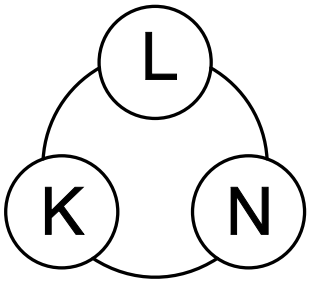

Technische Universität München
Lehrstuhl für Kommunikationsnetze
Prof. Dr.-Ing. Wolfgang Kellerer

Master‘s Thesis

Title of the Thesis

Author:	Last Name, First Name
Address:	Somestreet 007
	12345 City
	Country
Matriculation Number:	XXXXXXX
Supervisor:	Supervisor Name
Begin:	01. January 1900
End:	01. January 2000

Abstract

A short abstract of the thesis in English.

Statement

I assure the single handed composition of this master's thesis only supported by declared resources.

Hamburg, 19. December 2016

(John Doe)

Foreword

I would like to use this opportunity to express my gratitude to Northern Institute of Technology Management (NIT). I spent two fabulous years with colleagues, staffs and teachers from NIT. What makes these two years so special and unforgettable is not only the excellent education, but also the international atmosphere and the friendships we developed over time.

I am also appreciated for the help and guidance I received from Prof. Dr. Corneliue Herstatt, Prof. Dr. Christian Lüthje and from supervisor Dipl.-Ing. Moritz Göldner. I am really grateful for their patience over these four months and their attitudes towards scientific researches inspired me a lot.

Special thanks to all my friends who make my study life more enjoyable. Lastly, I sincerely thank my families. Although they live in the other side of the continent, their continuous supports and cares encourage me to step forward.

Hamburg, 19. December 2016

Contents

Chapter 1

Introduction

In recent years, more and more robots are deployed in not only industrial environments but also human populated areas. In order to fulfill their tasks, it is necessary for robots to interact and even cooperate with people. Therefore, tracking the locations of people in those environments has gained enormous attentions in robotics community. Besides, with the increasing popularity of artificial intelligence, robots are also expected to be intelligent enough to predict possible future locations of walking human even when encountered with occlusions or missing of sensor data. Enabled with an accurate tracking and prediction of human movements, robots are able to have a better understanding of the environment, which will facilitate interactions between robots and human.

The very fist requirement for a robot to operate is to model the environment. A simple yet effective way is to use occupancy grid map representation of the environment (Elfes, 1989). It decomposes the environment into cells with a predefined resolution, and the state of each cell is a random variable with values of either *occupied* or *not occupied*. This representation has been extensively used in many different kind of robot tasks like simultaneous localization and mapping (SLAM). The classical grid map representation treats the environment as static, which is not always the case in real scenarios. In other words, the environment can be dynamic since objects can move around in the environment. Object tracking addresses dynamics in environment explicitly, since it tracks and predicts how objects move. If there are more than one object involved at the same time, the tracking problem becomes multiple object tracking (MOT). In people tracking systems, the dynamics in environment refer to people location changes along the time horizon (i.e., trajectories).

Commonly, a tracking system is implemented as a multiple stage pipeline, which consists of object detection, data association, motion modeling and occupancy generation. When deal with a multiple object tracking problem, data association becomes very tricky. The classical way to perform data association is to maintain a list of known objects, and associates new observations with those objects. The main difficulty of this approach is to deal with *birth*

(whether observation is from a new object) and *death* (whether a maintained object should be deleted) of tracks explicitly (Gauvrit et al., 1997).

To address the data association problem, Coué et al. (2006) proposed *Bayesian occupancy filter* (BOF), which is an object tracking algorithm based on grid map representation of environment. It essentially avoids data association step in the tracking pipeline, since concepts of *objects* and *tracks* dose not exist in BOF framework. Rather than treat tracking problem from an *object* point of view, BOF addresses tracking from a *cell* perspective. That is to say, tracks are replaced by transitions of occupancies between cells over time. Meanwhile, BOF is robust to object occlusion thanks to the combination of *prediction* and *estimation* steps. This two-step mechanism is well suited for handling the consistency between occupancy predictions and new observations, therefore uncertainties incurred by occlusion is naturally handled in a probabilistic way. Over the past few years, there has been some extensions over BOF proposed in literature (Gindele et al., 2009; Brechtel et al., 2010; Llamazares et al., 2013).

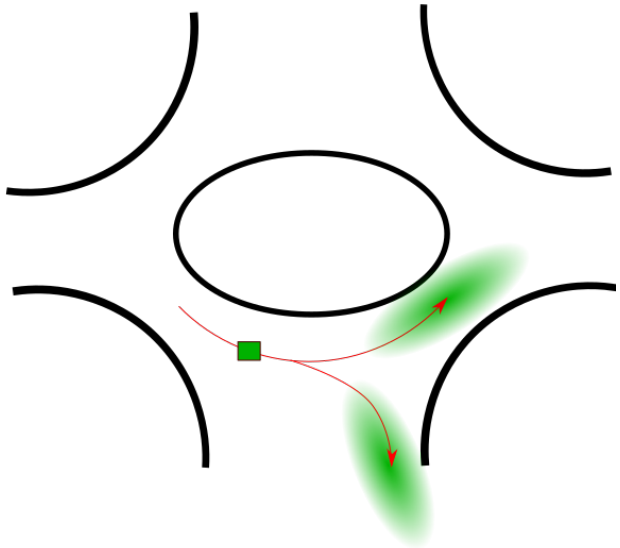

Figure 1.1: *This shows an illustration of a round-about. Cars can only drive in directions indicated by red arrows. The green rectangle represents a car at that location. Since we know that the car must drive in the specified directions, the possible locations of the car after a few time steps are indicated by the two green eclipses. This prior knowledge helps us track the dynamic objects since we have better chances to locate the car according to the motion pattern.*

A tracking system usually predicts the state of the world (e.g., object location and velocity) based on past observations. Many tracking algorithms also incorporate motion models of the objects being tracked, since this allows us to utilize prior knowledge of object's motion cues. As an intuitive example, let us consider a round-about as shown in Figure 1.1, where cars can only drive in one direction. To track a car driving in a round-about, prediction

of next possible locations of the car should always be on the side which allowed driving direction indicates, since we know that the car must follow that direction. Likewise, human tracking in indoor environments can also benefit from human motion patterns. For BOF and its variants, the objects being tracked are assumed to perform linear motion, which indicates that objects always move in the same direction as in the last time step. Although Gindele et al. (2009) proposed to incorporate prior map knowledge (BOFUM) to better model the motion dynamics based on cell context, linear motion is too simplistic to model actual human motion in indoor environments.

Based on our observations of human trajectories, we found two aspects very important for explaining human motion: 1) Unlike in free space, people have to move under spatial constraints in indoor environments. For example, people tend to walk along the central area between walls in a corridor and change their walking directions when they have to make turns. That is to say, the spatial constraints limit human motion patterns and therefore human motion is very place dependent. 2) People move continuously in time and space, which means they does not disappear or appear out of thin air. If a person is currently at a cell of a grid map, he or she has to walk through its neighboring cells first. In other words, future movement events starting from a region of interests are highly influenced by state changes of its neighboring cells. These observations motivates us to model human motion in a way that captures both **place dependency** as well as **spatial correlation** between cells.

In computer vision community, researches in machine learning over last few years has shown a lot of successes. Image classification algorithms based on convolutional neural networks (CNNs) has won all ImageNet classification challenges since 2012 (Russakovsky et al., 2015). It turns out that CNNs can achieve good results in not only computer vision tasks but also many other domains. For example, recurrent neural networks (RNNs) can be applied in automatic language translation (Cho et al., 2014). Deep reinforcement learning are good choice for teaching robots to perform human actions like grasping(Levine et al., 2016). Essentially, the reason why CNN works in those domains is that it is able to capture complex structure in data. Once a CNN learns that structure, it can generalize to cases that never occurs during training.

In order to utilize the powerful generalization abilities of CNNs, we proposed to model motion patterns in a way that can be easily expressed as the output of a CNN. In this way, if we feed a gird map as input and human motion patterns as ground truth to a network, it should be able to learn the patterns after training with lots of data. We model human motion patterns as a set of conditional probabilities for every cell on the map. They represent how likely a person moves to one of the neighboring cells, conditioned on which neighboring cell he or she comes from. Since these probabilities incorporate information about the incoming cell, the motion model captures spatial correlations between cells. Besides, they are different for each cell based on cell's context on the map, which means the learned motion pattern is expressive enough to predict different motions at different locations. In other words, our motion model is place dependent. In fact, similar idea of

making motion patterns place dependent has been proposed by Kucner et al. (2013), but in order to learn motion pattern on a map, they need sensor observations from that specific map. In other words, their method lacks of generalization ability. On the contrary, with a large training set, we exploit the power of CNNs so that our method can generalize to maps that have never occurs.

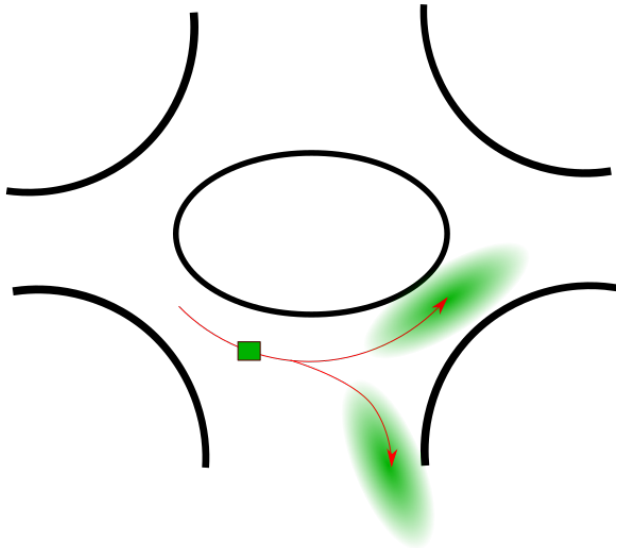

Figure 1.2: *An example of future occupancy predictions of our method. The map shows a T-section, where only white spaces are walkable. Initially, a person is shown as a red rectangle and with velocity towards left. Since no further sensor reading is given, BOFMP propagates occupancies over time. After some time interval, most occupancies appear in both upper and lower branches of the corridor, which proves that our motion model successfully learns that people tend to make turns at the intersection. Besides, our BOFMP predicts more occupancies in the middle of corridors, since it learns that people are less likely to walk near walls.*

The data we use for training network is generated from simulated human walking trajectories on SLAM-generated maps of real world offices. Once the network finishes learning, we can feed new maps to the network and obtain human motion patterns from network outputs. These motion patterns are then incorporated into BOF framework to perform human tracking in indoor environments. We call our tracking method as *Bayesian Occupancy Filter with Motion Patterns* (BOFMP). If only initial state of a person is given and no further observations are provided, our method is able to predict reachable areas after some time steps. Figure 7.1 shows an example of future occupancy predictions of our method without sensor observations. It shows that our way of modeling human motion patterns enables BOF to propagate occupancies to reasonable areas.

The main contributions of this thesis work are:

1. We present a way of modeling human motion patterns that captures both place

dependency and spatial correlations between cells. Besides, thanks to powerful generalization abilities of CNNs, our method can generate motion patterns on maps that are never seen by our model.

2. We incorporate the learned motion patterns into BOF framework seamlessly for human tracking in indoor environment, and achieve better tracking performance than baseline method. The whole pipeline of modeling motion pattern and tracking presented in this thesis work can be applied in other scenarios, such as car tracking in ADAS systems.

The rest of this thesis is organized as follows: Chapter 2 summarizes the related works and highlights the similarities and differences between our method and others in literatures. Chapter 3 introduces how BOF is derived and mathematical formulation of CNNs. Chapter 4 explains the detailed implementations, such as generation of dataset and BOF. Chapter 5 summarizes the dataset and presents the tracking performance of our method and the baseline. Chapter 6 concludes the thesis work and presents possible improvements on our method.

Chapter 2

Literature Review

This chapter lists literatures that are related to methods or concepts used in this thesis work. Section 2.1 discusses two classical ways of tracking object and introduces BOF and its variants. Section 2.2 presents literatures that try to model dynamics in environment. Section 2.3 details recent researches on neural networks and correlations between RNNs and Bayesian filters.

2.1 Object Tracking

Perception of environment can be done with different sensors, which also differentiates object tracking applications. Visual tracking refers to object tracking with RGB cameras (Ross et al., 2008). In this thesis work, however, we mainly concern object tracking with 2D laser scanners. One popular paradigm of performing tracking is tracking by detection: moving objects are firstly detected and then determine how to pair them with existing tracks. Generally, there are two approaches to deal with detection based object tracking: *model-free* and *model-based* (Wang et al., 2015).

The detection of model-free object tracking works based on motion cues. In this case, prior knowledge on neither object's semantic information (whether it is a car or a person) nor object's shape or geometric properties is needed. However, this approach only tracks objects that show instantaneous motion dynamics and potential moving objects might be neglected. BOF can be regraded as a model-free tracking algorithm since it predicts occupancies based on cells' velocities. Similar to BOF, Ross et al. (2008) models the environment with occupancy gird map. Their work combines SLAM with dynamic object tracking in outdoor environment. Once the vehicle equipped with laser sensors is self-located and objects are detected by motion cues, they apply a method called Global Nearest Neighborhood (GNN) for tracking. On the contrary, for model-based tracking, semantic class of the objects being tracked is given, and detection is done with the help of

a parametric model of the object's shape. For human tracking, a person is represented as point blob or modeled based on detection of legs (Arras et al., 2008; Cui et al., 2006).

The tracking by detection paradigm has to address *data association* explicitly, which is known to be as the main difficulty in tracking. Data associations refers to match detections of moving objects in new observations to a set of tracked trajectories from last time step. To address data association problems in tracking, many methods have been proposed in literature. Historically, nearest neighbors based approaches are used in the early stage (Fod et al., 2002). Schulz et al. (2001) detect people by their legs in 2D laser scans and propose the Joint Probabilistic Data Association Filter (JPDAF) for tracking. Arras et al. (2008) is similar in identifying people with legs but their tracking is done with Multi-hypothesis tracking (MHT). Although the mentioned methods partially address the data association problem, their performance is not stable in densely cluttered environments, where occlusions happen quite often.

On the contrary to tracking by detection paradigm, tracking can also be done without detection, e.g., tracking by filtering. Coué et al. (2006) propose Bayesian Occupancy Filter for object tracking in automotive applications. The environment is represented by a grid map, and the state of each cell is characterized by its occupancy and velocity. Inspired by Bayesian Filter, tracking works in a form of recursive *predictions* and *estimations* of cell's state. One of important advantages of BOF is that the concepts of *objects* and *tracks* are replaced by transitions of occupancies between cells over time. That is to say, data association is addressed implicitly by BOF. Later works extend BOF to better adapt to real world situations. BOF's linear motion model cannot adapt to traffic scenarios like curved roads. Therefore, Gindele et al. (2009) propose to enrich motion model of BOF with prior map knowledge (BOFUM), which can be obtained from navigation systems. By this way, BOFUM predicts occupancies according to motion preferences at different locations, which results in reliable estimates even when occlusion happens.

Despite of various variants of BOF, all of them assume that objects perform linear motion. However, linear motion is obviously over-simplified for real world scenarios. Besides, since objects cannot appear or disappear suddenly, the state changes of a cell's neighboring cells contain valuable information for prediction of its future state. Therefore, we proposed to model human motion patterns in a way that captures both place dependency and spatial correlation of cells. This motion model is then incorporated into BOF framework to perform human tracking. A similar idea is used in the work of Luber et al. (2011), which proposed a place dependent people tracking algorithm. Unlike BOF which tracks objects by filtering, they perform people tracking by detection and data association. They proposed to model human activity events as a three-layer spatial affordance map with each layer characterizing the probability distribution of new tracks, matched tracks and false alarms respectively. Each layer is represented by a *spatial* Poisson process which introduces spatial dependency and the parameter of each cell is learned from a sequence of observations. Figure 2.1 shows one example of two layers of the learned spatial affordance map. Their motion model is place dependent because it is dependent on the spatial affordance map. They

use boosted features for people detection (Arras et al., 2007) and they extend multiple hypothesis tracking (MHT) for data association. Their results show they achieve more accurate tracking behavior in terms of data association errors.

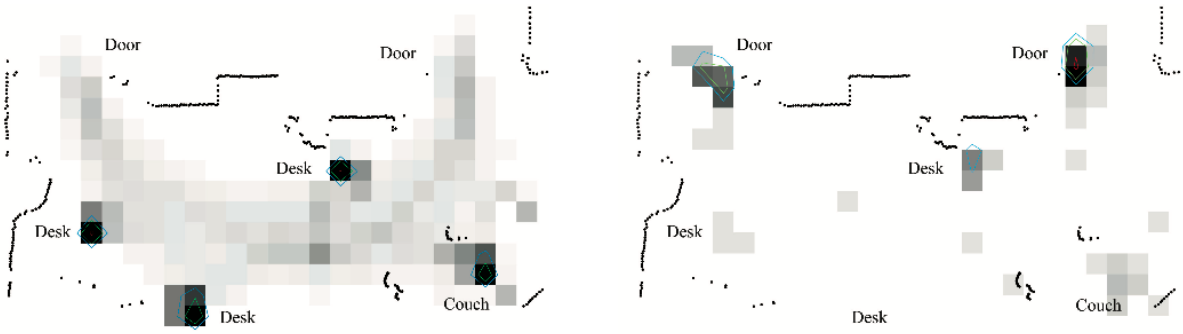

Figure 2.1: *Two layers of a learned spatial affordance map (Luber et al., 2011). On the left shows probability distribution of matched track events, and on the right shows that of new track events. Note that the maxima show at different regions in two maps. While on the left the maxima appear at places frequently used by people (desks and coach), on the right the maxima of new track events indicate places where people normally appear from, i.e., doors, desks and coach.*

2.2 Dynamics Modeling

Nowadays, more and more robots are entering *dynamic* environments. In order to fulfill their task, it is imperative for robots to understand these dynamics. In the context of human tracking, these dynamics refer to human movements in the environment. In literature, some researchers try to model human dynamics explicitly by a motion model. Meanwhile, others model them implicitly by regarding human dynamics as part of the environment and therefore model the environment directly. We discuss dynamics modeling separately according to these two approaches. In both cases, we represent the environment as a gird map with each cell representing a possible location.

2.2.1 Human Motion Modeling

In the early stage of human tracking, researches adopts simple and conservative motion models. Montemerlo et al. (2002) uses Brownian motion model for human tracking with Bayesian filters. The Brownian motion assumes people take random directions at each time step and there is no dependence between time steps. In other words, Brownian motion does not assign human dynamics with any pattern other than dispersion. As a consequence, when there is no observations, the predictions of people locations spread out over a large

area very quickly. This is a poor estimate as we know people normally does not move randomly. In reality, people normally go from the starting point and follow some motion patterns until they reach destination. A more realistic model is the first order motion model (a.k.a. linear motion model). For example, Meier and Ade (1999) used this model together with Kalman filter for human tracking. First order motion model assumes people always move in the same direction as in the last time step. This assumption might be true if a person is walking along a straight corridor, but in many cases it is invalid because people often need to make turns around corners. Some researchers proposed better ways to model people motion patterns. For example, Bruce and Gordon (2004) learns destinations by clustering real trajectories and uses a path planer to those destinations as a reference for human motion patterns. Liao et al. (2003) assumes that human tend to move along Voronoi graph of the environment and therefore constraint motion patterns by Voronoi graph.

Our motion model is different to above methods in two aspects: fineness and cell dependency. The Voronoi graph is constructed based on a *global* representation of the environment. It is predefined and only well suited for applications where high-level motion clue is needed (e.g., which rooms a person has visited). This kind of motion clues are too less precise to be used for recovering a person's exact location. Therefore, in order to get a finer motion model, we decompose the *global* task of modeling human motion pattern into *local* tasks at *cell level*. On the other hand, although a path planer to the learned destination defines an effective path, it does not necessarily cover the motion dynamics at every possible locations and assumes no dependency between these locations. One intuition we captured based on observations of human motion is that, for a cell in the map, the state changes of its neighboring cells is a strong indicator of predictions of its future state. To capture this *spatial* correlation, we propose to model changes of movement directions. For each cell, we learn based on human trajectories how likely a person's moving direction changes for the next time step, conditioned on the direction with which he or she enters this cell. Moreover, this way of motion modeling also captures *temporal* correlations, since it builds connections for each cell with its neighboring cells from both last time step and next time step.

In fact, after we came up with above way of modeling human motion, we found in literature the same idea has been used by Kucner et al. (2013) in an autonomous navigation scenario. Their model, which is named as Conditional Transition Map, is created by "learning the probability distribution of an object leaving to a certain neighboring cell, given the cell from which it entered into the current cell." They demonstrate their method with a roundabout, and the corresponding model is illustrated in Figure 2.2. One can see that at different locations in the roundabout, different motions are learned, which accurately models how cars drive through the roundabout. The difference between our method and theirs lies in how those probabilities are learned. The cross-correlation of temporal occupancy signals extracted from observations is used for learning in their method, while our model learns these probabilities by training a neural network. With lots of different maps as training data, our network is able to learn the motion patterns that are constrained by spatial

structures of these maps. One of the most important advantages of our method is that our network can generalize motion patterns for maps that never occur in training data. Therefore, our method can work in new environments without learning these probabilities from scratch.

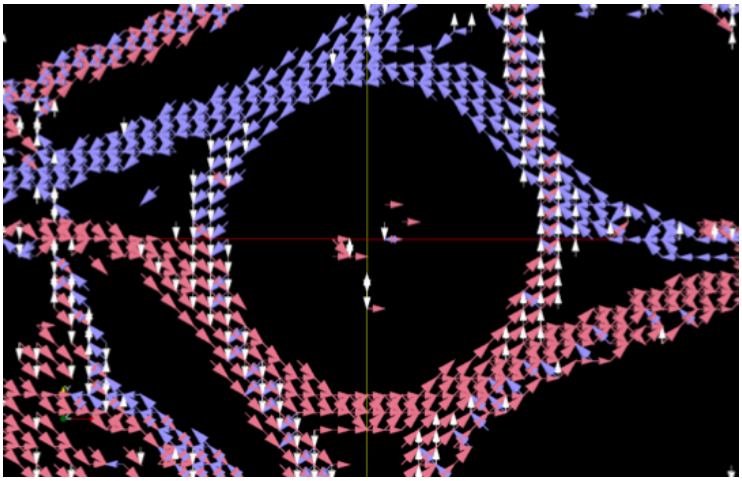

Figure 2.2: *Conditional transition map of a roundabout (Kucner et al., 2013). Arrows indicate the exit directions for that cell. For simplicity, the entering directions are not drawn. One can see that this model captures how cars drive through the roundabout.*

2.2.2 Dynamic Environment Modeling

For a mobile robot to operate, a proper way to model the environment is necessary. Elfes (1989) introduces occupancy grid as a representation of environment, which divides the environment into a grid of cells with a predefined resolution. Each cell is a random variable with values of either *occupied* or *not occupied*. The occupancy grid assumes the environment being static, which does not hold in real world situations. For example, a robot might need to work in a traffic intense environment with driving cars and pedestrian.

Early attempts to model dynamics in environment extents occupancy gird with a timescale framework. Arbuckle et al. (2002) propose to model dynamics with a stack multiple occupancy grids with each layer corresponding to a different timescale. They call this representation as Temporal Occupancy Grid (TOG). Objects' motion pattern can then be classified based on objects' occupancy traces across different layers of TOG. For example, if a cell is occupied across all layers of TOG (i.e., occupied at all timescales), it is likely to be part of background. Similarly, traffic patterns with various speeds can be identified based on its occupancy traces on a certain layer of TOG. Biber and Duckett (2009) uses multiple map representations of the dynamic environment with different timescales for long-term SLAM. Updates of mapping is calculated based on a (dynamic) sample set of observations that is updated over time by random replacement. Different timescales correspond to different

learning rate. Map with higher learning rate adapts to new observations faster than these with lower learning rates.

Although the above methods can be applied for modeling dynamics, essentially their static nature remains if we look at each layer that characterizes a specific timescale. In order to capture the dynamic nature of environment, Meyer-Delius et al. (2012) propose to model state changes of cells by a two-state hidden Markov models (HMMs). Therefore, the dynamics in environments are explicitly characterized by the state transition probabilities of HMMs. The parameters of these HMMs are estimated with an Expectation Maximization (EM) algorithm.

However, although their method relaxes the static cell state assumption made by occupancy grid, they assume that state changes of cells are caused by a *stationary* process, i.e., cell's state transition probabilities are constants over time. Moreover, the assumption of independences between cells is not always valid (e.g., state changes of a cell's neighboring cells are strong indicators of that cell's future state because objects normally move in a continuous manner). Due to above reasons, cell-specific input-output HMMs (IOHMMs) are used by Wang et al. (2014) to better model dynamic environments. An IOHMM imposes conditional dependences on latent variables and observed variables with an input sequence. An illustration of IOHMM used in their method is depicted in Figure 2.3. The input sequence is determined by past observations in neighboring cells. By this way, the transition model of their method is adaptive to input sequence that varies over time, which relaxes the assumption of stationary process made by Meyer-Delius et al. (2012). It also incorporates spatial correlations by making input sequence dependent on neighboring cells.

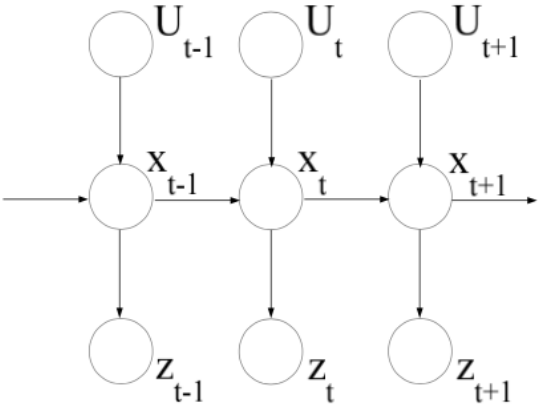

Figure 2.3: *An illustration of IOHMM used by Wang et al. (2014). Note in this IOHMM only latent variables are dependent on input sequence, which is determined by past events in neighboring cells.*

Our method is conceptually equivalent to theirs. In both methods, the influence of neighboring cells is captured by conditional dependence. Although our motion model is fixed for

a specific map, the occupancies propagate to a certain cell only when its neighboring cells have velocities towards it. This mechanism acts as a "trigger" which only get triggered at certain time and essentially makes our model time dependent.

2.3 Neural Networks

Since Krizhevsky et al. (2012) applied deep convolutional neural networks (CNNs) in large scale image classification from 2012, CNNs have gained a lot of successes in computer vision. Researches show that the depth of network plays a very important role in CNNs' performance (Simonyan and Zisserman, 2014). However, deep networks are more difficult to train, possibly due to poor gradient flow during back-propagation. To address this problem, He et al. (2016) propose to fit a *residual* mapping with stacked layers of CNNs instead of fitting the desired underlying mapping directly. This idea is implemented as "shortcut connections" between layers, which branches from one layer and connects to one of the followed layers. Moreover, (Huang et al., 2016) extends the idea of residual learning so that each layer has a shortcut connection to every other layer in the network.

Thanks to the development of ResNet (He et al., 2016) and DenseNet (Huang et al., 2016), the state-of-the-art CNNs can be as deep as over 1000 layers without training difficulties. Besides, fully convolutional neural networks (Long et al., 2015) are proven to be very powerful in solving semantic segmentation tasks thanks to its pixel-to-pixel classification ability. Jégou et al. (2017) extends the DenseNet to a fully convolutional neural network with an upsampling path for dealing with the problem of semantic segmentation.

Chapter 3

Background Knowledge

3.1 Bayesian Occupancy Filter (BOF)

3.1.1 Bayesian Filtering

3.1.2 Bayesian Occupancy Filter Formulation

3.1.3 BOF with Map Knowledge and Motion Model

3.2 Fully Convolutional Neural Network

3.2.1 Densely Connected Convolutional Networks (DenseNets)

3.2.2 Fully Convolutional DenseNets

3.3 Metrics

Chapter 4

Implementation Details

4.1 Human Trajectory Simulation

4.2 Architecture of Neural Network

4.3 Implementation of BOFUM

Before tracking starts, the occpancy and velocity probabilities are initialized uniformly, i.e.,

$$P_c\{O = occ\} = P_c\{O = nocc\} = 0.5P_c\{V = v\} = 1/number\ of\ velocities$$

4.4 Hyperparameter Tuning

Chapter 5

Results

5.1 Overview of Datasets

5.1.1 Simulated Dataset

5.1.2 Real Dataset

5.2 Evaluation of Tracking Performance

5.2.1 Tracking on Simulated Data

5.2.2 Tracking on Real Data

Chapter 6

Outlooks

6.1 End to End Training

6.2 Future Work

Chapter 7

Summary of Results

This thesis focuses on human tracking in indoor environments, such as offices and factories. In those environments, since there exists static obstacles, e.g., walls, tables, the tracking algorithm needs to differentiate between static obstacles and moving humans. One general tracking algorithm from literature is called as *Bayesian Occupancy Filter*(BOF). It represents the environment as a grid map, and it predicts the occupancy probability of each cell for every time step. An improved version of BOF is *Bayesian Occupancy Filter Using Map knowledge*(BOFUM). As its name indicates, it utilizes prior map knowledge about how likely a tracking object might be on different locations on a given map.

Human trajectories in indoor environment follow some motion patterns. For example, when there is corridor, humans tend to walk in the middle of the corridor, instead of walking besides the walls. To incoporate this human motion into tracking algorithm, we proposed *Bayesian Occupancy Filte Using Motion Pattern*(BOFMP). The idea is shown in Figure 7.1.

The main contributions of this thesis work are summarized as following:

1. Neural Network Training.
2. Human Trajectory Simulation.
3. Object Tracking using Bayesian Occupancy Filter
4. Comparision between BOFUM and BOFMP

7.1 Neural Network Training

To capture human motion patterns, we trained neural networks to learn how a walking person changes directions at different locations on a given map. Mathmatically, there are two ways to represent the motion pattern, either using conditional probability or joint

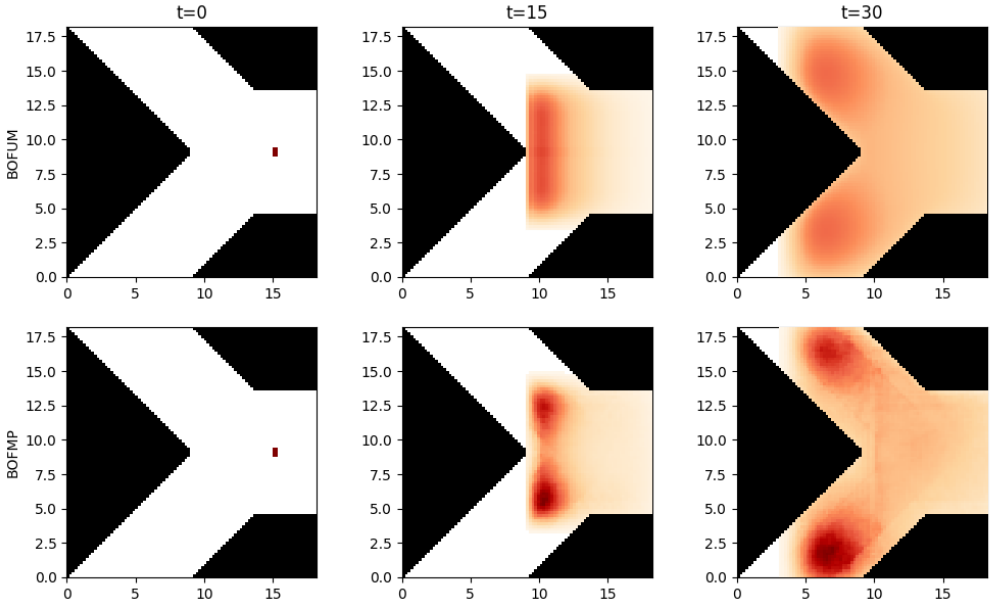

Figure 7.1: *Occupancy predictions for BOFUM and our proposed BOFMP after several time steps. The map shows a T-section. At $t = 0$, a person is shown as a red rectangle and with initial velocity towards left. At $t = 15$, the person encounters intersection. BOFUM has no information about human motion pattern, and continues to propagate occupancy towards left. Our BOFMP knows that humans are likely to turn to either upper or lower corridors. At $t = 30$, since occupancies going left vanish due to the wall, BOFUM predicts occupancies in corridors, but they are biased towards walls on the left. Our BOFMP predicts more occupancies in the middle of corridors, since it knows humans are more likely to walk in the middle.*

probability. For the former one, given a grid map as input, the network learns for each given grid cell c on the map, the probabilities of next possible velocity V conditioned on last velocity V^-:

$$P_c\{V|V^-\}$$

Alternatively, the network can also learn the joint probability:

$$P_c\{V, V^-\}$$

Theoretically, it is always better to have joint probability, since conditional probability can be caculated from joint probability by marginalization and Baye's rule. However, due

to reasons explained in Section 7.2, we are not able to get accurate joint probabilities as ground truth. Therefore, the network is trained to learn $P_c\{V|V^-\}$.

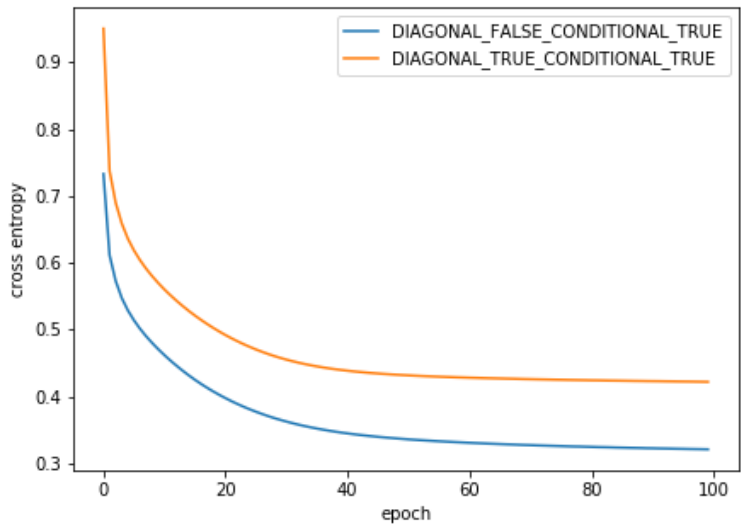

Figure 7.2: *Cross entropy loss for trainning networks. We trained networks to learn conditional probabilities $P_c\{V|V^-\}$. The green line shows trainning dynamics using data generated with directions "left, right, up and down". The yellow line also takes diagonal directions into account, which has 8 directions in total. Naturally, since more directions implies higher complexity, the overall loss for yellow line is higher than green line.*

The network consists of 31 convolutional layers. It has both down-sampling path for extracting high-level features and up-sampling path for recovering full resolution. The network is trained with mini-batches of size of 128, and is optimized with Adam optimizer. The training runs for 100 epoches, with early stopping patience of 15 epoches. Figure 7.2 shows the cross entropy loss during training.

7.2 Human Trajectory Simulation

To acquire enough amount of data for trainning our netwrok is expensive, especially when we have to consider all possible motion changes for every cell in a grid map. For non-diagonal directions, there are $4 \times 4 = 16$ possible motion changes. For diagonal directions, it goes up to $8 \times 8 = 64$. To get statistically sound motion pattern probabilities, it requires to record human trajectories on many different maps over a long period of time. However, due to practical reasons, we are not able to get that much real data. Instead, we simulate human trajectories with A-star algorithm from 6 real world maps with a total free space

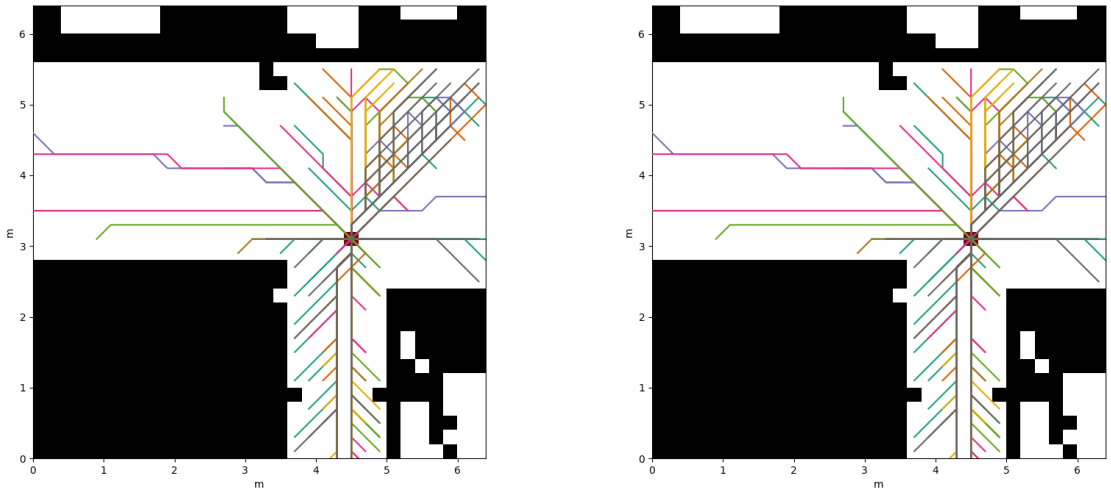

Figure 7.3: ***Left****: One example map crop as netwrok input. The map has size of* 32×32 *cells, with resolution of* $0.2m/cell$*. It also shows trajectories that goes through the red cell.* ***Right****: Visualization of conditional probability* $P\{V|V^-\}$ *for the red cell on left map. The axis shows velocities on* x, y *directions. Outter axes represent last velocity* V^-*, and inner represent next velocity* V*. One can see that* $P\{V = (-1,-1)|V^- = (-1,-1)\} = 0.21$ *and* $P\{V = (0,-1)|V^- = (-1,-1)\} = 0.79$*. This indicates that if a person reaches the red cell from upper right, it is very likely he will go downwards.*

.

area of ca. $6.6 \times 10^3\, m^2$. For each cell on the map, we try to sample 5 trajectories starting from that cell. With those trajectories, we can caculate motion pattern probabilities. Then we take random crops of size 32×32 cells from the real world maps as network inputs, and their corresponding probabilities as outputs.

Figure 7.3 shows one example of network input and the ground truth for one cell on the map. The number of samples we generated are summarized as follows:

	trainning	validation	test
number of samples	27,119	4,785	3,760

7.3 Object Tracking using Bayesian Occupancy Filter

Generally, Bayesian filter works in a recursive way and the filtering process can be decomposed into two stages: **prediction** and **correction**. For each time step, it firstly predicts the next state. After prediciton, when the new measurement is obtained, it corrects its

prediction based on measurement. For Bayesian occupancy filter used in tracking applications, the state of world is represented by, for each cell on the grid map, the probabilities of cell's velocities and occupancy. However, the measurement gives information about only whether a cell is occupied or not at each time step. In order to make predicitions for the next time step, the filter has to infer velocities of each cell based on past occupancy information from measurements. Figure 7.4 shows how BOFMP filter updates at one time step.

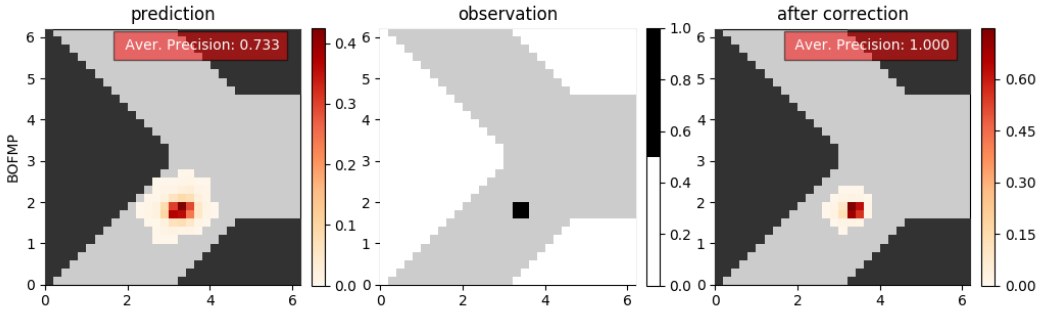

Figure 7.4: *One filtering step of BOFMP filter. At last time step, the tracking object goes from up to down. Based on motion pattern, BOFMP predicts that there are possibilities that this object will trun to left-down and keep going downwards. After measurement shows that this obejct still goes downwards, BOFMP corrects its predictions and attenuates probabilities of turning left-down. Since the measurement adds additional information for BOFMP to make better predictions, the average precision increases after correction step.*

7.4 Comparision between BOFUM and BOFMP

To compare the performance of BOFUM and BOFMP, we consider two scenarios:

1. **tracking**. Measurements are given at each time step, and we evaluate consistency between occupancy predicton and the ground truth at every time step before a certain time point t .

2. **future prediction**. From time point t on, the measurement is no longer given. Then we evaluate occupancy predictions with ground truth for the next n time steps.

The parameters of a BOF filter are:

1. **extent** e. It determines the maximum velocity of a cell. For example, if extent is 7, the velocities are in range $[-3, 3]$ cells/time step on both x and y axis. Since our neural network only models velocities within $[-1, 1]$, we need firstly extend it to higher velocities. The details on how we do this will be explained in the thesis.

2. **noise variance** δ^2. Both BOFUM and BOFMP assume the tracking object has a constant velocity, with a Gaussian distributed acceleration noise. This parameter determines how likely an object accelerates or decelerate.

3. **omega** Ω. In the correction step of BOF filters, measurments from sensors are incorporated into filter's prediction. Since sensor could be noisy, this parameter determines how much do we trust our measurements. The sensor used for our tracking application is laser rangefinder, which is rather physically reliable and therefore has a low Ω value.

The possible metrics that could be used for measuring the consistency between occupancy predicition and ground truth are: *cross entropy*, *f1 score* and *average precision*. We choose average precision as our metric and the reasons will be detailed in the thesis. To prove that our method is able to make predictions according to human motion pattern, we tune the parameters based on filter's performance for future predictions. For both filters, we randomly sample 100 sets of parameters from parameter ranges listed in Table **??**, evaluate on tracking cases with time steps of 16 and caculate average precision for every time step over all tracking cases. Note that value range of noise variance δ^2 for real data is higher than for simulated data. This is because in real data, there are higher uncertainties with human motion and sensor failures. The measurement is lost at time step $t = 9$, and we select the best set of parameters based on the average of average precisions for the next 8 time steps.

	value range for simulated data	value range for real data
e	$\{3, 5, 7\}$	$\{3, 5, 7\}$
δ^2	$[0.1, 0.6]$	$[0.2, 0.7]$
Ω	$[0.01, 0.2]$	$[0.01, 0.2]$

Table 7.1: *Parameter ranges for BOF filters.*

7.4.1 On simulated data

We generated 500 tracking cases as validation set for tuning filter parameters and 500 tracking cases for test set. Validation and test cases are generated from different maps, and on each map, there are either one or two walking humans. We firstly tuned the parameters on validation set for BOFUM and BOFMP individually. Then we apply both filters with their best parameters on test set. The best set of parameters and its corresponding average precision from $t = 9$ to $t = 16$ on test data are listed in Table **??**. Figure 7.5 shows mean of average precision over 500 test cases for each time step.

	e	δ^2	Ω	average precision for $t = 8 : 16$	mean
BOFUM	7	0.556	0.0164	0.767 0.599 0.500 0.416 0.349 0.279 0.232 0.192	0.417
BOFMP	7	0.373	0.0353	0.785 0.637 0.552 0.479 0.416 0.358 0.311 0.252	0.474

Table 7.2: *Best parameters for BOFUM and BOFMP on simulated data.*

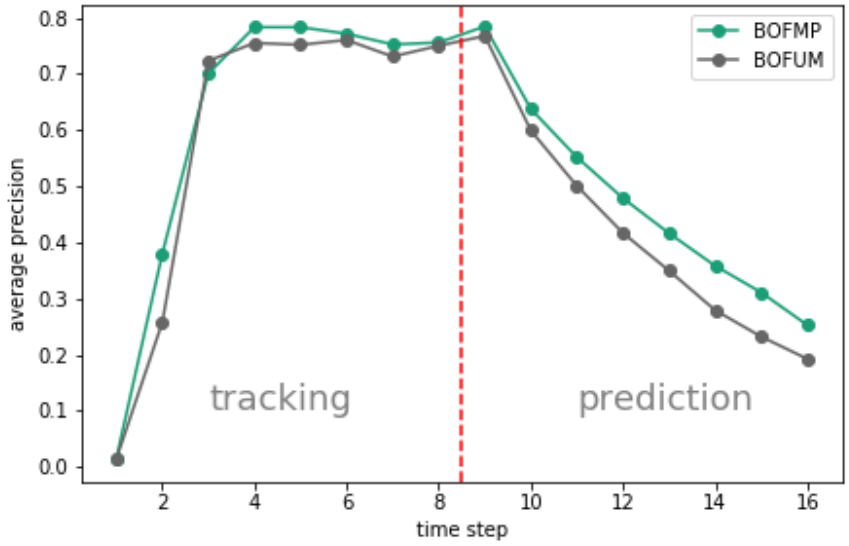

Figure 7.5: *Evaluation results on test data. The average precision for both filters start with values close to zero. Since we highly trust our measurments (low Ω), both filters are able to successfully track the objects within 3 time steps. The fact that average precision keeps a high vaule from $t = 3$ to $t = 8$ indicates that both filters can predict very well for the next immediate time step. Starting from $t = 9$ (see the red dash line), measurements are no longer given. The prediction is still accurate for the next time step ($t = 9$), but decreases progressively over time. This is expected, since without measurements, the state of world becomes more and more uncertain. Even though, we can see that our BOFMP have a higher average precision value than BOFUM at almost every time step, which indicates the improvements of our method in both tracking and future predicition stages.*

Figure 7.6 shows how BOFUM and BOFMP perform tracking on one case from test data. In this example, a person is walking from the lower door towards upper door. The grey curve on the map shows the trajectory of the person. At $t = 8$, the person walks with upwards velocity of 1 cell/time step. Both algorithms track the person very well with average precision of 1.0. At $t = 9$, the measurement is lost and future prediction stage of tracking starts. At $t = 10$, BOFUM predicts that the person will still goes upwards, with a low possibility going other directions. However, since BOFMP knows there is a door above, it predicts that the person is also very likely going to that door, and therefore turns

to upper left. At later time steps, BOFMP continues to predict occupancy probabilities towards upper door as well as other possible directions (i.e., door on the left and empty space on the right). On the contrary, BOFUM still propagates most occupancies upwards. At $t = 16$, since the object accelerates to higher velocity, most of occupancy predictions of BOFMP are left behind the tracking object.

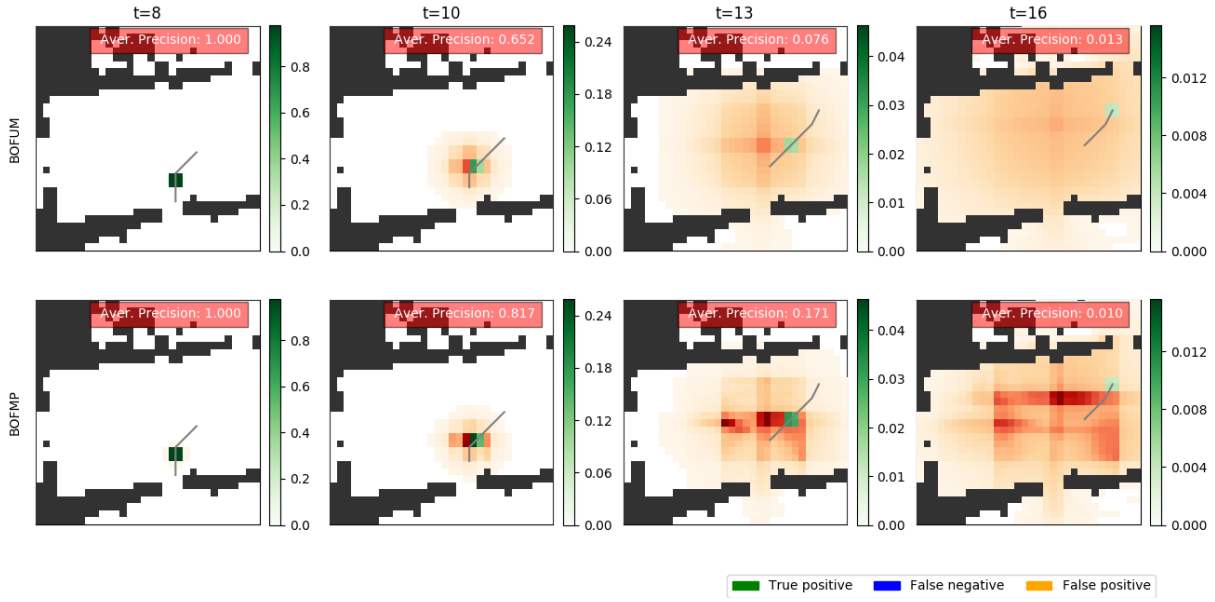

Figure 7.6: *One example of tracking case from test data.*

7.4.2 On real data

Although our neural network are trained on simulated human trajectories, we expect that our method also outperforms BOFUM on real tracking cases. We recorded human trajectories on two different ground plans by a laser rangefinder mounted on a robot. After processing raw laser data, we get 500 tracking cases for validation from one of the ground plans and 244 for test from the other.

Spatial blurring of motion probabilities

The simulated trajectories do not always reflect real human's motion patterns. This is because, as shown in Figure 7.7, humans are more flexible to decide when to make turns.

Therefore, to better adapt to real data, we introduce a techinique that blurs the motion probabilities $P_c\{V|V^- = v\}$ of a cell c spatially into its neighbors if a *turn* is detected. A *turn* on cell c for velocity v^- is defined as:

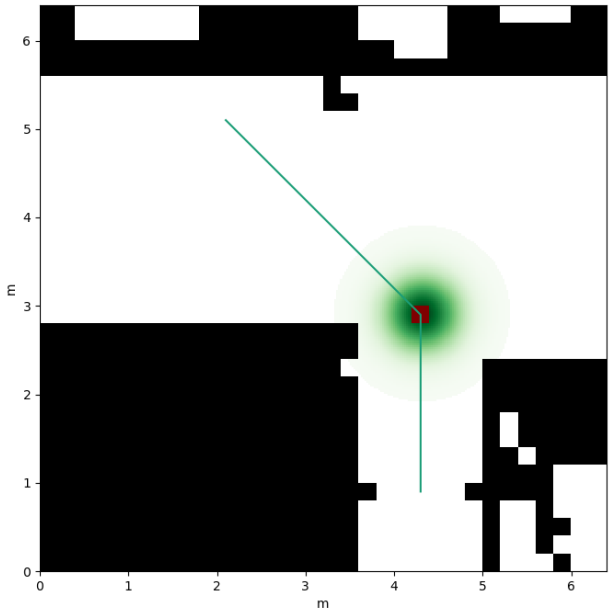

Figure 7.7: *A turn on simulated human trajectory. In order to reach the goal location in upper left corner, the simulated trajectory shows that a person will make a turn from going up to going up-left at location indicated by the red square. However, in real scenarios, a person is more flexible in deciding where to make that turn and he might turn at any location in the green area.*

$$turn_c(v^-) = \begin{cases} True, & \text{if} \quad \arg\max_v(P_c\{V = v|V^- = v^-\}) \,!= \, v^- \\ False, & \text{otherwise} \end{cases}$$

For each turn detected from motion pattern, we apply Gaussian blur spaitally to its neighbor cells' motion probability $P\{V|V^- = v^-\}$. As a consequence, another two parameters, blur extent $blurExt$ and blur variance $blurVar$, are introduced and their value ranges are listed in Table **??**.

	value range	*note*
$blurExt$	$\{3, 5, 7, 9\}$	determines how far the Gaussian blur can reach
$blurVar$	$[0.5, 2]$	variance of the Gaussian kernel used for blurring

Table 7.3: *Parameters introduced by spatial blurring and their value ranges.*

Motion keeping for future prediction

Our proposed BOFMP, just like BOFUM, is *memory-less*. That is to say, the last step of trakcing stage has absolute influence on future predictions, and the steps before the last

step has no influence at all. In the real data we recorded, a time step equals to 0.25 second in real time. However, this short period of time is not able to summarize human's motion in the past time steps. Therefore, we propose to add moving average velocity of last few steps to the predicted velocity $P\{V_{pred}\}$, and then caculate next velocity $P\{V\}$ from it. This process of incorporating moving average velocitiy is called as *motion keeping*. It starts from the beginning of future prediction stage and the influence of moving average velocity is exponentially decreased over the following time steps.

Motion keeping also introduces new parameters. Assume the measurement is lost since time step t_{lost}, the new parameters are:

1. **window size** w. It determines how many last time steps are considered when caculate moving average velocity .

2. **initial motion factor** $initMF$. This coefficent determines at the beginning of future predicition stage (i.e., $t = t_{lost}$), how much of moving average velocity $P\{V_{ma}\}$ is incorporated into predicted velocity $P\{V_{pred}\}$.

3. **keep motion factor** $keepMF$. This constant is the base of the exponential function used to decrease moving average velocity $P\{V_{ma}\}$ over the following time steps. Therefore, for $t \geq t_{lost}$,

$$factor = initMF \times keepMF^{(t-t_{lost})} \tag{7.1}$$

$$P\{V_{merge}\} = factor \times P\{V_{ma}\} + (1 - factor) \times P\{V_{pred}\} \tag{7.2}$$

The value ranges of these parameters are listed in Table **??**. One example of tracking on real data using motion keeping is shown in Figure 7.8.

	value range
w	$\{2, 4, 6\}$
$initMF$	$[0.3, 0.8]$
$keepMF$	$[0.3, 0.8]$

Table 7.4: *Parameters introduced by motion keeping and their value ranges.*

We randonly sample 100 sets of parameters for each scenario, evaluate them on validation set, and select the best set of parameters based on the mean of average precisions for the future prediction stage ($t = 9 : 16$). The best parameters are shown in Table **??**.

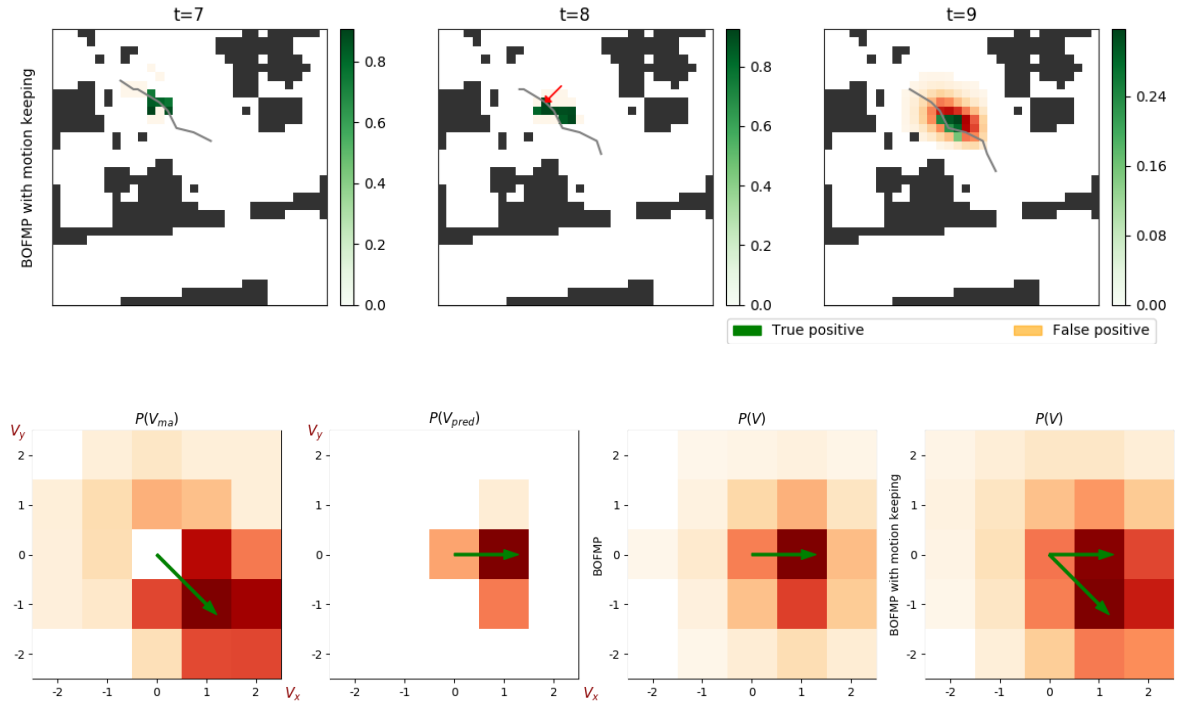

Figure 7.8: ***Up****: Three tracking steps of BOFMP with motion keeping. A person is walking from upper left towards lower right. The measurement is lost at $t = 9$.* ***Down****: Velocities of the cell pointed by the red arrow at $t = 8$. On each plot, the most possible direction is shown by the green arrows. Based on measurements from $t = 7$ and $t = 8$, the predicted velocity $P\{V_{pred}\}$ shows it is more likely the occupancy will propagates towards right for the next time step. However, past motion trend, which is represented by $P\{V_{ma}\}$, shows it is more likely to move to lower right. As a consequnce, BOFMP with motion keeping shows there are possibilities going both right and lower right, which is more realistic in this tracking example.*

	e	δ^2	Ω	$blurExt$	$blurVar$	w	$initMF$	$keepMF$	mean a.p.
BOFUM	5	0.677	0.152	-	-	-	-	-	0.302
BOFMP	5	0.649	0.191	-	-	-	-	-	0.321
BOFMP spatial blurring	5	0.636	0.100	5	1.093	-	-	-	0.327
BOFMP motion keeping	7	0.744	0.026	-	-	4	0.563	0.707	0.381

Table 7.5: *Best parameters for BOFUM and BOFMP on real data.*

Then we apply each filter with its best parameters on test data of 244 tracking cases. Figure 7.9 shows the average precision for each time step, and Table **??** lists the average precision in future prediction stage and their mean. Compared with BOFUM, our methods have performance gain of 29%, 31% and 43% respectively.

	average precision for $t = 9 : 16$	mean
BOFUM	0.670 0.512 0.384 0.314 0.252 0.205 0.167 0.148	0.331
BOFMP	0.705 0.570 0.468 0.424 0.351 0.336 0.305 0.255	0.427
BOFMP spatial blurring	0.694 0.564 0.480 0.439 0.371 0.349 0.311 0.264	0.434
BOFMP motion keeping	0.762 0.675 0.561 0.496 0.405 0.353 0.305 0.238	0.474

Table 7.6: *Future predictions for BOFUM and BOFMP on real data.*

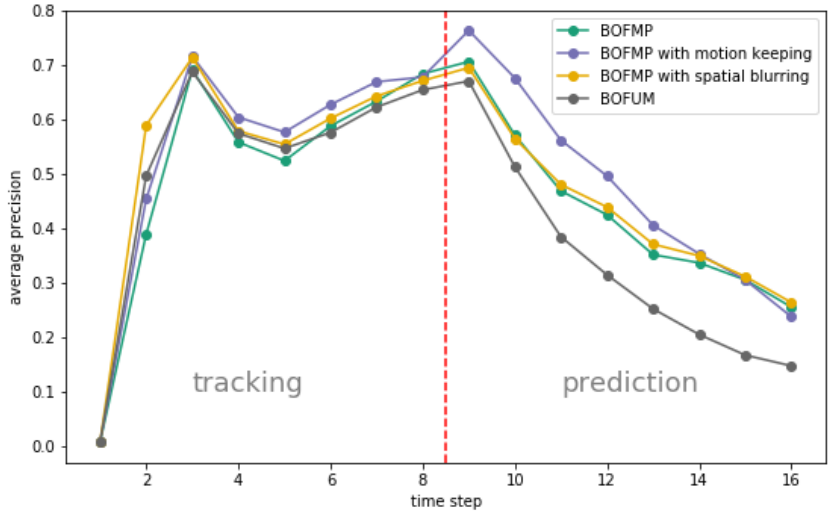

Figure 7.9: *Evaluation results on real test data. Like for simulated data, average precision starts with values close to zero, and increase rapidly over the next two time steps. From $t = 3$ to $t = 8$, average precisions keep rather stable at high values, which proves that filters predict very well for the next immediate step. In future prediction stage ($t = 9 : 16$), average precisions decrease severely as time horizon increases, since the state of the world becomes more uncertain. However, our BOFMP and its variations are still better than BOFUM for every time step in this stage.*

To conclude, we proposed a Bayesian occupancy filter that utilizes human motion pattern for tacking humans in indoor environments. The motion patterns are gained from a neural network which is trained on simulated human trajectories. Our proposed method outputforms BOFUM on both simulated and real tracking cases. One of the most important advantages of our method is that, although the motion pattern is derived from simulated data, it is applicable on real tracking scenarios.

Appendix A

The appendix may contain some listings of source code that has been used for simulations, extensive proofs or any other things that are strongly related to the thesis but not of immediate interest to the reader.

Appendix B

Notation und Abkürzungen

This chapter contains tables where all abbreviations and other notations like mathematical placeholders used in the thesis are listed.

AP	Access Point
CQI	Channel Quality Indicator
DCI	Downlink Control Information
D-SR	Dedicated Scheduling Request
D2D	device to device
eNodeB	evolved Node B or E-UTRAN Node B
FDD	Frequency Division Duplexing
H-ARQ	Hybrid-Automatic Repeat Request
IoT	Internet of Things
LTE	Long Term Evolution
MCS	Modulation and Coding Scheme
OFDM	Orthogonal Frequency Division Multiplexing
PDCCH	Physical Downlink Control Channel
PDSCH	Physical Downlink Shared Channel
PRB	Physical Resource Block
PUCCH	Physical Uplink Control Channel
PUSCH	Physical Uplink Shared Channel
RACH	Random Access Channel
SC-FDMA	Single Carrier Frequency Division Multiple Access
SR	Scheduling Request
SRS	Sounding Reference Signal
TDD	Time Division Duplexing
UE	User Equipment

Bibliography

Arbuckle, D., Howard, A., and Mataric, M. (2002). Temporal occupancy grids: a method for classifying the spatio-temporal properties of the environment. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, pages 409–414. IEEE.

Arras, K. O., Grzonka, S., Luber, M., and Burgard, W. (2008). Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*, pages 1710–1715. IEEE.

Arras, K. O., Mozos, O. M., and Burgard, W. (2007). Using boosted features for the detection of people in 2d range data. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 3402–3407. IEEE.

Biber, P. and Duckett, T. (2009). Experimental analysis of sample-based maps for long-term slam. *The International Journal of Robotics Research*, 28(1):20–33.

Brechtel, S., Gindele, T., and Dillmann, R. (2010). Recursive importance sampling for efficient grid-based occupancy filtering in dynamic environments. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3932–3938. IEEE.

Bruce, A. and Gordon, G. (2004). Better motion prediction for people-tracking. In *Proc. of the Int. Conf. on Robotics & Automation (ICRA), Barcelona, Spain*.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Coué, C., Pradalier, C., Laugier, C., Fraichard, T., and Bessière, P. (2006). Bayesian occupancy filtering for multitarget tracking: an automotive application. *The International Journal of Robotics Research*, 25(1):19–30.

Cui, J., Zha, H., Zhao, H., and Shibasaki, R. (2006). Laser-based interacting people tracking using multi-level observations. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 1799–1804. IEEE.

Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57.

Fod, A., Howard, A., and Mataric, M. (2002). A laser-based people tracker. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 3, pages 3024–3029. IEEE.

Gauvrit, H., Le Cadre, J.-P., and Jauffret, C. (1997). A formulation of multitarget tracking as an incomplete data problem. *IEEE Transactions on Aerospace and Electronic Systems*, 33(4):1242–1257.

Gindele, T., Brechtel, S., Schroder, J., and Dillmann, R. (2009). Bayesian occupancy grid filter for dynamic environments using prior map knowledge. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 669–676. IEEE.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Huang, G., Liu, Z., Weinberger, K. Q., and van der Maaten, L. (2016). Densely connected convolutional networks. *arXiv preprint arXiv:1608.06993*.

Jégou, S., Drozdzal, M., Vazquez, D., Romero, A., and Bengio, Y. (2017). The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2017 IEEE Conference on*, pages 1175–1183. IEEE.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Kucner, T., Saarinen, J., Magnusson, M., and Lilienthal, A. J. (2013). Conditional transition maps: Learning motion patterns in dynamic environments. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1196–1201. IEEE.

Levine, S., Pastor, P., Krizhevsky, A., Ibarz, J., and Quillen, D. (2016). Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, page 0278364917710318.

Liao, L., Fox, D., Hightower, J., Kautz, H., and Schulz, D. (2003). Voronoi tracking: Location estimation using sparse and noisy sensor data. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 1, pages 723–728. IEEE.

Llamazares, A., Ivan, V., Molinos, E., Ocana, M., and Vijayakumar, S. (2013). Dynamic obstacle avoidance using bayesian occupancy filter and approximate inference. *Sensors*, 13(3):2929–2944.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440.

Luber, M., Diego Tipaldi, G., and Arras, K. O. (2011). Place-dependent people tracking. *The International Journal of Robotics Research*, 30(3):280–293.

Meier, E. B. and Ade, F. (1999). Using the condensation algorithm to implement tracking for mobile robots. In *Advanced Mobile Robots, 1999.(Eurobot'99) 1999 Third European Workshop on*, pages 73–80. IEEE.

Meyer-Delius, D., Beinhofer, M., and Burgard, W. (2012). Occupancy grid models for robot mapping in changing environments. In *AAAI*.

Montemerlo, M., Thrun, S., and Whittaker, W. (2002). Conditional particle filters for simultaneous mobile robot localization and people-tracking. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 1, pages 695–701. IEEE.

Ross, D. A., Lim, J., Lin, R.-S., and Yang, M.-H. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1):125–141.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252.

Schulz, D., Burgard, W., Fox, D., and Cremers, A. B. (2001). Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 2, pages 1665–1670. IEEE.

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Wang, D. Z., Posner, I., and Newman, P. (2015). Model-free detection and tracking of dynamic objects with 2d lidar. *The International Journal of Robotics Research*, 34(7):1039–1063.

Wang, Z., Ambrus, R., Jensfelt, P., and Folkesson, J. (2014). Modeling motion patterns of dynamic objects by iohmm. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 1832–1838. IEEE.