PLACE LEARNING WITH A HUMAN TRACKING MOBILE ROBOT

by

Serhat İşcan

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2013

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University
2019

ACKNOWLEDGEMENTS

First, I would like to express my gratitude to my supervisor, Prof. H.Işıl Bozma for all of her support and assistance during the thesis. She has come up with bright solutions to any issue I have been facing during my research and has encouraged me throughout the whole period.

I would like to thank Prof. Yağmur Denizhan and Prof. Erhan Öztop for their participation in my thesis committee.

The project has been supported in part by TÜBİTAK-EEEAG project #118E857.

I would like to express my thanks and appreciation to Kadir Türksoy, Doğan Patar, Meriç Durukan and Kemal Bektaş for their support, assistance, and contributions to my work. I also would like to thank all previous members of ISL that I worked with before.

Finally, I would like to use this opportunity to express my sincere gratitude to my family and all of my friends for their encouragement and endless support.

ABSTRACT

PLACE LEARNING WITH A HUMAN TRACKING MOBILE ROBOT

This thesis is concerned with human-guided spatial cognition. This is an important problem in human-robot interaction as it can yield human-like knowledge of places. It is a difficult problem as it requires the robot to have three capabilities that can function in an integrated manner - namely human tracking, human following, and spatial reasoning while doing so. The goal of tracking is human detection and estimating the relative position of the human as long as s/he remains within the robot's field of view. We consider a robot endowed with an RGB-D sensor and propose an approach that consists of five stages: human detection, target region selection, detection improvement, target distance calculation, and relative position derivation. Human detection is based on one of two alternative existing methods depending on whether the processing power or accuracy is of priority. In case the target cannot be found or if more than one target is found, the continuity of tracking is ensured through position estimation. The goal of human following is to ensure that the robot keeps the target human within its field of view - even if the human is bodily moving. This is achieved using a reactive navigation approach - based on previous work. As such, the robot is able to follow the target while avoiding collisions along the way. The final stage is spatial reasoning. Here, we utilize a topological spatial cognition model. In this model, a place refers to an area with spatial extent as defined by its appearances. The model works in conjunction with a place memory and places are detected, recognized or learned if necessary. Our experimental results indicate that the three capabilities can operate in an integrated manner.

ÖZET

İNSAN TAKİP EDEN ROBOT İLE ORTAM ÖĞRENİMİ

Bu tezde, bir gezgin robotun bir insanı takip ederek uzamsal bilişini geliştirmesi ve kullanması problemi üzerinde odaklanılmaktadır. Bu problem, robotun insan benzeri mekan bilgilerine sahip olmasına imkan sağlayacağından, insan-robot etkileşimi için son derece önemli bir problemdir. Aynı zamanda zor bir problemdir; zira robotun insanı görsel olarak takibi, fiziksel olarak hareket ederek izlemesi ve bunları yaparken de uzamsal muhakemesini işletmesi gibi farklı üç beceriye sahip olması ve bunların tümleşik olarak çalışabilmesi gerektirmektedir. İnsanın görsel olarak takibi için insanın sezimi ve sonra da göreceli konumunun kestiriminin yapılması gerekmektedir. Bunun için, üzerindeki RGB-D algılayıcı verilerini kullanarak, beş aşamalı bir yöntem önerilmiştir: insan sezimleme, hedef bölgenin bulunması, sezimin iyileştirilmesi, derinlik hesabı ve göreceli konum hesabı. Hedefin bulunamadığı veya birden fazla hedefin bulunduğu durumlarda, insan seziminde ortaya çıkabilecek bu eksiklikler doldurularak izlemenin sürekliliği sağlanmaktadır. İnsan sezimi için, gerekli işlem gücü ve doğruluk bakımından birbirinden farklı avantajlara sahip olan iki mevcut yöntem kullanılmıştır. Takip edilen insanın robotun görsel alanından çıkması durumunda, robotun fiziksel olarak hareket ederek takibe devam edebilmesi gerekmektedir. Bunun için, önceki çalışmalarda geliştirilmiş olan tepkin yöngüdüm yaklaşımı kullanılmaktadır. Böylece robot hem hedefe kilitlenmekte, aynı zamanda da karşılaşabileceği engellerden kaçabilmektedir. Robotun hareket ederken, aynı zamanda uzamsal muhakemesini de çalıştırabilmesi gerekmektedir. Burada topolojik uzamsal biliş modeli kullanılmıştır. Bu model robotun içinden geçtiği mekanları önce sezimlemesine, devamında tanımasına veya öğrenebilmesine imkan sağlayan bir modeldir. Robotla yapılan deneylerde, önerilen yaklaşımla robotun insan rehber tarafından götürüldüğü mekanları öğrenebildiği ve daha sonra tekrar gittiğinde de tanıyabildiği gösterilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

A	Kalman state transition matrix
$a(\hat{o}_{k-1})$	Covered area of the candidate region $\hat{o}_{k-1}$
$a(u_{kj})$	Covered area of the candidate region u_{kj}
$c(t) \in R^2$	Position of the robot
$d(o)$	Depth reading of pixel o
H	Kalman measurement matrix
g_k	Target position as navigation goal
$g_N(D)$	Minimum cost function for recognition decision-making
h_k	Height of the Kalman predicted region
$I(c_k) \in R^d$	Encoded descriptor for location c_k
m_k	Center point of depth extraction in the target region
M_p	The set of associated appearances for each learned place
N_1	Width of the input image
N_2	Height of the input image
$N_\epsilon(m_k)$	ϵ neighborhood of m_k
$\mathcal{O}$	The set of encountered obstacles
O_w	The set of obstacles inside the working radius
$\hat{o}_{k-1}$	Target region at frame $k-1$
$\mathcal{P}$	The set of learned places
r_w	Working radius in reactive navigation
S	Width and height of each grid piece constituted for YOLO
$\mathcal{U}_k$	Set of local candidate regions
u_{kj}	Center pixel of each of the candidate regions
v_{max}	Linear speed limit for the robot
w_k	Width of the Kalman predicted region
w_{max}	Angular speed limit for the robot
x_k	Kalman state vector at discrete time k
X_k	Relative horizontal distance to the human target

y_k	Kalman measurement vector at discrete time k
Z_k	Relative vertical distance to the human target
$\alpha(t) \in S^1$	Heading of the robot
α_ν	Gradient factor of the reactive controller
$\beta_{\mathcal{O}}(x)$	Encoded distance to all so-far detected obstacles
$\gamma_{g,\theta}$	Encoded distance to human in terms of position and heading
θ_k	Associated heading towards human target
θ_M	Horizontal field of view angle of the sensor
μ_k	Kalman process noise at discrete time k
ν_k	Kalman measurement noise at discrete time k
ρ	Radius of the differential wheel robot
ρ_k	Robot-human target distance at discrete time k
ρ'_k	Approximation of ρ_k distance
Σ_x	Process noise covariance matrix for Kalman filter
Σ_y	Measurement noise covariance matrix for Kalman filter
τ_a	Maximum pixel area change ratio between two frames
τ_c	Goal proximity threshold for position
τ_d	Maximum acceptable pixel distance between two frames
τ_r	Recognition cost threshold
τ_α	Goal proximity threshold for heading
τ_μ	Mean threshold for appearance informativeness
τ_ρ	Maximum allowed ratio of change in consecutive distance readings
τ_σ	Variance threshold for appearance informativeness

LIST OF ACRONYMS/ABBREVIATIONS

2D	Two Dimensional
APF	Artificial Potential Fields
CNN	Convolutional Neural Network
GPU	Graphical Processing Unit
HOG	Histogram of Oriented Gradients
IR	Infrared Radiation
NMS	Non Maximal Suppression
RGB	Red Green Blue
RGB-D	Red Green Blue and Depth
ROS	Robot Operating System
SLINK	Single Linkage Clustering
SVM	Support Vector Machine
ToF	Time of Flight
TSC	Topological Spatial Cognition
YOLO	You Only Look Once

1. INTRODUCTION

Spatial cognition is a critical ability in human-robot interaction. As such, the robot can be aware of its surroundings and thus can possibly accomplish tasks that require spatial reasoning. Spatial cognition enables the robot to relate to the place it is in. One of the important concepts is the definition of a place as an extended spatial area with perceptual signatures and possibly (although not necessarily) physical boundaries. In case of missing or unreliable odometric data, place knowledge is solely based on appearances. Typically, the robot accumulates its appearance knowledge through teleoperation. More recently, there has been work on autonomous exploration through human-robot interaction [1].

In this thesis, we consider a third alternative - namely robot evolving its spatial cognition as guided by a human. As such, the robot can learn from the human guide by following him/her and accumulate knowledge about places accordingly. This is an important problem in human-robot interaction as it can yield human-like knowledge of places and thus can possibly make human-robot interaction easier. Our goal is to build a mobile robot that is capable of following a human target and simultaneously activating its spatial cognition. The robot is assumed to be endowed with an RGB-D Kinect sensor as shown in Figure 1.1. The incoming color and depth data are used for sensing. It is a difficult problem because it requires the robot to be both aware of the human guide as well as its surroundings while also possibly moving simultaneously.

1.1. General Approach

The proposed approach consists of three separate stages that function in an integrated manner: human tracking, human following and human-guided spatial reasoning. The flowchart of the general approach is given in Figure 1.2.

Figure 1.1. Mobile robot with a RGB-D sensor

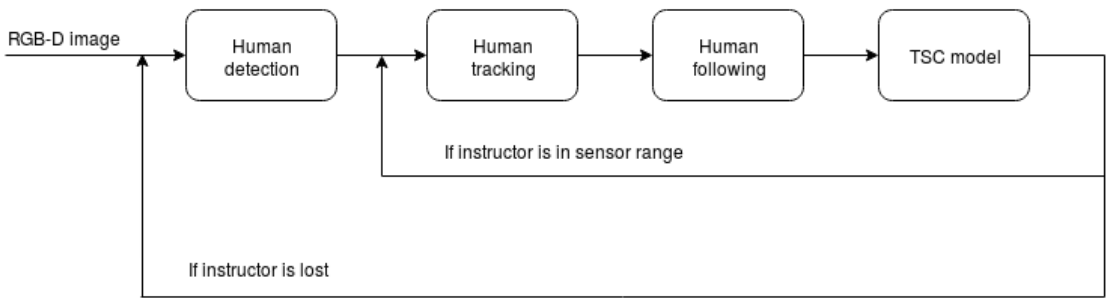


Figure 1.2. Flowchart of the proposed approach

(i) *Human Tracking:* The goal of tracking is to detect a human and to estimate the relative position of the human as long as s/he remains within the robot's field of view. The proposed approach consists of five stages: human detection, target region selection, detection improvement, target distance calculation, and relative position derivation. Human detection is based on one of two alternative existing methods depending on whether the processing power or accuracy is of priority. The detection action results in determining rectangular image regions that contain the human targets inside. The detected human is tracked through predicting the movement of the local region. In order to deal with possible temporary lapses in detection, the continuity of tracking is ensured by utilizing a Kalman filtering based estimation approach. In case of detecting multiple human targets, one region is selected based on geometric proximity as well as appearance similarity.

(ii) *Human Following*: The goal of human following is to ensure that the robot keeps the target human within its field of view - even if the human is bodily moving. The proposed approach is based on reactive navigation [2]. The method provides a series of waypoints and headings towards the target, and dynamically change its path if a new obstacle is observed. This way, smooth movement of the robot is achieved without crashing the robot elsewhere on the path.

(iii) *Spatial reasoning*: The goal of spatial reasoning is to enable the robot to either learn or recognize the places it navigates through. This is done continuously as the robot is moving. We use a previously proposed topological spatial cognition (TSC) model [3]. In this model, a place refers to a spatially extended area as defined by a set of appearances. The appearances are internally encoded using bubble descriptors [4]. The robot detects new places, then either recognizes or learns the detected place. Differing from previous work, this is accomplished while either visually tracking or bodily following a human. As such, all incoming visual data now contain human targets.

1.2. Contribution

The major contributions of this thesis are as follows:

(i) *Human Tracking*: We propose a method that consists of five components with two alternatives for human detection. While the individual components are not new, to the best of our knowledge, the proposed pipeline is. Using this approach, the robustness of tracking is improved and mistakes due to environment and faulty or missing detections are eliminated.

(ii) *Human-guided spatial cognition*: We propose a complete integrated approach to human-guided spatial cognition. As such, the robot can acquire the spatial knowledge from the human guide through following him/her around.

1.3. Thesis Outline

The outline of this thesis is as follows:

- Human tracking is presented in Chapter 2. First, related literature is summarized. Following, our proposed method consisting of five stages is explained in detail. Experimental evaluation results are then discussed.
- In Chapter 3, human following is considered. First, related literature is provided. Reactive navigation is explained and experimental results are evaluated.
- In Chapter 4, human-guided spatial place cognition is presented. The adopted TSC model is summarized. Experimental results for human-guided and teleoperation based movements of the robot will be presented and discussed.
- Chapter 5 will conclude the report with a brief summary and ideas for future work.

2. VISUAL TRACKING OF A HUMAN

2.1. Introduction

The goal of tracking is to detect a human and to estimate the relative position of the human as long as s/he remains within the robot's field of view. This needs to be done continually and as close to being in real-time as possible. It is a difficult task due to illumination variations as well as environmental clutter. In most cases, simplifying assumptions are made in order to deal with these difficulties. For example, a robot may rely on textural features and detect the human target if (s)he wears a red shirt. If that is not enough, the robot will mostly need multiple sensing modalities and higher processing power [5]. This chapter is focused on visual tracking of a human target that is in robot's (or specifically its sensor's) field of view. It is assumed that the robot has a sensor that provides RGB-D data, such as Kinect, and that there is only one person in the field of view when the robot starts working. The main goals of this chapter are the following:

- A tracking method with high recall and precision. Otherwise, the robot is likely to track false targets.
- The robot should track the first person it detects and ignore other people on the image.
- Tracking results should be continuous, hence if human detection techniques cannot find any human target on the image, these deficiencies should be filled with predictions. If the missing detections last for a longer period, the robot may conclude that the tracked target left the sensor's field of view.

The outline of the rest of this chapter is as follows: First, related literature will be summarized. Following, the overall approach is presented. Then, each stage is explained in detail. The chapter will conclude with experimental results which demonstrate that the robot is able to track a human within the field of view with acceptable accuracy in real-time.

2.2. Related Literature

Human detection and tracking on a mobile robot is a popular topic among researchers and there has been extensive work done in this area. Related work can be categorized under many topics such as utilized sensors, their placement on robots or detection approach. These selections are tailored to fulfill the actual needs of the robot in a specific scenario. For example, if a robot needs to follow a specific human target, face recognition will be required. Then, a sensor that provides RGB data should be used and this sensor should be placed conveniently on the robot so that faces of the people around can be observed.

In most work, sensing has been based on cameras, lasers or sonars. In general, using only one sensing modality has proven to be reliable under restricted conditions. For example; while a robot may use only a laser sensor to track the legs of a human target, the resulting system assumes the environment to be both known and static [6]. Thus, using multiple sensing modalities has been shown to enable more flexible working conditions and higher accuracy [7–12]. RGB-D sensors diminished the need for multiple sensors by providing both color and depth images of 640×480 size at 30 frames per second.

The problem of human detection is defined as the detection of local data regions containing a human [13, 14]. The data considered may be RGB, point cloud or RGB-D [15]. In applications where the human to be detected covers a smaller portion of the data and is thus visible only at low resolution, this problem is commonly referred to as pedestrian detection [16, 17]. When RGB data is used, human detection is generally accomplished via an exhaustive or selective search over the image. The proposed approaches vary depending on the descriptors and classifiers that are used in determining whether a given candidate region contains a human or not. A variety of different features such as shape, appearance or motion can be considered while building a specific descriptor. The two popular descriptors are the histogram of oriented gradients ('HOG') that use shape features [18] and Haar wavelets [19]. The classifiers that are used along with the descriptors are mostly support vector machines ('SVM') or cas-

cade classification model. If the utilized data is point cloud, human detection depends on segmentation based on clustering and using human-related features for candidate selection [20, 21]. However, as the total number of candidate regions to process in a search may be large, the computation is constrained to reduce the number of locations considered. In addition, non-maximal suppression is applied in most cases in which human detection provides several overlapping regions.

In conjunction, human tracking is defined as the continuous determination of local regions that contain people in an image stream - starting from a bounding box containing the tracked human [13, 14]. Hence, differing from human detection which happens in a single instance, the position change of the human over time in the time-varying data is taken into account. In some cases where the robot is stationary, background subtraction can be applied and detection results may be improved [22]. RGB-D sensor based methods can track certain parts of the body by identifying the human skeleton in 20 joints, but these methods perform best if both environment and robot are stationary [23, 24]. In the scenarios with sensors on the move, most tracking methods do not perform well since the movement adds significant noise to depth perception.

In general, the proposed methods pick one of the detection approaches to implement human tracking algorithms and estimate the relative position of the human target. One of the most popular approaches is to detect legs by using a laser sensor and follow the legs with tracking algorithms. There has been also some work in which a person is tracked by detecting face [25], head-upper torso [26,27] or whole body [11]. As said before, these options differ in the used sensors and their placements on the robot. Usually, the tracking is achieved through employing filters such as particle filter, correlation filter or Kalman filter. As such, the robustness of results under conditions of severe occlusion, target missing or redetection is improved through estimations.

In recent years, with the development of convolutional neural networks ('CNN'), an alternative for special design descriptors and classifiers has become popular. Unlike the previous approaches, the network structure is designed to detect windows that are worth evaluating at first instead of using different sized sliding windows over the

image. One of the most advanced models of the convolutional neural network approach is YOLO ('You Only Look Once') [28]. Even with the first impressions, one can say that YOLO method is much more advanced than the HOG+SVM method. In addition to people, this method can detect 80 different objects such as animals, cars, traffic lights concurrently.

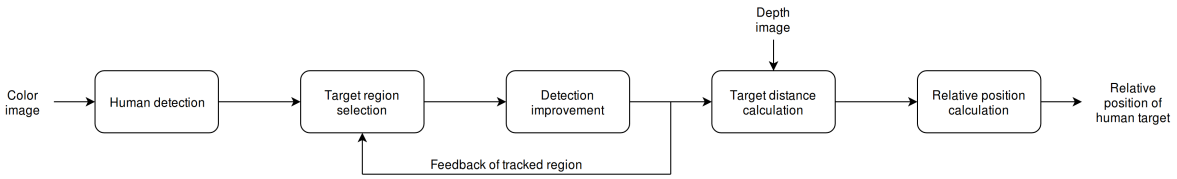


Figure 2.1. Flowchart of the human tracking approach

2.3. Overall Approach

The proposed method consists of five stages as shown in Figure 2.1: human detection, target region selection, detection improvement, target distance calculation, and relative position derivation. For human detection, one of the existing methods (HOG+SVM or YOLO, which will be introduced in the following sections) is used. These methods have different advantages in terms of the required computing power or performance. At the same time, when more than one person is detected, it is ensured that the tracked target is not lost. In cases where detection may be inaccurate, a local area containing human target is estimated using a Kalman filter. Once the robot determines the position of the human on the image, the overall distance between the human target and the robot is calculated using the depth image. Then, the relative position of the human according to the robot is obtained.

2.4. Human Detection

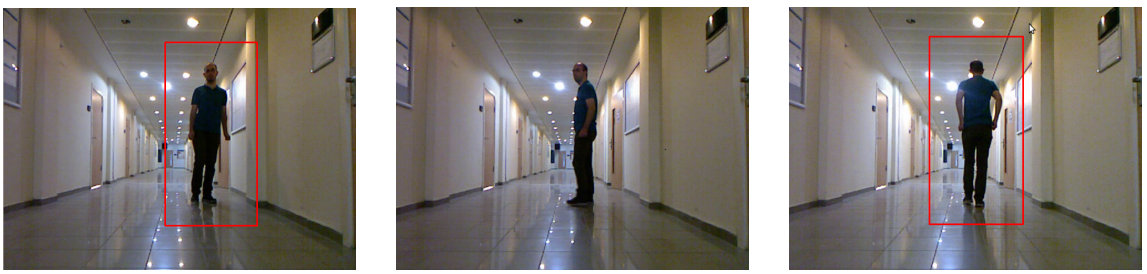
The first step is the human detection using the $N_1 \times N_2$ color image obtained from the RGB-D sensor. Horizontal field of view of this sensor is defined as $\pm\theta_M$. For Kinect sensors, the angle is $\theta_M \approx 30^\circ$ and obtained image size is 640×480. When the robot starts working, it is assumed that there is only one person within the field of

view. Otherwise, it is necessary to define which person to follow. Here, two different methods described in the introduction can be used separately.

- HOG descriptor [18] and linear SVM [18]
- CNN based YOLO method [28]

2.4.1. HOG + SVM method

The HOG descriptor is evaluated by examining the gradient direction and size distribution within any rectangular image window. As the target region can vary in size depending on the proximity of the respective human, the search is conducted with varying scale. As the scale resolution increases, the accuracy of the results and the required processing power both increase. Another factor that increases the required processing power is the size of the test windows where gradient direction and size distributions are observed. Smaller test windows cause the method to run with slower frame rates. Nevertheless, pedestrian detection algorithms working with HOG based human descriptors can operate in real-time without the need for increased computational resources.



(a) Front profile (b) Side profile (c) Back profile

Figure 2.2. Human detection with HOG+SVM method at different profiles

The reliability of detection depends on the human stances used in learning. As this method is typically developed for pedestrian detection, it may not be able to identify a person standing sideways or raising his/her arms. In these cases, human detection will fail. For example, human detection results with HOG method with the front, side, and back profiles are as shown in Figure 2.2. As expected; since the front

and back profiles are highly similar to the defined shape, the detection is successful, whereas in the side profiles the method does not detect the human on the image. In addition, the complexity of the image background or the noise on the gradients in the boundary zones due to illumination may cause the human detection method to produce incorrect or incomplete results.

2.4.2. YOLO method

YOLO method is an alternative CNN-based method for human detection. Its most recent version is YOLO v3, and this version will be used throughout this study. It uses a 106-layer convolutional neural network. The network is trained to learn 80 common objects including human beings in a supervised manner. In detection mode; the color image, which is reduced to a format with equal width and length, is divided into rectangular pieces with the dimensions $S \times S$. By creating pieces with different sizes, the detection is done in three different scales. The probability of including each of the 80 objects for each piece is calculated. Pieces with a probability of containing an object above a certain ratio are combined so that local areas containing each object are obtained. The fact that the convolutional neural network generates these probabilities in one stage makes this method much faster than other deep learning methods.

Differing from HOG+SVM method, this method is more robust to cluttered background, bad illumination or varying human postures. A sample detection is as shown in Figure 2.3. However, the increase in performance also comes with an increase in the required computational resources. Thus, for real-time applications, it requires a high-performance graphical processing unit.

2.5. Multiple Region Scenarios

After obtaining local regions as the result of human detection, a local region needs to be selected as the region that contains the tracked human target. This is necessary since there may be more than one region that is hypothesized as containing the human target. This selection is done in two stages: First, redundant regions that stem from

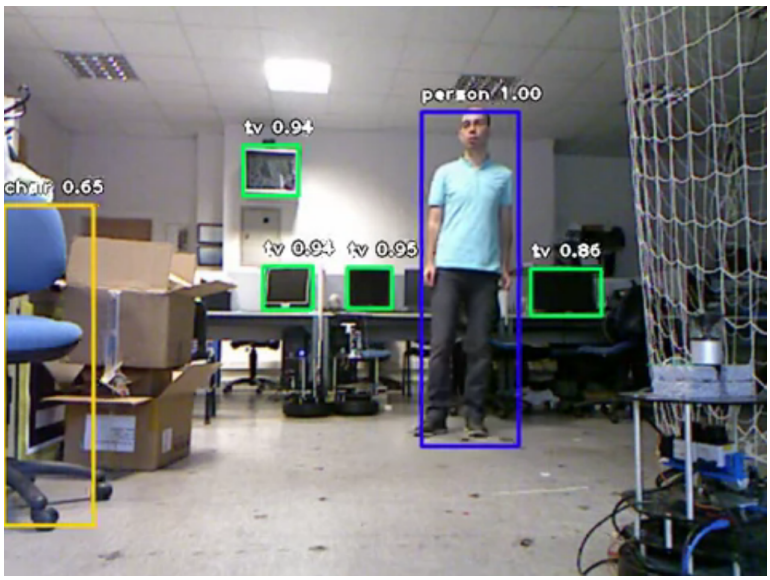


Figure 2.3. Human detection with YOLO method in a cluttered background

the inaccuracy of the human detection technique should be eliminated. Second, one of the regions should be selected according to the regional proximity to the target region of the previous frame.

2.5.1. Suppression of Overlapping Areas

In some cases, human detection techniques provide more than one local region for a human in the field of view. Such problems are often seen in methods that use shape-based features. Since the shadow created as a result of the illumination can create a human-like figure, there may be more than one region for a human in the process output. A similar situation can occur when inadequate parameters are set in order to increase the operating speed of the system while running human detection algorithms in processors with low processing power. An example of this is shown in Figure 2.4. To alleviate this problem, the intermediate results obtained after human detection should first be checked and if the nested areas are detected, all other regions except the region with the largest area should be suppressed. This operation is called as non-maximal suppression ('NMS').

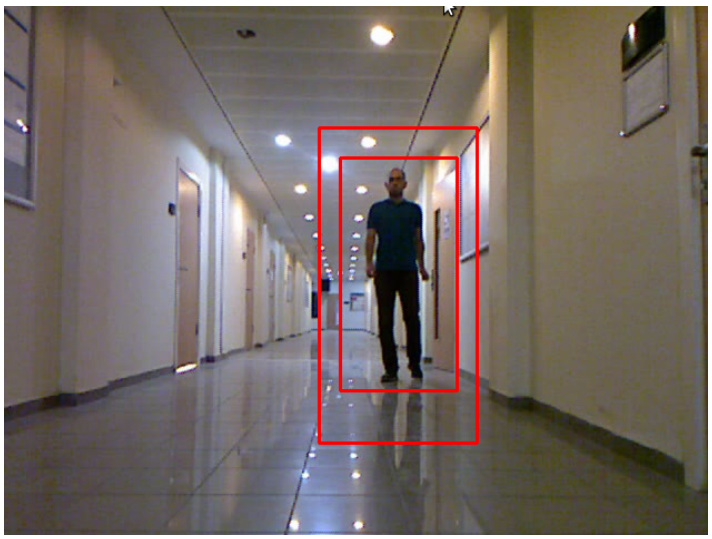


Figure 2.4. A scenario with multiple regions obtained for a human target

2.5.2. Selecting the Target Region

In cases where more than one person is present on the image, it is necessary to choose the target region and ensure that tracking continues. If more than one region passes the non-maximal suppression unaffected, one of these regions is selected as the region containing the target by comparing the positional similarities of these regions to the target region detected in the previous image.

The parameters used to compare the similarity of rectangular regions are the pixel distance between the centers and the pixel area they cover. The set of candidate regions to be detected as the target region is defined as $\mathcal{U}_k = \{u_{kj}, j = 1, .., n\}$ where $u_{kj} = (u_{kj_1}, u_{kj_2})^T$ expresses the center pixels of each of the candidate regions. Each candidate region u_{kj} covers the pixel area $a(u_{kj})$. The following conditions are required for one of these regions to be the successor of $\hat{o}_{k-1}$, which is the target region in the previous image:

$$\|u_{kj} - \hat{o}_{k-1}\| < \tau_d \tag{2.1}$$

$$\frac{|a(u_{kj}) - a(\hat{o}_{k-1})|}{a(\hat{o}_{k-1})} < \tau_a \tag{2.2}$$

Here, $\hat{o}_{k-1}$ is defined as the pixel location of the center of the detected region at frame $k-1$, and $a(\hat{o}_{k-1})$ is the area of this region. τ_d and τ_a are the system variables, which express the maximum acceptable pixel distance and pixel area change ratio respectively. If there is more than one rectangular area which satisfies these two conditions, it is assumed that the candidate region with shorter pixel distance between centers contains the target and the method proceeds to the next stage.

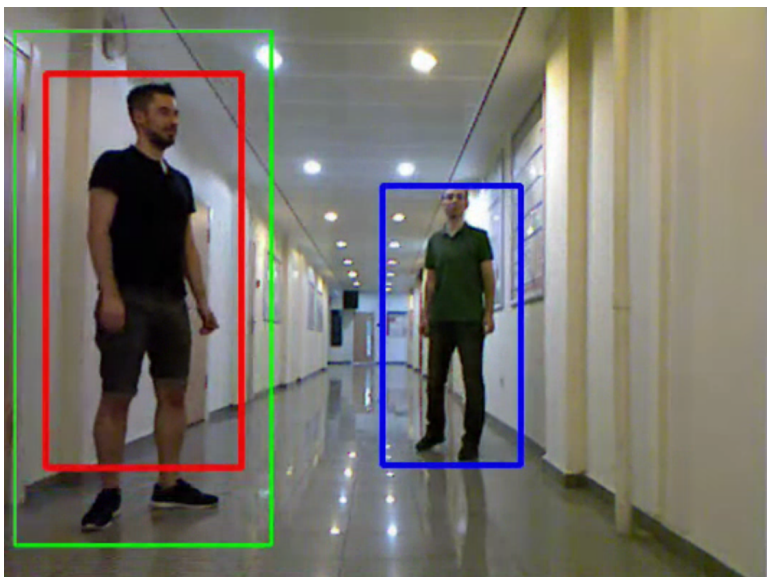


Figure 2.5. Target region selection in a multiple target scenario

A scenario with multiple targets on the image is shown in Figure 2.5. There are two distinct people on the image and the person to the left is the tracked target before the person to the right enters the field of view of the sensor. When two distinct regions are obtained, the region selection algorithm continues giving valid feedback to the target prediction algorithm. Color notation on the image is as follows: The selected region is shown with red, discarded region is shown with blue and predicted target region is shown with green.

2.6. Detection Improvement

The third stage is to improve the local rectangular region that has been obtained. This is because the robot may have failed to detect a human or there may be a faulty detection. For example, with the HOG+SVM method, a human may not be detected if the corresponding image region is near the image boundary or if the human has changed his/her posture to a problematic one such as a side profile. Thus, problems occurring in the detection cause discontinuities in the tracked target position output.

This problem is addressed through using a detection improvement method such as Kalman filtering. As such, output continuity and accuracy can be better maintained. YOLO method does not experience these problems as much as HOG+SVM method, nevertheless, the usage of Kalman filter can still be useful to increase the system output frequency. Instead of detecting images at certain intervals, the real-time operating speed of the method can be increased by filling the intermediate results with Kalman filter outputs.

Kalman filter consists of a prediction and feedback mechanism. System state parameters are generated by estimation using previous data, while a correction is performed by noisy values measured at regular intervals. Update of the status parameters is affected by the process noise that is included in the algorithm as an empirical value. Let $x_k = [o_{k_1}, o_{k_2}, \dot{o}_{k_1}, \dot{o}_{k_2}, w_k, h_k]^T$ denote the state at k. discrete time. Here, o_{k_1} and o_{k_2} are the center coordinates of the local rectangular region, $\dot{o}_{k_1}$ and $\dot{o}_{k_2}$ are the center's speed in pixels per second and w_k and h_k are the width and height of the region. System dynamics are defined by the following equations:

$$x_k = Ax_{k-1} + \mu_{k-1} \tag{2.3}$$

$$y_k = Hx_k + \nu_k \tag{2.4}$$

Here, $y_k = [o'_{k_1}, o'_{k_2}, w'_k, h'_k]^T$ denotes the observation. In particular, o'_{k_1} and o'_{k_2} are the center coordinates of the local rectangular region that is the result of human detection and w'_k and h'_k are its width and height. System matrices A and H are constructed using relationships between x_k and y_k:

$$A = \begin{bmatrix} 1 & 0 & \delta t & 0 & 0 & 0 \\ 0 & 1 & 0 & \delta t & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

The state x_k depends on the previous state x_{k-1} and process noise μ_{k-1}. The measurement y_k is also subject to a measurement noise ν_k. Both process noise μ_k and measurement noise ν_k are generated with zero mean and Gaussian distribution with covariance matrices Σ_x and Σ_y. The respective noise parameters are determined experimentally according to the model that is being predicted and affect filter performance.

The target region resulting from human detection is given to the Kalman filter through feedback, and the deviation of the target region is corrected. Therefore, the estimate $\hat{x}_k$ of x_k is given by:

$$\hat{x}_k = A\hat{x}_{k-1} + K_k(y_k - HA\hat{x}_{k-1})$$

Here, K_k is the Kalman gain matrix, which is recursively calculated. If the target cannot be detected with human detection, the Kalman filter continues to provide region output through estimation. This ensures continuity in the target region output.

2.7. Target Distance Calculation

At this stage, the distance ρ_k of the robot to the tracked human target is determined. This is done by averaging the depth readings from the slightly above center pixel point of the target region and its neighborhood. If depth reading of pixel o is $d(o)$, the depth calculation is as follows:

$$\rho_k = \frac{1}{N_\epsilon(m_k)} \sum_{o \in N_\epsilon(m_k)} d(o) \tag{2.5}$$

In the methods where the entire human body is detected, such as pedestrian detection, in order not to get misleading depth readings, a few pixels above the center point and a few pixel points in that neighborhood are targeted instead of the center point of the target area. Here, $m_k = [\hat{o}_{k_1}, \hat{o}_{k_2} - \hat{h}_k/8]^T$ refers to the coordinates of the slightly above center pixel point - considering the target region. Therefore, the use of pixels which can give incorrect depth is avoided. The requirement for this method to function properly is that the selected pixels on the target region must be positioned on the human target. Higher accuracy of human detection will increase the accuracy of the distance calculation results. The ϵ variable determines the extent of averaging.

The reliability of the target region is checked by ensuring that consecutive distance calculations are in agreement - namely

$$\frac{|\rho_k - \rho_{k-1}|}{\rho_{k-1}} \leq \tau_\rho \tag{2.6}$$

As such, in cases of prolonged faulty human detection, the accuracy of target regions will decrease even if Kalman filtering is applied. With such a check, the robot is able to detect such cases and thus will not proceed to the next stage.

2.8. Relative Position Computation

The next step is to determine the relative position of the human. For this, XZ plane in the robocentric coordinate system is considered as shown in Figure 2.6. The distance ρ'_k to the human is approximated as $\rho'_k \cong \rho_k$. Indeed, this will hold if the RGB-D sensor on the robot is positioned at around a height that is half the average human height. In this case, their difference is less than 1%. The relative horizontal distance X_k and vertical distance Z_k can then be approximated as:

$$X_k \cong \rho'_k \sin\left(\frac{\hat{o}_{k_1} - 0.5N_1}{0.5N_1}\theta_M\right) \tag{2.7}$$

$$Z_k \cong \sqrt{\rho'^2_k - X^2_k} \tag{2.8}$$

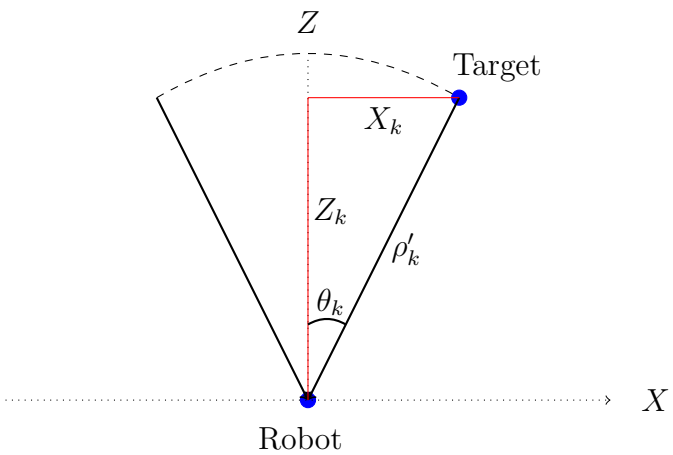


Figure 2.6. Placement of human target on the robocentric coordinate plane

The computation of X_k is realized by assuming every degree change in θ_k results in the same amount of change in length of X_k, which is not true indeed. However, when comparing true positions and resulting positions, it is observed that the error rate is acceptable. With increasing Z_k, the error rate is expected to increase; but in the effective sensor range, position tracking suffers a reasonable amount of error margin.

2.9. Experimental Results

This section presents experimental results that are obtained using a mobile robot endowed with a Kinect sensor. The sensor is positioned at 75 cm above the ground. The robot used in experiments is shown in Figure 2.7. As this height corresponds to half the height of an average grown-up person, the approximation $\rho'_k \cong \rho_k$ holds. With the effective operating distance of the Kinect sensor being between 0.8 and 4 meters, the incoming images contain the full body only if human distance is at least 2 meters. As such, when using HOG+SVM method for human detection, it is possible to track the human only if the distance is between 2 and 4 meters. In contrast, with the YOLO method, human detection is possible in the full active range of the sensor.

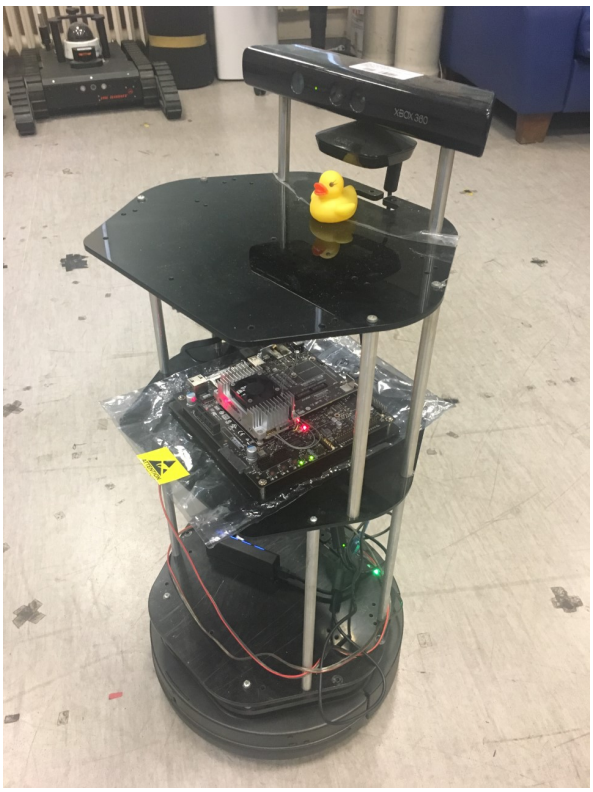


Figure 2.7. The robot used in experiments

In target region selection, the parameter settings are as follows: $\tau_d = 120$, $\tau_a = 0.2$. These values were set experimentally as to ensure the continuity of reliable tracking. For the Kalman filter, process and measurement noise covariance matrices

were experimentally determined as Σ_x and Σ_y:

$$\Sigma_x = \begin{bmatrix} 0.01 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.01 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.01 \end{bmatrix} \quad \Sigma_y = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 \end{bmatrix}$$

In target distance calculation, $\epsilon = 1$ so that averaging is done over the neighborhood that consists of the eight adjacent pixels. Also, $\tau_\rho = 0.2$ is used for eliminating incorrect distance results.

We first observe performance in two alternative background scenarios varying in the environmental clutter - namely simple and complex [29]. It is observed that human detection using HOG+SVM is not as robust as YOLO in the cluttered background. Nevertheless, the detection improvement stage alleviates this problem to a certain extent. However, if failure to detect the human is prolonged, then local rectangular areas cannot be accurately determined and thus the robot cannot track the human.

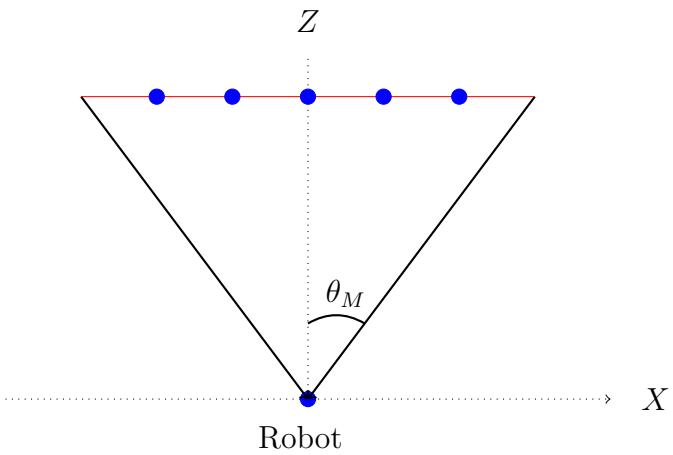


Figure 2.8. Movement of human target in the experiment

The proposed approach is evaluated in a statistical manner considering a variety of different scenarios as shown in Figure 2.8 is applied. In each scenario, the human

is walking on a line at a vertically constant distance Z with respect to the robot. We consider three different vertical distance values $Z^* = \{2, 3, 4\}$ as human detection since the operation range of HOG+SVM is within this range with the positioning of the RGB-D sensor. For each vertical distance, we then consider four different horizontal distance values $X^* = \{0., 0.33, 0.66, 1\}$ in both directions. We compute mean errors $E_X = |X - X^*|$ and $E_Z = |Z - Z^*|$ and their standard deviations σ_X and σ_Z respectively.

When using HOG+SVM method for human detection, the results are as presented in in Table 2.1. The results with using YOLO v3 method for human detection are given in Table 2.2. It should be noted that for some (Z^*,X^*) pairs such as $(2, 1)$, no results are provided since human position falls outside the field of view of the sensor.

Table 2.1. Human tracking results based on HOG+SVM method

Z^* (m)	2			3				4		
X^* (m)	0	0.33	0.66	0	0.33	0.66	1	0	0.33	0.66
E_X (mm)	28.72	42.79	79.36	19.04	36.72	82.31	96.22	18.93	52.28	87.42
σ_X (mm)	10.22	14.68	13.87	7.36	12.54	13.97	16.78	6.26	13.92	11.77
E_Z (mm)	36.56	49.71	82.28	23.16	39.85	86.74	98.28	32.14	62.95	99.72
σ_Z (mm)	8.58	9.27	9.83	5.29	7.62	11.71	10.94	6.32	8.19	9.73

Table 2.2. Human tracking results based on YOLO method

Z^* (m)	2			3				4		
X^* (m)	0	0.33	0.66	0	0.33	0.66	1	0	0.33	0.66
E_X (mm)	10.76	36.82	73.92	13.26	26.42	75.63	92.34	17.93	41.05	79.26
σ_X (mm)	2.25	4.93	6.01	1.93	6.28	5.67	4.84	4.18	5.62	7.23
E_Z (mm)	15.27	42.75	82.19	8.52	36.81	81.32	92.63	21.03	48.95	89.25
σ_Z (mm)	3.24	6.72	5.13	4.86	6.04	9.28	7.22	4.06	8.21	8.07

A few remarks regarding the comparison of the tracking results based on these two methods are as follows: By comparing mean errors, we observe that YOLO method has better accuracy. That means YOLO method can provide local rectangular regions that have better coverage on the human to be tracked. The comparison of standard deviations has a similar ending, with YOLO method providing lower values. These results may be explained with the continuity of Kalman feedback when using YOLO method, whereas HOG+SVM method fails to provide consecutive detection results in cases like human stance change.

Experimental results show that both human detection methods enable the calculation of horizontal and vertical distances using depth information with a small error rate. Assuming that the position errors up to 15 cm are acceptable in the pursuit of a target, the result of this experiment shows that the position tracking of the target person can be carried out in a healthy manner. In other words, a person follower robot can use these position values to follow a human target behind a safe distance.

The processing power of the robot is altered depending on the human detection method used. With HOG+SVM based human detection, the robot is provided with a standard Intel Core i7-4700HQ processor. As such, human detection takes around 0.1 seconds. On the other hand, if the robot is to employ the YOLO method for human detection, its processor is changed to a Nvidia Jetson TX2. Furthermore, RGB-D images are resized to 416×416 as YOLO v3 network has been trained with images at that size. In addition, YOLO rectangular region size parameters are set as 32×32, 16×16 and 8×8. With this setup, human detection takes around 0.285 seconds. It is reported that with a more powerful processor, this may go down to 0.03 seconds [28]. YOLO v3 has also a version called 'Tiny YOLO v3', which is less computationally demanding. In this case, human detection takes around 0.05 seconds, however, the reliability of human detection is lessened as expected.

3. FOLLOWING THE HUMAN TARGET

3.1. Introduction

Once the target human is successfully tracked and his/her relative position is obtained, the next thing to do is ensure tracking even if the human is walking. In that case, the human presence within the robot's field of view does not necessarily hold any longer and the robot needs to start moving as well as to adjust its position and heading accordingly. The main challenge here is that no prior information regarding the traversed areas is assumed to be known. As such, while following, the robot should also simultaneously detect all obstacles encountered and navigate in such as to have no collisions. The main source of environmental sensing is the visual data received from the RGB-D sensor. An important note here is the following: Since RGB-D sensors act as a time-of-flight ('ToF') sensor that employs IR laser structured patterns, their depth perception weakens under direct sunlight. Therefore, the human following robot should be used indoors, or at least the user should wait for cloudy weather if the robot is to be used outdoors. The main goals of this chapter are the following:

- The robot needs to follow the tracked human target while maintaining a safe distance.
- The robot should avoid collisions with any obstacles encountered along the way - regardless of these obstacles being static or on the move. Therefore, using predetermined routes is not an option.
- The robot should move towards the target by following the shortest path. There is a trade-off between safe traversing and the shortest movement, and the robot should pick an optimal solution between these extremities.

The remainder of this chapter is as follows: First, related work is summarized. Then, we present our approach to this problem and discuss our experimental resulting in regards to human following.

3.2. Related Literature

The problem of human following has proven to be a tougher challenge compared to human tracking. This is attributed to the fact that both the human and the robot are moving. As such, determining the topological relation between them becomes harder due to increased noise in depth perception. Nevertheless, this approach is used in many applications that require following a human target [30, 31]. The applications vary in the way they follow the human. In most cases, the robot follows its target from behind a safe distance [32]. However, recent research showed that side-by-side following of the target is also possible [10]. Earlier research on this topic is concentrated on indoor implementations, but executing human following task outdoors is also possible via selecting appropriate sensors [33].

In general, the most convenient way to generate a path that follows the footprints of the human target is the use of a feedback controller proportional to the distance between the human and the robot [8]. Of course, achieving smooth movement on the robot is crucial if the robots are to be used in crowded environments in our daily lives. Basically, the movement and orientation change speed should be close to that of the followed human and reactions of the robot in any scenario like avoiding obstacles should be human-like. There has been considerable progress in this area, but it cannot be said that a fully durable and reliable system has been built yet [34].

In order to reliably follow the human target, the robot needs to gain the capability of moving towards a tracked position in an unknown environment without crashing any obstacles along the way. The work on autonomous navigation can be divided into two categories, namely map-based navigation and mapless navigation [35]. If there is a map with static obstacles, a predetermined route can be generated and employed. In some applications, the robot is used for exploration at first, and then the navigation is achieved by making use of the newly generated map [36–38]. By assuming dynamic obstacles, generating place maps become obsolete. To traverse safely through an environment with dynamic obstacles, path planning should be reactive.

Making the navigation reactive is done via considering a working radius. Instead of working on the complete map (if there is one), the path planning module should consider a smaller area at first and move toward the target as close as this area permits. Then, the process should continue until the goal is reached. This may be called as 'divide and conquer' approach and many applications that consider reactive path planning use this approach. An example might be the work in [2]. By reducing this working radius, obstacle avoidance will be better, but it will bring an extra burden on the processing unit.

3.3. Reactive Navigation

We formulate human following as a reactive navigation problem. Our approach is based on previous work on navigation with topological maps. In that problem, there is a planned sequence of nodes leading to a goal node as defined by their relative positions and the robot needs to travel through each node in the sequence in order to eventually reach a goal. With reactive navigation, the robot keeps track of obstacles as obtained from the laser data and then moves through a series of way points that are generated dynamically in order to reach each node until the final destination [2]. Here, this approach is adapted as follows:

- Depth data is not from laser, but rather from Kinect sensor. Thus, depth images obtained from the RGB-D sensor are converted to laser scan form. This is based on finding smallest value in each column of the depth image and converting it to polar coordinates [39].
- A sequence of goal nodes is now replaced by a single goal node $g_k = \begin{bmatrix} X_k & Z_k \end{bmatrix}$ corresponding to the relative position of the human guide that is also possibly time-varying. As the human moves, the robot updates his/her relative position g_k. In parallel, the associated heading θ_k changes.

For completeness, reactive navigation approach with these modifications is summarized as follows:

Consider a differential wheel robot where the radius of the robot is ρ. The state of the robot is defined with its position and heading such as $x(t) = [c(t), \alpha(t)]^T$ with position $c(t) \in R^2$ and heading $\alpha(t) \in S^1$, where base point space is defined in $X = R^2 \times S^1$. The reactive controller is parameterized by i) human target $g_k \in \mathbb{R}^2$ with the associated heading θ_k; and ii) the so-far encountered obstacles $\mathcal{O}$. It is defined as follows:

$$u = \begin{bmatrix} v_{max} \min(1, |c - g_k|) \cos(\alpha_\nu - \alpha) \\ \omega_{max} \sin(\alpha_\nu - \theta_k) \end{bmatrix} \tag{3.1}$$

where

$$\alpha_\nu = -\nabla_c \varphi_{g_k, \alpha_k, \mathcal{O}}(x)$$

Here, v_{max} ve ω_{max} are linear and angular speed limits. The linear speed of the robot is also scaled according to the distance to the target location. As the magnitude of difference $\alpha_\nu - \theta_k$ becomes smaller, the robot will have only linear velocity. Alternatively, as it reaches 90°, there will be only rotational motion.

The function $\varphi_{g_k, \theta_k, \mathcal{O}} : \mathcal{X} \to R^{\geq 0}$ is an analytic function:

$$\varphi_{g_k, \theta_k, \mathcal{O}}(x) = \sigma_d \circ \sigma \circ \hat{\varphi}_{g_k, \theta_k, \mathcal{O}}(x) \tag{3.2}$$

The function $\sigma : [0, \infty) \to [0, 1)$ is defined as $\sigma(x) = \frac{x}{1+x}$ and $\sigma_d : [0, 1) \to [0, 1)$ is defined as $\sigma_d(x) = x^{1/k}$. Here, the function $\hat{\varphi}_{h,\theta,\mathcal{O}}$ encodes the human target g_k, arrival heading α_k and so-far detected obstacles $\mathcal{O}$. It is constructed as the ratio of two terms:

$$\hat{\varphi}_{g_k, \theta, \mathcal{O}}(x) = \frac{\gamma_{g_k, \theta_k}(x)}{\beta_{\mathcal{O}}(x)} \tag{3.3}$$

Here, $k \in \mathbb{Z}^+$ is the relative weighting parameter of the numerator term with respect to the denominator term. The numerator term $\gamma_{h,\theta} : \mathcal{X} \to \mathbb{R}$ encodes the distance to

the human h as well as deviation from the goal heading θ:

$$\gamma_{g_k,\theta_k}(x) = (c - g_k)^T (c - g_k) \left(\eta_1 + \eta_2 \left(atan2\left(e_2^T (g_k - c), e_1^T (h - c)\right) - \theta_k\right)\right) \quad (3.4)$$

The parameters $\eta_1, \eta_2 \in \mathbb{R}$ weigh the distances to goal position and heading. Here, both are taken to be equal to 1.

The denominator term $\beta_{\mathcal{O}}(x)$ encodes the distance to all the so-far detected obstacles:

$$\beta_{\mathcal{O}}(x) = \prod_{o \in \mathcal{O}} \left(|c - o|^2 - (\rho + \rho_o)^2\right) \quad (3.5)$$

where ρ_o is the radius of obstacle o. Assuming a working radius of r_w, the robot detects obstacles closer than r_w using the depth data and adds to the set O_w which contains obstacles inside the working radius. The obstacles obtained from the depth values are first clustered using a connected components algorithm, then each obstacle is represented by a set of disk with radius equal to ρ. These obstacles are then added to the obstacle memory $\mathcal{O}$ in which they are retained for a fixed period of time. As the obstacles change when the robot navigates or over time, $\beta_{\mathcal{O}}$ changes accordingly.

The robot continually checks the target and associated heading as it is navigating and updates it as necessary until it reaches its goal. Navigation continues until the robot approaches within the τ_c vicinity of the human target while its heading is also withing the τ_α vicinity of the goal heading. *i)* $|c - g_k| < \tau_c$ and *ii)* $|\alpha - \theta_k| < \tau_\alpha$. The parameters τ_c and τ_α denote proximity thresholds for position and heading respectively.

3.4. Experimental Results

In order to evaluate the performance of human following robot, we have conducted experiments in different settings varying in clutter. Here, we summarize the results from these experiments. The robot has an differential wheel base with radius $\rho = 0.2$ m. The

maximum linear and angular velocity values v_{max} and w_{max} are set to 0.5 m/s and 1 rad/s respectively. The distance proximity threshold τ_c is set to 2 m so that the robot will follow the human target from this distance. The working radius r_w is set to 4 m, which is the maximum active range of the Kinect sensor.

The first experiment is conducted at a corridor with 3 m width. This setting is a relatively uncluttered setting. Here, the challenge is due to people passing while the robot is trying to follow the human guide or people opening the doors of rooms on this corridor. In order to avoid such passing people, the robot has to get closer to one of the walls. An instant from the experiment is shown in Figure 3.1. At this instant, the robot is following its target while moving away from some obstacles such as the door and the wall. The next experiments is done in a large room with medium clutter. The tests in the cluttered background aim to observe the effect of clutter in human following, The reactions that are expected from the robot are as follows:

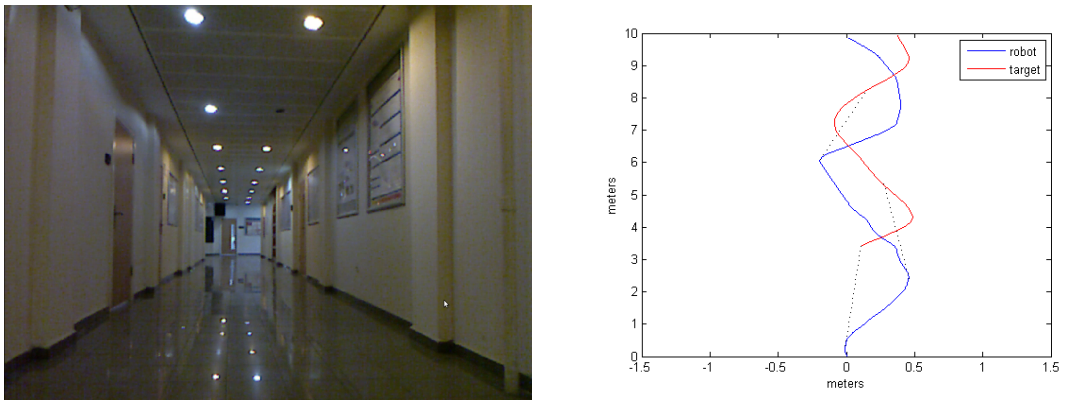
(i) Correcting its position and orientation so that the human target will stay inside the field of view of the RGB-D sensor
(ii) Avoiding any collisions in both static and dynamic obstacle scenarios

Human following performance in both uncluttered and cluttered background is shown in Figure 3.2 and Figure 3.3 respectively. In Figure 3.2(b) and Figure 3.3(b), robot positions are calculated by robot odometry and target movement is predicted from a series of relative positions provided by the human tracking algorithm. On the figures, dotted lines are added to denote the synchronization of positions of the human target and the robot. Observations on multiple experiments show that both goals are reached in most cases. Since the line of sight with the distance of 2 m must be preserved, some actions cannot be done such as entering through a door. Success of the human following method is affected by the used human detection technique and field of view limitations of the RGB-D sensor.

The first factor is the tracking performance of the human detection technique. In Chapter 2, two distinct techniques are used in human tracking, namely HOG + SVM

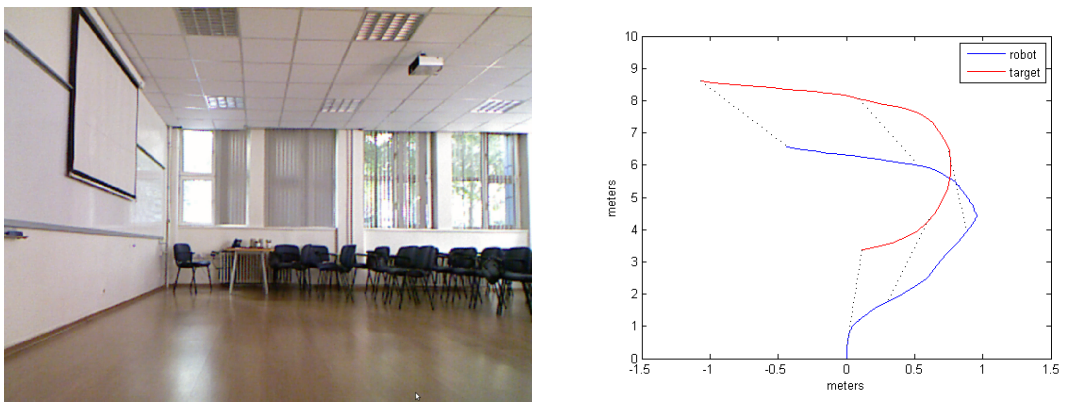
Figure 3.1. An instant in human following in a corridor

and YOLO. Even if the missing detections are filled with the predictions of a Kalman filter, there is still a need for consistency in giving feedback to the filter. The accuracy of the relative position obtained depends on the structure of the scene, the number of people in the scene, and the speed of the tracked human target. In general, the negative effects that stem from these parameters become more substantial when the HOG + SVM method is utilized. The HOG + SVM method is not as successful as the YOLO method in different postures or backgrounds; which results in obtaining distance measurements that are mostly completed with Kalman filter outputs. On the other hand, the YOLO method performs well on the image boundaries as shown in Figure 3.4. The detection by observing a half body gives a significant boost to the reliability of human following. However, the detection by observing a hand does not provide the same effect since the derived relative position from that local region does not give a valid target. At this stage of the study, YOLO method is utilized as the human detector since it enhances human following performance.



(a) Uncluttered background (b) Human following action

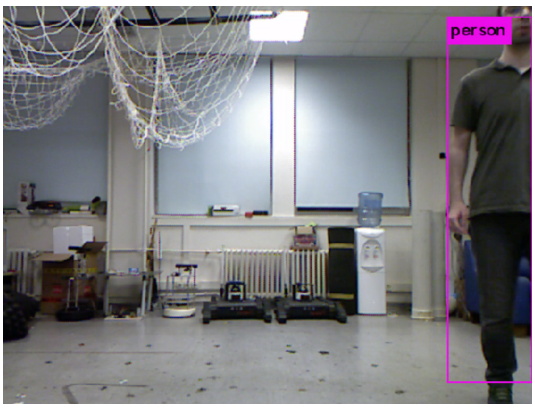
Figure 3.2. Human following performance in uncluttered background



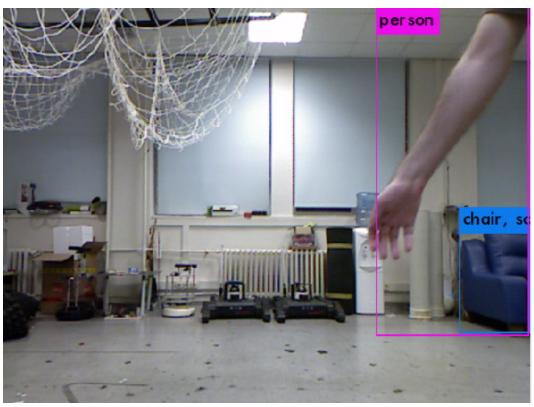
(a) Cluttered background (b) Human following action

Figure 3.3. Human following performance in cluttered background

The second factor is the use of an RGB-D sensor instead of a laser scanner. The main difference between the two sensors is the area coverage. The RGB-D sensor in use has a field of view of 57.5 x 43.5. Considering the laser scanner can scan the whole area around the robot, the observed area decreases to almost one sixth of the surrounding area. Due to this area blindness of the robot, refreshing the obstacle positions inside the working radius can only be done in the observed area. This creates a disturbance in the reactive navigation approach and it becomes increasingly possible for the robot to be stuck in a position oriented towards a wall.



(a) Detection by observing half body



(b) Detection by observing a hand

Figure 3.4. Detection effectiveness of YOLO at image boundaries

4. HUMAN GUIDED PLACE LEARNING

4.1. Introduction

The third and final stage is to couple spatial reasoning while the robot is either visually tracking or following the human guide. Spatial reasoning is based on the concept of a place that refers to an area with spatial extent as defined by its appearances. The main goals of this chapter are as follows:

- The robot needs to detect new places while it passes through the environment.
- The robot should build a long-term place memory that stores the knowledge of learned places. After detecting a new place, this place should either be learned as a new place or recognized as a revisited place.

Here, we use the previously proposed topological spatial cognition ('TSC') model introduced in [3]. We compare performance results obtained with those of teleoperation.

4.2. Related Literature

In real-world applications, the main deficiency of robots is the lack of understanding of the surrounding environment. Robots need to observe their environment and learn or recognize the place it is in just like humans do. Therefore, the ability to use human-like symbolic representations like rooms and corridors is desired [40]. The robot should understand not only the room or any place, but also analyze notable objects and use their topological relations to define the place. In order to construct such base of knowledge, unifying perceptual features and actions from object recognition and categorization is suggested [41].

Semantic mapping on mobile robots generally utilize topological maps, which are graph representations of the explored environment [42]. Each node on the map corresponds to a different place with distinct perceptual features. The usage of a

topological map resembles the human sense as well since human-like mapping is thought as a connected node network in which each node refers to rooms or hallways.

The learning process on robots is mostly aided with human-robot interaction [43]. In order for the robots to learn specific objects in a room, specific visual codes like pointing with a finger are used [44]. Interestingly, to the best of our knowledge, there is no learning and recognition of places and objects via human following. We attribute this primarily to the fact that spatial cognition is known to be a complex problem. Our view is that the complexity may be partially overcome by resorting to a human guide.

4.3. TSC model

The TSC model enables a robot to build its knowledge of places and refer to this knowledge in a completely autonomous manner. The model works in conjunction with a place memory that is powered by the operations on the consecutive appearances. No prior information of environment or localization are required. There are two integral parts: place memory and processing modules. In this section, we briefly summarize the relevant aspects. Due to space restriction, interested readers are kindly referred to [3] for details.

Place memory enables the robot to store and retrieve the knowledge of learned places as indexed by the set $\mathcal{P}$ [45]. It is organized hierarchically as defined by a nested sequence of partitions of the set $\mathcal{P}$. The partition at the top level is the whole set $\mathcal{P}$. All inner nodes correspond to particular subclusters while each of terminal nodes corresponds to a distinct place $p \in \mathcal{P}$. Such an organization allows the robot to associate with its learned place knowledge efficiently. Furthermore, the memory is ensured of having both storage and construction efficiency as well as being order-invariant. Its structure evolves in an unsupervised and incremental manner and is viewed as encoding the semantic hierarchy among different places. In particular, places belonging to each cluster can be viewed as sharing certain common attributes. As there are no externally provided labels expressed in natural language, human users can make

such associations after analyzing the memory contents.

There are three relevant capabilities: place detection, recognition, and learning. These get activated when necessary as the robot navigates through a sequence of locations $c_k \in R^2$ and headings $\alpha_k \in S^1$ $k \in \mathcal{K}$ where $\mathcal{K}$ is the index set. In order to minimize perceptual aliasing, at each location c_k, the robot should have full view of its surroundings as much as possible. The appearance from each location c_k is then internally encoded by a d-dimensional descriptor $I(c_k) \in R^d$. The descriptors can be constructed using a variety of local or global features - as long as nearby appearances generate similar descriptors. In this work, we use the bubble space representation due to demonstrated advantages such as nearby appearances generating similar descriptors, encoding both local sensory features and their relative S^2 geometry, incorporating any number of observations and being rotationally invariant as discussed in the related work. In order not to repeat this presentation, the interested reader is kindly referred to [4] for details including its comparison with other representation methods. However, let it be noted that other descriptors could be used - as long as nearby appearances generate similar descriptors.

While the robot moves through the base points with position and heading such as $x(t) = [c(t), \alpha(t)]^T$ with position $c(t) \in R^2$ and heading $\alpha(t) \in S^1$, consecutive image frames are being assessed in terms of coherency and informativeness. The information that each appearance holds is deducted by the use of bubble descriptors, which are holistic representations of bubble surfaces. The bubble space approach is based upon a controlled deformation on a hypothetical spherical surface surrounding the robot and the deformation is based on the appearances and the information it holds. Bubble descriptors are constructed using the double Fourier series representation of bubble surfaces. These descriptors are used as a coherency checking tool between consecutive frames. Not every appearance holds valuable information, therefore the informativeness is also needed to be checked for each frame.

The goal of place detection is to partition the index set $\mathcal{K}$ so that appearances belonging to each distinct place are grouped together as $\mathcal{D} \subset \mathcal{K}$. The partitioning

process enables the robot to know where a place starts and ends. It is achieved by the iterative clustering of the index set $\mathcal{K}$ - considering the informativeness, coherency, and plenitude of the associated visual descriptors that are obtained from the incoming sequence of appearances. The clustering is done by identifying maximally coherent descriptor neighborhoods that correspond to distinct places and temporal windows that correspond to in-between transitions. Uninformative descriptors that arise in problematic environmental conditions such as low illumination or viewpoint (robot looking at a large object or being very close to one) are not taken into consideration. Informativeness is checked by comparing the mean and variance of bubble surface deformations with mean and variance thresholds τ_μ and τ_σ and uninformative frames are eliminated from the place representation. A place detection starts after detecting a transition region that is not temporary (such as a human passing through the scene). The maximum size of the temporal window and the incoherency extension threshold are also parameterized in the approach. As such, each detected place is associated with a set of appearances that describe the place. The plenitude check ensures that the number of locations is large enough as to enable the robot to have a sufficient knowledge of the place.

Whenever a place $\mathcal{D}$ is detected, the robot then attempts to recognize it as one of the learned places $p \in \mathcal{P}$ via relating the collected appearances to its place memory - as detailed in [45]. This is based on associating the current appearances (if possible) with those retained in the place memory through traversing down the memory hierarchy. The traversal proceeds downwards level-by-level and decisions are progressively combined to hierarchically refine the final decision. Consider a place memory with N_l levels. At each level l, $l = 1, \ldots, N_l$, a decision regarding one attribute is made by choosing a particular node $N \in S(N^{l-1})$ among children $S(N^{l-1})$ nodes of node N^{l-1} that has been reached at level $l-1$. The decision-making is based of finding the node N with the minimum cost function $g_N(D)$ while ensuring that the minimum cost is below a recognition cost threshold τ_r.

$$N \in \arg \min_{N' \in S(N^{l-1})} g_{N'}(D) \tag{4.1}$$

subject to

$$g_N(D) \leq \tau_r \tag{4.2}$$

This process is repeated either until the condition of Eq. 4.2 is not satisfied or terminal nodes (final level) is reached. In the former case, no decision is made while in the latter, the place is recognized to be the place associated with the terminal node. Due to space restrictions, the interested reader is referred to [45] for details. The recognition threshold τ_r is a designating factor in the tradeoff between precision and recall. As its value is decreased, while precision increases, recall decreases. In case of recognition, the robot simply updates its memory via incorporating the new knowledge appropriately and goes back to the beginning where it waits for new sensory input.

In case of no recognition, it invokes place learning in order to add the detected place D into its place memory. Place learning enables the robot to add the new knowledge to its place memory. Whenever a place is learned, the cardinality of the set $\mathcal{P}$ increased by one. This is achieved using the hierarchical single link clustering method SLINK [45]. The clustering is incremental with both storage and construction efficiency as well as the resulting hierarchy being order-invariant. Thus, it enables the robot to evolve its place memory over time as it detects places, but cannot recognize them. The clustering is done in the appearance space so that each place p is learned based on appearances from a set of $M_p = |D|$ locations associated with the corresponding detected place D. The number M_p of learned locations (while being greater than the plenitude threshold) actually depends on how much the robot moves in this place during place detection. Let these locations be denoted by $c_j(p) \in R^2$. Note that learning is based purely on appearances as encoded by the corresponding descriptors $I(c_j(p))$. Thus, if there are large appearance changes over time or if the robot visits a previously unvisited part of a place (since place detection does not ensure complete coverage), the robot will learn the corresponding areas as a new place.

4.4. Integration with Human Following

When the robot is following a human, the human appears in the incoming images. As such, the human target covers some area on the image and the background data from this area cannot be observed, and it means a possible loss of information. The target presence on each image also negatively affects the coherency checking operation in consecutive appearances and thus, place detection becomes harder. Human target appearances on the image at different distances are given in Figure 4.1. By initial observations, the range of 2-4 m seems the best option for concurrently running human following and place learning algorithms.

(a) 1 m distance

(b) 2 m distance

(c) 3 m distance

(d) 4 m distance

Figure 4.1. Observing the human target at different distances

4.5. Experimental Results

To test the human following based place learning method, there are certain limitations about the testing environment. First, the environment should be indoors since RGB-D sensor is used. Second, the distance between the robot and the tracked person should be over 2 m and the line of sight should be maintained. Hence, room-to-room navigation is not possible with this setup unless a room can be entered through a large door that a robot can enter through moving straightforward or making a wide turn. The experiment is done by completing clockwise laps in the map shown in Figure 4.2. The width of each corridor is 3 m and there are relatively wide areas at the corners of the map. On the right bottom corner, there is a large room that the robot can enter and return to the corridor. Thus, the complete parkour of one lap consists of a large room and a square hallway with wide areas at corners. As for ground truth, the first impressions on the map shows three places, which are the large room, a corner of the map and the corridor that binds each corner. In the experiment, some notable objects are added to one of the corner areas in order to increase appearance-based ground truth to four places.

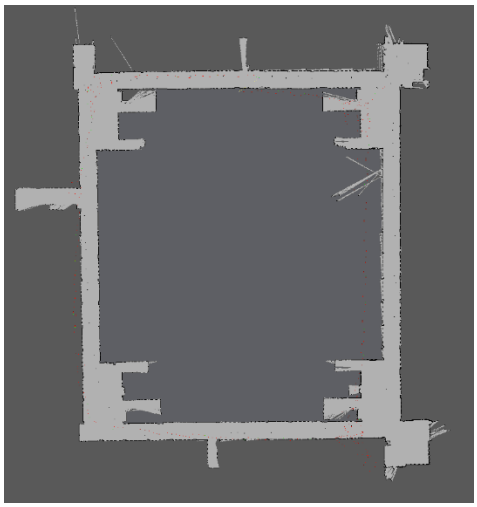


Figure 4.2. The floor plan where the experiments take place

TSC parameters used in the experiment are given in Table 4.1. Note that since there are not enough notable objects on the corridors, coherency thresholds are cali-

brated at a value lower than optimal so that the test can give enough detected places for performance comparison.

Table 4.1. TSC parameters used in experiments

Parameter definition	**Value**
Temporal window extent parameter	5
Incoherency extension threshold	10
Place size threshold	20
Intensity channel coherency threshold	0.3
Hue channel coherency threshold	0.2
Mean threshold for informativeness check	0.2
Variance threshold for informativeness check	0.01
Recognition threshold	1.99

Through the experiment, the robot makes first and second visits in order to assess place learning and recognition performance. Each visit is completed by finishing one lap around the square floor plan. Since the main motivation is to compare results of human following based place learning method and that of teleoperation based method, these first and second visits are repeated for three times. In the first stage of the experiment, teleoperation based navigation is used. In contrast, human following based navigation is used in the next stage. In the final stage, mixed navigation is utilized, which means navigation is accomplished by human following for the first lap, then the last lap is completed using teleoperation based navigation. Number of detected and learned places at each stage of the experiment are given in Table 4.2.

Table 4.2. Number of detected and learned places for each navigation type

	Teleoperation		Human Following		Mixed	
	First Visit	Second Visit	First Visit	Second Visit	First Visit	Second Visit
Detected	12	8	9	5	9	11
Learned	5	1	5	0	5	1

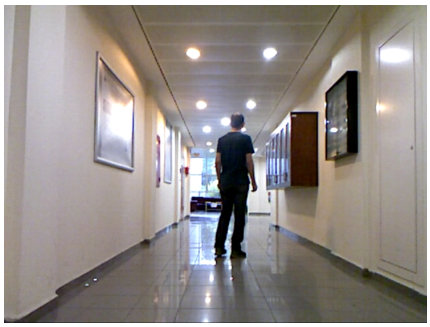
As expected, the teleoperation based method detected more places. However, compared to that of detected places, the numbers of learned places are relatively low. This result stems from the nature of the testing environment which only consists of narrow corridors and room-like corners that almost all images contain the same type and style etc. Plus, the whole area does not contain enough notable objects to make a difference. Completing a lap resembles entering and exiting a room and using the same hallway. Thus, even in the first visit, recognized places are obtained at different locations. Upon entering a corridor, the algorithm gives a recognition of the previous corridor if the informativeness rate of the appearance is high enough. These false positives on recognized places decrease total precision of the system but they should not be recalled as 'false'.

However, this issue is hardly related to human presence on the image. Since all navigation techniques suffer from the same issue, we can still compare the results and find the effects of the navigation method. Since the tracked human is followed from a distance at more than 2 m, the human covers an acceptable portion of the area. On the informativeness check, we can't say the human presence makes a positive or negative impact since the information flow from the covered area is blocked, but the coherency check is negatively affected by the human on the image and its effect grows when the covered area becomes larger.

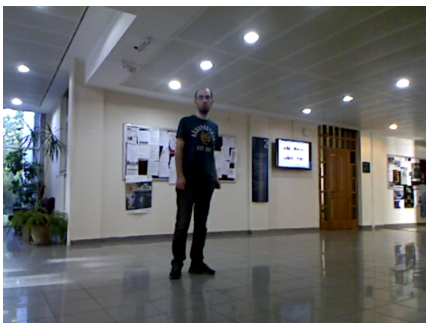
The human target being on the image affects the construction of the place memory as well. Each appearance of the learned place will include the human, thus if the detected place no longer contains the human, recognition will be less likely. Hence, in this case, the human will act like a notable object discovered on the image.

Still, since the results of teleoperation and human following are almost on par, the results can act as proof of the suitability of human following based navigation to work concurrently with place learning algorithms. For human following based place learning, a sample appearance for each detected place is given in Figure 4.3. Comparing to the ground truth, the set of learned places have more elements. The main reason for this is the nature of the algorithm: If the robot revisits a place with a different heading, a

different set of notable objects can be seen and the place might be learned as a new place. An example for this result is two distinct places shown in Figure 4.3(b) and Figure 4.3(e). These appearances are observed at almost the same location, but two different places are learned due to heading angle differences.



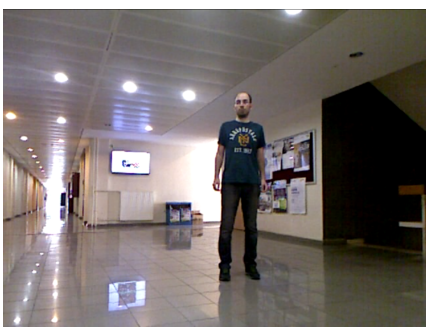
(a) place 1



(b) place 2

(c) place 3

(d) place 4



(e) place 5

Figure 4.3. Sample appearances from each learned place

5. CONCLUSION

In this thesis, we presented a method to enhance the spatial awareness of the robot while the robot is guided by a human guide through an unknown environment. In order to accomplish this task, we integrated three distinct stages of research, which are human tracking, human following, and human-guided place learning. For human tracking; a five-stage method is proposed. While the operations at each stage are not newly introduced, the proposed pipeline is considered as novel. This pipeline can be used along with any human detection technique and enhances its results. To prove this statement, two distinct human detection techniques are used and enhanced tracking position results are obtained. This part of the work can also be seen as a benchmark study of two different human detection techniques.

In addition, human following action is accomplished on a differential wheel robot base and tests are done at places with distinct backgrounds. The human following robot is endowed with an integrated spatial cognition model so that human-like place understanding can be realized on the robot. Experiments on the TSC model shows that, compared to teleoperation based methods, human following based place learning methods can also provide acceptable results.

As the future work, the addition of extra sensors on the robot would be the first improvement. The problems of robot movement and place learning mostly stem from the low range and field of view of the RGB-D sensor being used. This sensor can only observe almost one-sixth of the surrounding area, and this issue deeply affects APF approach and avoidance of dynamic obstacles. Compared to laser scanners, the maximum active range of an RGB-D sensor is also lower, which results in lower perception.

As another improvement, the robot should find a way to recover the line of sight with the human target if lost. For now, our implementation cannot do this job, hence room-to-room navigation is out of range. If the robot loses the line of sight, it may follow the predicted target position using the most convenient path. An addition might

be the recognition of the same human target by comparing textural features. However, this task might be hard to accomplish reliably without a place map.

Another extension of this study might be feeding different, or even multiple data to the TSC model as incoming appearances. If area coverage of each appearance increases, the detection and recognition results might increase in recall and precision.

REFERENCES

1. Mu, X., L. Tan, Y. Tian and C. Wang, "Recent Development of Human-Robot Natural Interaction in Spatial Cognition Tasks", *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, Vol. 1, pp. 446–450, IEEE, 2016.

2. Turksoy, K. and H. I. Bozma, "Topometrik Haritalar Üzerinde Algılayıcı Temelli Tepkin Yöngüdüm (in Turkish)", *Türkiye Robotbilim Konferansı*, 2018.

3. Karaoguz, H. and H. I. Bozma, "An integrated model of autonomous topological spatial cognition", *Autonomous Robots*, Vol. 40, No. 8, pp. 1379–1402, 2016.

4. Erkent, Ö. and H. I. Bozma, "Bubble space and place representation in topological maps", *The International Journal of Robotics Research*, Vol. 32, No. 6, pp. 672–689, 2013.

5. Bellotto, N. and H. Hu, "Multisensor-based human detection and tracking for mobile service robots", *IEEE Trans. on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 39, No. 1, pp. 167–181, 2009.

6. Horiuchi, T., S. Thompson, S. Kagami and Y. Ehara, "Pedestrian tracking from a mobile robot using a laser range finder", *2007 IEEE International Conference on Systems, Man and Cybernetics*, pp. 931–936, IEEE, 2007.

7. Sonoura, T., T. Yoshimi, M. Nishiyama, H. Nakamoto, S. Tokura and N. Matsuhira, "Person following robot with vision-based and sensor fusion tracking algorithm", *Computer vision*, IntechOpen, 2008.

8. Alvarez-Santos, V., X. Pardo, R. Iglesias, A. Canedo-Rodriguez and C. Regueiro, "Feature analysis for human recognition and discrimination: Application to a person-following behaviour in a mobile robot", *Robotics and Autonomous Systems*,

Vol. 60, No. 8, pp. 1021 – 1036, 2012.

9. Fotiadis, E. P., M. Garzón and A. Barrientos, "Human detection from a mobile robot using fusion of laser and vision information.", *Sensors*, Vol. 13, No. 9, pp. 11603–11635, 2013.

10. Ferrer, G., A. G. Zulueta, F. H. Cotarelo and A. Sanfeliu, "Robot social-aware navigation framework to accompany people walking side-by-side", *Autonomous robots*, Vol. 41, No. 4, pp. 775–793, 2017.

11. Wang, M., D. Su, L. Shi, Y. Liu and J. V. Miro, "Real-time 3D human tracking for mobile robots with multisensors", *IEEE Int'l Conf. on Robotics and Automation*, pp. 5081–5087, May 2017.

12. Ali, B., K. F. Iqbal, Y. Ayaz and N. Muhammad, "Human detection and following by a mobile robot using 3D features", *IEEE Int'l Conf. on Mechatronics and Automation*, pp. 1714–1719, 2013.

13. Nguyen, D. T., W. Li and P. O. Ogunbona, "Human detection from images and videos: A survey", *Pattern Recognition*, Vol. 51, pp. 148–175, 2016.

14. Smeulders, A. W. M., D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan and M. Shah, "Visual Tracking: An Experimental Survey", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 36, No. 7, pp. 1442–1468, 2014.

15. Camplani, M., A. Paiement, M. Mirmehdi, D. Damen, S. Hannuna, T. Burghardt and L. Tao, "Multiple human tracking in RGB-depth data: a survey", *IET Computer Vision*, Vol. 11, No. 4, pp. 265–285, 2017.

16. Enzweiler, M. and D. M. Gavrila, "Monocular Pedestrian Detection: Survey and Experiments", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 31, No. 12, pp. 2179–2195, 2009.

17. Dollar, P., C. Wojek, B. Schiele and P. Perona, "Pedestrian Detection: An Evaluation of the State of the Art", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 34, No. 4, pp. 743–761, 2012.

18. Dalal, N. and B. Triggs, "Histograms of oriented gradients for human detection", *Int'l Conf. on Computer Vision & Pattern Recognition*, Vol. 1, pp. 886–893, IEEE Computer Society, 2005.

19. Viola, P., M. Jones *et al.*, "Rapid object detection using a boosted cascade of simple features", *CVPR*, Vol. 1, pp. 511–518, 2001.

20. Zhang, H., C. Reardon and L. E. Parker, "Real-Time Multiple Human Perception With Color-Depth Cameras on a Mobile Robot", *IEEE Trans. on Cybernetics*, Vol. 43, No. 5, pp. 1429–1441, Oct 2013.

21. Liu, H., J. Luo, P. Wu, S. Xie and H. Li, "People detection and tracking using RGB-D cameras for mobile robots", *International Journal of Advanced Robotic Systems*, Vol. 13, No. 5, p. 1729881416657746, 2016.

22. Stauffer, C. and W. E. L. Grimson, "Learning patterns of activity using real-time tracking", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 8, pp. 747–757, 2000.

23. Nergui, M., Y. Yoshida, N. Imamoglu, J. Gonzalez, M. Sekine and W. Yu, "Human motion tracking and recognition using HMM by a mobile robot", *Int'l Journal of Intelligent Unmanned Systems*, Vol. 1, No. 1, pp. 76–92, 2013.

24. Cu, G., A. G. Ang, A. R. Ballesteros and J. C. Rentoy, "Human Following Robot using Kinect Sensor", *Research Congress*, pp. 1–7, 2013.

25. Sidenbladh, H., D. Kragic and H. I. Christensen, "A person following behaviour for a mobile robot", *IEEE Int'l Conf. on Rob. and Aut.*, Vol. 1, pp. 670–675, 1999.

26. Jafari, O. H., D. Mitzel and B. Leibe, "Real-time RGB-D based people detection and tracking for mobile robots and head-worn cameras", *IEEE Int'l Conf. on Robotics and Automation*, pp. 5636–5643, IEEE, 2014.

27. Vo, D. M., L. Jiang and A. Zell, "Real time person detection and tracking by mobile robots using RGB-D images", *IEEE Int'l Conf. on Robotics and Biomimetics*, pp. 689–694, Dec 2014.

28. Redmon, J. and A. Farhadi, "YOLOv3: An Incremental Improvement", *arXiv*, 2018.

29. Song, S. and J. Xiao, "Tracking Revisited Using RGBD Camera: Unified Benchmark and Baselines", *IEEE Int'l Conf. on Computer Vision*, pp. 233–240, Dec 2013.

30. Jung, E. J., J. H. Lee, B. J. Yi, J. Park, S. Yuta and S. T. Noh, "Development of a laser-range-finder-based human tracking and control algorithm for a marathoner service robot", *IEEE/ASME Trans. on Mechatronics*, Vol. 19, No. 6, pp. 1963–1975, 2014.

31. Rawashdeh, N. A., R. M. Haddad, O. A. Jadallah and A. E. To'ma, "A person-following robotic cart controlled via a smartphone application: design and evaluation", *2017 International Conference on Research and Education in Mechatronics (REM)*, pp. 1–5, IEEE, 2017.

32. Hirai, N. and H. Mizoguchi, "Visual tracking of human back and shoulder for person following robot", *Proceedings 2003 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM 2003)*, Vol. 1, pp. 527–532, IEEE, 2003.

33. Kobilarov, M., G. Sukhatme, J. Hyams and P. Batavia, "People tracking and following with mobile robot using an omnidirectional camera and a laser", *Proceedings 2006 IEEE International Conference on Robotics and Automation*, pp. 557–562,

IEEE, 2006.

34. Honig, S. S., T. Oron-Gilad, H. Zaichyk, V. Sarne-Fleischmann, S. Olatunji and Y. Edan, "Toward Socially Aware Person-Following Robots", *IEEE Transactions on Cognitive and Developmental Systems*, Vol. 10, No. 4, pp. 936–954, 2018.

35. Bonin-Font, F., A. Ortiz and G. Oliver, "Visual navigation for mobile robots: A survey", *Journal of intelligent and robotic systems*, Vol. 53, No. 3, pp. 263–296, 2008.

36. Oriolo, G., G. Ulivi and M. Vendittelli, "Real-time map building and navigation for autonomous robots in unknown environments", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 28, No. 3, pp. 316–333, 1998.

37. Kidono, K., J. Miura and Y. Shirai, "Autonomous visual navigation of a mobile robot using a human-guided experience", *Robotics and Autonomous Systems*, Vol. 40, No. 2-3, pp. 121–130, 2002.

38. Thrun, S., "Learning metric-topological maps for indoor mobile robot navigation", *Artificial Intelligence*, Vol. 99, No. 1, pp. 21–71, 1998.

39. Drwiega, M. and J. Jakubiak, "A set of depth sensor processing ROS tools for wheeled mobile robot navigation", *Journal of Automation Mobile Robotics and Intelligent Systems*, Vol. 11, 2017.

40. Randelli, G., T. M. Bonanni, L. Iocchi and D. Nardi, "Knowledge acquisition through human–robot multimodal interaction", *Intelligent Service Robotics*, Vol. 6, No. 1, pp. 19–31, 2013.

41. Lim, G. H., I. H. Suh and H. Suh, "Ontology-based unified robot knowledge for service robots in indoor environments", *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, Vol. 41, No. 3, pp. 492–509, 2010.

42. Charalampous, K., I. Kostavelis and A. Gasteratos, "Recent trends in social aware robot navigation: A survey", *Robotics and Autonomous Systems*, Vol. 93, pp. 85–104, 2017.

43. Rouanet, P., P.-Y. Oudeyer, F. Danieau and D. Filliat, "The impact of human–robot interfaces on the learning of visual objects", *IEEE Transactions on Robotics*, Vol. 29, No. 2, pp. 525–541, 2013.

44. Fischer, K., L. C. Jensen, F. Kirstein, S. Stabinger, Ö. Erkent, D. Shukla and J. Piater, "The effects of social gaze in human-robot collaborative assembly", *International Conference on Social Robotics*, pp. 204–213, Springer, 2015.

45. Erkent, O., H. Karaoguz and H. I. Bozma, "Hierarchically self-organizing visual place memory", *Advanced Robotics*, Vol. 31, No. 16, pp. 865–879, 2017.

46. Quigley, M., K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler and A. Y. Ng, "ROS: an open-source Robot Operating System", *ICRA workshop on open source software*, Vol. 3, p. 5, Kobe, Japan, 2009.

47. Bradski, G. and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*, O'Reilly Media, Inc., 2008.

48. Redmon, J., *Darknet: Open Source Neural Networks in C*, `http://pjreddie.com/darknet/`, 2013, accessed in June 2019.

49. Bjelonic, M., *YOLO ROS: Real-Time Object Detection for ROS*, `https://github.com/leggedrobotics/darknet_ros`, 2016, accessed in June 2019.

APPENDIX A: ROBOT HARDWARE

The human following robot is built on a differential wheel base and the base carries an RGB-D sensor mounted on top. Plus, whole system is operated by a processing unit. Some important specifications of these hardware elements will be explained in this chapter.

A.1. Differential Wheel Base

For robot base, Kobuki Turtlebot is used. This differential wheel base has the following specifications:

- Maximum translational velocity: 70 cm/s
- Maximum rotational velocity: 180 deg/s
- Payload: 5 kg (hard floor)
- Expected Operating Time: 3 hours
- Power Output: 5V/1A, 12V/1.5A, 12V/5A

A.2. RGB-D Sensor

For the RGB-D sensor, Kinect v1 is used. Some important specifications are the following:

- Image size: 640×480
- Image frequency: 30 frames per second
- Effective range: 0.8 - 4.0 m
- Angle of view: 57.5° / 43.5° for horizontal /vertical

A.3. Processing Unit

For operating the whole system, Nvidia Jetson Tx2 is chosen since some of the algorithms need a graphical processor unit to operate in real-time. However, any processor with the same GPU performance rating can be used for the realization of the robot system.

For Nvidia Jetson Tx2, the typical power consumption under load is 7.5W. At maximum performance mode, power consumption may increase up to 15W.

A.4. Powering up the System

The robot base can provide power through the connectors at 12V/1.5A and 12V/5A. 12V/5A connector is used for powering up the RGB-D sensor. In initial tests, 12V/1.5A connector is used for powering up Jetson Tx2. However, system shutdown is observed on multiple occasions due to Jetson Tx2 being sensitive to instant power drop spikes. Therefore, the processing unit is powered up via 14.8 V LiPo batteries during the experiments. In the future work, the reader may use a voltage regulator or a simple capacitor circuit to power up the processing unit via Kobuki power output.

APPENDIX B: SOFTWARE

B.1. Required Software and Libraries

The whole system is realized under ROS framework [46] and OpenCV computer vision library [47] is used for image processing tasks. Prerequisites for running the software are as follows:

(i) Ubuntu 16.04
(ii) ROS Kinetic with catkin build system
(iii) OpenCV 3.3 or higher
(iv) Qt4 or higher
(v) emptydb folder which includes empty database files: detected places.db and knowledge.db under home directory
(vi) visual_filters folder including related filters within the home directory
(vii) specified catkin workspace path that consists of all required nodes

The whole system consists of the following ROS nodes:

- *freenect_launch*: This node is the intermediary between Kinect and the ROS framework. It publishes color and depth images which are subscribed by other nodes. Available at: http://wiki.ros.org/freenect_launch
- *laserscan_kinect*: This node converts the depth image to 2D laser scanner format [39]. Available at: http://wiki.ros.org/laserscan_kinect
- *kobuki_node*: This node is the Kobuki driver. The movement of the robot can be controlled by using this node. Available at: http://wiki.ros.org/kobuki_node
- *person_tracker*: This is the human tracker node that uses HOG+SVM method for human detection. If YOLO method is in use, this node is not needed.
- *darknet_ros*: YOLO method needs Darknet framework [48] to realize the whole network. This node is the ROS implementation of Darknet [49]. Available at: https://github.com/leggedrobotics/darknet_ros

- *yolo_pp*: This node works in conjunction with the *darknet_ros* node and gives human tracking output that uses YOLO method. If HOG+SVM method is in use, this node is not needed.
- *apes_base*: This node is devoted to reactive navigation. It controls the movements of the Kobuki base.
- *place_detection_isl*: This node detects any new place the robot enters. Available at: https://github.com/islboun/place_detection_isl
- *create_bdst_isl*: This node is devoted to place recognition and constructing place memory. Available at: https://github.com/islboun/create_bdst_isl

B.2. Running Software

(i) Open a terminal window and type:
sudo nvpmodel -m 0 — This enables maximum performance on Jetson Tx2.

(ii) On the same terminal window:
roslaunch freenect_launch freenect.launch — This will run *freenect_launch* node.

(iii) On another terminal window:
roslaunch laserscan_kinect laserscan.launch — This will run *laserscan_kinect* node.

(iv) On another terminal window:
roslaunch kobuki_node minimal.launch — This will run *kobuki_node* node.

(v) On another terminal window (if using HOG+SVM for human detection):
rosrun person_tracker person_tracker_node — This will run *person_tracker* node.

(vi) On another terminal window (if using YOLO for human detection):
roslaunch darknet_ros yolo_v3.launch — This will run *darknet_ros* node.

(vii) On another terminal window (if using YOLO for human detection):
rosrun yolo_pp yolo_pp_node — This will run *yolo_pp* node.

(viii) On another terminal window:
rosrun create_bdst_isl create_bdst_isl_node — This will run *create_bdst_isl* node.

(ix) On another terminal window:
roslaunch place_detection_isl jaguar.launch — Using parameters from the launch file, this will run *place_detection_isl* node.

(x) On another terminal window:

rostopic pub /place_detection_isl/nodecontrol std_msgs/Int16 1 — This will publish start command to *place_detection_isl* node.

(xi) On another terminal window:

rosrun apes_base apes_base_node — This will run *apes_base* node.

If all stages are done correctly and in order, the robot will start human tracking and follow the first person on sight. While moving, place detection and spatial memory construction is active and database entries are created within the home directory.