Autonomous Robots in Dynamic Indoor Environments:

Localization and Person-Following

Raghavender Sahdev

A Thesis submitted to the Faculty of Graduate Studies
in partial fulfilment of the requirements
for the degree of

Master of Science

Graduate Program in
Electrical Engineering and Computer Science
York University
Toronto, Ontario

March, 2018

Abstract

Autonomous social robots have many tasks that they need to address such as localization, mapping, navigation, person following, place recognition, etc. In this thesis we focus on two key components required for the navigation of autonomous robots namely, person following behaviour and localization in dynamic human environments. We propose three novel approaches to address these components; two approaches for person following and one for indoor localization. A convolutional neural networks based approach and an Ada-boost based approach are developed for person following. We demonstrate the results by showing the tracking accuracy over time for this behaviour. For the localization task, we propose a novel approach which can act as a wrapper for traditional visual odometry based approaches to improve the localization accuracy in dynamic human environments. We evaluate this approach by showing how the performance varies with increasing number of dynamic agents present in the scene. This thesis provides qualitative and quantitative evaluations for each of the approaches proposed and show that we perform better than the current approaches.

Acknowledgements

I would like to thank my supervisor Professor John K. Tsotsos for providing me the extreme flexibility to work on projects that I love. I thoroughly enjoyed working with him throughout the course of three years. I have been fortunate to work under his supervision and would like to thank him for his support, helpful discussions and guidance during not only my Master's research but also my Bachelor's research since the day I met him first in August 2014. I would like to thank my committee member Professor Michael Jenkin for his helpful discussions. All meetings with him have left me extremely motivated and inspired with a sense of passion towards robotics. I would also like to thank Professor Costas Armenakis for being on my Oral examination committee.

I would like to thank Bao Xin Chen for all the helpful discussions we had in the lab. I would like to acknowledge his support for participating in the empirical analysis of my research, helping me while deploying robots out in the real world, for working with me on the Person Following Robot project and being a great friend. I want to thank my brother Sidharth Sahdev for his words of encouragements, suggestions on planned executions of demo

videos, building datasets and his support throughout my Masters thesis. He has been a very motivational agent during my research career.

I would like to acknowledge the support of Toni Kunic for his linux debugging skills, uploading datasets, videos, etc. on the project pages for my publications, latex help and feedback during lab meetings. I would like to thank Omar Abid and Vassil Halachev for the moral encouragement, support, creating a friendly and warm atmosphere in the lab. Amir Rasouli for his extremely helpful and critical comments which have been a great source of encouragement throughout. Iuliia Kotseruba for her support in linux issues when I first joined the lab and feedback during lab meetings. Calden Wloka for his helpful suggestions during the lab meetings and for being a great friend. Asheer Bachoo for helping me with linux issues and c++ programming. Markus Solbach for research discussions and introducing me to the powerful IDE for C++ which is Clion. Amir Rosenfeld and Sang Ah for their useful feedback during lab meetings. I would also like to thank Stephen Sutherland from Crosswing Inc. for encouraging discussions on the virtualMe robot and about social robots in general.

I would like to thank Mitacs for providing me a graduate fellowship during the first year of my graduate studies which has been a great help financially during the course of this thesis. I would like to thank NSERC Canadian Field Robotics Network (NCFRN) for providing me with the invaluable opportunity to deploy my robots in indoor hotel environments and outdoor environments during the NCFRN annual field trials. I would additionally like to thank Isabelle Lacroix for helping me getting access/permissions to rooms for my indoor robots during the NCFRN Field trials. I would

like to thank Francisco Moreno, Jose Luis Blanco Claraco and Feroze Naina for proving me an opportunity to work on app developement in C++ for the MRPT code-base which strengthened my C++ skills in summer 2017.

I would like to thank the participants in the localization in dynamic human environments part. All human participants based studies were sanctioned by the Office of Research Ethics at York University (Ethics Approval certificate number: STU 2017-065).

Finally I would like to thank my mom and dad, Kiran and Sunil Sahdev for providing me with love and understanding me in all times when I was stuck. For their motivation and encouragement throughout my research career.

This thesis is dedicated to my brother and my parents, Sidharth, Sunil and Kiran Sahdev, for their unconditional love and support throughout my life.

Contents

List of Figures

List of Abbreviations

CNN	Convolutional Neural Networks
CPU	Central Processing Unit
fps	frames per second
GPS	Global Positioning System
GPU	Graphical Processing Unit
IMU	Inertial Measurement Unit
OAB	Online Ada-Boosting
PID	Proportional Integral Derivative
RANSAC	Random Sampling and Consensus
RGB	Red Green Blue
RGBD	Red Green Blue Depth
RGBSD	Red Green Blue Stereo Depth
ROS	Robot Operating System
SOAB	Selected Online Ada-Boosting
SLAM	Simultaneous Localization and Mapping
VO	Visual Odometry
WO	Wheel Odometry

Chapter 1

Background

1.1 Introduction and Motivation

Robotics finds application in a range of different fields. A key issue in many potential application areas is the need for the robot to operate within an environment that is populated by other users (people) who execute independent motions thus complicating sensing and planning tasks. To take but one example, imagine the deployment of an autonomous robot in a hospital environment. Such a robot would have to be able to navigate in the hospital corridors autonomously while avoiding moving people, beds, and other dynamic events that take place in the corridors. The robot might have to follow a particular nurse/doctor to a specific room using a person following behaviour [1], [6]. The robot would need to address the localization problem [7], [8] which means knowing its current pose/location with respect to its environment. The robot might need to have the ability to do place recognition [9] to know the place it is in currently, perform autonomous

navigation [10], finding navigable collision free space [11], generating a map of the environment the robot is operating in [12], address the problem of simultaneous localization and mapping [13] and many more capabilities.

Although there are many computational tasks required of such a robot operating in dynamic environments, one enabling capability is having the robot to be able to know its current pose in this environment, and it is this problem that is one of the major components of this thesis. Localization of a robot in static environments with a known map is much easier [14], than when the map is unknown and the environment is not static. Basic localization approaches for known and unknown static environments can be found in most texts on robots (e.g., [15]) and for properly conditioned robots and sensors this problem can be considered solved. This thesis addresses a more complex version of the problem of localization of a mobile robot in a dynamic environment with a known 2D occupancy map with dynamic obstacles with unknown trajectories.

Another major contribution of this thesis is the ability for a robot to be able to follow a given person (the target) in complex dynamic environments under challenging situations. Basic person following behaviour under controlled environments is addressed in [16]. Keeping track of the target under challenging situation and following the target over long periods of time remains an open problem. In this thesis we address this problem as well. The robot follows a given target agent (human) and follows it under varying illumination conditions, appearance changes, partial/complete occlusions, etc. Two approaches are proposed to address this problem. A convolutional neural networks (CNN) based approach and an Ada-boosting

Figure 1.1: Pioneer 3AT robot mounted with a Point Grey Bumblebee stereo camera.

based approach are developed.

In this thesis, we make use of a standard RGBD sensor (a stereo camera) for environmental sensing and a commercial robot base for locomotion (see Figure 1.1). This thesis concentrates on two critical issues when operating a vehicle in a pre-mapped environment occupied with dynamic obstacles: localization and person following behaviour. Each of these topics is considered in the thesis and discussed in detail.

We use stereo vision in our work as other sensors are error prone or are limited by other factors. For example, shaft encoders are not too reliable because wheels slip or lose contact with the ground and this leads to accumulation of error over time leading to the problem of dead reckoning over long distances [17]. Other sensors like sonar, radar, lasers could be inappro-

priate in places like hospitals, schools, universities, hotels, etc. Additionally these sensors might be deemed inappropriate due to reasons of concealment or possible confusion with broadcasts of other robots nearby.

1.2 Thesis Outline

The thesis is divided into five chapters.

- **Chapter 1** describes the motivation behind this thesis, and provides an overview of relevant literature and background.
- **Chapter 2** describes one of the approaches used for person following robots using Selected Online Ada-Boosting, provides empirical evaluation and results obtained from this approach. The material presented in Chapter 2 is based on and extends the paper "Person Following Robots using Selected Online Ada-Boosting with a Stereo Camera" published in the *14th Conference on Computer and Robot Vision*.
- **Chapter 3** describes the second approach used for person following robots using Convolutional Neural Networks (CNNs), provides empirical evaluation and results obtained from this approach. The material presented in Chapter 3 is based on and extends the paper "Integrating Stereo Vision with a CNN tracker for a person-following robot" published in the *11th International Conference on Computer Vision Systems*.
- **Chapter 4** describes the approach used for Indoor Robot Localization in Dynamic Human environments using visual odometry and a global

pose refinement technique. It provides an empirical analysis and shows the results obtained from the work. The material presented in Chapter 4 is based on and extends the paper "Indoor Localization in Dynamic Human Environments using Visual Odometry and Global Pose Refinement" which has been accepted to be published in the *15th Conference on Computer and Robot Vision*.

- **Chapter 5** provides a conclusion to the thesis and provides some interesting future work that can be done using the components proposed in the thesis.

1.3 Literature Review

In this section we review existing relevant work in person following robots and mobile robot localization.

1.3.1 Person Following Robots

Here we provide existing literature about person following robots. We divide this section into real time trackers being used for person following robots, provide some literature on object tracking approaches in general and provide some relevant work on CNN-based trackers.

Real-Time Tracking

Person following robots have been researched as early as 1998 [18] where the authors used basic color and contour information of the target for tracking. In 1999, Ku et al. [19] attached a rectangular shape to the back of the

person as the interest region with a particular color. Their method could solve the simple detection problem, but it did not provide any robustness. In 1998, Piaggio et al. [20] started using optical flows for a person following robot. Similar work was done in [21] and [22] as well. However, optical flow has the restriction that the person and background must have different motions which is not always the case. In 2003, Beymer et al. [23] used wheel odometry to subtract background motion and estimate the person location. However this only works well on uniform surfaces. In 2003, Tarokh et al. [24] used colour and shape of the person's clothes as features for detection. Although their method improved the robustness over Ku et al. [19], they did not consider situations when the target changes his/her appearance heavily. In 2006, Yoshimi et al. [25] used feature points (edges or corner points) detection and combined the pre-registered color and texture of the clothes. This method provided good robustness when the person is making a turn or walking in upright poses. In 2007, Calisi et al. [26] used a pre-trained appearance model to detect and track the person. Their method could provide a good tracking result if they trained the model well enough with a lot of data. However, dynamic environments are unpredictable, and the target might change appearance from time to time. Similarly, in 2007, Chen et al. [27] used sparse Lucas-Kanade features to track the target. But the features could be lost if the person is turning, or changing appearance. Again in 2007, Takemura et al. [28] used the H-S Histogram in hue-saturation-value (HSV) color space, where HSV is robust to illumination since V (lightness) can be considered separately. In 2009, Satake et al. [29] used depth templates and SVM to train a human upper body classifier to track the person. However,

this method did not handle cases such as crossing, partial occlusion, etc. In 2010, Tarokh et al. [30] used HSV and controlled the light exposure to handle light variations. An update was made in 2014 to improve the following speed [31]. Some other fundamental feature tracking algorithms were also used in later literature, e.g., SIFT feature based [32] in 2012, HOG feature based [33] in 2013 and [34] in 2014, etc. In the latest work (2016), Koide et al. [35] applied height and gait with appearance features for person tracking and identification, but height and gait are only limited to the target walking in an upright position. The method is not robust when the target changes its clothes or puts on a backpack ([36] also has this problem).

People have been using various other sensors for person following robots like laser based approaches [37], [38] and RGBD camera based approaches, e.g., Kinect [39], [40]. Kinect has the drawback of only working indoors. Laser based approaches might not be suitable for places like hospitals, schools, or retail stores which might have a restriction on the usage of laser.

Object Tracking

Real-time object tracking is an important task for a person-following robot. Many state of the art algorithms exist that can achieve high accuracy (robustness), e.g., MGbSA [41], CNN as features [42], Proposal Selection [43], deep learning [44], Locally Orderless tracking [45], etc. However, these approaches do not target real-time performance. Some other works that focus on computation speed include (Struck SVM with GPU) [46], (Structure preserving) [47], (Online Discrimination Feature Selection) [48],(Online AdaBoosting) [2], etc. Recent work from Camplani et al. [4] (DS-KCF) used

RGBD image sequences from a Kinect sensor to track objects under severe occlusions and rank highly on the Princeton Tracking Benchmark [49] with real-time performance (40fps). One of the earliest works using convolutional neural networks (CNNs) for tracking appeared in 2010 by Fan et al. [50]. They considered tracking as a learning task by using spatial and temporal features to estimate location and scale of the target. Hong et al. [51] used a pre-trained CNN to generate features to train an SVM classifier. Zhai et al. [44] also used a pre-trained CNN, but added a Naive Bayes classifier after the last layer of the CNN. Zhang and Suganthan [52] used one single convolutional layer with 50 4-by-4 filters in the CNN structure. The network was trained from scratch and updated every 5 frames. Gao et al. [42] used a pre-trained CNN as a feature generator to enhance the ELDA Tracker [53]. Held et al. [54] proposed deep regression networks with which they were able to track with high accuracy and their approach could run at 100 fps. However their network had to be trained with huge amounts of data in order to have a good performance.

CNN Using RGBD images

Training a CNN model with RGB and stereo depth images is another approach proposed for the person following behaviour. Previous work includes using RGBD CNNs for object detection [55] and object recognition [56]. Couprie et al. [57] used RGBD images to train a single stream CNN classifier to handle semantic segmentation. Eitel et al. [56] trained RGB layers and D layer separately in two CNN streams. These two streams were combined in the fully connected layer.

1.3.2 Robot Localization

Localization of mobile robots refers to the ability of the robot to know its pose at any given time instance. Essentially this requires the robot to answer the question, *"Where am I?"*. To answer this question, the robot may rely on a variety of sensors, techniques such as wheel odometry using shaft encoders [58], laser odometry using LIDAR [59], inertial navigation systems using gyroscopes and accelerometers [60], visual odometry using cameras [61], global positioning systems [62] and Sonar / Ultrasonic sensors [63]. Each of these approaches have their own strengths and weaknesses. For instance wheel odometry suffers from accumulation of errors due to slippage, lasers provide long range depth information but provide no visual context about the scene and do not work with glass walls, cameras provide good visual information about context but are not be able to provide long range depth information, GPS does not work in indoor environments or its signal might degrade in city environments. Often approaches rely on techniques known as sensor fusion to leverage data from multiple devices and provide an accurate estimate about the pose of the robot. One of the current state of the art techniques for localization is based on a sensor fusion approach using data from a 3D laser and monocular camera by Zhang and Singh [14]. Their approach ranks at top of the KITTI visual odometry benchmark [64]. Another interesting sensor fusion based localization technique is that of Tsotsos et al. [65] where they used data from an IMU and monocular camera and performed better than the current state of the art Systems.

Their technique, when first presented, performed better than Google Tango[1] visual odometry for smart phones.

In this section we focus primarily on localization using visual sensors particularly stereo vision. The process of estimating ego-motion (translation and orientation of an agent (e.g., vehicle, human, and robot)) by using only the input of a single or multiple cameras attached to it is called Visual Odometry [66]. The work of Aqel et al. [61] provides an overview on the different techniques for addressing localization using visual odometry. The term visual odometry was first introduced by Nister et al. [67]. Nister provided a basic approach to compute ego-motion of a vehicle based on stereo and monocular images. The basic steps for estimating the motion are (*i*) Match features between left and right images (*ii*) Track features for a certain number of frames and use RANSAC for outlier rejection to further refine the pose, (*iii*) Triangulate all new feature matches and repeat step *(ii)* a number of times to estimate the pose. The concept of visual odometry (VO) was also used in 1987 by Matthies and Shafer [68] where they used stereo cameras to model errors during navigation. They perform stereo matching and solve for motion estimation by finding out the rotation and translation matrix between each successive frame. Matthies and Shafer's work [68] and Nister et al. [67] form the basis of most approaches of visual odometry today. Most VO approaches today try to optimize these approaches in an efficient manner to produce optimal results. Visual odometry approaches are primarily based on features, appearance or a combination of both feature and appearance based. Howard's work [69] is one of the seminal works in

[1] `https://developers.google.com/tango/`

the field of stereo visual odometry. Howard's approach makes a modification in the inlier detection stage during feature matching. They use the fact that a pair of feature matches is consistent if the distances between two features in frame a is identical to the distance between the corresponding features in frame b. Any pair of matches for which this does not hold true are rejected. By improving the quality of features being selected for motion estimation the performance of the algorithm is greatly improved. Kitt et al. [70] used an iterated sigma point Kalman Filter together with a RANSAC-based outlier rejection approach to estimate ego motion of the vehicle. They bucketed their features to extract information from most parts of the image. Feature based approaches have been used by NASA on the Mars rovers in Maimone et al. [71].

In 2007, Klein and Murray [72] presented a SLAM approach known as PTAM (Parallel Tracking and Mapping) to create a map of the scene and in parallel estimating the pose of the camera. They separated the mapping and pose estimation techniques into two parallel threads one for mapping and the other for tracking of features for pose estimation. They were able to map and estimate motion of a hand-held camera with high accuracy and speed. Following the approach of PTAM, Pire et al. [73] proposed S-PTAM in which they overcame the limitations of the PTAM approach. They used a stereo camera for doing localization and mapping in seperate threads minimizing the inter-thread dependency. They used binary features to describe visual point landmarks thereby reducing storage requirements and improving speed. They also have a maintenance process which iteratively refines the map. Other details and empirical results about

SPTAM can be found in [74] where they compare their localization accuracy to other Visual odometry based techniques like ORB SLAM 2 [75] and LSD-SLAM [76].

Cvišić and Petrović [77] proposed a visual odometry technique SOFT to estimate vehicle pose. They extract features in an intelligent manner by carefully selecting features based on its age, strength, initial descriptor, refined current position in image, etc. and track the reliable features. They also use a 3 point RANSAC scheme fused with IMU Measurements to further refine the pose. Geiger et al. [5] proposed the libviso SLAM algorithm to compute the pose of the robot and construct 3d maps from high resolution stereo images in real time. Their approach runs successfully on a CPU at 25 fps for the localization part and 3-4 fps for the map construction. Similar to PTAM [72] they also separated localization and mapping into two different threads which does not limit the computation of the localization to be bottle-necked by the costly mapping operation. Their approach takes as input a pair of stereo images. For feature extraction they first pre-filter their image with a 5x5 blob and corner masks and then employ non-maximum and non-minimal suppression on these filtered images. When they get the feature candidates, features are matched in a circular fashion. Starting from the current left image, a match is searched in the previous left image, next in the previous right image, next in the current right image and finally in the current left image again. All features that successfully match the initial feature in current left image are retained as good features. They also employ a bucketing technique which ensures features are extracted from most parts of the image. The ego motion is estimated by minimizing the

sum of re-projection errors and refining obtained velocity estimates by a Kalman filter. A RANSAC approach is used to estimate the inliers and reject the outliers in the process of estimating the rotation and translation matrices which describe the motion between each frame. We build on top of this localization approach in this work by making some modifications to their visual odometry approach and integrating wheel odometry and a global refinement based on the floor plan in our approach. Their approach performs well on the KITTI Odometry benchmark and is able to handle sparsely populated dynamic scenes quite well.

Localization has also been addressed using Place Recognition based techniques as in [78] and [9]. Recently in 2017, Zhu [79] proposed an approach GDVO (gradient dense visual odometry) for visual odometry using a stereo camera. They extract features in the gradient domain which makes their system robust to illumination changes. The main contribution of their approach was using a dual Jacobian based optimization which is integrated with a multi-scale pyramid scheme while estimating the ego motion of the vehicle. Their approach ranked as 2nd on the KITTI benchmark using vision only algorithms at the time of publication. Another interesting work for estimating pose of the vehicle was proposed by Mur-Artal et al. [80] which they termed as ORB-SLAM. They proposed a SLAM approach using a monocular camera and used ORB features to estimate pose with high accuracy. Engel et al. [76] proposed LSD-SLAM in which they estimate pose accurately using direct image alignment and construct a pose graph of the key-frames. Their approach is featureless and they track key-frames based on image alignment and depth estimation techniques.

Pink [81] and Pink et al. [82] proposed an approach to estimate the pose of the vehicle by visually matching local features with a global feature map obtained from geo-referenced aerial imagery. They matched lane markings in the global map to local lane markings to estimate ego-motion of the vehicle. Chu et al. [83] used a similar concept to estimate the pose of the vehicle by using GPS measurements and a 2D city plan. They refine the position of the GPS location and estimate the camera pose, based on detecting vertical corner edges from a single image by mapping the cuboidal buildings to a 2D city map with building outlines. For doing so they compute TICEP (Tilt-Invariant Corner Edge Position) features by estimating vanishing points, identifying vertical buildings' corner edges and normalizing the tilt angles. After TICEP feature extraction, LOHs (location orientation hypothesis) are used to choose the location that best geometrically corresponds between the corners on the 2D map and extracted TICEP features. Our work also used a similar refinement stage to refine poses obtained from Visual Odometry which we correct using information from the global map. We do refinements at points where we can do a geometric matching between the interest points in our map to the image that the robot might see when its traversing a path around that area.

In the context of indoor localization, Chu et al. [84] used floor plans to address localization. They do matching of the video stream of the camera to estimate the pose of the camera. They do piece-wise point cloud and free space matching to align the geometric structure with the given floor plan. Their localization technique is similar to that of a particle filter based localization approach [85] where initially all poses are equally likely and

gradually weaker particles die out and the pose can be estimated with high accuracy.

In 2002, Wang and and Thorpe [86] introduced the concept of detection and tracking of moving objects in SLAM (Simultaneous Localization and Mapping). They divide the map into stationary object map and moving object map and do not consider the moving objects while map generation and localization. They used laser scans obtained from the objects to segment out moving objects. Yang and Wang [87] estimated ego motion of the vehicle in highly dynamic environments using laser information. They were able to addresses the pose estimation problem even when more than 50% of the scene was covered with the dynamic agent. They used a multi-model RANSAC approach to classify the motion of the feature belonging to either static, unknown or moving type. They did not employ any geometric features which might sometimes not be reliable in urban settings. They also did not use odometry information, only laser data. In 2016, Sun et al. [88] proposed a localization technique for dynamic environments. Their approach was based on a bayesian estimation process and used laser data and odometry information. They addressed the localization problem by means of a particle filter integrated with a distance filter and a scan matching approach which helps them handle dynamic obstacles in the environment. They compared their approach with the AMCL (Adaptive Monte Carlo Localization) technique in the ROS framework [89]. They did not provide information about the quality of the dataset nor about the dynamic nature of the environment. However these approaches use a laser scanner which may not be permitted in places like hospitals, schools, etc.

1.4 Objective of the Work

The objective of the thesis can described below:

- **Person Following Robot**: To have a robot system equipped with a stereo camera to follow a given target under challenging situations. The robot should be able to follow the target in difficult situations like appearance changes, pose changes, illumination changes, following the target even when the target is transiently out of the robot's camera view. Two approaches are developed which are successfully able to achieve this aim.

- **Indoor Robot Localization in Dynamic Indoor Environments:** To have a visual system which allows the robot to localize in challenging dynamic environmwnts. We aim to provide a wrapper to traditional visual odometry algorithms, so they can use our proposed approach and work with a higher accuracy in dynamic environments.

1.5 Significance and Contributions

Today with the increasing trend of Artificial Intelligence, the world is building a lot of autonomous agents in the form of robots, autonomous cars, autonomous driving assistants, drones, underwater robots, etc. We have focused on the aspect of indoor robots in this thesis particulary robots that are deployed among humans.

The outcome of this thesis is a system which enables a robot to follow a given person under challenging situations and a system which enables

an autonomous agent to localize in crowded spaces in an efficient manner. This has wide application today. Person following robots can be used as autonomous carts in grocery stores [90], personal guides in hospitals, or airports for autonomous suitcases [91]. It can also be used in hotels to welcome guests and escort their luggage to their respective rooms, the robot follows the person while the luggage is kept on top of the robot. This can be used in hospitals for autonomous following behaviour to help hospital staff transport heavy objects, in corporate offices to transfer small objects from one location to another, in factories to transfer equipment from one point to another and many more such places. The localization approach proposed in the thesis can be integrated with standard navigation approaches which makes the applications of this thesis even wider.

In this thesis we propose three approaches for two key components involved in navigation of robots. We propose a convolutional neural netowrk based tracker and an online ada-boosting approach for person following robots. We deployed this robot in the real world and showed that the robot was able to successfully follow a given target under challenging situations. We also present a localization framework which estimates the robots pose using an approach which integrates wheel and visual odometry and further uses a global refinement stage to get rid of error accumulation. We tested this component by deploying our robot in a university corridor and reported empirical results in the thesis.

Parts of thesis have been published or will be published in the following articles and conference papers:

- B.X. Chen, R. Sahdev and J.K. Tsotsos, "Person Following Robot using Selected online ada-boosting with stereo camera", in *14th Conference on Computer and Robot Vision (CRV)*, pp 48-55, IEEE, 2017. (Best Robotics Paper Award.)

- B.X. Chen, R. Sahdev and J.K. Tsotsos, "Integrating Stereo vision with a CNN tracker for a person-following robot", in *International Conference on Computer Vision Systems (ICVS)*, pp. 300-313, Springer, 2017. (Finalist for the Best Conference paper award.)

- R. Sahdev, B.X. Chen and J.K. Tsotsos, "Indoor Localization in Dynamic Human Environments using Visual Odometry and Global Pose Refinement ", in *15th Conference on Computer and Robot Vision (CRV)*, IEEE, 2018. (accepted to be published)

Chapter 2

Person Following Robots using Selected Online Ada-Boosting[1]

2.1 Introduction

Person following robots need a robust and real-time algorithm to solve the tracking problem in a dynamic environment which may encounter unexpected circumstances; for example, the tracking target might be occluded by other instances, the lighting condition in the scene might change rapidly, and the target might change its pose dramatically (eg: squat down and pick up something from the floor or removing a bag from the person (see Figure 2.1)). To the best of our knowledge this is the first work which can

[1] this chapter is an extended version of the paper which appeared in 14^{th} *Conference on Computer and Robot Vision* in [1]

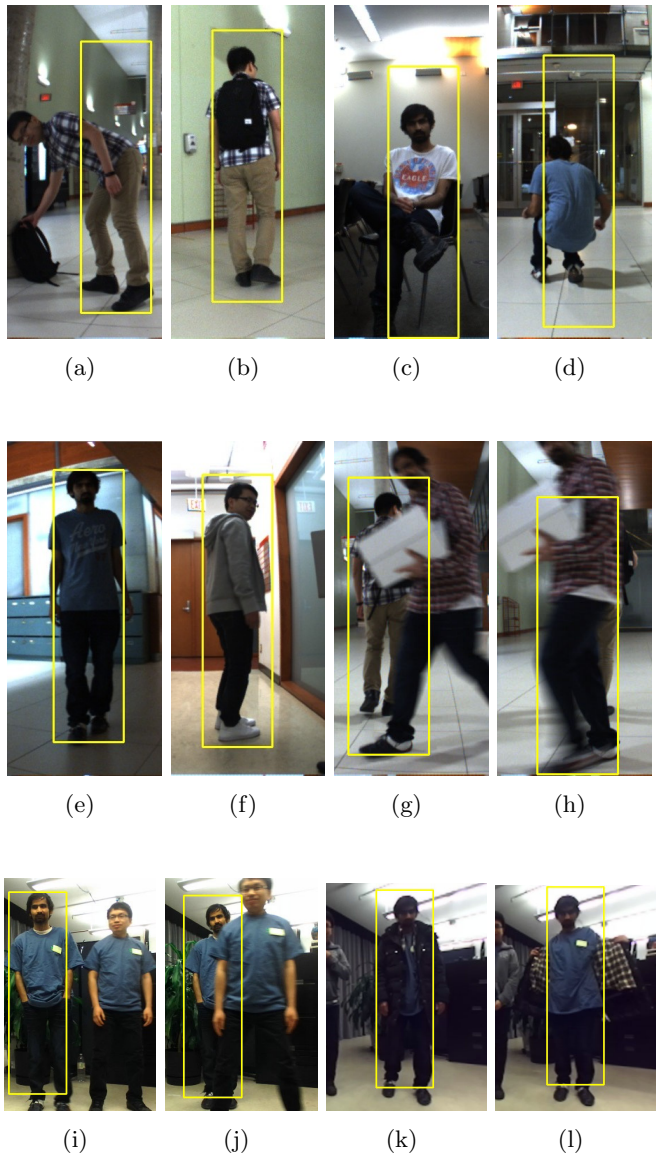


Figure 2.1: Different cases that our approach (Selected Online AdaBoosting) can handle. (a) picking bag. (b) wearing bag. (c) sitting. (d) squatting. (e) illumination. (f) side facing. (g) partial occlusion. (h) complete occlusion. (i) standing side-by-side with the same clothes. (j) front crossing with the same clothes. (k), (j) appearance changed.

handle situations when two people are wearing the same clothes and the tracker can still track the correct target under partial and complete occlusions; it can also deal with appearance changes, like removing a jacket, the tracker still tracks the target (human) and not the jacket. Another challenge is maintaining a given distance from the robot to the target, a natural consequence of following behaviour of the robot. The robot being used here is the Pioneer 3AT robot as shown in Figure 1.1. The main contributions of this chapter are as follows: (*i*) a novel approach building on the Online Ada-Boosting tracker, (*ii*) a novel algorithm named Selected Online Ada-Boosting which can run in real-time to follow a given target and is more robust than the current state of the art (see Figure 2.1), (*iii*) a novel stereo dataset of different indoor environments for person following. This chapter is organized as follows. In Section 2.2, we describe our proposed approach which modifies the Online Ada-Boosing algorithm to make it more robust. Section 2.3 describes the system design of the proposed approach. In Section 2.4, we provide the experimental results of our approach and describe the dataset. Finally, Section 2.5 concludes the chapter and provides possible future work.

2.1.1 Depth Detection

In this work, we use depth to assist the tracking model for improving the reliability. Yoon et al. [92] gained aid from depth information to improve the computational speed and accuracy. Depth could also help with background and foreground issues by eliminating the sudden depth changing pixels, e.g., occlusion. Doisy et al. [39] used the Kinect camera and a laser sensor to

propose an algorithm which solves the person depth information for person following.

Nowadays, there are many different types of depth sensors in the market. In the modern publications, researchers prefer RGB-D cameras, eg: Kinect [40], ASUS xTion [36] and [92]. These cameras provide very good depth information only if the robot is running indoor without strong sunlight. Our approach uses a Point Grey Bumblebee 2 stereo camera which can be used both indoors and outdoors. Laser sensors provide another approach to detect depth [33] and [34]. But a laser sensor is expensive and often not permitted in places like hospitals, universities, malls and other similar places.

To obtain the depth information of each pixel in an image, we use a stereo image based algorithm to compute the depth. Since focal length and baseline are constants in a single stereo camera, we are only interested in disparity [93].

2.1.2 Online Ada-Boosting (OAB) Tracker

Boosting algorithms have been used in many areas in machine learning and computer vision ([95], [96], [97], [98]). Boosting usually trains with offline datasets. Online Ada-Boosting algorithm for tracking an object in real-time has been described by Grabner et al. in [94] and [2]. To achieve real-time tracking, Grabner et al. used Haar wavelet features to improve robustness when appearance changes gradually, which was described by Wang et al. in [99].

In OAB, the tracking target is assumed to be given in the first frame

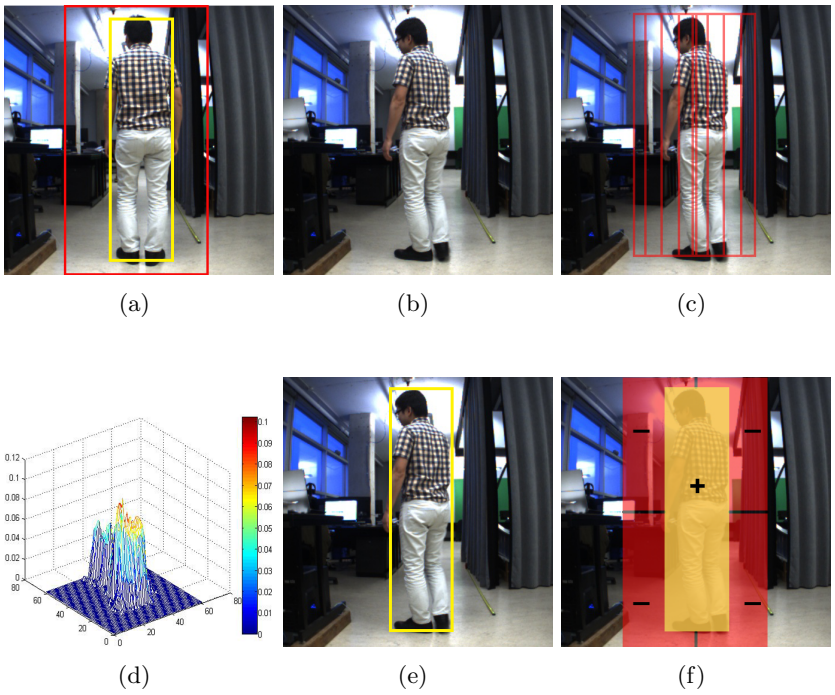


Figure 2.2: OAB updating process: (a) yellow box is the target region, the red box is the search region. (b) is the next frame. (c) is searching and evaluating the patches in the search region. (d) is the confidence map of the evaluation. (e) is the best matching with minimum error. (f) update the classifier with positive and negative patches. After (f) then go back to (a) to search in the next frame. Similar to [94], [2]

(selected by human or detected by an off-line detection algorithm). The selected patch is used as a positive example to train the classifier. Then random patches are extracted from four regions (upper right, upper left, bottom right, bottom left, see Figure 2.2(f)) in the search area as negative examples. These random patches contain negative features, e.g., windows, wall, furniture, etc. An initial classifier is trained from these positive and negative patches. In the second frame, the target is detected using the

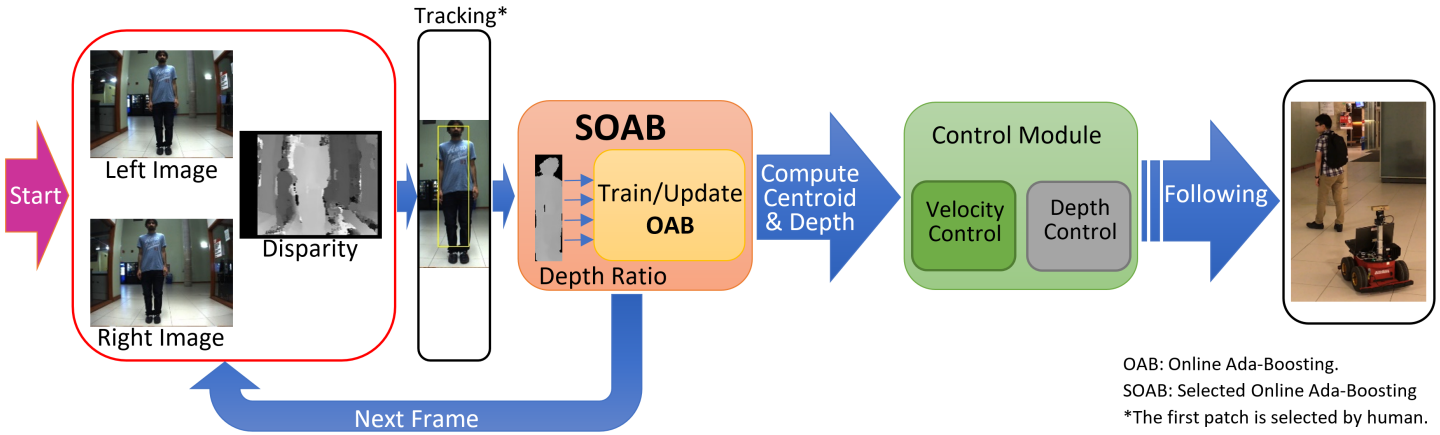


Figure 2.3: Proposed Approach: Tracking Module and the Control Module

classifier. The patch in the search region with minimum error is the best responding example. This patch is used as a positive example and the surrounding random patches from the four regions as negative examples to update the classifier. The steps performed on the second frame are continued on the subsequent frames (see Figure 2.2).

In order to achieve real-time boosting, OAB does not use all weak classifiers to calculate a strong classifier [2]. Instead, it selects N weak classifiers from all M global weak classifiers. In the following equations, H^{weak} is the set of all weak classifiers, $H^{selected}$ is the set of selected weak classifiers from H^{weak}, y is the prediction of boosting, and α_n is the weight of each selected classifiers.

$$H^{weak} = \{h_1^{weak}, ..., h_M^{weak}\} \tag{2.1}$$

$$H^{selected} = \{h_1^{selected}, ..., h_N^{selected}\} \tag{2.2}$$

$$h_n^{selected} = h_m^{weak} \tag{2.3}$$

$$y = \sum_{n=1}^{N} \alpha_n * h_n^{selected} \tag{2.4}$$

α_n in Equation 2.4 is calculated according to the error of selected weak classifier $h_n^{selected}$.

2.2 Approach

To the best of our knowledge, this is the first work that introduces the Online Ada-Boosting tracking algorithm (OAB) [2] for a person following robot. On top of the OAB algorithm, we add a depth image as an additional

tool to assist the Ada-Boosting approach. We call this new modification as Selected Online Ada-Boosting (SOAB).

2.2.1 Computing Depth From Stereo Images

Assuming that the cameras are parallel, have identical focal lengths identical pixel aspect ratios and parallel axes and are separated by a distance, B along the common x axes then the depth of each pixel can be easily calculated from the following equation [100]:

$$Z = \frac{fB}{x_l - x_r} \tag{2.5}$$

f is focal length. B is the baseline. x_l and x_r are the left and right image coordinates.

2.2.2 Classifier Initialization

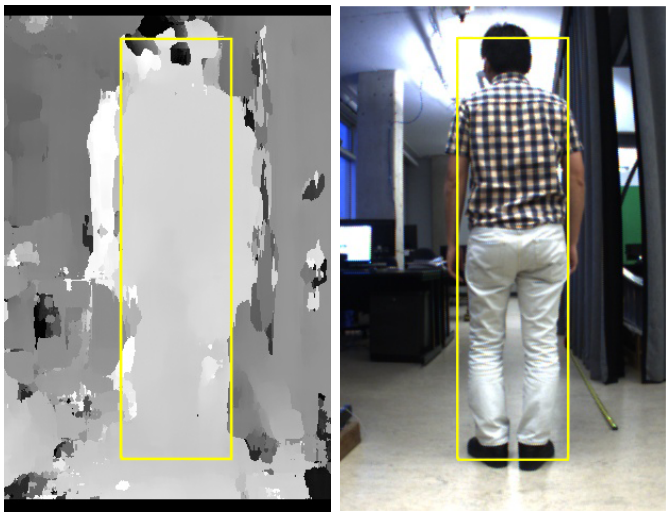
[94] and [2] initialized the first frame with a human to draw the bounding box. Here we present two ways to initialize the target to the tracker: *user defined* and a *pre-defined* bounding box.

For the *pre-defined* case, a bounding box was placed in the center of the image frame. The target has to walk into the bounding box at a particular distance from the robot. If all these conditions are satisfied, then the robot starts to initialize the classifier and follows the person. In our experiment, we draw a bounding box at pixel coordinates (272, 19) with the width equal to 100 pixels and the height as 390 pixels, and the default disparity is 200 (this is the initial disparity for the first frame).

For the *user defined* case, we proceed as follows. Since the initial position of the person is known, and the depth image is given by equation 2.5, we can estimate the initial disparity of the person easily. Here we need to overcome a big problem: the depth image is very noisy (see Figure 2.4(a)). Let the initial patch be called I_p. We sort the pixels in I_p according to their disparity value (Note: the larger the disparity, the closer the distance. See equation (2.5)). Then we remove the disparities before 50th percentile and remove the disparities after 75th percentile. After that, we compute the mean of the remaining disparities as the initial depth. This method works, because the body of our target will almost fill the whole initial patch from our experiments (see Figure 2.4(b)), and noisy disparities are typically not be more than 25% in the initial patch. Removing 75% of the disparities will definitely give us a precise result. This was found experimentally that retaining the disparities in this range gives best performance. We tried different numbers here and found that for our chosen stereo camera and disparity estimation, 75% was an appropriate number. This would change with change in choice of the disparity estimation algorithm and stereo camera. In next subsection 2.2.3, we will discuss when to update the classifier.

2.2.3 Selected Online Ada-Boosting (SOAB)

In this section, we will describe how to optimize OAB with given the depth information on each pixel from Section 2.2.1. One of the weaknesses of the OAB algorithm is that the target might not always maintain the same size in the scene. The size of the target could be changed when it is occluded, changing poses, or tracking improperly in the current frame (see Figure 2.1).



(a) (b)

Figure 2.4: (a) is a normalized disparity image. (b) is from the left camera.

These weak detections pollute our classifier badly. Once the classifier adapts to those unwanted features, the tracker looses the target easily. Here unwanted features include background and foreground features. So, the depth of each pixel plays a significant role to calculate the proportion of unwanted features in the current positive patch. We call this proportion the depth ratio, R. Before computing this depth ratio for the positive patch, we need to determine where our target is in the previous frame (here we focus on the distance between the robot and the person).

Once the initial disparity (called $preDisp$) is computed from Section 2.2.2, we estimate the disparity in the second frame. To do this, we run the original OAB algorithm to detect the positive patch in the second frame. Assuming that the displacement of the target can not be more than a threshold β,

the possible disparities that belong to the person are $preDisp \pm \beta$. Then we compute the mean of the pixels in $preDisp \pm \beta$ range as the current disparity (called $curDisp$). We assign $curDisp$ to $preDisp$ and repeat this for later frames to perform tracking.

$$curDisp = Mean(I_p[I_p \in preDisp \pm \beta]) \tag{2.6}$$

The next step is to update the classifier. We do this differently than OAB. We introduce the depth ratio R to evaluate the current positive patch containing a minimum amount of unwanted features. R equals the ratio of the number of pixels that are used to calculate $curDisp$ to that of the total number of pixels in the current patch. The width of patch I_p is w, and the height is h.

$$R = \frac{\sum [I_p \in preDisp \pm \beta]}{w * h} \tag{2.7}$$

Now our algorithm (SOAB) makes the decision. If the depth ratio R is greater than a threshold γ, then we update the classifier using the current positive patch. Otherwise, we do not update the classifier.

2.3 System Design

In this part, we describe the design of our system. Here we use a Pioneer 3AT robot (see Figure 1.1.) which is a four wheeled differential drive robot with an on-board computer. It is configured with a Point Grey Bumblebee Stereo Camera which acts as the only sensor on the robot to sense its environment.

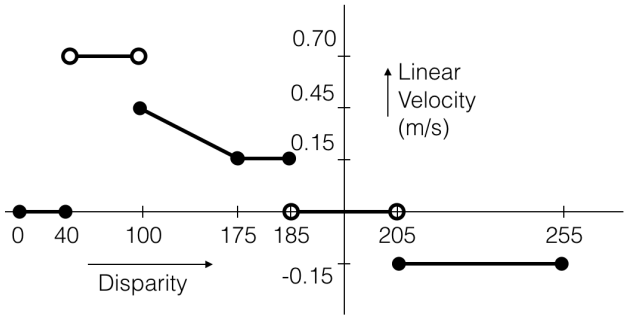
Algorithm 1 SOAB (Selected Online Ada-Boosting)

Data: Camera Stream

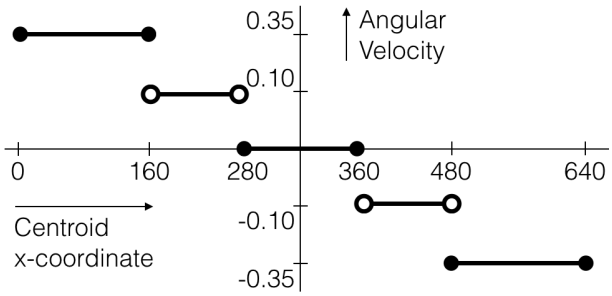
1. fetch left and right image from Camera Stream;
2. select target to track
3. calculate *curDisp*
4. $preDisp \leftarrow curDisp$
5. pre-train OAB;
6. **while** *true* **do**
7. fetch left and right image from CameraStream
8. run OAB to extract a positive patch
9. fetch left and right image from CameraStream
10. run OAB to extract a positive patch I_p
11. $curDisp \leftarrow Mean(I_p[I_p \in preDisp \pm \beta])$
12. $R \leftarrow \frac{\sum [I_p \in preDisp \pm \beta]}{w*h}$
13. **if** $R \geq \gamma$
14. update the classifier
15. $preDisp \leftarrow curDisp$

The system is built using the robot operating system (ROS) to integrate different components involved in the system. Figure 2.3 gives an overview of our system design.

Initial components of the system are responsible for tracking the target (human) and computing the centroid and depth of the target being tracked. Based on these values, the control module computes the corresponding linear and angular velocities for the robot (see Figure 2.5). The controller maintains a predefined distance from the human being followed. It is ensured that the centroid of the human target bounding box is always near the centre of the image within a pre-specified area. This is done by simply steering in the direction to which the person is moving. If the person appears to be moving left in the image, the robot moves leftwards to keep the centroid of the detected human near the center of the image. The robot maintains a set depth from the target. If the person is moving towards the robot, the



(a) Linear velocity vs. Disparity Plot



(b) Angular Velocity vs. centroid of the target

Figure 2.5: Controller Module of our system. (a) The function represents the linear velocity as a function of the target's disparity in the current frame (b) represents the angular velocity as a function of the x-coordinate of the centroid of the target

robot moves backward and vice versa. The linear velocity of the robot is a function of the disparity alone and the angular velocity is a function of the x-coordinate of the centroid of the human being tracked (see Figure 2.5). These functions were obtained experimentally and would change with the change in the robot platform.

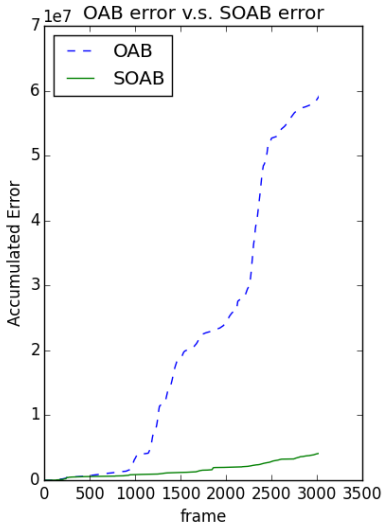
We run this system on a laptop with Intel i7 2.5GHz processor and 16GB RAM (the requirement is lower for our algorithm). The design of various

components involved here is presented in Figure 2.3.

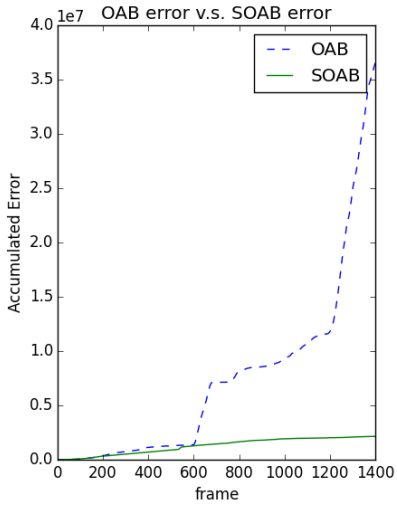
2.4 Experiments and Evaluation

Since the proposed method relies on a dataset that is different from what people did in the past, we were unable to find an existing dataset which satisfies our need (a stereo dataset for a human following robot under challenging situations). As a result, we constructed a dataset of four image sequences to test the robustness of the person following robot system. The person being followed in our dataset exhibits varying motions and challenging poses in different indoor environments (see Figure 2.1). The dataset is built from image sequences captured by the robot in these places. The robot is following a person in a university hallway, a living room, and a lecture hall. We make the dataset of these three places publicly available at our project page[2]. Videos of the robot following behavior of our proposed approach can also be found at the project page[2]. The dataset consists of the person being followed under varying illumination conditions, different poses of the person being followed, partial and complete occlusion of the person being followed and multiple people present in the scene. The resolution of the images is $640 * 480$ pixels. Using the algorithm described above we are able to track people while the robot is moving at up to speeds of 0.70 m/s. The person being tracked also has a maximum speed of 0.70 m/s. It should be noted that our system could be deployed on any mobile robot platform. We tested our proposed approach on a Pioneer 3AT robot (see Figure 1.1).

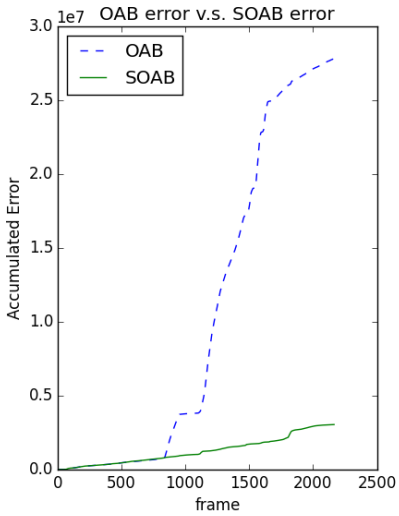
[2] `http://jtl.lassonde.yorku.ca/2017/02/person-following`



(a) Sequence Hall Way



(b) Sequence Multiple Crossing



(c) Sequence Same Clothes Crossing

Figure 2.6: The graphs are comparing the accumulated square error on three different image sequences captured in different places and the target acted very differently. OAB is the Online Ada-Boosting approach and SOAB is our approach Selected Online Ada-Boosting

Our algorithm can run in real-time at a frame rate of 15fps on a single CPU core.

First, we tested our algorithm on an experimental sequence of images. The target in the sequence is turning around, and squatting down (see Figure 2.7). From the result, we could distinguish that SOAB with depth ratio threshold $\gamma = 0.60$ outperforms the original OAB. By selecting the patches to update the classifier does make a huge improvement. In Figure 2.7(f), OAB did a mistake and updated the classifier. The classifier then learned the background as the important feature and as a result continuously made mistakes in later frames. On the other hand, SOAB avoids this problem by using the depth information to make decision on whether or not to update the classifier. We also select a depth ratio threshold $\gamma = 0.80$ for testing. Since the threshold is too high, SOAB skipped most of the frames. This is not how we want SOAB to behave. In the later experiment, we fixed the depth ratio threshold as 0.60.

We made another image sequence to test more challenging scenarios. The target is picking up a backpack from the ground, and someone is passing between the robot and the target in the sequence (see Figures 2.8, 2.9, 2.10). Again in this test case, SOAB achieved the best result overall. Comparing Figure 2.8(b), OAB learnt the background features leading to a mistake in Figure 2.8(c). From Figure 2.8(e-g), OAB learned the features of the crossing person. The second person became the target of the OAB tracker. Since the depth information is used as a gate, SOAB did not update the classifier with unwanted features when depth ratio is less than the threshold. Figure 2.6(a) shows the accumulated square error of OAB and

SOAB. The green line in the graph increases very smoothly meaning that SOAB performed very well without losing track. But, OAB looses track at about frame 900 and becomes very unstable later, roughly at the occlusion in Figure 2.8(f).

Another image sequence was made to test multiple crossing with different speeds. The comparison between OAB and SOAB can be seen in Figure 2.6(b). There are 12 crossing actions in this sequence. SOAB completed this test case without failure. But, OAB failed after the fourth crossing.

The third sequence is for testing when two people are wearing the same clothes. This sequence is the most significant one in our dataset. The result can be seen in Figure 2.6(c). In this sequence, two people are crossing each other, walking in a circle. As expected, the robot is following the same person all the time using SOAB.

2.5 Summary

In this chapter, we described a robust person following robot system using a modified version of Online Ada-Boosting algorithm with only a stereo camera. The system was optimized to perform well in a dynamic environment. Our modified version of OAB performs much better than the original algorithm (see Figure 2.6). We handled difficult situations dealing with similar clothes of people crossing, appearance changes in terms of removing the target's jacket, partial and complete occlusions and were able to run our approach in real-time on a mobile robot. It should also be noted that even though we present our approach for the human following robot, this can be

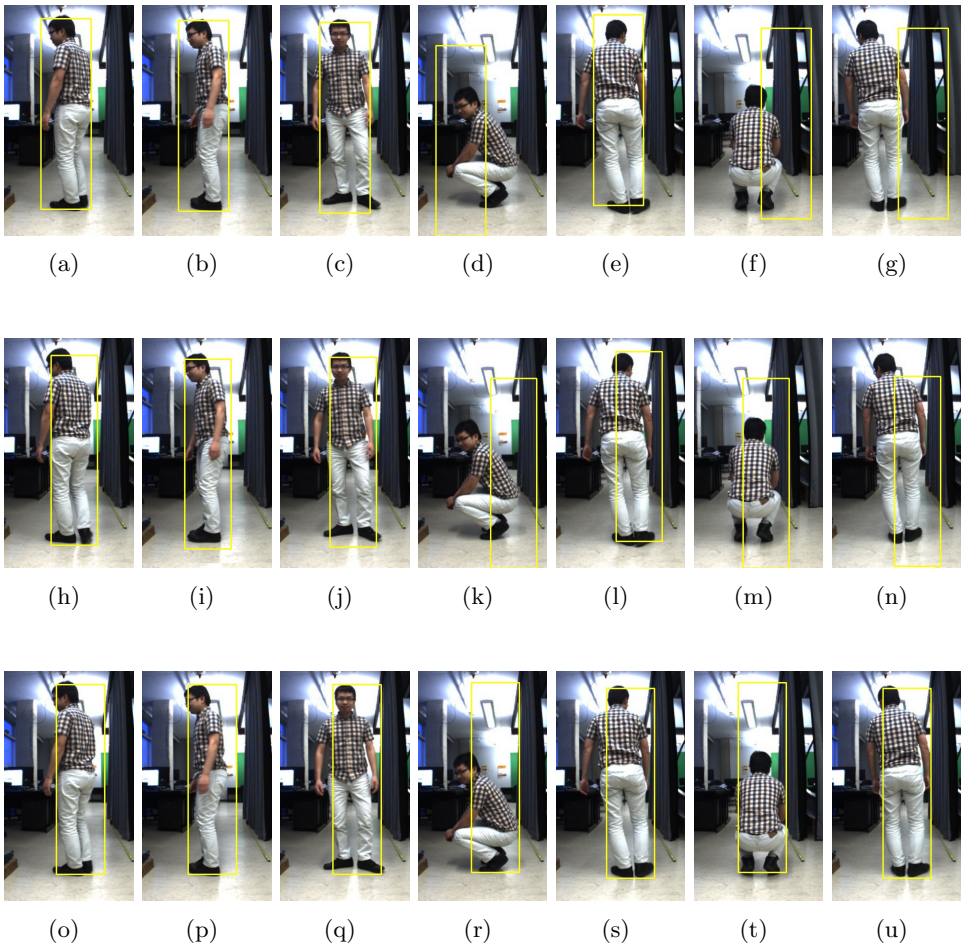


Figure 2.7: (a-g) is tracking using original OAB algorithm. (h-n) is tracking using SOAB with depth ratio threshold $\gamma = 0.30$. (o-u) is tracking SOAB with with depth ratio threshold $\gamma = 0.60$.

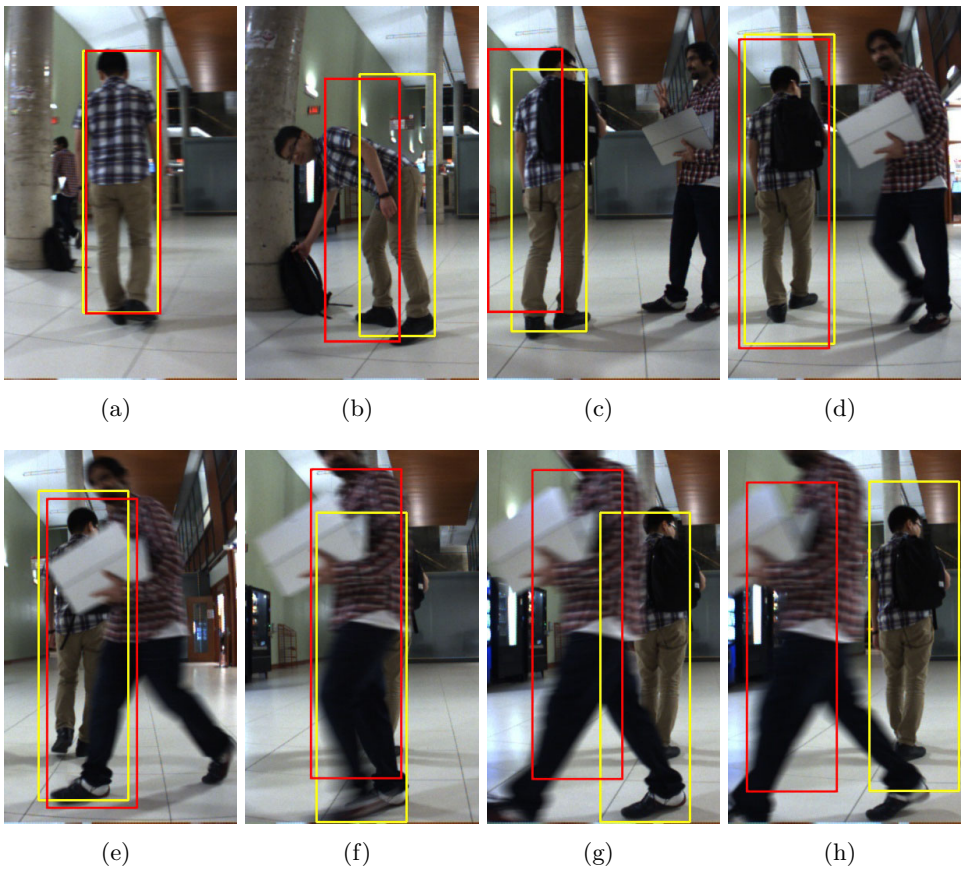


Figure 2.8: Red box is tracking using original OAB algorithm. Yellow box is tracking using SOAB with depth ratio threshold $\gamma = 0.60$. (a-h) are sequences from a hallway.

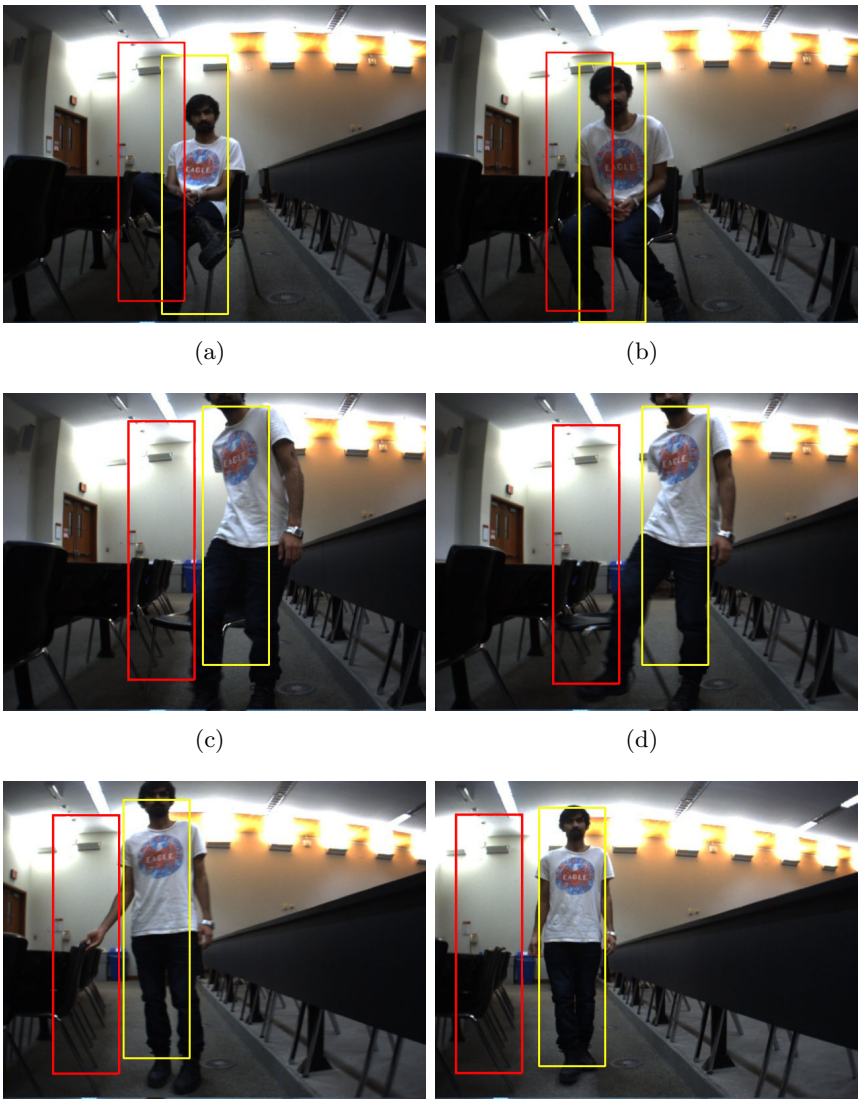


Figure 2.9: Red box is tracking using original OAB algorithm. Yellow box is tracking using SOAB with depth ratio threshold $\gamma = 0.60$. (a-f) are sequences from a lecture hall.

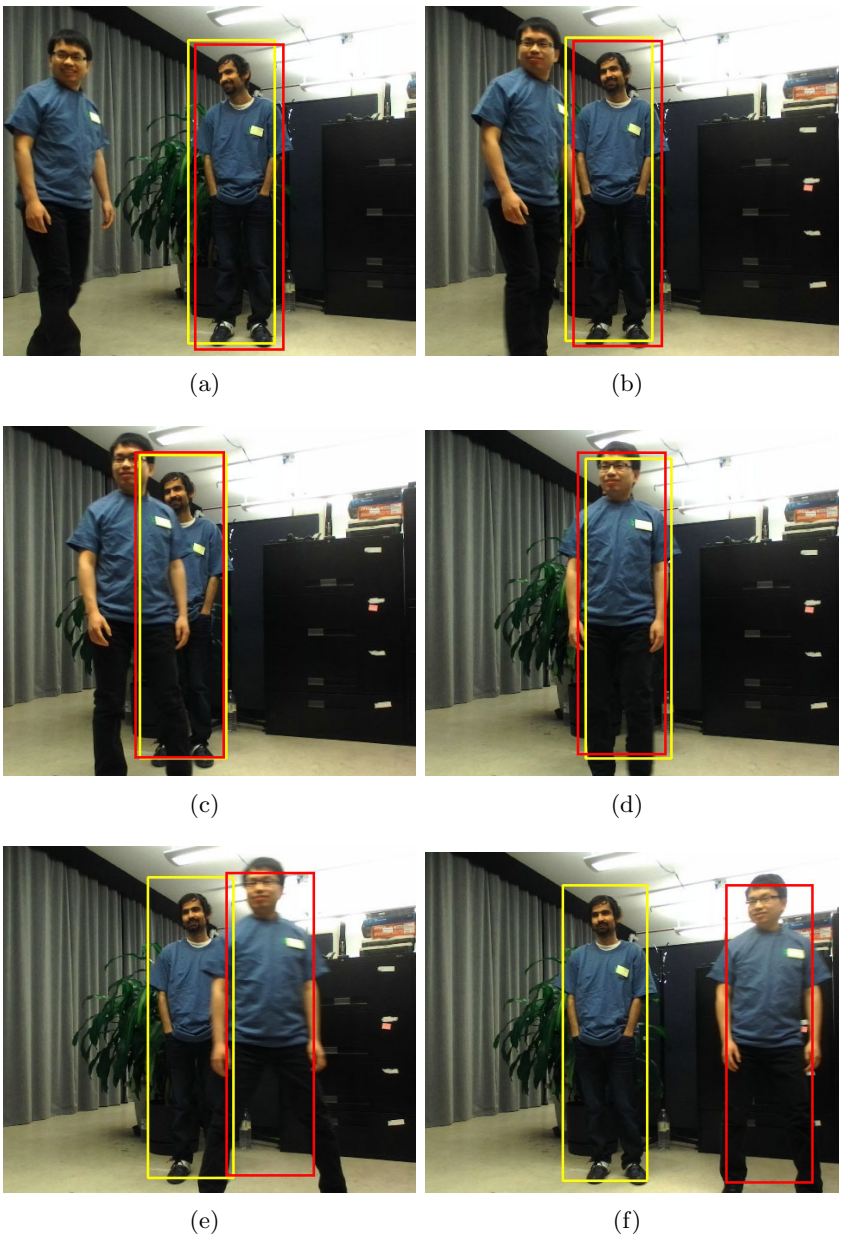


Figure 2.10: Red box is tracking using original OAB algorithm. Yellow box is tracking using SOAB with depth ratio threshold $\gamma = 0.60$. (a-f) are sequences showing crossings with same clothes.

applied to any object following robot as well, but the object needs to be known *a priori*. For instance the robot can follow objects like a handbag, shopping cart, an animal (cat/dog), etc. In this sense our approach targets not only the human following task but also generalizes to other objects as well.

We proposed changes to the OAB algorithm. We believe that there could be further improvements, e.g., using a more robust online boosting tracking algorithm called Online Multiple Instance Learning [101], or increasing the classification error if the bounding box jumps unstably from frame to frame. Another possible future work would be to include the recognition aspect by making use of a human detector to aid in the process of having a better model for the classifier. Another approach of making the following system more reliable could be adding a path planning and obstacle avoidance strategy to the robot control module in our system.

Chapter 3

Person Following Robots using CNNs[1]

3.1 Introduction

In the previous chapter we presented a person following approach which could robustly track the person under challenging situations. Our approach Selected Online Ada-Boosting (SOAB) was built on top of the Online Ada Boosting (OAB) technique [94]. OAB uses traditional hand crafted features like Haar like features from [102] and Local Binary Patterns [103]. In this chapter we propose a new approach where we extract the features using a convolutional neural network (CNN) and integrate information obtained from the stereo depth image to update the person classification model when required. Features extracted from our CNN are of a better quality than the

[1]This chapter is an extended version of the paper which appeared in 11^{th} *International Conference on Computer Vision Systems* in [6]

ones used in OAB. Hence we proposed another tracking module using CNNs in this chapter.

Here, an online convolutional neural network (CNN) is used to track the given target under different situations. In addition to the situations mentioned in Chapter 2, the target being tracked might also move around corners making it transiently disappear from the field of view of the robot. We address this problem by computing the recent poses of the target and have the robot replicate the local path of the target when the target is not visible in the current frame. The robot being used is a Pioneer 3AT robot which is equipped with a stereo camera. We tested our approach with two stereo cameras namely the Point Grey Bumblebee2[2] and the ZED stereo camera[3]. We also evaluate the performance of our CNN based approach with that of SOAB [1] and some other stereo based trackers like the ASE (accurate scale estimation) tracker [3] and the DS-KCF (depth sensitive kernelised correlation filter) tracker [4].

The main contributions of this chapter are: (*i*) A Person Following Robot application using a CNN trained online in real-time (≈20 fps) making use of RGB images and a stereo depth image for tracking, (*ii*) a robot following behaviour which can follow the person even when the person is transiently not in the field of view of the camera, (*iii*) a novel stereo dataset for the task of person following. This chapter is organized as follows. In Section 3.2, we describe our proposed CNN model and the navigation system of the robot. We describe the dataset and experimental results of our approach in

[2] `http://www.ptgrey.com/stereo-vision-cameras-systems`
[3] `https://www.stereolabs.com`

Section 3.3. Finally, Section 3.4 concludes the chapter and provides possible future work.

3.2 Approach

Here we describe our proposed CNN models and the learning process. The input to the CNN is the RGB channel and the computed depth from the stereo images, we call this as RGBSD (RGB-Stereo Depth). Stereo Depth (SD) is computed using the ZED SDK[4]. The CNN Tracker outputs the depth and the centroid of the target. The depth and centroid are then used by the navigation module of the robot to follow the target and replicate the path when required.

3.2.1 CNN Models With RGBSD Images

We develop three different CNN models and use each of them separately to validate our approach. The first model (CNN_v1) uses RGBSD layers as a single image to feed the ConvNet. Similar to conventional CNN architectures, the network contains convolutional layers, fully-connected layers, and an output layer (see Fig. 3.1). The second model (CNN_v2) uses 2 convolutional streams and the input is RGB channels for one stream and just the stereo depth image for the other (see Fig. 3.1). In the fully connected layer, the input is a combination of the flattened output from those two convolutional streams. The third ConvNet (CNN_v3) is a regular RGB image based CNN. It has a similar structure as that of the first model. Table 3.1 shows

[4] `https://github.com/stereolabs/zed-opencv`

Table 3.1: CNN Model 1 (CNN_v1) RGBSD architecture details. Activation function used at each layer was ReLu (Rectified linear units)

type	**input size**	**filter size/ stride**	**Number of Filters**	**output size**
convolution1	28x28x4	3x3x4/1	32	28x28x32
max-pool1	28x28x32	2x2/2	-	14x14x32
convolution2	14x14x32	32x32x3/1	64	14x14x64
max-pool2	14x14x64	2x2/2	-	7x7x64
Fully Connected	3136	-	-	128
OutputLayer	128	-	-	1

the details for one of the architectures of our CNN models. Now we describe our approach to initialize and update the CNN tracker.

Initial training set selection

In order to use the CNN model to track a person, we must initialize the CNN classifier. The initialization is done from scratch using random weights. A pre-defined rectangular bounding box is placed in the center of the first frame. To activate the robot following behaviour, a person must stand inside the bounding box at a certain distance from the robot or the target to be tracked can be manually selected. Once the CNN is activated, the patch in the bounding box is labeled as class-1. The patches around the bounding box are labeled as class-0. Since these two classes are highly unbalanced, we uniformly select n patches from class-0, and copy the class-1 patch n times to form the training set ($n = 40$ in our experiment). This initial training set is used to train a CNN classifier until it has a very high accuracy on the training set. This might make the classifier overfit the training set. To handle this strong over-fitting, we assume that the target pose and appearance should

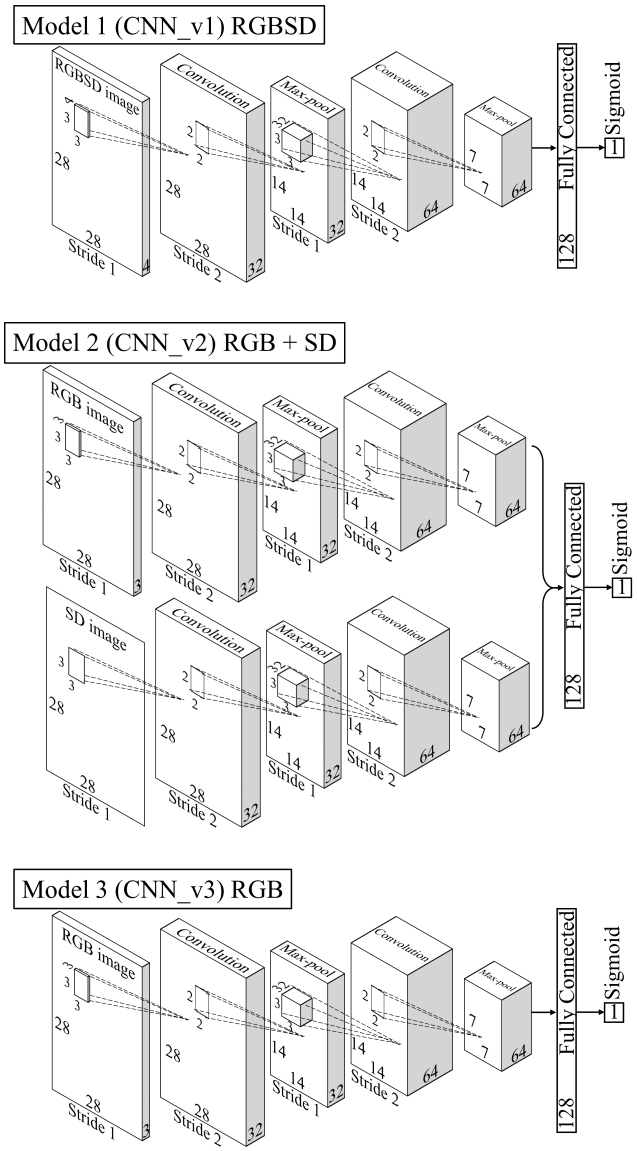


Figure 3.1: Three CNN models: Model 1 takes a 4-channel RGBSD image as input; Model 2 takes an RGB image and an SD image as input; Model 3 takes an RGB image only as the input. The parameters of the CNN in each of the layers are chosen empirically for real-time performance.

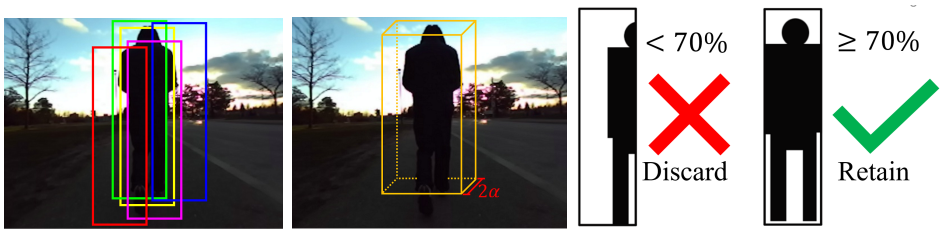
not change dramatically in the first 50 frames (about 2-3 seconds).

Test set selection

Once the CNN classifier is initialized or updated, we use it to detect the target in the next frame. When a new frame is available along with the stereo depth layer, we search the test patches in a local image region as shown in Fig. 3.2(a). We also restrict the search space with respect to the depth as shown in Fig. 3.2(b). If the patches in the image do not have the depth within $previous\_depth \pm \alpha$, we do not consider them (Fig. 3.2(c)), where α is the search region in depth direction (we use $\alpha = 0.25$ meters). By doing this, most of the patches belonging to the background will be filtered out before passing to the CNN classier. Only the highest responses on class-1 will be considered as the target in the current frame. If no target is detected (e.g., highest responses on class-1 < 0.5) after 0.5 sec, it will enter the target missing mode. Then, the whole image is scanned to create a test set.

Update CNN tracker

To update the classier, a new training set needs to be selected. The update step is performed only if the detection step finds the target (class-1) in the test set. In order to maintain robustness, the most recent 50 class-1 patches are retained from the previous frames to form the class-1 patch pool which is implemented as a First-In-First-Out queue. The patches around the target form the class-0 patch pool. In this new training set, we again uniformly select n patches from class-0 patch pool. For selecting n patches from class-



(a) (b) (c)

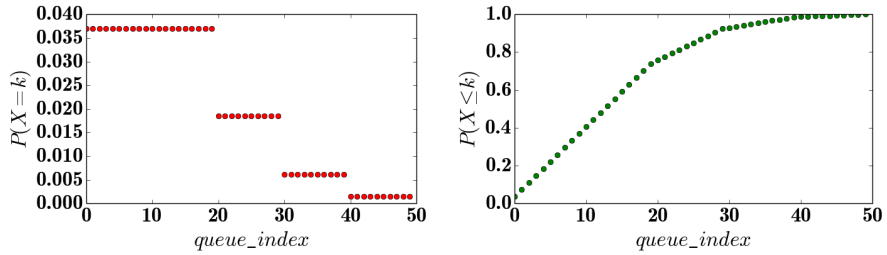
Figure 3.2: 3D search region for test set (a) candidate test patches in 2D region (based on a sliding window approach), (b) search region with respect to depth, (c) pixels in black are within $\pm\alpha$ *meters* from the previous depth. If black pixels are less than 70% of the patch, the patch will be discarded, else, it will be retained. The number 70% is chosen experimentally as this covers the human body completely in most of the cases. According to (c), the red and blue patches in (a) will be discarded, the green, pink, and yellow patches will be retained.

1 patch pool, we sample the patches based on a Poisson distribution with $\lambda = 1.0$ and $k = \lfloor \frac{queue\_index}{10} \rfloor$ (see Equation 3.1 and Fig. 3.3). This gives a higher probability of selecting patches from the recent history rather than selecting older patches. This training set is used to update the classifier. The Poisson distribution based sampling of class-1 patches avoids overfitting and provide a chance to recover from bad detection in the previous frame(s).

$$P(k) = e^{-\lambda}\frac{\lambda^k}{k!} \tag{3.1}$$

3.2.2 Navigation of the Robot

In this section, we describe the navigation aspect of the robot. There are two cases: (*i*) when the robot can see the target (human) in the image; (*ii*) when the robot cannot see the target. A proportional integral deriva-



(a) normalized PMF (b) CDF

Figure 3.3: Poisson distribution with $\lambda = 1.0$ and $k = \lfloor \frac{queue\_index}{10} \rfloor$, where *queue_index* is the patch index in First-In-First-Out queue. To select an index, just randomly generate a real number from 0 to 1.0. Then, base on (b) the CDF graph, an index is selected.

tive (PID) controller [104] is used in the former case while the path of the target is replicated in the latter. A local history of the target poses is maintained to compute the local path of the robot. The robot moves to the last observed pose of the target to find the target and continue the following behaviour. There are 4 basic components involved here: (*i*) Localization of the robot, (*ii*) Target Pose Estimation, (*iii*) Robot following using a PID based controller, and (*iv*) A local path planner (trajectory replication).

Robot Following using PID controller

In this section we describe the robot following behavior for the case when the human can be seen in the image. A pre-specified distance, D is maintained between the robot and the target. The linear velocity, v of the robot is directly proportional to the error in current depth, $(d - D)$, where d is the current depth of the target. The angular velocity, ω is proportional to the error in the x coordinate of the target $(x - X\_mid)$. X_mid is the centre of the image in the horizontal direction. Only the Proportional and Integral

components of the PID controller are used. We use $D = 1.0m$. Following equations detail the velocities as a function of the error terms.

$$v = K_p * (d - D) + K_i * \int_T (d - D)dt; \tag{3.2}$$

$$\omega = K'_p * (x - X\_mid) + K'_i * \int_T (x - X\_mid)dt; \tag{3.3}$$

where K_p, K_i, K'_p, K'_i are the PI constants, $(d-D)$, $(x-X\_mid)$ are the error terms for the linear and angular velocities and dt is the time difference between successive frames.

Localization

Localization of the robot requires estimating the robot pose with respect to a global coordinate frame. In the 2D case, this is x,y coordinates and the orientation, θ of the robot. The robot must maintain an estimate of its pose as it moves in the presence of dynamic obstacles. Here we address localization using wheel odometry. Wheel odometry is reliable for short distances with an error of less than 4% for environments with a smooth surface (e.g., indoor flooring, outdoor pavement, sidewalk, etc.) for our robot (Pioneer3AT). For this work, the robot is tested in university hallways/corridors which often have minimal features or are featureless (blank walls), hence Visual Odometry based approaches [105] do not give accurate localization. Moreover, the environment is dynamic (has humans walking) which makes Visual odometry even less reliable.

For our work, it is only important that the pose of the robot is accurate

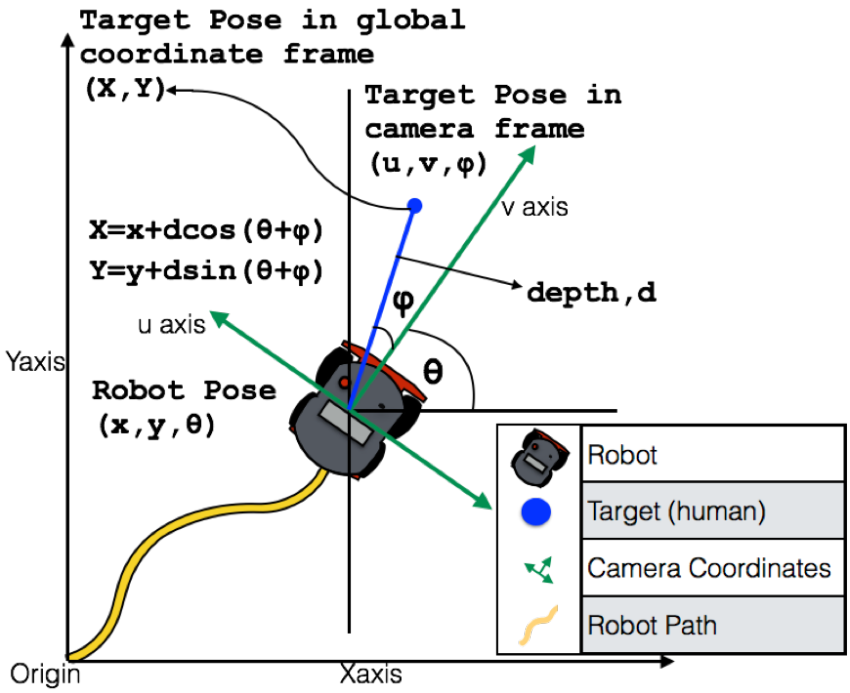
for any short time (e.g., 5 seconds). This is the time we require localization information of the robot to compute the local path of the target and previously accumulated errors due to dead reckoning [17] do not matter.

Target Pose Estimation

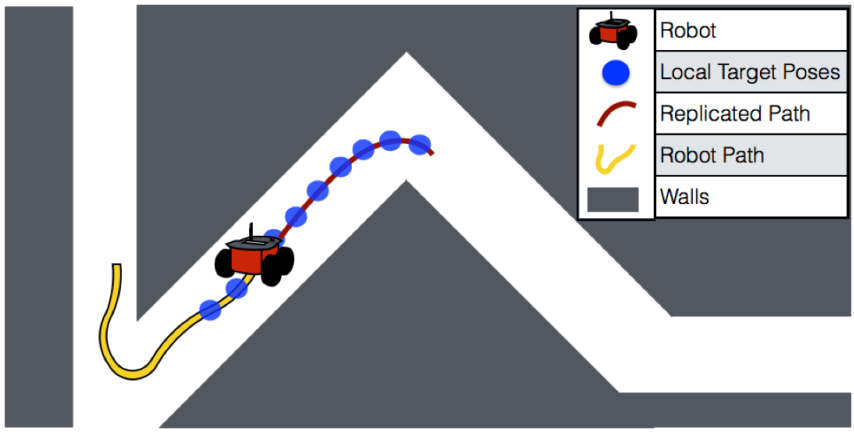
The pose (World coordinates) of the target with respect to the camera frame is estimated using the depth and the focal length of the camera [100]. Knowing the pose of the robot and target pose with respect to the camera frame, the 2D pose of the target can be estimated accurately in a global frame. Fig. 3.4(a) shows the top view for computing target pose.

Trajectory Replication / Path Planner

Here we describe the navigation algorithm used to follow the human when the robot cannot see the human. This part is used when the person is turning around a corner or around a tree in an outdoor context. The robot always keeps a local history of the recent p poses of the target with respect to the global coordinate frame, this is called the recent trajectory of the target (See Fig. 3.4(b)). We use $p = 100$ here. If the robot cannot see the target transiently for 0.5 seconds, it implies that the human turned around a corner or is blocked by something else, so the robot replicates the recent trajectory of the target. By doing so, the robot reaches the last observed pose of the target. After reaching this position, the robot should be able to find the target and resume the following behaviour using the PID based controller. If for some reason the robot cannot find the target after replicating the path, the robot turns on the spot to see if it can find the target, if not the robot



(a)



(b)

Figure 3.4: (a)Estimation of the target pose in the global frame (top view) (b) Local Trajectory of the target poses is stored, when the robot cannot see the target in the image the robot simply replicates the latest local history of target poses stored to find the target. In this work, local history of 100 poses is stored.

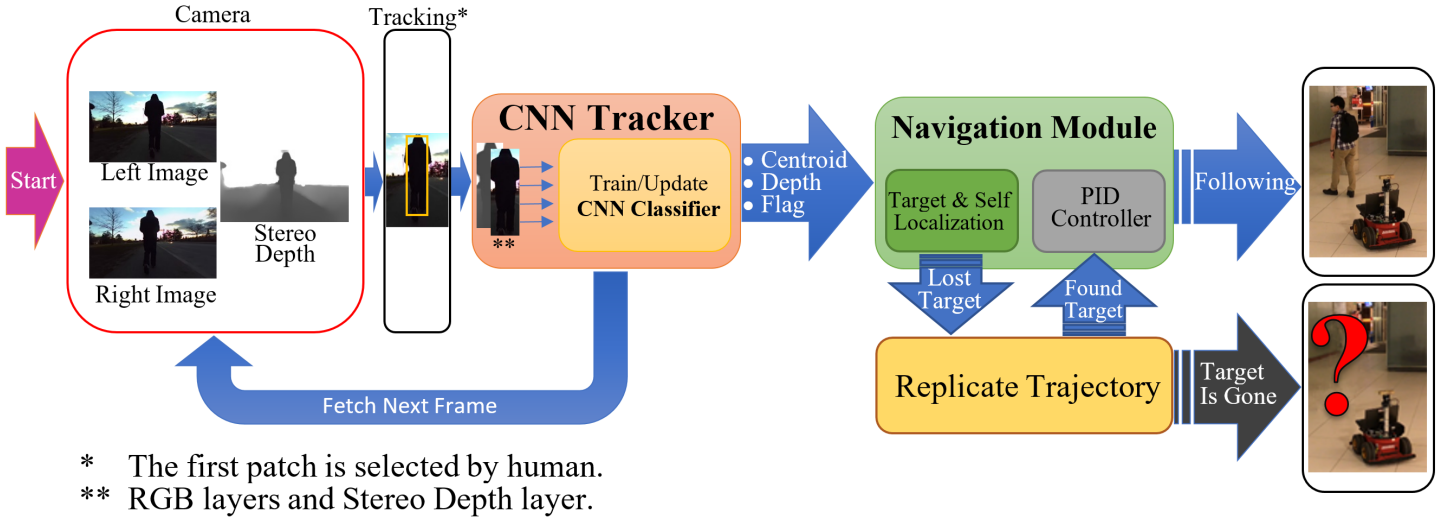


Figure 3.5: Overview of the System Design of our approach.

stops there and the following behaviour terminates. On the other hand, if the robot finds the target while replicating the local path, the robot shifts to the PID based following behavior. Some of the cases when the target might not be found include when target runs away after the turn or turns somewhere else unexpectedly or vanishes due to some reason. In all these cases it is reasonable to assume that the robot would not be able to find the target. A similar behaviour is expected if a human is following another human.

The overview of our proposed approach is described in Fig. 3.5. The input to our system is an RGB image and a computed stereo depth image. These images are then run through an online CNN which runs at a frame rate of 20 fps. The CNN returns the depth, the centroid coordinates of the target being tracked and a flag which indicates the presence/absence of the target. If the target is present in the scene a PID based controller is used to steer in such a way so as to keep the target in the center of the image; in case of absence of the target, the local path of the target is replicated by the robot to continue the following process. We run our robot at speeds up to 1.0 m/s. The Robot Operating System (ROS) was used for integrating the different components in this work. We tested our approach on a Dell Alienware Laptop with Intel core i7, 7th Gen, 2.8 GHz processor and a GTX 1070 mobile graphics card.

3.3 Dataset and Experiments

3.3.1 Dataset

Several Datasets exist for pedestrian detection and tracking[5]. In particular, the Princeton Tracking Benchmark [49] provides a unified RGBD dataset for object tracking which includes various occlusions and some appearance changes. But, each sequence is very short (maximum 900 frames, most of them are under 300 frames). Many other works exist that aim at solving the person following problem, but there is a lack of a standardized dataset which could be used to validate the tracking algorithm used for person following robots. In this work, we built an extensive stereo dataset (left, right, and depth images) of 9 indoor and 2 outdoor sequences. Each sequence has more than 2000 frames and up to approximately 12000 frames. The dataset has challenging sequences which have pose changes, intense illumination changes, appearance changes (target removing/wearing a jacket, exchanging jacket with another person, removing/wearing a backpack or picking-up/putting-down an object), crouching and walking, sitting on a chair and getting up, partial and complete occlusions, occlusions by another person wearing same clothes and some other different situations. The dataset also has image sequences when the target is not visible transiently in the image and reappears after some time. The dataset is built in different indoor and outdoor environments in a university context. Some of the samples from the dataset can be seen in Fig. 3.6. The images are captured at a frame rate of 20 Hz and the resolution is standard VGA (640 x 480) for bumblebee2

[5] `http://homepages.inf.ed.ac.uk/rbf/CVonline/Imagedbase.htm#people`

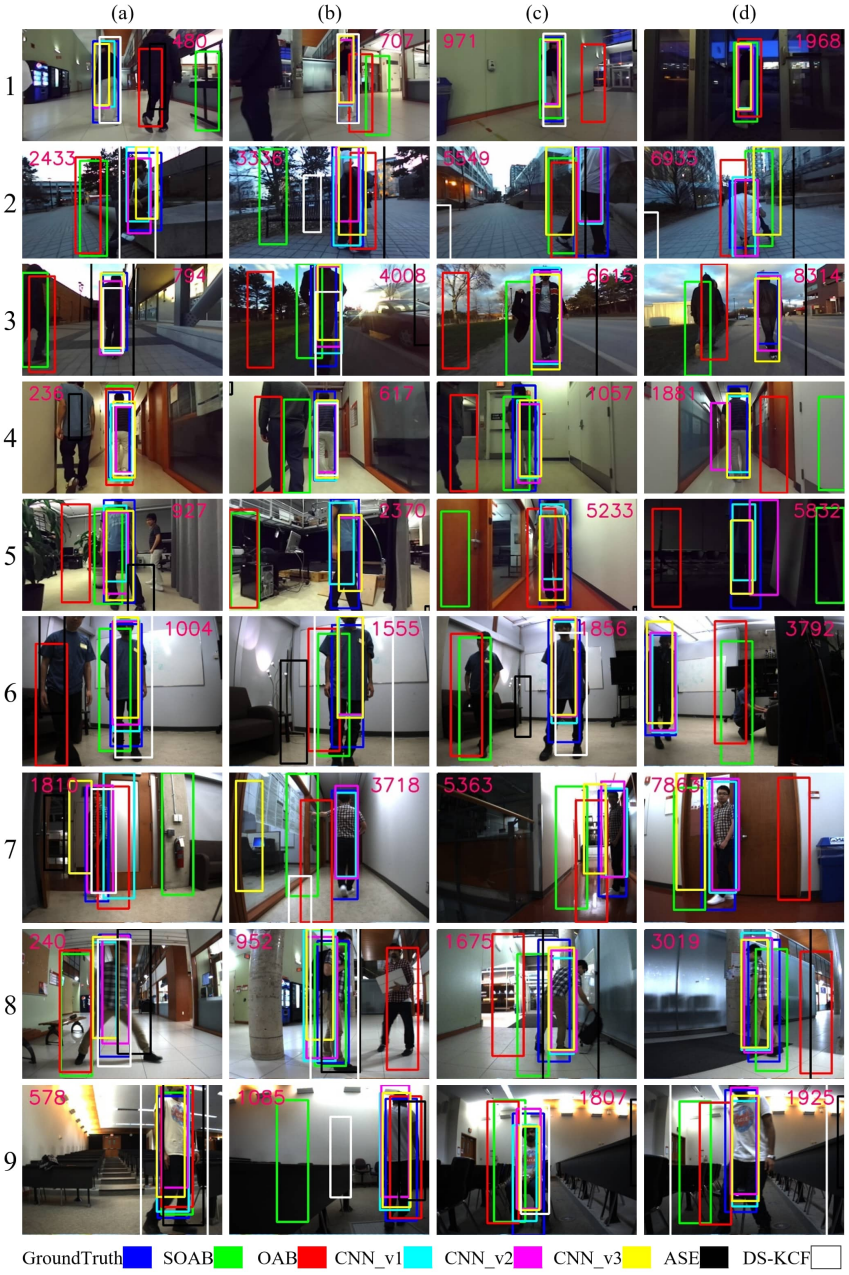


Figure 3.6: Comparison of some tracking algorithms on our dataset. (1): Hallway 2; (2): Walking Outdoor; (3): Sidewalk; (4): Corridor Corners; (5): Lab & Seminar; (6): Same Clothes 1; (7): Long Corridor; (8): Hallway 1; (9): Lecture Hall. (SOAB [1], OAB [2], ASE [3], DS-KCF [4])

and (672 x 376) for ZED. We also provide with ground truth of the image sequences[6]. The ground truth contains the bounding box labeled for the target (human) which is manually labeled by human annotators for each frame.

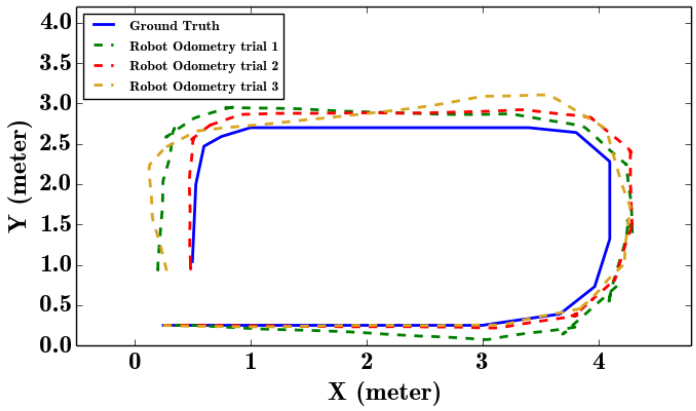
3.3.2 Evaluation Metric

The interest of person following task is to follow a person, so the size of the bounding box is not important for the robot. However, the centroid of the target plays an important role. The evaluation of tracking algorithms has been done in numerous ways. Wu et al. [106] provide details about various existing evaluation metrics that have been used for tracking. For our dataset we use the *precision-plot* as defined in [106] as the metric to evaluate the performance of our approach. We report the percentage of frames in which center of the detected bounding box is within a specific range of pixels from the ground truth (See Figures 3.8, 3.9). Since the initial bounding box size is about (100 x 350) for all the video sequences, we compute the average precision of all sequences using location error threshold 50 pixels to evaluate tracker performance(see Fig. 3.10(a)). Fig. 3.10(b) shows the average precision plot over all sequences from Figures 3.8, 3.9.

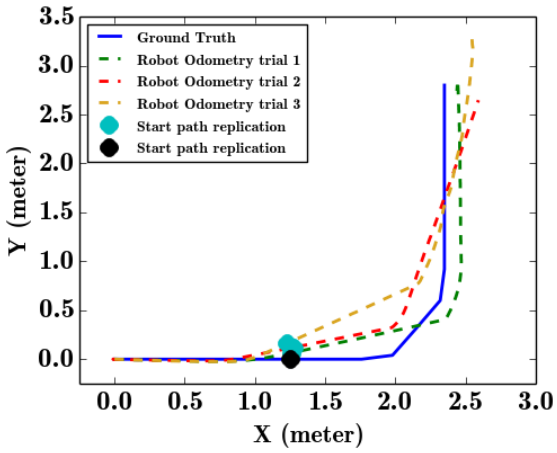
3.3.3 Experiments

We validated our proposed approach in different indoor and outdoor environments. We achieved a frame rate of approximately 20 fps depending on

[6]demo videos and dataset available at `http://jtl.lassonde.yorku.ca/2017/05/person-following-cnn/`

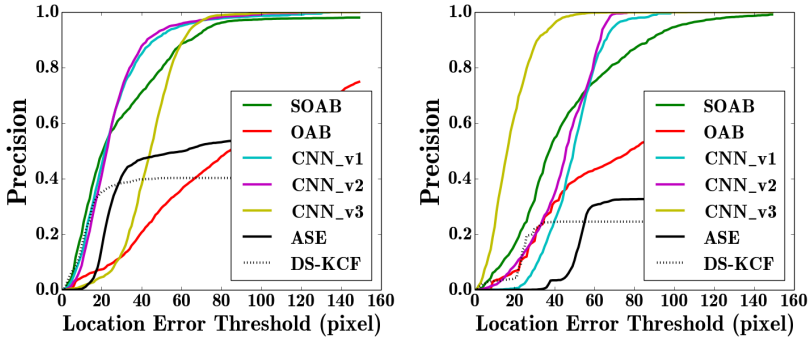


(a) normal person following case



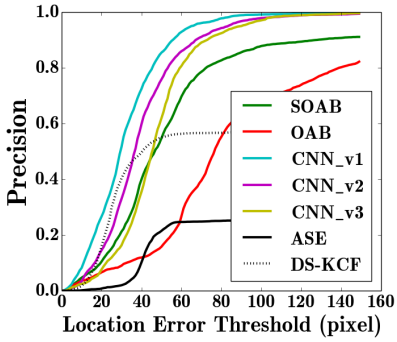
(b) path replication case

Figure 3.7: Overall performance of our robot system. (a) *Ground truth* is the path the robot should have taken ideally maintaining a 1-meter distance from the target. A tape was drawn on the ground on which the robot was supposed to drive, the target was walking at a 1-meter distance from each point on this tape. *Robot Odometry trials* are the robot paths based on wheel odometry. (b) *Ground truth* is the same as the human path we are testing the path replication behaviour here. We have a maximum error (includes tracking, control, and wheel odometry errors) of roughly 30 centimeters which is not high for our task.

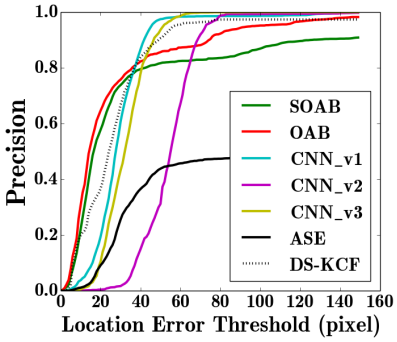


(a) Hallway 1

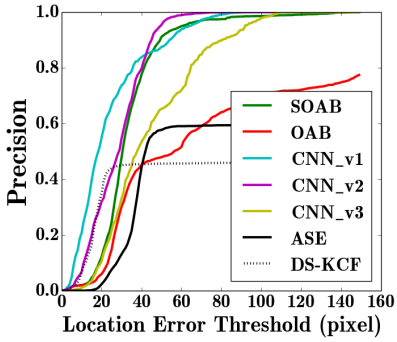
(b) Multi-crossings



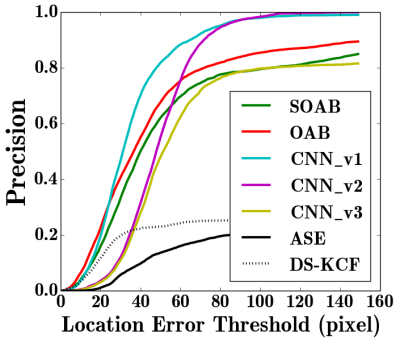
(c) Same Clothes 1



(d) Lecture Hall

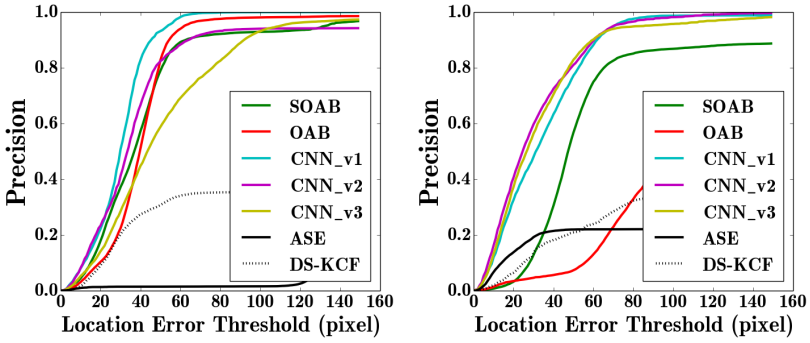


(e) Same Clothes 2



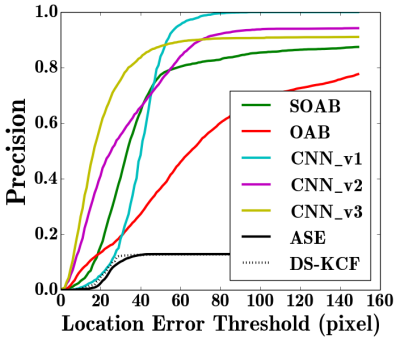
(f) Long Corridor

Figure 3.8: *Precision-plots*: comparison between our trackers and different tracking algorithms, SOAB [1], OAB [2], ASE [3], DS-KCF [4] in 6 different situations

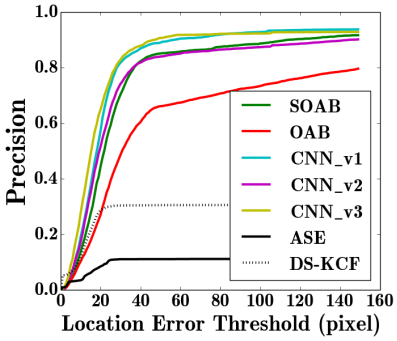


(a) Walking Outdoor

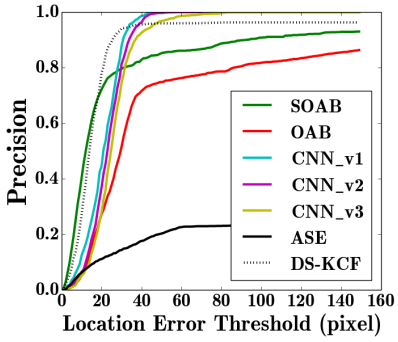
(b) Sidewalk



(c) Lab & Seminar

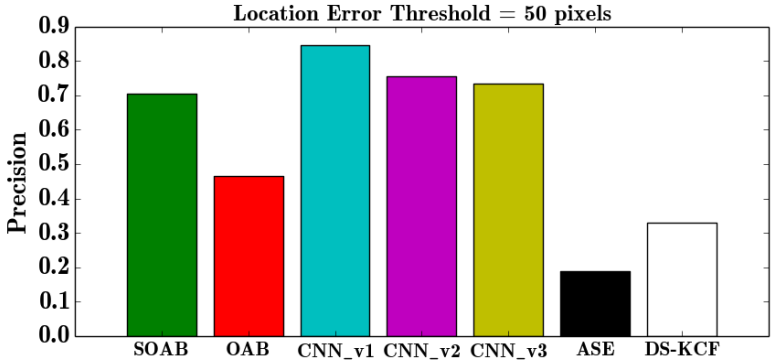


(d) Corridor Corners

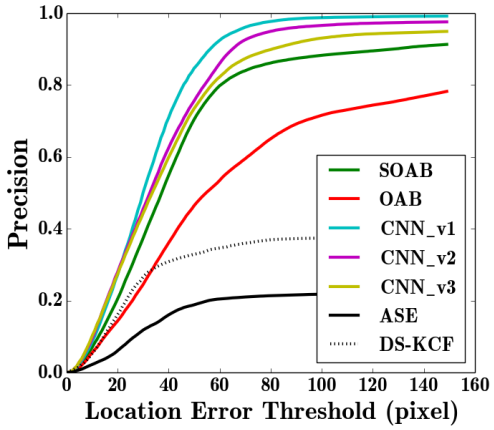


(e) Hallway 2

Figure 3.9: *Precision-plots*: comparison between our trackers and different tracking algorithms, SOAB [1], OAB [2], ASE [3], DS-KCF [4]



(a) Precision at location error threshold 50 pixels



(b) Precision Plot

Figure 3.10: Comparison over 11 sequences (SOAB [1], OAB [2], ASE [3], DS-KCF [4]) in 5 different situations

the search window size that we use for the depth range and the local image search region. For evaluation, we compare three versions of our tracking algorithm with four other existing stereo vision based trackers (for which the code is publicly available). We used the *precision-plot* evaluation metric as defined in [106] to report the performance of our system. The performance can be seen in Fig. 3.6, 3.8, 3.9 and 3.10. We evaluated the performance of our approach on 11 challenging sequences which exhibit varying situations as described in the previous section. It was found that the RGBSD based CNN (CNN_v1) outperformed all other existing approaches. The RGB based CNN (CNN_v3) could not perform better than SOAB [1] in some sequences. We also compare our approach with Martin et al. [3] (ASE with monocular images) and Camplani et al. [4] (DS-KCF with RGBD images). We show the performance of our overall robot system in Fig. 3.7. A demo video of our approach on the robot under different situations can be found at the link [6].

3.4 Summary

In this chapter, we described a robust person following robot system using an online real-time Convolutional Neural Network in the context of robotics. The proposed system was compared with some of the existing stereo vision based trackers and it was shown that our approach outperforms other approaches. Our technique could find the person even when the robot could not see it by replicating the local trajectory of the target being followed. Possible future work includes incorporating dynamic obstacle avoidance techniques

with the person following robot to give it more intelligence. Person following could also be addressed for places with known maps like using a social robot to follow people in a specific house, malls, retail stores and other places.

Chapter 4

Localization in Dynamic Human Environments[1]

4.1 Introduction

Indoor Localization is a primary task for social robots. We are particularly interested in how to solve this problem for a mobile robot using primarily vision sensors. This chapter examines a critical issue related to generalizing approaches for static environments to dynamic ones: (*i*) it considers how to deal with dynamic users in the environment that obscure landmarks that are key to safe navigation, and (*ii*) it considers how standard localization approaches for static environments can be augmented to deal with dynamic agents (e.g., humans). We propose an approach which integrates wheel odometry with stereo visual odometry and perform a global pose re-

[1] this chapter is an extended version of the paper accepted to be published in the 15^{th} Conference on Computer and Robot Vision in [7]

finement to overcome previously accumulated errors due to visual and wheel odometry.

We address the localization task in dynamic human environments using a known 2D occupancy map with dynamic agents moving with unknown trajectories. The map has certain interest points which serve as important landmarks at which a global refinement is performed. Due to this refinement we overcome any previously accumulated errors introduced by visual or wheel odometry. The chapter has detailed empirical analysis to evaluate our approach through a series of controlled experiments to see how localization performance varies with increasing number of dynamic agents present in the scene. For this work, we make use of a standard RGBD sensor (a stereo camera) for environmental sensing and a commercial robot base (Pioneer 3AT).

In this work, we are able to localize the robot with high accuracy in challenging situations (see Figure 4.1) like partial or complete occlusions of the camera view, significant number of dynamic agents present in the scene, robot navigating in a texture-less corridor, robot facing blank walls, etc. A map of the environment is assumed to be known apriori. The map could be a 2D occupancy map or a floor plan of the world in which the the robot operates. As opposed to loop closure techniques for pose refinement where one needs to have visited the place in advance to perform a refinement, in our approach, we do not need to have visited the place before. The major contributions of this chapter are: (i) an approach which can act as a wrapper for traditional localization approaches to handle challenging dynamic situations, (ii) empirical analysis of our approach to see how visual

Figure 4.1: Different situations our approach can localize in (a-b) crowded corridor with 4 people; (c-d) corridor with 2-3 people (e-f) robot moving in a texture-less corridor and motion blurr; (g) robot moving in narrow spaces; (h) camera view occluded

odometry behaves as number of dynamic agents are increased, (*iii*) a dataset in which the number of dynamic agents vary which can be used by others to validate their alternative approaches.

The chapter is structured as follows. Section 4.2 presents the proposed approach. In Section 4.3, we provide detailed empirical results for our work. Section 4.4 provides the conclusion and possible future work.

4.2 Our Approach

Localization of the robot requires estimating where the robot is with respect to a global coordinate frame. The robot must know its pose - in the 2D case this is x,y coordinates and the orientation, θ of the robot in some global coordinate frame. The initial position of the robot is assumed to be known. Now, we describe our proposed localization approach. We enable the robot to maintain an estimate of its pose as it moves in the presence of dynamic obstacles. Dynamic obstacles do not provide any useful information to the robot in terms of localization. Worse, their presence can degrade the quality of localization of the robot as they may obscure some of the visual landmarks required for the localization of the robot. The robot needs to find a way to make use of its wheel odometry and the information it perceives from the stereo camera about its environment to accurately localize itself in the map in the presence of these potentially intermittent visual landmarks.

Visual sensors are known to be very accurate in static environments, however a detailed analysis of their performance in terms of dynamic environments remains open. We propose to use a combination of information

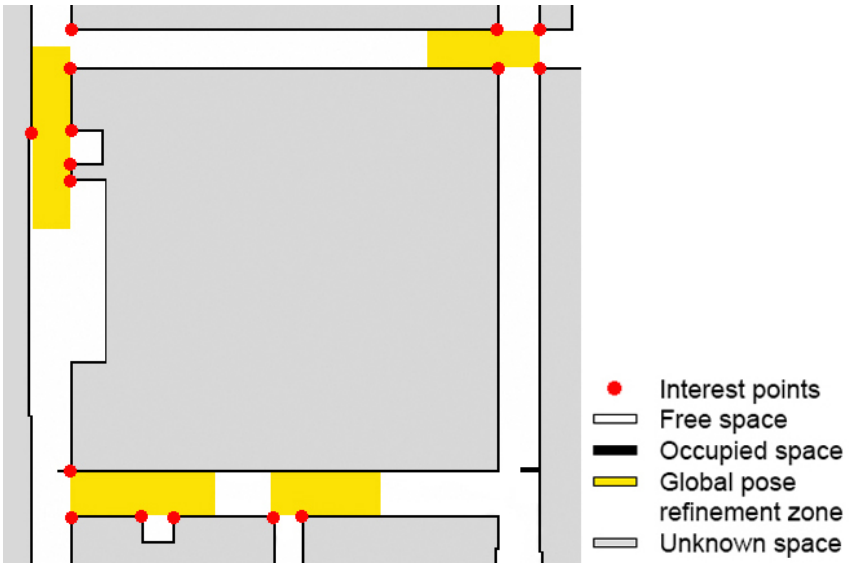


Figure 4.2: An occupancy grid map for the environment we deploy our robots in. Yellow zone is the global pose refinement zone.

obtained from cameras wherever possible and use wheel odometry whenever the camera's current view is obscured by humans or dynamic objects. Wheel odometry is known to perform with good accuracy for short distances as shown in [107] and [62]. This short-term accuracy is leveraged in our approach to integrate with traditional visual odometry approaches. We additionally use a global pose refinement technique to update the pose of the robot with respect to known landmarks in the occupancy map. The input to our approach is a 2D occupancy map/floor plan and a known start point w.r.t. the global map. Mapping is assumed to be known/solved. Now we describe our approach.

4.2.1 Interest Point Detection in the Map

Mapping refers to knowing information about the environment which would help localize the robot with respect to a global coordinate frame. In this work, we use a simple form of map known as Occupancy grids [108]. An occupancy grid is a 2D top down representation of the environment. It divides the given environment into 2D cells and each cell indicates the probability of it being occupied or not. A sample occupancy map we used in our approach can be seen in figure 4.2. Occupancy maps provide valuable information about the geometric structure of the environment. They are similar to floor plans without the semantic annotations in them. From the given occupancy maps, we mark certain points in the map as interest points. These are the points where a global refinement can be performed to accurately localize the robot in the map. In this work we mark these points manually using the occupancy map. Figure 4.2 shows these interest points in a sample map.

The occupancy map in our approach is generated from a SICK tim 551 2D laser scanner [2] (the scanner is used only for initial map creation and not for any subsequent operation of the robot). A sample gmapping package in ROS [3] is used to create the map. After creating occupancy map, it is cleaned manually to remove any inconsistencies in the map. Now we detect interest points where a global refinement is performed during the localization step. Knowing the resolution of the map to be 5 cm for one pixel, we get the coordinates of each of the marked interest points in the map. These interest points serve as candidate landmarks which if detected successfully will

[2] `http://wiki.ros.org/sick_tim`
[3] `http://wiki.ros.org/gmapping`

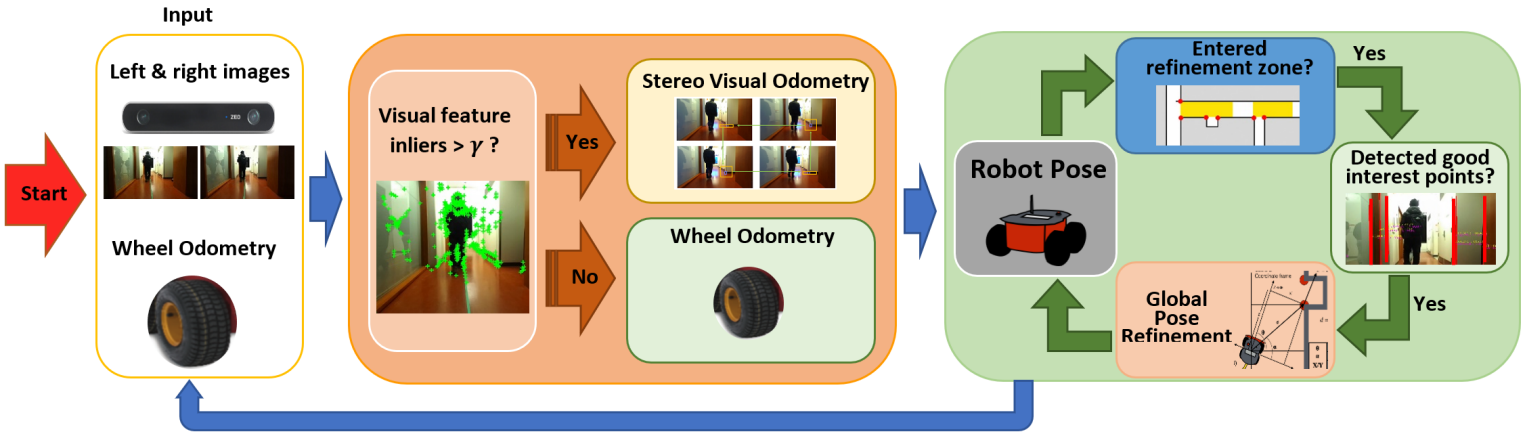


Figure 4.3: Overview of our proposed localization approach.

improve the quality of localization and remove error accumulation. Similar interest point detection can also be manually done easily using a 2D floor plan of the building.

4.2.2 Localization in the Presence of Dynamic Obstacles

Our approach is a hybrid approach using wheel encoders, visual odometry and a global pose refinement scheme to overcome previous accumulated errors in visual/wheel odometry. Figure 4.3 provides a basic overview of our approach. Now, we describe the 3 basic components involved in the Localization phase namely: (*i*) Visual Odometry by tracking features, (*ii*) Wheel Odometry using Shaft encoders, and (*iii*) Global Pose Refinement using Known map. Each of these components are described below:

Visual Odometry by tracking features

The visual odometry component in our approach is same as that of Geiger et al. [5]. Features are extracted and then tracked to estimate ego-motion. In [5], features are matched within a set of 4 images: current left image, current right image, previous left image and the previous right image. In order to find stable feature locations, the input images are initially filtered with 5x5 blob and corner masks. Next non-maximum and non-minimum suppression is applied resulting in features belonging to one of the 4 classes (i.e. blob max, blob min, corner max, corner min). Features are matched only between these 4 classes. Features are matched in a circle to be qualified as a successful match. We extract features from the current left image, match it with the best point in the previous left image within a MxM search window, then

in the previous right image, then the current right image and finally in the current left image again. A feature point gets accepted only if the last feature point co-incides with the first one.

A RANSAC based approach is used to estimate the transformation matrix $T = (r, t)$ which is the transformation (rotation, r and translation, t) between two subsequent images. The number of feature matches and the percentage of inliers here play a crucial role. Based on the number of matches and inliers percentage, we make use of wheel odometry when the inliers percentage is not promising enough.

Wheel Odometry Integration using shaft encoders

From the previous visual odometry component, if the percentage of inliers obtained is less than a threshold, γ, this means that the visual odometry component estimated the r, t matrices with fewer feature matches. This could happen due to lack of sufficiently good static features, tracking a dynamic consistent set of patches from a human, etc. In such cases, we rely on wheel odometry to transiently update the pose of the robot. Cases when visual odometry would not provide us with a sufficient number of feature inliers include when the robot is facing a blank featureless wall, too many moving people in front of the camera, limiting visibility of static content, motion blur, low quality of features detected, etc. In all such circumstances, we estimate and update the motion using wheel odometry. Say the robot at time, t was at position, p and upto time $t + \delta$T visual odometry cannot be relied on. So the motion of the robot during δT is computed using wheel odometry.

Using wheel odometry, we get the pose of the robot at each time instance in the form of position $P(x, y, z)$ and orientation $Q(x, y, z, w)$ in quaternion form. This is converted to a transformation matrix, T (consisting of rotation, r and translation, t) of size 4x4. Say the transformation matrix at time t_1 is given by R_{t1} and at t_2 by R_{t2} so the motion during $t_2 - t_1$ is given by $(R_{t1})^{-1} * R_{t2}$. This motion is then used to update the pose obtained from visual odometry.

It should also be noted that a standard inertial measurement unit (IMU) can also be used instead of shaft encoders in wheel odometry.

Global Pose Refinement using Known Map

This step is used to update pose of the robot whenever the robot is near known landmarks/interest points. Interest points are unique points in the occupancy maps which the robot can use to refine its pose and reduce any previously accumulated errors in the pose estimation process. The global refinement component is only run when the robot's pose obtained from the integration of the visual and wheel odometry is within a predefined range. These ranges of robot poses form zones in which this component is run. An example of refinement zones and interest points can be seen in figures 4.2, 4.4.

The interest points are typically at the intersection of two perpendicular walls, but could also be at the intersection of two walls at an angle or a pillar. These interest points in the occupancy map are straight lines perpendicular to the ground when observed from a camera's view. Figure 4.4 shows a correspondence between interest points on the occupancy map

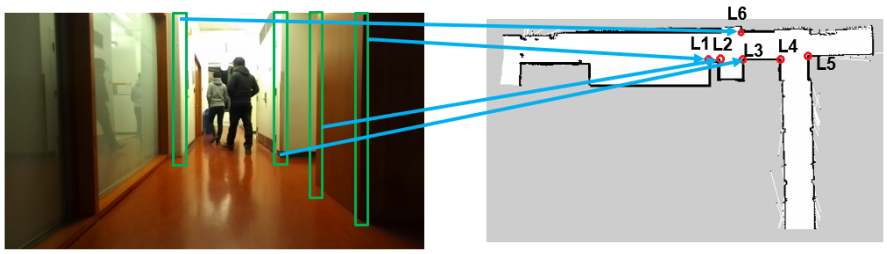


Figure 4.4: Interest Points detection from camera view and corresponding match in the occupancy map

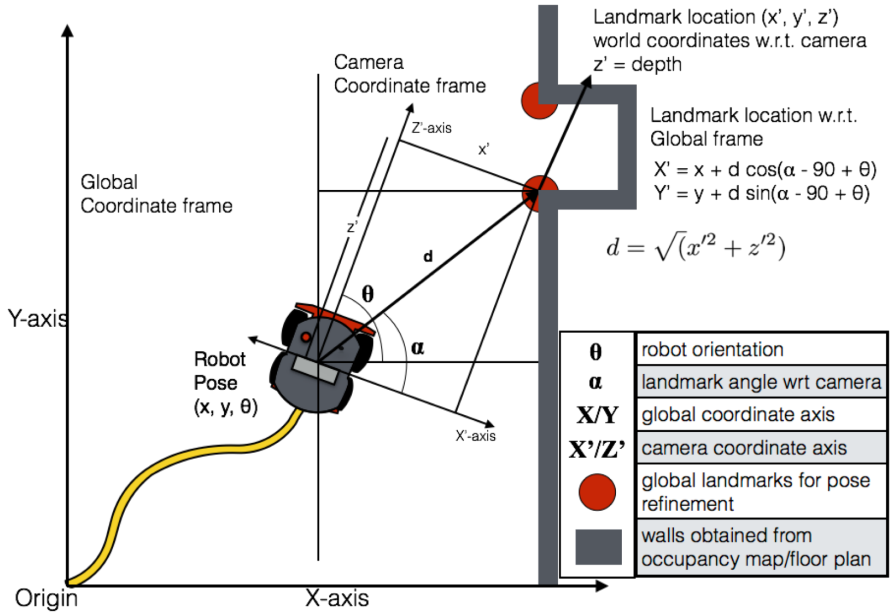


Figure 4.5: Estimation of the predicted landmark location (Robot pose + World Coordinates of feature point w.r.t. camera)

and a camera view. To detect these interest points we need to detect points in the highlighted regions in Figure 4.4. Now, we explain the process of detection of points on the specific landmarks. As these points lie on vertical lines, we need to detect points on these lines. First, we filter the images with an oriented gabor filter at 90 degrees to detect vertical edges/lines. Gabor Filters have been widely used for texture analysis, feature extraction, disparity estimation, etc. These filters are special types of filters which only allow a certain band of frequencies to pass through and reject the others. Now, we get only vertical edges from the image. Next, we employ a Line Segment Detector (LSD) [109] on the filtered image. After doing this, we retain the feature points on vertical edges only. We only detect lines greater than a specific length and at an angle of approx. 90 degrees.

Now, we have a set of n interest points, $P = P_1, P_2, ..P_n$ from the image. Each point belongs to a vertical edge. Knowing the depth, focal length and base line, we can compute the world coordinates of each feature point in the camera coordinate system [93]. Knowing the pose of the robot obtained from visual and wheel odometry in the global coordinate frame, we can compute the global coordinates of each point detected as shown in Figure 4.5. Now, we have a set of n global world coordinates of the interest points P; lets call these transformed points as $W = W_1, W_2, ..W_n$, where $W_i = (x_{w_i}, y_{w_i}, z_{w_i})$. Since its a ground robot (2D case) we only care about the x and y coordinates. For the interest points as shown in Figure 4.2, say each of these landmarks/interest points, $L = L_1, L_2...L_m$ have world coordinates as $L_j = (x_{l_j}, y_{l_j})$. We know these location of the landmarks as we have the ground truth occupancy map, so we can estimate

the absolute values of these landmarks with respect to the start position of the robot. From the set W, we find the closest point, P_i for each of the landmarks, L_j based on the distance metric $\sqrt{(x_{l_i} - x_{w_i})^2 + (y_{l_i} - y_{w_i})^2}$. Now, we have m points which are closest to each of the landmarks. We have the distance error metric for each of these points to landmark assignment. Let the error in distances be $E = (e_1, e_2, ...e_m)$. From the given map, we make a set of pairs of landmarks that are adjacent to each other. Figure 4.2 shows 16 landmarks and 4 zones, so we make the pairs in each zone, e.g., $(L1, L2); (L2, L3); (L3, L6); (L4, L5)$ as in Figure 4.4 depending on the distance between 2 landmarks in a particular zone. Now, for each pair (L_i, L_j), we compute the quality of the matched point's distance as (e_i, e_j). If both e_i and e_j are less than an empirically determined threshold, β then we consider that as a good pair and the corresponding matched points as good matches. Now, we update the absolute robot pose based on these two landmarks using triangulation [110]. Doing the update at this stage gets rid of any previously accumulated errors due to wheel and visual odometry. Algorithm 2 formulates this.

4.3 Empirical System Performance

In this section, we describe our generated dataset and provide a detailed analysis of our results. Our algorithm was deployed on a mobile robot in a real world environment in a university corridor. To validate our proposed approach we developed a dataset for the purposes of localization of mobile robots in dynamic environments. We first describe our generated dataset

Algorithm 2 Pseudocode for Global Pose Refinement

`Input:`

- Set of n Key Points' world coordinates w.r.t. camera frame, $P_c = \{p_{c_1}, p_{c_2}, ..., p_{c_n}\}$; $p_{c_i} = (x_{p_i}, y_{p_i}, z_{p_i})$
- Set of landmark coordinates, $L = \{L_1, ..., L_m\}$; $L_i = (x_{l_j}, y_{l_j})$
- Pairs of adjacent Landmarks in zone k, $L_{k_{pairs}} = \{(L_1, L_2), (L_2, L_3)...(L_i, L_j)\}$
- Zone number, k
- Empirically determined threshold, β

`Output:`

Refinement succeeded or not
Refined robot pose, $R : (x_{refined}, y_{refined}, \theta_{refined})$

`Procedure 1, Global Pose Refinement:`

1. $W = GlobalCoordinatesOfPoints\,(P)$
2. $C = (C_{L1}, C_{L2}, ..C_{Lm})$, set of closest points to landmarks
3. $E = (e_{L1}, e_{L2}, ..e_{Lm})$, errors of closest points to landmarks
4. **for** $L_i \in L$ **do**
5. $\quad min = \inf$
6. $\quad$ **for** $W_j \in W$ **do**
7. $\quad\quad e_i = \sqrt{(x_{l_i} - x_{w_i})^2 + (y_{l_i} - y_{w_i})^2}$
8. $\quad\quad$ **if** $e_i < min$
9. $\quad\quad\quad min = e_i$
10. $\quad\quad\quad C_{Li} = W_j$
11. **for** $(L_i, L_j) \in L_{k_{pairs}}$ **do**
12. $\quad$ **if** $e_i < \beta \;\&\; e_j < \beta$
13. $\quad\quad$ **update pose wrt to** L_i**,**L_j **using triangulation**
14. $\quad$ **else**
15. $\quad\quad$ **do not update robot pose**
16. **return** *robotpose*

`Procedure 2, Global Coordinate of Point (`P`):`

1. **for** $p_{c_i} \in P_c$ **do**
2. $\quad W_k = GlobalCoordinates(P_i)$ using Figure 4.5
3. **return** $W = W_1, W_2, .., W_n; W_i = (x_{w_i}, y_{w_i}, z_{w_i})$

and later describe the localization results we obtained. The number of dynamic agents in the scene are varied and an empirical performance analysis is reported.

4.3.1 The Dataset

Several datasets exist for computing the localization of a mobile platform equipped with vision sensors. Strum et al. [111] built an RGB-D dataset in indoor environments (industrial hall and office scene) to evaluate visual odometry where they generated ground truth from motion capture systems. Their dataset was built using a handheld Kinect sensor in indoor environments, which for most of the sequences have no presence of humans/dynamic agents or are sparsely populated by one or two people. Smith et al. [112] built a SLAM dataset using a laser, stereo and omni directional cameras in a university environment outdoors. Their dataset was built while the robot was driving several kilometers through a park and university campus. It was built using a segway robot equipped with the sensors like IMU, GPS, stereo, omni-directional, panaromic cameras and Lasers. This dataset also does not have a lot of humans/dynamic agents moving in the environment. One of the most famous benchmarks for ego-motion estimation in outdoor environments (for autonomous driving) is the KITTI dataset [64] which is also sparsely dynamic. The dataset is primarily used for benchmarking various computer vision tasks in the context of autonomous driving. As there is not a dataset having a high number of dynamic agents in the scene, we built a new dataset to validate our approach.

Now, we describe our dataset to address the shortcomings of existing

datasets. We build a dataset which has many dynamic agents (humans) navigating in the scene in an indoor office-like corridor of size 18m x 18m. Our dataset was built using a mobile ground robot in a university environment on the 3rd floor of the lassonde building at York University, Toronto, Canada. The dataset was created using a Pioneer 3AT robot using on board stereo vision sensors (ZED stereo camera) with wheel odometry. While building this dataset, the robot was driven manually to evaluate the localization framework. Our dataset consists of wheel odometry information obtained from the mobile base, stereo image pairs and a depth image from a ZED stereo camera. The images were captured with a 720p resolution (1024 x 720) RGB stereo camera at a frame rate of 30 fps. The camera was mounted on the robot at a height of 76 cm above the ground plane. Images in the dataset were taken indoors during night time in the winter season (January 2018). We created 5 different types of sequences:

- *Type 1* is the situation of with no dynamic agent present in the scene, only the static scene.
- *Type 2* indicates the situation where there is only one person in the environment.
- *Type 3* implies presence of one or two .
- *Type 4* implies presence of at most 3 people.
- *Type 5* implies presence of at most 4 people.

Each situation differs from the other in terms of the number of dynamic agents and pose changes. The data acquisition phase was spread over a

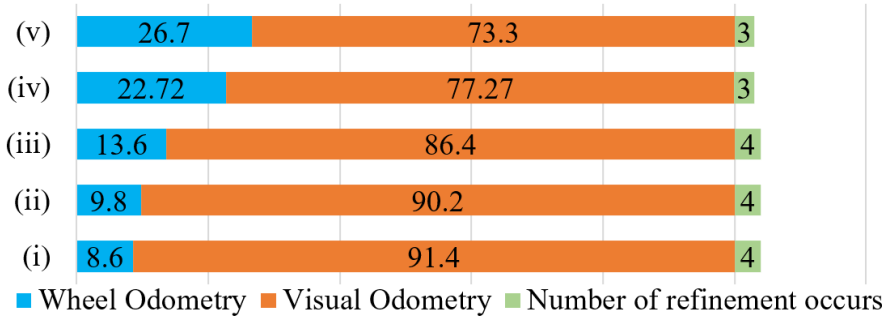


Figure 4.6: An analysis about the percentage of times wheel odometry is used and when visual odometry can be relied on. Experiments performed in a university corridor. (i): Static environment without any dynamic agents, (ii): Dynamic environment with at most one person in the scene, (iii) Dynamic environment with at most 2 people in the scene, (iv) Dynamic Environment with at most 3 people in the scene, and (v) Scene with at most 4 people present.

week. Each sequence has 6000-8000 images. Some sample sequences from our dataset can be seen in Figure 4.1. We make the dataset and demo video publicly available for download at the project web-page[4]. The ground truth, map coordinates and interest points coordinates are also available at the project page. Ground truth pose of the robot was generated by manually driving the robot on a path pre-defined by a marking tape on the floor. The coordinates on this path were known and were measured manually with a measuring tape.

4.3.2 Results

We validate our approach through a set of controlled experiments to have a quantitative analysis using our dataset. We show how performance varies as the number of dynamic agents present in the scene are changed.

[4] `http://jtl.lassonde.yorku.ca/2018/03/localization-among-humans/`

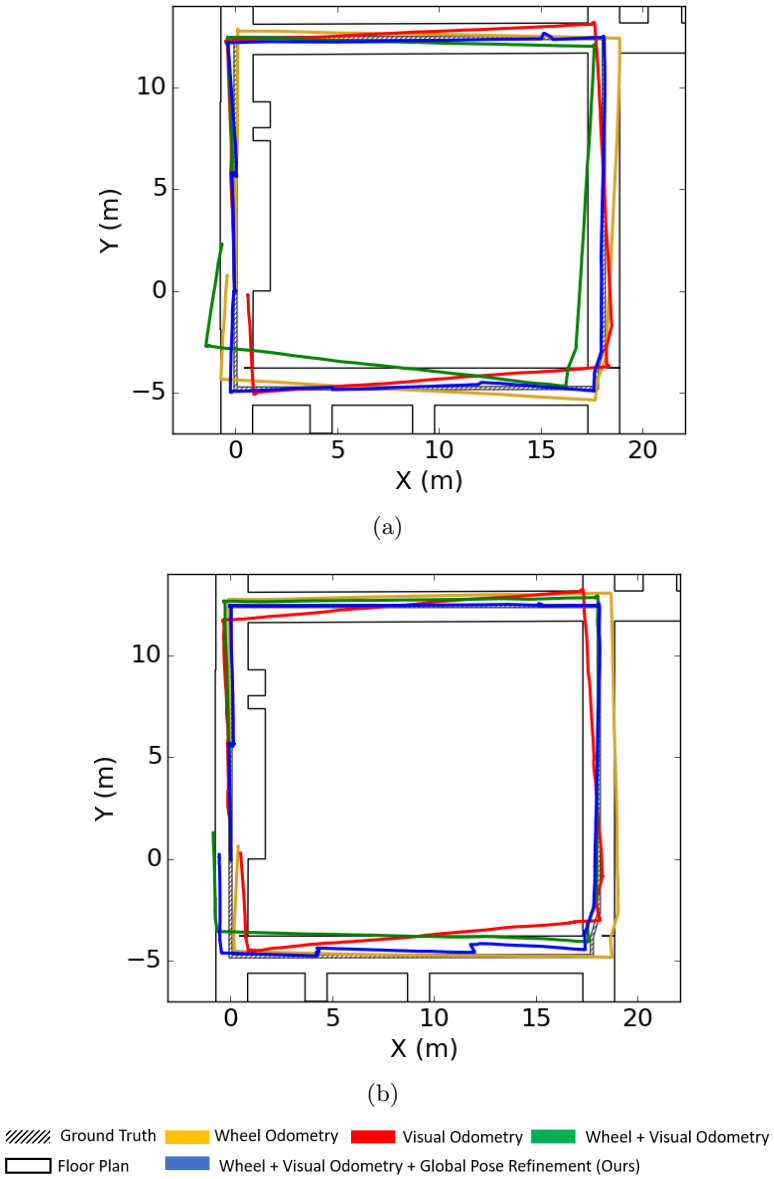


Figure 4.7: Trajectory of our approach against *(i)* wheel odomery, *(ii)* visual odometry (method proposed in [5]), *(iii)* visual + wheel odometry, *(iv)* visual + wheel odometry + global-refinement, and *(v)* ground truth (a) Type 1 (no people), (b) Type 2 (one person)

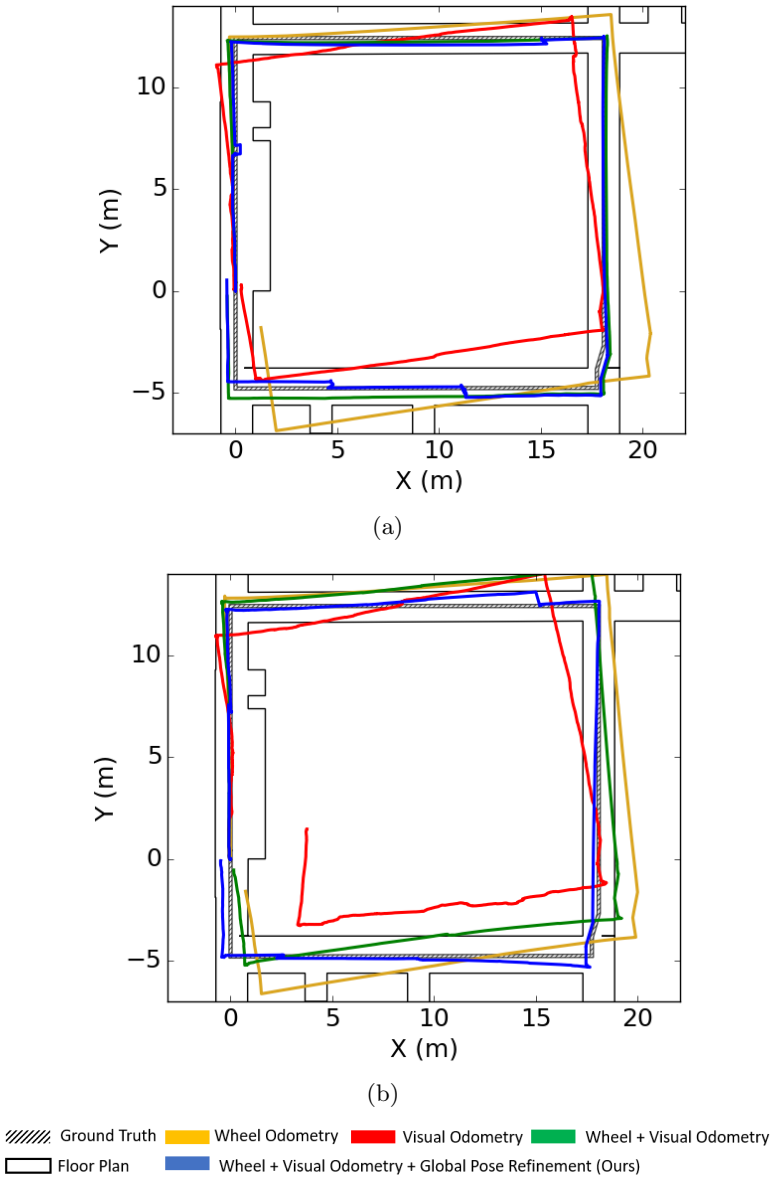


Figure 4.8: Trajectory of our approach against *(i)* wheel odomery, *(ii)* visual odometry (method proposed in [5]), *(iii)* visual + wheel odometry, *(iv)* visual + wheel odometry + global-refinement, and *(v)* ground truth (a) Type 3 (2 people), (b) Type 4 (3 people)

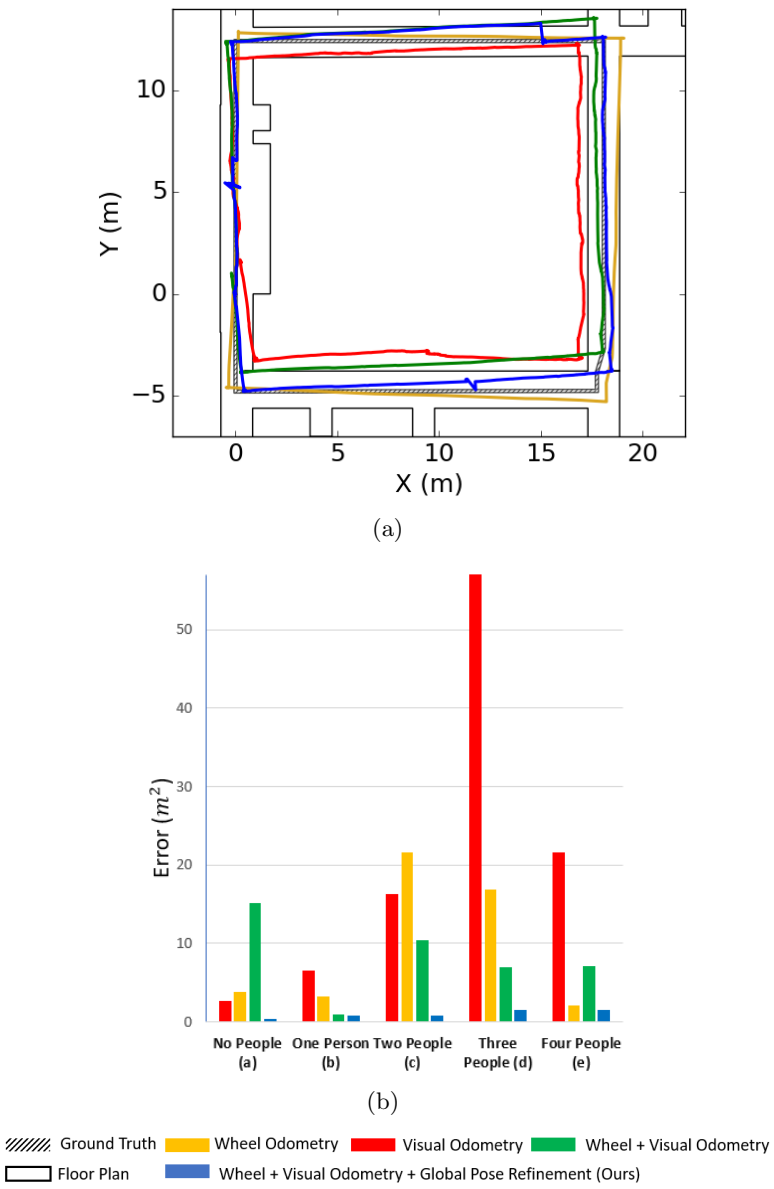


Figure 4.9: Trajectory of our approach against *(i)* wheel odomery, *(ii)* visual odometry (method proposed in [5]), *(iii)* visual + wheel odometry, *(iv)* visual + wheel odometry + global-refinement, and *(v)* ground truth (a) Type 5 (four people), (b) sum of squared errors of 4 corners and the terminal point of the trajectory with the ground truth.

We compute the localization errors of the robot in the presence of dynamic obstacles and compare it to that when the robot moves in the static environment and with the ground truth. We report the performance of our approach on 5 different sequences in our generated dataset. The sequences differ in the number of humans present in the scene. Varying the number of dynamic agents in the scene implies varying the number of dynamic and static visual features present in the environment. As the number of dynamic agents increases the number of static visual features decreases and robot may not be able to trust its vision for estimating its pose, hence in such cases wheel odometry comes to our rescue. Wheel odometry is transiently relied on under such circumstances. On the other hand, with no dynamic agents present in the scene, the dependence of the robot on visual odometry is maximum and wheel odometry is minimally used. Some of the situations where wheel odometry is solely relied on include when a particular person blocks the view of the camera, too many moving agents in front of the camera limiting visibility of static content, robot facing a blank featureless wall/door, motion blur, etc. Figure 4.6 shows the proportion of times when wheel and visual odometry is relied on under varying dynamic agents. To avoid accumulation of errors, we do a global pose refinement based on landmarks from the 2D map. As opposed to traditional loop closure techniques, we do not need to visit the place once to perform a refinement. Knowing the map and a few interest points, the robot knows when to perform a refinement. Our approach runs at 25 fps in real time.

We report the trajectory that the robot takes based on its visual odometry and compare it to the following: (*i*) Wheel Odometry alone, (*ii*) Vi-

sual Odometry alone, (*iii*) Wheel+Visual Odometry, (*iv*) Our Approach (Wheel+Visual+Global-Refinement), (*v*) Ground Truth. Figures 4.7, 4.8, 4.9 show the trajectory under each of the approaches, it can be seen that our approach performs better than visual or wheel odometry alone. Ground truth was generated by driving the robot on a predefined path (the coordinates of which were known $\pm 7.5cm$). For generating the ground truth, a predefined path was created by using a marking tape on the floor. The robot was manually driven on this path as closely as possible. An error of $\pm 7.5cm$ could be there due to inaccuracy introduced due to manual driving of the robot. Due to global refinement we correct the pose and remove any accumulation of error due to both wheel and visual odometry which gives us a better trajectory closer to the ground truth. As can be seen, as the number of dynamic agents are increased quality of traditional visual odometry approach reduces, however using our approach we maintains a good alignment with the ground truth.

4.4 Summary

In this chapter, we presented an approach as to how standard localization techniques can be extended to deal with dynamic agents present in the scene. One of the existing localization algorithms was chosen and integrated with our proposed refinements. An empirical analysis was performed to see how the task of localization differs in a static environment to that of a dynamic environment as number of people in the scene are increased. Some of the possible future works include integrating this approach with a navigation

approach to have an autonomous agent navigating among humans. In this work, only one of the current localization approach was built on top of. Our proposed additions to the localization framework can also be applied to other localization techniques and an analysis can be done on the performance of other existing algorithms as to how they perform in a dynamic context after incorporating our integrations.

Chapter 5

Discussions and Conclusions

5.1 Summary

In this thesis we presented three novel approaches for two key components involved in the navigation of autonomous robots. Two approaches were presented for Person Following robots and one for localization of robots in dynamic human environments. Each of the presented approaches was evaluated extensively and it was shown that our approaches perform better than current approaches. In chapter 2, we used an existing appoarch of online ada-boosting and proposed a modification to the approach which we called SOAB (Selected Online Ada-Boosting) by using depth as a gate to decide when to update the classifier. This neat trick allowed us to handle challenging situations and perform the tracking in a robust manner. Our approach SOAB used Haar wavelet features to learn the model for the target. To further improve the quality of tracking we used a convolutional neural network in Chapter 3 which learns a better feature representation of the target

and performs even better than SOAB. A similar depth trick was used to update the CNN model to handle dynamically changing appearance of the target. Each of these approaches were empirically compared with existing approaches and were able to handle more challenging situations than what the current person following robots are able to address. Finally in Chapter 4, we proposed a localization approach which enables a mobile robot to estimate the pose in dynamic human environments. We proposed a wrapper based approach which can be used with traditional visual odometry algorithms to enhance the performance of these algorithms in dynamic scenes. An empirical evaluation was reported which showed how the performance of our system varies with increasing number of dynamic agents present in the scene. In conclusion our proposed approaches are robust, comparable to or better than the current state of the art approaches. We also tested all the approaches proposed in real world environments and ran everything in real time. Videos for each of the approaches are avalailable at the respective project pages shown in each chapter.

5.2 Future Work

As discussed in the motivation Section 1.1 of the thesis an autonomous agent needs to address a number of tasks including place recognition, localization, navigation, person following, mapping, and SLAM. Some of these components are connected with each other in a direct way. One component uses another to perform a bigger goal. For example to perform autonomous navigation a robot must know its pose at each instant, hence localization is

a significant component for robot navigation. Now we discuss some of the future work that is planned using the components proposed in this thesis.

Autonomous Navigation: An immediate future work arising from the proposed thesis would be integrating the localization based approach with a navigation approach to avoid dynamic agents present in the scene. The robot can integrate the localization component with a path planner to do intelligent planning and navigation from a given start point to a known goal localization using a 2D occupancy map. A similar analysis as was done in the localization chapter of this thesis would be performed with respect to the navigation component. The number of dynamic agents would be varied and it would be analysed how the performance varies with increasing number of dynamic agents present in the scene.

Localization in Dynamic Environments: In chapter 4, we showed how traditional localization approaches can be extended to handle dynamic agents present in the scene. We integrated our approach with only one of the existing Stereo Visual Odometry algorithms [5], some of the future work involves integrating our approach with a few other visual odometry approaches and report the performance of the integrated approaches. This would be an interesting task as it would provide more empirical evidence for the validation of our proposed wrapper based approach to visual odometry techniques. We performed localization in a crowded indoor office-like university corridor environment of size 18m by 18m. Some future work involves testing this approach in bigger and different environments like university hallways, hotels, etc. Increasing the number of landmarks obtained from the occupancy map or the floor plan to possibly increase the accuracy

of pose estimation is another future work which needs more experimental analysis.

Person Following Robots: The Person Following approaches could be extended to make use of the map of the environment to follow the target even when the target disappears after a corner by using information obtained from the map of the environment. The robot can employ searching algorithms to scan the map and find the given target. Knowing the map, the robot could compute the path it should take where it might find the lost target. Since the model of the target is already learnt from the approach proposed in thesis, as soon as the target appears in the camera view the robot would resume the following behaviour. While doing the searching for the target, the robot would also need to avoid dynamic agents present in the scene for navigation. The robot should know its pose while searching for the given target. Our proposed localization approach would be directly employed in this context.

Bibliography

[1] B. X. Chen, R. Sahdev, and J. K. Tsotsos, "Person following robot using selected online ada-boosting with stereo camera," in *14th Conference on Computer and Robot Vision (CRV)*, pp. 48–55, IEEE, 2017.

[2] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting.," in *Proceedings of the British Machine Vision Conference, Edinburgh*, pp. 47–56, 2006.

[3] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *British Machine Vision Conference, Nottingham, September 1-5, 2014*, BMVA Press, 2014.

[4] M. Camplani, S. L. Hannuna, M. Mirmehdi, D. Damen, A. Paiement, L. Tao, and T. Burghardt, "Real-time RGB-D tracking with depth scaling kernelised correlation filters and occlusion handling.," in *British Machine Vision Conference, Swansea, UK, September 7-10, 2015*, BMVA Press, 2015.

[5] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *Intelligent Vehicles Symposium (IV), 2011*

IEEE, pp. 963–968, Ieee, 2011.

[6] B. X. Chen, R. Sahdev, and J. K. Tsotsos, "Integrating stereo vision with a cnn tracker for a person-following robot," in *International Conference on Computer Vision Systems*, pp. 300–313, Springer, 2017.

[7] R. Sahdev, B. X. Chen, and J. K. Tsotsos, "Indoor localization in dynamic human environments using visual odometry and global pose refinement," in *15th Conference on Computer and Robot Vision (CRV)*, IEEE, 2018.

[8] D. Fox, W. Burgard, F. Dellaert, and S. Thrun, "Monte carlo localization," in *Proceedings of the 1999 16th National Conference on Artificial Intelligence (AAAI-99)*, AAAI, 1999.

[9] R. Sahdev and J. K. Tsotsos, "Indoor place recognition system for localization of mobile robots," in *13th Conference on Computer and Robot Vision (CRV)*, pp. 53–60, IEEE, 2016.

[10] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of human–robot cooperation," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 335–356, 2015.

[11] R. Sahdev, "Free space estimation using occupancy grids and dynamic object detection," *arXiv preprint arXiv:1708.04989*, 2017.

[12] M. Labbe and F. Michaud, "Online global loop closure detection for large-scale multi-session graph-based slam," in *2014 IEEE/RSJ Inter-*

national Conference on Intelligent Robots and Systems (IROS 2014), pp. 2661–2666, IEEE, 2014.

[13] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[14] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: Low-drift, robust, and fast," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2174–2181, IEEE, 2015.

[15] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*. Cambridge university press, 2010.

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

[17] J. Borenstein and L. Feng, "Umbmark: A benchmark test for measuring odometry errors in mobile robots," in *Mobile Robots X*, vol. 2591, pp. 113–125, International Society for Optics and Photonics, 1995.

[18] C. Schlegel, H. Jaberg, and M. Schuster, "Vision based person tracking with a mobile robot," in *In Proc. British Machine Vision Conf*, Citeseer, 1998.

[19] C.-H. Ku and W.-H. Tsai, "Smooth vision-based autonomous land vehicle navigation in indoor environments by person following using

sequential pattern recognition," *Journal of Robotic Systems*, vol. 16, no. 5, pp. 249–262, 1999.

[20] M. Piaggio, R. Fornaro, A. Piombo, L. Sanna, and R. Zaccaria, "An optical-flow person following behaviour," in *Intelligent Control (ISIC), 1998. Held jointly with IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA), Intelligent Systems and Semiotics (ISAS), Proceedings*, pp. 301–306, IEEE, 1998.

[21] T. Yamane, Y. Shirai, and J. Miura, "Person tracking by integrating optical flow and uniform brightness regions," in *1998 IEEE International Conference on Robotics and Automation, 1998. Proceedings.*, vol. 4, pp. 3267–3272, IEEE, 1998.

[22] G. Chivilò, F. Mezzaro, A. Sgorbissa, and R. Zaccaria, "Follow-the-leader behaviour through optical flow minimization," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings*, vol. 4, pp. 3182–3187, IEEE, 2004.

[23] D. Beymer and K. Konolige, "Tracking people from a mobile platform," in *Experimental robotics VIII*, pp. 234–244, Springer, 2003.

[24] M. Tarokh and P. Ferrari, "Case study: Robotic person following using fuzzy control and image segmentation," *Journal of Field Robotics*, vol. 20, no. 9, pp. 557–568, 2003.

[25] T. Yoshimi, M. Nishiyama, T. Sonoura, H. Nakamoto, S. Tokura, H. Sato, F. Ozaki, N. Matsuhira, and H. Mizoguchi, "Development of

a person following robot with vision based target detection," in *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pp. 5286–5291, IEEE, 2006.

[26] D. Calisi, L. Iocchi, and R. Leone, "Person following through appearance models and stereo vision using a mobile robot.," in *VISAPP (Workshop on on Robot Vision)*, pp. 46–56, 2007.

[27] Z. Chen and S. T. Birchfield, "Person following with a mobile robot using binocular feature-based tracking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007*, pp. 815–820, IEEE, 2007.

[28] H. Takemura, K. Ito, and H. Mizoguchi, "Person following mobile robot under varying illumination based on distance and color information," in *IEEE International Conference on Robotics and Biomimetics, 2007. ROBIO 2007*, pp. 1500–1505, IEEE, 2007.

[29] J. Satake and J. Miura, "Robust stereo-based person detection and tracking for a person following robot," in *ICRA Workshop on People Detection and Tracking*, pp. 1–10, 2009.

[30] M. Tarokh and P. Merloti, "Vision-based robotic person following under light variations and difficult walking maneuvers," *Journal of Field Robotics*, vol. 27, no. 4, pp. 387–398, 2010.

[31] M. Tarokh and R. Shenoy, "Vision-based robotic person following in fast walking," in *2014 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pp. 3172–3177, IEEE, 2014.

[32] J. Satake, M. Chiba, and J. Miura, "A sift-based person identification using a distance-dependent appearance model for a person following robot," in *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 962–967, IEEE, 2012.

[33] M. Awai, T. Shimizu, T. Kaneko, A. Yamashita, and H. Asama, "Hog-based person following and autonomous returning using generated map by mobile robot equipped with camera and laser range finder," in *Intelligent Autonomous Systems 12*, pp. 51–60, Springer, 2013.

[34] M. Awai, A. Yamashita, T. Shimizu, T. Kaneko, Y. Kobayashi, and H. Asama, "Development of mobile robot system equipped with camera and laser range finder realizing hog-based person following and autonomous returning," *Journal ref: Journal of Robotics and Mechatronics*, vol. 26, no. 1, pp. 68–77, 2014.

[35] K. Koide and J. Miura, "Identification of a specific person using color, height, and gait features for a person following robot," *Robotics and Autonomous Systems*, vol. 84, pp. 76–87, 2016.

[36] Y. Yoon, W.-h. Yun, H. Yoon, and J. Kim, "Real-time visual target tracking in rgb-d data for person-following robots," in *22nd International Conference on Pattern Recognition (ICPR)*, pp. 2227–2232, IEEE, 2014.

[37] M. Kobilarov, G. Sukhatme, J. Hyams, and P. Batavia, "People tracking and following with mobile robot using an omnidirectional camera

and a laser," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*, pp. 557–562, IEEE, 2006.

[38] S. Shaker, J. J. Saade, and D. Asmar, "Fuzzy inference-based person-following robot," in *International Journal Of Systems Applications, Engineering & Development*, vol. 2, pp. 29–34, Citeseer, 2008.

[39] G. Doisy, A. Jevtic, E. Lucet, and Y. Edan, "Adaptive person-following algorithm based on depth images and mapping," in *Proc. of the IROS Workshop on Robot Motion Planning*, 2012.

[40] A. Cosgun, D. A. Florencio, and H. I. Christensen, "Autonomous person following for telepresence robots," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4335–4342, IEEE, 2013.

[41] F. Sardari and M. E. Moghaddam, "A hybrid occlusion free object tracking method using particle filter and modified galaxy based search meta-heuristic algorithm," *Applied Soft Computing*, vol. 50, pp. 280–299, 2017.

[42] C. Gao, H. Shi, J.-G. Yu, and N. Sang, "Enhancement of elda tracker based on CNN features and adaptive model update," *Sensors*, vol. 16, no. 4, p. 545, 2016.

[43] Y. Hua, K. Alahari, and C. Schmid, "Online object tracking with proposal selection," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[44] M. Zhai, M. J. Roshtkhari, and G. Mori, "Deep learning of appearance models for online object tracking," *arXiv preprint arXiv:1607.02568*, 2016.

[45] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," *International Journal of Computer Vision*, vol. 111, no. 2, pp. 213–228, 2015.

[46] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, and P. H. Torr, "Struck: Structured output tracking with kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 2096–2109, 2016.

[47] L. Zhang and L. van der Maaten, "Structure preserving object tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition*, pp. 1838–1845, 2013.

[48] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time object tracking via online discriminative feature selection," *IEEE Transactions on Image Processing*, vol. 22, no. 12, pp. 4664–4677, 2013.

[49] S. Song and J. Xiao, "Tracking revisited using rgbd camera: Unified benchmark and baselines," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 233–240, 2013.

[50] J. Fan, W. Xu, Y. Wu, and Y. Gong, "Human tracking using convolutional neural networks," *IEEE Transactions on Neural Networks*, vol. 21, no. 10, pp. 1610–1623, 2010.

[51] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network.," in *ICML*, pp. 597–606, 2015.

[52] L. Zhang and P. N. Suganthan, "Visual tracking with convolutional neural network," in *Systems, Man, and Cybernetics (SMC), 2015 IEEE International Conference on*, pp. 2072–2077, IEEE, 2015.

[53] C. Gao, F. Chen, J.-G. Yu, R. Huang, and N. Sang, "Robust visual tracking using exemplar-based detectors," *IEEE Transactions on Circuits and Systems for Video Technology*, 2015.

[54] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European Conference on Computer Vision*, pp. 749–765, Springer, 2016.

[55] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *European Conference on Computer Vision*, pp. 345–360, Springer, 2014.

[56] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust rgb-d object recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 681–687, IEEE, 2015.

[57] C. Couprie, C. Farabet, L. Najman, and Y. Lecun, "Indoor semantic segmentation using depth information," in *International Conference on Learning Representations (ICLR2013), April 2013*, 2013.

[58] J. Borenstein, H. Everett, L. Feng, *et al.*, "Where am i? sensors and methods for mobile robot positioning," tech. rep., University of Michigan, 1996.

[59] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time.," in *Robotics: Science and Systems*, vol. 2, 2014.

[60] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *Proceedings of the 10th international conference on Ubiquitous Computing*, pp. 114–123, ACM, 2008.

[61] M. O. Aqel, M. H. Marhaban, M. I. Saripan, and N. B. Ismail, "Review of visual odometry: types, approaches, challenges, and applications," *SpringerPlus*, vol. 5, no. 1, p. 1897, 2016.

[62] A. Noureldin, T. B. Karamat, and J. Georgy, *Fundamentals of inertial navigation, satellite-based positioning and their integration*. Springer Science & Business Media, 2012.

[63] A. Sanchez, A. de Castro, S. Elvira, G. Glez-de Rivera, and J. Garrido, "Autonomous indoor ultrasonic positioning system based on a low-cost conditioning circuit," *Measurement*, vol. 45, no. 3, pp. 276–283, 2012.

[64] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[65] K. Tsotsos, A. Chiuso, and S. Soatto, "Robust inference for visual-inertial sensor fusion," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5203–5210, IEEE, 2015.

[66] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE Robotics & Automation Magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[67] D. Nistér, O. Naroditsky, and J. Bergen, "Visual odometry," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004*, vol. 1, pp. I–I, Ieee, 2004.

[68] L. Matthies and S. Shafer, "Error modeling in stereo navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 239–248, 1987.

[69] A. Howard, "Real-time stereo visual odometry for autonomous ground vehicles," in *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008. IROS 2008*, pp. 3946–3952, IEEE, 2008.

[70] B. Kitt, A. Geiger, and H. Lategahn, "Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*, pp. 486–492, IEEE, 2010.

[71] M. Maimone, Y. Cheng, and L. Matthies, "Two years of visual odometry on the mars exploration rovers," *Journal of Field Robotics*, vol. 24, no. 3, pp. 169–186, 2007.

[72] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *6th IEEE and ACM International Symposium on Mixed and Augmented Reality, 2007. ISMAR 2007*, pp. 225–234, IEEE, 2007.

[73] T. Pire, T. Fischer, J. Civera, P. De Cristóforis, and J. J. Berlles, "Stereo parallel tracking and mapping for robot localization," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pp. 1373–1378, IEEE, 2015.

[74] T. Pire, T. Fischer, G. Castro, P. De Cristóforis, J. Civera, and J. J. Berlles, "S-ptam: Stereo parallel tracking and mapping," *Robotics and Autonomous Systems*, 2017.

[75] R. Mur-Artal and J. D. Tardós, "Orb-slam2: an open-source slam system for monocular, stereo and rgb-d cameras," *arXiv preprint arXiv:1610.06475*, 2016.

[76] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European Conference on Computer Vision*, pp. 834–849, Springer, 2014.

[77] I. Cvišić and I. Petrović, "Stereo odometry based on careful feature selection and tracking," in *Mobile Robots (ECMR), 2015 European Conference on*, pp. 1–6, IEEE, 2015.

[78] M. Cummins and P. Newman, "Appearance-only slam at large scale with fab-map 2.0," *The International Journal of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, 2011.

[79] J. Zhu, "Image gradient-based joint direct visual odometry for stereo camera," in *International Joint Conference on Artificial Intelligence*, pp. 4558–4564, 2017.

[80] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[81] O. Pink, "Visual map matching and localization using a global feature map," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08*, pp. 1–7, IEEE, 2008.

[82] O. Pink, F. Moosmann, and A. Bachmann, "Visual features for vehicle localization and ego-motion estimation," in *Intelligent Vehicles Symposium, 2009 IEEE*, pp. 254–260, IEEE, 2009.

[83] H. Chu, A. Gallagher, and T. Chen, "Gps refinement and camera orientation estimation from a single image and a 2d map," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, pp. 171–178, 2014.

[84] H. Chu, D. Ki Kim, and T. Chen, "You are here: Mimicking the human thinking process in reading floor-plans," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2210–2218, 2015.

[85] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *1999 IEEE International Conference on*

Robotics and Automation, 1999. Proceedings, vol. 2, pp. 1322–1328, IEEE, 1999.

[86] C.-C. Wang and C. Thorpe, "Simultaneous localization and mapping with detection and tracking of moving objects," in *Proceedings. ICRA'02. IEEE International Conference on Robotics and Automation, 2002*, vol. 3, pp. 2918–2924, IEEE, 2002.

[87] S.-W. Yang and C.-C. Wang, "Multiple-model ransac for ego-motion estimation in highly dynamic environments," in *IEEE International Conference on Robotics and Automation, 2009 ICRA'09.*, pp. 3531–3538, IEEE, 2009.

[88] D. Sun, F. Geißer, and B. Nebel, "Towards effective localization in dynamic environments," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4517–4523, IEEE, 2016.

[89] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, p. 5, Kobe, 2009.

[90] S. Nishimura, K. Itou, T. Kikuchi, H. Takemura, and H. Mizoguchi, "A study of robotizing daily items for an autonomous carrying system-development of person following shopping cart robot," in *9th International Conference on Control, Automation, Robotics and Vision, 2006. ICARCV'06*, pp. 1–6, IEEE, 2006.

[91] P. Alves-Oliveira and A. Paiva, "A study on trust in a robotic suitcase," in *Social Robotics: 8th International Conference, ICSR 2016, Kansas City, MO, USA, November 1-3, 2016 Proceedings*, vol. 9979, p. 179, Springer, 2016.

[92] Y. Yoon, H. Yoon, and J. Kim, "Depth assisted person following robots," in *RO-MAN, 2013 IEEE*, pp. 330–331, IEEE, 2013.

[93] M. Kanbara, T. Okuma, H. Takemura, and N. Yokoya, "Real-time composition of stereo images for video see-through augmented reality," in *IEEE International Conference on Multimedia Computing and Systems, 1999*, vol. 1, pp. 213–219, IEEE, 1999.

[94] H. Grabner and H. Bischof, "On-line boosting and vision," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 260–267, IEEE, 2006.

[95] C. Zhang and Z. Zhang, "Boosting-based face detection and adaptation," *Synthesis Lectures on Computer Vision*, vol. 2, no. 1, pp. 1–140, 2010.

[96] A. Beygelzimer, S. Kale, and H. Luo, "Optimal and adaptive algorithms for online boosting," in *25th International Joint Conference on Artificial Intelligence (IJCAI 2016)*, pp. 4120–4124, 2016.

[97] Y. Sha and G.-y. Zhang, "An adaptive weighted boosting algorithm for road detection," in *2010 International Conference on Networking, Sensing and Control (ICNSC)*, pp. 582–586, IEEE, 2010.

[98] K. Nguyen, T. Ng, and L. Nguyen, "Adaptive boosting features for automatic speech recognition," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4733–4736, IEEE, 2012.

[99] J. Wang, X. Chen, and W. Gao, "Online selecting discriminative tracking features using particle filter," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2, pp. 1037–1042, IEEE, 2005.

[100] M. Kanbara, T. Okuma, H. Takemura, and N. Yokoya, "A stereoscopic video see-through augmented reality system based on real-time vision-based registration," in *Virtual Reality, 2000. Proceedings. IEEE*, pp. 255–262, IEEE, 2000.

[101] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *in IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011.

[102] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001. CVPR 2001.*, vol. 1, pp. I–I, IEEE, 2001.

[103] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary

patterns," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 7, pp. 971–987, 2002.

[104] A. O'Dwyer, *Handbook of PI and PID controller tuning rules*. World Scientific, 2009.

[105] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, "Visual simultaneous localization and mapping: a survey," *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2015.

[106] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.

[107] J. Borenstein, H. R. Everett, L. Feng, and D. K. Wehe, "Mobile robot positioning: Sensors and techniques," 1997.

[108] A. Elfes, "Sonar-based real-world mapping and navigation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 3, pp. 249–265, 1987.

[109] R. G. von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: a line segment detector," *Image Processing On Line*, vol. 2, pp. 35–55, 2012.

[110] J. M. Font and J. A. Batlle, "Mobile robot localization. revisiting the triangulation methods," *IFAC Proceedings Volumes*, vol. 39, no. 15, pp. 340–345, 2006.

[111] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 573–580, IEEE, 2012.

[112] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The new college vision and laser data set," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, 2009.