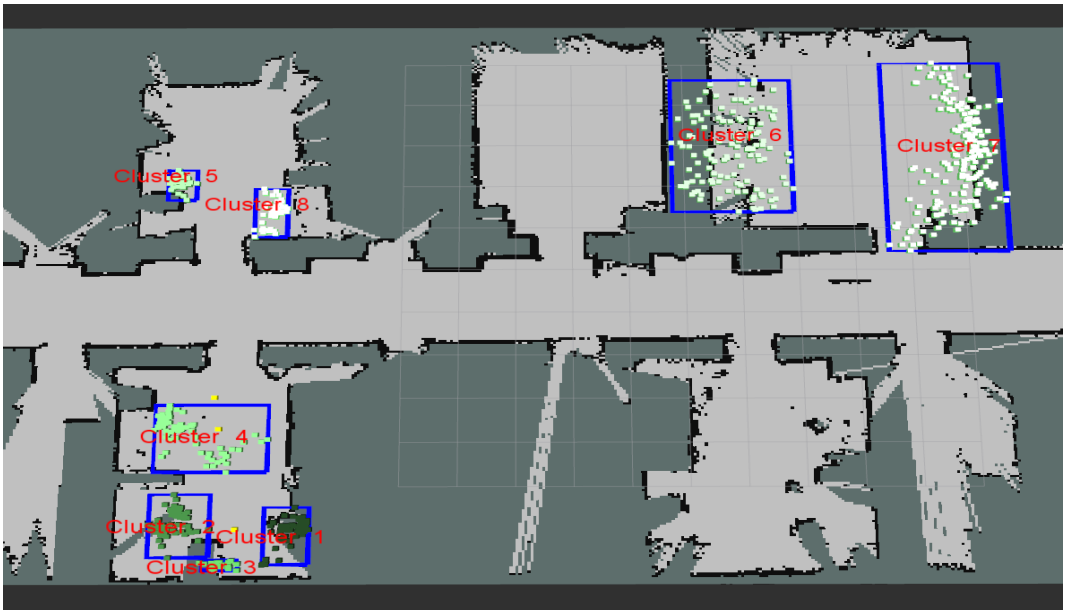

Figure 6.29: KTH Longterm dataset clustering results.

6.5.3 Analysis

According to the results, there are several clusters. In Figure 6.29, each cluster is related to the marks we made on the map, such as cluster 6 and 7 representing a kitchen (mark No.3) and cluster 5 and 8 representing an office (mark No.1). However, mark No.2 which is a meeting room has no cluster in it. We assume that the meeting room is not always occupied, and when it has humans inside, the robot perhaps patrols in another place. The definition of a HAR is a region in which humans have frequent presence for long periods, therefore, the system will not recognize this meeting room as a HAR.

Considering other clusters, such as clusters 1 to 4, which are shown in Figure 6.30, it can be seen that the system can separate a big region into small partitions.

In this situation, it generates HARs for every human working in this office.

Comparing the results of the system with the real world environment, our system can handle different input data from different environments. By using different datasets generated by different environments, the system can also generate correct partitions compared to the real world environment, this verifies the generalization of the system.

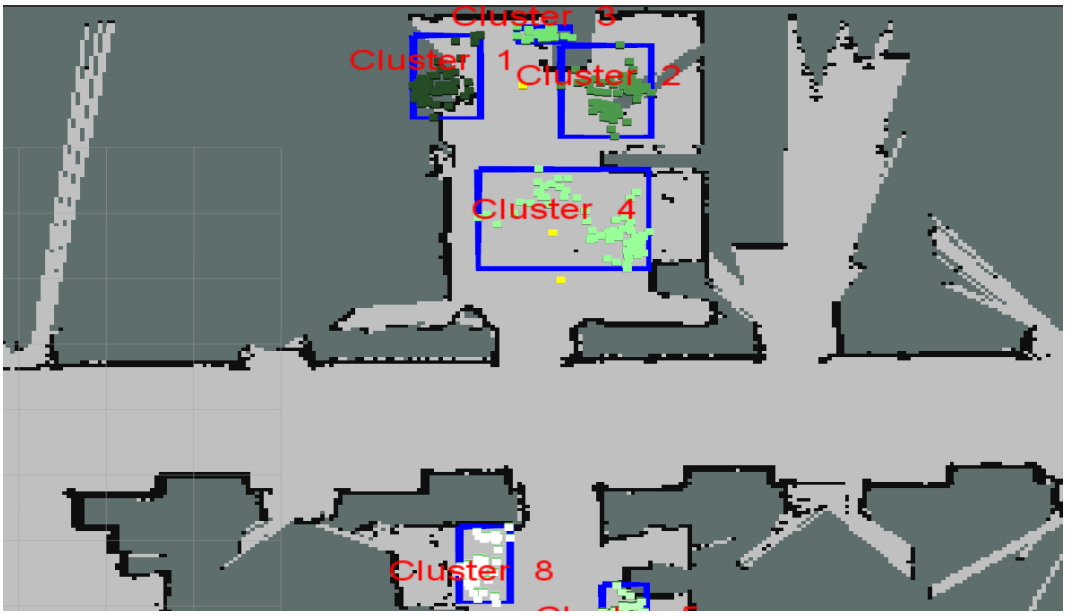

Figure 6.30: Cluster example for human office.

Chapter 7

Conclusions and Future Work

7.1 Error Sources

In the experiment results, if the partitions is different from the real world environment, we recognize them as incorrect partitions. The reason for the errors concerning the incorrect partitions could be twofold. The first one is the wrong localization of the robot. This kind of error will generate wrong positions in the dataset, then conclude incorrect partitions of the environments.

The second one is the randomness of the robot's presence during its trajectory with respect to time. In our system, because we want the system generate HARs automatically, we should assume that the robot's trajectory is not influenced by humans too much, then the automatic generating method is more reliable. Because of the randomness of the robot's presence, the robot may not patrol in a HAR such as meeting room when it is occupied by humans. The robot will record little or no position data for that region, and the region will not be partitioned. Another reason could be that the robot is not allowed entering the meeting room when there is a meeting holding inside. But to a human mind, a meeting room should be a HAR, thus, this may be an incorrect partition. Errors like this could be reduced by allowing the robot to enter the meeting room or increasing the size of the input dataset.

For evaluation results, errors could result form the human intervention. For example, in clustering evaluation, Section 6.4, we manually mark the regions based on the real world environment. Thus, the size of the boundaries should have influences on evaluation results. However, in this thesis, we assume that influences of the human intervention are slight compared with influences caused by the robot itself, such as reasons mentioned in the last paragraph. As a result, we use this kind of manually method to evaluate the system and decide to regard removing this kind of human intervention as a future work.

7.2 Conclusions

Based on the results in Chapter 6 and the system requirements in Chapter 1, there are some conclusions for the whole system.

The results in Sections 6.1 and 6.2, the results of detectors, show that the system can handle data acquired by different sensors as inputs and generate position sets which are used for HARs generation. Also the results in Sections 6.3 and 6.4, the results of the data handling part, show that the system can generate reliable environment partitions with inputs that contain noise.

The results of Section 6.5 show that the system can handle different environments and give reliable region partitions. This verifies the requirement of the generalization of the system. The two parts of the system, the detecting part and the data handling part, are independent, thus, we can conclude that the system can use independent datasets which contain human positions as inputs to generate HARs.

Based on the research question, we can conclude that our system uses human positions generated by detectors as input, uses pre-processing to remove system errors, uses density estimation based filtering to remove random noise, and uses clustering methods to generate partitions for the real world environment.

7.3 Future Improvements

There are two main improvements to be made to the system.

The first one is the timestamps handling method. We know that HARs are regions which have a time attribute, so the time interval could be a good prior probability for deciding a region is a HAR or not. The method can generate a timestamp list for each area and use time series analysis to analyze the rule of human presence in the area. If the presence of humans shows some rules, then it could be a HAR, if not, the area should have random presences with respect to time, in passing areas such as a hallway, for example.

Another improvement is that after partitioning the environment, there could be methods for classifying the partitions. The system could use different training models for the network to detect objects, then based on the object detection, classify the type of HAR. For example, if the system finds many cooking items or fruits inside a HAR, it might recognize it is a kitchen; if it finds desks and chairs inside a HAR, it might recognize it as an office.

Bibliography

[1] K. O. Arras, B. Lau, S. Grzonka, M. Luber, O. M. Mozos, D. Meyer-Delius, and W. Burgard, "Range-based people detection and tracking for socially enabled service robots," in *Towards Service Robots for Everyday Environments*. Springer, 2012, pp. 235–280.

[2] A. Fod, A. Howard, and M. Mataric, "A laser-based people tracker," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 3. IEEE, 2002, pp. 3024–3029.

[3] M. Kleinehagenbrock, S. Lang, J. Fritsch, F. Lomker, G. A. Fink, and G. Sagerer, "Person tracking with a mobile robot based on multi-modal anchoring," in *Robot and Human Interactive Communication, 2002. Proceedings. 11th IEEE International Workshop on*. IEEE, 2002, pp. 423–429.

[4] E. A. Topp and H. I. Christensen, "Tracking for following and passing persons," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 2321–2327.

[5] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers, "People tracking with mobile robots using sample-based joint probabilistic data association filters," *The International Journal of Robotics Research*, vol. 22, no. 2, pp. 99–116, 2003.

[6] D. Hahnel, R. Triebel, W. Burgard, and S. Thrun, "Map building with mobile robots in dynamic environments," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 2. IEEE, 2003, pp. 1557–1563.

[7] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the royal statistical society. Series B (methodological)*, pp. 1–38, 1977.

[8] K. O. Arras, O. M. Mozos, and W. Burgard, "Using boosted features for the detection of people in 2d range data," in *Robotics and Automation, 2007 IEEE International Conference on*. IEEE, 2007, pp. 3402–3407.

[9] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Machine learning*, vol. 37, no. 3, pp. 297–336, 1999.

[10] J. Liu, Y. Liu, Y. Cui, and Y. Q. Chen, "Real-time human detection and tracking in complex environments using single rgbd camera," in *Image Processing (ICIP), 2013 20th IEEE International Conference on.* IEEE, 2013, pp. 3088–3092.

[11] C. Dondrup, N. Bellotto, F. Jovan, M. Hanheide *et al.*, "Real-time multisensor people tracking for human-robot spatial interaction," 2015.

[12] N. Bellotto and H. Hu, "Computationally efficient solutions for tracking people with a mobile robot: an experimental evaluation of bayesian filters," *Autonomous Robots*, vol. 28, no. 4, pp. 425–438, 2010.

[13] ——, "Multisensor-based human detection and tracking for mobile service robots," *IEEE Trans. Systems, Man, and Cybernetics, Part B*, vol. 39, pp. 167–181, 2009.

[14] J. Ko, D. J. Kleint, D. Fox, and D. Haehnelt, "Gp-ukf: Unscented kalman filters with gaussian process prediction and observation models," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on.* IEEE, 2007, pp. 1901–1907.

[15] D. Jifeng, L. Yi, H. Kaiming, and J. Sun, "R-FCN: Object detection via region-based fully convolutional networks," *arXiv preprint arXiv:1605.06409*, 2016.

[16] H. Karaoguz, N. Bore, J. Folkesson, and P. Jensfelt, "Human-centric partitioning of the environment," in *IEEE International Symposium on Robot and Human Interactive Communication (ROMAN2017)*, Lisbon, Portugal, 2017, accepted.

[17] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena, "Semantic labeling of 3d point clouds for indoor scenes," in *Advances in neural information processing systems*, 2011, pp. 244–252.

[18] I. Kostavelis and A. Gasteratos, "Semantic mapping for mobile robotics tasks: A survey," *Robotics and Autonomous Systems*, vol. 66, pp. 86–103, 2015.

[19] A. J. Trevor, S. Gedikli, R. B. Rusu, and H. I. Christensen, "Efficient organized point cloud segmentation with connected components," *Semantic Perception Mapping and Exploration (SPME)*, 2013.

[20] L. Kunze, C. Burbridge, M. Alberti, A. Thippur, J. Folkesson, P. Jensfelt, and N. Hawes, "Combining top-down spatial reasoning and bottom-up object class recognition for scene understanding," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on.* IEEE, 2014, pp. 2910–2915.

[21] P. Viswanathan, T. Southey, J. J. Little, and A. Mackworth, "Automated place classification using object detection," in *Computer and Robot Vision (CRV), 2010 Canadian Conference on*. IEEE, 2010, pp. 324–330.

[22] A. Pronobis and P. Jensfelt, "Large-scale semantic mapping and reasoning with heterogeneous modalities," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3515–3522.

[23] S. Hemachandra, M. R. Walter, S. Tellex, and S. Teller, "Learning spatial-semantic representations from natural language descriptions and scene classifications," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 2623–2630.

[24] B. W. Silverman, *Density estimation for statistics and data analysis*. CRC press, 1986, vol. 26.

[25] Z. Huang, "Extensions to the k-means algorithm for clustering large data sets with categorical values," *Data mining and knowledge discovery*, vol. 2, no. 3, pp. 283–304, 1998.

[26] S. Jiang, J. Ferreira, and M. C. González, "Clustering daily patterns of human activities in the city," *Data Mining and Knowledge Discovery*, pp. 1–33, 2012.

[27] I. Jolliffe, *Principal component analysis*. Wiley Online Library, 2002.

[28] M. Ghaseminezhad and A. Karami, "A novel self-organizing map (som) neural network for discrete groups of data clustering," *Applied Soft Computing*, vol. 11, no. 4, pp. 3771–3778, 2011.

[29] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[30] J. Kittler and J. Illingworth, "On threshold selection using clustering criteria," *IEEE transactions on systems, man, and cybernetics*, no. 5, pp. 652–655, 1985.

[31] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.

[32] "ROS tf ros wiki," http://http://wiki.ros.org/tf, accessed: 2010-09-30.

[33] T. Krajník, J. P. Fentanes, O. M. Mozos, T. Duckett, J. Ekekrantz, and M. Hanheide, "Long-term topological localization for service robots in dynamic environments using spectral maps," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[34] R. Ambrus, J. Ekekrantz, J. Folkesson, and P. Jensfelt, "Unsupervised learning of spatial-temporal models of objects in a long-term autonomy scenario," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 5678–5685.

Appendix A

Social Aspects

A.1 Sustainability and Ethics

HAR generation and classification can help humans decide what kind of robot should be used in a specific region. For example, in a kitchen, we can use a cooking robot. The robot will only be activated when a human asks for food. This kind of robot does not need to move around, thus it can save energy and achieve sustainable development.

The system itself does not contribute to sustainability much, but it is aimed at helping create autonomous robots. Autonomous robots can play an important role in sustainable development. Industrial robotics currently increases the productivity of human workers, which could actually have a destabilizing effect on sustainable development. Because with the increment of robots, workers will lose their jobs. However, autonomous systems make an enormous contribution to production. For example, robots could be able to repair products by handling tiny components. This could make it possible to repair and/or upgrade electronic appliances or peripherals rather than scrapping them. In this situation, robots would contribute to sustainability.

The results of this thesis could be used for removing potential danger from human-robot coordination environments. For example, in future factories humans will work with robots, and analyzing human activity regions could provide robots with safer navigation; robots could be more sensitive in HARs in order to avoid hurting humans.

However, there are also ethical issues in the area of autonomous systems. For example, the system collects data for HAR generation. This could have potential risk on data privacy issues. In our system, the robot detects humans and saves their images. This data could be used to analyze the activities of humans. If the data is collected without humans' permission, there will be a risk of infringement of individual rights.

A.2 Society

From a societal point of view, HAR recognition could help big organizations to analyze tendencies and trends in some issues. For example, it could help the World Health Organization (WHO) analyze the development of public health emergencies. If the system had permission to collect data in hospitals, it could collect data from each consultation room representing a certain health issue. The system will generate regions with their density. The WHO can sort the average density for each kind of health issue. Based on that, they could analyze the trends in a current health issue and attempt to mitigate it.

Moreover, using robots for tedious work instead of humans will allow humans more time to work on other things that robots can not do. However, tasks taken on by robots will decrease the opportunity for humans to get a job and cause a loss of labor. And this will also lead to a concentration of wealth in small parts of human society.

The sum of the workload and the division of the labor in a stable society are constant. Senior producers produce social wealth and cheap labor forces produce social foundations. Robots will replace cheap labor forces in some fields. However, senior producers will not be replaced. As a result, the balance of dividing social wealth between senior producers and cheap labor forces will change. Considering that robots need less resources to produce products than cheap labor forces, the surplus social wealth will increase. For example, if the division of the social wealth in a stable society is 30% to senior producers and 70% to cheap labor forces, after using robots to replace cheap labor forces, the part used for producing social foundations may decrease to 50%, then there is 20% left. This surplus social wealth will flow to senior producers, and they will it to make more contributions such as new technologies. This is a positive influence on social development. On the contrary, this is a negative influence on social stability, because social wealth concentrates with a small group of humans and many humans lose their jobs.