Mobile Robot Control and Navigation in Public Spaces

by

Chengyuan Yan

Submitted to the Department of Mechanical Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2015

Author ..... Signature redacted
Department of Mechanical Engineering
May 8, 2015

Certified by ..... Signature redacted
H. Harry Asada
Ford Professor of Engineering
Thesis Supervisor

Accepted by ..... Signature redacted
David E. Hardt
Chairman, Department Committee on Graduate Theses

Mobile Robot Control and Navigation in Public Spaces

by

Chengyuan Yan

Submitted to the Department of Mechanical Engineering
on May 8, 2015, in partial fulfillment of the
requirements for the degree of
Master of Science in Mechanical Engineering

Abstract

Public service robots have been more and more popular due to their social and economical influences. This thesis investigated two issues about control and navigation of public service robots deployed in crowded environments such as airports and shopping malls. Our solutions facilitates the synergy of a distributed sensor network and a central computer. A distributed sensor network provides multi-view measurements without occlusion, and a central computer can use these data to estimate the state of everything in the environment in real-time.

A public service robot should have human-like mobility, but a wheeled robot is vulnerable to fall when it transits to escalators/moving walkways that are commonly seen in public places. A compliant coupler is inserted between the wheel and the drive motor, which could block instantaneous impacts during the transition. A feedback control is designed to improve the transmission system's damping and regulate the robot's ground speed. A simulated robot was able to transit between ground and moving walkways smoothly using the series elastic transmission and unified velocity control.

To help a public service robot reach its destination efficiently without causing much annoyance to nearby humans, we developed a three-layer hierarchical path planner. Every layer plans at a different temporal and spatial scale, and the plans are refined as they are passed from top level to the bottom level. The intermediate level planner bridges global path optimality and local path optimality, and is discussed in detail. Using a fluid analogy, the medium planner treats individual passengers as fluid particles, and tries to find a path so that the total pressure received is minimized. Navigation maps are introduced as an augmentation to navigation functions, which indicate the shortest path towards goal to a robot. Using a finite-horizon optimization, the medium planner can react to the dynamic crowd promptly. Simulations show that the planner is able to plan appropriate paths in many different scenarios.

Thesis Supervisor: H. Harry Asada
Title: Ford Professor of Engineering

Acknowledgments

I am more than grateful to Professor H. Harry Asada for the help and guidance that were necessary to finish this thesis. I also wish to express my sincere thanks to East Japan Railway Company, who sponsored the project and created the opportunity for me to investigate these fascinating problems. Federico Parietti, Alexandre Girard, Baldin Llorens âĂŞ Bonilla and James Torres from d'Arbeloff Laboratory at MIT provided informative feedback and hands-on assistance to my research. Jun Maruyama and Takeshi Saito from JR East inspired me a lot in our discussion, and I owe many interesting ideas to them. Finally I would like to express my deepest appreciation to my parents and my girlfriend, whose care and support have always been most important to me.

Public service robots have been more and more popular due to their social and economical influences. This thesis investigated two issues about control and navigation of public service robots deployed in crowded environments such as airports and shopping malls. Our solutions facilitates the synergy of a distributed sensor network and a central computer. A distributed sensor network provides multi-view measurements without occlusion, and a central computer can use these data to estimate the state of everything in the environment in real-time.

A public service robot should have human-like mobility, but a wheeled robot is vulnerable to fall when it transits to escalators/moving walkways that are commonly seen in public places. A compliant coupler is inserted between the wheel and the drive motor, which could block instantaneous impacts during the transition. A feedback control is designed to improve the transmission system's damping and regulate the robot's ground speed. A simulated robot was able to transit between ground and moving walkways smoothly using the series elastic transmission and unified velocity control.

To help a public service robot reach its destination efficiently without causing much annoyance to nearby humans, we developed a three-layer hierarchical path planner. Every layer plans at a different temporal and spatial scale, and the plans are refined as they are passed from top level to the bottom level. The intermediate level planner bridges global path optimality and local path optimality, and is discussed in detail. Using a fluid analogy, the medium planner treats individual passengers as fluid particles, and tries to find a path so that the total pressure received is minimized. Navigation maps are introduced as an augmentation to navigation functions, which indicate the shortest path towards goal to a robot. Using a finite-horizon optimization, the medium planner can react to the dynamic crowd promptly. Simulations show that the planner is able to plan appropriate paths in many different scenarios.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

People have been actively exploring the application of robots in non-industrial scenarios over the past few decades. Probably the best known example is Roomba, an autonomous vacuum cleaner by iRobot, used by millions of families [10]. Besides home appliances, other successful trials include robotic guides to museums [1] and shopping malls [11], companion robots [6, 18] and humanoid robots[19].

Public service robots are those deployed in public environments to satisfy the increasing need for high-quality daily services. They are of particular interest because of their potential social and economical impacts. Compared with home robots, public service robots are unique in the following aspects: [2]

- The environments they operate in tend to be more structured. For example, homes can be very different in layout and design, while shopping malls often have similar facilities and functionalities.

- Usually the initial installation costs are less important, and operational costs are more crucial due to the long operation hours.

- They can concentrate on some specific tasks. For instance, a museum guide robot only focuses on navigation and explanation. On the other hand peo-

ple may expect a home robot with manipulator arms to be versatile for many services.

The similarities in the workspace makes it possible to design a robot and deploy it in numerous places. Higher initial setup budget allows for sensors embedded in the environments and more advanced technologies. The concrete objectives lead to more clearly defined market. These specialties make public service robots more likely to be prototyped and commercialized.

Yet there are many challenges faced by public service robots, such as

- Safety issues. Public places are occupied by both humans and robots, and collisions are risky and unacceptable.
- Sensor obstructions. The passengers are like dynamic obstacles, which may block the photoelectric sensors and ultrasonic sensors.
- Human-like accessibility. A public service robot has to adapt to environments designed for human. For example, it should be able to follow its human user, even if he/she is using the escalator.
- Human-machine interactions. A robot interacting with humans should respect their conventions and avoid being confusing or aggressive. For example, it is reported that robots approaching people from certain directions are very unwelcome[16].
- Scalability. A typical robot works alone, and it only needs to observe a small vicinity. However, there may be hundreds of public service robots operating in the same environment, and much computational power is required for scheduling and world modeling.

Special attention should be paid to service robots in busy public transportation junctions such as train stations and airports. These junctions are usually multi-story, consisting of many concourses and platforms. Besides, they are often integrated with shopping and dining areas, which brings in even more people. The spacial complexity

and highly dynamic crowds render it more difficult to control and navigate robots in those junctions. However, despite the growing need for public service robots in busy transportation junctions, few researches have been done on that matter. This thesis aims to develop solutions to some issues of public service robots in crowded environments, with the help of a distributed sensor network and a connected central server.

1.2 Related Works

An autonomous robotic tour-guide for museums is presented in [1], which features interactive human-machine interface. To deal with uncertainty in the environment and measurements, probabilistic representations are incorporated for perception, reasoning and planning. An entropy gain filter is devised to remove invalid sensor measurements caused by obstruction, so that Markov localization can be applied even in the presence of many people. The filtered out data are then used to estimate the position of nearby people, which is integrated with the floor plan to produce an instantaneous occupancy-grid map. A modified dynamic window approach implements local collision avoidance, while global minimum-cost paths are achieved by dynamic programming, both using the instantaneous occupancy grids. The software system on the robot is modular, decentralized and asynchronous. Time-critical modules such as collision avoidance are run in parallel on-board, while computation-intensive tasks such as path planning are carried out by off-board workstations. In the trial deployment in a normally-operating museum, the robotic tour-guide worked for 47 hours continuously, traveling a total of $18.6km$ and serving more than 2400 visitors, with only six mild collisions.

[17] describes a robotic wheelchair name MAid, whose task is to transport mobility-impaired people in crowded public environments. It receives commands from users and overwrites those that may result in collisions. An on-board laser range finder is used to detect surrounding objects. By comparing sequential observations, the velocities of nearby objects could be estimated, which are extrapolated to predict future

trajectories. The wheelchair's maneuvers that would result in collision are identified and removed, and an action that optimizes a given metric is chosen among the valid reachable velocities, which is similar to the dynamic window approach. MAid has been successfully tested in crowded railway stations and exhibition halls, and when the traffic is too heavy, it would stop and wait for the people to pass.

In the presence of sensors fixed in the environment, the accuracy of robot localization and human tracking can be improved[7]. The external sensors are usually mounted high enough to avoid obstructions and provide larger fields of view. They also serve as fixed position references, which is crucial in decreasing estimation covariance. An association-correction method is developed to fuse the measurements from both onboard and offboard sensors. Simulations show that the method yields much smaller localization error, and field tests demonstrate the system's ability to recover quickly from tracking errors.

When planning paths, a robot should take into consideration the velocity constraints attached to some regions, as suggested by [12]. In the "regions with velocity constraints" (RVC's), the robot is restricted to a velocity predefined by human operator, so as to avoid danger and minimize total traveling time. A new discrepancy-grid map is generated for every instant, based on which a trajectory satisfying the velocity constraints are generated. This method is verified with robots running along real public paths with humans.

A finite-state-machine motion planner is introduced in [23]. The onboard sensor readings are converted to the input alphabet and fed to a finite state-machine. The output from the state-machine is converted to linear and angular velocities, which are used to control the robot. In the first conversion, a coarse occupancy-grid map is obtained by projecting the sensor readings onto the robot's neighborhood. The weighted occupancy comprises the input alphabet, which in used to look up the next state in a transition table. In the test runs in shopping malls and congress centers, the robot usually changes its state to "pause" to wait for a moment when there is more space to move.

1.3 Fundamental Assumptions

The methods developed in this thesis assumes that there are many sensors installed in the public environment, which constitutes a distributed sensor network, and that a central server is connected to the sensor network. The central server processes the data from the sensor network and estimates in real-time the states of all passengers in the environment.

This assumption of distributed sensor network is not unrealistic, because it is common to see airports and train stations installed with many surveillance cameras. Other than the surveillance cameras, ticket gates, designated laser range finders and deployed robots all provide adequate information about the environment. The distributed sensor network is advantageous in that an object is usually detected by many sensor, which rules out the chances of occlusions. The estimation accuracy can be boosted as a consequence of multiple-view object recognition.

The central server could be seen as an augmentation to the inspection and data storage nodes in existing surveillance camera systems. It analyses the plentiful data collected by the sensor network, and produces real-time estimates of everything in the environment. It could also identifies statistical motion models of passengers from past data. These models play important parts in estimating and predicting the passengers states. The colossal computing power the central server possesses render it possible to perform centralized scheduling and planning for all deployed robots, thus providing the scalability needed.

The synergy of the central server and the distributed sensor network is remarkably influential to the control and navigation of robots in crowded public environments, as we will describe with details in the following chapters.

1.4 Thesis Organization

This thesis studies the control and navigation of a kind of service robot designed for highly populated public environments. The work assumes that a distributed sen-

sor network is available, and the sensors are connected to a central computer that is capable of integrating all measurements and estimating the states of the entire public environment. Furthermore, any robot deployed in the public environment has access to the real-time estimates as well as the computing power of the central computer. Two engineering challenges for public service robots are stressed and tackled facilitating the above assumptions.

The first chapter gives the background and motivation of the thesis. Some related works on public service robots are surveyed, whose perks and inadequacies are pointed out. The fundamental assumptions of this thesis are established, including a distributed sensor network that monitors the public place and a central server that estimates the states of the environment in real-time The synergy of the distributed sensor network and the central server is briefly discussed.

The second chapter investigates the transition of a robot between normal floor and moving walkways. The disturbance caused by the transition is modeled as an instantaneous external input to the system, which is clearly separated from the normal inputs in the frequency domain. Series elasticity is introduced to the transmission system to absorb the high-frequency disturbance. A velocity controller is devised to increase the transmission's damping and regulate the robot's ground speed.

The third chapter focuses on navigating a robot in the presence of crowds. A three-layer online planner is proposed for path planning in uncertain dynamic environments. The Medium Planner, a novel layer that bridges the global optimality and local optimality, is studied in detail. Two concepts are put forward to formulate the medium-range path planning as a finite-horizon optimization. To solve the two-point boundary-value optimization, the structures of the problem is examined and iterative methods are discovered to approximate the solution.

The last chapter summarizes the work and lists a few remarks on possible extensions of the work.

Chapter 2

Smooth Transition

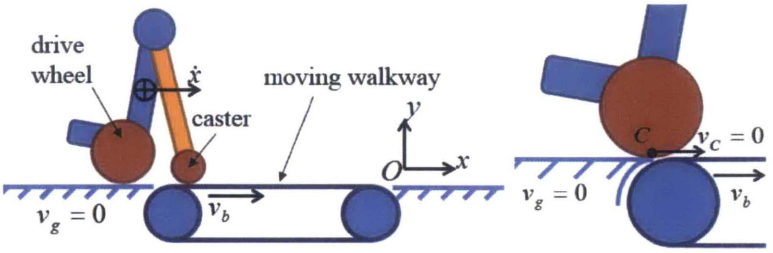
Public service robots operate in environments that are designed primarily for human. They have to possess human-like mobility in order to deliver expected services. However, most mobile robots today are wheeled, which are unable to ride on escalators and moving walkways that are commonly seen in public transportation junctions. This chapter presents a solution for wheeled robots to transit smoothly between ground and moving walkways. The solution could be easily altered to work for escalators.

2.1 Design Concepts

The overarching goal of the thesis work is to deploy mobile robots for services in public areas. Robots should be able to move around without obstructing surrounding passengers. This requires the footprint, or the area occupied by the robot, to be as small as a human. Fig.2-1a illustrates a wheeled mobile robot with a small footprint. Having both front and rear wheels, the robot is statically stable. Yet the stability margin is rather narrow, since the distance between the front and the rear wheels is short, compared with the high center of mass. Transition to and from moving walkways or escalators is challenging for this type of mobile robots.

As shown in Fig.2-1b, the floor speed changes discretely as the robot steps on and off the moving walk. When the powered wheel arrives at the edge of the stationary floor, the velocity of the contacting point C on the wheel is zero, assuming no slip

on the floor. On the other hand, when the powered wheel arrives at the edge of the moving walkway, the velocity of the contacting point on the moving walkway is not zero. The velocity discrepancy between the contacting points on the wheel and the moving walkway makes the wheel slide on the moving walkway. The sliding friction gives an impact to the wheel and the rest of the robot body, which may cause the robot to stagger, stumble, or fall over. Similarly, the robot is suddenly stopped at the end of the moving walkway, as it transits back to the ground.



(a) overview (b) close-up

Figure 2-1: Transition between the ground and a moving walkway

The impact that acts on the robot during the transition can be characterized by its short duration and large magnitude. Preliminary calculation reveal that the duration is on the order of $10^{-2}s$, and the sliding-induced impact is proportional to the robot's weight. The wheel velocity control system is unable to react to such an instantaneous disturbance. However, in the frequency domain there is a clear separation between characteristic frequency of the impact and that of the wheel control system. Fig.2-2 illustrates the separation in the frequency spectrum, where ω_{cr} is the critical frequency. Since the wheel velocity control system is unable to respond to such a high-frequency disturbance, it is reasonable to consider a passive damper to absorb the impact. Nonetheless, it should be noted that the impact in this case acts in the horizontal direction along the floor. Unlike traditional shock absorbers of automobiles and mobile robots, which suppress impacts normal to the ground, this shock absorber has to block the horizontal impact along the tangential direction of the wheel.

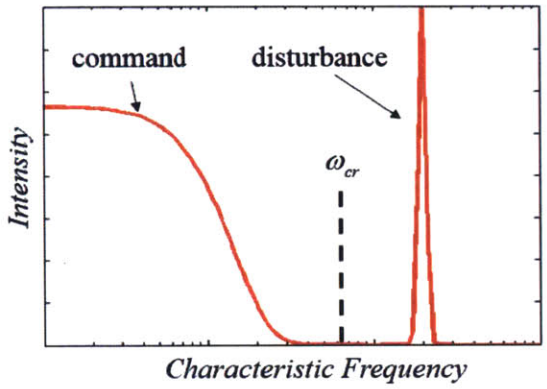


Figure 2-2: Frequency spectrum of the input signal

Here we propose to use series elasticity in the transmission to absorb the tangential impact. See Fig.2-3, a torsional spring is inserted between the wheel axle and the actuator shaft. The torsional spring serves as a mechanical low-pass filter, so that high-frequency signals would be blocked. When the wheel steps on or off the moving walk, the wheel is forced to accelerate or decelerate by the impact Because of the elastic coupler, the wheel is virtually disconnected from the rest of the robot at high frequencies. It is then free to rotate for a short period of time, while the momentum of the robot body forces the body to move forward during the transition.

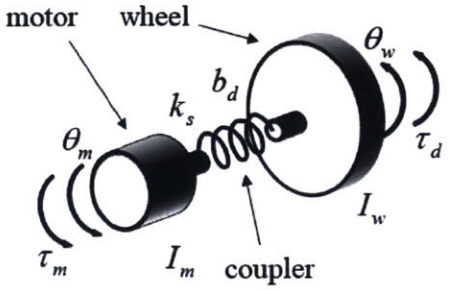


Figure 2-3: Compliant coupler in the transmission

Impact absorption is not the only function needed for this mobile robot. Once the robot gets on the moving walk, its wheel speed must drop to zero to stop relative motion, which might lead to collisions with surrounding passengers. As the robot gets off the moving walk, its wheel speed must rev up so that the robot can continue to move at the same ground speed, for the same reason of collision avoidance. The wheel velocity control system must kick in at the right moment in response to the

abrupt change of the floor speed. The elastic coupler can facilitate to detect the impact-induced wheel rotation and form a feedback control loop that regulates the wheel speed. As shown in Fig.2-3, by measuring the rotation angles of the wheel axle and the actuator shaft, we can compute the transmitted torque with the help of Hook's Law. This provides the robot control system with useful information for

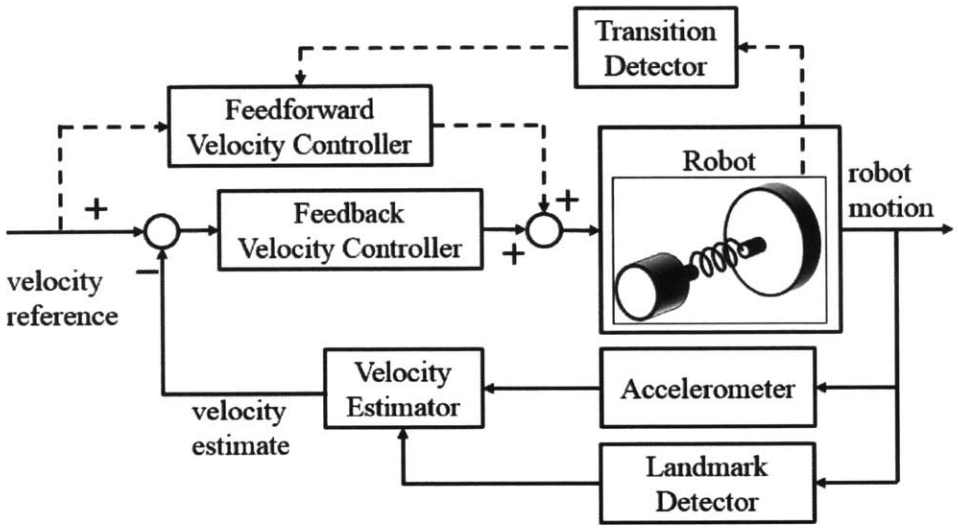
- Regulating the wheel speed;
- Detecting the discrete change of the floor speed, so that the reference input to the wheel speed control system can be switched as soon as the floor speed has changed; and
- Improving the dynamics of the wheel control system based on the measurement of the transmitted torque and the phase lag between the motor and the wheel axles.

The following sections will explore these features in details.

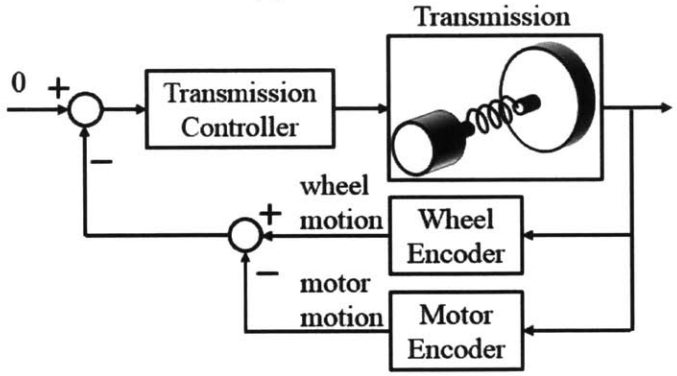
2.2 Dynamics Analysis and Control Synthesis

The design concepts described in the previous section are depicted in Fig. 2-4. The robot is equipped with series elasticity to absorb impacts in the tangential direction of the wheels. The stiffness of elastic coupler is chosen to reject high-frequency disturbances while maintaining the command signals. A feedback control is designed to cooperate with the series elasticity in the transmissions. The outer loop of the control, which regulates the robot's ground speed, is closed with a velocity estimator. The estimator estimates the robot's velocity by synthesizing the readings from the accelerometer and the landmark detector. The dashed paths in the upper part constitutes the feedforward loop, which is proposed to speed up the response of the robot when it is running on the ground. The "Transition Detector" recognizes the discrete change of the floor speed, using the angular positions of the motor shafts and the wheel axles. The "Feedforward Velocity Controller" will output zero when the robot is on the moving walkway, which can be easily determined by counting the

total number of transitions. The inner loop of the control is designed to increase the damping and suppress the oscillations in the transmission induced by the impacts.



(a) Outer loop



(b) Inner loop

Figure 2-4: Block diagram of the Unified Velocity Control

Before moving on to detailed dynamics analysis and control synthesis, first we list the nomenclature in Table 2.1, whose visual explanations can be found in Fig.2-3.

2.2.1 Series Elastic Transmission

The block diagram of the Series Elastic Transmission is shown in Fig. 2-5. A compliant coupler connects the shaft of a geared motor and the wheel axle. The coupler is represented with the Kelvin-Voigt model, which consists of a spring and a damper in

Table 2.1: Nomenclature of Series Elastic Transmission

Symbol	Description
θ_m	angular position of the motor
θ_w	angular position of the wheel
I_m	moment of inertia of the motor
I_w	moment of inertia of the wheel
τ_d	external disturbance torque
τ_m	motor torque

parallel. There are two encoders measuring the angular positions of the motor shaft and wheel axle, respectively.

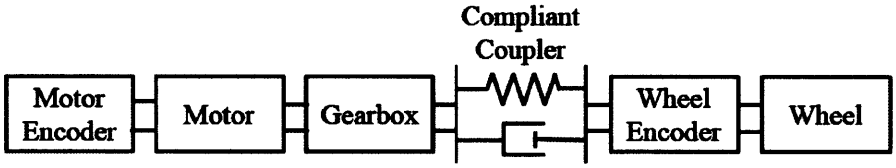


Figure 2-5: Block diagram of Series Elastic Transmission

The equations of motion of the transmission system can be easily obtained as

$$\begin{cases} \tau + \tau_d &= I_w \ddot{\theta}_w \\ \tau &= k_s(\theta_m - \theta_w) + b_d(\dot{\theta}_m - \dot{\theta}_w) \\ \tau_m - \tau &= I_m \ddot{\theta}_m \end{cases}, \tag{2.1}$$

where τ is the torque transmitted through the coupler.

Take the Laplace transform of both sides, and we find the characteristic equation of the open-loop transmission system as

$$D^{OL}(s) = s^2[I_m I_w s^2 + (I_m + I_w)b_d s + (I_m + I_w)k_s]. \tag{2.2}$$

The characteristic equation (2.2) reveals that the open-loop transmission system has two repeated poles at the origin, and two complex poles whose natural frequency ω_n^{OL}

and damping ratio ζ^{OL} are given by

$$\omega_n^{OL} = \sqrt{\frac{k_s(I_m + I_w)}{I_m I_w}}, \tag{2.3a}$$

$$\zeta^{OL} = \frac{b_d(I_m + I_w)}{2I_m I_w \omega_n}. \tag{2.3b}$$

The compliant coupler is made from visco-elastic materials, which have relatively small damping, $b_d \approx 0$. Therefore the transmission system's break frequency ω_b can be approximated to

$$\omega_b \approx \omega_n^{OL}. \tag{2.4}$$

In order to suppress high-frequency disturbances, we choose $\omega_b = \omega_{cr}$, the critical frequency that separates the user commands and the impacts. The desired stiffness of the compliant coupler is then

$$k_s = \frac{\omega_{cr}^2 I_m I_w}{I_m + I_w}. \tag{2.5}$$

With the stiffness derived, there is no design freedom in the damping coefficient b_d, because it is connected with the stiffness as a consequence of the material property and coupler geometry.

2.2.2 Unified Velocity Control

The Unified Velocity Control improves the damping of the transmission system, and regulates the robot's ground speed at the same time. We will first carry out the control synthesis using the angular coordinates of the wheel and the motor, and then rewrite it using Cartesian coordinates under the no-slip assumption. Strictly speaking, the wheel does slip on the floor/moving walkway during the transitions. However, in simulations and prototype experiments the control works robustly despite the invalid assumption.

We simplify the control in Fig. 2-4 to a linear paradigm shown in Fig. 2-6, where

$G_w(s)$ is the transfer function from the motor torque to wheel motion, and $G_m(s)$ from the motor torque to motor motion:

$$G_w(s) = \frac{\theta_w(s)}{\tau_m(s)} = \frac{b_d s + k_s}{D_T(s)}, \tag{2.6a}$$

$$G_m(s) = \frac{\theta_m(s)}{\tau_m(s)} = \frac{I_w s^2 + b_d s + k_s}{D_T(s)}. \tag{2.6b}$$

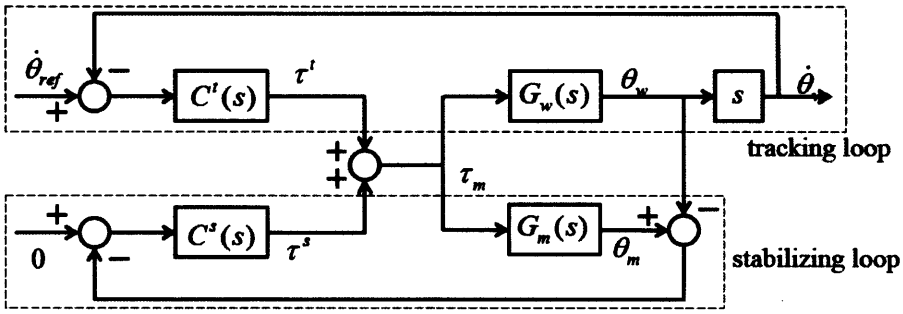


Figure 2-6: Block diagram for Unified Velocity Control

The tracking loop in the upper part realizes the outer loop in Fig. 2-4a, which aims to regulate the robot's ground speed. The stabilizing loop in the lower part implements to the inner loop in Fig. 2-4b, which improves the damping of the transmission system. $C^t(s)$ and $C^s(s)$ are the tracking controller and stabilizing controller, respectively. The motor torque τ_m, according to the principle of superposition, is computed as the sum of the tracking torque τ^t and the stabilizing torque τ^s,

$$\tau_m = \tau^t + \tau^s \tag{2.7a}$$

$$\tau^t = C^t(s)(\dot{\theta}_{ref} - \dot{\theta}_w) \tag{2.7b}$$

$$\tau^s = C^s(s)(0 - \theta_m) \tag{2.7c}$$

Noting that the reference for the stabilization loop is constant zero, we can combine the stabilizing controller $C^s(s)$ with the transmission system $G_w(s)$ and $G_m(s)$, and

obtain an SISO system as shown in Fig. 2-7. The closed-loop transfer function is

$$T^{CL} = \frac{\dot{\theta}_w}{\dot{\theta}_{ref}} = \frac{C^t G}{1 + C^t G}, \tag{2.8}$$

where $G(s)$ is the lumped dynamics of the transmission system and the stabilizing controller,

$$G(s) = \frac{\dot{\theta}_w}{\tau^t} = \frac{sG_w}{1 + C^s(G_m - G_w)} \tag{2.9}$$

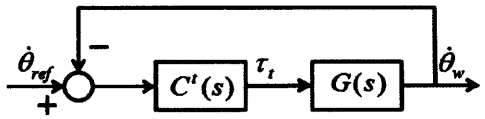


Figure 2-7: Simplified block diagram for the Unified Velocity Control

Applying proportional-derivative controller to both loops

$$C^t(s) = K_P^t + K_d^t s, \tag{2.10a}$$

$$C^s(s) = K_P^s + K_D^s s, \tag{2.10b}$$

and noting that $\dot{\theta}_{ref}$ is constant, the lumped dynamics $G(s)$ becomes

$$G(s) = \frac{s(b_d s + k_s)}{D^{CL}(s)}, \tag{2.11}$$

where

$$D^{CL}(s) = D^{OL}(s) + s^2(I_m K_D^s s + I_w K_P^s) \tag{2.12}$$

is the characteristic equation of the closed-loop system. The two oscillatory poles now have natural frequency ω_n^{CL} and damping ratio ζ^{CL}:

$$\omega_n^{CL} = \sqrt{\frac{k_s(I_m + I_w) + K_P^s I_w}{I_m I_w}}, \tag{2.13a}$$

$$\zeta^{CL} = \frac{b_d(I_m + I_w) + K_D^s I_m}{2I_m I_w \omega_n} \approx \frac{K_D^s}{2I_m \omega_n}. \tag{2.13b}$$

The approximation 2.13b comes from the fact that $b_d \approx 0$.

Comparing the open-loop pole positions (2.3) with the closed-loop pole positions (2.13), we can draw some conclusions about the stabilizing controller gains as follows.

K_P^s is similar to the stiffness of the compliant coupler. K_P^s increases the natural frequency of the system, which degrades the intrinsic low-pass property of the Series Elastic Transmission. In (2.5) the stiffness of the compliant coupler has been designed to achieve frequency separation, therefore we can simply choose

$$K_P^s = 0. \tag{2.14}$$

K_D^s is similar to the damping coefficient of the compliant coupler. Because $b_d \approx 0$, the damping of the system comes mainly from K_D^s. By increasing K_D^s we can bring the poorly damped poles to the left, thereby improving the closed-loop transmission system's damping. However, if K_D^s is too large, the two complex poles will meet and become real, which might deteriorate the dynamic response. The pole trajectories with respect to K_D^s are plotted in Fig. 2-8. Note the two repeated poles at the origin do not move. If desired damping ratio ζ_d is given, then K_D^s can be roughly computed

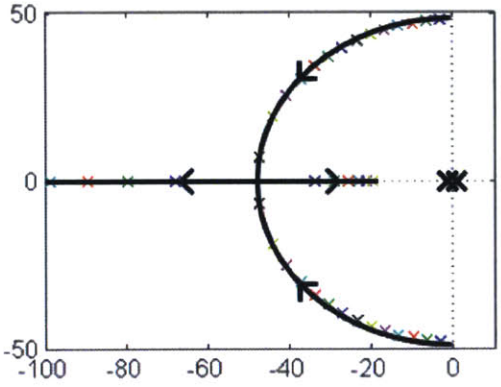


Figure 2-8: Pole trajectories with varying K_D^s

as

$$K_D^s = 2\zeta_d I_m \omega_n^{CL}. \tag{2.15}$$

With K_P^s and K_P^s determined, $G(s)$, the lumped dynamics of the transmission system and the stabilizing c, now has two complex poles, two repeated poles at the

origin and two zeros. One zero originates from the coupler's material viscosity, and the other zero sits at the origin. The relative degree of $G(s)$ is three, which means that a proportional-derivative controller like (2.10a) could provide guaranteed stability. Selecting the gains for the tracking controller $C^t(s)$ is simple and not discussed here.

Now we convert the Unified Velocity Control from angular coordinates to Cartesian coordinates. Assuming no slip between the wheel and floor/moving walkway, the robot's ground speed $\dot{x}$ is related to the wheel speed $\dot{\theta}_w$ by $\dot{x} = -R\dot{\theta}_w$, and the reference velocity for the wheel $\dot{\theta}_{ref}$ is related to the reference ground speed v_{ref} by $\dot{\theta}_{ref} = -v_{ref}/R$. Substituting $\dot{\theta}_{ref}$, $\dot{\theta}_w$ and $\ddot{\theta}_w$ in (2.7) and (2.10) with v_{ref}, $\dot{x}$ and $\ddot{x}$, we arrive at the following control law,

$$\tau_m = K_P(v_{ref} - \dot{x}) + K_D(-\ddot{x}) + K_P^s(\theta_w - \theta_m) + K_D^s(\dot{\theta}_w - \dot{\theta}_m), \tag{2.16}$$

where $K_P = -K_P^t/R$ and $K_D = -K_D^t/R$.

2.2.3 Velocity Estimator

In a crowded environment, it is very important for a robot to maintain a constant speed, otherwise it may bump into nearby humans. We propose the Unified Velocity Control to regulate the robot's ground speed and improve the transmission system's damping. However, the loop cannot be closed without velocity feedback. Here we briefly discuss a velocity estimator that is backed by the setup of a distributed sensor network and a central server.

The sensor network consists of not only active sensors such as cameras, but also static landmarks that are visible only to robots. These landmarks encode location information to assist indoor localization. A robot has a landmark detector that could identify these landmarks and parse the location information carried by them, which is referred to as global localization. Moreover, a landmark detector is also able to achieve local localization by measuring the distance and angle between a robot and a landmark. The robot can perform indoor localization using the global information and local information. Examples for such landmark detectors include ultrasonic range

devices[3] and visual fiducial systems[15]. Yet it should be noticed that the landmarks are sparsely installed in the environment, and a robot may only find landmarks once in a while. These low-rate position measurements are insufficient to supply the velocity feedback needed a closed-loop control.

The remedy is to fuse the position measurements with acceleration measurements, as depicted in An onboard inertial measurement unit could measure a robot's acceleration at a high rate. Despite that the acceleration measurements are depreciated by noise and bias, integrating them with occasional location measurements using a multi-rate Kalman filter yields good velocity estimate. Details about formulating of the Kalman filter can be found in[21].

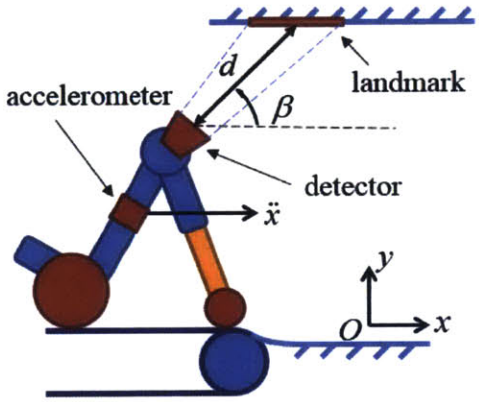
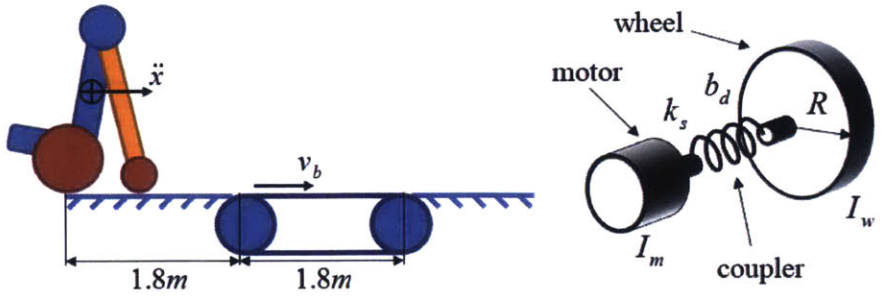


Figure 2-9: Illustration of ground speed estimation.

2.3 Simulation

We perform simulations to examine the performance of the Series Elastic Transmission and the Unified Velocity Control. The robot starts from rest on the ground and moves forward. It steps onto a moving walkway, keeps advancing and finally transits back to the ground. Fig. 2-10a depicts the scenario. The robot's initial position is $1.8m$ away from the moving walkway. The length of the moving walkway is $1.8m$, and its speed is $v_b = 0.3m/s$. Throughout the procedure, the Unified Velocity Control regulates the robot's ground speed to a reference velocity, $v_{ref} = 0.3m/s$.



(a) scenario (b) configuration

Figure 2-10: Simulation setup

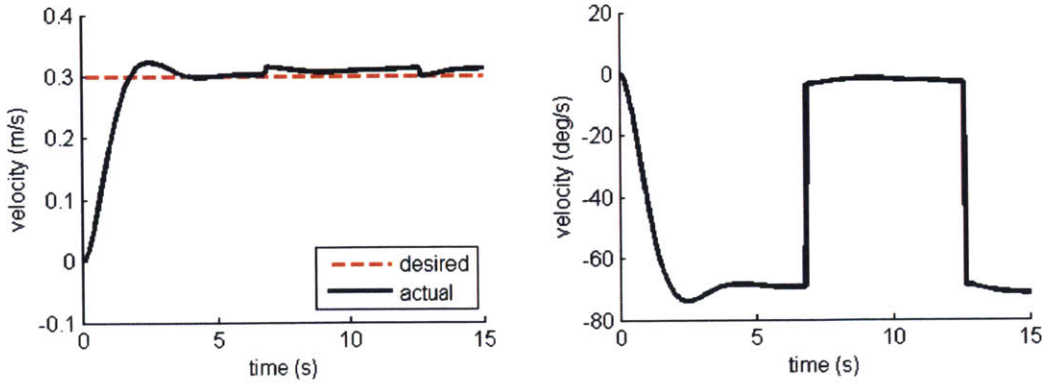
Fig. 2-10b shows the configuration of the Series Elastic Transmission. The wheel radius is $R = 0.25m$. The inertia of the gear motor is $I_m = 0.765kgm^2$, and the inertial of the wheel is $I_w = 0.135kgm^2$. The mass of the wheel is $m_w = 5kg$, and the rest of the robot has mass $m_b = 45kg$. The critical frequency that separates the commands and transition impacts is $\omega_{cr} = 10Hz$. Referring to (2.5), the desired stiffness of the compliant coupler is $k_s = 113.25Nm/rad$, and as a consequence, the damping coefficient is $b_d = 0.566kgm/s$.

The ground speed is estimated by integrating the accelerometer measurements. The accelerometer is modeled with additive band-limited white noise. We apply the following friction model

$$f = \mu(m_w + m_b)g\tanh(500V_r), \tag{2.17}$$

where $\mu = 0.7$ is the friction coefficient; $g = 9.81m/s^2$ is the gravitational acceleration; V_s is the slip velocity between the tire and the floor/moving walkway.

The simulation results are plotted in Fig. 2-11. Fig. 2-11a shows that the robot's ground speed converges to the reference velocity in less than four seconds. There is a small step at $t \approx 7s$, indicating the transition from the ground to the moving walkway. Another small step at $t \approx 12.7s$ represents the transition from the moving walkway to the ground. In Fig. 2-11b, soon after the robot gets on to the moving walkway, its wheel stops rotation. The wheel's angular speed is almost zeros when the robot is on the moving walkway, which means that the robot is being carried



(a) robot ground speed (b) wheel angular speed

Figure 2-11: Simulation results

forward by the moving walkway. As the robot transits back to the ground, the wheel starts rotating again. The robot's ground speed is maintained well during the two transitions.

2.4 Prototype Experiment

Experiments with a proof-of-concept robot are conducted to verify the simulation results. We build the prototype robot with 80/20 aluminum frames, and install Series Elastic Transmission to drive the two rear wheels. The two front caster wheels are for static support only. The test track is designed to replicate the simulation, which has a mock moving-walk between two fixed platforms. See Fig. 2-12. The robot is commanded to move with $v_{ref} = 0.3m/s$, which is the same velocity as the moving walkway's velocity v_b. The robot's ground speed is estimated with an IMU and an ultrasonic range finder.

See Fig. 2-13 for the break-down of a Series Elastic Transmission. Every transmission has a Maxon EC-90 brushless motor, a HarmonicDrive 50:1 gearbox and a US Digital 10000 PPR encoder. The compliant coupler consists of four pieces of polyurethane, with connections to both the motor shaft and the wheel axle. The wheel radius is $R = 0.25m$. The inertia of the gear motor is $I_m = 0.765kgm^2$, and the inertial of the wheel is $I_w = 0.135kgm^2$. The critical frequency that sepa-

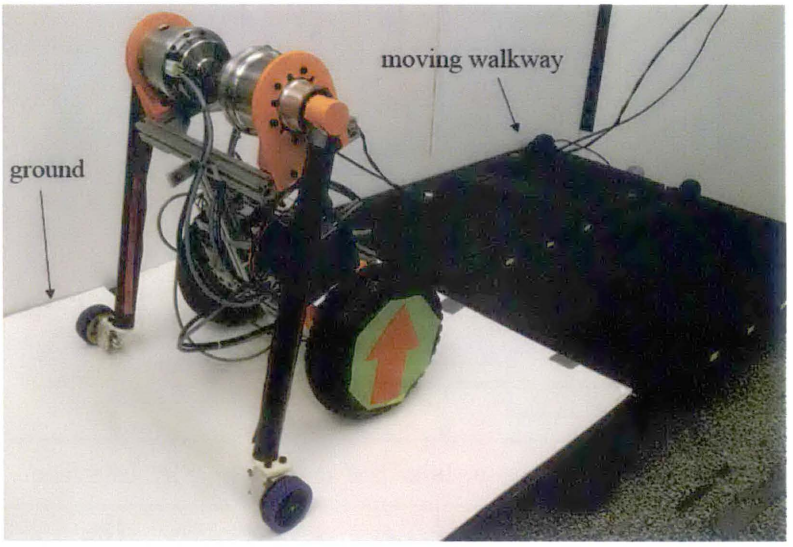
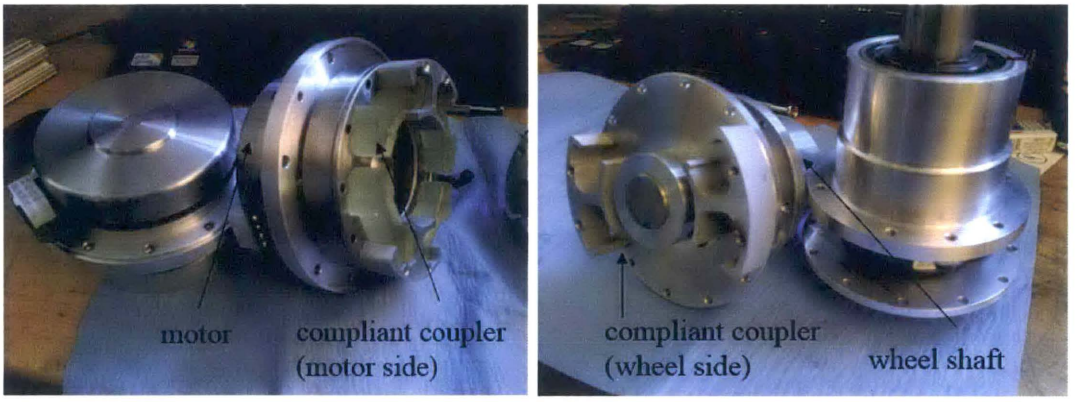


Figure 2-12: The prototype robot on the test track

rates the commands and transition impacts is $\omega_{cr} = 10Hz$. The coupler's stiffness is $k_s = 135.22N/rad$, which is obtained in the same way as the simulation.



(a) motor side (b) wheel side

Figure 2-13: Breakdown of a prototype Series Elastic Transmission

The control system of the prototype robot is implemented with NI LabVIEW and NI cRIO, whose structure is portrayed in Fig. 2-14.

Results from the experiment are plotted in Fig. 2-15.

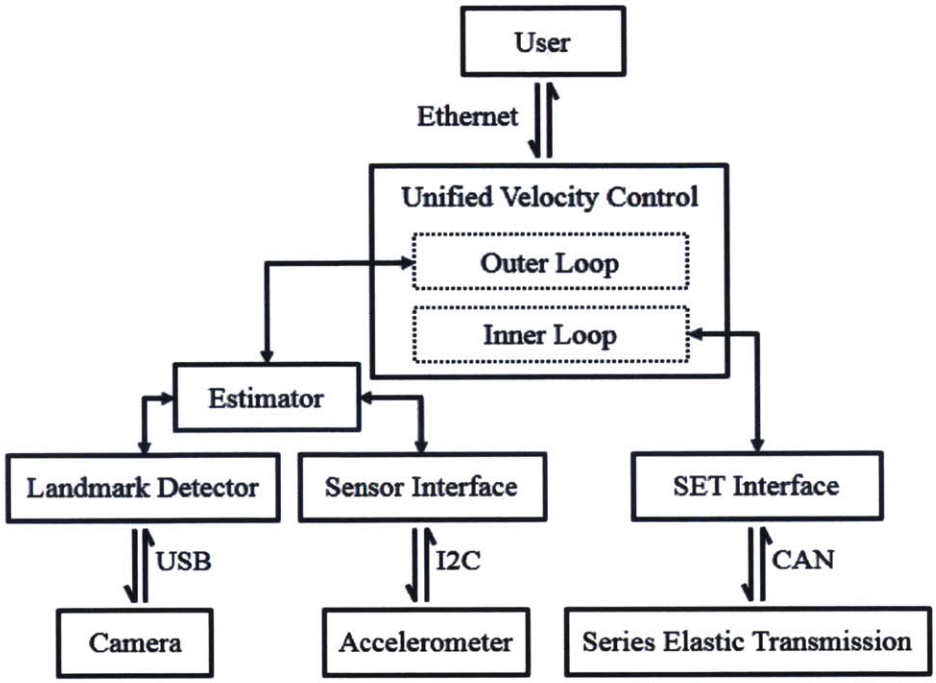
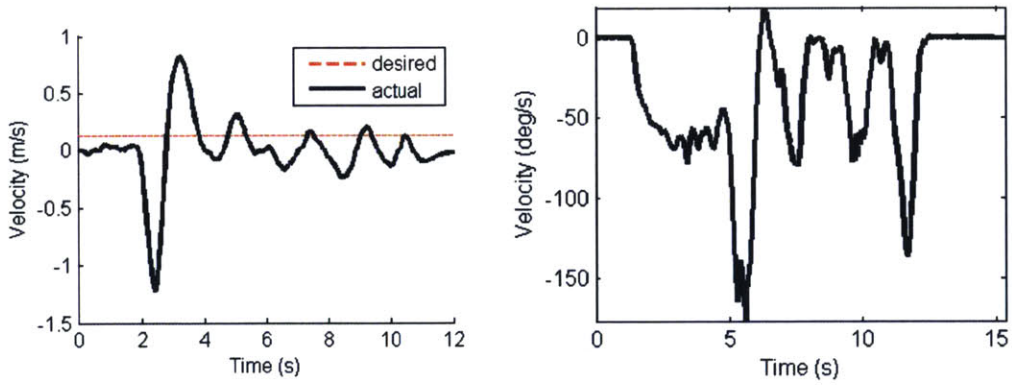


Figure 2-14: Control architecture of the prototype robot.



(a) robot ground speed

(b) wheel angular speed

Figure 2-15: Experiment result

It is observed that both wheels stops when the robot gets on the moving walk, and they starts to move again soon after the robot gets off, which is in accordance with the simulation results. However, the ground speed regulator needs more tuning.

Chapter 3

Navigation In Crowds

One of the engineering challenges for a public service robot is finding its way in crowded places. The robot should be able to reach its goal position quickly for the sake of efficiency. In the meantime, it must not cause much annoyance to nearby passengers, for that it is traveling together with them in the same public environment.

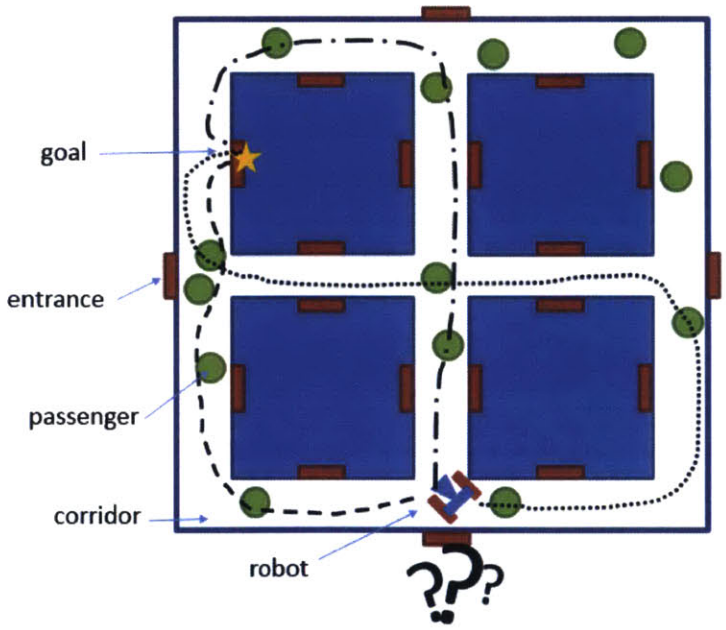


Figure 3-1: Navigation scenario example

The challenge arises from the fact that public environments can be very complicated, dynamic and uncertain. For example, a train station may be a multistory

complex with many static objects like unattended chairs appearing randomly; people are entering and leaving the station all the time; pedestrians can be standing or walking, whose states change cannot be well anticipated.

A typical mobile robot today only works in static environments where the sole source of changes is robot itself; or there are several agents whose behaviors are known to the robot. They cannot handle situations as complex as the train station depicted above, because they all rely on limited sensors onboard, which cannot provide enough information for the robot to fully comprehend the environment. Some robots have access to external sensors like surveillance cameras, and they could observe more objects and larger space. But that convenience comes with a constraint that their applications are limited to controlled environments such as laboratories or factories.

The preceding chapters introduce the synergy of a distributed sensor network and a central server. The central server can estimate the states of the environment in real-time with the help of the connected sensor network. Any robot deployed in the station is able to access these estimates, so that it can percept a world far more tremendous than before. Besides, high-speed Internet connection render it feasible for a robot to make full use of the massive computing power of the central server. With these advantages, the navigation problem can be tackled in a multi-scale and iterative fashion, which welcomes both global optimality and local optimality. The following sections explain a hierarchical online planning scheme for navigating a robot in a crowded environment.

3.1 Hierarchical Online Planning Scheme

A public service robot needs to find its way inside a crowded environment with least effort and minimal disturbances to other passengers. This task is not easy because

- The space in which the robot moves can be huge and complicated. For example, the train station of Tokyo is an expanding multi-story complex, with many platforms, hallways, intersections, shopping areas and underground connections;

- Crowds are highly dynamic. An unloading airplane can fill up the platform with passengers, making the condition of the platform change from "nobody around" to "people everywhere" within 1 minute;
- Individual's motion is hard to anticipate. A pedestrian's velocity is affected by numerous factors; one may suddenly stop walking for no explicit reason;
- Powerful computers are needed for path planning in high dimensional space due to the great number of people.

In the past few decades, numerous methods have been proposed for robot navigation. Some methods come with reference systems and algorithms to locate an object, and pilot a robot from an initial location to a desired goal position. This kind of method is called "Global Navigation", and the most commonly seen examples are navigation devices in automobiles, which locate vehicles using the Global Position System and compute best routes between start locations and destinations. Other methods try to determine a robot's position relative to nearby objects, and guide it to approach or avoid them. This so-called "Local Navigation" can be illustrated by educational robotic platforms, which uses infrared proximity sensors to avoid collision with walls.

However, no single method could deal with a dynamic environment comparable to a crowded train station. Global methods are hard to apply because there is no convenient system for indoor localization. The point-to-point plans usually fail to handle collision avoidance with passengers in the station. Local methods are not proper either, because they tend to be conservative, making the robot stop unnecessarily often for pedestrians. The plans they generate are short-sighted, giving no guarantee on the optimality of the overall trajectory.

The synergy of a distributed sensor network and a central server creates new opportunities for robot navigation in crowded places. First, all surveillance cameras, ticket gates and other sensors installed in the environment form a network that offers comprehensive observations of the station to the central server. Furthermore, every deployed robot can act as a mobile probe to sense specific areas. The enormous com-

puting power of the central server render it possible to process the rich information, and estimate the state of the all humans and robots in real-time [24]. These real-time estimates are delivered to the robots to help them navigate around. Moreover, by analyzing the recorded data we could build models of individual motion and crowd motion, which help forecast how the states would evolve in the future. Equipped with the real-time estimates and statistical models, a robot is aware of not only the passengers near it, but also conditions beyond its reach, spatially and temporally. These notable advantages give rise to a planning algorithm that embraces both global path optimality and local collision avoidance.

We propose a multi-scale online planning scheme for the robot navigation in order to fully facilitate the benefits explained above. Three planners form a hierarchical structure, every one of which produces a finite-horizon plan that solves the navigation problem at a different spatial and temporal scale. All planners re-plan after the first few actions in the corresponding plans are executed, so that a robot can adapt to the dynamic and uncertain environment. Table 3.1 lists the summary of the planning scheme. The plan from the Macro Planner is concise and abstract, which is passed

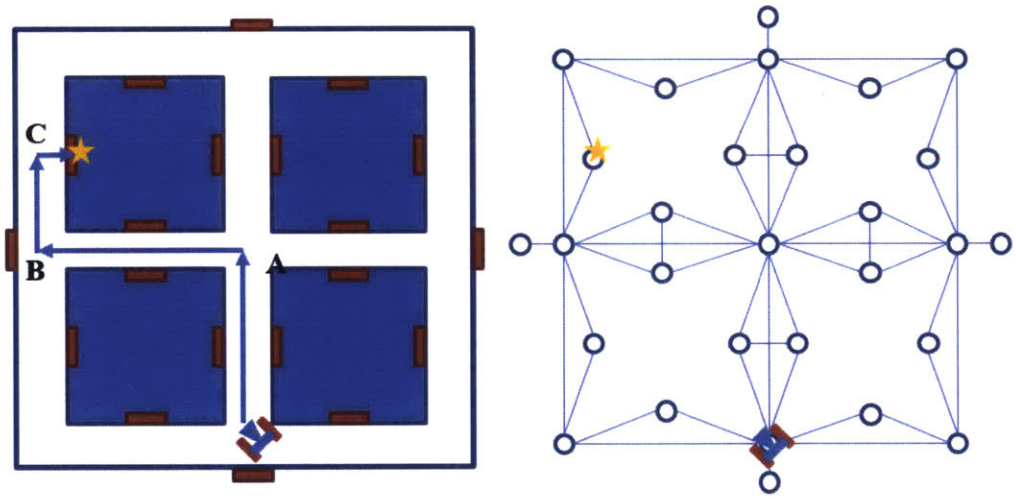
Table 3.1: Summary of multi-scale online planning architecture

Layer	Spatial Scale	Temporal Scale	Task
Macro	100m	10s	Find keypoints to connect start and goal positions.
Medium	10m	1s	Join or avoid passenger flows.
Micro	1m	0.1s	Avoid collision with individual human or robot.

down to the Medium Planner. The Medium Planner augments the abstract plan with more details and send it to the Micro Planner. Located in the lowest level, the Medium Planner grounds the plan with actual robot dynamics and environment states, and generates a continuous trajectory. Using an analogy of animation making, Macro Planner generates key frames; Medium Planner adds breakdowns to describe the transitions between key frames; Micro Planner creates in-betweens that fill the gaps and make the animation smooth.

3.1.1 Macro Planner

Being the highest-level planner, the Macro Planner solves the navigation problem from a large-scale persepective. As illustrated in Fig. 3-2, the Macro Planner models the station as a highly abstract graph. The nodes in the graph amount to *key locations* such as entrances, shops and intersections of pathways. An edge in the graph represents the pathway connecting the key locations corresponding to the two end nodes. The plan for the scenario depicted in Fig. 3-2a would be similar to: "Given the current and predicted states of the environment, the robot should go to intersection A, then to intersection B and place C, and the goal is on the right-hand side".



(a) actual scenario

(b) graph representation

Figure 3-2: Macro planner example

star: goal position; red shades: entrances; blue shades: walls

Thanks to the "large-scale" presumption, we are allowed to put aside vehicle dynamics, and treat the crowd in a pathway as a whole, whose effects are encoded by properties associated with that edge. With these expedient setup, the navigation problem is simplified to determining a sequence of nodes, or key locations in the actual environment, to visit one after another, so that the robot can reach its destination with minimal expected traveling time, i.e.,

$$\{n_0, n_1, ..., n_p\} = \underset{n_k}{\operatorname{argmin}} \sum_{k=0}^{p-1} \frac{l(< n_k, n_{k+1} >)}{S(< n_k, n_{k+1} >)}, \tag{3.1}$$

subject to

$$\begin{cases} n_0 &= initial\_position \\ n_p &= goal\_position \end{cases}, \tag{3.2}$$

where

- $e =< n_i, n_j >$ is the directed edge from node n_i to node n_j;
- $l(e)$ is the nominal length of the pathway that is represented by edge e;
- $S(e)$ is the expected speed for a robot moving in the pathway referred by e.

It is not difficult to realize that the denser a crowd, the slower one can move in it. And it is less struggling to follow a group of people than going against or cut through them. These observations help connect the robot's expected speed $S(e)$ to the edge's properties,

$$S(e) = S(d(e), v(e)), \tag{3.3}$$

where

- $d(e)$ is the average crowd density of the pathway corresponding to edge e;
- $v(e)$ is the average velocity (magnitude and direction) of the crowd in pathway referred by e.

With the plentiful and easily available data stored in the central server, we can identify the function $S(d(e), v(e))$ with data-driven methods. $S^+(e)$, the upper bound of $S(e)$, leads to an admissible heuristic function $l(e)/S^+(e)$, because the traveling time determined by the heuristic function is always smaller or equal to the actual traveling time. Furthermore, $l(e)/S^+(e)$ naturally follows the triangle inequality, making it a consistent heuristic function. *Dynamic A** is guaranteed to yield the optimal solution given the admissible and consistent heuristic function.

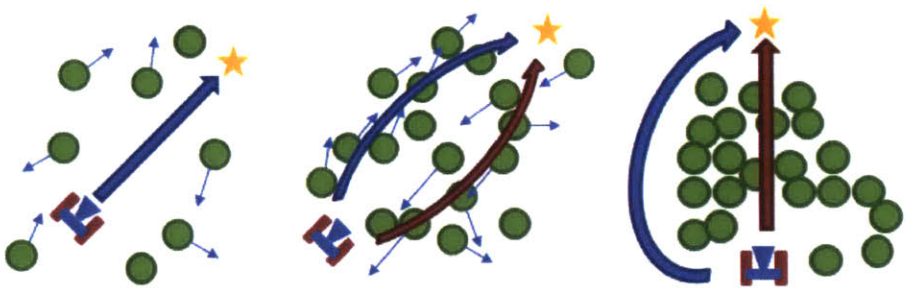
3.1.2 Medium Planner

The Medium Planner acts as a transition layer between the Macro Planner and the Micro Planner. Any two consecutive key locations in the plan from Macro Planner are interpreted as a sub-goal by the Medium Planner. The Medium Planner sets some waypoints between two consecutive key locations to describe the transition more meticulously, with considerations of mid-range optimality. These waypoints are passed down to the Micro Planner for further plan refinement. In this way global navigation is linked with local navigation more smoothly.

The mid-range optimality in short is defined as "following the trend". Passengers in busy train stations tend to form lines or flows. As explained in Section 3.1.1, it is more comfortable to move in accordance with pedestrian flows than to go against them. Fig. 3-3 exhibits some examples about mid-range optimality. In Fig. 3-3a, the space is not highly populated, and the best strategy for the robot is head directly towards the goal (blue arrow). In Fig. 3-3b, there are two distinguishable pedestrian flows, and the robot should join the one marked by the blue arrow instead of the adverse flow, which is marked by the red arrow. In Fig. 3-3a, lots of people are standing between the robot and its destination, and the robot is better to make a detour as indicated by the blue arrow than to follow the red arrow and cut through the crowd.

The Medium Planner should be able to identify a passenger group whose velocity is favorable for a robot to reach its destination, and lead the robot to join the group. When no such group exists, as in the case where all people are standing still, the Medium Planner guides the robot to avoid highly-populated regions.

We present an optimization-based solution for this task. Inspired by Smoothed Particle Hydrodynamics, the crowd is treated as an imaginary fluid flow, with humans corresponding to fluid particles. A velocity field and a density field are constructed from estimated passengers' states to describe the fluid. A new concept named Crowd Pressure is defined to measure the effort of traveling in crowds, which is analogous to the resistance an object receives as it moves in water. The classical Navigation



(a) (b) (c)

Figure 3-3: Medium planner example

star: goal position; green circles: passengers; short arrows: velocities of passengers

Function is augmented to Navigation Map, which is a vector field that encodes the floor plan, goal position and mid-range optimality. A cost function is established facilitating the above two concepts. The data and models in the central server help predict the evolution of the imaginary fluid flow in the near future. The plan for the robot is obtained solving an optimization on the expected cost over a finite-horizon. Details about Medium Planner will be covered in Section 3.2.

3.1.3 Micro Planner

The Micro Planner is located in the lowest level in the hierarchical architecture. It grounds the plan from the upper level planner with actual robot dynamics and nearby environment, and plans a continuous trajectory for the robot. The trajectory could be seen as a B-spline whose control points are the key locations and waypoints generated by the Macro Planner and Medium Planner.

At the finest scale, passengers are treated as individual moving obstacles whose position and velocity are made available by the estimator on the central server. The current estimates and predictions of these obstacles are projected to position-time space and any obstacle-free trajectory connecting the initial state and goal state is a guaranteed collision-free path for a robot. See Fig. 3-4.

Several algorithms exist for this kind of local collision avoidance, such as Virtual Force Field and Dynamic Window Approach. The latter one is preferred because it

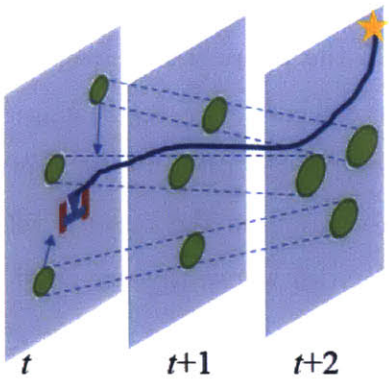


Figure 3-4: Micro planner example

is derived directly from the dynamics of robots. We will briefly explain the Dynamic Window Approach and its extension to dynamic obstacle avoidance here. Details about the methods could be found in [5, 20, 17].

The Dynamic Window Approach assumes zero accelerations in the a finite-horizon after the first interval, which simplifies the problem. The planning is conducted in the velocity space of the robot, i.e. the control variables are the translational velocity and the rotational velocity. The robot has limited actuation, so that in the velocity space only a small region near the robot's current velocity is achievable in the next interval. The blue rectangle in Fig. 3-5 represents the feasible velocities.

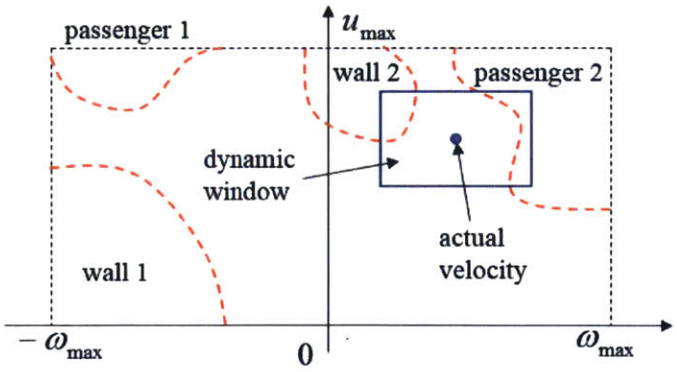
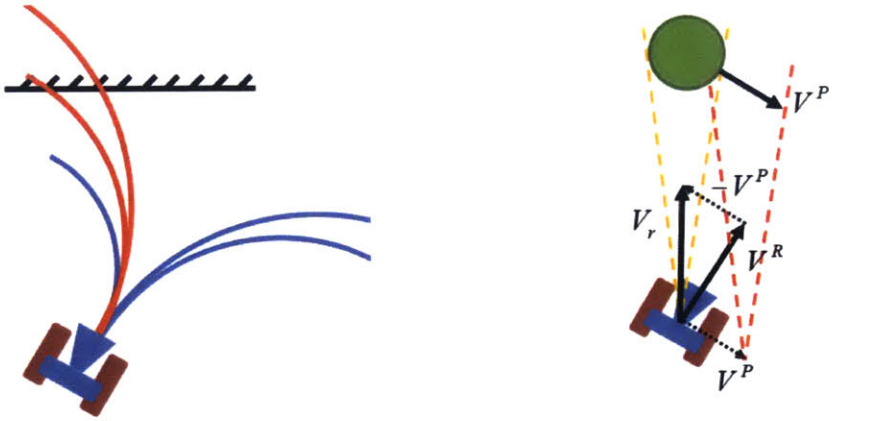


Figure 3-5: An example velocity space in Dynamic Window Approach

Given the states of nearby obstacles, any velocities that lead to collisions in the finite-horizon are ruled out, as marked by the red dashed curves in Fig. 3-5. The collision-checking method with static obstacles is illustrated in Fig. 3-6a. The robot's

trajectory consists of lines and arcs. If a trajectory intersects with a static obstacle, for example the red curves in the figure, the corresponding velocity is regarded as dangerous and would not be considered in the planning. For a dynamic obstacle, assuming its states are delivered to the robot by the central server, we can construct a "Velocity Obstacle" [17] according to the relative velocity between the obstacle and the robot. Only the velocities whose tips are outside the velocity obstacle are considered safe.



(a) static obstacles

(b) dynamic obstacles

Figure 3-6: Collision-checking for static and dynamic obstacles

An objective function is defined over the velocity space to reflect collision avoidance, minimal travel distance and goal attraction. We remove the velocities that may cause a collision from the velocity space, and search the feasible region for the velocity that maximizes the objective function. As long as the robot executes the action, the planning is performed again.

3.2 Medium Planner Design

The Medium Planner generates waypoints to connect consecutive key locations by solving a finite-horizon optimization. It is assumed that the Macro Planner has given the destination key location and task speed to the Medium Planner, and the real-time estimates of every passenger is accessible to the robot.

3.2.1 Optimization Formulation

The optimal waypoint sequence $\{\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \cdots, \hat{\boldsymbol{x}}_l\}$ minimizes the following estimated cost that is summed over a finite horizon:

$$\{\hat{\boldsymbol{x}}_1, \cdots, \hat{\boldsymbol{x}}_l\} = \underset{\hat{\boldsymbol{x}}_k}{\operatorname{argmin}} \Big\{ \sum_{k=1}^{l} \gamma^k \{P(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, k) + R(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, k)\} + H(\hat{\boldsymbol{x}}_l) \Big\}, \quad (3.4)$$

subject to

$$\begin{cases} \hat{\boldsymbol{x}}_{k+1} &= \boldsymbol{A}(\hat{\boldsymbol{x}}_k, \boldsymbol{u}_k) \\ \hat{\boldsymbol{x}}_0 &= \boldsymbol{x}_0 \\ \boldsymbol{\Phi}(\hat{\boldsymbol{x}}_k) &= 0 \\ \boldsymbol{\Psi}(\boldsymbol{u}_k) &= 0 \end{cases}, \quad (3.5)$$

where

- l is the length of horizon, similar to that in Model Predictive Control;
- γ is the discount factor that reflects the uncertainty of the future;
- $\hat{\boldsymbol{x}}_k = [x_k, y_k, \theta_k]^T$ is the waypoint, a.k.a. robot's position and heading, for future time step k;
- $\boldsymbol{x}_0$ is the robot's actual state at the beginning of the planning;
- $\boldsymbol{u}_k = [\omega_k, u_k]$ is the robot action for future step k, consisting of the angular velocity and linear velocity;
- $P(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, k)$ is the Crowd Pressure, i.e. the cost of robot with state $\boldsymbol{x}_{k-1}$ moving in the crowd with velocity $\boldsymbol{u}_k$;
- $R(\boldsymbol{x}_{k-1}, \boldsymbol{u}_k, k)$ is the penalty from the velocity regulator. The velocity reference is read from the Navigation Map;
- $H(\hat{\boldsymbol{x}}_l)$ is the estimated cost to reach goal from state $\hat{\boldsymbol{x}}_l$, as discussed in [4];
- $\boldsymbol{A}(\boldsymbol{x}_k, \boldsymbol{u}_k)$ describes the state transition under action $\boldsymbol{u}_k$;
- $\boldsymbol{\Phi}(\boldsymbol{x}_k)$ models the constraints on the robot's state from walls etc.;

- $\boldsymbol{\Psi}(\boldsymbol{u}_k)$ represents the action limits of the robot.

Both Crowd Pressure and Navigation Map are computed using estimated passengers' states. Current crowd state, i.e. every passenger's position and velocity, is readily available from the central server. Future crowd states are forecast with the statistical models that are identified from previous data offline.

Let ΔT denote the planning interval. One can easily calculate $\hat{\boldsymbol{x}}_k$ from $\boldsymbol{u}_k$ for a linear transition model:

$$\begin{aligned} \hat{\boldsymbol{x}}_k &= \begin{pmatrix} x_k \\ y_k \\ \theta_k \end{pmatrix} = \boldsymbol{A}(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k) \\ &= \begin{pmatrix} x_{k-1} + u_k \Delta T \cos\theta_k \\ y_{k-1} + u_k \Delta T \sin\theta_k \\ \theta_{k-1} + \omega_k \Delta T \end{pmatrix}. \end{aligned} \tag{3.6}$$

Therefore it makes no difference whether to write the plan as $\{\hat{\boldsymbol{x}}_1, \hat{\boldsymbol{x}}_2, \cdots, \hat{\boldsymbol{x}}_l\}$ or $\{\boldsymbol{u}_1, \boldsymbol{u}_2, \cdots, \boldsymbol{u}_l\}$. For the ease of cost function formulation, we will use the latter representation in the rest of the section.

3.2.2 Crowd Pressure

When an object moves in a steady flow, it feels drag force because of the fluid pressure acting on its surface. The pressure is proportional to the fluid density and quadratic to the relative velocity between the object and the fluid around it. As a result of the drag force, the object's velocity would eventually converge to the same velocity as the nearby fluid, when the relative velocity becomes zero and so does the pressure.

If we regard humans as fluid particles and the crowd they form as water, then a similar conclusion can be drawn for a robot moving in that crowd: If the robot has a velocity similar to the surrounding people, then it needs small effort to move; Conversely, if the relative velocity between the robot and its surrounding people is large, as in the case of going against a pedestrian flow, then the robot would have a difficult

time reaching its destination and it is causing much annoyance to nearby passengers. The crowd-fluid analogy has been adopted by researchers in transportation [8] and computer graphics [22].

Crowd Pressure P is defined in a similar way to the fluid pressure,

$$P = \rho \|V_r\|^2, \tag{3.7}$$

where ρ is the local crowd density and V_r is the relative velocity (vector) between the robot and the surrounding pedestrian flow. It is a measure of the effort to move inside crowds, because low pressure indicates either low crowd density or low relative velocity, both of which lead to small traveling resistance. In other words, if a robot stays in low-pressure regions, it is either moving in free space or following a group of passengers whose velocity is favorable to it.

To calculate the Crowd Pressure that a robot traversing in a public environment receives, we first construct a velocity field and a density field from estimated/predicted crowd states. The area near the robot is meshed uniformly. The velocity for any passenger inside the area is distributed among the four grid points which make up of the cell he/she occupies. The distribution coefficients are

$$\begin{cases} W_1 &= (1-a)(1-b) \\ W_2 &= (1-a)b \\ W_3 &= a(1-b) \\ W_4 &= ab \\ \sum W_k &= 1 \end{cases}, \tag{3.8}$$

where

$$\begin{aligned} a &= (x - x_{bl})/(x_{br} - x_{bl}), \\ b &= (y - y_{bl})(y_{tl} - y_{bl}). \end{aligned} \tag{3.9}$$

are the the normalized coordinate of the passenger's location in the cell. (x_{bl}, y_{bl}), (x_{br}, y_{br}) and (x_{tl}, y_{tl}) are the coordinates of the bottom-left grid, bottom-right grid and top-left grid in the global frame, respectively. See Fig. 3-7 for illustration of

velocity decomposition. The density field is obtained using the same decomposition

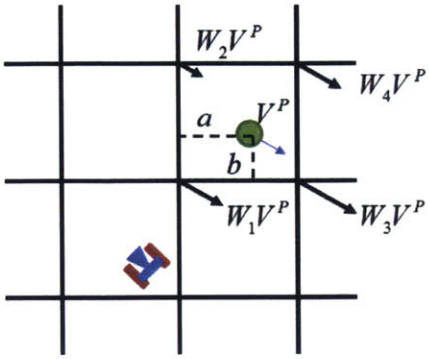


Figure 3-7: Decomposition scheme for constructing density field and velocity field

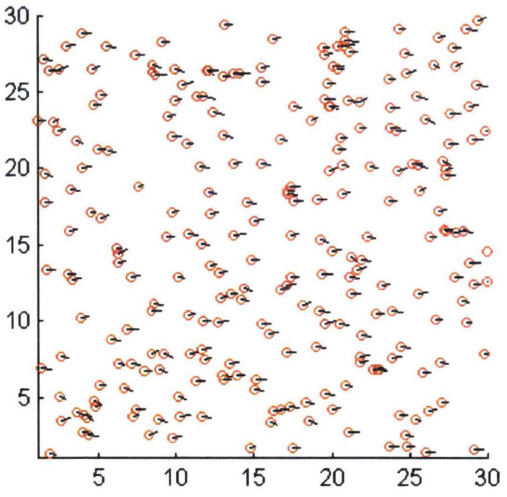
V^P: a passenger's velocity; W_i: distribution coefficients

scheme, with free space carrying density 1 and every passenger carrying density ρ_0 ($\rho_0 > 1$). Fig. 3-8a illustrates an example crowd, whose average velocity is $0.5m/s$ to the right. Every small red circle marks a passenger's position and the attached short line indicates the velocity. The velocity field and the density field after decomposition are presented in Fig. 3-8b and Fig. 3-8c, respectively.

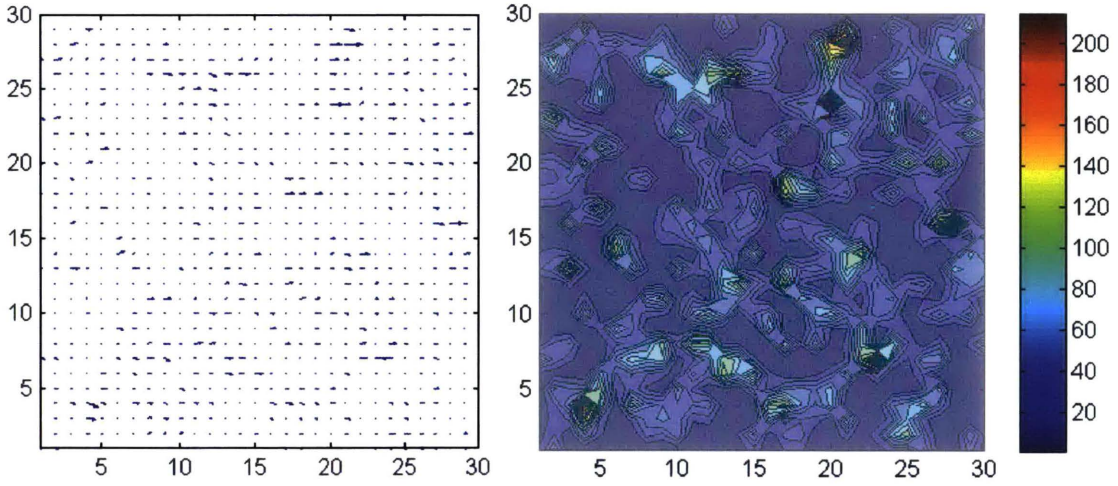
Then, we smooth the velocity field and the density field using a kernel function such as moving average. The velocity field and the density field obtained through the decomposition can be very thorny. The field values jump at interfaces between cells. And small offset in a passenger's estimated location can result in large difference in the fields. Therefore we smooth the raw fields to alleviate the discontinuities and errors caused by estimation inaccuracy. Additionally, smoothing gives some hints about future crowd motion. For example, two people approaching each other have to stop before they collide, and in the smoothed velocity field some place between them has zero velocity, which captures this tendency. Fig. 3-9 shows the smoothed velocity field and smoothed density field for the same example crowd.

Next, we calculate the relative velocity field by subtracting the robot's velocity from the smoothed velocity field. Fig. 3-10 plots the relative velocity field for a robot moving to the right with $0.3m/s$.

Finally, we compute the Crowd Pressure as $P = \rho V_r^2$, where ρ is the local crowd

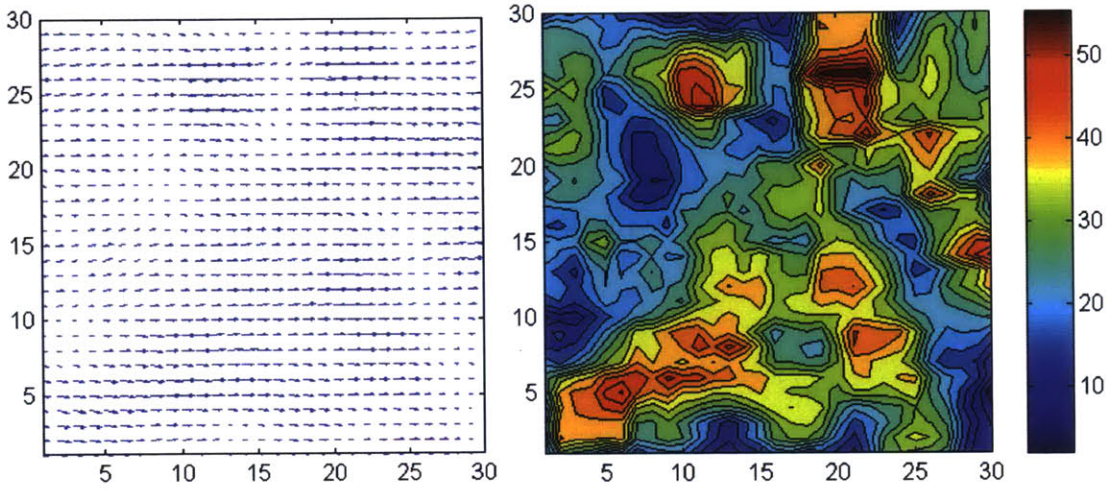


(a) actual scenario



(b) velocity field (c) density field

Figure 3-8: Velocity field and density field for an example crowd



(a) velocity field (b) density field

Figure 3-9: Smoothed velocity field and density field for an example crowd

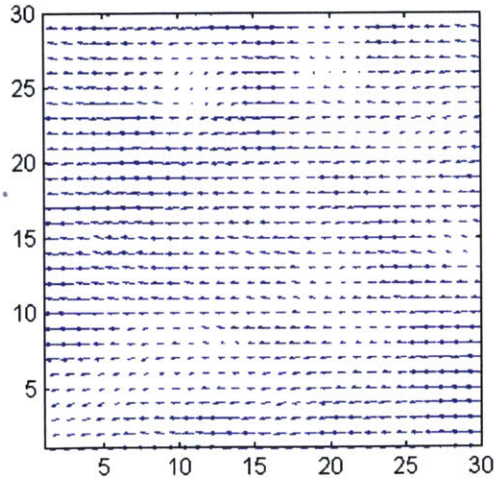


Figure 3-10: Relative velocity field for a robot moving in the example scenario

density and V_r is the relative velocity between the robot and nearby people. According to Fig. 3-11, we can find out that the low-pressure areas correspond to either the low-density areas in Fig. 3-9b (e.g. bottom right corner, around $(30, 5)$) or the small-relative-velocity areas in Fig. 3-10 (e.g. near top boundary, around $(12, 25)$).

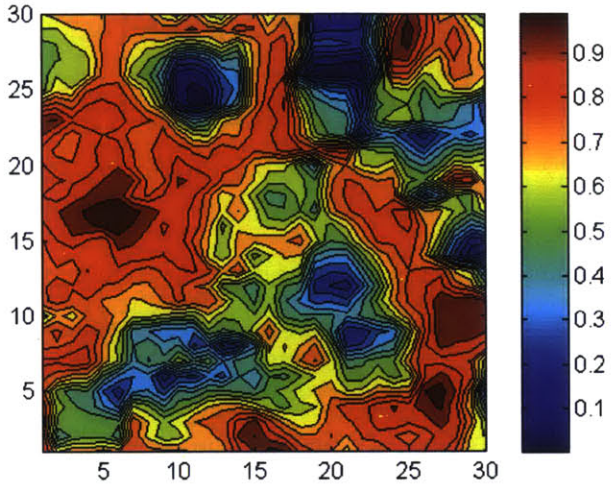


Figure 3-11: Crowd pressure for a robot moving in the example scenario

3.2.3 Navigation Map

In the literature of robot navigation, navigation functions are introduced as a tool to create feasible paths that allows a robot to move from its initial position to its destination while avoiding obstacles [14, 13]. It can be a potential function whose gradients represent goal-attraction and obstacle-repulsion. Sometimes it is constructed with optimal control, with the control variable being a robot's velocity and the cost function being the distance-to-goal. In both cases, a robot can obtain its reference velocity by plugging its position into the navigation function. Fig. 3-12 gives a visual explanation of navigation function, where the mahogany arrows indicate the reference velocities.

There are some issues with navigation functions, however. The potential method may suffer from local minima and instability in narrow corridors. A typical navigation function is static, in that it assumes an exactly known environment where all obstacles are static. To deal with a dynamic environment, one has to compute the navigation function frequently with updated obstacles' positions. In a train station

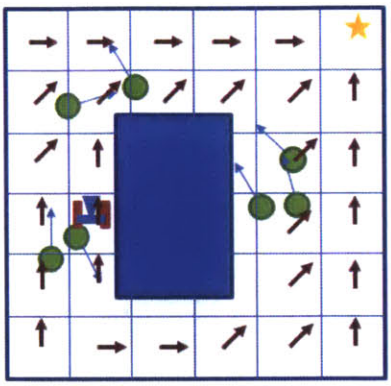


Figure 3-12: Navigation function for an example scenario

where hundreds of people are walking or standing, a robot may not be able to find a navigation function that tells a path to goal due to the occlusions.

To better facilitate the hierarchical planning architecture and the real-time estimates offered by the central server, we present the concept of Navigation Map as a replacement for the navigation function. A Navigation Map is an instantaneous vector field that describes the optimal action for any give position, so that if a robot follows the velocity suggested by the Navigation Map for every instant, it will reach its goal position with a minimum of the objective function (3.4), while satisfying the constraints posed by walls, corridors, etc. With the Navigation Map indicating the robot which way to preceed, i.e. the nominal trajectory , the navigation problem becomes "follow the nominal trajectory and make necessary changes if there are unexpected people". See Fig. 3-16 for a visual explanation of Navigation Map. Comparing the arrows between Fig. 3-12 and Fig. 3-16, we can see the reference velocities in the Navigation Map is influenced by the existence of pedestrians.

The navigation functions treat "collision avoidance" as a hard constraint that should never be violated. However, since Navigation Maps model crowds as fluids, the hard constraint of "collision avoidance" is replaced with a soft constraint of "minimize Crowd Pressure" This change is acceptable because it is the Micro Planner's responsibility to avoid collision with individual passengers. If Medium Planner successfully creates keypoints that make a robot stay out of crowd or follow pedestrian flows, the chances of collision can be reduced dramatically.

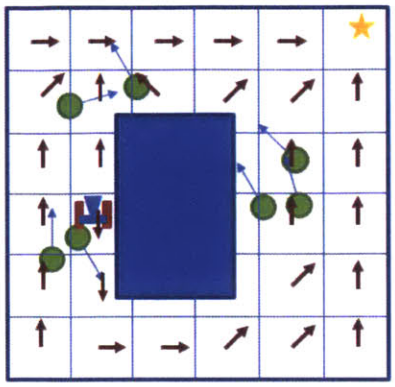


Figure 3-13: Navigation Map for an example scenario

Navigation Maps are spatial and temporal. One may find two neighboring locations in a Navigation Map have suggested velocities pointing to each other. This should not be interpreted as a local minimum, because a Navigation Map is useful only in the planning interval it is compute for. When a robot decides to follow the suggested velocity at current position and arrives at a location that seems to push it back according to the same Navigation Map, it should in fact consult the successive Navigation Map for the next planning interval, whose suggested velocity may no longer push it backwards.

However, true Navigation Maps cannot be obtained. As defined, a Navigation Map entails the control policy $\boldsymbol{u}_h(x, y, t)$ and cost-to-go $h(x, y, t)$: for every robot position (x, y) there is such an action $\boldsymbol{u}_h(x, y, t)$ that if the robot applies that action, the expected cost to reach its destination $h(x, y, t)$ would be minimal. Nonetheless it is impossible to construct the Navigation Map without solving (3.4) first. This is like the "chicken-or-egg" dilemma: on one hand we want to compute the Navigation Map in order to solve the optimization; on the other hand we have to solve the optimization to construct the Navigation Map.

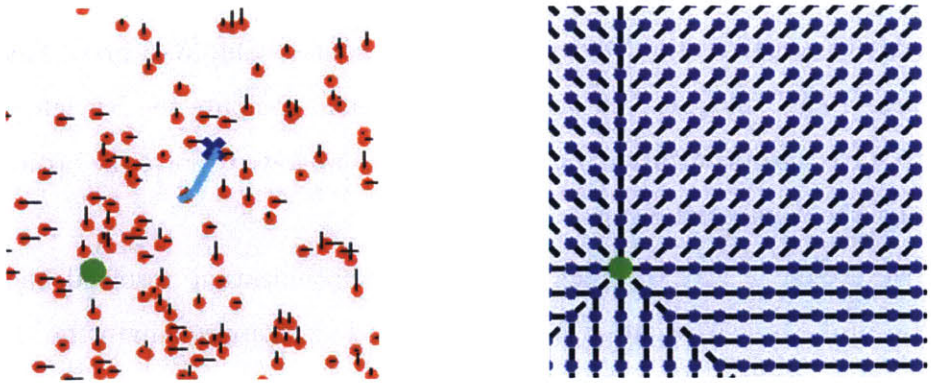
Here we present a way to break the tie. The optimization is solved backwards first, and a grid-based Navigation Map is constructed to approximate the true one. Then the grid-based Navigation Map is interpolated to solve the optimization forward, whose results are the final output of the Medium Planner.

The first step is to initialize the terminal-time Navigation Map. Since Navigation

Maps are spatial and temporal and we don't know exactly when the robot would reach its goal, we cannot simply apply the approach for calculating navigation functions. Instead, we set the terminal-time to be $l+1$, where l is the length of planning horizon, and calculate the Navigation Map at $l+1$ as

$$\begin{cases} \boldsymbol{u}_h(x,y,l+1) &= \text{argmin}\{distance\_to\_goal\} \\ h(x,y,l+1) &= \min\{distance\_to\_goal\} \end{cases}, \tag{3.10}$$

where the magnitude of $\boldsymbol{u}_h$ is the task speed S assigned by Macro Planner, and the direction points to the eight neighboring grid points. The underlying reasoning is that the cost-to-goal consists of two proportions: one is due to distance traveled and the other is associated with the existence of crowds. Because of the discount factor γ, the effect of crowd in the far future is discounted so much (e.g. $0.7^{10} \approx 0.03$) that we can ignore the cost associated with it. This makes $h(x,y,l+1)$ the lower bound of $H(\hat{\boldsymbol{x}}_l)$ in (3.4). Fig. 3-14b shows the Navigation Map initialization with $l=5$. The red circles mark the estimated passenger positions, and black line attached to a circle implies the estimated velocity for that passenger. The blue cross and the green circle represent the initial and goal positions for the robot, respectively. In the Navigation Map, the blue dots depict the grid points, and the black lines attached to them indicate the suggested velocity.



(a) predicted scenario ($t=6$) (b) Navigation Map ($t=6$)

Figure 3-14: Navigation Map initialization example

For any planning time $0 \leq t \leq l$, the Navigation Map is obtained by

$$\begin{cases} \boldsymbol{u}(x,y,t) &= \underset{(x_n,y_n)\in D(x,y)}{\operatorname{argmin}} \{h(x_n,y_n,t+1) + C(x,y,\boldsymbol{u},t)\} \\ h(x,y,t) &= \underset{(x_n,y_n)\in D(x,y)}{\min} \{h(x_n,y_n,t+1) + C(x,y,\boldsymbol{u},t)\} \end{cases}, \tag{3.11}$$

where

- $D(x,y)$ is the set of all neighboring grid point of (x,y)
- $\boldsymbol{u}$ has direction from (x,y) to (x_n,y_n), and its magnitude is the robot's task speed;
- $C(x,y,\boldsymbol{u},t)$ is the discounted cost to apply action $\boldsymbol{u}$ at (x,y) at future step t.

As stated before, we calculate the grid-based Navigation Map by solving the optimization backwards from terminal-time. Therefore the cost C has the same structure as (3.4), with the only difference being the value of parameters,

$$C(x,y,\boldsymbol{u},t) = \gamma^t\{P(\boldsymbol{x},\boldsymbol{u},t) + R(\boldsymbol{x},\boldsymbol{u},t)\}, \tag{3.12}$$

where

- $\boldsymbol{x} = (x,y)$ is the position in consideration;
- $P(\boldsymbol{x},\boldsymbol{u},t)$ is the predicted Crowd Pressure for a robot to apply $\boldsymbol{u}$ from position $\boldsymbol{x}$ at future step t;
- $R(\boldsymbol{x},\boldsymbol{u},t)$ is the velocity regulator whose reference is given by the successive Navigation Map $\boldsymbol{u}_h(x,y,t+1)$.

See Fig. 3-15 for the rule of value iteration.

Fig. 3-16 shows a few steps in the calculation of Navigation Map after the initialization in Fig. 3-14b.

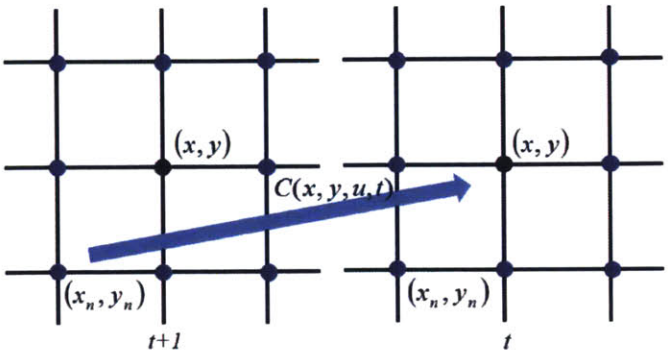
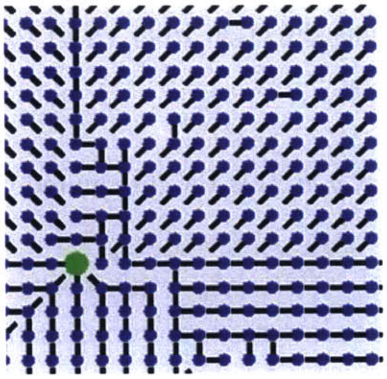


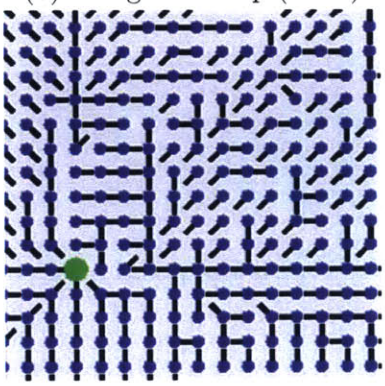
Figure 3-15: Value iteration rule for Navigation Map

(a) predicted scenario ($t = 5$)



(b) Navigation Map ($t = 5$)

(c) predicted scenario ($t = 4$)



(d) Navigation Map ($t = 4$)

Figure 3-16: Navigation Map calculation example

3.2.4 Solving Optimization

The optimization is solved using Branch-and-Bound search [9]. Starting with the robot's initial state $\boldsymbol{x}_0$, we choose an allowable angular velocity ω_1 and an allowable

linear velocity u_1, and obtain a robot state $\hat{\boldsymbol{x}}_1$ using (3.6). This procedure is repeated for every planning step, which results in a search tree demonstrated in Fig. 3-17.

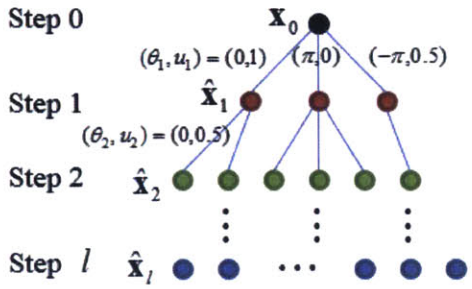


Figure 3-17: Brand-and-Bound search tree

However, enumerating both angular velocity and linear velocity makes the branching factor large, which slows down the planning. We observed that for a given robot state $[x, y, \theta]^T$, once the angular velocity is chosen, the linear velocity is no longer an independent variable. In the rest of this section we will develop a control law for determining the linear velocity.

Recall from Section 3.2.2 that robot with state $\hat{\boldsymbol{x}}_{k-1}$ would receive resistance from the crowd, if it executes action $\boldsymbol{u}_k$

$$\begin{aligned} P(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, k) &= local\_crowd\_density \times robot\_relative\_velocity^2 \\ &= \rho(x_{k-1}, y_{k-1}) \left\| \begin{matrix} V_x(x_{k-1}, y_{k-1}) - u_k \cos\theta_k \\ V_y(x_{k-1}, y_{k-1}) - u_k \sin\theta_k \end{matrix} \right\|^2 \\ &= \rho\{u_k^2 - 2(V_x \cos\theta_k + V_y \sin\theta_k)u_k + (V_x^2 + V_y^2)\} \\ &= \rho(u_k^2 - 2C_1 u_k + C_2) \end{aligned} \tag{3.13}$$

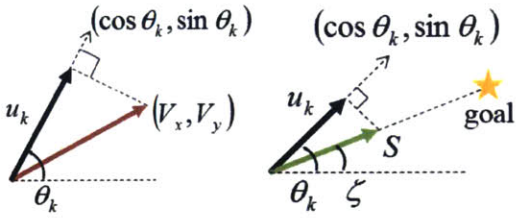
where ρ is the local crowd density and (V_x, V_y) is the velocity of surrounding passengers. See Fig. 3-18a. For a specified robot position (x_{k-1}, y_{k-1}), the coefficient $C_1 = V_x \cos\theta_k + V_y \sin\theta_k$ only depends on θ_k, the robot's heading at future step k. If θ_k is given, then both C_1 and C_2 are constants, and P becomes quadratic in u_k, the robot's linear velocity.

In (3.4) the velocity regulator pushes the robot to adhere to a prescribed velocity suggested by Navigation Map. Any deviation from the prescribed velocity will be

penalized by

$$\begin{aligned} R(\hat{\boldsymbol{x}}_{k-1}, \boldsymbol{u}_k, k) &= regulator\_gain \times robot\_velocity\_error^2 \\ &= \alpha \left\| \begin{matrix} S\cos\zeta(x_{k-1}, y_{k-1}) - u_k \cos\theta_k \\ S\sin\zeta(x_{k-1}, y_{k-1}) - u_k \sin\theta_k \end{matrix} \right\|^2 \\ &= \alpha\{u_k^2 - 2S\cos(\zeta - \theta_k)u_k + S^2)\} \\ &= \alpha(u_k^2 - 2D_1 u_k + D_2), \end{aligned} \tag{3.14}$$

where α is the regulator gain; S is the magnitude of the prescribed velocity and ζ is the direction of the prescribed velocity, both of which are from the Navigation Map $\boldsymbol{u}_h(x, y, k)$. See Fig. 3-18b For a specified robot position (x_{k-1}, y_{k-1}), the coefficient $D_1 = \cos(\zeta - \theta_k)$ only depends on θ_k, the robot's heading at future step k. If θ_k is given, then both D_1 and D_2 are constants, and R becomes quadratic in u_k, the robot's linear velocity.



(a) Crowd Pressure (b) Navigation Map

Figure 3-18: Nomenclature for solving optimization

The one-step cost F is the sum of the above two terms,

$$F = P + R = (\rho + \alpha)u_k^2 - 2(\rho C_1 + \alpha D_1)u_k + (\rho C_2 + \alpha D_2). \tag{3.15}$$

Note that for a given robot position (x_{k-1}, y_{k-1}), both C_1, and D_1 only depend on θ_k, the robot's heading at future step k. If θ_k is given, then all of C_1, C_2, D_1 and D_2 are constants, and F becomes quadratic in u_k, the linear velocity of the robot. The value that minimizes the one-step cost is $u_c = (\rho C_1 + \alpha D_1)/(\rho + \alpha)$. For a real robot

with speed limit u_{max}, we choose the linear velocity according

$$u_k = \begin{cases} 0 & \text{if } u_c < 0 \\ u_c & \text{if } 0 \leq u_c \leq u_{max} \\ u_{max} & \text{if } u_c > u_{max} \end{cases}, \tag{3.16}$$

which is illustrated in Fig. 3-19.

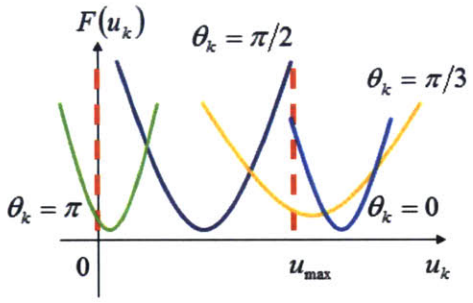


Figure 3-19: One-step cost as a function of robot action.

3.3 Medium Planner Simulation

This section gives several simulation results for the Medium Planner. In all simulations, the world is a $30 \times 30m^2$ square. The passengers are modeled as random-walking particles, whose future velocities are assumed to be equal to the current ones and the positions are linearly extrapolated. The parameters used by the Medium Planner are

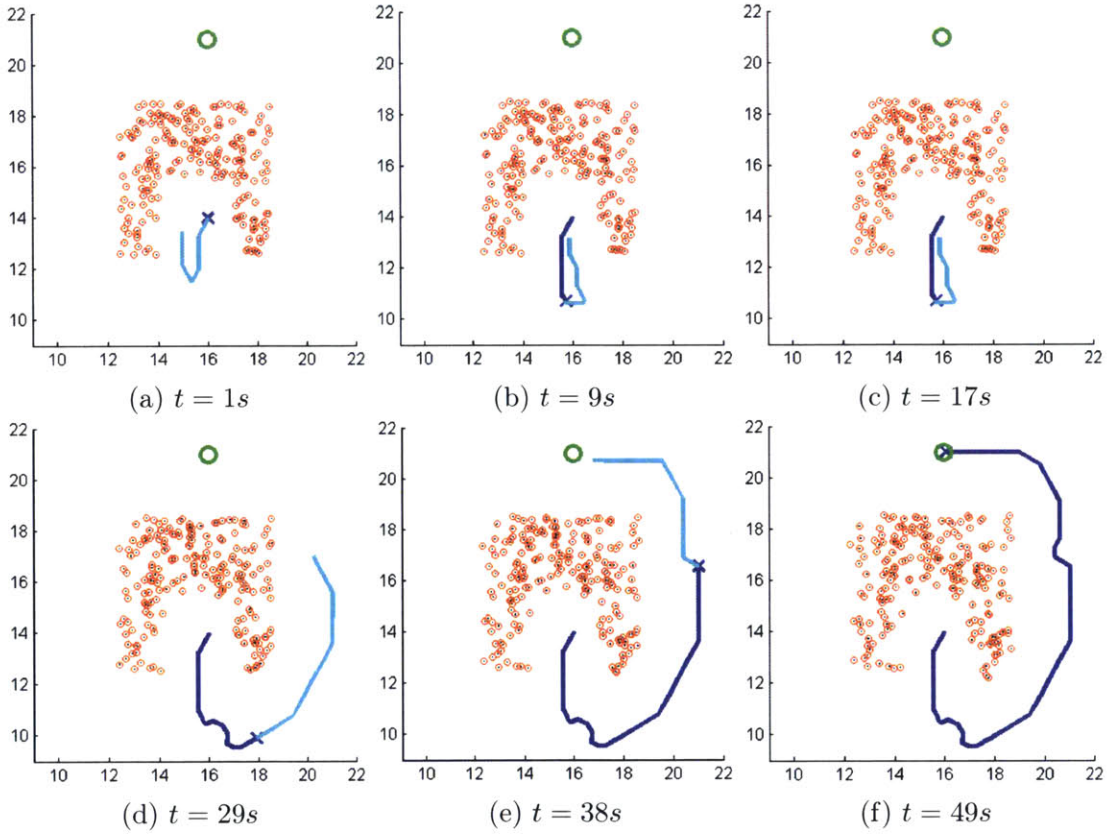
- horizon length $l = 10$ seconds;
- planning interval $\Delta T = 1$ second;
- discount factor $\gamma = 0.75$;
- unit density carried by a passenger $\rho_0 = 100$;
- velocity regulator gain $\alpha = 100$;
- velocity regulator gain for calculating the Navigation Map $\alpha^{NM} = 50$;

- $cell\_size = 1 \times 1$;
- $smoothing\_kernel\_size = 3 \times 3$;
- max linear velocity $u_{max} = 1m/s$;
- max angular velocity $\omega_{max} = \pi/3deg/s$;
- task speed $S = 1m/s$;

The plans from the Medium Planner are overlaid on top of the simulated scenario. In the following plots, the red circles represent the positions of passengers, and the short black lines attached to the circles indicate the velocities. The blue cross and the green circle mark the current position and the goal position of the robot, respectively. The blue lines connect the past waypoints, while the cyan lines connect the future waypoints planned by the Medium Planner.

In the first example, passengers are barely moving and they form a box canyon. The robot initially stands inside the box canyon and its destination is on the other side of the crowd. Because the passengers' velocities are trivial, the quasi-static crowd is very similar to walls and other permanent obstructions. The robot realizes that it should not walk through the crowd and steers away as suggested by the Navigation Map, which is illustrated by the cyan lines in Fig. 3-20. The robot hesitates between $t = 9s$ and $t = 24s$, because Navigation Map directs it to go back to the crowd while the estimated high Crowd Pressure prevents it from doing so. The obvious wrong direction of the Navigation Map is probably caused by the coarse mesh with which the Navigation Map is obtained. At $t = 25s$, the robot finds an alternative route to bypass the crowd. It then marches to its destination with desired task speed. Despite the hesitation and wait, the robot is successful in escaping the box canyon, which shows both local optimality (avoiding people) and global optimality (choosing low-cost path to goal).

The second simulation demonstrates four passenger flows. The horizontal flow in the top area is from left to right, while the flow in the bottom area is from right to left. The vertical flow in the left area is from bottom to top, while the flow in the

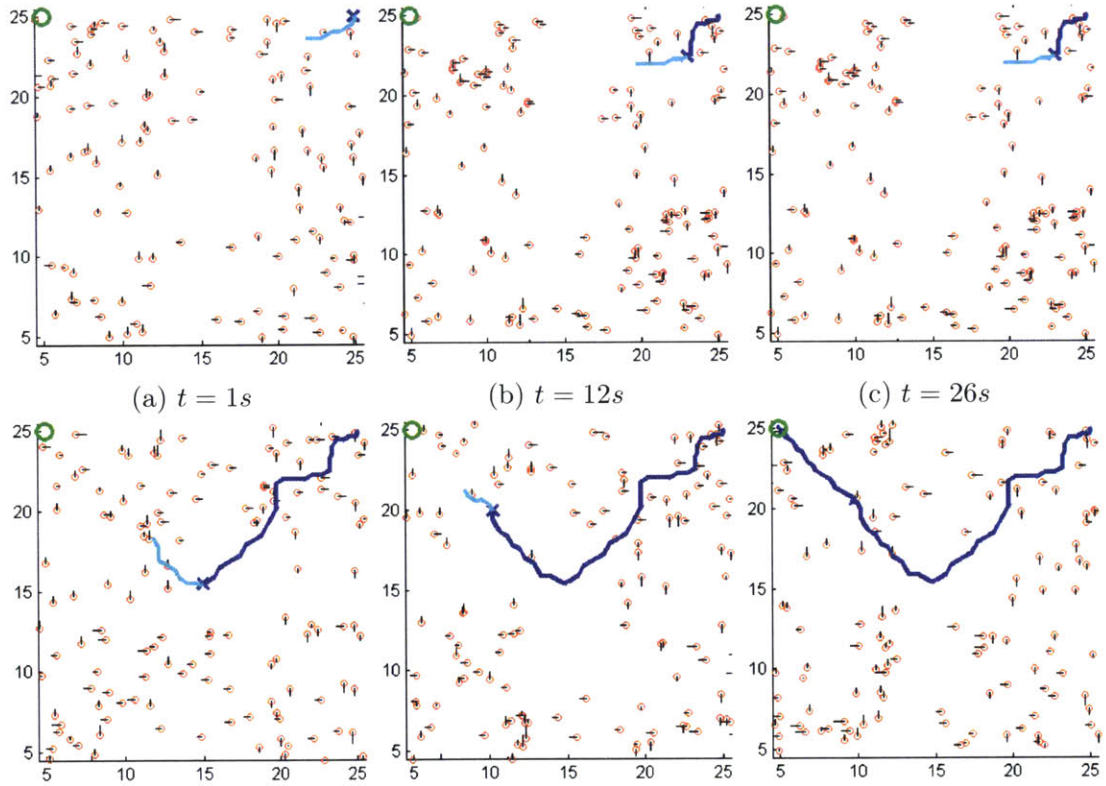


(a) $t = 1s$ (b) $t = 9s$ (c) $t = 17s$

(d) $t = 29s$ (e) $t = 38s$ (f) $t = 49s$

Figure 3-20: Medium Planner simulation result 1

right area is from top to bottom. The robot starts from the top right corner and its destination is in the top left corner. Instead of walking through the adverse flow in the top area, the robot decides to make a detour and take advantage of the other flows. It joins the vertical flow and goes downwards, and finds the less-populated space in the center of the environment, from where it heads to goal with the favorable vertical flow.

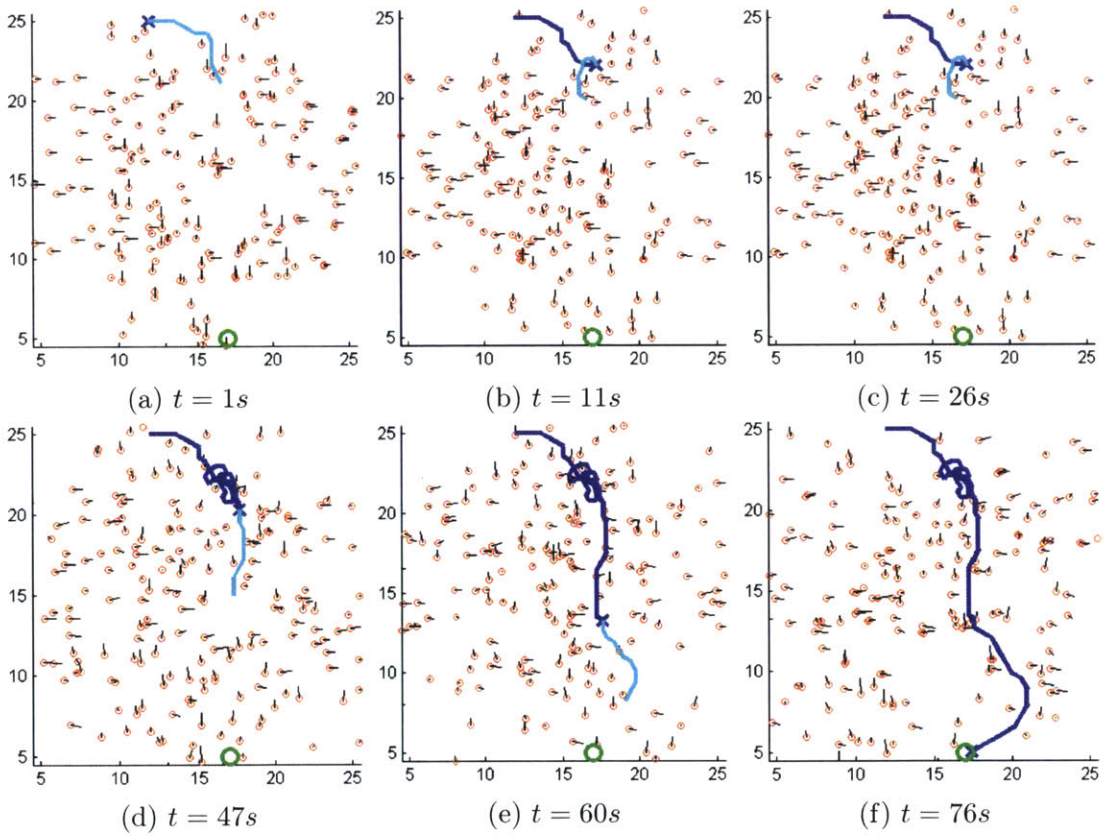


(a) $t = 1s$ (b) $t = 12s$ (c) $t = 26s$

(d) $t = 40s$ (e) $t = 55s$ (f) $t = 80s$

Figure 3-21: Medium Planner simulation result 2

In the third simulation, the pedestrians form two flows: one vertical from bottom to top, and the other one horizontal from left to right. The robot starts from the top and its destination is in the bottom part. There is no pedestrian flow to make use of, so the robot can only wait for the hollows in the adverse flow and move. Note the trajectory tangles near $(16, 21)$.



(a) $t = 1s$ (b) $t = 11s$ (c) $t = 26s$

(d) $t = 47s$ (e) $t = 60s$ (f) $t = 76s$

Figure 3-22: Medium Planner simulation result 3

Chapter 4

Conclusions

In this thesis, we discussed two engineering challenges faced by service robots in crowded public environments. Connected with a distributed sensor network and a central server, a robot can transit smoothly between ground and escalators/moving walkways, and navigate optimally in the presence of a great number of passengers without cause much annoyance.

Public service robots are designed to work in public places, such as shopping malls and train stations, to provide daily services to humans. Because of their public exposure and frequent interaction, they have a lot of potential in social and economical impacts. The public places they are deployed in usually share similarities in functionalities and facilities, and the concentration on a few tasks and a higher budget rule out some constraints on robot design. However, the existence of many moving passengers adds to the requirements on robotic perception, control and navigation. Conventional methods cannot adequately meet these requirements.

We propose to tackle the challenges with the synergy of a distributed sensor network and a central server. The distributed sensor network and the central server can be seen as an upgrade of the existing security camera systems commonly seen in public spaces like banks and airports. The higher initial installation budget allows for better-quality sensors that are embedded to the environment. The service robots deployed in the environment act as mobile sensors. Any object would be reported by multiple sensors, and the multiple-view not only reduces chances of occlusion but

also improves estimation accuracy. The central server is capable of synthesizing the information collected by the sensor network, and produce real-time estimates of every human and robot in a public environment, which contributes to control and navigation of the robots. From past data, models for passenger motion could be identified statistically, which are applied to estimate and predict passengers' states. The server can also perform centralized scheduling and planning for all robots in the same environment.

To better serve the users, any robot deployed in a public environment should be able to follow its user to any place she/he goes. This requires the robot to access the facilities that are designed for human, for example, escalators and moving walkways. However, the transition between fixed floors and moving walkways introduces severe impacts that makes a wheeled robot vulnerable to fall over. The impacts are strong, and the robot's velocity controller cannot handle such instantaneous disturbances. By inserting a torsional spring between the motor shaft and the wheel axle, the impacts in the tangential direction of the wheel can be absorbed. This Series Elastic Transmission serves as a mechanical low-pass filter, and at high frequencies, the wheel is virtually disconnected from the rest of the robot, so that the wheel speed can change freely while the robot moves unaffected. The compliant coupler offers high-fidelity torque measurement at the same time. The Unified Velocity Control is devised to work with the Series Elastic Transmission. The outer loop of the Unified Velocity Control regulates the robot's ground speed, while the inner loop enhances the transmission system's damping. Simulation shows that a robot equipped with Series Elastic Transmission and Unified Velocity Control can maintain a consistent speed during transitions. Prototype experiments are conducted, but the results are not very ideal. A possible reason is that the prototype robot uses a discrete-time control system while the design is conducted for continuous-time control, and the lags deteriorate the performance.

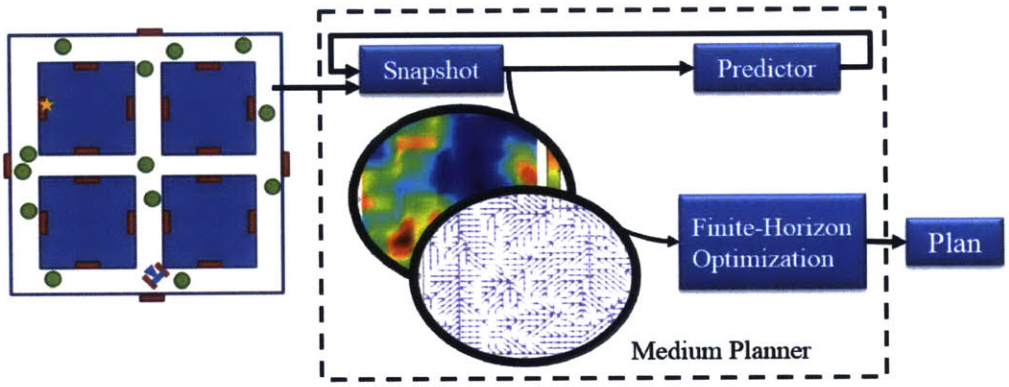
A public service robot needs to plan its way to perform tasks. In a crowded environment, the robot must find a path to move efficiently while avoid causing much annoyance to nearby humans. The task is not easy because the uncertain

dynamic environment consists of many passengers and robots, and the gap between global path optimality and local path optimality is significant. We put forward an online planning scheme facilitating the synergy of the distributed sensor network and the central server. Three planners form a hierarchical structure, and the plan from an upper level planner is refined by a lower level planner. The top layer, namely the Macro Planner, adopts abstract models, and finds a sequence of key locations for the robot to visit. The intermediate layer, namely the Medium Planner, treats the crowds using the analogies of fluids, and generates waypoints to describe the transition between two consecutive key locations. The bottom layer, namely the Micro Planner, grounds the plan with actual robot dynamics and nearby passengers' states, and produces a continuous trajectory. The intermediate layer is novel because it bridges the global plan optimality with the local plan optimality. The Medium Planner is formulated as a finite-horizon optimization, whose cost is defined with respect to two concepts, the Crowd Pressure and the Navigation Map. The Crowd Pressure is a measure of the effort to move in the presence of crowds. Low pressure indicates either low crowd density, or small relative velocity between a robot and its surrounding people. The Navigation Map is an adaptation of the classical navigation function to dynamic environment. It tells a robot the best velocity to take at the robot's current position. The optimization is a two-point boundary-value problem, and we develop an iterative method to obtain approximate solutions. Simulations show that the simple formulation is pertinent to many typical navigation scenarios, and the planner is robust in terms of parameters and estimation errors.

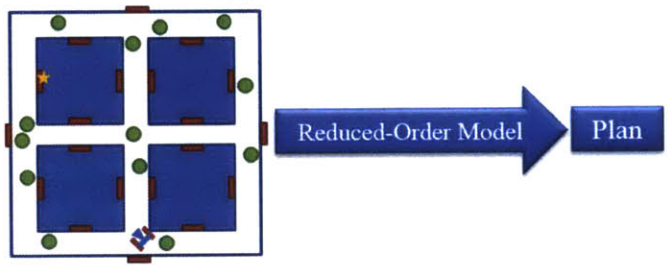
There are many potential applications for the technologies presented in this thesis. For example, in airports one may often see the staffs helping disabled passenger to navigate around and board the plane. This kind of service are not regularly needed, but an airport has to hire people exclusively for that, which costs a lot. With the synergy of a distributed sensor network and a central server, an autonomous or semi-autonomous robot is able to locate and navigate, thereby providing convenient mobility service for disabled travelers. When there is no call, the robot could routine services such as security surveillance and luggage transportation. East Japan Railway

Company has planned to build a train station with embedded sensors to test some public service robots, as a preparation for 2020 Tokyo Olympics and Paralympics.

In the simulation of the Medium Planner, we observe that the iterative solver for the optimization (3.4) may not be able to find an optimal solution in real-time. Sometimes the solver takes several seconds to compute a path, while the update interval is one second. We have modified the solver so that it can come up with a solution quickly and continuously improve it. However, we are not satisfied with this any-time algorithm. See Fig. 4-1a, current methods iteratively predicts future



(a) iterative solver



(b) reduced-order model

Figure 4-1: Solvers for Medium Planner

crowd states and searches for an optimal path. When the crowd is dense where the robot's speed is quite limited, the solver wast a lot of time examining the small area. Intuitively we feel that the crowd motion are statistically predictable, and the optimal plans strongly depend on the current state of the environment. Therefore we propose to identify a reduced-order model of the iterative solver, so that whenever the estimated states of nearby obstacles are available to the robot, it can feed the

information to the reduced-order model and instantaneously obtain the optimal plan, or an suboptimal plan that is very close to the optimal one.

Bibliography

[1] Wolfram Burgard, Armin B Cremers, Dieter Fox, Dirk Hähnel, Gerhard Lakemeyer, Dirk Schulz, Walter Steiner, and Sebastian Thrun. Experiences with an interactive museum tour-guide robot. *Artificial intelligence*, 114(1):3–55, 1999.

[2] Woojin Chung, Gunhee Kim, and Munsang Kim. Development of the multi-functional indoor service robot psr systems. *Autonomous Robots*, 22(1):1–17, 2007.

[3] James L Crowley. World modeling and position estimation for a mobile robot using ultrasonic ranging. In *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on*, pages 674–680. IEEE, 1989.

[4] Sjur D Flåm. Existence results and finite horizon approximates for infinite horizon optimization problems. *Econometrica: Journal of the Econometric Society*, pages 1187–1209, 1987.

[5] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4(1):23–33, 1997.

[6] Masahiro Fujita. On activating human communications with pet-type robot aibo. *Proceedings of the IEEE*, 92(11):1804–1813, 2004.

[7] Dylan F Glas, Yoichi Morales, Takayuki Kanda, Hiroshi Ishiguro, and Norihiro Hagita. Simultaneous people tracking and robot localization in dynamic social spaces. *Autonomous Robots*, pages 1–21, 2015.

[8] LF Henderson. On the fluid mechanics of human crowd motion. *Transportation research*, 8(6):509–515, 1974.

[9] Reiner Horst and H Edwin Romeijn. *Handbook of global optimization*, volume 2. Springer Science & Business Media, 2002.

[10] J.L. Jones. Robots at the tipping point: the road to irobot roomba. *Robotics Automation Magazine, IEEE*, 13(1):76–78, March 2006.

[11] T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro, and N. Hagita. A communication robot in a shopping mall. *Robotics, IEEE Transactions on*, 26(5):897–913, Oct 2010.

[12] Tae Hyon Kim, Kiyohiro Goto, Hiroki Igarashi, Kazuyuki Kon, Noritaka Sato, and Fumitoshi Matsuno. Path planning for an autonomous mobile robot considering a region with a velocity constraint in a real environment. *Artificial Life and Robotics*, 16(4):514–518, 2012.

[13] Daniel E Koditschek and Elon Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 11(4):412–442, 1990.

[14] Yoram Koren and Johann Borenstein. Potential field methods and their inherent limitations for mobile robot navigation. In *Robotics and Automation, 1991. Proceedings., 1991 IEEE International Conference on*, pages 1398–1404. IEEE, 1991.

[15] Edwin Olson. Apriltag: A robust and flexible visual fiducial system. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3400–3407. IEEE, 2011.

[16] Elena Pacchierotti, Henrik I Christensen, and Patric Jensfelt. Design of an office-guide robot for social interaction studies. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4965–4970. IEEE, 2006.

[17] Erwin Prassler, Jens Scholz, and Paolo Fiorini. A robotics wheelchair for crowded public environment. *Robotics & Automation Magazine, IEEE*, 8(1):38–45, 2001.

[18] Pranav Rane, Varun Mhatre, and Lakshmi Kurup. Study of a home robot: Jibo. In *International Journal of Engineering Research and Technology*, volume 3. ESRSA Publications, 2014.

[19] Yoshiaki Sakagami, Ryujin Watanabe, Chiaki Aoyama, Shinichi Matsunaga, Nobuo Higaki, and Kikuo Fujimura. The intelligent asimo: System overview and integration. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2478–2483. IEEE, 2002.

[20] Marija Seder and Ivan Petrovic. Dynamic window based approach to mobile robot motion control in the presence of moving obstacles. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 1986–1991. IEEE, 2007.

[21] Andrew Smyth and Meiliang Wu. Multi-rate kalman filtering for the data fusion of displacement and acceleration response measurements in dynamic system monitoring. *Mechanical Systems and Signal Processing*, 21(2):706–723, 2007.

[22] Adrien Treuille, Seth Cooper, and Zoran Popović. Continuum crowds. *ACM Transactions on Graphics (TOG)*, 25(3):1160–1168, 2006.

[23] S Vechet and V Ondrousek. Motion planning of autonomous mobile robot in highly populated dynamic environment. In *Mechatronics*, pages 453–461. Springer, 2012.

[24] Huijing Zhao and Ryosuke Shibasaki. A novel system for tracking pedestrians using multiple single-row laser-range scanners. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 35(2):283–291, 2005.