The Network Robot System:
Enabling Social Robots in the Real World

A dissertation submitted to
THE GRADUATE SCHOOL OF ENGINEERING SCIENCE
OSAKA UNIVERSITY
in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY IN ENGINEERING

BY

DYLAN FAIRCHILD GLAS

MARCH 2013

Abstract

The field of social human-robot interaction has grown over the past decade to become a mainstream research topic in robotics, and great advances have been made in many of the essential technologies required for the realization of robots able to interact with people as social peers.

Yet, such robots are seldom seen outside of the laboratory, indicating that there are still limitations preventing the real-world deployment of social robots to provide services in society. In this thesis, I will address several key limitations and present a software framework designed to enable social robots to be deployed in real environments using today's technology.

What I propose is a "Network Robot System" approach, in which environmental sensor networks and ambient intelligence systems are used to augment a robot's recognition capabilities for navigational interactions; a human operator is employed to assist a fleet of robots in difficult recognition or judgment tasks for conversational interactions; and centralized servers and knowledge resources are utilized to coordinate robots and deliver personalized services.

Each of the topics addressed in this thesis is grounded in practical experience, and all of the concepts and elements presented here have been tested not only in the laboratory, but in experiments and demonstrations of social robots in real public and commercial spaces. Based on this experience, I present design requirements and working prototypes of each element in the framework and show several examples of their use in field experiments and demonstrations of multiple social robots in real public and commercial spaces.

Table of Contents

Chapter 1

Introduction

Robots have always evoked an image of a “mechanical human” - a humanoid machine with the ability to think, act, and communicate with people. While the image of humanoid robots as companions, servants, or friends to people is strong in books, movies, and popular culture, the reality of robots today is quite different. The robots in our world today wield machine tools in factories, vacuum our floors, collect soil samples on Mars, inspect oil pipelines under the sea, and drop bombs from above. But they are not social peers.

The growing field of Human-Robot Interaction (HRI) research is investigating the elements necessary to create robots which can engage with people in conversation and social interactions, which can take orders, provide advice or information, build trust, express emotions, and be recognized in some capacity as social peers, rather than devices, tools, or vehicles.

As great progress is being made in the development of elements such as social interaction models, recognition techniques, and natural language processing, the deployment of simple social robots to provide services in real commercial situations begins to appear more and more feasible.

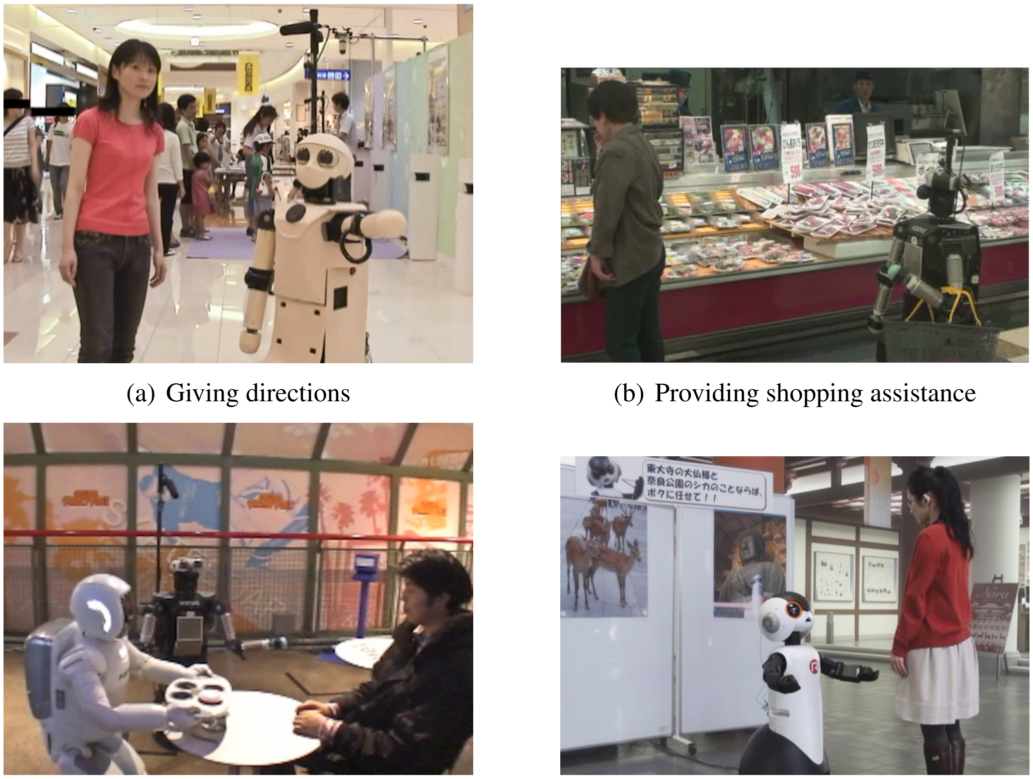
Why, then, do we not yet see social robots in our everyday world?

My objective in this dissertation is to address several of the key limitations which are holding back the real commercial deployment of social robots. To overcome these limitations, I propose a “Network Robot System” approach, in which environmental sensor networks and ambient intelligence systems are used to augment a robot’s recognition capabilities; a human operator is employed to assist a fleet of robots in difficult recognition or judgment tasks; and centralized servers and knowledge resources are utilized to coordinate robots and deliver personalized services.

1.1 Robots in Society - Potential Applications

1.2 Services

The focus of this work is on “social robots” in the role of peer-type human partners. Potential applications for these robots would thus include services that are typically provided by people. Fig. 1.1 illustrates a few examples of social robot application concepts we have demonstrated in field trials. This section will discuss examples of services that could be performed by social robots.



(a) Giving directions

(b) Providing shopping assistance

(c) Taking orders and serving customers

(d) Presenting tourist information

Figure 1.1: Examples of social robot applications.

Informational services

Although the physical embodiment of robots might suggest that they would be best suited for performing physical tasks, the value of robots in performing purely informational services should not be dismissed.

The physical presence of humanoid robots in people's social space gives them a great advantage over more passive, or "background" media such as posters or virtual agents in terms of presenting or gathering information.

Furthermore, humanoid robots can provide services in a way that is intuitive and needs no instruction manual - a humanoid robot can naturally make use of our built-in capabilities for social interaction which we already use when communicating with other people.

Some examples of informational services could include advertising of goods or services in a shopping center, answering people's questions at a convention or other event, soliciting responses to survey questions, providing public service announcements, or directing people in emergency situations such as evacuation of a building.

Companionship

Finally, social robots can provide service in the form of entertainment or companionship. In such situations, the value of the service primarily lies not in the content of the information provided by the robot, but in the feeling of the person interacting with the robot.

This sort of service has been demonstrated in the form of a robot playing with children at an elementary school[83], or by a robot accompanying an elderly person through a supermarket, chatting about topics like the weather while they are shopping [78].

Physical assistance

Social robots could provide physical assistance to people, helping them carry bags or groceries while shopping, or help to carry heavy luggage at an airport. Some studies have already investigated the use of "intelligent shopping carts" to provide both informational and baggage carrying services in a shopping scenario[60].

1.2.1 Environments

Some examples of field environments where social robots like these could be deployed include institutions such as schools, hospitals, museums, shopping malls, elderly care centers, train stations, supermarkets, airports, and so on.

Each of these locations has different characteristics - spaces such as schools and elderly care centers contain relatively "closed" communities, where a social robot could engage in repeated interactions with individuals over the long term and would benefit from having personal knowledge about individual members. Environments like train stations and supermarkets, on the other hand, are "open" spaces where large numbers of people are constantly passing through. While there may be many repeat visitors to these places, it might be more useful for a social robot to be able to identify a person's needs from their behavior and use this information to offer services to anonymous visitors.

These environments vary in scale as well, from small, enclosed spaces such as a classroom, to large, open areas like the central corridor of a shopping mall. Tracking and identification of people can be particularly difficult in these large spaces, if a robot is to rely only on onboard sensors with limited range.

Additionally, some environments in this list, such as shopping centers, are also highly changeable. When product displays, signs, and vendor booths are regularly moved around, map-based tasks such as localization and safety can become extremely difficult for a robot relying only on onboard sensors.

Noise levels also vary greatly between environments. A busy train station or shopping mall may have very high levels of background noise, which can interfere with speech recognition.

1.3 Research Challenges

Considering the state of the art in robotics technology today, there are a number of major challenges which prevent the immediate deployment of social robots in real applications. In this section, I will summarize some of these challenges and introduce the techniques I propose for enabling social robots to be deployed in spite of these limitations.

1.3.1 Navigation and Spatial Perception

Mobile robots designed to interact socially with people require reliable estimates of human position and motion. Social environments can require complex path planning, such as navigation through crowds [176], and even the seemingly simple act of approaching a walking person can actually be quite difficult, requiring precise tracking of the person's position while they are still quite far from the robot [139].

Movement and positioning can also contain implicit information about a person's intentions, social relationships, mood, and status. A person's walking speed, trajectory, proximity to other people, and body orientation can all provide information which can contribute to an understanding of social context.

Such knowledge could be used by service and communication robots to identify people who have lost their way or are in need of help, to stay out of the way of people in a hurry, to identify group leaders for guidance or sales applications, to understand when the robot is the center of attention and when it is being ignored, to identify booths in an exhibition or exhibits in a museum that a person has missed, and for many other purposes.

A major impediment to performing such tracking, however, is the fact that on-board sensors are usually effective only in tracking people close to a robot - limited resolution and occlusions make it quite difficult to track people at a distance, which is necessary if a robot is to approach or interact with people walking through any large space.

In this work, I propose a solution to the issue of accurate tracking, presenting a **human tracking system** which uses laser range finders embedded in the environment, rather than on-board sensors. I will present an algorithm which not only identifies the locations of people with high precision, but also estimates the direction in which they are facing.

Not only can this approach provide robust tracking of people over a wide area, but I will also show that as part of an "ambient intelligence" system, it can provide a basis for modeling and anticipating the behavior of people in a social space, which is a powerful tool for enhancing the ability of robots to provide services.

1.3.2 Conversation and Uncovered Situations

While ubiquitous sensors and ambient intelligence can provide valuable assistance to the navigational and spatial perception problems faced by mobile service robots, they are of little help in assisting robots with the conversational aspects of social interactions.

There are two essential limitations which make the deployment of autonomous conversational robots difficult at present.

The first is speech recognition, which is one of the most essential functions for conversational interactions, and also one of the most difficult to achieve in real-world environments. A large field of research has developed to address the problems of speech recognition and natural language processing, and many commercial products exist for these purposes.

Yet, at the time of this writing, problems such as noise rejection and utterance boundary detection are still unsolved for noisy spaces such as shopping malls, and this is a particular problem when using a microphone located on a robot rather than in front of the speaker's mouth [153].

In addition to speech, other tasks such as gesture recognition, emotion recognition, gaze direction tracking, and estimation of interest or intention, can all be useful in social situations.

The second major limiting factor is the complexity of social situations. Even if a robot is programmed to handle a great variety of contingencies in an interaction, there is always a risk that something unexpected will happen, and the robot will be put into an "uncovered situation" for which it has no ability to react meaningfully.

In this work, I propose the use of a **human operator** to assist semi-autonomous robots with difficult tasks such as speech recognition, and also to help the robot recover from errors and uncovered situations. While many studies have used human operators as a temporary workaround for problems like these, I propose to consider the operator an essential part of the system, and to consider the degree of autonomy as a factor when designing social robot systems, rather than requiring complete autonomy from the beginning. Increasing the level of robot autonomy will enable a single operator to supervise and assist larger teams of robots at once.

The real-time nature of conversational interactions, however, provides particular challenges for multiple-robot teleoperation - our experiences from field deployments show that operator unavailability or delayed response time can result in unacceptably slow conversational interactions.

Thus, in addition to proposing and demonstrating a teleoperation system for teams of semi-autonomous social robots, I will also present techniques for addressing these timing issues, both through coordination of robot behaviors, and through user interface design.

1.3.3 Autonomy and HRI Models

Although external sensor networks and assistance from human operators can enhance and assist robot performance in many ways, the heart of a robot still resides in its internal program logic. To deploy social robots in the real world, a practical means for developing autonomous robot behaviors and scripts that enable high-quality social interactions is necessary.

A wide variety of techniques have been developed to enable natural and comfortable human-robot interaction. Human-appropriate motion planning [160] and proxemics studies [114, 183] have been conducted to determine models by which robot locomotion can be adjusted for smooth social interactions. Other studies have contributed models describing the role of gaze direction in human-robot interaction [157, 189, 116, 117].

Aspects of the conversational side of interactions have also been studied, including the use of “conversation fillers” [154] and nodding [158] during dialogue. Techniques have also been developed for specific situations, such as making deictic spatial references [166, 63].

Other studies have explored models for simulating lifelikeness. Some approaches use random noise, such as Perlin noise [129] to create lifelike motions. The lifelikeness of a robot can also be enhanced by modeling a robot’s internal drives [10] or using contingent behavior [187].

All of these techniques can be considered “HRI models”, and each contributes in some way to creating the detailed behaviors which enable humanlike and comfortable interaction. Yet, the practical matter of combining these techniques to create a coherent service application for a social robot poses a significant programming and maintenance challenge.

In this work, I will present an **application design framework** which enables many of these models and behaviors to be encapsulated into reusable components which can easily be assembled into larger-scale robot services which can easily be developed and maintained, even by non-programmers.

1.3.4 Coordination and Resource Allocation

Finally, robot service scenarios in the real world will most likely involve deployments of not one, but many robots. This raises a variety of issues which cannot be easily handled using only the system elements presented so far.

Coordination between robots is one major concern. If robots are operating in the same space, then the issue of path coordination needs to be resolved, in order to prevent deadlock and ensure smooth service operation. Other coordination issues include allocation of limited resources such as battery charging stations, and assignment of robots to provide requested services.

Multi-robot collaboration is another possibility for consideration. In this case, the robots need to have some means of communicating with each other to coordinate their actions, and it may be useful for a central planning agency to direct the robots in their collaborative task.

Finally, knowledge sharing is an important concern. Up-to-date information about customers, robot capabilities, and environments should be available to all robots and planning servers on demand.

To address these and other issues, I propose a **networked system architecture** which incorporates central planning logic for path, service, and resource allocation, as well as a set of globally-accessible knowledge stores from which data can be retrieved on demand.

1.4 Proposal

In this work, I propose a "Networked Robot System" approach to enabling social robotic applications, integrating all of the above elements:

1. Ubiquitous sensing and ambient intelligence to assist a robot's spatial perception for navigational interactions.

2. Assistance by a human operator to handle difficult recognition tasks and uncovered situations for conversational interactions.

3. On-board autonomy which can incorporate common HRI models within simple and manageable interaction scripts.

4. Centralized planning and shared knowledge stores to enable coordination and knowledge sharing among teams of service robots.

This is the first work to comprehensively address these issues for social robots. The contribution of this work lies not only in the proposal of a complete system which can flexibly enable the deployment of teams of social robots in real-world scenarios, but also in the practical implementation of such a system and in the lessons learned over several years of field deployments.

The ultimate goal of this work is to enable social robot deployments using today's technology, employing an approach which is not only appropriate for research and prototype systems, but which could also realistically be applied to real commercial robot applications.

1.5 Organization of Sections

The proposed "networked robot system" design combines ambient intelligence, supervisory teleoperation, social behavior logic, and central coordination of robot teams.

The ambient intelligence components are described in two chapters: Chapter 2 presents a human tracking system designed to robustly track the motion of pedestrians in social spaces, and Chapter 3 also presents examples of the use of this system to model and anticipate the spatial behaviors of people.

Next, Chapter 4 addresses the issue of supervisory teleoperation, describing a sliding-autonomy approach to enabling a single teleoperator to assist a team of several robots in conducting conversational and navigational interactions. A particular challenge of this approach is the time-critical nature of conversation, and a technique called “proactive timing control” is presented to prevent long wait times during conversation.

Continuing the discussion of teleoperation, Chapter 5 investigates another aspect of the time-criticality issue, revealing a phenomenon wherein a teleoperator’s time perception is distorted when engaging in complex tasks or heavy multitasking, and providing recommendations for user interface design when time criticality is a concern.

Chapter 6 addresses the third area: logic and autonomy within the robot. A framework is presented which enables reusable software behavior modules to be created which incorporate models of human-robot interaction. The framework then enables these modules to be easily used in the design of social interaction sequences in a visual programming language that is easily accessible to non-programmers.

Ultimately, these elements are tied together along with a networked system of planning servers and knowledge stores in Chapter 7, which describes the Network Robot System architecture and presents design requirements, implementation details, and a field experiment using the system.

Finally, the results of this work and its implications for the future are discussed in Chapter 8.

Chapter 2

Human Tracking

This chapter presents a system for simultaneously tracking the position and body orientation of many people, in order to support the spatial perception of robots for navigational interactions. The proposed technique combines data from a network of laser range finders mounted at torso height. In the tracking algorithm, an individual particle filter is created to estimate the position and velocity of each human over time, and a parametric shape model representing the person's cross-sectional contour is fit to the observed data at each step.

I will demonstrate the system's tracking accuracy quantitatively in laboratory trials and present results from a field experiment observing subjects walking through the lobby of a building. The results show that this method can closely track torso and arm movements even with noisy and incomplete sensor data, and I will present examples of social information observable from this orientation and positioning information that may be useful for social robots.

2.1 Introduction

A new class of service robots is emerging, one in which social interaction is a fundamental aspect of a robot's performance. Experimental field trials have demonstrated the possibility of robots acting as museum guides [15], receptionists [54], classroom assistants [83], guides in shopping centers, and other social roles in everyday life. As the natural-language and gestural communication capabilities of these robots improve, people's expectations of the robots' interaction skills will commensurately increase, and these robots will need to be responsive not only to speech, but to cues of physical motion and nonverbal communication as well.

Although a robot's on-board sensors can be used for some of these tasks, ubiquitous sensor networks can monitor larger areas and are subject to fewer size, power, and bandwidth restrictions. In this work, *laser range finders* are used for tracking people's positions as they are easier to install and less obtrusive than floor sensors, require far less processing than video tracking systems, and have a much higher precision and faster response time than RFID tracking or GPS.

To use these resources effectively, one goal of this research is to extract as much information as possible from this laser scan data. If nuances of a person's movement, such as the direction in which they are facing, can be extracted from the same laser scan data already used to determine their location, then information which is potentially useful for understanding social context will have been gained at no additional cost.

In this chapter I present an algorithm for tracking people using laser range finders, using a parametric shape model which includes arm positions and facing direction in addition to basic position tracking. The algorithms used in this system are described, and quantitative results of a laboratory experiment to characterize the system's tracking accuracy are presented. A second experiment was conducted in the entrance lobby of an office building, to demonstrate the system's performance with multiple subjects in natural walking situations. Qualitative results from that experiment are presented, illustrating the system's effectiveness in tracking many people simultaneously and suggesting types of social information that can be observed in the tracking results. Finally, considerations concerning performance tuning and real-time operation of the system are discussed.

2.2 Related Work

Human tracking itself is not a new field, and many aspects of the problem have already been explored extensively. Like many of its predecessors, our system tracks people by using particle filters to estimate their position and velocity. Particle filters are a well-known tool in the robotics community and have often been used in conjunction with laser scan data for the purposes of robot localization and mapping [28, 112] as well as human tracking. A general overview of applications of particle filters in robotics can be found in [171].

Much of the human-tracking research to date has been based on leg tracking, for both mobile robotics [143, 181] and environmental monitoring [11, 23, 190]. This has historically been motivated in part by the fact that many robots use laser sensors for obstacle avoidance, and for that reason already have laser sensors mounted near the ground. However, their visibility is often limited by those same obstacles, making floor-level sensors a good choice for on-board robot systems but less so for wide-area environment monitoring in cluttered spaces.

In our work, the laser sensors constitute an essential part of a ubiquitous sensor network used exclusively for human tracking in real environments. For this reason, it is important for the sensors to be mounted higher, above furniture and ground clutter. Thus the sensors in our system are mounted at a height of 85-90 cm, where the arms and torso can be clearly observed.

Although less common than leg-tracking, torso-level tracking is not without precedent in research. For example, Fod *et al.* created a system using a Kalman filter to track people's trajectories with waist-height laser scanners [38], and Almeida *et al.* developed a real-time torso-level laser-based human tracking system utilizing particle filters in [2]. These systems, however, were focused specifically on position tracking, whereas our work is concerned with observing body orientation and pose in addition to position.

2.3 Position Tracking

Our algorithm was developed to track both human position and orientation. The strategy of this algorithm is to first estimate each person's position using a particle filter, and then to fit a shape model, representing the person's body orientation and arm positions, to the observed contour data.

Our initial approach to this problem had been to calculate both position and orientation using the particle filter. This resulted in an unacceptably slow system for our real-time applications. However, we observed that a majority of the computation time for each particle was being spent on orientation calculations.

In fact, the edge-based calculations used for orientation are not particularly well-suited for use in a particle filter. For position, calculations are efficient because their likelihood distributions are stable over time (regions are clearly defined and change slowly), relatively smooth in space, and easy to calculate from raw sensor data. Edge-based likelihood distributions are more complex to calculate, not stable over time (the number and placement of detected points can change rapidly between frames with a great deal of randomness), nor are they smooth in space, as the best-fit orientation can change wildly over even small variations of the assumed center position. It is thus difficult to obtain a meaningful average orientation value over a scattered set of particles.

In our technique, the orientation calculations are highly dependent upon position, but the position calculations do not depend on orientation. Thus the orientation calculations can be removed from the particle filter and performed after the position estimate is evaluated. Having done so, at each time step we need only calculate orientation once for each particle filter, rather than once for each particle. In addition, by removing variables from the particle filter we are able to reduce its dimensionality, consequently reducing the number of particles necessary for accurate tracking. By separating the calculations into a two-step process, we are thus able to dramatically increase real-time performance. More details on this topic can be found in Sec. 2.7.2.

2.3.1 Detection and Association

A common problem in tracking is the association between detected features and objects being tracked. In our algorithm, each person is tracked by a single particle filter. Doing so enables these feature-object associations to be handled implicitly by the particle filters, which follow the detected features over time. Thus explicit feature-object associations only need to be made when creating new particle filters for previously untracked humans.

To identify new humans, the raw data is segmented at every time step, to extract continuous segments of foreground data roughly corresponding to expected human widths. Clusters of these patterns are grouped together and flagged as human candidates. Candidates coinciding with humans already being tracked are removed from the list, and those remaining are propagated to the next time step, where they are merged with the candidates detected during that step. If a human candidate survives beyond a threshold number of time steps, it is considered to be a valid detection, and a new particle filter is assigned to that location, initialized with the position and velocity of the human candidate it replaces.

The removal process is much simpler than the addition process. When the particles within a filter spread out beyond a defined dispersion threshold, or when their average likelihood value goes below a defined probability threshold, that particle filter is assumed to no longer be tracking a human, and it is removed.

2.3.2 Particle Filtering

A key component of our tracking algorithm is the particle filter, the basic principles of which will be very briefly explained here. For a more in-depth explanation, [173] provides a thorough treatment of particle filters and many other state estimation techniques.

Particle filtering is a method of estimating the state $\mathbf{x}_t$ of a system by using a cloud of "particles", each of which represents a hypothesis about that state. The following four-step procedure is performed at each iteration of a particle filter.

1. **Update** The state of each particle is updated by applying an internal *motion model*, reflecting the dynamics of the system, to the previous state estimate. The motion model used in our work is described in Section 2.3.4.

2. **Assign Weights** Particles are assigned weights representing their relative likelihoods according to a *likelihood model*. The likelihood model provides an approximation of the conditional probability $p(z_t|\mathbf{x}_t^{[m]})$ for particle $m, (m = 1..M)$ and measurement vector z_t taken at time step t of the particle filter. Our likelihood model is described in Section 2.3.5.

3. **Estimate State** An estimate of the state is then calculated, generally as a weighted average of the states of the particles.

4. **Resample** Particles are removed or propagated based on their weights to produce a new set of particles which more accurately reflects the true state of the system. Several resampling techniques exist; our system uses the sampling importance resampling technique [58].

In this way, the cloud of particles converges on the most likely state and follows it over time.

2.3.3 State Model

The state vector tracked by the particle filter consists of four variables: x, y, v, and θ. The variables x and y represent the position of the human being tracked. Although the speed v, and direction θ of motion could be calculated *a posteriori* from the position data, these variables are included in the state and updated at every step to enable the person's position to be projected forward through time for more accurate tracking. These variables are used in the motion model, described below.

2.3.4 Motion Model

At every update of the particle filter, each particle is propagated according to a motion model. The purpose of this motion model is to approximate the probability of a state $\mathbf{x}_t$ based on the previous state $\mathbf{x}_{t-1}$.

As has been observed in [13], the modeling of human motion presents difficulty because it is neither Brownian in nature, nor can it be modeled as a smooth linear function, since people may stop or change direction abruptly. Thus, as a compromise between the two, a Gaussian noise component is added to each particle's v and θ values to capture the randomness of human motion. We then propagate the (x,y) motion linearly according to the resultant v and θ values of the particle.

2.3.5 Likelihood Model

The purpose of the likelihood model is to approximate the value of $p(z_t|\mathbf{x}_t^{[m]})$. In this case, the measurement vector z is an array of raw sensor range measurements. An effective likelihood model must provide a robust likelihood estimate in spite of noisy sensor data, partial and full occlusions, and the irregular and varying shapes of human bodies.

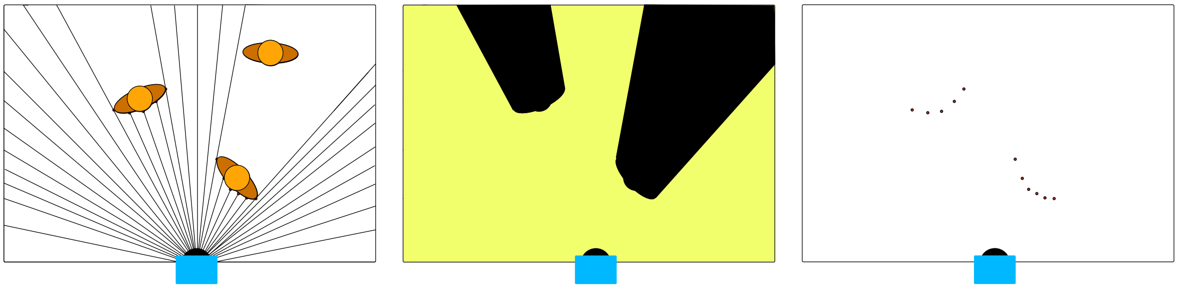


Figure 2.1: A typical single-sensor laser scan. (Left) The positions of humans relative to the scanner can be seen. (Center) Occupancy information. (Right) Edge information.

Laser scan data provides two qualitatively distinct types of information useful for estimating human positions: *occupancy information*, indicating whether a certain point is occupied or empty, and *edge information*, indicating a contour which may correspond with the edge of a detected object. Fig. 2.1 illustrates the distinction between these two kinds of information.

To determine likelihood values from the raw sensor data, it is first necessary to create a background model. Our system uses an adaptive background model which is updated over time to determine the best estimate of the true background distance. Occupancy likelihood is then determined by dividing the world into three regions: "open", "shadow", and "unobservable". The "unobservable" region is beyond the background model for that sensor, and thus can contribute no information. The "open" region has been observed by the sensor to be unoccupied, and the remaining space is considered "shadow". Note also that every "shadow" region lies behind an "edge".

The likelihood model used to compute $p(z_t|\mathbf{x}_t^{[m]})$ is expressed in Eq. 2.3.1 and 2.3.2 and includes components reflecting both occupancy and edge information.

$$p(z_t|\mathbf{x}_t^{[m]}) = \frac{1}{n_{sensors}} \sum_{i=1}^{n_{sensors}} p_i(z_t|\mathbf{x}_t^{[m]}) - p_{collocation} \tag{2.3.1}$$

$$p_i(z_t|\mathbf{x}_t^{[m]}) = \begin{cases} p_{shadow} + p_{edge}(z_t|\mathbf{x}_t^{[m]}) & \text{in a shadow region} \\ p_{open} & \text{in an open region} \end{cases} \tag{2.3.2}$$

For a point in a shadow region (strictly speaking, we consider only those regions wide enough to contain a human), the likelihood in Eq. 2.3.2 is calculated as the sum of a constant value p_{shadow} and a likelihood $p_{edge}(z_t|\mathbf{x}_t^{[m]})$, calculated as a normal distribution centered upon a point located one approximate human radius behind the observed edge. (In our calculations a value of 25cm was used.) This reflects the fact that people are highly likely to be found just behind an observed edge, yet can plausibly exist anywhere in a shadow region (*e.g.* the occluded person in Fig. 2.1).

For a point in an open region (or in a shadow region too narrow to contain a human), the likelihood is theoretically zero, but for reasons described below is set to a small but nonzero constant value p_{open}. In this case, edge information is irrelevant.

Finally, in Eq. 2.3.1, these likelihood values are averaged across all $n_{sensors}$ sensors for which the proposed point lies within the sensor's "open" or "shadow" range, *i.e.* not "unobservable" to that sensor. To prevent two particle filters from tracking the same human, a value $p_{collocation}$ is subtracted from this result. Its value is calculated as a sum of normal distributions surrounding each of the other humans, based on the list of human positions from the previous time step.

Error Tolerance

In an ideal system, the "open" regions could be assigned a likelihood value of zero. However, in real systems there are many possible sources of error, such as calibration errors (the exact position and angle of each sensor may not be properly calibrated, leading to imperfect alignment of shadow regions), measurement errors (some textures of clothing cause noisy sensor readings and thus apparent gaps in people's bodies), timing synchronization errors (sensor data feeds are sent in real-time over a network and may arrive asynchronously, causing old data to be mixed with new), and hardware or transmission errors (which produce occasional bursts of sensor noise). The binary discretization of space into "open" and "shadow" regions is thus a slightly imperfect representation of reality. Consequently, we set the likelihood of "open" regions to a small but nonzero value p_{open}. This adds a small amount of resilience to the system, allowing particles to survive outside of the shadow regions for a short time in order to provide smoother performance with respect to such sources of error. This does not destabilize the particle filter since the likelihoods of these particles are substantially lower, and particles lying outside of the shadow regions for too long will naturally be culled in the resampling process.

2.4 Orientation Estimation

Our algorithm for calculating a person's orientation uses a parametric shape model, which we describe in Section 2.4.1. An angular array representation, presented in in Section 2.4.2, is used to store laser scanner data as a set of edge distances. As a tool for our calculations, an empirical distribution of expected distances for such an array, relative to the person's forward-facing direction, was generated based on laboratory motion-capture data. We describe the derivation of this distribution in Section 2.4.3.

The computation itself consists of first determining a rough estimate of body orientation, described in Section 2.4.4, based on the observed contour shape and the empirical distance distribution mentioned above. The second step, explained in Section 2.4.5, is to determine the individual arm angles, based on this rough estimate. The arm angles are then used to generate a refined estimate of orientation. Finally, Section 2.4.6, presents a technique for reducing accidental 180-degree reversals by considering motion direction and velocity.

2.4.1 Theoretical Shape Model

Large variations in cross-sectional contour shape were observed between subjects. This is due in part to individual differences in body shape, and also to differences in height. For example, arm motion is more pronounced for taller subjects, and their arms sometimes disappear if their hands briefly swing out of the scan plane.

Clothing also affects contour shape. For example, a loose shirt or a heavy coat can make a person's torso appear unusually large or asymmetrical, as can a backpack or purse.

Taking these factors into consideration, the amount of variation between subjects makes it difficult to develop a precise, yet generalizable, model. Thus a simple three-circle model was used for determining body orientation.

Our model is illustrated in Figure 2.2. A central, large circle represents the person's torso, and two smaller circles represent the arms. This model has six parameters which can

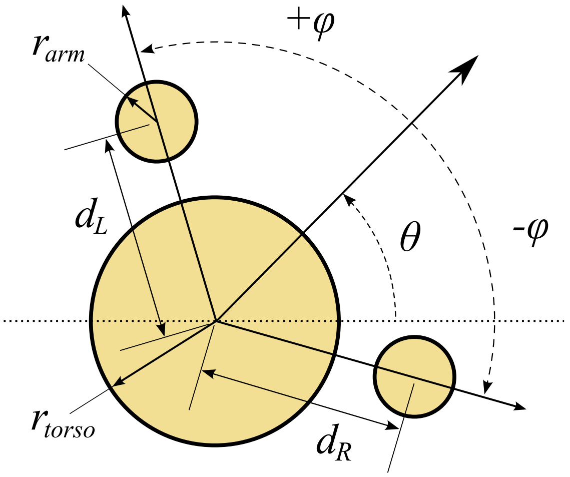


Figure 2.2: Our three-circle model, with the six variable parameters indicated.

Table 2.1: Model Parameters

Parameter	**Description**
θ	Average direction of body orientation
φ	Arm separation angle $\varphi_L = \theta + \varphi$ for left arm $\varphi_R = \theta - \varphi$ for right arm
d_L	Distance of left arm from body
d_R	Distance of right arm from body
r_{arm}	Arm radius
r_{torso}	Torso radius

be varied to best match a subject's cross-sectional body contour.

The parameters describing the state of this model are summarized in Table 2.1. The two parameters of primary interest to us are θ and φ. The other parameters are held constant for this application, although they can be estimated from the data if necessary.

We have designated θ to represent the angle midway between the two arms. When a subject is standing still, this coincides with the direction of torso orientation. While the subject is walking, the swinging of the arms and torso cause θ to oscillate around the direction of motion.

The parameter φ represents the angle of separation between each arm and the center angle designated by θ. This tends not to vary far from 90 degrees, as the arms swing in alternate directions during walking.

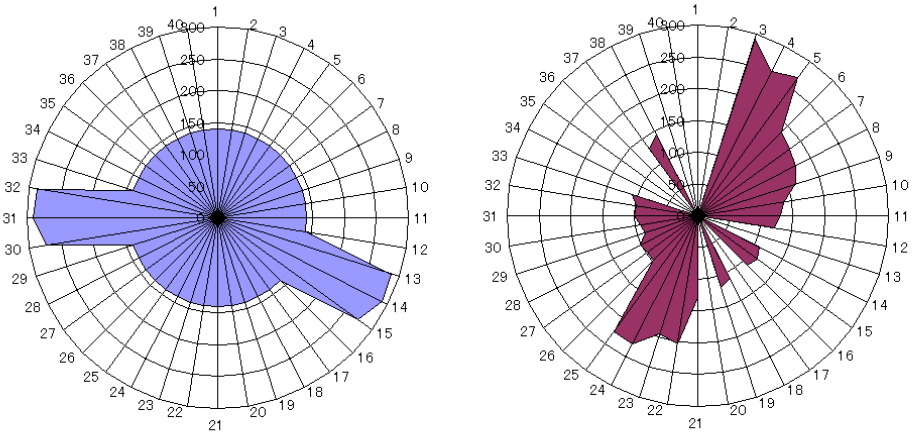


Figure 2.3: Examples of populated radial arrays. Left: Radial array reflecting an ideal human shape model. Right: Radial array populated with observed sensor data.

2.4.2 Radial Data Representation

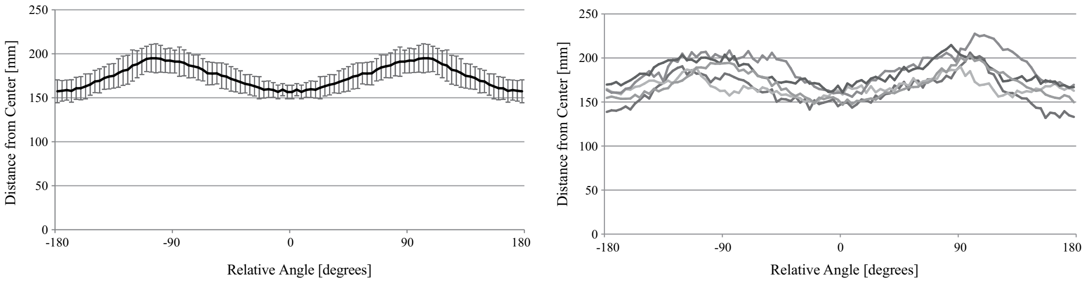
For these calculations, we need a way to represent 2D edge data in a consistent way for analysis. To achieve this, the information contained in these points is mapped to an angular array of distances. Distance values from the body center to the detected edge points are stored in an array of bins which represent an angular discretization of the space surrounding the estimated human position. For each angular division, the distance to the furthest observed data point within 50 cm of the estimated human position is stored in that bin. Fig. 2.3 illustrates such array representations of both the ideal shape model and a set of actual shape data. A linear representation of such an array is shown in Fig. 2.5(a).

2.4.3 Empirical Distance Distribution Model

A predictive distribution of radial distances is also needed for these calculations. An empirically-derived predictive distribution function representing average expected distance values as a function of angular deviation from θ was constructed from the laser scan and motion capture data gathered in the laboratory trials described in Sec. 2.5. This distribution function is shown in Fig. 2.4(a).

Two minutes of laser scan and motion capture data were recorded for each of five subjects. Each subject's angle at each time step was computed using the motion capture system, and a radial accumulator with 100 divisions (3.6 degrees each) was populated with the laser scan data for that time step, oriented relative to that angle. This distance data was collected over approximately 4500 time steps and averaged to determine an expected distance distribution function for each subject. These distribution functions are shown in Fig. 2.4(b).

Next, the data distributions were averaged between subjects. The resultant function was still somewhat noisy and asymmetrical. Making the assumption that this distribution should be symmetrical (and if there is a physiological reason for the asymmetry, to eliminate any bias based on handedness) the mirror images of the subjects' data distributions were also included in the average. Fig. 2.4(a) shows the standard deviation error bars for this combined distribution. The resultant distribution was then smoothed using a sliding 3-point window to reduce remaining noise. Finally, a constant offset was subtracted from the filter and it was normalized, steps which do not alter its effectiveness as a convolution filter.



(a) Predictive arm distribution filter

(b) Raw data used to derive filter

Figure 2.4: Predictive arm distribution filter showing standard deviation error, and raw data used to derive the filter

2.4.4 First-Pass Theta Determination

The strategy for the first approximation of theta involves two radial arrays. The first is populated with actual observed distance of data points from the body center, with the angular divisions corresponding to absolute angles. The second array holds the expected distribution of distances derived in Section 2.4.3, where the angular divisions represent angles relative to θ, the person's forward direction. By convolving these arrays with each other, we can compute a goodness-of-fit function between the predicted distribution and the observed distribution, as a function of θ. The maximum point of that function is the point where the observed data best fits with the expectation model, and is thus a good first-pass estimate for θ.

To begin, we need to construct an approximate model of the actual shape profile, beginning with the radial array shown in Fig. 2.5a. There will nearly always be angular divisions in the radial array with no points in them. Since we have no knowledge of the actual distances of these points, we set those bins to the average value across all occupied bins, to produce a model with no gaps, as shown in Fig. 2.5b. (This same array will be normalized and used later as a probability distribution function for arm positions, as explained below.)

This distribution, shown in Fig. 2.5c, is convolved with the data array shown in Fig. 2.5b to generate a function representing the goodness-of-fit between the observed data and the predicted data distribution. The maximum point of the resultant distribution indicates the θ value which gives the best match between the empirical distribution and the observed data.

One challenge in this determination of θ lies in the near-symmetry of the human body. Although the expected value of the arm angles is less than 180 degrees, the observed distribution and its 180-degree mirror image overlap significantly. Thus, particularly with noisy and incomplete data, it is possible that the best-fit angle is actually rotated 180 degrees from the true θ direction. To stabilize this variable, the secondary maximum in the θ likelihood function is designated as a second θ candidate. The angular distance from the previous θ estimate to the two new θ candidates is compared and the nearest neighbor selected as the first-order θ approximation. Correction of these reversals is discussed in Sec. 2.4.6.

2.4.5 Second-Pass Theta Determination

Using this rough θ estimate, the next step is to determine the arm angles φ_L and φ_R, which will be used for determining the final θ estimate. For this step, it is necessary to derive a probability distribution function (PDF) for the arm positions from the observed data.

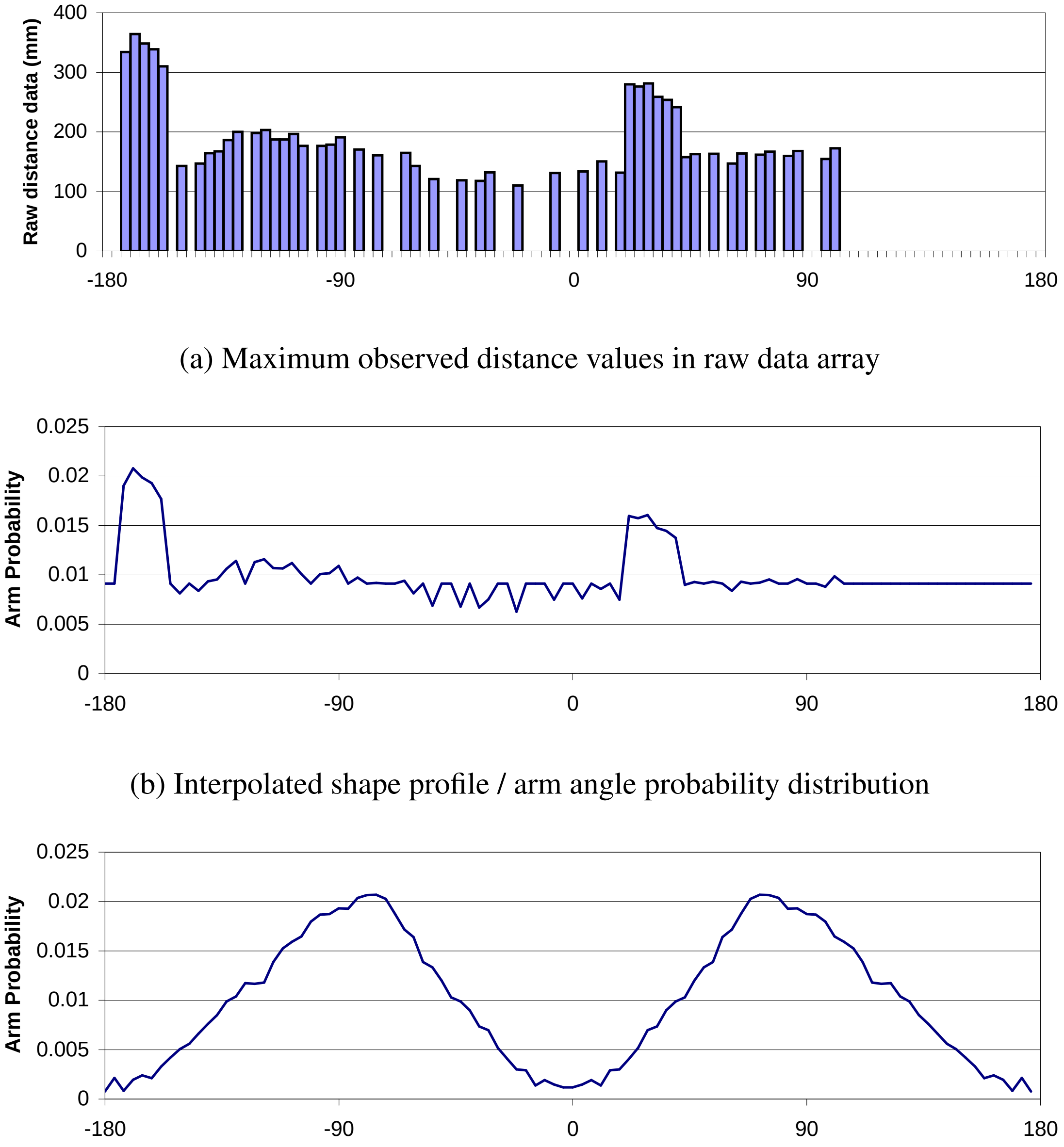
For this purpose, the shape profile model derived in the previous step can be used as a rough approximation of the arm position PDF, as it exhibits many of the essential features of such a distribution. For example, the presence of distant points indicates a high likelihood that an arm is in that direction. Likewise, the presence of closer-than-average points indicates a low likelihood of an arm being in that direction. Several points observed in a row give a higher-confidence estimate than a single point, high or low, and points with no data provide no information about the presence or absence of an arm. All of these features are found in both a theoretical PDF for arm distribution as well as the data array derived above. Thus, by normalizing that array, we obtain a rough approximation of that PDF.

The arm probability distribution in the radial array is then masked into two 180-degree regions by using trapezoidal masking filters on either side of the selected θ direction as shown in Fig. 2.6e (trapezoidal rather than rectangular masks were used for stability). These masks are multiplied with the data array from Fig. 2.5b to generate the two probability distributions shown in Fig. 2.6f. The peaks of these distributions are used as estimations of the left and right arm angles φ_L and φ_R, respectively. A refined estimate of θ is then calculated as the midpoint between these angles.

Note that at this point, if desired, the shape profile can be revisited to calculate parameters such as d_L, d_R, r_{arm}, and r_{torso}. However, this step is not necessary if θ is the only parameter of interest.

2.4.6 Correction of Reversals

One of the greatest difficulties in determining the person's facing direction lies in resolving the 180-degree ambiguity between forward and backward orientation. The human shape is nearly symmetrical, and even by eye it is sometimes quite difficult to discern front and back from laser scan data alone.

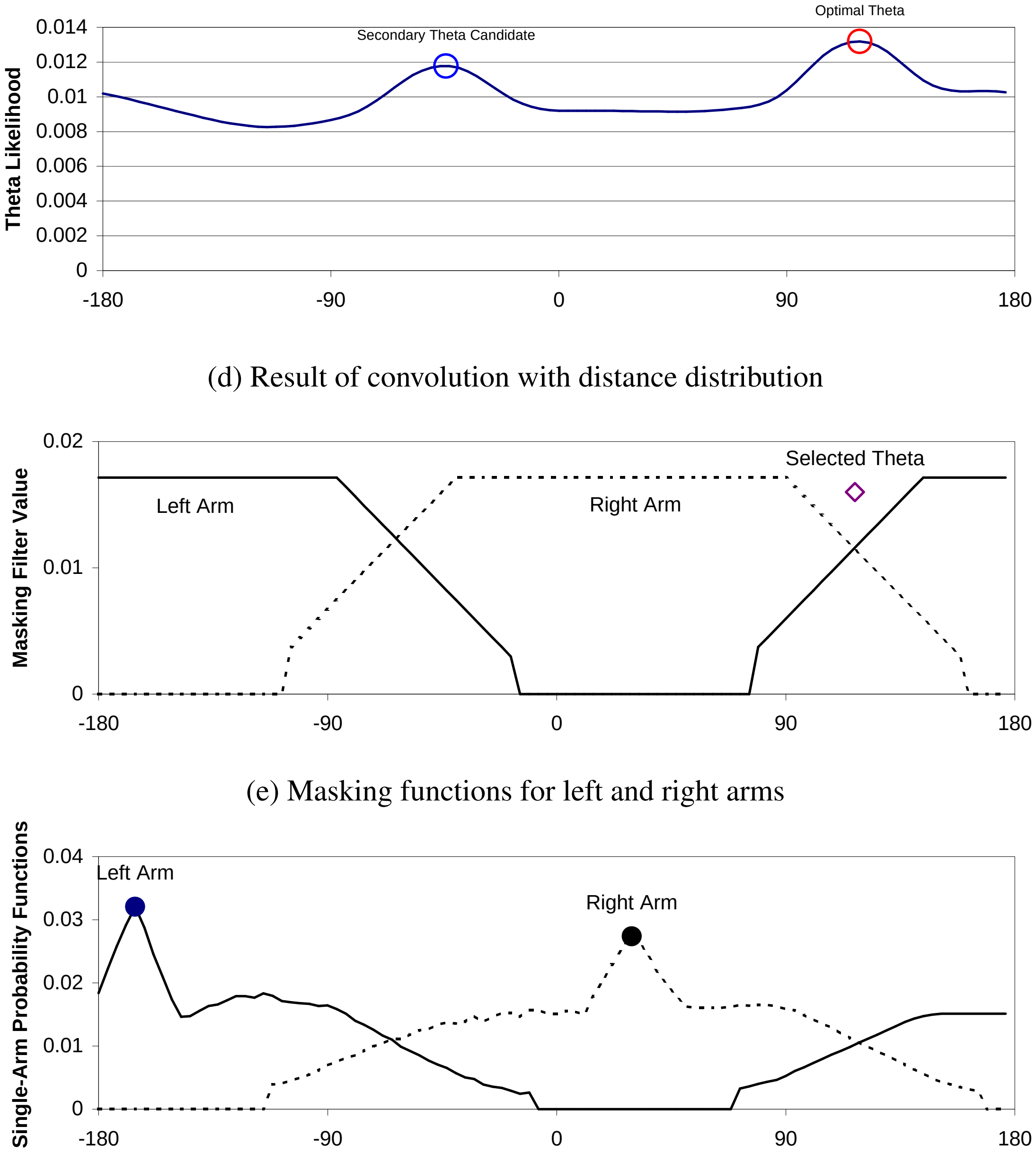


(a) Maximum observed distance values in raw data array

(b) Interpolated shape profile / arm angle probability distribution

(c) Empirically-derived theta-centric distance distribution

Figure 2.5: Initial steps in arm angle determination.



(d) Result of convolution with distance distribution

(e) Masking functions for left and right arms

(f) Probability functions for left and right arms

Figure 2.6: Final steps in arm angle determination.

To resolve this ambiguity, we utilize the assumption that motion direction generally tends to coincide with the forward orientation direction. We verified this assumption quantitatively using the data recorded in the trials described in Sec. 2.5.

By running the basic human-tracking program without any reversal correction, we generated a dataset of human positions and orientations. Reversals (defined as periods in which the directional error was greater than 90 degrees) were identified by comparing these results with the ground-truth data from the motion-capture system. A velocity distribution was then computed for each set of data points. The results of this analysis are illustrated in Fig. 2.7.

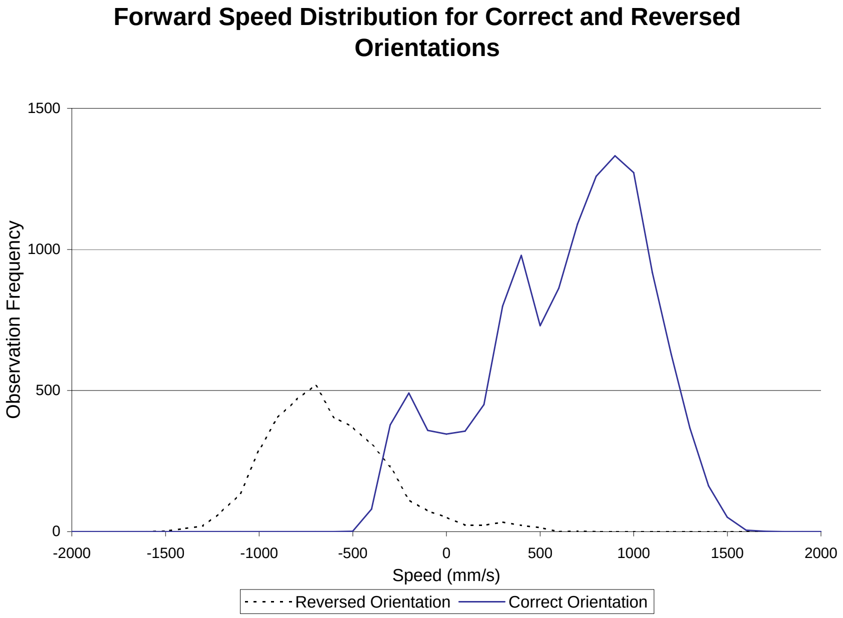


Figure 2.7: Distribution of forward velocity. The forward component of the velocity vector was calculated for each time-step, and a frequency histogram was computed using bin sizes of 100 mm/s. Nearly every observation with a forward velocity component below -500 mm/s was the result of a reversed direction estimate.

An examination of this velocity distribution reveals that retrograde motion at low velocities is common, probably due to a combination of actual motion, noise, and tracking lag of the particle filter; however, higher retrograde velocities (above 500 mm/s) are almost nonexistent. Thus, any retrograde motion larger than a threshold speed of 500 mm/s is interpreted as a reversal and corrected. A time-averaged velocity estimate is used to minimize the influence of noise.

2.5 Laboratory performance analysis

We performed an experiment in our laboratory to verify the accuracy of the human tracking system, and to gather empirical data to refine the reversal-detection and theta-approximation functions in our tracking algorithm.

2.5.1 Setup and Procedure

We used a Vicon motion-capture system to measure the accuracy of our laser tracking system. The Vicon system uses several infrared cameras to track reflective markers with an accuracy of 1 mm at a frequency of 60 Hz. Four SICK LMS-200 laser scanners were used, each set to a maximum range of 8 m, a distance resolution of 10mm, an angular range of 180 degrees, an angular resolution of 0.5 degrees, and a scan frequency of 37.5 Hz.

The space used for our experiment was a four-meter-square area with the four laser scanners situated outside the center of each edge of the square. The scan plane for each laser scanner was located at a height of 85 cm from the ground. Additionally, numbered markers were placed on the floor, as depicted in Fig. 2.8.

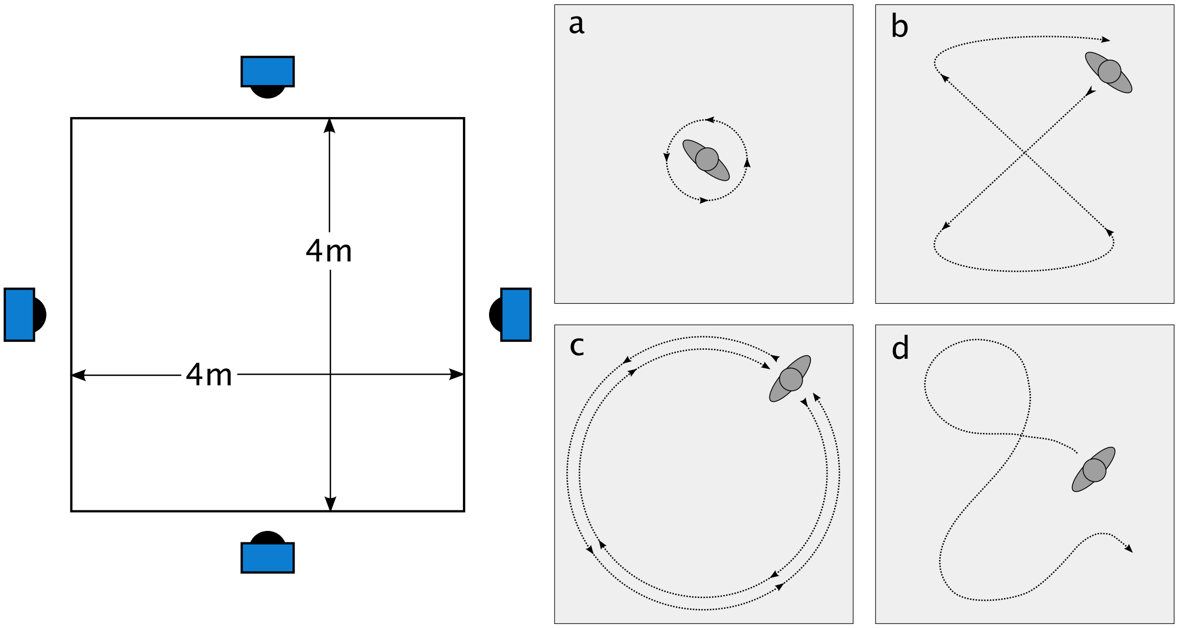


Figure 2.8: Floor layout for our laser tracking validation tests. Subjects were observed by four laser range finders while walking several patterns within a 4m by 4m square.

Five subjects were instructed to walk a series of patterns within the square. First, they stood in the center of the square and turned in a circle, stopping at each of the four cardinal directions for two seconds. Second, they walked figure-eight patterns, touching each of the numbered markers in order, twice. Third, they walked in a circular path inside the square, twice clockwise and twice counterclockwise. Finally, they walked randomly within the square until a total of two minutes had elapsed.

Each subject wore four reflective markers for the Vicon system. One marker was placed on the outside of each wrist, one on the subject's sternum, and one in the middle of the subject's back.

Raw data from each of the laser scanners was recorded, and the human tracking algorithm was executed offline.

2.5.2 Results

To compare the motion-capture data with the laser-tracking data, the midpoint between each subject's sternum and back markers was used as an estimate of the subject's body center. The absolute positional error (in the x,y plane) and absolute angular error between the laser-tracking data and the motion-capture data were then calculated for every time step in the laser-based tracking data.

The average positional error over all five subjects was 4.6 cm ± 2.7 cm, and the average angular error was 8.2 ± 13.8 degrees. During the 10 minutes of tracking, there were 9 brief 180-degree reversal errors. One of these lasted for 2.2 seconds, and all others were automatically corrected within 0.2 seconds. The average error with those intervals are excluded from the data was 7.4 ± 7.9 degrees.

2.6 Natural walking experiment

Although the trials in our motion capture room provided useful data for verifying the system's accuracy, it is difficult to simulate natural human walking motion in such a restricted space.

To verify that the system could also work with natural walking data, we ran several trials in an open lobby, roughly 19 meters long and 8 meters wide. Experimental subjects were instructed to walk through the area several times under a number of different conditions, *e.g.* individually, in groups, wandering aimlessly, walking purposefully, making U-turns, and stopping to ask for directions.

Raw data from a network of six laser range finders monitoring this area was recorded for each trial, which we processed offline to determine human positions.

2.6.1 Setup and Procedure

The area of interest in our experimental environment was a space within the lobby roughly 19 meters long and 8 meters wide. We used six SICK LMS-200 laser scanners, set to scan an angular area of 180 degrees at a resolution of 0.5 degrees, covering a radial distance of

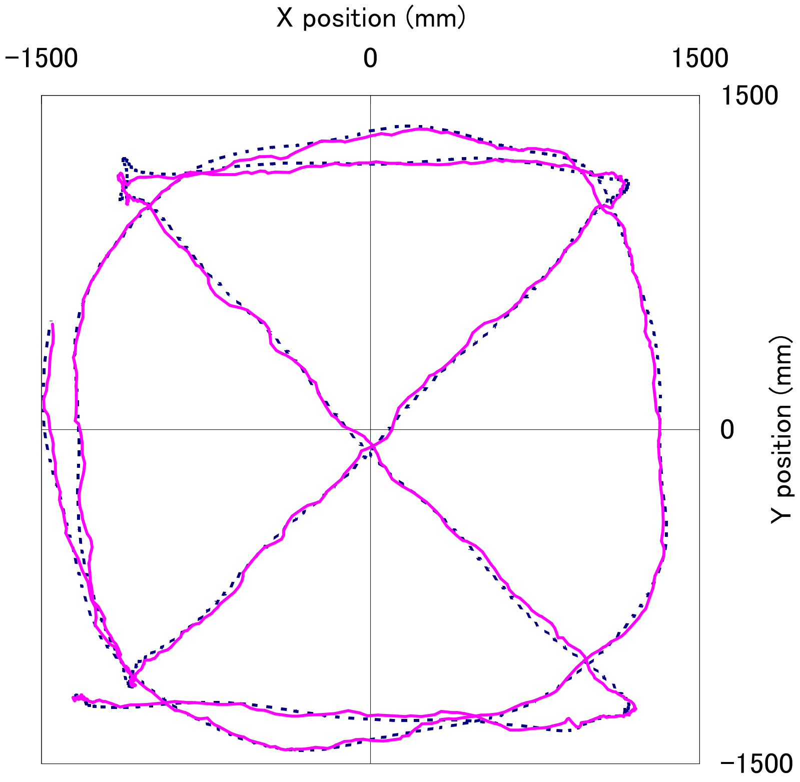


Figure 2.9: Tracking example from walking data. The dashed line represents ground-truth data from the motion capture system, and the solid line represents laser tracking data.

8 meters with a nominal system error of ± 20 mm, providing readings of 361 data points every 26 ms. These were placed around the periphery of the experimental area such that every point within the area of interest would be covered by at least two sensors, to minimize occlusions.

The sensors were mounted at a uniform height of 90cm, slightly above waist-level for most subjects. Tables, benches, and a small mobile robot were also placed within the walking area, but all of these were below 90cm and thus not visible to the laser scanners.

Twelve adults participated as subjects in this experiment, although at any given time only a subset of the group was walking within the sensor area. Six trials were conducted, and a total of 172 minutes of raw sensor data was collected.

2.6.2 Results

Two aspects of the results of this experiment will be considered here. The first is the accuracy of our method in tracking the subjects' motions, and the second is the ability to interpret this data in terms of actual body language and behavior.

Tracking Individuals

Quantifying the accuracy of this tracking technique is challenging due to the lack of more precise measurement techniques to establish a ground truth for evaluation. A side-by-side visual comparison of the raw data with the model-based estimate is perhaps the most effective indicator of the tracking accuracy.

Figure 2.10 shows raw data from five frames taken during the course of a single stride, and compares them with the model-based estimates for those time frames. Note that the swinging of the arm is clearly visible from the data, and that the model follows this movement closely.

Another indicator of the tracking accuracy of our technique is the resolution of movement that is visible over time. Figure 2.11 shows a sample path walked by one of the subjects during our experiment. The variations in θ due to the swinging of the arms and torso with each stride are quite clearly visible, with little noise present. The more subtle change in angle as the subject walks around a curving path is also quite clearly visible from the data.

These tracking results were then visually compared with video recorded during the experiment. The subjects' arm-swinging motions were observed to match with the data. The subject's torso rotations were not as exaggerated as the variations of θ in our model, which suggests the possibility that modeling the motion of the arms during walking may offer a better estimate of torso orientation.

Interestingly, our tracking results for one trial indicated an asymmetry of motion, with one arm moving much more than the other. Inspection of the video revealed that this was not a tracking error at all, but an idiosyncrasy of the subject's walking style, an observation which

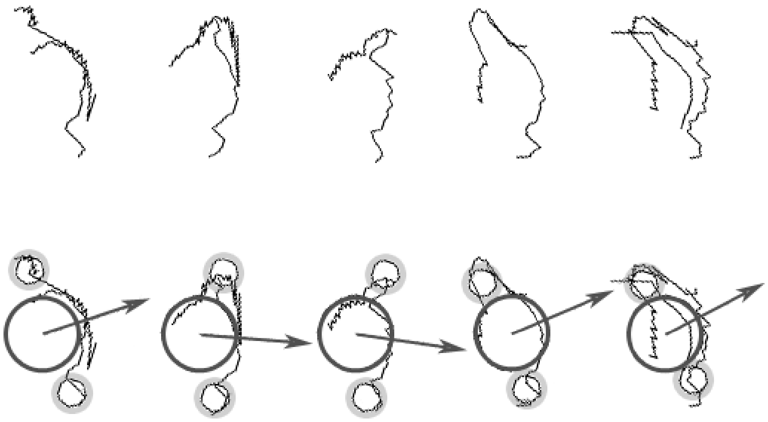


Figure 2.10: Example of arm and torso movement during a single stride. Top: Five frames of raw data from laser scanners taken at 320ms intervals. Bottom: Corresponding human shape model positions for each frame.

suggests the possibility of using the information in this model for identifying individuals or making inferences about personality or mood.

Observing Interactions

In addition to the model's tracking accuracy, it is important to consider what information can be observed regarding groups of people in social situations.

Figure 2.12 shows three scenes from our experiment. In the top scene, two subjects are seen walking together. The model correctly shows that they are walking side-by-side, facing slightly towards each other. It is possible that the relative directions in which people face while walking together might include information about their social relationship.

In the center scene, one of the two subjects is asking a third subject for directions. The model clearly shows the social situation, in which Subjects A and B are focusing their attention on subject C. Subject A is standing back at a respectful distance, which seems to imply that A and B are not part of the same group, or perhaps that their relationship is very formal.

The bottom scene illustrates the tracking of a group of subjects. Again, the group dynamic is apparent, in that all of the subjects are listening to instructions from Subject A. (Note that the model is unable to correctly determine the direction of Subject A because he is sitting and holding his arms in an unusual position.)

All three of these examples illustrate information that could not have been determined from location alone, and they suggest many possible types of social information that may be observable from this data.

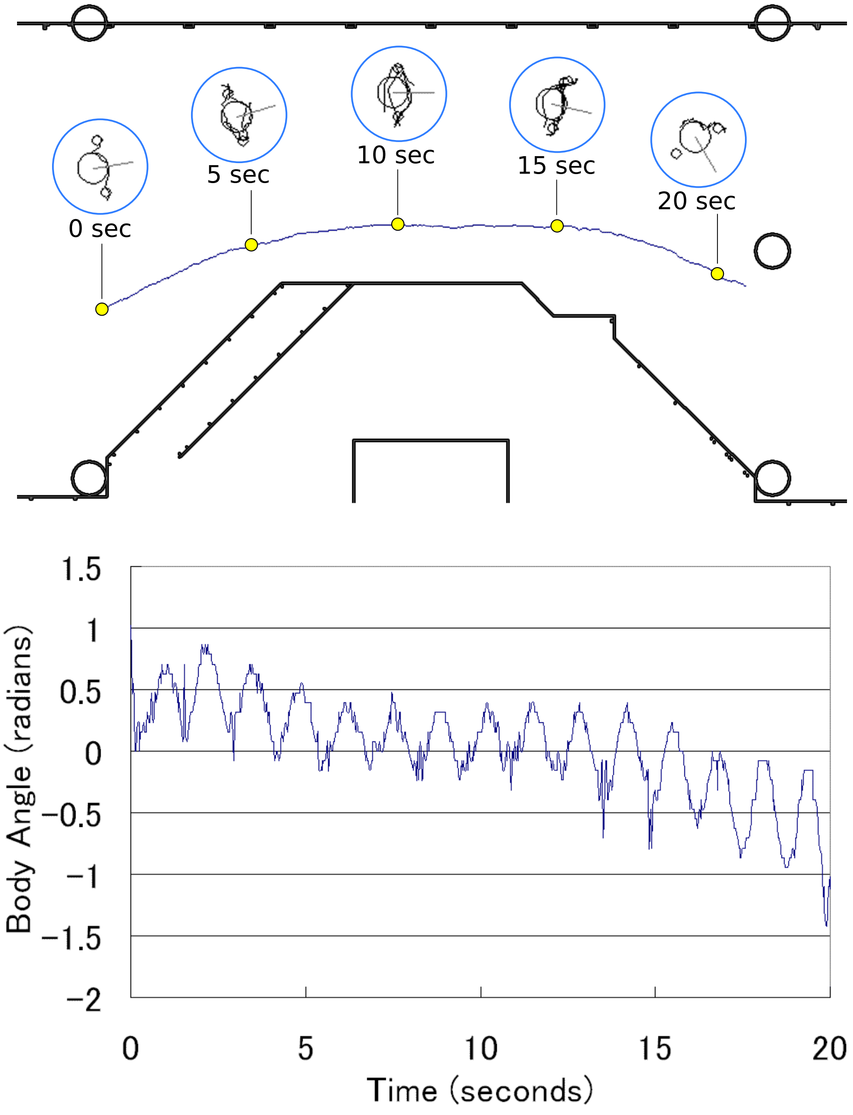


Figure 2.11: Body angle tracking during 20 seconds of walking motion. Top: An overview of the walking path in our lobby experiment shows the subject's walking path, as well as close-up views of the subject's body position at several points along the path. Bottom: Observed body angle variations (in room-centric coordinates). Periodic oscillations due to natural arm-swinging motion during each stride are clearly visible.

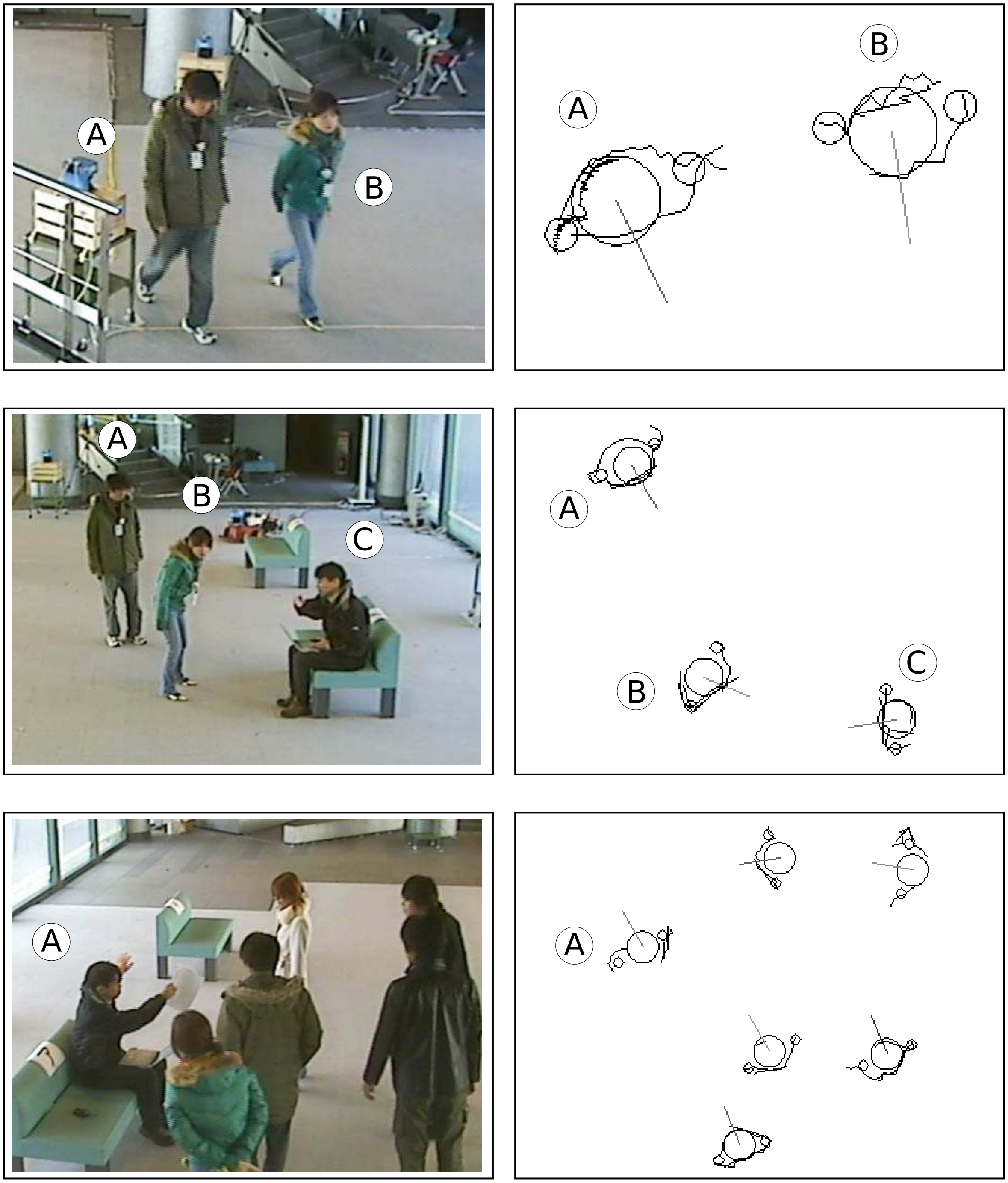


Figure 2.12: Scenes from the experiment.

2.7 Discussion

2.7.1 Performance Tuning

Many variables affect the performance of the system in terms of operating speed, position and angle accuracy, smoothness of motion, and false or missed detections. By reducing the velocity noise added during the motion model updates, for example, higher positional accuracy and smoother trajectories can be attained, but the particle filter becomes less able to follow trajectories that change abruptly.

The number of particles is another variable. If a large number of particles are used, the particle cloud's trajectory stabilizes and becomes smoother, but this comes at the cost of an increased reaction delay and increased computation time. Our algorithm uses the technique of KLD-sampling [43] to adapt the number of particles based on the density of their distribution, down to a fixed minimum limit.

2.7.2 Real-time Operation

Although the results presented in this paper were generated offline, this tracking software has primarily been developed for use with real-time data streams. Using this software in a real-time system raises the critical issue of processing speed. If the time required to process the data for one time step exceeds the sampling interval of the sensors, then data will be lost and tracking accuracy will begin to decrease.

Here we present a performance analysis using a Windows XP system with a 2.4 GHz Intel Pentium 4 processor and 1 GB of RAM. The tracking software was implemented in Java and executing using a Java 6 Virtual Machine. The tracking analysis was performed on a 4.5-minute data sample from a shopping center, during which between 3 and 18 people were tracked simultaneously. Six sensors were used in this experiment, with a frequency of 37.5 Hz, i.e. a sampling interval of 26.6 ms. A minimum of 50 particles was used for each person.

To illustrate the importance of the speed improvement gained by performing the orientation calculations separately from the particle filter, Fig. 2.13 compares our system's performance against an algorithm in which orientation calculations are integrated with the position calculations within the particle filter.

This performance comparison illustrates two key points. The first point is that even with the relatively slow Pentium 4 machine used here, it can be expected that 10-12 people can be tracked without any loss of data, *i.e.*, the tracking calculations can be completed within one data update cycle. With 18 people, incoming data would be dropped, but every second data frame would still be processed.

The second point is that, as stated in Sec. 2.3, the orientation computations are not very well-suited for integration with the particle filter. Fig. 2.13 shows that the integrated algorithm requires about four times the computation time of the two-step algorithm. In other

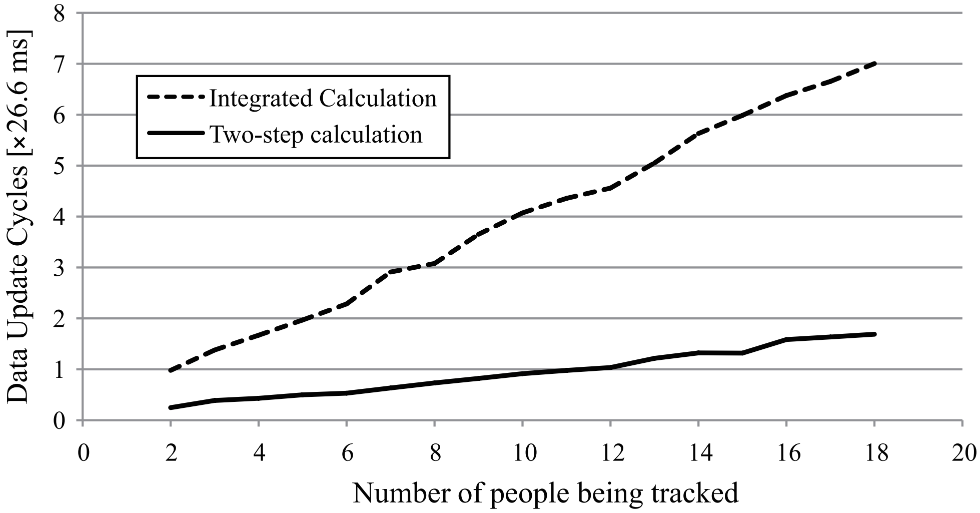


Figure 2.13: Variation of average computation time with number of humans being tracked. These results indicate that 10-12 people can be tracked before computation time exceeds the sensor sampling interval of 26.6 ms.

words, the improved efficiency of the two-step algorithm enables four times as many people to be tracked at once. To address the question of whether the choice of algorithm affects tracking accuracy, we repeated the analysis from Sec. 2.5.2 using the integrated algorithm. Results were substantially worse than with the two-step algorithm. First, many more reversals were observed with the integrated algorithm. Even correcting for the reversals, the average angular error was still 25.2 degrees, as opposed to 7.4 degrees for the two-step algorithm. This was most likely due to the issues stated in Sec 2.3, such as the non-smooth likelihood model and high sensitivity to position error.

2.7.3 Future Work

The next step in this research is to use the generated position and orientation data to improve robotic applications. Techniques should be developed for analyzing a person's trajectory through a given environment to learn about that person's intentions. Information about the directions in which people in a group are facing and their relative standing or walking positions may be helpful in identifying social rank within that group. Trajectory and orientation data might be useful in identifying people in a crowd who are interested in talking with the robot, or who have lost their way and need guidance.

Another possible area for future research is the addition of anatomically-based physical dynamics. Rather than simply modeling motion using a geometric circular model, incorporation of arm swinging and stride motion into the model could provide much more stable and accurate results. Currently, the system is able to extract a person's torso direction, which has been observed to oscillate from left to right while walking. A more detailed dynamic model could incorporate walking speed and rhythm to determine an even better estimate of the person's direction of attention.

Finally, the integration of this system with other tracking technologies, such as a leg-based laser tracking system, could provide a very robust estimate of a person's pose and enable the interpretation of more subtle expressions of body language.

2.8 Conclusions

We have developed a system in which a network of laser range finders is used for tracking the positions and orientations of people.

Comparison with results from a motion capture system verified the position accuracy to be 4.6 cm ± 2.7 cm and the orientation accuracy of to be 7.4 ± 7.9 degrees (excluding 180 degrees reversals). The system is expected to perform without performance degradation while tracking 10-12 people in real time on a Pentium 4 Windows PC.

This human tracking system has already been used extensively for providing ground-truth data and tracking humans in several experiments and field trials. The system is also actively being used as a platform for extracting useful social information from human movement for social robotics applications.

Chapter 3

Abstracting Trajectories and Anticipating Behavior

To continue the discussion of augmenting a robot's spatial perception in order to enhance navigational interactions, this chapter will present a series of techniques for abstraction of people's trajectories and a service framework for using these techniques in a social robot.

For a robot providing services to people in a public space such as a shopping mall, it is important to distinguish potential customers, such as window shoppers, from other people, such as busy commuters. The framework presented here enables a designer to make a robot proactively approach customers who exhibit some target local behavior, e.g. walking idly or stopping.

The techniques proposed in this chapter also enable information about the use of space and people's typical global behaviors to be automatically extracted from the data produced by the tracking framework presented in Chapter 2. This information enables the robot to anticipate spatial areas in which people are likely to perform the target behaviors, as well as anticipating the probable local behaviors of specific individuals a few seconds in the future.

3.1 Introduction

[86, 82] We believe that the robot can be a powerful device for bridging the gap between the digital and physical worlds. Since robots are mobile and embodied, they are well-suited for presenting digital information in the physical world. Previous studies have demonstrated that social robots can be used as museum guides [15, 159], as receptionists for assisting visitors [55], and as peer-tutors in schools [83].

On the other hand, robots have only weak sensing capabilities, which limited these robots to waiting for visitors to initiate interactions. Since we aim to realize a robot that proactively provides services in public spaces, it needs reliable observations of the positions and motion of people. However, a robot using onboard sensors can usually recognize people only within a few meters, and its sensing is not robust. To overcome these limitations, we use a "network robot system" approach [138], in which a robot is supported by a ubiquitous sensor network which observes and interprets information about people. Such an approach combines the stability and wide-area sensing capability of a ubiquitous sensor network with the intuitive presentation capabilities of the robot.

This paper describes a service framework for a network robot system, in which a mobile humanoid robot proactively approaches customers to provide information. It consists of a series of three abstraction techniques for people's trajectories: local behavior, use of space, and global behavior. We define the term **local behavior** to refer to basic human motion primitives, such as walking, running, going straight, and so on. The observation of these local behaviors can then reveal information about the **use of space**, that is, general trends in people's behavior in different areas of the environment. Finally, for more insight into the structure of people's behaviors, we look at **global behavior,** that is, overall trajectory patterns composed of several local behaviors in sequence, such as "entering through the north entrance, walking across a street, and stopping at a shop." Global behaviors are highly dependent on the specific environment.

In addition, since timing is highly critical for social interactions, we also focus on the problem of anticipating the motion and behavior of customers, to determine where the robot should move and which customers the robot should approach. For example, if a robot is designed to invite customers to a shop, it should approach people who are walking slowly and possibly window-shopping. To approach those customers, two anticipation techniques are presented: location-based anticipation and behavior-based anticipation. The detection of local behaviors and analysis of the use of space can be valuable in anticipating where behaviors are statistically likely to occur, i.e. location-based anticipation; however, an analysis of global behavior patterns is far more powerful for predicting *individual* behavior, i.e. behavior-based anticipation. As people using the space have a variety of goals, an understanding of global behavior is essential in enabling the robot to anticipate the future behaviors of individuals.

Moreover, one of the notable features of the service framework is that a designer needs only to specify a target local behavior in order to make a robot proactively approach customers. The effectiveness of the service framework is demonstrated with a field trial with two examples of applications: one is for entertainment, and another is to invite customers to a shop.

3.2 Related Work

This section provides a survey of previous studies regarding these three concepts: local behavior, use of space, and global behavior. Table 3.1 provides a summary of this survey.

Table 3.1: Related studies concerning position, place, and positional context

	Recognition				Service		
	Local behavior	The use of space	Global behavior	Anticipation	Framework	Human input required?	Domain
Reality Mining [33]	✓					-	Personal (city)
Liao et al. [98]	✓					-	Personal (city)
Subramanya et al. [165]	✓					-	Personal (city)
Suzuki et al. [167]	✓	✓				-	Public (shop)
Shao et al. [147]	✓	✓				-	Public (station)
Nurmi et al. [123]		✓				-	Personal (city)
Aipperspach et al. [1]		✓				-	Personal (home)
Activity zone [90]	✓	†				Required	Personal (home)
Museum wearable [161]		†	†			Required	Public (museum)
Pre-destination [92]			✓	✓		-	Personal (car)
Bennewitz et al. [8]				✓		-	Public (corridor)
This study	✓	✓	✓	✓	✓	Not required	Public (mall)

† : A human designer needs to assist the definition or provide information

3.2.1 Position and Local Behaviors

People's positions and trajectories have frequently been studied in robotics and computer vision (for example, [74, 148, 147]). In ubiquitous computing, positioning devices are often used, such as GPS, or the signal strength of radio (GSM, WiFi, Bluetooth, RFID, and power line) [95, 100, 127, 144].

Ubiquitous computing technology is increasingly being used to identify people's local behavior as well. For example, Eagle and Pentland developed a Bluetooth-based device attached to a mobile phone that enables the analysis of activities such as being at home, at the office, or elsewhere [33]. Liao *et al.* also used locations obtained via GPS with a relational Markov model to discriminate location-based activities such as being at home, at the office, and out dining [98]. Subramanya *et al.* included motion states (such as stop, walk, run) and velocity into a model to estimate people's low-level activity and spatial context [165].

These techniques all used wearable or mobile personal devices. Our focus is on applications in an anonymous public space, so we chose a method independent of such devices. We measure walking motion using laser range finders, sensors often used in robotics due to their precision, simplicity, and non-invasiveness. A number of techniques exist for tracking people using multiple laser range finders [148, 147, 52].

3.2.2 The Use of Space

Humans' spatial behavior has attracted scientific interest for a long time. In the 1970's and 1980's, a technique named "space syntax" was developed to analyze town-level use of space with pre-defined logic [71]. People's route choice and a form of trail were modeled as "active walker models" [67].

Such early studies required labor-intensive effort to collect data, which limit them to reveal only broad patterns; however, recent sensing technologies enable us to automatically accumulate large amount of trajectories with precise accuracy. Previous studies revealed that trajectories enable the identification of pausing points [167] and traffic paths [147, 167].

Information on the general use of space has also been retrieved. Nurmi *et al.* applied a spectral clustering method for identifying meaningful places [123]. Aipperspach *et al.* applied clustering to UWB sensor data to identify typical places in the home [1]. Koile *et al.* conducted a clustering of spaces with a focus on the relationships between velocity and positions, which enabled a partitioning of space into "activity zones." For example, places for walking, working, and resting were separated [90]. Our work involves partitioning space in a similar manner, but based on position and local behavior. In addition, we also consider how the distribution of these zones varies as a function of time.

3.2.3 Global Behavior

Models of human walking have been developed for transportation engineering and architectural design. These models are usually concerned with how environmental information affects people's behavior, such as a line of sight toward environmental structures [177] and movement of individuals in a crowd [4]. Positioning techniques could contribute to these models by providing automated, accurate position information.

In previous studies, positioning techniques have been used for categorizing people, and estimating people's goals and intentions [16]. In a museum context, Sparacino developed the "museum wearable," where people were classified into three visiting patterns. Depending upon the pattern, the system adjusted the way it presented information [161]. This is a good example of the use of global behavior; however, the places and the model of global behaviors were carefully prepared by a human designer.

In contrast, we have applied a clustering technique to identify typical visiting patterns in a museum without providing any environmental information [86]. One of the novel points of our current work is that the designer of the system provides information only about the *target local behavior*, with no knowledge about the structure of the space or of people's global behaviors. In addition to the previous work, this paper provides a method of online estimation of global behavior, which is indispensable for providing services.

The online estimation of global behaviors is difficult as, by definition, any global behavior being observed in real time is unfinished and thus not completely observable. Thus, it is necessary to estimate the true global behavior from a limited data set. Krumm *et al.* developed a technique they call "Predestination", which enables someone's driving destination to be estimated [92]. Liao *et al.* developed a technique for a person wearing GPS to infer her destination, transportation mode, and anomalous behavior [99].

While personal history of previous destinations was an important part of those studies, our anticipation technique for the shopping arcade assumes zero knowledge of a given person's individual history. Our technique is predicated on our observations of tens of thousands of people and the expectation that a new person's global behavior will be similar to those previously observed.

The concept of behavior anticipation is not without precedent in robotics. For example, Hoffman *et al.* demonstrated the value of anticipatory action in human-robot collaboration [72]. However, our use of global behaviors is a unique approach to behavior anticipation in this field.

3.2.4 Human-Robot Interaction

In the field of human-robot interaction, there have been several studies about mobile robots that provide services to people. For example, Dautenhahn *et al.* studied the appropriate behavior of a robot when it approaches a person, and found that the robot should approach people from the side but not the front [27]. Gockley *et al.* developed a natural way for a robot to follow a person [56]. Michalowski *et al.* observed how people approach a robot, and changed the robot's behavior according to their approaching style [110]. Yamaoka *et al.* established a model for a robot to appropriately position itself to effectively explain exhibits [188]. Bennewitz *et al.* developed a technique for predicting trajectories of persons for avoiding persons around it [8]. The need for this is apparently due to a lack of observation capability, which is solved in our study by having laser range finders distributed in the environment.

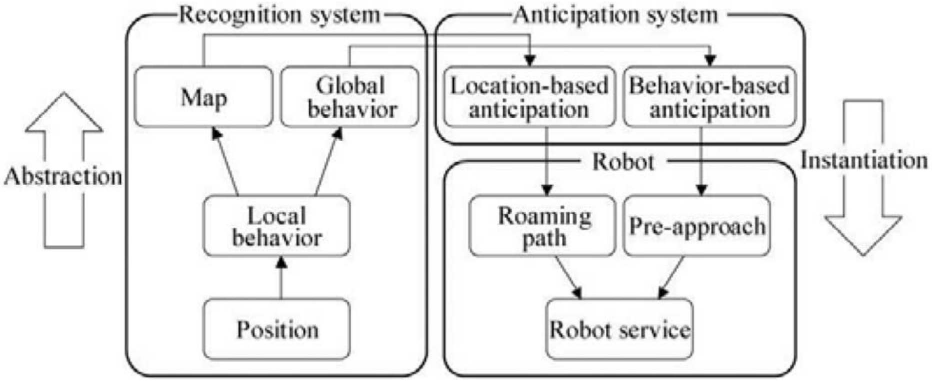


Figure 3.1: Service framework

3.3 Recognition System

Figure 3.1 shows the service framework presented in this paper. This section explains the details of the recognition system.

3.3.1 Position

We conducted our experiments in a popular entertainment and shopping arcade located by the entrance to Universal Studios Japan, a major theme park. We operated the robot within a 20 m section of the arcade, with shops selling clothing and accessories on one side and an open balcony on the other. The motion of people through this area was monitored using a ubiquitous sensor network consisting of six SICK LMS-200 laser range finders mounted around the perimeter of the trial area at a height of 85 cm (Figure 3.2).

The tracking platform presented in Chapter 2 was used to track people's trajectories through this space. The location of each person in the scan area was calculated based on the combined torso-level scan data from six laser range finders.

To illustrate the robustness of the system in our field environment, we analyzed two sets of data from one of the days in the middle of our experimental data set. For this analysis we considered only trajectories of at least 5 seconds in length, and each data set contained 100

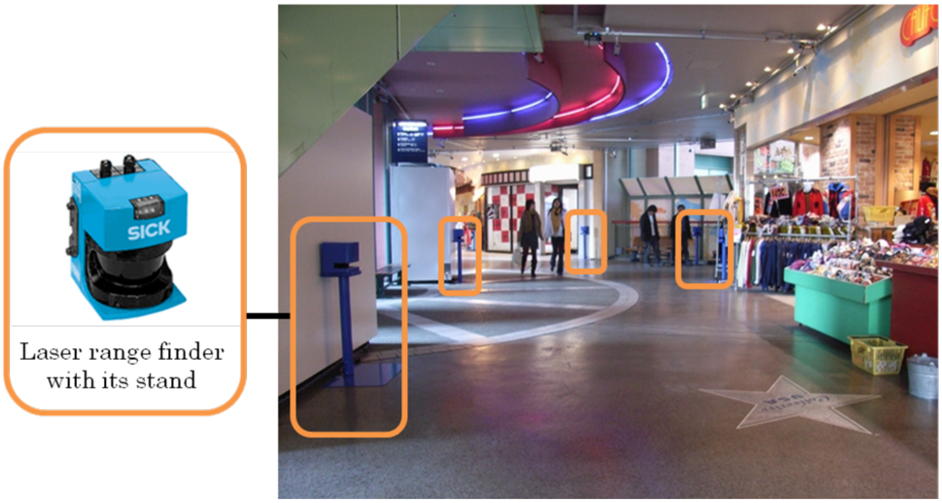


Figure 3.2: Shopping arcade and laser range finders

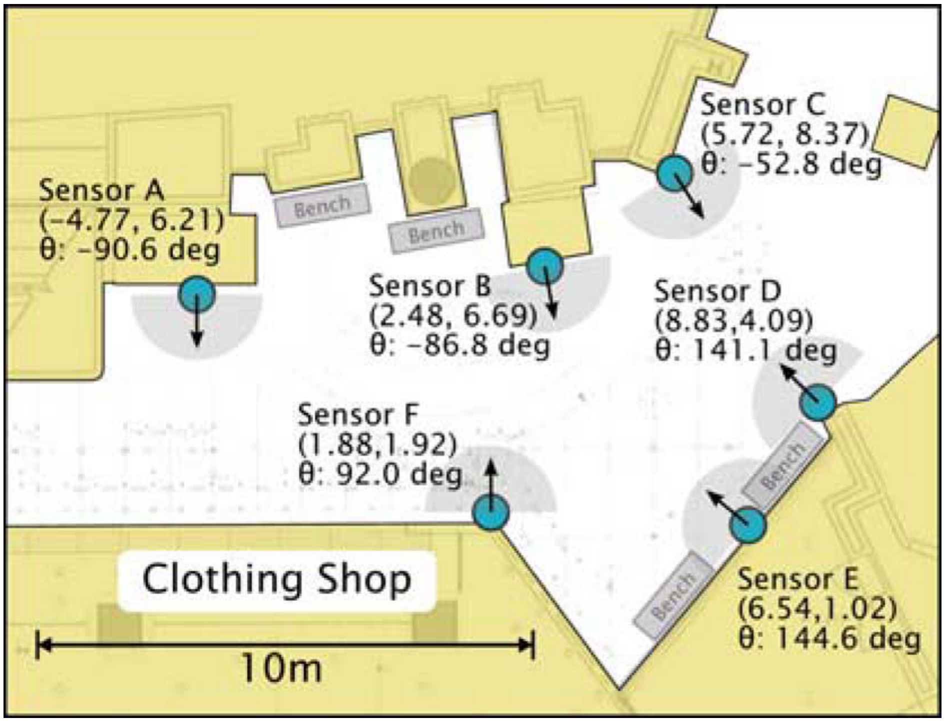


Figure 3.3: Placement of laser range finders

trajectories. The first set was taken starting at 11:30am, a time when very few people were passing through the area, and the other was taken at 5pm, when the area was more crowded. The morning data set lasted 42 minutes, with an average trajectory length of 17.1 seconds, and the evening data set lasted 12 minutes, with an average trajectory length of 18.0 seconds.

For each of these data sets, the entry and exit times of each person passing through the space were identified manually by inspection of the raw laser scan data (this enabled more exact estimation of people's positions and entry times than inspection of video data). Any tracking errors during this period were also recorded, *e.g.* if a person entered the space and was not tracked, if two people were mistakenly switched with each other in mid-trajectory, or if a trajectory was lost in the middle of the space and reacquired with a different ID.

In fact, our system successfully tracked 100% of the people passing through the space in both cases. No tracking errors occurred, and no people entered the space without being tracked. However, the system did have some difficulty distinguishing couples walking close together. Couples were sometimes initially misinterpreted as a single person, but after a few seconds, the system always correctly identified them as two people. Since this phenomenon results in a short time lag before the system begins tracking the second person, we calculated the system's tracking success rate as the ratio between the total amount of time people were successfully tracked to the total amount of time people were present in the area. This ratio is presented in Figure 3.4 for the two 100-trajectory datasets.

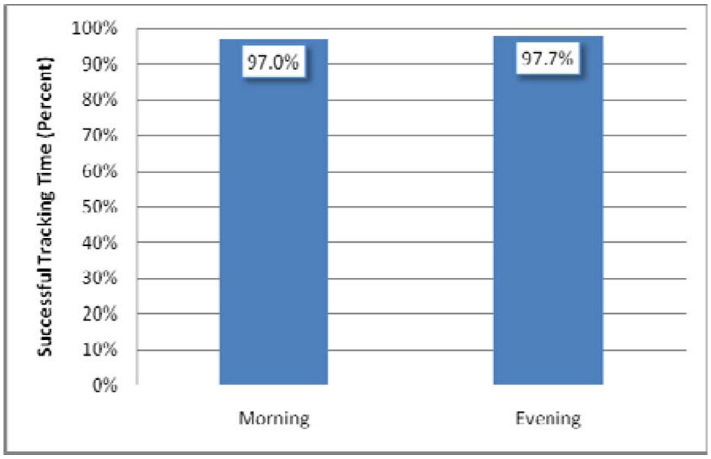


Figure 3.4: Tracking success rate for two datasets

Based on this analysis, we consider our tracking system to be highly robust, particularly in terms of maintaining continuity of trajectories from beginning to end, an important requirement for the analysis we present in this paper.

3.3.2 Local Behavior

As defined earlier, “local behaviors” represent basic human motion primitives. We began our analysis with a classification system which uses SVM (support vector machine) to categorize trajectories based on their velocity, direction, and shape features.

Specifically, the following features were used for the SVM to classify the local behaviors:

(i) The end point of the normalized trajectory

Normalization refers to a rotation of the trajectory to fit its starting point to the origin and its longest direction to the x axis (Figure 3.5 (a) and (b)). Then, three points were sampled from the normalized trajectory: at times N/3, 2N/3, and N seconds, where N represents the duration of the trajectory. At each point, four dimensions of features were retrieved: x-coordinate, y-coordinate, arc tangent of this x-y position, and the distance of this x-y position from origin. Overall, 12 dimensions of features were retrieved.

(ii) The size of rectangle that covers the normalized trajectory

We retrieved the max value, min, and average value of x-coordinate and y-coordinate among all of the points sampled per 100ms in the N seconds of the trajectory. Overall, 6 dimensions of features were retrieved.

(iii) The angles of the trajectory

As shown in the Figure 3.5 (c), we calculated a sub-angle in a trajectory. For this calculation, the trajectory was separated into three sub-trajectories, at time “0 to N/3”, “N/3 to 2N/3”, and “2N/3 to N” seconds. For each sub-trajectory, the angle between start and end point was calculated. In addition, we also calculate the maximum angle as well as deviation of the angles among each sub-trajectory, within a sliding 500ms-window for each 100 ms from the start to end of the sub-trajectory. Overall, 9 dimensions of features were retrieved.

(iv) The velocity

For each 100ms interval, an immediate “sub-velocity” was calculated. The average, min, max, and variance of the sub-velocities were used as features. In addition, travel efficiency was computed by calculating the overall velocity from the start point to the end point, and dividing this by the sum of all sub-velocities. (It is nearly 1.0 if the trajectory moves straight, and nearly 0.0 if it only oscillated at the same point). Overall, 5 dimensions of features were retrieved.

In total there were 32 features. All of the features are float values and scaled within the range of 0 to 1. The SVM for the *Style* category uses all of the 32 features of (i) to (iv), while the SVM for the *Speed* category uses the features of (i), (ii), and (iv), the SVM for the *Short-term style* category uses the features of (i), (ii), and (iii), and the SVM for *Short-term speed* category use the features of (i) and (iv). Our SVM was implemented using LIBSVM [17]. The one-against- one method was used for multiclass classification [73]. For all SVM’s, an RBF Kernel (Gaussian Radial Basis Function Kernel) was used.

To include a wide variety of movement types, we initially defined the following four categories. Each category has about 200 samples for learning, consisting of 2- or 5-second trajectory segments. We selected typical trajectory segments that fit with the concept of each

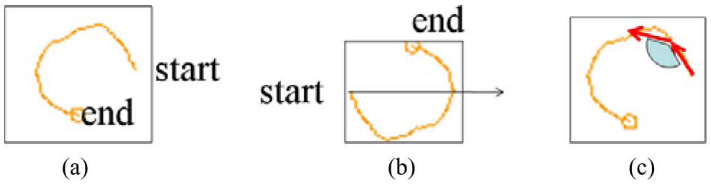


Figure 3.5: Feature vector for calculatingmotion primitives. (a)Original trajectory. (b) Normalized trajectory. (c) Subangle.

class of the categories, labeled them by hand, and put them into the training data set. We did not include trajectories which were ambiguous between classes.

(a) *Style*

This category consists of the following six classes. It requires 5.1 seconds of trajectory data for classification.

- straight (Figure 3.5 (a))
- left turn (Figure 3.5 (b))
- right turn (Figure 3.5 (c))
- wandering (Figure 3.5 (d))
- U-turn (Figure 3.5 (e))
- not walking (Figure 3.5 (f))

We labeled 226 trajectories and tested the system with the leaving-one-out method, a cross-validation-method where each of the data elements is tested by using the remaining elements for training; i.e. we created 226 subsets, each of which has one unique trajectory for testing and the remaining 225 for training, and averaged the classification accuracy of the 226 subsets. It classified with 84.5% accuracy on average. The confusion matrix shows relatively-frequent confusion between U-turn and wandering, recognizing U-turn with 79.4% accuracy and wandering with 76.6% accuracy.

(b) *Speed*

This category consists of the following five classes. It requires 4.9 seconds of trajectory data for classification. - running

- fast-walk (Figure 3.6 (a))
- idle-walk (Figure 3.6 (b))
- stop : short stop is observed in a trajectory while some movements is also observed
- wait : only stopping, but no motion observed

In the the labeling, we judged the difference between ”idle walk“ and ”fast walk“ based on the speed of the trajectory. The difference between “stop” and “wait” is defined by whether the trajectory remains stopped for the full duration or not.

We labeled 166 trajectories and tested the system with the leaving-one-out method; it classified with 92.8 % accuracy on average. The confusion matrix shows frequent confusion between stop and wait, recognizing stop with 66.7 % accuracy.

(c) *Short-term style*

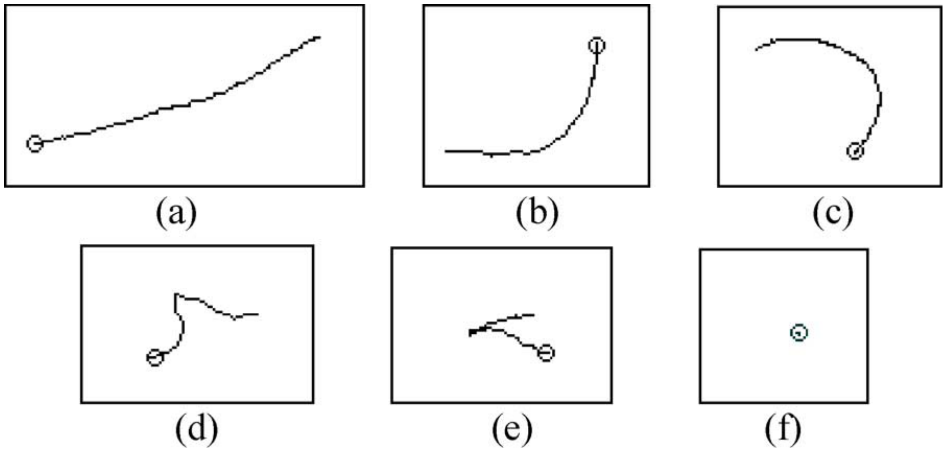


Figure 3.6: Examples of style category. (a) Straight. (b) Left-turn. (c) Right-turn. (d) Wandering. (e) U-turn. (f) Stop.

This category is similar to (a) *Style*, but to enable faster recognition we reduced the duration required for the classification. It requires 2.1 seconds of trajectory data for classification.

- straight
- left turn
- right turn
- U-turn
- not walking

We labeled 150 trajectories and tested the system with the leaving-one-out method; it classified with 93.3 % accuracy on average. There was no particular confusion in the confusion matrix.

(d) *Short-term speed*

This category is similar to (b) *Speed*, but to enable faster recognition we reduced the duration required for the classification. It requires 2.2 seconds of trajectory data for classification.

- running
- fast-walk
- idle-walk
- stop

We labeled 159 trajectories and tested the system with the leaving-one-out method; it classified with 95.6 % accuracy on average. There was no particular confusion in the confusion matrix.

Note that each category requires different length of trajectories, which is the result of our minimization of the time to recognize each of these categories. For example, (c) *Short-term style* requires 2.1 seconds while (a) *Style* requires 5.1 seconds, since *Style* has a category of wandering which is confused with U-turn more if the duration is smaller than 5.1 seconds.

Short-term style does not have the category of wandering, which make the system easily categorize even with a shorter duration of time.

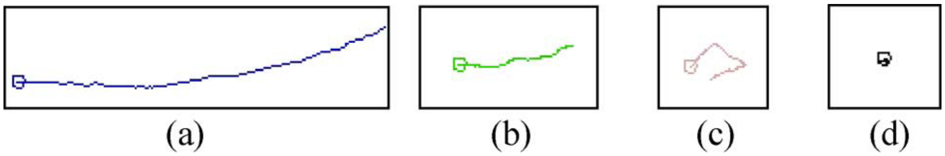


Figure 3.7: Example trajectories for local behaviors

In the subsequent analysis, we merged several local behavior classes for simplicity. Within *style*, “left turn”, “right turn”, and “U-turn” were merged into “wandering”. Within “Style”, the classes *left-turn*, *right-turn*, and *U-turn* were all merged into the *wandering* category. Within “Speed”, we merged *stop* and *wait* into the *stop* category. We also merged classes for short-duration and 5-second behavior. Thus, we reduced the set to the following four local behaviors: *fast-walk*, *idle-walk*, *wandering*, and *stop*. Figure 3.7 shows examples of these local behaviors. We define the position P_t^n of visitor n at time t to include the x-y coordinates (x, y) as well as Boolean variables [1] indicating the presence or absence of local behavioral primitives.

Each trajectory has a sequence of local behaviors represented by these Boolean variables at each time step t. The system split a segment of trajectory from the time step t to the past for the required length of each classifier, and sent it to the classifier. They remain undetermined if t is smaller than the minimum required time of SVMs, i.e. 2.1 seconds.

3.4 Analysis of Accumulated Trajectories

Based on the position and local behavior data thus obtained, an analysis was performed to obtain a higher-level understanding of the use of space and people’s global behaviors. This analysis constitutes the foundation for the robot’s ability to anticipate people’s local behaviors.

3.4.1 Data Collection

Human motion data was collected for a week in the shopping-arcade environment, from 11am-7pm each day, including 5 weekdays and 2 weekend days. We chose this time schedule because the shops open at 11am, and the number of visitors drops after 7pm, after the theme park closes in the evening.

[1] These Boolean variables allow each stateto have a combination offast-walk,idle,wander and stop.One4-state variable might be appropriate depending on the purpose.For this study,our intention was to provide a local behavior classifier as capable as possible.

In this environment, the major flow consisted of customers crossing the space from the left to the upper right or vice versa, generally taking about 20 seconds to go through. We removed trajectories shorter than 10 seconds, in order to avoid noise from false detections in the position tracking system. In all, we gathered 21,817 visitor trajectories.[2]

3.4.2 Use of Space (Map)

The first analysis task was to identify how the space was used, and how the use of space changed over time. We applied the ISODATA clustering method [7] to achieve this. First, we partitioned the time into one-hour segments categorized as weekday or weekend. We then partitioned the space into a 25cm grid, mapping the environment into 2360 grid elements.

The local behaviors represented by the Boolean variables are all mapped into the histogram prepared for each grid elements. Each grid element containing histogram data of local behaviors:

denotes the number of occurrences of local behavior *x* at time slice *t* within grid element *i,* which is normalized for each local behavior *x*. Specifically, we normalized each histogram $H_x(i,t)$ to have a mean value of 0.0 and a standard deviation of 1.0.

To make the data set more manageable, we first combined time slices based on their similarity. The difference between time slices t_1 and t_2 is defined as:

$$\sum_i \sum_x |(H_x(i,t_1) - H_x(i,t_2)| \tag{3.4.1}$$

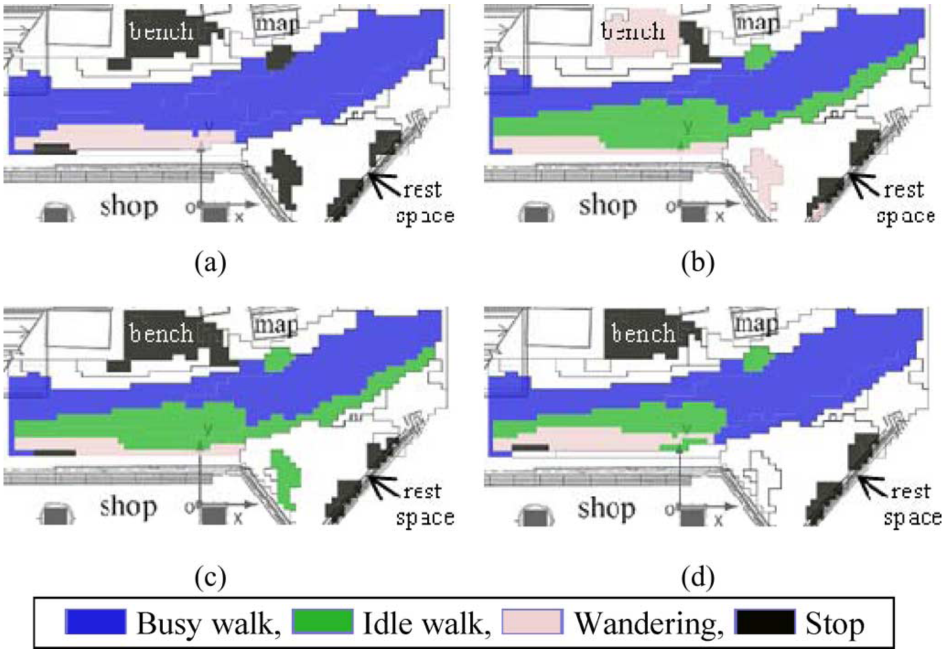
We then combined spatial grid cells where the distance was smallest and the grid was spatially connected. The distance between grid cells *i* and *j* is defined as:

$$\sum_t \sum_x |(H_x(i,t) - H_x(j,t)| \tag{3.4.2}$$

As is usual for this type of explorative clustering, we arbitrarily set the number of partitions to help us intuitively understand the phenomena occurring in the environment. We chose to use 40 spatial partitions and 4 temporal partitions. Figure 3.8 shows a visualized output of the analysis. The partitions are color-coded according to the dominant local behavioral primitive in each area. Blue (medium gray on monochrome printouts) represents the areas where the *fast-walk* behavior occurred more frequently than any other local behaviors. Thus, people tend to pass directly through this area, which can be thought of as "corridor" space.

The areas where the *idle-walk* primitive occurred most frequently are colored with green (or light gray).

[22]In this study,we obtained approval fromshopping mall administratorsfor this recordingunder the condition thatthe information collectedwould be carefully managed and only used for researchpurposes.The experimental protocol wasreviewed andapproved byour institutional review board.



(a)

(b)

(c)

(d)

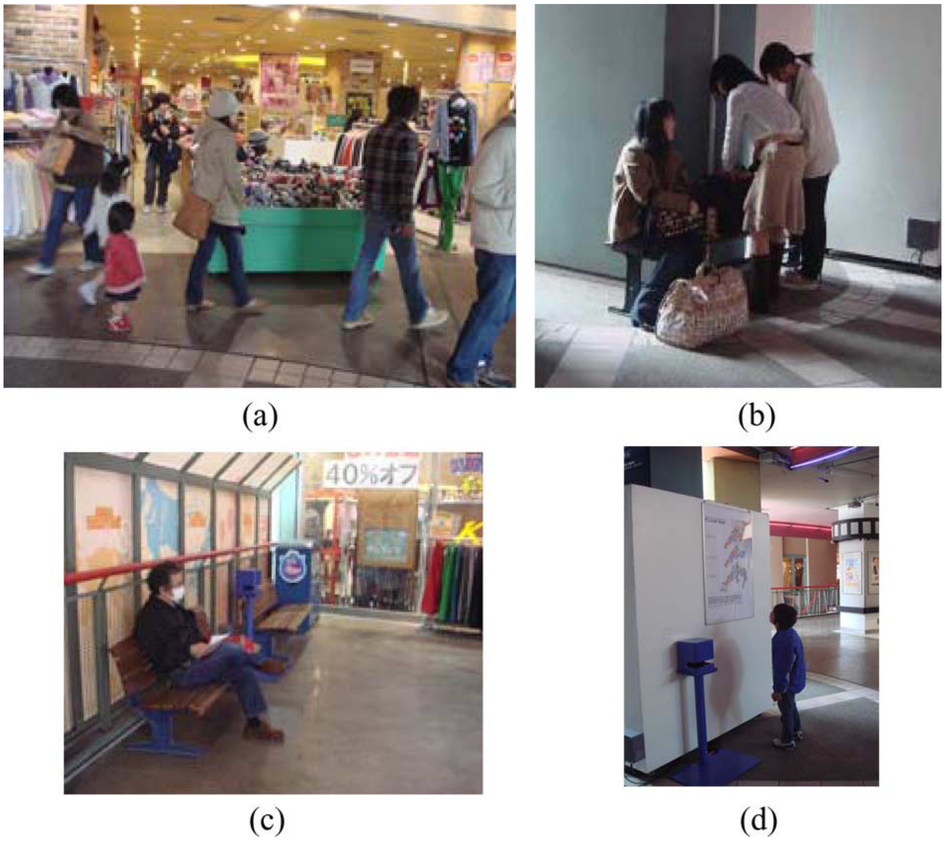
Figure 3.8: Analysis of the use of space. (a)Weekday 11am-5pm, weekend 12-1pm (b)Weekday 5-6pm (c)Weekday 6-7pm (d)Weekend 11am-12pm, 1-7pm

In some areas, the use of space was very clearly observed to change as a function of time. The lower left area is in front of a shop. When the shopping arcade was busy in the evening, as in Figure 3.8 (b), with people coming back from the theme park, many people were observed to slow down in front of the shop, and the "corridor" space changed into "in front of shop" space with *idle-walk* becoming dominant (photo: Figure 3.9 (a)); however, when there were not so many people, such as midday during the week as in Figure 3.8 (a), these areas disappeared and became similar to other "corridor" space. The lower right side of the map represents the side of the corridor, where people tend to walk slowly when the arcade is busy (Figure 3.8 (b) and (c)); these areas also disappeared and became similar to other "corridor" space (Figure 3.8 (a) and (d)).

The areas where the *stop* primitive was most frequent are colored with dark brown (or dark gray). In Figure 3.8, these areas can mainly be found in the upper center (photo: Figure 3.9 (b)) and the bottom right (photo: Figure 3.9 (c)). These areas contain benches, and can be considered "rest space".

In the upper center area, below the word 'map', there is a small space where *stop* is the dominant primitive in Figure 3.8 (a) whereas *idle-walk* is dominant in (b) through (d). A map of the shopping arcade is placed on that wall. Customers sometimes slowed down, stopped, and looked at this map (Figure 3.9 (d)). The statistical analysis clearly revealed this phenomenon as defining a distinct behavioral space.

The areas where the *wandering* primitive was dominant are colored with pink (or very



(a) (b)

(c) (d)

Figure 3.9: Examples of the actual use of the space. (a) Idle-walk in front of a shop. (b) Stop at a bench. (c) Stop at rest space. (d) Map.

light gray). All maps in Figure 3.8 show the space immediately in front of the shop as having this property. The areas where none of the primitives were dominant, such as the bottom-right space, are colored white. These areas were not used so much.

To summarize, we have demonstrated that through this analysis technique, we can separate space into semantically meaningful areas such as the corridor, the space in front of the shop, the area in front of the map, and the rest space. It also reveals how usage patterns change over time, such as the change of dynamics in the space in front of the shop.

3.4.3 Global Behavior

Based on the accumulated trajectories, we analyzed how people visited the shopping mall. In this section we introduce a method of extracting typical global behaviors.

Preparation: State chain models

We analyzed trajectories based on the *state chain* model illustrated in Figure 3.10. That is, we converted , represented in x-y coordinates, to a sequence of states, based on spatial partitioning. is defined as, where A_n is the partition the point in trajectory p belongs to. In the example in Figure 3.10, the trajectory starting from partition 1, stayed in partition 1 for 3 time steps, then entered briefly into partition 2, and moved back to the partition 1 . . . , which is represented as the sequence of states 1, 1, 1, 2, 1, . . .

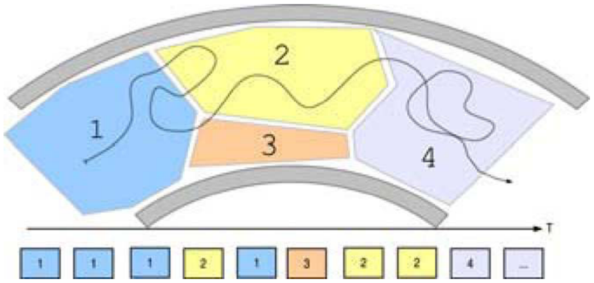


Figure 3.10: State chain model.

Distance between trajectories

We calculate the distance between two state chains, and , by using a DP matching method (widely used in many research domains, e.g. [136]), which is identical to the comparison of strings known as the Levenshtein distance. Figure 3.11 illustrates this trajectory comparison technique. Here, we set the distance between partitions as the distance between the centers of the partitions. The cost for “insert” and “delete” operations is calculated as this partition distance plus a constant parameter, which represents the tradeoff cost between time and space.

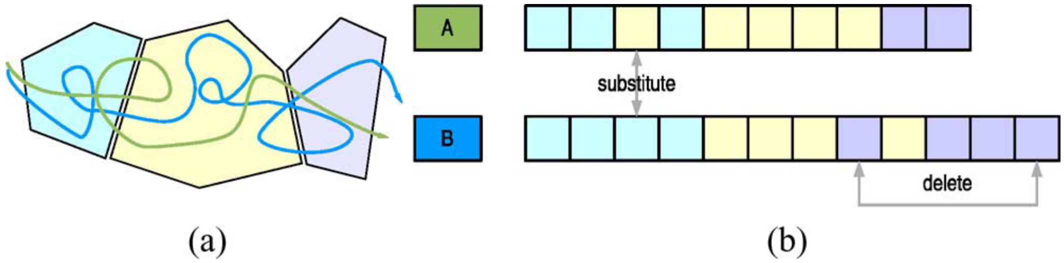


Figure 3.11: Comparison of trajectories based on DP matching. (a) Two trajectories. (b) Comparison of state chains of trajectories.

For the DP matching, we again partitioned the space into a 25cm grid (2360 grid elements), to easily compare trajectories. The DP matching method was chosen for its simplicity and the fact that it does not require particular tuning of parameters. Since global behaviors naturally emerge through the interactions between people and their environment, we believe that it is best to minimize the number of parameters that need to be adjusted manually, keeping the process simple and generalizable.

The trajectories are segmented into 500 ms time steps, and they are compared with each other based on the physical distance between them at each time step. To this is added a cost function, based on "insert" and "delete" operation costs in the DP matching, where we defined the cost of a single insertion or deletion to be 1.0 m.

In addition, this state-chain representation reduces calculation cost. For example, we compared calculation cost based on raw trajectory P^i and state chain S^i for retrieving global behavior with a k-means clustering method from 28 trajectories. The state-chain method costs 0.53 sec while the raw-trajectory-based method costs 9.56 sec. Thus, using the state chain is eighteen times faster. We cached the calculation of distance between partitions in the state-chain-based method (that is, *insert*, *delete*, and *substitute* costs in DP matching), which also greatly improved the calculation speed.

Clustering and Visualization

We classified trajectories with a k-means method to identify typical visiting patterns. The distance between trajectories was provided from DP matching method mentioned above. We separated the space into 50 similarly-sized partitions by the k-means method [86] for this visualization, although the actual computation used 2360 partitions. We did not use these 2360 partitions or the result of analysis of the space shown in Figure 3.8 for the purpose of this visualization, since we are interested only in the transition pattern. K-means clustering of the space is one method which can provide similarly-sized polygonal spatial divisions distributed over the map with an arbitrary resolution, which are useful features for the visualization of global behavior.

Figure 3.11 shows a visualization of the global behaviors at k=6. In this visualization, each area is colored according to its dominant local behavior primitive, and transitions between adjacent areas are shown as arrows. For example, blue represents *fast-walk*, and green represents *idle-walk*. Solid colors indicate a frequency of occurrence of at least one standard deviation above average, and lighter tints represent weaker dominance, down to white if the frequency is at least one standard deviation below average.

The transitions between adjacent areas are computed for each pair of adjacent areas by counting the transitions in the state chains of the trajectories that belong to each global behavior. Frequent transitions between adjacent areas are shown by arrows. An arrow is drawn from partition i to j when $(N_{ij} - N_{ji})$ is larger than a threshold (here, set as 0.1) where N_{ij} indicates a transition from i to j.

Of course, we can analyze behavior patterns at any k value; a larger number k will result in more detailed separation of visiting patterns.

We can interpret about six typical global behaviors from Figure 3.12:

(a) Pass through from right to left (7768 people)

This pattern represents one of the major flows of people, who are coming back from the theme park (on the right) on their way to the train station (on the left). In this pattern, most of the areas are colored blue because the most frequent primitive in those areas was *fast-walk*. In front of the shop, there are some areas colored green, which represent spaces where people slow down to look at the shop.

(b) Come from the right, and stop at the shop (6104 people)

This pattern is similar to the pattern (a); but people either stop at the shop or go through the shop to go to the left area, as trajectories mostly disappeared at the shop.

(c) Pass through from left to right (7123 people)

This is also a major pattern, where people are coming from the train station and going in the direction of the theme park. In contrast to the patterns in (a) and (b), people rarely stopped or slowed down in front of the shop.

(d) Rest at the rest space (213 people)

In this pattern, people mostly spent time in the bottom right rest space (Figure 7 (c)) where benches were placed.

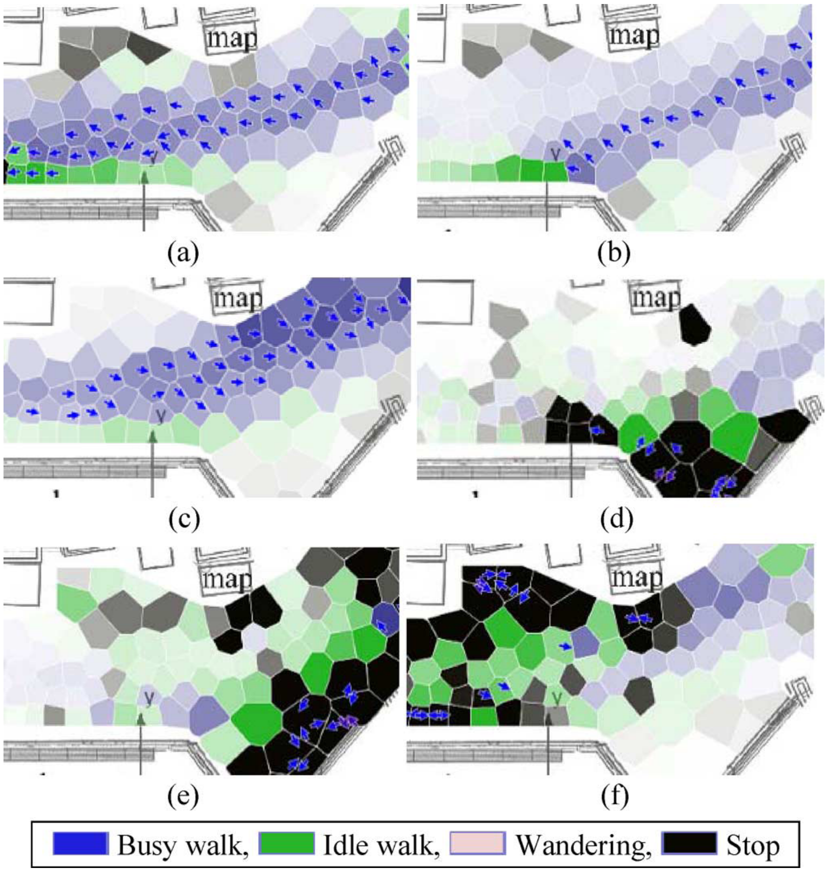
(e) Around the rest space and right (275 people)

Similar to the pattern in (d), but people moved around the right area more, and not around the shop area. Some people also stopped in front of the map or the upper rest area.

(f) Around the shop and bench (334 people)

People mainly came from the left side, walking slowly, and stopped in front of the shop as well as in front of the map.

In summary, this analysis technique has enabled us to extract typical global behavior patterns. These results show that most people simply pass through this space while a smaller number of people stop around the rest space or the map area. People tend to stop at the shop more often when they come from the right, a result which makes intuitive sense, as the shopping arcade is designed mainly to attract people coming back from the theme park.



(a) (b) (c) (d) (e) (f)

Figure 3.12: Six typical patterns of global behavior. (a) From right to left. (b) From right and stop at the shop. (c) From left to right. (d) Rest at the rest space. (e) Around the rest space and right. (f) Around the shop and bench.

3.5 Anticipation System

Robots differ from other computing systems in that they are mobile, and it takes some time for a robot to reach a person in need of its service. Thus, the ability to anticipate people's actions is important, as it enables the robot to proactively pre-position itself so it can provide service in a timely manner.

We assume here that the robot's service is targeted towards people who are performing some particular local behavior, such as *stop* or *idle-walk*. The robot system uses the results of the analysis about the use of space and global behavioral primitives to anticipate the occurrence of this "target behavior". At the same time, the robot system tries to avoid people who are performing particular local behaviors, such as *fast-walk*, which we refer to as "non-target behavior". To anticipate local behaviors, we use two mechanisms: location-based anticipation and behavior-based anticipation.

3.5.1 Location-Based Anticipation

As shown in Figure 3.8, the system has use-of-space information about the frequency of the local behaviors associated with spatial and temporal partitions. The robot uses this information to estimate the locations in which people will be statistically likely to perform the target behavior. In addition, we assume that a moving robot would attract people's attention more than a robot standing still, which makes it easier for the robot to initiate interaction; thus, the system provides a path for the robot to roam around such locations, rather than choosing a single point at which to wait.

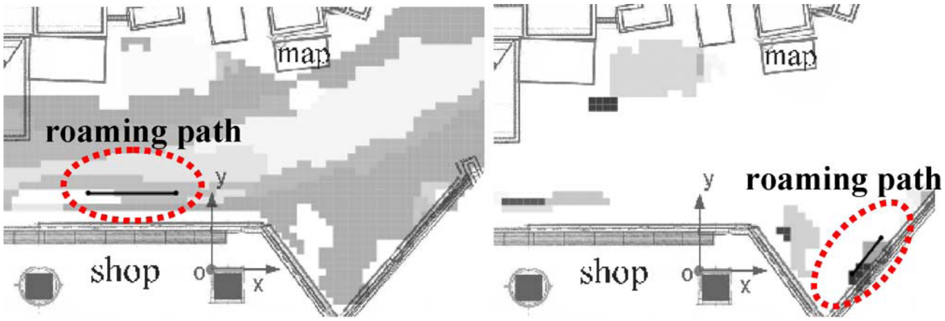
Figure 3.13 shows an example anticipation map. The darker areas represent areas where the system anticipates both a high likelihood of the target behavior and a low likelihood of the non-target behavior. In the graph, areas where the likelihood of the non-target behavior is higher than the likelihood of the target behavior are shown in white.

The robot roams through this high-likelihood area looking for people. At each time slice t, the system updates the roaming path, $\vec{P_x}$, to maximize the roaming value calculated from candidates of all possible straight-line paths from 1m to 5m in length on the 25cm-grid, using the following equation.

$$roaming\_value(\vec{P_x},t) = \sum_{i \in \vec{P_x}} (H_{target}(i,t) - H_{non-target}(i,t)) \tag{3.5.1}$$

where represents the histogram of the target behavior at the point grid i at time slice t (see IV B for the calculation to retrieve the histogram).

After finding the best path, the system modifies it according to safety considerations; the robot is constrained to operate within a safety buffer of two grid elements from the outside of observed area (these areas are too close to a wall for the robot to pass through), so the points of the path are translated to the nearest points within the safe area. The black line in Figure 12 represents its automatically-generated roaming path.



(a) (b)

Figure 3.13: Example of anticipation map. (a) Weekday 11am-5pm, idle-walk. (b) Weekday 11am-5pm, stop.

In one scenario, the robot's task might be to invite people to visit a particular shop. In this case, selecting *idle-walk* as the target behavior and *fast-walk* as the non-target behavior might be appropriate, since the robot wants to attract people who have time and would be likely to visit the store. Figure 3.13 (a) is the anticipation map for this scenario, calculated for the behavior patterns observed on weekdays between 11am and 5pm. Several areas away from the center of the corridor are colored, and the roaming path is set in front of the shop. Note that the best path in this case is slightly below the line shown in the figure, but this area is very close to the boundary of the observed map. The robot's final path was translated about 50cm away from the edge for safety reasons.

In a different scenario, the robot's task might be to entertain idle visitors who are taking a break or waiting for friends. Particularly because this shopping arcade was situated near a theme park, this is quite a reasonable expectation. In this case, it would be more appropriate to select *stop* as the target behavior and *fast-walk* as the non-target behavior. Figure 3.13 (b) is the anticipation map for this second scenario. In this case, only a few areas are colored. The roaming path is set to the bottom-right area.

Note that since the roaming path was automatically calculated based on the anticipation map, no additional knowledge about the space was provided by designers.

3.5.2 Behavior-Based Anticipation

The second technique used for anticipating local behaviors is to estimate the global behaviors of people currently being observed, and then to use that information to predict their expected local behaviors a few seconds in the future.

To ensure prediction accuracy, we used a large number of clusters for the global behavior analysis. We clustered the human motion data collected earlier into 300 global behavior patterns. For this analysis, since we are interested in behaviors several seconds in the future, we only used trajectories observed for a sufficient amount of time. We filtered out trajectories less than 20 seconds long, leaving 11,063 trajectories for analysis. Next, to predict the global behavior of a new trajectory which has been observed for *T* seconds, the system compares the new trajectory with the first *T* seconds of the center trajectory of each of the 300 clusters, using the same DP matching technique applied earlier for deriving the global behaviors. The cluster with the minimum distance from the new trajectory is considered to be the best-fit global behavior for that trajectory.

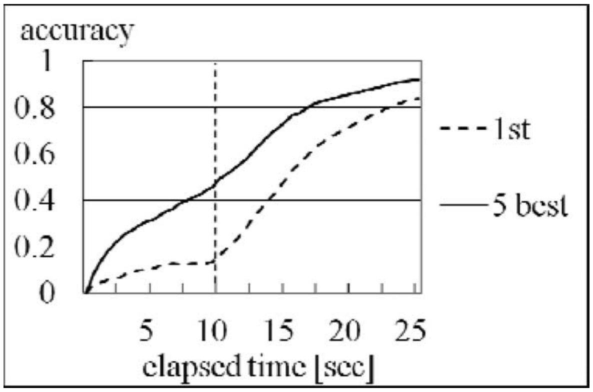


Figure 3.14: Accuracy of the prediction of global behavior

Figure 3.14 shows the prediction accuracy for observed trajectories from 0 to 25 seconds in length. Here, we used 6 of the 7 days of data to create the prediction model, and tested its ability to predict the remaining one day of the accumulated data. The prediction is counted to be successful if the predicted global behavior matches with the one the trajectory belongs to, i.e. the classification result after observing the whole length of the trajectory. The accuracy accounts for only trajectories of total length greater than 20 seconds, as we filtered out shorter trajectories for calculating global behaviors. The result labeled “1st” represents the case where the best-fit global behavior at time *T* was the correct one (the cluster the trajectory finally fit with at completion). The result labeled “5 best” is the result if we define success to mean that correct global behavior falls within the top 5 results. Performance levels off after 20 seconds. Since there are 300 global behaviors, we believe that a success rate after 10 seconds of 45% and after 15 seconds of 71% for “5 best” represents fairly good performance.

After the most likely global behaviors are selected, the person’s future position and local behavior are predicted based on an “expectation map.“ An expectation map is a data structure prepared *a priori* for each global behavior. For each 500-ms time step along the trajectories, a 25-cm grid representation of the observed space is added to the map. Each element of this grid contains likelihood values for each of the four local behaviors to occur in that location at any time *after* that time step. These likelihood values are empirically derived from the original observed trajectories falling within the chosen global behavior cluster, and they represent the average frequency of the occurrence of each local behavior after that time step. We used the 5-Best result to create an expectation map for the person by combining expectation maps from each of the 5-Best global behaviors.

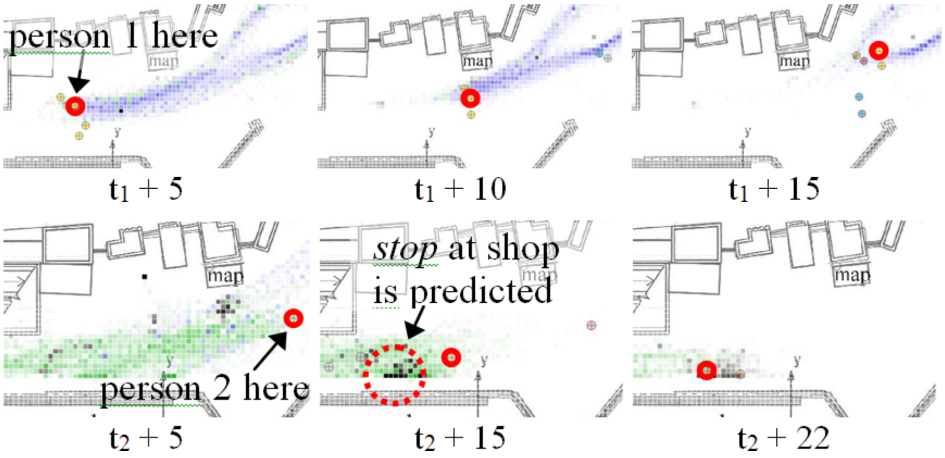


Figure 3.15: Example of prediction of future behaviors.

Figure 3.15 shows expectation maps for various time increments. The solid circles represent the positions of people walking through the space, with the person of interest outlined in red. The expectation map for that person's estimated global behavior is shown, where the area colored blue represents the area where *fast-walk* is expected, and the green area represents the area where *idle-walk* is expected. The three figures in the top row show the trajectory for person 1, who was first observed at time t_1. The first figure shows time t_1 + 5 sec, where the expected local behaviors can be seen tracing a path through the corridor, heading toward the upper right. In fact, this course was correctly predicted, and the person followed that general path. The second line is the trajectory for person 2, first observed at time t_2. Here, since the person walked slowly, it predicted the course to the left with *idle-walk* behavior. At time t_2+15, it started to predict the possibility of *stop* at the shop, which finally came to be true at time t_2+22.

We measured the accuracy of position prediction for four time windows: 0-5, 5-10, 10-15, and 15-20 seconds in the future. Predictions were begun after a trajectory had been observed for 10 seconds, as the estimation of global behavior is not stable until then. We again used 6 days of data from the accumulated trajectories to predict the data of the remaining day. Our method predicts the future position as the center-of-mass of the expectation map. Figure 3.16 compares our method with position prediction based on the velocity over the last second. As the velocity method cannot account for motions like following the shape of the corridor, our method performs about twice as accurately.

We then measured the correctness of the system's predictions of the future positions and local behaviors for each person, evaluated in four places (indicated by three-meter circles in Figure 3.17) where qualitatively distinct behaviors were observed in the use-of-space analysis. For each place, at each moment, the system predicted whether the person would exhibit each of the local behaviors at that place for forecast windows of 0-5, 5-10, 10-15, and 15-20 seconds.

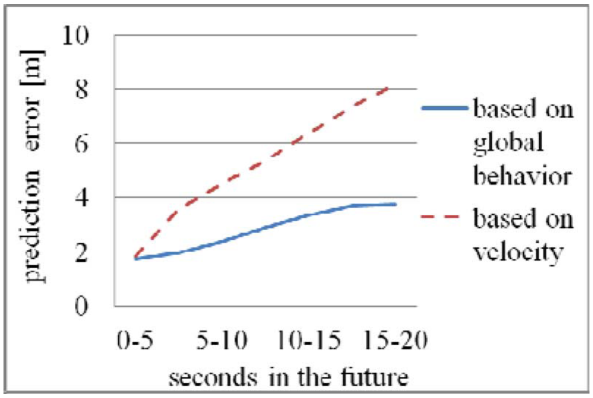


Figure 3.16: Prediction accuracy for position.

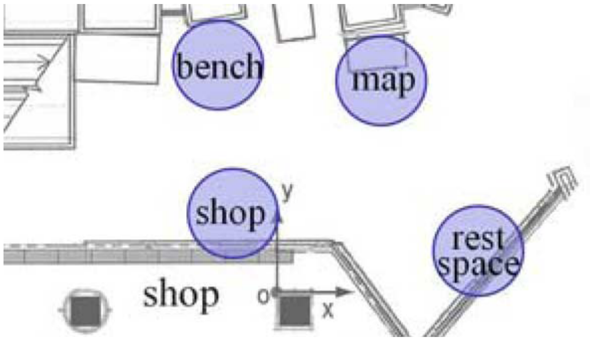


Figure 3.17: Places used for measuring performance.

Figures 3.18 and 3.19 show the system's prediction performance. In each figure, the left graph shows the accuracy of the prediction for the case where the target local behavior occurred at each place, and the right graph show the accuracy of the prediction where the behavior did not occur. We define the occurrence of the local behavior as the case where the person appeared at the place in the predicted 5-second window (*e.g.* between 5 sec and 10 sec), and performed the target local behavior more than other local behaviors. The accuracy value used for each person is the average across all predictions made for that person, and the value shown in the graph is the average across all people.

Figure 3.18 shows that the prediction was fairly accurate for the *stop* behavior, particularly at the bench and the rest space. Prediction was 92% accurate at the bench even for 15-20 seconds in the future, while non-occurrence was predicted with 88% accuracy. This good performance was due to the fact that people who stay in these areas often stay for a long time. Results were more marginal at the map and shop, with 62% accuracy for occurrence and 63% for non-occurrence predicted at the shop for 0-5 seconds in the future. For 15-20 seconds in the future, the performance is still marginal, with 48% accuracy for occurrence and 71% for non-occurrence predicted at the shop.

In contrast, as Figure 3.19 shows, the system predicted *idle-walk* with high accuracy 0-5 seconds ahead at the map and the shop. Even for 15-20 seconds ahead, the system was able to predict 33% of the occurrences at the shop as well as 86% of the non-occurrences, which we consider to be a good result, as it is rather difficult to predict walking behavior in the future. The prediction of occurrence was not successful at the rest space, as the system mostly predicted non-occurrence, since *idle-walk* rarely happened there.

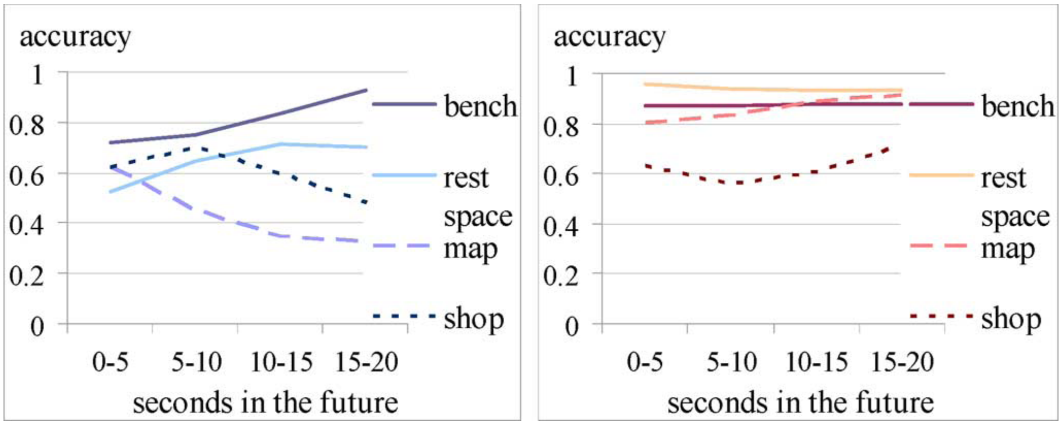


Figure 3.18: Prediction accuracy for *stop* behavior. (a) Behavior occurred at the place. (b) Behavior did not occur at the place.

Regarding the remaining two behaviors, for *wandering* (Figure 3.20), the system predicted over 50% of occurrences and 85% of non-occurrences for 0-5 seconds ahead at all four places. For the 15-20 second window, it predicted 73% of occurrences and 93% of non-occurrences at the bench but not so well for the map and shop. It predicted *fast-walk* (Figure

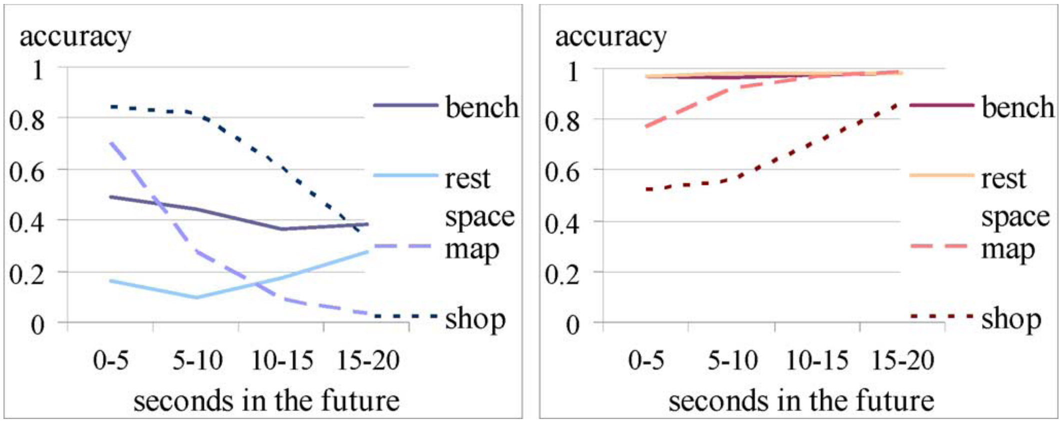


Figure 3.19: Prediction accuracy for *idle-walk* behavior. (a) Behavior occurred at the place. (b) Behavior did not occur at the place.

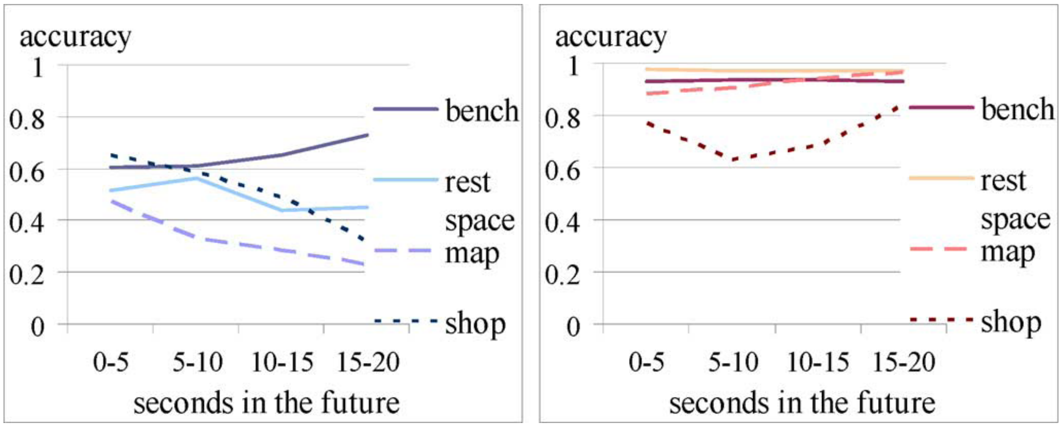


Figure 3.20: Prediction accuracy for *wandering* behavior. (a) Behavior occurred at the place. (b) Behavior did not occur at the place.

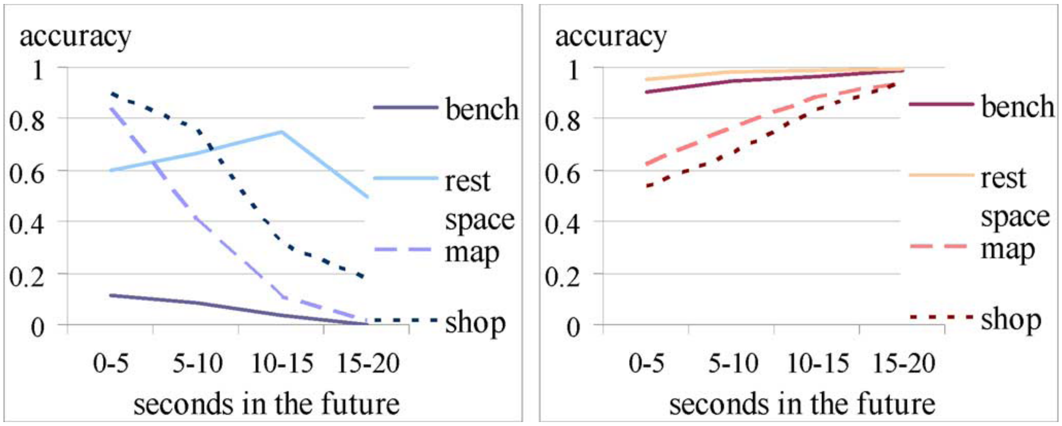


Figure 3.21: Prediction accuracy for *fast-walk* behavior. (a) Behavior occurred at the place. (b) Behavior did not occur at the place.

3.21) at map and shop well until 10 seconds; for example, it predicted 86% of occurrences and 60% of non-occurrences at the shop for 5-10 seconds in the future, though it does not predict the future well beyond 10 seconds.

We believe these anticipation results are useful for the robot. The robot is designed to wait for people in areas where it anticipates frequent occurrence of the target behavior. Behavior-based anticipation performs particularly well in areas where the anticipated behaviors occur often, such as *stop* near the benches and rest space, and *idle-walk* in the corridor in front of the map and shop. As these are the areas predicted by the location-based anticipation method, the two anticipation techniques complement each other nicely.

3.6 Service from A Social Robot

In this section, we show examples where a social robot provides services using our system. A human designer defines the contents of the service as well as the context in which the robot should provide the service. Here, the notable point is that the designer only specifies the target local behavior, such as “stopping”. The robot system then automatically computes the information about space and global behavior so that the robot can efficiently wait for people in promising areas, and then proactively approach people who are anticipated to perform the target local behavior.

For these services a robot has an advantage over cellular phones or other mobile devices, in that people do not need to carry any hardware; however, there is the additional challenge that robots need to approach the person quickly enough to start the service. For this purpose, anticipation plays an important role.

3.6.1 Robot Hardware

"Robovie" is an interactive humanoid robot characterized by its human-like physical expressions and its various sensors [84](Figure 3.22). Robovie has a head, two arms, a body, and a wheeled mobile base. Its height and weight are 120 cm and 40 kg. The robot has the following degrees of freedom (DOFs): two for the wheels, three for its neck, and four for each arm. On its head it has two CCD cameras as eyes and a speaker for a mouth. It is equipped with basic computation resources, and it communicates with the sensor network via wireless LAN. We used a corpus-based speech synthesis [87] for generating speech.

3.6.2 Entertainment Application

The first example of an application that we would like to discuss is an entertainment robot, which interacts with people in the form of chatting. As mentioned earlier, the shopping arcade is next to an amusement park, so it is a reasonable for the robot to be entertaining people who have free time. In addition, we think that such an entertainment service would be reasonable for a robot in other environments as well, as robots today are still an exciting novelty.

The chat was about the attractions in the amusement park. For example, the robot says, "Hi, I'm Robovie. Yesterday, I saw the Terminator at Universal Studios. What a strong robot! I want to be cool like the Terminator. 'I'll be back...' ". We set the target local behavior as *stop*, and non-target as *fast-walk*, in order to serve people who are idle.

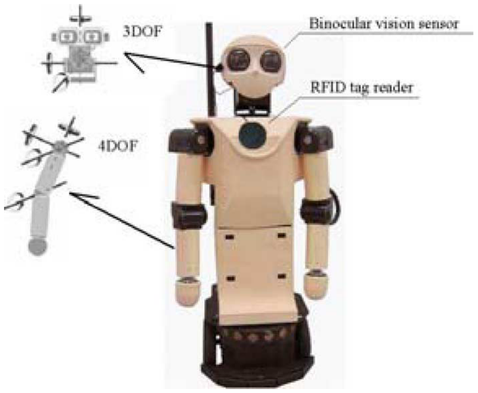


Figure 3.22: Robovie.

We conducted a field trial to investigate the effectiveness of the system. Figure 3.23 is a scene where the robot is approaching a person who is "stopping". Based on the anticipation mechanism and its current position, the robot set its roaming path near the bench and waited for a person to approach. When the robot predicted that a detected person would probably do the *stop* behavior, the robot began positioning itself near her general area (pre-approach) (Figure 3.23 (a)). When she came in front of the shop, she stopped (partly, we assume,

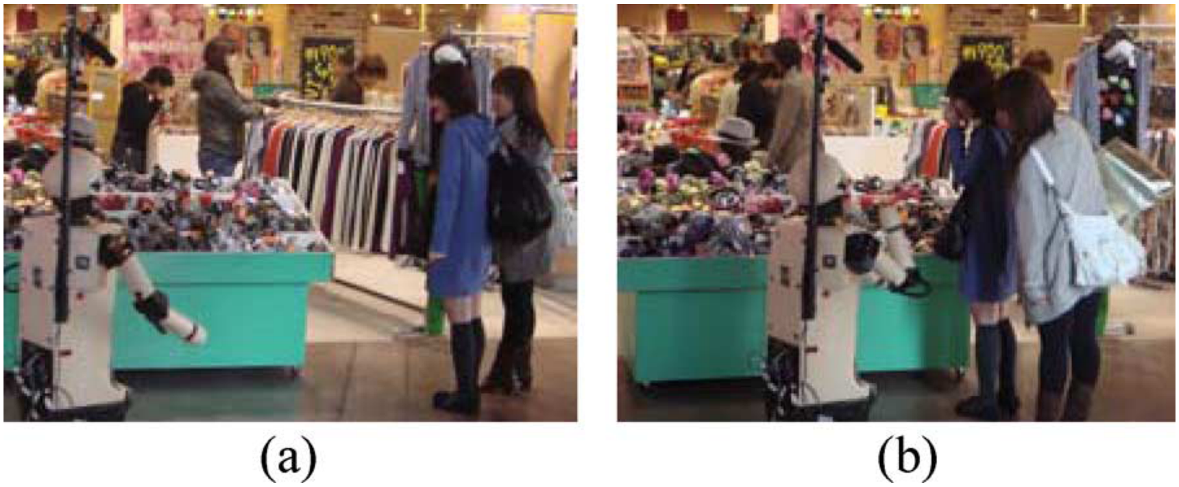


Figure 3.23: Robot approaching a person to chat.

because she was intending to stop regardless of the robot, and partly because she noticed the robot approaching her). Once she stopped, the robot approached her directly, and they had a chat (Figure 3.23 (b)). This is a typical pattern illustrating how people and the robot started to interact. Overall, people seemed to enjoy seeing a robot that approached them and spoke.

To evaluate the performance, we compared the situation with the developed system "with anticipation", and "without anticipation", and measured how much the anticipation mechanism improved the efficiency. In the "without anticipation" condition, the robot simply approached the nearest person who is doing the *stop* behavior. We measured the performance for one hour in total for each condition. We prepared several time slots and counter-balanced the order.

Figure 3.24 shows the number of people to whom the robot provided services. Due to the novelty of the robot, people often initiated interactions on their own; in such cases, the anticipation mechanism is irrelevant. Thus, we classified the robot's interactions into two categories. The first case, "robot-initiated", is the situation where the robot initiated the service by approaching the person and entering into conversation distance. Thus, the number of "robot initiated" services indicates how the robot's anticipation system improved the efficiency of the service. The second case, "person-initiated", is the situation where the person approached the robot while it was talking to someone else. Figure 3.23 shows one of such scenes. In this scene, when the robot was talking with the girls, a child came from the left. When the girls left, the child stood in front of the robot to start talking with it.

The results in Figure 3.24 indicate that the number of "robot-initiated" services in "with anticipation" is much larger than "without anticipation." In other words, anticipating enables the robot to provide the service more efficiently. Due to the novelty factor of the robot, the number of "person-initiated" services is quite large. We believe that in the future when robots are no longer so novel to people, there will be less person-initiated interaction, and the results concerning anticipation will become much more significant.

3.6.3 Invitation Application

The second example is one in which the robot recommends and invites the customer to visit a shop. In the shopping arcade, attracting people's attention to shops and products is an important task. We believe that this is also a reasonable service to expect from a robot, as the novelty of robots makes them very effective in attracting people's attention. The contents the robot provided were simple; for example, the robot said, "Hello, I'm Robovie. Do you enjoy shopping? I'd like to recommend this shop, where they sell clothes by the kilogram¡' Whenever it mentioned a shop, it pointed the direction of the shop with a reference term "this" or "that" [166].

We chose *idle-walk* as the target local behavior, because people who are walking slowly might be window-shopping. We set the non-target local behavior as *fast-walk*, so as not to bother people who seem uninterested in shopping. We used anticipation and the pre-approach function for the *idle-walk* behavior; when the robot predicted a person's future behavior as *idle-walk*, it moved towards that person's location.

We ran a field trial with the invitation robot in the shopping arcade as well. Just as in the entertainment application, the robot modified its behavior in accordance with the anticipation mechanism; the robot roamed around in front of the shop, where *idle-walk* was anticipated to be most likely, and approached people who were window-shopping.

In the demonstration, many people were interested in the robot and listened to its invitations. Figure 3.26 shows an impressive example where the robot approached a couple who were performing *idle-walk*. When the robot pointed to the shop and gave its recommendation (Figure 3.26 (c)), they smiled with surprise to see a robot performing a real business task. After the robot mentioned the shop, the woman walked directly to the shop and entered it (Figure 3.26 (d)). Observing such behavior indicates that such an invitation task can be a promising application. As indicated above, the robot was able to attract people's attention and redirect their interests to shops and products.

3.7 Discussion

3.7.1 Does the presence of the robot affect global behavior?

Our model is based on data recorded without having a robot in the environment. Thus, the system tried to predict people's behavior independent of the presence of the robot. However, as a robot is still a novel object, some people were attracted by the robot, slowed down, approached the robot, and even talked to the robot. In this case, the prediction cannot be correct, since such the behaviors are not in the model.

For the application shown in this paper, this had a positive effect on the robot's ability to provide the service. Even when the prediction from the robot was incorrect, as the robot

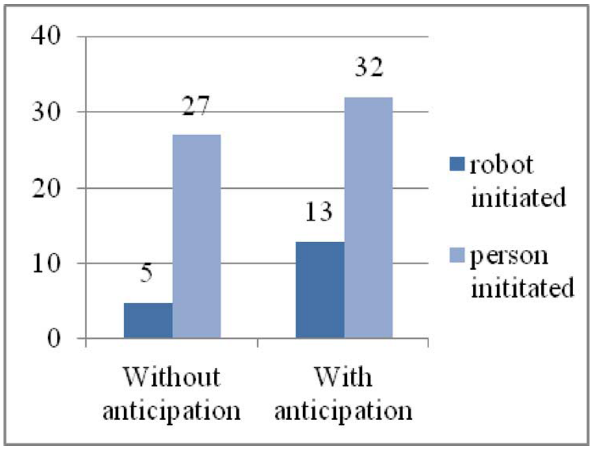


Figure 3.24: Number of services provided.

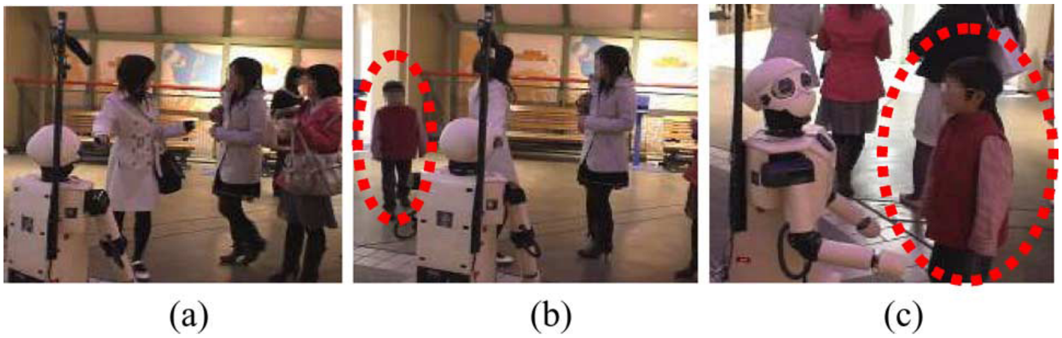


Figure 3.25: A child initiates interaction with a robot.

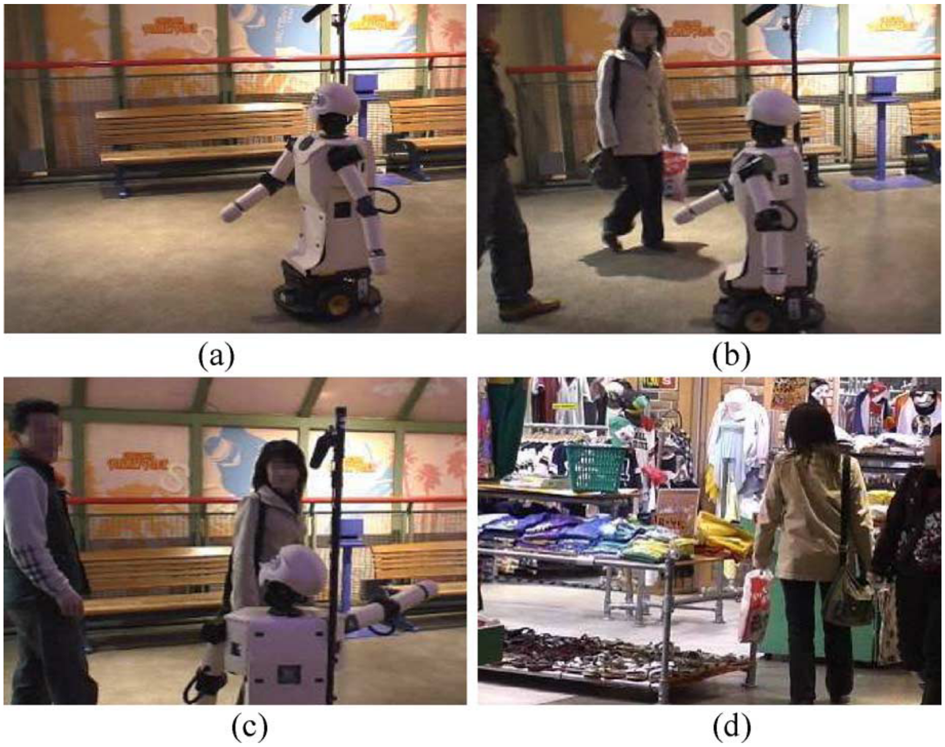


Figure 3.26: A robot successfully inviting a person to a shop.

approached, sometimes the person was nevertheless attracted by the presence of the robot, and stopped, which enabled the robot to provide its service.

For a different possible application such as a delivery task where the robot tries to avoid people in *idle-walk* and *stop*, however, this would affect the robot's ability negatively, as the robot's presence might attract a busy person to stop, and as a result the robot's route would be blocked. Thus, it will be useful to create a behavior model incorporating the effects of the robot.

3.7.2 To what extent is accuracy of positioning required?

In this study, we used a robust and accurate positioning technique with laser range finders; however, the whole approach does not depend on the positioning algorithm. In our previous work [82], we reported the analysis of global behavior where tracking was performed with RFID tags and readers, which provides people's position with 2.8 m error in an 80 x 40 m space. Like that example, our method is applicable for trajectories obtained through a different positioning technique. On the other hand, the classification of local behavior is based on some details of the position data. Thus, better positioning techniques will provide a better performance in local behavior classification.

One important characteristic of our positioning technique is robustness in terms of the continuity of the trajectory. Our method of analysis of global behavior requires that the whole length of the trajectories be observable. Thus, our method can be used with any tracking system that provides robust continuity of trajectories, even if it provides less positioning accuracy, e.g. our example with RFID tags and readers, but might be not feasible using a method without robustness in tracking.

3.7.3 Other possibilities of services with robots

Since we intended to highlight the connection between the robot and the infrastructure with ubiquitous sensors, we focused on the beginning part of the service (finding a person, approaching, and initiating conversation), and show two simple examples of services such a robot could provide. These services are appropriate under the situation where a robot is novel to people. Even such simple services provide enough value to people who are eager to experience an interaction with the robot. As a future scenario, we can extend the service by having a designer in the context design. For example, many people stopped in front of the map, which can be seen in the analysis of the use of space; after discovering this fact, we can design a robot to provide guidance services for a person who is standing in front of the map.

3.7.4 Other Possible Applications

We believe that the infrastructure shown in the paper can be useful for other systems, *e.g.* ubiquitous computing applications. One possible direction is to apply it to ambient intelligent environments, in which facilities (robots, display, music, illumination, etc.) are proactively controlled according to the types of users. For instance, an electronic poster could anticipate who is likely to stop nearby, and change its advertisement content in advance to something targeted to that person.

Another possibility is to combine it with mobile devices. Although GPS and WiFi have been used for locating people, laser range finders can provide more accurate positioning. The information provided by the infrastructure developed here could also complement other location-based services. For instance, if a user with a mobile device providing pedestrian navigation information entered a space with this infrastructure available, the device could then present additional information appropriate to that user's anticipated global behavior.

3.7.5 Privacy Concerns

Systems operating in public spaces should be carefully designed to protect the privacy of people. In our application, the system does not identify individuals (e.g. names), and it finishes tracking people when they leave the environment. We believe that this is a privacy-safe application. When the system is scaled up (e.g. extended to cover a large area, or associated with personal information), privacy should be more carefully considered.

3.8 Conclusion

We reported a series of abstraction techniques for retrieving information about people's behavior from their trajectories. Based on robust tracking with multiple laser range finders, more than ten thousand trajectories have been accumulated. Clustering techniques revealed how they used the space as well as their global behavior in the environment. Our service framework includes an anticipation system: it utilizes abstracted information to send a robot to provide services to people who are exhibiting a pre-defined local behavior associated with a particular service. It is notable that designers need to only specify target local behavior to use the anticipation system.

Results from our field trial demonstrated the effectiveness of the service framework, and also indicated that entertainment and invitation are promising applications for the robot. People appeared excited about the presence of the robot, enjoyed interacting with it, and sometimes followed its invitations. The service framework developed here enables the robot to provide such services in a real shopping arcade. Further details about people's response to the robot were examined in more detail in succeeding studies, e.g. a study of social behavior in approaching humans [139] and integration of different capabilities of robots [151], which are based on the techniques and service frameworks reported in this paper.

Chapter 4

Teleoperation of Multiple Social Robots

This chapter presents a system for assisting robots in conducting conversational interactions by employing a remote human operator to perform speech recognition and other tasks. As it would be inefficient to require one human to assist each robot, the proposed design enables a single human operator to serially assist several robots. As recognition capabilities and autonomy of the robots improve over time, efficiency will increase, and one operator will become able to supervise large robot teams.

Teleoperation of multiple robots by a single operator has been studied extensively for applications such as search and navigation; however, this concept has never been applied to the field of social, conversational robots. This chapter explores the unique challenges posed by the remote operation of multiple social robots, where an operator must perform auditory multitasking to assist multiple interactions at once. This chapter will discuss several important design issues, present models and metrics for task performance, and introduce a technique called “Proactive Timing Control,” an automated method for smoothly interleaving the demands of multiple robots for the operator’s attention.

4.1 Introduction

As rapid progress is being made on all frontiers of robotics technology, many of the key components necessary for developing socially-situated autonomous robot systems are falling into place. Field trials of social robots placed in real-world environments such as museums [15, 152, 159], schools [55, 83, 113], and train stations [65], have shown great success and provided valuable insights into real-world social phenomena which cannot be observed in the laboratory.

Yet, inspiring and exciting as it is to see robots operating in the field, the inescapable reality is that social dynamics and recognition problems are complex, and today’s technology is not yet capable of supporting a fully-autonomous robot playing a humanlike role in society. Any robot will eventually find itself in unanticipated circumstances, where failure to respond appropriately could lead to socially awkward, money-losing, or even dangerous situations.

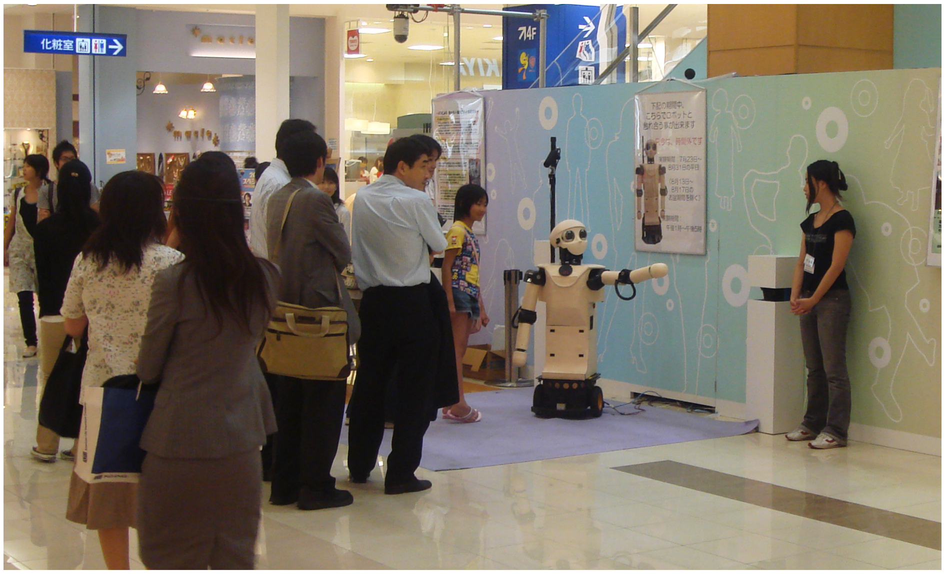


Figure 4.1: A robot providing route guidance in a shopping mall.

A field trial we recently conducted at a Japanese shopping mall [85] illustrates an example of a social robot application. We placed a humanoid robot in a central public space in the shopping mall for several hours a day, where it chatted with visitors and provided information and route guidance to locations within the mall. Customers were excited by the engaging interactions, and people crowded around the robot every day, waiting for a chance to talk with it (Fig. 4.1).

Although a large part of the attraction of social robots is their ability to "understand" natural language and engage people interactively, this task is still largely beyond the capabilities of today's robots to achieve without a human operator. Field trials using robots in social settings have often involved some degree of remote control, referred to as the "Wizard of Oz" (WoZ) method [59, 185]. Although pure teleoperation can be valuable for studying human reactions to robot behaviors, it does not necessarily represent progress towards creating fully-autonomous or highly-autonomous social robot systems.

With real-world semi-autonomous robot applications as a goal, our long-term approach is to begin with a partially-autonomous system, and to steadily decrease the role of the operator over time with improvements in robot technology. As robot autonomy increases, it will be possible for one operator to control several robots. The operator-to-robot ratio could be considered as one measure of the degree of autonomy of a robot system.

Seen from a commercial perspective, a fleet of ten service robots controlled by a single human operator would be more economically viable than the same number of robots requiring a team of twenty operators, and even highly autonomous industrial robots generally have a human in the loop in a supervisory role. Thus, while full autonomy for social robots is not yet feasible, partial autonomy with a low operator-to-robot ratio could enable social robot applications which would be otherwise impractical.

In this paper, we address the unique challenges of single-operator-multiple-robot (SOMR) operation for the case of social robots. In Sec. III we present a framework in which we categorize and discuss the key issues in designing such a system. Based on this conceptual framework, we implemented a semi-autonomous robot control system for social interactions, enabling a single operator to monitor and control several communication robots at once. The details of our implementation and solutions to key problems are presented in Sec. IV. In the remainder of the paper, we present results showing the effectiveness of our system in simulation, as well as laboratory trials demonstrating that a single operator is able to successfully control up to four robots at once as they simultaneously engage in conversational interactions.

4.2 Related Work

In this paper we are exploring semi-autonomous control of multiple robots for social human-robot interaction, by which we mean conversational interaction between a robot and one or more people. In other fields of robotics, such as search-and-rescue or space exploration, many aspects of both single- and multiple-robot teleoperation are active fields of research, but multiple-robot teleoperation has not yet been studied for social robots.

A substantial amount of work has been done regarding levels of autonomy for teleoperated robots. The concept of "shared autonomy" describes a system in which a robot is controlled by both a human operator and an intelligent autonomous system, a concept which has been used in fields such as space robotics [14] and assistive robotics [179]. The concept of "adjustable autonomy", also known as "sliding autonomy," has also been studied, in which varying degrees of autonomy can be used for different situations [80, 57, 141, 145].

Other teleoperation research has focused on control interfaces for teleoperation. A wide variety of teleoperation interfaces have been created for vehicle control [39, 18], and the unique problems of controlling body position in humanoid robots have also been studied [156].

Several aspects of simultaneous control of multiple robots have also been studied. Hill and Bodt presented field studies observing the effects of controlling multiple robots on operator workload [70]. Sellner et al. studied the situational awareness of an operator observing various construction robots in sequence [146], and Ratwani et al. used eye movement cues to model the situation awareness of an operator supervising several UAV's simultaneously [131].

A key issue in multiple-robot teleoperation is the concept of "fan-out," which describes the number of robots an operator can effectively control [26]. Crandall and Goodrich have laid a theoretical basis for the modeling of SOMR teleoperation, defining metrics such as Interaction Time (IT) and Neglect Tolerance (NT) to help with calculation of robot fan-out and predicting system performance [22]. Thus far, studies of fan-out in multiple-robot

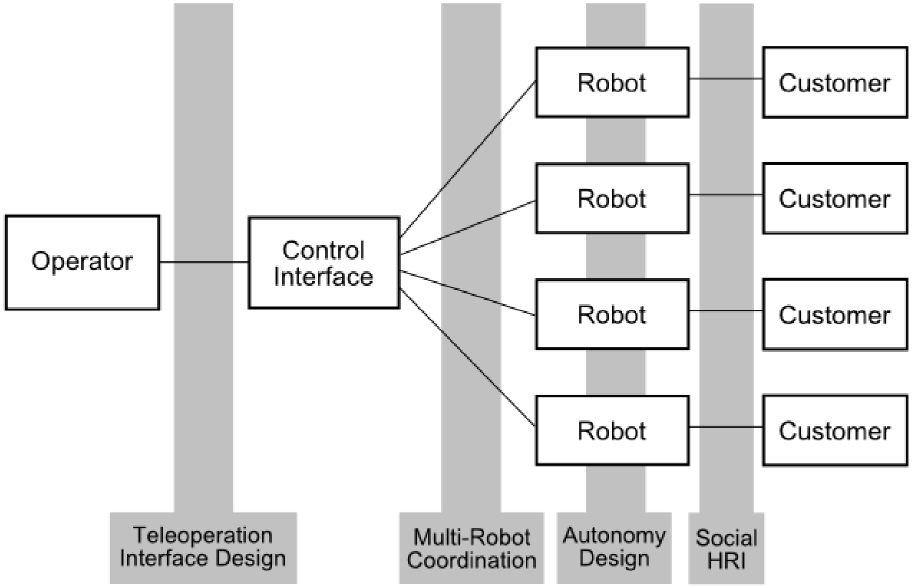


Figure 4.2: General overview of multi-robot control system showing key design areas.

teleoperation have focused on tasks such as search and navigation for mobile robots [21], or target selection for UAVs [24], but not social human-robot interaction.

In this paper we will build upon this research to define a new application domain: the teleoperation of multiple robots for social human-robot interaction tasks. In doing so, we aim to identify ways in which existing SOMR teleoperation principles can be applied to social robots, and to examine ways in which social robots differ from traditional systems.

4.3 Issues in Teleoperation of Multiple Social Robots

The teleoperation of multiple robots for social interaction is in some ways analogous to SOMR teleoperation for conventional robots, and in other ways presents new challenges. Extensive research has been done on teleoperation for tasks such as robot navigation, and we have summarized how the issues in teleoperation for conversational social interaction differ from those regarding many kinds of teleoperation for navigation (Table 4.1). Of these differences, perhaps the most significant is the time-critical aspect of conversational interaction. Time-criticality itself is not unique to social robotics, and time-critical tasks exist, for example, in UAV teleoperation; however in most SOMR systems, robots can buy time by “idling” or “loitering” until an operator becomes available, whereas a robot waiting in silence during a conversation would quickly cause failure of the interaction. Thus time-criticality is a central factor affecting several of the key issues in teleoperation of multiple social robots.

Fig. 4.2 shows the general organization of a SOMR system for social human-robot inter-

	Navigation	**Social interaction (this study)**	**New problems in social interaction**
Operator's role	Obstacle avoidance. Giving current position, path, goals.	Understanding the user's intention and providing required service	
Source of input to operator	Scenery + Map	Audition (+scenery)	Cannot monitor multiple sources simultaneously
Operator's output (low level control)	Velocity	Utterance, gesture, +(body orientation and position)	Typing and controlling many DOFs for gesturing are very slow
Operator's output (abstracted control)	Position (destination)	Behavior (combination of utterance and gesture)	Difficult to prepare for minor cases in advance
Consequence of ignoring errors caused by autonomy	Crash into obstacle, or lose the robot	Person might get lost, buy wrong product, or receive wrong service.	Definitely we should not ignore errors in either case
Can robots wait after an error is detected?	Yes, in most cases.	No. Users might soon leave if a robot stops.	An operator should take control of the robot immediately.
Can robots anticipate the timing of possible error?	Not usually.	Yes.	Most errors are from speech recognition, often after the robot asks a question.

Table 4.1: Differences in teleoperation between navigation (Fundamental tasks for mobile robots [163]) and social interaction.

action. In this paper we will use the terms "operator" and "customer" to describe the roles of humans in the system. This choice of terms is not meant to preclude other roles of the humans in the system, e.g. teacher-student or doctor-patient, but is only used to avoid the ambiguity of the term "user".

Four key design areas are identified in the system diagram in Fig. 4.2. The overall system requirements are driven by the target application, which in this case falls in the domain of **social human-robot interaction**. This area includes the design of the robot's behavior and dialogue with the goal of creating comfortable, natural, and functional interactions between the robot and a customer. To create semi-autonomous robots which can do this, an important issue is **autonomy design**, that is, how operator commands can be reconciled with the autonomous components of the robot control system. Next, due to the time-criticality of social interactions, **multi-robot coordination** is necessary to manage the attention of the operator between robots, and to reduce conflicts between demands for the operator's time. Finally, **teleoperation interface design** is necessary to enable interaction between the operator and the robot, providing the operator with situation awareness and controls for operating the robot.

In this section, we will present design considerations in these four areas and propose metrics for quantifying important characteristics of SOMR systems for social interaction.

4.3.1 Social Human-Robot Interaction

The target application of social human-robot interaction drives the design of the entire robot system. As the field of HRI covers a wide range of scenarios, it is important to clearly define the target of this paper.

In this study we are considering conversational humanlike interactions. The task of the robot is primarily dialogue-based, although nonverbal communication and gestures such as pointing may also be essential interaction components.

Some examples of this type of interaction might include a robot shopkeeper which provides information about various products, an information booth robot which gives directions and answers questions in a shopping mall, a tour-guide robot which explains exhibits in a museum, or a public relations robot which greets people and invites them to visit a shop.

In these examples, interactions can be expected to follow a flow which includes alternating phases: one in which a person is asking a question or giving information to the robot, and one in which the robot responds with some explanation or directions.

In the first type of phase, it is the customer's 'turn' to drive the conversation, and the robot (or operator) must correctly recognize the customer's utterances in order to respond appropriately. We call this type of phase a **critical section**, because a recognition failure in this phase is likely to result in a failure of the interaction, such as a customer becoming frustrated with the robot and walking away.

In the second type of phase, it is the robot's 'turn' in the conversation, and the customer is in a listening role. Responding to inputs from the customer is less important during this phase, which we call a **non-critical section**. This is not to say that the customer will never interrupt the robot, but such interruptions are expected to be the exception rather than the rule. Although recognition failures may occur in this phase, we assume that in comparison with critical sections, there is a lower likelihood that they will result in interaction failures.

Understanding this pattern of critical and noncritical sections defined by the social interaction design helps to enable the coordination of operator attention between multiple robots, as we will explain later.

4.3.2 Autonomy Design

In semi-autonomous social robot systems, it is important to define how an operator should interact with the autonomous components of the robot's control system. Generally speaking, an operator can direct high-level tasks or identify errors that the system cannot detect autonomously. For social robots, many necessary functions, such as tracking human positions or presenting information through speech and gesture, can be performed autonomously using available technology. Some core background processes, such as emotional dynamics, can also be automated for social robots [3]. It is in the recognition and interpretation of verbal and nonverbal communication and the ability to make common-sense judgments based on an understanding of context that an operator can add the greatest value.

For example, an elderly person in a shopping mall who is holding a map and looking around might need route guidance from the robot; on the other hand, a young person in a plaza looking around in a similar manner might just be looking for friends and not need the service. Although a human operator could easily distinguish between these two cases using intuition, visual cues, and implicit social context, such a recognition task would be quite difficult for a robot to perform autonomously.

An operator can provide input to a semi-autonomous system at several levels. Consider a simple framework for robot control, in which developers create sense-plan-act elements based on a pre-assumed world model. Fig. 4.3 shows an example of such a system, in which the robot can perform abstracted behaviors composed of low-level actions such as speech and gesture. These behaviors are chosen by decision logic, based on the results of autonomous sensor recognition.

In such a system, three categories of problems tend to occur, which define the three primary tasks of an operator.

Uncovered situations

The richness and diversity of human behavior makes it difficult to create a predictive model of the world for social interactions. This can lead to many **uncovered situations**, in which a robot does not have appropriate rules or behaviors implemented to act autonomously. Uncovered situations are of particular concern for systems which interact with humans.

Uncovered situations motivated the original use of WoZ, where a dialog system was controlled by a human operator to collect necessary dialog elements [25]. By monitoring the interaction, an operator can provide additional information to the system and improve its world model. The assumption behind this technique is that the robot can ultimately cover all situations after collecting a sufficiently complete world model.

Incomplete autonomy

Even assuming a good model of the world, there are still cases when we cannot prepare all the necessary sense-plan-act elements. In these cases an operator can be used as a substitute for **incomplete autonomy** and replace those individual elements. Many WoZ studies in HRI are of this type [164].

An example of replacing a *sense* element is speech recognition. Today's speech recognition technologies are unreliable in noisy environments, as observed by Shiomi *et al.* in field trials [153]. It is thus not currently possible to automate this sensing task. However, an operator can be employed to listen to the audio stream and manually input the recognized utterances into the system. Using those inputs, the robot can still perform the plan and act elements autonomously. Other examples in this class could include identifying a person or object, or monitoring the social appropriateness of a robot's actions by observing people's reactions to the robot.

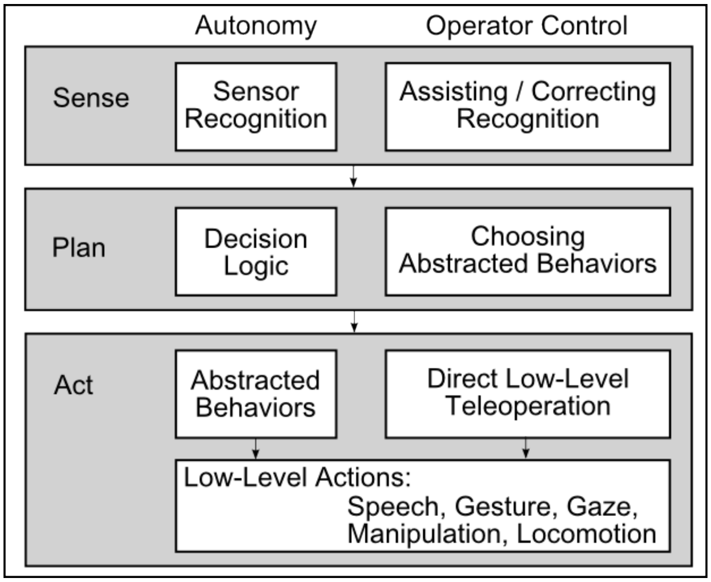

Autonomy
Operator Control
Sense
Sensor Recognition
Assisting / Correcting Recognition
Plan
Decision Logic
Choosing Abstracted Behaviors
Act
Abstracted Behaviors
Direct Low-Level Teleoperation
Low-Level Actions:
Speech, Gesture, Gaze, Manipulation, Locomotion

Figure 4.3: Autonomy and operator control tasks for sense-plan-act elements in a semi-autonomous robot control system.

An operator could likewise replace a *plan* element. If a robot's action requires particular expertise or authority, such as that of a doctor, technician, soldier, or law enforcement officer, an operator may be required for this step. Here the robot may be able to sense the environment and act on it, but lack the authority or accountability to make the decision to act.

For replacing an *act* element, an example could be a difficult actuation task like grasping. The robot might be able to identify an object to grasp and make the decision to grasp it, but need assistance in actually carrying out the grasping task [155].

Note that for this style of teleoperation, the system can often prompt the operator to perform some action. The operator acts as a "black box" in the system, performing some defined processing task on demand, like any other module in the system.

Unexpected errors

Finally, it is possible that even if we have prepared a good world model and developed appropriate sense-plan-act elements, the system may not always work as intended. That is, **unexpected errors** may occur during autonomous operation.

In this case, an operator needs to monitor the robot to identify possible errors. In the teleoperation tasks described above, the operator's focus is on the environment and people interacting with the robot, but when monitoring for errors the primary focus is on the performance and behavior of the robot itself.

4.3.3 Multi-robot Coordination

As stated in Table 4.1, we assume an operator can only correct errors or provide active support for one robot at a time. Particularly in the case of speech recognition, it is extremely difficult for an operator to concentrate on two or more conversations at once. This restriction makes the operator's attention a limited resource.

In this paper we will model a robot's interaction as consisting of **critical sections**, where there is a high risk of interaction failure and thus a high likelihood that operator assistance will be needed, and **non-critical sections**, which can safely be performed autonomously. Critical sections tend to occur when the actions of the robot depend strongly on recognition of inputs from the customer, and thus the consequences of a recognition error are severe. Critical sections can also occur when there is a high probability that an uncovered situation will arise.

Note that we consider errors in sensor recognition to be equally likely to occur in critical and non-critical sections. However, a recognition error is much more likely to result in interaction failure in a critical section than in a non-critical section. To prevent interaction failures, it is desirable for an operator to be monitoring a robot during critical sections.

A fundamental conflict arises when two or more robots compete for operator attention by entering critical sections at the same time. As noted above, social interactions are time-critical. While the operator serves one robot, the customer interacting with the other robot is made to wait, which will have a negative impact on the quality of service, and possibly even cause failure of the interaction.

In Sec IV-C, we will propose a mechanism for coordinating the interactions of the robots to eliminate such conflicts.

4.3.4 Teleoperation Interface Design

An operator has two tasks to perform: first, supervisory monitoring of all robots to identify unexpected situations, and second, assisting individual robots' recognition, planning, and actuation. Supporting both of these tasks provides a considerable challenge for the user interface design.

Both situation awareness and actuation requirements for the user interface differ for these two tasks as follows.

Controlling individual robots

When controlling a single robot, the operator needs to be aware of the robot's individual situation – with whom the robot is interacting, what that person is saying, and what the robot is doing. For simple systems, such as an information-providing robot in a shopping mall, this immediate information may be sufficient for the robot's interactions. For more elaborate systems where the robot has a long-term relationship with the customer, long-term interaction history or personal information about that customer might be required.

This interface also requires actuation controls for correcting sensor recognition, directing behaviors, and performing low-level control such as entering text for the robot to speak in uncovered situations.

Monitoring multiple robots

When acting in a supervisory role and monitoring multiple robots, the operator needs to identify and react to unexpected problems in a timely manner. A summary of the state information about each robot should be presented to the operator in such a way as to make errors and unusual behavior easily recognizable.

As stated in Table 4.1, an understanding of the conversation flow can make it possible to anticipate when errors in recognition are likely to happen. The highest risk of recognition error occurs during critical sections, so alerting the operator of which robots are in or entering critical sections can help manage the operator's attention most effectively.

It should also be noted that a summary of the robot's state information might not be sufficient for the operator to accurately identify some errors, so it may be important for the operator to periodically examine the detailed state information for individual robots as well.

Metric	Comments
Recognition Accuracy (RA)	Limited by technology. Higher RA increases fan-out.
Situation Coverage (SC)	Limited by scenario predictability. Higher SC increases fan-out.
Critical Time Ratio (CTR)	Determined by interaction design. Lower CTR increases fan-out.

Table 4.2: Task Difficulty Metrics

4.3.5 Task Difficulty Metrics

Finally, it is valuable to have metrics quantifying the capability of the robot system. For multiple-robot systems, a key quantity is the number of robots a single operator can manage, known as “fan-out”. High fan-out can be achieved if the robots can operate with high reliability without the support of an operator, whereas fan-out will be much lower if errors are likely to occur, for example, due to poor sensor recognition or high task difficulty. Thus, to predict fan-out, it is important to have metrics which describe the likelihood of error while the robot is unsupervised. In the terminology of Crandall and Cummings, such metrics are classified as “Neglect Efficiency” metrics [21]. In this section, we will define three neglect efficiency metrics reflecting the risk of interaction errors occurring while the robot is unsupervised. These metrics are summarized in Table 4.2.

Recognition Accuracy

Sensor recognition accuracy (RA) is a fundamental concern for robots in nearly every field. This is also true for social robots, as recognition of the nuances of communicative signals such as speech, gesture, intonation, and emotion in social interaction can be particularly challenging. An estimate of RA can help predict the frequency of unexpected errors in the “sense” element of the robot’s control architecture.

The RA of a system should be evaluated in the context of its intended application. Visual recognition accuracy varies greatly with lighting conditions, and audio recognition accuracy is dependent on levels of ambient noise. Variability in interactions can also affect RA. For example, a robot may perform excellent speech recognition while answering a predictable set of questions in an office setting, yet quite poorly in recognizing the unrestricted utterances and emotional signals of children telling stories to the robot at a day-care center.

From a designer’s perspective, increasing a robot’s RA through better sensors or better recognition technology can reduce the need for operator intervention, which can increase the number of robots a single operator can control. The designer’s freedom, however, is typically limited by available technology, and thus RA cannot be increased without bound.

	Phase	**Criticality**	**Duration**
1	Simple greeting	Non-critical	2s
2	Self-introduction	Non-critical	3s
3	Chat behavior	Non-critical	Variable
4	Offer guidance	Critical	1s
5	Wait for question	Critical	2-10s
6	Provide guidance	Non-critical	10-15s
7	Farewell	Non-critical	5s

Table 4.3: Interaction Sequence

Situation Coverage

The next metric we propose is Situation Coverage (SC), which describes the completeness of the "plan" and "act" elements in the robot system. We define a situation to be "covered" if the system would autonomously execute the correct behavior given perfect sensor inputs. Using this definition, SC is defined as the percentage of situations encountered by the robot that are covered.

To be precise, there are actually two aspects to SC, corresponding to the "plan" and "act" elements of the robot control system. $\mathrm{SC}_{\mathrm{act}}$ describes the percentage of situations encountered by the robot for which an appropriate action has been prepared. $\mathrm{SC}_{\mathrm{plan}}$ then describes the percentage of situations for which the decision logic has been developed which will trigger those actions.

For example illustrating the difference between $\mathrm{SC}_{\mathrm{act}}$ and $\mathrm{SC}_{\mathrm{plan}}$, consider a robot system which includes an implemented action to direct a customer to a supermarket (*i.e.* covered under $\mathrm{SC}_{\mathrm{act}}$). Assume decision logic has been implemented to execute this action only if a customer asks where the supermarket is. If a customer asks this robot where to buy broccoli, but the robot is not programmed to react to the word "broccoli", this situation is not covered under $\mathrm{SC}_{\mathrm{plan}}$, and is thus not considered to be a covered situation overall, even though it is covered under $\mathrm{SC}_{\mathrm{act}}$.

Overall, SC describes a theoretical limit of the system's capacity to operate autonomously. In an ideal system with no recognition errors or unexpected errors, a system with an SC of 70% will be able to successfully complete 70% of its tasks autonomously, and will require operator intervention 30% of the time. When errors are considered, real autonomous performance will fall somewhat below 70%, so SC is useful for describing the upper bound of the system's possible performance, or conversely, a lower bound on the fraction of time during which operator support may be necessary.

In application design, SC is more of a controllable variable than RA. Whereas RA is subject to technological limitations, SC can be increased through human effort. By spending more time researching potential situations the robot may encounter and developing the decision logic and actions to respond to those situations, it is possible to increase a robot's

SC.

Given the complexity and variety of real social situations, it is usually impractical to attempt to achieve 100% SC. Instead, an effective strategy for use of partial autonomy would be to design logic and actions to cover the most common situations, perhaps achieving an SC of 90%, and then to rely on operator assistance for the remaining situations.

Critical Time Ratio

The third metric we will introduce is the Critical Time Ratio (CTR). This is defined as the ratio of the amount of time spent in critical sections to the total duration of an interaction. For tasks with a low CTR, the likelihood of two robots entering a critical section at the same time is correspondingly low, and thus timing control behaviors will seldom be necessary. Tasks with a high CTR are more likely to conflict, which can lead to higher wait times for users and a heavier workload on the operator.

CTR is related to the concept of Robot Attention Demand (RAD) presented by Olsen et al. [125]. RAD represents the fraction of total time that human attention is required. CTR, on the other hand, only describes the pattern of critical and non-critical sections. The degree to which operator attention is required during these critical sections is dependent upon the overall risk of failure, which is in turn based on RA and SC.

For a designer, it is possible to achieve higher fan-out by creating interactions with a low CTR, e.g. by increasing the durations of non-critical sections and minimizing the number of critical sections in the interaction flow. However, this must be done carefully, as reducing CTR also runs the risk of reducing the robot's responsiveness to the customer, and thus reducing the quality of the human-robot interaction.

4.4 Implementation

Using the principles presented in this paper, we developed a system for the teleoperation of multiple robots for social interactions.

In this section we will present the implementation of our system, addressing each of the four design areas presented in Fig. 4.2: social human-robot interaction, autonomy design, multi-robot coordination, and teleoperation interface design. Finally, we will present an example of how an operator would interact with such a system while controlling multiple robots.

4.4.1 Social Human-Robot Interaction

The interaction flow we developed for this study was based on interactions used in our field trials in a shopping mall. Table 4.3 shows the sequence of conversation phases and their durations. When the robot detected a person in front of it, it would greet the person (1), then introduce itself and explain that it can give directions to locations in the shopping mall (2).

After this, the robot would briefly chat about some topic, usually related to current events in the shopping mall or the robot's "experiences" at various shops (3).

As noted earlier, critical sections include situations where a response from the user is expected, whereas non-critical sections include tasks such as greeting, talking, and giving directions, where the robot is primarily providing information. The critical sections in our flow consist of the robot asking where the customer would like to go (4), and then waiting for the customer's response (5).

After the question has been asked, the robot gives guidance (6), then says goodbye to the customer (7). All of these phases are considered noncritical.

4.4.2 Autonomy Design

For this study, we created a semi-autonomous robot control architecture which enables an operator to provide commands and assistance to an otherwise autonomous robot system.

Robot Platform

We implemented our architecture on Robovie II, a humanoid robot platform developed for human-robot interaction research. It is capable of humanlike expressions with a head that can be moved with 3 degrees of freedom (DOF), arms with 4 DOF each, eye cameras with 2 DOF each, and a wheeled base for locomotion. Each robot also has color CCD eye cameras, a microphone, and several touch sensors.

Audio from the robot's onboard microphone is processed by an automatic speech recognition (ASR) system. In our field trials we have found the ASR system to be unusable because of ambient noise from background music, crowds, and announcements. In our quieter laboratory environment we found it to be more reliable, but accuracy was still observed to be around 60%.

A common difficulty in speech recognition is that the signal-to-noise ratio goes down as distance between the microphone and the person speaking increases. Using headset microphones would certainly improve accuracy, but their use would be impractical for real robots interacting with customers in the field.

Robot behavior control

The behavior control system used in this study uses a software framework, illustrated in Fig. 4.4, in which short sequences of motions and utterances can be encapsulated into discrete units called "behaviors". The programmer then defines a set of transition rules called "episodes" which specify transitions between behaviors [84]. These rules can be based on execution history, like the following examples:

If behavior A was executed, execute behavior B next.

If behavior B was executed immediately after behavior A, then execute behavior C next.

The transitions can also be based on return values of the behaviors. This enables us to incorporate sensor information into the transitions. For example, a “check for customer” behavior could be defined which returns a 1 if a person is detected in front of the robot, and a 0 if no one is detected. This can be used to create a simple waiting loop, as follows:

If behavior A returns 0, repeat behavior A.

If behavior A returns 1, execute behavior B next.

In practice, we have used this framework to create “listen” behaviors which return tens to hundreds of different values based on speech recognition results, as well as action-oriented behaviors such as a “shake hands” behavior which offers to shake hands and returns different values based on the reaction of the person.

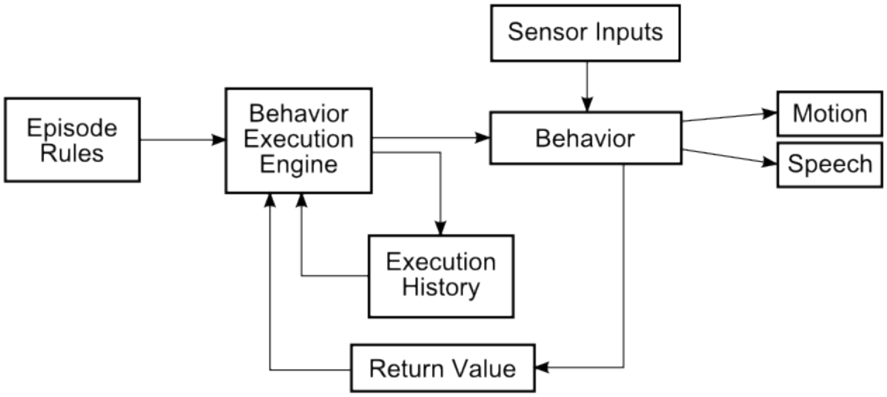


Figure 4.4: Behavior execution architecture.

With this framework, if we theoretically assume no errors in sensor recognition and user behavior only within the limits of Situation Coverage, it is possible for the robot to execute any length of behavior chains with full autonomy.

Operator intervention

As described in Sec. III-B, the operator needs to be able to intervene in robot operation to deal with uncovered situations, incomplete autonomy, and unexpected errors. This can be achieved either through direct control of the robot at a high or low level, or through correcting the robot’s recognition.

Direct Control Improvements in the efficiency of robot control can be made possible through layers of abstraction. For example, an operator could specify the individual joint angles for the robot’s arm at a low level, or achieve the same result by giving the robot a high-level command, e.g. “point to the left”. Most robot systems already incorporate abstractions like this. Joint angles can be grouped into poses, poses grouped into motions,

motions and utterances grouped into behaviors, and so on. Similar abstractions have been used in teleoperation systems for navigation [22, 57].

As layers of abstraction are added to the system, the robot usually becomes able to function with a higher degree of autonomy, thus reducing the workload for the operator. When high-level functions are not prepared for a situation, the operator can use low-level functions instead. For example, if there is no behavior prepared for giving directions to a Japanese restaurant, an operator might directly type phrases for the robot to say and control the arms manually to point the way.

Correcting Recognition An operator can also choose to correct a robot's sensory recognition errors, rather than completely taking over control of its behaviors. For example, an operator observes a scene where a user says the words "Japanese restaurant", but the speech recognizer fails to pick it up. If the robot has behaviors in place to react to those words, the operator can correct the robot's speech recognition results and allow the robot to complete the interaction as usual.

This kind of control requires less effort from the operator than taking over behavior control in order to generate a guiding behavior for directing the user to a Japanese restaurant.

To give a simple example, when a robot in an idling state detects a person approaching, the episode rules may trigger a transition from the idling behavior to a greeting behavior. After greeting the person, the next behavior might be to offer route guidance and wait for a response. The transition rules would then choose the next behavior based on input from the speech recognition system. If the person asked for directions to a bookstore, and if we assume the speech recognition system correctly recognized the word "bookstore", the system would then transition to the module for giving directions to the bookstore.

4.4.3 Multi-Robot Coordination

We will consider two possibilities for handling conflicts between robots for the operator's attention. First, a naï¿½ve method of simply alerting the operator of critical sections and second, a technique we call Proactive Timing Control.

In the first approach, each robot can notify the operator of a critical section, and then proceed in its interactions regardless of the state of the operator or other robot(s). If the interaction reaches a point where the robot is unable to respond without operator intervention, the robot will need to stall for time [154] until the operator becomes available.

The robot can simply wait in silence, or it can repeat phrases like, "hmm... hold on... please wait" until the operator can provide assistance. The drawback of this approach is that such behavior might leave a negative impression on a user impatiently waiting for a response.

To avoid making a customer wait in this way, we propose a mechanism for handling the problem of conflicting critical sections, which we call Proactive Timing Control (PTC). This mechanism enables interactions to be coordinated in order to prevent critical section conflicts from arising at all. One means of achieving this is for each robot to send a reservation request to the operator before a critical section begins. If the operator accepts, the robot can proceed to the critical section. Otherwise, the robot performs other behaviors in order to delay entry into the critical section.

This technique changes the robot's behavior in an important way from the customer's perspective. When PTC is not used, the delaying behaviors are executed after the user's "turn" in the conversation, that is, after the user has made a request or asked a question. There, the user is understood to have the initiative, and the robot is expected to react.

With PTC, however, the delaying behavior is executed before the user has spoken, while it is still the robot's "turn" in the conversation. The robot has not yet relinquished the initiative, and thus the extra behaviors integrate more smoothly into the flow of interaction. The effectiveness of this technique has been demonstrated in a study of the effects of wait time upon customer satisfaction [191].

4.4.4 Teleoperation Interface Design

Teleoperation interface software was developed to enable the operator to control one robot (referred to here as the "active" robot) while monitoring the others in the background. The interface used is pictured in Fig. 4.5. The four panels on the top left of the screen show the status of each robot, and the operator can click one to begin controlling that robot. Below those panels, the button panel on the left can be used to trigger robot behaviors. The column of buttons to the right of that can be used to correct speech recognition results, and the pop-up window on the right side shows a map of guide destinations from which the operator can trigger guide behaviors.

This interface is nearly identical to that used in [48], and further details of its functionality are explained there. Major differences from that interface include the addition of a map display of the robot's location (lower right), the addition of a panel showing video from the robot's eye camera (upper right) and the removal of the buttons for "reserving" a robot without switching to it, as this functionality is not particularly necessary unless PTC behaviors are very long.

4.4.5 Example Interaction

Here we will describe an example of a typical multi-robot control session from our experiment. In this example, the operator is controlling three robots, and the system is not using PTC, *i.e.* there is no attempt to prevent conflicts between robots demanding the operator's attention at the same time.

First, Robot 1 detects a person approaching. As it begins a greeting behavior, its Interaction Status light changes to yellow and the Countdown Timer on the robot's status panel begins counting down until the robot expects the human to speak.

The operator clicks on the robot's status panel to choose Robot 1 as the active robot, and the bottom half of the user interface refreshes to show Robot 1's current status, behavior history, and speech recognition results. The audio from Robot 1 is also streamed to the operator's headphones, and the operator listens as Robot 1 introduces itself, "My name is Robovie, and my job is giving directions. Where would you like to go?"

At this point, the operator notices that a person has approached Robot 3 as well. However, the operator stays focused on Robot 1, as its Countdown Timer is just reaching zero. The customer asks where to find an ATM. Unfortunately, due to background noise, Robot 1's speech recognition was unable to pick up the word "ATM", and so the operator goes to the expected phrases panel and clicks on "ATM". Robot 1 then begins giving directions to the customer, and the operator quickly switches to Robot 3, whose countdown timer has almost reached zero.

By this time, a customer has approached Robot 2 and begins asking directions while the operator is still busy helping Robot 3. Robot 2's Interaction Status light flashes red. By the time the operator finishes helping Robot 3, the customer talking to Robot 2 has already finished speaking. Robot 2's speech recognition system has picked up the word "hamburger", which is displayed on its Speech Results display, but the robot has no mapping between that word and a location in the mall. The operator quickly switches to Robot 2, opens the map, and clicks on a restaurant that specializes in hamburgers. Robot 2 then gives directions to that restaurant, as the Interaction Status indicators for Robots 1 and 3 return to green.

4.5 Experimental Validation

Preliminary experiments presented in [48] suggested that teleoperation of multiple social robots is possible and useful, and that PTC is a fundamental technique that can support it, but that work was only a preliminary study using internal subjects. We conducted a formal laboratory experiment with unbiased participants to evaluate the feasibility and effectiveness of our approach to teleoperation of multiple social robots, as well as the effectiveness of the Proactive Timing Control technique in particular.

4.5.1 Laboratory Experiment

Scenario

For this experiment, we chose route guidance as a realistic example of the kind of task a robot might be assigned to perform. It is easy to imagine a business such as a shopping mall, museum, or theme park placing a robot in a high-visibility location such as a central information booth. This task also lies in an interesting middle-ground between full predictability and open-endedness, and it provides a level of interactivity not found in primarily one-way interactions such as guiding visitors in a museum.

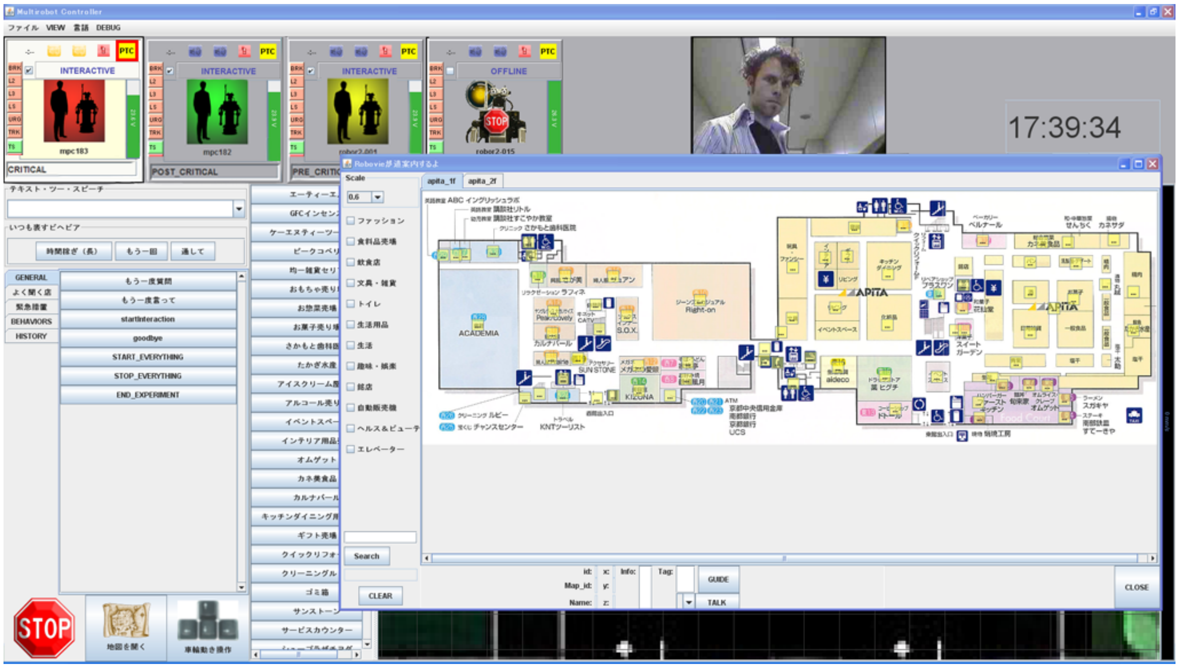


Figure 4.5: Teleoperation interface. Panels in the top left show the status of each robot, and can be clicked to select that robot. The video pane on the upper right shows the video feed from that robot's eye camera. The tabbed button panel on the left sends direct commands to the robot. The column of buttons to the right of that panel show expected utterances, allowing the operator to perform manual speech recognition for the robot. The pop-up map on the right allows the operator to select a location for commanding guide behaviors.

Experimental Design

The experiment was designed to evaluate performance of the operator-robot team while varying two factors. The first factor, *robot-number*, was examined in three levels: *2R*, *3R*, and *4R*, representing the number of robots being simultaneously controlled by the operator. The second factor, *PTC*, was examined in two levels: *with-PTC* and *without-PTC*.

The experiment was designed to evaluate two hypotheses. Our first hypothesis was that our system would improve performance compared with a purely autonomous system, regardless of whether or not PTC was used. To validate this hypothesis, we tested the performance of our system on an absolute scale, comparing *2R*, *3R*, and *4R* trials against two baseline cases: a single-robot case where the operator was always present, referred to as the *1R* condition, and a fully-autonomous case with no operator intervention, referred to as the *A* condition. This comparison was performed separately for the *with-PTC* and *without-PTC* configurations of our system.

We predicted that performance in the *with-PTC* condition should be comparable to that in the *1R* baseline, although performance in the *without-PTC* condition might be lower, particularly for large numbers of robots (*3R*, *4R*).

Our second hypothesis was that the use of PTC in particular would improve performance of the robot team relative to the *without-PTC* conditions, and that this improvement in performance would increase for larger numbers of robots. This was evaluated by a comparison between *with-PTC* and *without-PTC* conditions for each of the *2R*, *3R*, and *4R* cases.

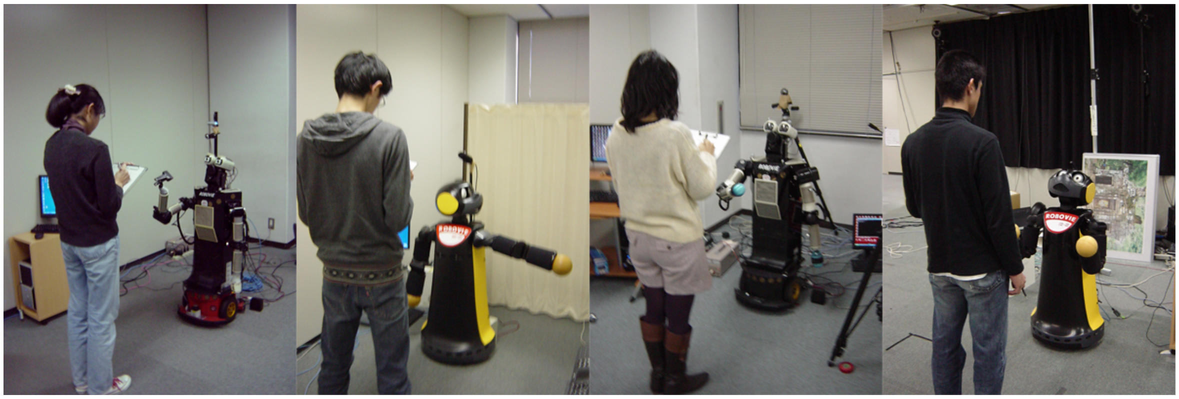


Figure 4.6: Four robots operated simultaneously in our experiment.

To test these two hypotheses, our experiment included a total of 8 conditions to be evaluated: *with-PTC* and *without-PTC* variations for each of the *2R*, *3R*, and *4R* cases, and the two baseline cases, for which the use of PTC is not relevant.

Setup

The behaviors and decision logic for the route guidance scenario were adapted from a recent deployment of our robots in a shopping mall. We used the interaction flow described in Section IV-A, with the chat behaviors (Phase #3 in Table 4.3) adapted for use as PTC behaviors.

The PTC behaviors consisted of interruptible sequences of short behaviors with an average duration of 4.4 seconds. After each behavior, the sequence could be interrupted or continued based on the presence or absence of an operator.

An example of such a sequence is the following: "Hi, I'm Robovie. / I know many things about this shopping mall. / This week the mall is having a special anniversary celebration. / There are many discount campaigns and exciting activities planned! / There is a 10% off sale in the clothing section. / And next Sunday there will be a classical music concert!" When an operator became available, the sequence could be interrupted after any of these utterances so the robot could begin the critical section by offering to give route guidance, and the entire sequence would flow in a fairly natural way. Four of these PTC sequences were prepared for the experiment, with a maximum possible length of 12 behaviors each, and one sequence was chosen at random for each interaction.

It is important to note that these behaviors were not merely time-killing behaviors. These chat behaviors had originally been part of the natural conversation flow. When the robot spoke about these topics in the field trial, they were relevant to the customers, who enjoyed their interactions with the robot.

The experiment was conducted in our laboratory, using two Robovie-II and two Robovie-R2 robots, as shown in Fig. 4.6. Each robot also had an automatic speech recognition (ASR) system, which operated in parallel with the operator.

Participants

16 paid participants played the role of customers in this experiment (12 male, 4 female, average age 22.3, SD=2.5 years). All were native Japanese speakers.

One expert operator, an assistant in our laboratory, was employed to control the robots for all trials. The operator was trained in the use of the control interface and thoroughly familiar with the map of guide destinations prior to the experiment, so we assume negligible improvement in operator performance across trials.

Procedure

To provide the operator with consistent task difficulty in the different experimental conditions, each trial consisted of 24 interactions in total, *i.e.,* 6 interactions per robot in the *4R* case, 8 in the *3R* case, 12 in the *2R* case, and 24 in the *1R* case. The *A* condition was conducted with four robots but no operator.

In these trials, one “interaction” included a greeting from the robot, possible chat behaviors for PTC, a question from the customer, and a response and farewell from the robot. Eight trials were run on each day of the experiment, one for each of the conditions (*2R-with, 2R-without, 3R-with, 3R-without, 4R-with, 4R-without, 1R,* and *A*).

On the customer side, 4 participants took part in every trial, and each participant interacted with the robots a minimum of 6 times per trial. Participants were assigned evenly across the robots. To achieve even distribution in the *3R* conditions, three participants interacted 6 times each with assigned robots, while one participant moved between the robots, performing two interactions with each. In other conditions, participants did not move between robots.

This experimental procedure was repeated on four days with a different group of 4 customer participants on each day, for a total of 16 participants acting as customers. The order of the eight trials on each day was counterbalanced with respect to both the *robot number* and *PTC* factors.

For consistency in timing, interactions were robot-initiated, with the robot inserting a pause of 0-5 seconds between interactions. To provide a consistent level of workload for the operator, participants continued interacting with the robots for the entire duration of each trial, going beyond the 6 evaluated interactions if necessary.

Evaluation

There is a causal chain of effects which we expect to produce different results between the *with-PTC* and *without-PTC* conditions. First, the use of PTC should increase the number of critical sections for which the operator is present. This should consequently increase the interaction success rate, because the speech recognition system is used less often. Finally, this improved success rate combined with reduced wait time in the critical section should improve customer satisfaction.

Accordingly, to evaluate the performance of the system, we measured three variables: the rate of operator supervision in the critical section, the overall ratio of successful interactions, and customer satisfaction on a scale of 1 (unsatisfied) to 7 (satisfied). Interaction success (whether the robot had successfully answered the question) and customer satisfaction were reported by participants after each interaction.

4.5.2 Experimental Results

The results of this experiment are illustrated in Fig. 4.7, showing operator supervision during the critical sections; Fig. 4.8, showing the interaction success rates; and Fig. 4.9, showing results from the customer satisfaction questionnaire.

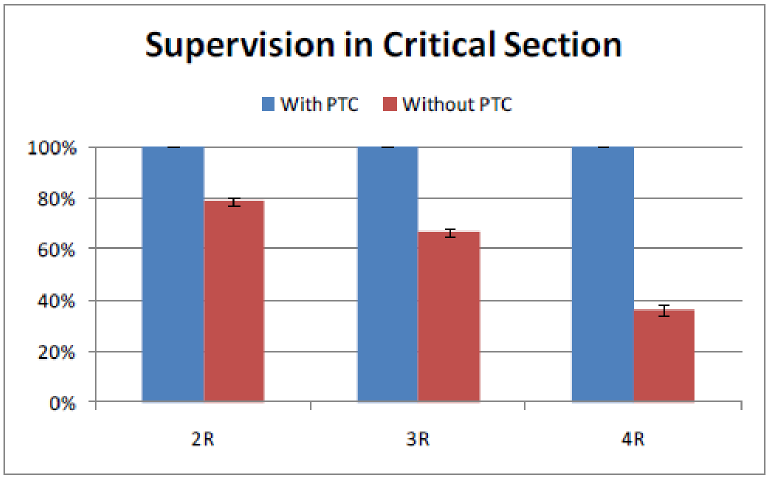


Figure 4.7: Operator supervision during critical sections.

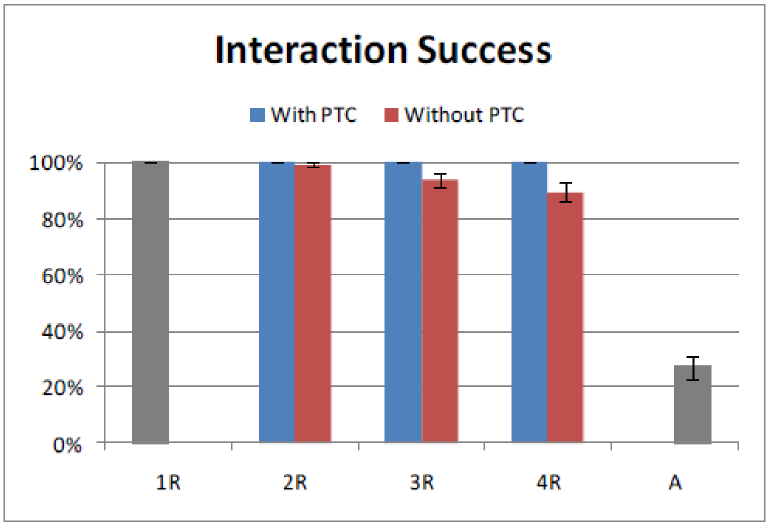


Figure 4.8: Interaction success rate. Error bars show standard error.

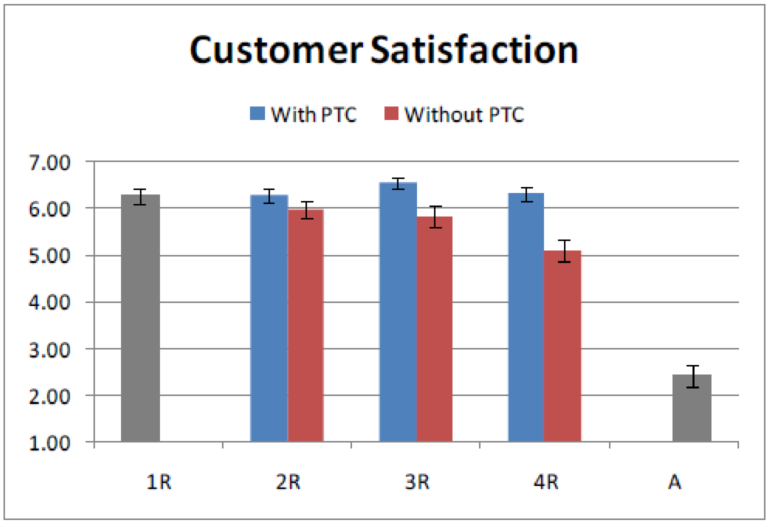


Figure 4.9: Customer satisfaction. Error bars show standard error.

Absolute comparison

To evaluate the absolute performance of the system between *with-PTC* and *without-PTC*, we examined each *PTC* condition separately, comparing the *2R*, *3R*, and *4R* levels of that condition with the *1R* and A baseline cases.

Operator supervision in critical section: Due to the use of PTC, the operator availability during critical sections was 100% for every trial in the *with-PTC* condition (Fig. 4.7). In the *without-PTC* condition, operator availability decreased markedly as the number of robots increased.

Interaction success: For the *with-PTC* conditions, 100% of the robot's responses were correct, which is to be expected as the operator was present for all interactions. In the *without-PTC* conditions, the interaction success rate decreased as the number of robots increased, up to a 10% failure rate in the *4R* condition.

In both conditions, there was a significant difference when compared with the autonomous case, which was successful only27% of the time (*with-PTC* condition: $\chi^2(4)= 327.805$, $p<.01$, residual analysis: *1R*, *2R*, *3R*, and *4R* to *A*: $p<.01$, *without-PTC* condition: $\chi^2(4)= 247.307$, $p<.01$, residual analysis: *1R*, *2R*, and *3R*, to *A*: $p<.01$, and *4R* to *A*: $p<.05$).

Customer satisfaction: For the *with-PTC* condition, customer satisfaction did not vary significantly between the *1R* – *4R* conditions. A repeated-measures ANOVA revealed a significant difference in the main effect of *robot number* ($F(4,15)= 189.786$, $p<.001$). A Bonferroni test revealed *1R*, *2R*, *3R*, and *4R* to be significantly better than A ($p<.001$), but no significant difference was found among *1R*, *2R*, *3R*, and *4R*.

For the *without-PTC* condition, customer satisfaction did not vary significantly between the *1R* – *3R* conditions, but decreased at *4R*. A repeated-measures ANOVA revealed a significant difference in the main effect of number of robots (*F(4,15)= 108.571, p<.001*). A Bonferroni test revealed that *1R*, *2R*, *3R*, and *4R* were significantly better than *A* (*p<.001*), and *1R* and *2R* were significantly better than *4R* (*p<.001 and p<.01*). The difference between *3R* and *4R* was approaching significance (*p=.077*). There were no significant differences among *1R*, *2R*, and *3R*.

These results confirm our hypothesis that performance in all teleoperated cases would be higher than the autonomous baseline. For the *4R* case, the significant decrease in customer satisfaction for the *without-PTC* condition also agrees with our prediction.

Relative comparison

To confirm the relative effect of PTC, we directly compared the customer satisfaction for *with-PTC* and *without-PTC* for each level of the number of robots. A paired t-test revealed significant differences for *3R* (*t*=4.442, *p<.001*), and *4R* (*t*=4.986, *p<.001*), and an almost-significant difference for *2R* (*t*=1.813, *p=.090*).

This result is consistent with our hypothesis that the use of PTC will improve performance, and that the performance improvement will be stronger for larger numbers of robots.

4.5.3 Operator Experience

During this experiment, the operator often remarked that she felt a high level of pressure and frustration during the trials without PTC, because she was aware that many robots were entering critical sections at the same time. She said she felt relaxed, and that the interactions seemed to go smoother when PTC was used.

4.6 Simulation

Our laboratory trials provided a practical demonstration of a single operator controlling multiple robots in conversational interactions. However, due to logistical limitations such as the number of robots available, it was not possible to evaluate our system with more than four robots, or to observe the effects of varying parameters such as CTR. We created a simulation based on the interactions observed in our experiment, in order to explore the dynamics of PTC and to make projections about the performance of our system under a variety of conditions.

4.6.1 Interaction Model

The interaction model used in the simulation represents each interaction as a sequence of phases, as shown in Table 4.4. The length of each phase is modeled as a normal distribution with mean and standard deviation calculated from the interactions conducted in our experiment.

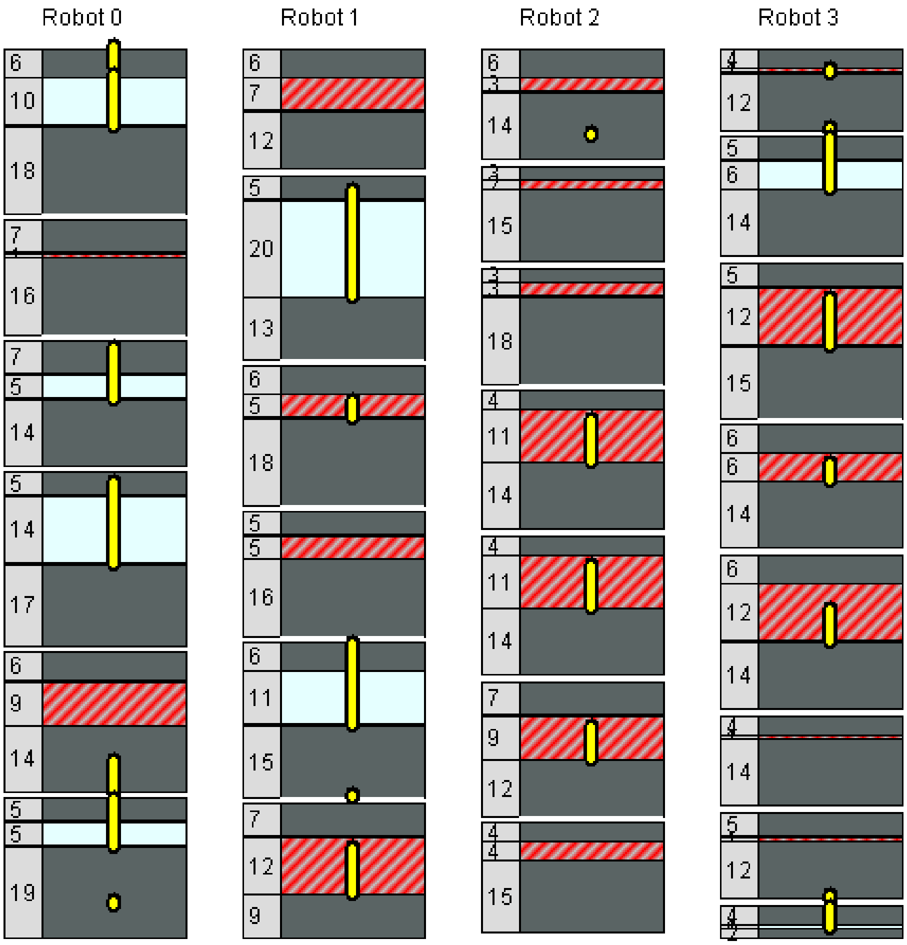


Figure 4.10: Examples of simulated interactions *without* Proactive Timing Control. Dark gray boxes represent non-critical interaction phases. Light-colored boxes represent attended critical sections, and diagonally shaded red boxes represent unattended critical sections. Numbers to the left of each phase indicate its duration in seconds. Vertical bars indicate which robot the operator is attending at any given time.

Interactions normally proceed in sequence through the Pre-Critical, Critical Section, Post-Critical, and Non-Interacting phases. If Proactive Timing Control is being used, then the system will transition to a PTC Behavior rather than a Critical Section if the operator is unavailable.

The simulator included an optional limit on the number of PTC behaviors, instructing the simulator to transition to the Critical Section when the operator becomes available, or after the maximum number of PTC behaviors have been executed.

4.6.2 Task Success

Task success is estimated by categorizing each Critical Section as attended or unattended. For our simulation, if an operator is present for an entire Critical Section, it is considered to be attended. If the operator is absent for any fraction of the critical section, it is considered to be unattended. Note that this method of counting is used because it is important to attend a critical section from the beginning in order to guarantee that the customer's question is heard in its entirety. If the operator is late, the speech recognition system may have already provided an incorrect response, or the operator may need to repeat the question.

In our experiment, the operator's accuracy rate during attended interactions was 100%, whereas the speech recognition system's success rate in the autonomous case was 27%. Our simulation thus assumes a response accuracy of 100% for attended interactions and 27% for unattended interactions.

4.6.3 Operator Allocation

The simulated operator is allocated to robots according to the following simple algorithm:

If the operator's current robot is in a critical section, do not switch to a new robot.

Otherwise, if any other robot is currently in a critical section, switch to the robot which has been in its critical section the longest.

Otherwise, switch to the robot for which the anticipated critical section begins soonest.

This algorithm is not necessarily guaranteed to be optimal, but it is roughly based on the way operators were observed to operate the system during testing.

Figures 4.10 and 4.11 illustrate typical interaction flows with and without Proactive Timing Control.

4.6.4 Patterns of operation

Figures 4.10 and 4.11 show how PTC dramatically reduces the number of unattended critical sections. The operator in Fig. 4.10 is only present for the beginning of 31% of critical sections, whereas the operator in Fig. 4.11 is present for 100%. These diagrams also show the dynamics of the system – at the beginning, when customer arrivals are nearly simultaneous, the operator requires long PTC behaviors to start the interleaving of interactions, but after

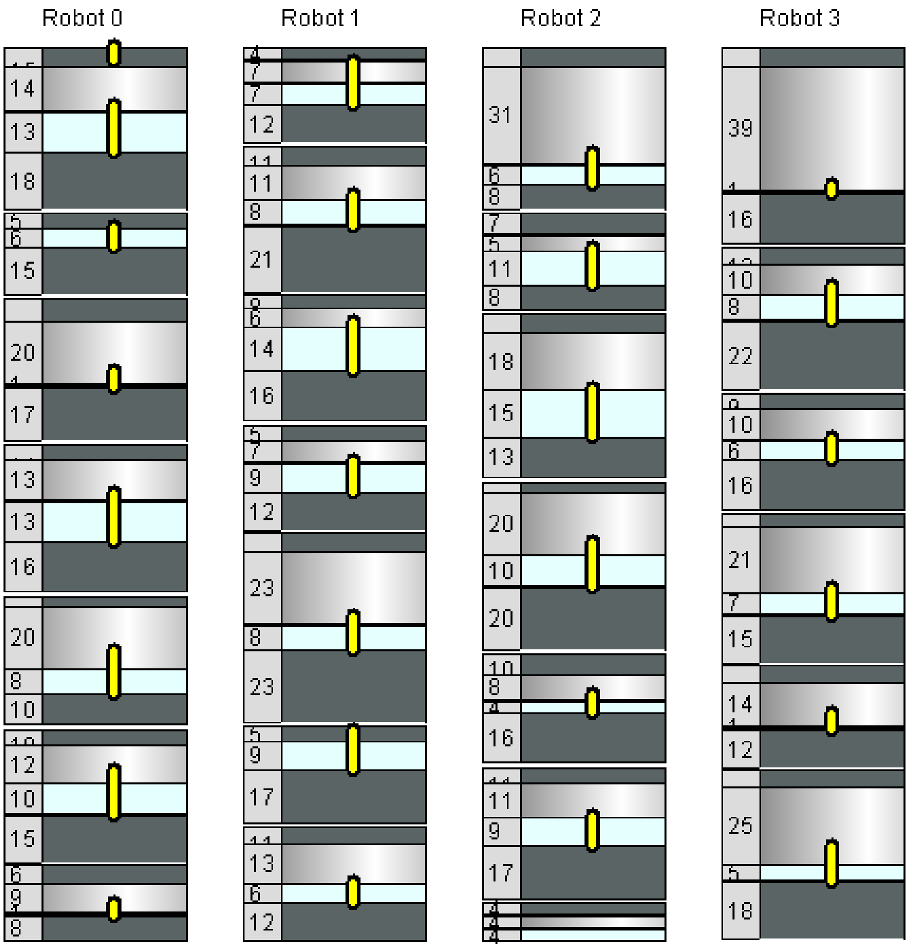


Figure 4.11: Examples of simulated interactions *with* Proactive Timing Control. Dark gray boxes represent non-critical interaction phases. Light-colored boxes represent attended critical sections, and boxes with metallic shading represent PTC delay behaviors. Numbers to the left of each phase indicate its duration in seconds. Vertical bars indicate which robot the operator is attending at any given time.

Interaction Phase	**Mean Duration (s)**	**Standard Deviation (s)**
Pre-Critical	4.9	1.1
PTC Behavior	4.4	1.7
Critical Section	6.3	5.0
Post-Critical	14.8	2.8
Non-Interacting	0.5	0.0

Table 4.4: Interaction Phases and Durations

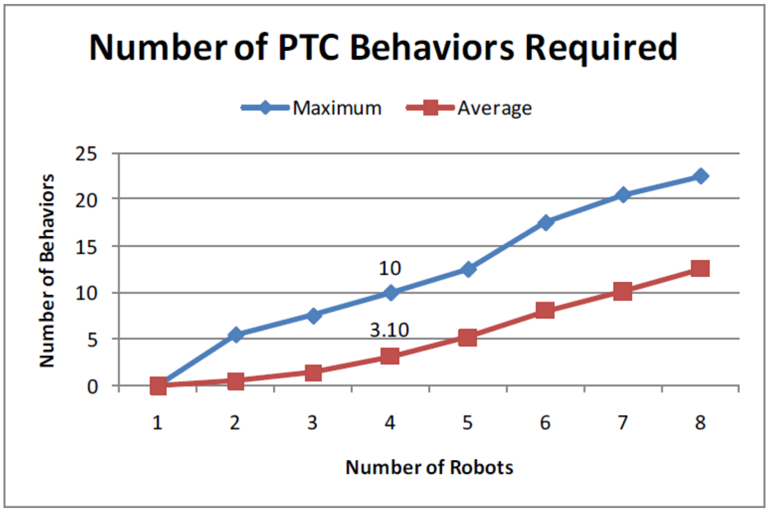


Figure 4.12: As the number of robots increases, more PTC behaviors are required to guarantee that an operator can attend all Critical Sections.

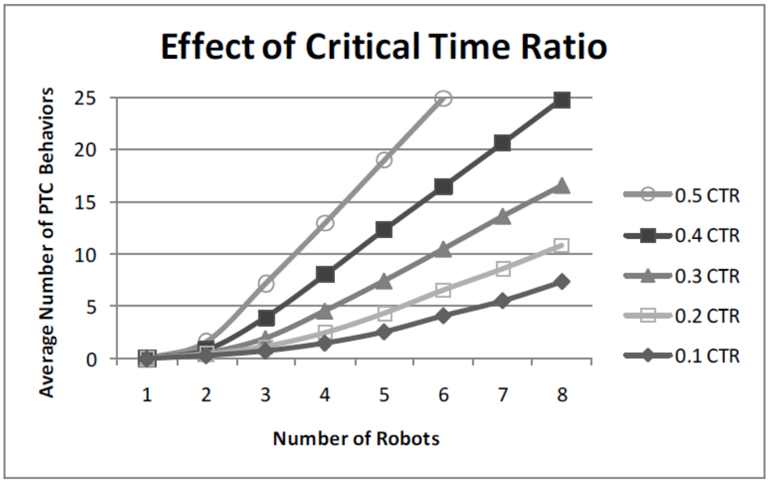


Figure 4.13: The average number of PTC behaviors required for a given number of robots increases as a function of Critical Time Ratio.

this point shorter PTC behaviors are sufficient to handle the random variation in interaction lengths. Such a pattern might be observed in a busy case where customers were waiting their turn to talk to the robots.

4.6.5 Number of PTC Behaviors

As the number of robots increases, more PTC behaviors will be required, and the average length of interactions will increase. We examined this trend using our simulation.

Figure 4.12 shows the maximum and average number of PTC behaviors used by our simulated system in runs of 1000 interactions using 1-8 robots. Here, one PTC behavior consists of a short utterance of around 4.4 seconds in length.

The results from this simulation agreed closely with our experimental results, as our operator used a maximum of 10 and an average of 3.9 PTC behaviors for the 4-robot case, compared with a maximum of 10 and average of 3.1 in the simulation.

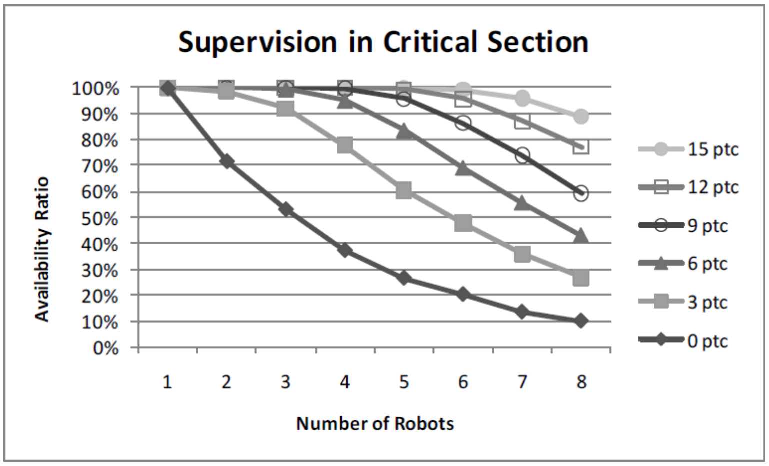


Figure 4.14: Variation in operator supervision during critical sections as maximum number of PTC behaviors varies.

As discussed in Section III, Critical Time Ratio (CTR) is determined by the design of an interaction. A highly interactive robot application would have long critical sections, and thus a high CTR, whereas a robot mostly performing fixed behaviors with less responsiveness to a customer would have a low CTR. Figure 4.13 shows the average number of PTC behaviors used in our simulations for interactions using a base CTR (not including PTC behaviors) ranging from 0.1 to 0.5. The figure illustrates how an interaction designer can balance the CTR of an interaction with the desired average PTC duration to target a given number of robots.

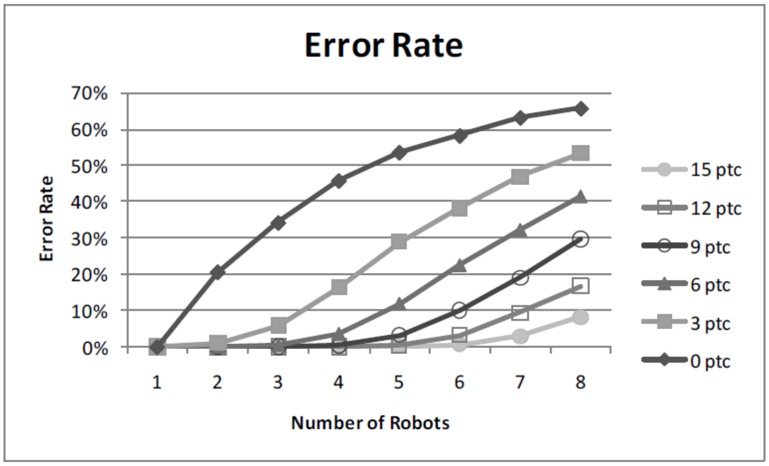


Figure 4.15: Change in error rate as maximum number of PTC behaviors varies.

4.6.6 Relying on Autonomy

The results so far assume an unlimited number of PTC behaviors and a target of perfect operator attendance during critical sections. However, the choice of how many PTC behaviors to use can be seen as a tradeoff between the desired level of response accuracy and its cost in terms of design difficulty and extended interaction time. Limiting the number of PTC behaviors causes the system to rely more on autonomy. Figures 4.14 and 4.15 show how system performance degrades when the number of PTC utterances is limited.

In these interactions, the limited number of PTC behaviors increases the number of unattended critical sections, and consequently increases the error rate due to failures caused by the autonomous system. For a route guidance application, errors are not acceptable, so the maximum number of PTC behaviors shown in Fig. 4.12 should be prepared. However, it is conceivable that some conversational robot applications might permit a small number of errors, and so the designer can make the trade-off between PTC duration and target error rate.

As the capabilities of recognition systems improve over time, it may be possible to rely more heavily on autonomy and thus achieve very high performance with minimal use of PTC.

4.7 Discussion

We were actually quite surprised by the positive results of the laboratory experiment and the operator's success in controlling four robots. Theoretical predictions notwithstanding, we had initially expected three robots in a real-world situation to be a challenge and four to be nearly impossible. However, the results from our experiment showed our approach to multi-robot control for conversational interactions to be much more effective than we had anticipated.

Here we will discuss several results from our experiment and how the principles can be generalized to other systems.

4.7.1 Maximum fan-out

As the simulation results illustrate, the maximum number of robots an operator can control depends on a variety of factors, including sensor reliability, critical time ratio, maximum number of PTC behaviors, and acceptable error rate. For the most difficult interaction settings in our experiment, the operator was successfully able to control four robots with 90% task success, and for the trials using PTC the operator was 100% successful in conducting all 288 interactions with no errors. Both of these results are dramatically superior to the low 27% success rate of the robots operating autonomously.

4.7.2 Defining Criticality

One conceptual model contributing to the success of our system was the division of interactions into critical and non-critical sections. It is fairly straightforward to apply this model to transactional interactions such as giving directions, particularly when a question is followed by a long explanation.

This model can be applied to many kinds of interactions, such as providing information, giving directions, and providing services requested by a customer. It can also be adapted for more complex interactions. For example, if a robot needs to ask a series of several questions, it may make sense to extend the critical section to encompass all of them in a single block. This may result in a small amount of wasted time for the operator while the robot is giving explanations or asking questions, but the operator is also guaranteed to be present for each of the follow-up questions, at a time where it may be awkward to insert delay behaviors.

In the general case, it will be important to consider both the risk of error and the cost of that error, both of which can be continuous variables. These subtleties may become more important in complex or long-term interactions; however, for the simple interactions in this study we will consider only two levels of criticality and model all failures as having equal cost.

4.7.3 Proactive Timing Control

From a system-level perspective, the Proactive Timing Control technique improves the operator's span-of-control in two distinct ways. To illustrate this, consider the critical sections of a robot's interactions to be like teeth in a gear, with noncritical sections represented by the gaps between the teeth. For an operator to control two robots, two gears must mesh, that is, the critical sections cannot overlap. The first way PTC achieves this is by synchronizing the gears – that is, holding one gear in place briefly while the other turns, until the critical section of one falls into the gap of the other. Adjustments like this are occasional and probably small. For example, with a hypothetical set of gears with perfectly regular spacing (*i.e.* when the lengths of the conversation phases are fixed) this adjustment would only be made once.

The second way PTC improves interleaving of tasks is by reducing the Critical Time Ratio, that is, by widening the gaps between the gear teeth overall. This is necessary when the time between critical sections is not sufficient to allow the gears to mesh during normal operation. This is a less desirable use of PTC, as delay behaviors must be executed for nearly every interaction. For behaviors such as those in our implementation, the content of the delay behaviors is generally not related to the context of the interaction. Thus, to create more natural interactions, it would be better to reduce the CTR at design time by extending or inserting behaviors relevant to the current interaction, rather than rely on PTC to make up for an insufficient gap between critical sections.

4.7.4 Limitations

The user study presented in this work was conducted in a laboratory environment with a pool of 16 customers performing repeated interactions with a robot. The results demonstrate that the proposed technique significantly improves performance, however, the use of PTC in real-world deployments of robots may have a stronger or weaker effect on customer satisfaction due to factors such as the novelty effect of the robots, customers' lack of familiarity with the robot's conversation style due to non-repeated interactions, quality and appropriateness of the robot's utterances, and variation based on the deployment context, *e.g.* whether people in that environment are in a relaxed or rushed mood.

Likewise, the simulation results are based on the user studies, so the results should not be necessarily seen as numerical predictors of customer satisfaction in the field. However, these results do serve to illustrate the dynamics of the system and the effects of varying different parameters, results that will be useful in designing and tuning systems for the real-world deployment.

4.8 Conclusions

In this study, we have presented a general framework for enabling the simultaneous teleoperation of multiple social robots, focusing on four key design areas: human-robot interaction design, autonomy design, multiple-robot coordination, and teleoperation interface design. While many key aspects of autonomy design and teleoperation interface design are similar to issues faced in other fields of robotics, the areas of human-robot interaction design and multi-robot coordination present many new issues which are unique to social robots.

Based on this conceptual framework, we implemented a robot system to demonstrate the new concept of a single operator controlling multiple robots in simultaneous social interactions. Our laboratory evaluations showed our system to be quite successful, with an operator achieving over 95% task success while controlling up to four robots in one experiment. These results demonstrate the value of our conceptual framework as well as the effectiveness of our specific solutions, such as Proactive Timing Control.

In our experiment, task success and customer satisfaction in every condition were far superior to those attainable by the same system operating in a fully-autonomous mode. Furthermore, our simulation results show that PTC reduces or eliminates conflicts between robots for an operator's attention. Even when PTC behaviors are limited, and the operator is forced to rely on automatic speech recognition some of the time, our simulation results indicate that PTC will provide a substantial increase in task success over a system with no timing control.

Most importantly, we have tested this system using an actual task often performed by our robots in the field, suggesting that this technology can be immediately put to use in real-world field trials. This study introduces the new field of teleoperation for multiple social robots, and several of the topics addressed in this paper are promising areas for further in-depth research.

Chapter 5

Temporal Awareness in Teleoperation

This chapter continues the discussion on teleoperation of conversational robots, focusing now on the teleoperator's perception of the passage of time. As discussed in Chapter 4, the real-time nature of conversational interactions imposes strict constraints on a robot's permitted response time. However, field trials using the proposed teleoperation system showed that operators sometimes had a distorted perception of time, believing that the responses they provided through the robot were quick and appropriate when, in reality, they were unacceptably slow.

This chapter presents an investigation of this phenomenon through a series of laboratory experiments. The experimental results confirm the problem of distorted time perception, and various user interface design strategies are compared for assisting the operator's awareness of the passage of time.

5.1 Introduction

Social robots operating in field environments face recognition challenges far beyond the abilities of today's autonomy, and some level of teleoperation is necessary to support conversation. We have found that the highly time-sensitive nature of conversation presents unique challenges in teleoperation, particularly regarding the awareness of time.

Consider these two anecdotal reports from a field trial we recently conducted, in which an operator simultaneously controlled the conversations of four robots conversing with customers in a shopping center (Fig. 5.1):

Operator: *The operator sat tensely in the control booth, watching the flashing robot status indicators. Gripping the mouse tightly, he scrambled to find destinations on maps and choose robot commands from menus, punctuating the intense silence with frustrated outbursts, "Gaaa! No! Wait!"*

Finally he emerged from the control booth, exhausted from the ordeal but yet grinning,

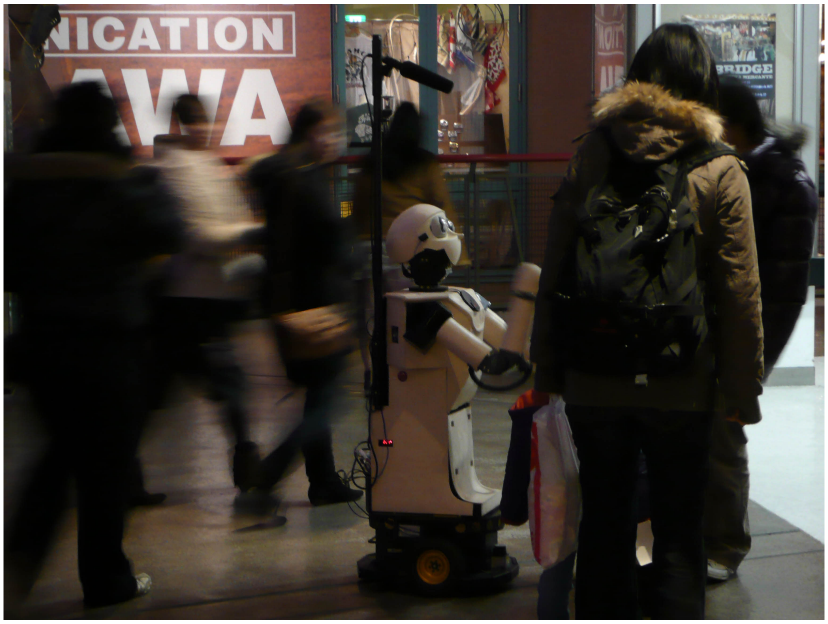


Figure 5.1: Robovie gives directions to customers in a busy shopping center.

like an athlete walking off the field after a hard-won victory. "I did it!" he exclaimed. "I think I might even be able to handle 5 robots!"

Customer: *We spoke with one of the customers after he had interacted with one of the robots. "It was very disappointing," he said. "The robot didn't seem to listen to me. I stood there for almost a minute before it finally answered my question."*

The magnitude of this disconnect was perplexing. The customer considered the interaction a failure, while the operator believed he had been successful. How could the operator fail to understand how long the customer had been waiting?

Situation awareness is an important focus of many studies in the field of robot teleoperation. It is common for robot operators deeply engaged in a task to develop "tunnel vision," a condition in which awareness is highly focused, and the operator loses the ability to monitor background information. In studies of robots for navigation, search, and manipulation, situation awareness is usually considered in terms of the perception of spatial phenomena [32]. Many studies measure the amount of time required to gain situation awareness, but few address the direct awareness of time itself, *i.e.* time estimation. However, we found awareness of time to be an important consideration for conversational robots. We believe that in our case, the operator's "tunnel vision" was focused on the immediate informational tasks, and that the operator consequently lost awareness of the customer's situation and the passage of time.

Robot operators have many tasks to perform. They may need to type in sentences, search for information, send commands to control gestures and speech, or use "conversation fillers"

[154] to stall for time while performing these tasks. With such a high workload, we have seen the operators of such systems lose awareness of the passage of time. Thus not only might an operator with a heavy workload make a customer wait for an excessive period of time, but in our experience the operator is often *not even aware of this fact!*

In this paper we address the phenomenon of temporal awareness loss in teleoperation of conversational robots. We show experimental results confirming that operators under high workload underestimate the passage of time during operation. We then propose two mechanisms for addressing the problem and evaluate them with respect to situation awareness, perceived workload, and overall effectiveness.

5.2 Related Work

In psychology and cognitive science, *time estimation* has been studied. In the context of teleoperation studies in HRI, *situation awareness* is considered to be important, but mainly about spatial situations. It has also been shown that *shared autonomy* can be a great help if well prepared. In social robots, the issue of *timing* has been found to be important. Here, we summarize literature in three domains: *time estimation, situation awareness and shared autonomy in HRI,* and *timing in social robots*, all of which come together in this study.

5.2.1 Time Estimation

Literature in psychology and cognitive science has revealed how people's sense of time varies. First, they have found that perception of short time, ranging from 30 ms to a few seconds (between 1 second [96] and 5 seconds [45]), and perception of long time are different problems. The former is called *time perception*, and the latter is called *time estimation*. In the context of our study, since each operation of conversational robot usually takes more than a few seconds, we are interested in *time estimation*. In addition, our study is concerned with the case where a person knows that they need to estimate time, which is categorized as *prospective* time estimation in the literature, in opposed to the case where a person is only asked afterwards, called *retrospective* time estimation [45].

For the *prospective time estimation* problem, the literature is in agreement that busy people estimate time as being shorter than the actual elapsed time. For instance, it was found that the passage of the time is estimated to be shorter when a person is engaged in a concurrent task in addition to the time estimation task, and when the concurrent task is interesting and complex [45]. Devoting more attention to non-temporal events and having a higher information processing load result in shorter time estimation [69]. Having greater demands on short-term-memory also results in shorter time estimation [41, 42, 40]. Researchers have started to integrate previous theories into a cognitive architecture [168].

5.2.2 Situation awareness and shared autonomy in HRI

In studies of teleoperation of robots for navigation and finding targets, the importance of *situation awareness* has been demonstrated [32]. Various methods have been developed to assist an operator's situation awareness, such as visualization of directions [75], maps [121, 120], and surrounding scenes [36].

While these studies address situation awareness for spatial information, to our knowledge few studies have addressed the awareness of the passage of time, that is, the problem of time estimation. In contrast, teleoperation of social robots is highly time-critical [49, 50]. This does not simply mean that an operator needs to make quick decisions; instead, the operator needs to make appropriate decisions based on time estimation. Note that previous studies considered the importance of time, but only as a metric, *e.g.* temporal demand [62], and efficiency measured by time [163], not as a problem of operator perception during operation.

In a study of *shared autonomy* (adjustable/sliding autonomy*)*, researchers have found that autonomy can help operators. For instance, autonomy was used for supporting navigation and manipulation by replaying scenes in the past [146], and for alerting about obstacles and helping with path planning [57]. Strategies for shared autonomy have also been studied. For instance, Hardin & Goodrich found that a mixed-initiative strategy performed better than adjustable and adaptive autonomy in search and rescue tasks [61].

5.2.3 Timing in social robots

When a customer is interacting with a teleoperated robot, the customer is engaged in a face-to-face interaction; however, the operator is engaged in information-management tasks using a graphical computer interface. Studies have shown that computer-mediated communication has different temporal qualities from face-to-face communication [68], suggesting that there may be an imbalance between the customer's temporal context and the operator's temporal context. This disconnect could prevent the operator from relying on an intuitive sense of the flow of time during the conversation.

Recent studies in social robots have started to highlight the importance of timing. In human communication, there is a pause during turn-taking [134, 107]. The length of the pause ranges from 620 to 770 ms [79]. In human-robot interaction, such natural pauses in human communication have been replicated [186]. Robins *et al.* explored how different response times change user reactions to a robot in a setting where a child and a robot are playing drums together [132]. It is reported that people sometimes prefer longer pauses, *e.g.* in the case where a robot is providing route directions, when people need to process information extensively [124].

One of the important related works is a study about *conversational fillers*. Shiwa *et al.* considered the problem of moderating people's negative feelings when a robot cannot make a quick response within a second. They demonstrated that such conversational fillers as "*etto*" can help a robot comfortably placate a user when it cannot respond immediately [154]. This technique has already been used in teleoperation of social robots in a field trial to moderate customers' frustration towards slow responses [85].

5.3 Problem Verification

In conversation, we rely on our time-estimation abilities and intuition to manage timing. If an operator has a distorted sense of the passage of time, it follows that we cannot rely on that operator's intuition to manage the timing of the interaction. Errors in time estimation can lead to awkward interactions, excessive wait times, inappropriate utterances, and a false perception of task success.

5.3.1 Experimental verification

We performed a laboratory experiment to verify whether this distortion of temporal awareness can be shown to occur in teleoperation of conversational robots.

As the literature suggests that having higher workload (*e.g.* information processing load or short-term-memory demand) results in shorter time estimation [45, 69, 41, 42, 40], we hypothesized that the operator would underestimate the amount of time that had passed, and that the magnitude of this error would increase with the operator's workload.

Experimental Setup

For this experiment, a computer functioning as a teleoperation console was placed in one room, and a robot was placed in another. In a camera shop scenario, participants controlled the robot to answer questions from an experimenter about different models of digital cameras.

12 undergraduate, native Japanese speakers (5 female and 7 male, average age 20.8, standard deviation 2.05 years) participated in this study, for which they were paid. None had any experience teleoperating our robots.

Robot For all of our experiments, Robovie II humanoid robots were used, as shown in Fig. 5.2. Robovie II is capable of humanlike expressions with a 3-DOF (degrees of freedom) head, 4-DOF arms, and 2-DOF eye cameras. It can gesture and perform speech synthesis according to commands sent from a teleoperation system, and it can stream video and audio to a remote operator.

Teleoperation interface The teleoperation interface used for this experiment was a Java application showing a video feed from the robot's camera at the top of the screen, and a control panel for the operator in the lower part of the screen. The control panel was very simple, with only two buttons available to the operator at any time.

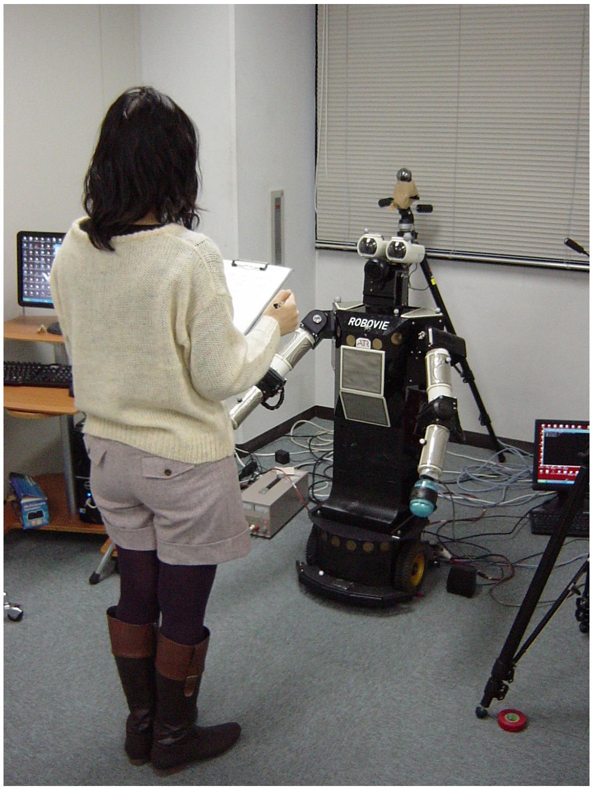
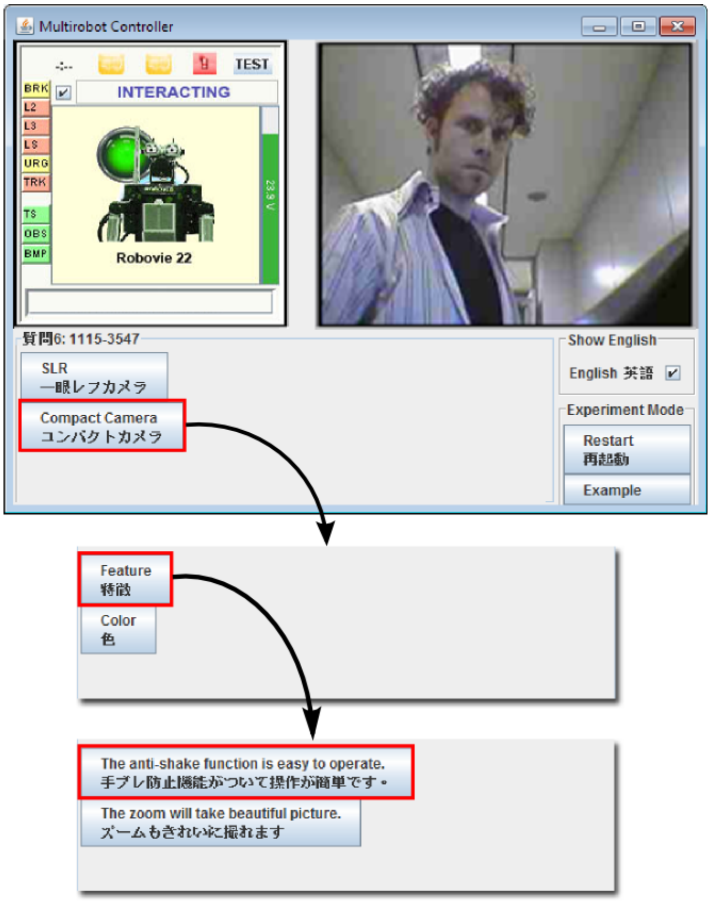


Figure 5.2: Robovie II, the communication robot used in our experiments.

Procedure

Each participant controlled the robot for six interactions, two for each of the three workload conditions (low, medium, and high). The order of these conditions was counterbalanced.

For each question, the operator was presented with a choice of two category buttons. After choosing one of the options, the operator was faced with another binary choice, continuing until the end of the tree, where the operator could choose one of two utterances for the robot to speak. This binary tree design enabled the workload of the task to be controlled precisely by adjusting the depth of the tree. In this way, we were able to create low, medium, and high workload conditions, using 1, 3, and 6 choices respectively. Fig. 5.3 shows an example of our interface. For workload consistency, operators were instructed to continue choosing the categories that seemed most appropriate, even after making a mistake.



質問6: 1115-3547

The anti-shake function is easy to operate.
手ブレ防止機能がついて操作が簡単です。

The zoom will take beautiful picture.
ズームもきれいに撮れます

Figure 5.3: Example of the "binary tree" interface to answer the question, "does this compact camera have shake reduction?"

After each interaction, the operator recorded an estimate of the absolute number of seconds which had elapsed between the asking of the question and the operator's response.

Results

As we had predicted, the participants underestimated the amount of elapsed time when workload was high. Fig. 5.4 compares the average operation time for each condition with the average time estimated by the operators. Note that each participant performed two tasks for each condition, so we took the average of two measurements for each condition.

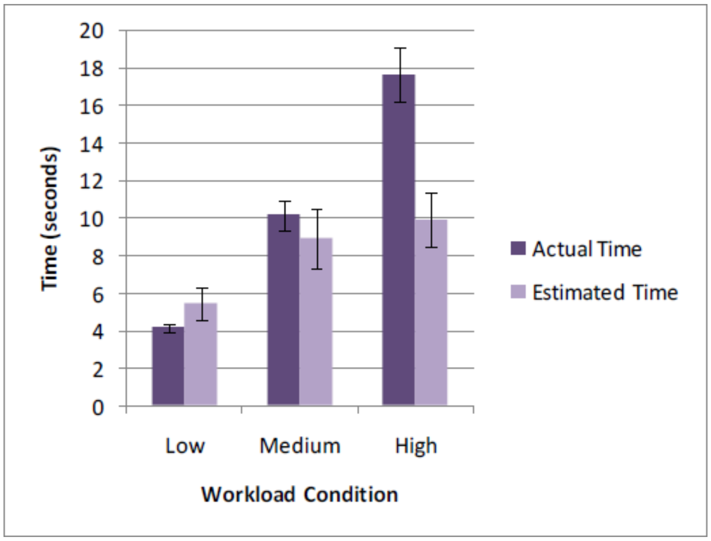


Figure 5.4: Comparison of operator time estimates with actual elapsed time for three workload conditions.

As these results show, the operators slightly *over*estimated the time by 1.2 seconds in the low-workload case, and they underestimated it by 1.3 seconds in the medium workload case, and by 7.7 seconds in the high-workload case.

For this time-estimation gap (*i.e.* real time minus estimated time), a repeated-measures ANOVA (Analysis of variance) was conducted with one within-subject factor, workload. The Huynh-Feldt ε correction was used to evaluate F ratios for repeated measures. A significant main effect was found (F(2, 26)=22.790, p<.001, ε=.772, partial η^2=.637).

A multiple comparison with the Bonferroni method was conducted for the workload factor, revealing significant differences among all pairs (p<.001 for comparison of the high-low pair, p<.01 for the medium-low pair, and p<.05 for the high-medium pair).

Discussion

These results clearly show the phenomenon with which we are concerned: operators tend to underestimate the passage of time in high-workload conditions, sometimes dramatically.

In a real conversational interaction, this phenomenon could result in an operator making a customer wait for an unreasonably long time, without even realizing how much time was passing.

The participants in this experiment were not experienced robot operators, and it is probable that time estimation could be improved through training. However, it is our hope that the problem can be addressed through user interface design to allow a wide range of operators to control robots without extensive and specialized training, and as our field trial experiences show, even expert operators experience this phenomenon to some degree.

5.4 Techniques For Assisting Teleoperation

Having verified the problem, we next examined the basic tasks necessary for conversational teleoperation, and we developed two techniques to help mitigate the problem of impaired temporal awareness.

5.4.1 Teleoperation Task

In real-world teleoperation situations, it is necessary to maintain a customer's attention when an operator is unable to respond quickly. For this purpose, we often use *conversational fillers*. These are interjections such as "hmm" which provide some feedback to the customer until the operator can provide a proper response. The study in [154] demonstrates that the appropriate use of conversational fillers improves customer satisfaction.

In our field trials, the operators manually actuate these conversational fillers in addition to operating the other controls in the interface. If an utterance will take a long time to type, the operator will click a button to start a filler before typing, with the goal of keeping the silence time low, *e.g.* below 5 seconds.

The operator is thus responsible for two main tasks with different temporal awareness requirements. The first is selecting or typing appropriate utterances. For this task, overall awareness of whether or not a customer has been made to wait too long may influence the operator's choice of utterances. Temporal awareness on this scale is our primary concern.

The second task is actuating conversational fillers when necessary. This task requires a more precise awareness of the amount of time that has passed, and task success is sensitive to small time estimation errors.

5.4.2 Assisting Operator Awareness

The first technique we evaluated was using a clock display to explicitly assist the temporal awareness of the operator. We chose to use a clock (shown in Fig. 5.5) which displays time through a rotating second hand and a digital display of seconds.

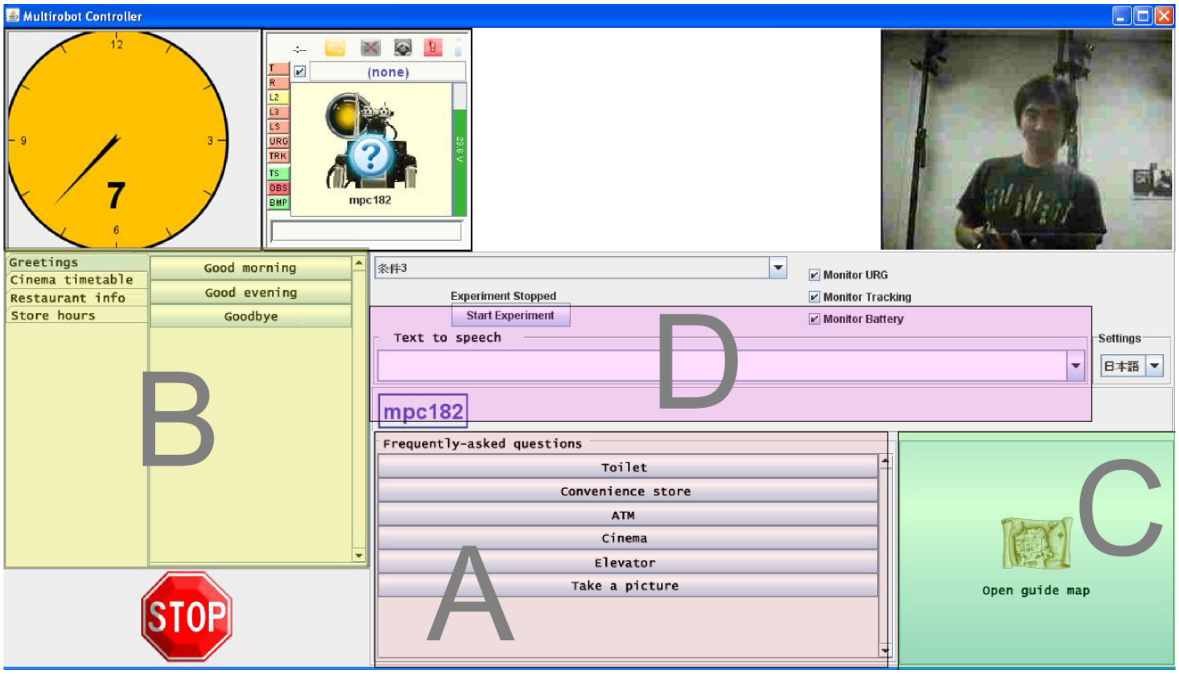


Figure 5.5: Screenshot of the teleoperation interface. The clock in the upper left was shown only for some conditions of the study presented in Sec. 5.6

Our hypothesis was that this mechanism would improve the operator's time estimation. We expected it would also help the operator use conversational fillers more effectively, which should reduce the number of long silences. However, as it does not change the operator's task, we predicted that it would not reduce the operator's workload or improve overall response time.

5.4.3 Automating Conversational Fillers

The second approach we evaluated was the automation of conversational fillers to simplify the operator's task.

Note that different kinds of conversational fillers are appropriate for short and long pauses, and a conversational filler should not be used if the operator is expected to respond quickly. We developed a simple model to predict the operator's response time, and used this prediction to make decisions about the timing and usage of conversational fillers.

Our hypothesis was that this would improve the timing of conversational fillers and reduce long silences, which is assumed to improve customer satisfaction. We also hypothesized that this mechanism would reduce the operator's workload and improve the operator's response time, as it simplifies the operator's task. We did not predict that it would necessarily improve the operator's estimation of time.

5.5 Estimating Operation Time

To model the amount of time required by the operator to respond to a question from the customer, we will consider the operator's *response time* to be the sum of the operator's *thinking time* and *actuation time*. For robots providing simple services, we assume that the majority of inquiries will be simple, factual questions, for which *thinking time* can be approximated as being constant.

Next, we model *actuation time* as being a function of the type of input task being performed by the operator, such as entering a phrase, or finding a place on a map. From our field trial experiences, we have observed that this is often the case, as text entry and map selection tasks take much longer than simply clicking a button or choosing an option from a menu.

While recognizing that many factors, such as training time and computer experience, can affect individual response times, we performed a study to generate a basic model to predict the amount of time required by college-age, first-time operators using our interface to respond to a set of predefined questions.

5.5.1 Objective

The objective of this study was to create a simple empirical model enabling us to predict operation times for an operator using our interface, based on the input task being performed. The four input tasks we investigated were as follows:

- Simple choice: Clicking a single button
- Categorized choice: Choosing an item from a tabbed menu
- Find a place: Choosing a location from a map
- Enter a phrase: Direct text entry via the keyboard

5.5.2 Setup

This study was based on a robot providing guidance and information services in a shopping mall. Each participant remotely operated the robot while one of our staff members, playing the part of a customer, asked the robot questions.

8 undergraduate, native Japanese speakers (3 female and 5 male, average age 22.5, standard deviation 1.85 years) participated in this study, for which they were paid. No participants had had prior experience operating our robots.

The teleoperation interface used for this study was based on an interface we developed for our field trials. Fig. 5.5 shows the graphical layout. The upper panel shows the robot's status and video from the robot's camera. The lower panel contains operator controls, featuring a fixed list of buttons in the center (area "A" in Fig. 5.5), a tabbed menu of buttons representing categorized behaviors on the left (area “B”), a button on the right for opening a map panel showing guide locations within the shopping mall (area “C”), and a text entry field at the top for directly entering text for the robot to speak (area “D”). The clock in the upper left was not shown during this study.

5.5.3 Procedure

Five sets of four questions (4 questions for training, and 16 questions for evaluation) were prepared, with each set including one question for each of the four input tasks. Responses to those questions were prepared for the interface.

The simple choices included answers to commonly-asked questions from our field trials, such as “where is the toilet?” and “may I take a picture?” Note that although giving directions to a location requires more than a simple utterance, it is still a closed-ended question for which a response including several gestures and utterances can be pre-programmed. Thus from an operator’s perspective, responding to the question “where is the toilet?” is as simple as responding to a yes-no question such as “may I take a picture?”

The categorized responses included movie start times, sorted by movie title; restaurant recommendations, sorted by restaurant type; and shop closing times, sorted by type of shop.

For the map-based tasks, we used guide maps taken from one of our field trials, modified to show only two floors of the shopping center, and eight shops on each floor.

Finally, the text field was prepared to simulate situations that are not covered [48], *i.e.* a predefined answer for that question has not been implemented in the robot. In our field trials, operators had the background knowledge to answer such questions. Since participants lacked such knowledge, we prepared a list of questions and answers which could not be answered using the buttons on the interface. Participants were instructed to use the text field in such cases. An example of one of these questions is, “What special events are happening this week?”

Every control in the interface was explained individually, including those inside the tabbed menus and every location on the map. The stock answers for the text entry questions were also presented. Each participant then operated the interface for four practice questions, one for each type of input task.

Participants then responded to the remaining 16 questions, asked in random order. The average response times recorded for each of the input tasks are shown in Fig. 5.6. Unsurprisingly, the results showed that the simple and categorized input tasks were much faster than the others, and that text entry was the slowest by far. Operator response time directly translates into customer wait time, so these values can help to predict how long a customer will be made to wait, as a function of the operator’s input task.

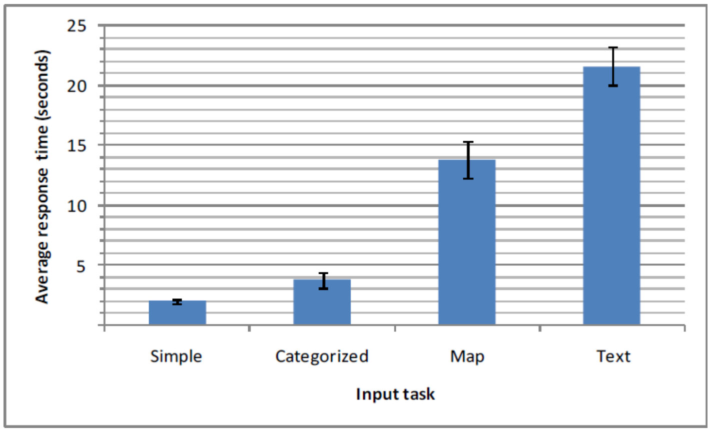


Figure 5.6: Average operator response times for four types of input tasks.

5.5.4 Application

This model enables us to develop an automatic mechanism for inserting conversational fillers in an appropriate way.

Conversational filler Strategies

We developed three strategies for generating conversational fillers based on the operator's estimated response time: "no filler", "short filler", and "long filler".

No filler: According to the findings in [154], it is important for the robot to respond in some way within about two seconds. If the operator can respond in that time, no filler is required.

Short filler: If the operator is expected to take slightly longer than two seconds, a short filler is necessary. Our system uses "*etto,*" a thinking sound similar to "hmm" in English.

Long filler: For response times longer than two seconds, a long filler may be more socially appropriate than simply repeating "*etto*" several times. For long fillers, our robot says different phrases, like "*chotto matte ne*" ("please wait a moment"). The robot then continues saying fillers every 4 seconds, to signal to the customer that it is still "thinking".

Applying the Model

By monitoring user interface events, we can identify which input task is being performed by the operator. If the mouse pointer is detected in the fixed button panel or the tabbed menu, we assume that the operator is searching for one of the fixed or categorized choices. A click on the map button or text box indicates that the operator will use the map or enter text.

Using these detected actions and the model created here, we can make a rough prediction of the operator's response time. The predicted response time can then be used to choose the conversational filler strategy, as shown in Table 5.1. The 7-second demarcation between the short and long filler strategies comes from the initial filler time (2 seconds), plus the time for the filler utterance (around 1 second), and 4 seconds of silence.

5.6 Experimental Comparison of Solutions

5.6.1 Experiment

A 2x2x4 within-participants factorial design was used to compare the effectiveness of these two proposed techniques. The first factor, *clock*, represents the use of the clock mechanism described in Sec. 5.4.2, in two levels: *clock* and *no clock*. The second factor, *filler*, represents the use of the automatic filler technique described in Sec. 5.4.3, in two levels: *auto-filler* and *manual-filler*. The third factor is the input task for each question, represented by the *input-task* factor in four levels: *simple*, *categorized*, *map*, and *text*.

Procedure

Participants operated a robot in a shopping mall scenario, using an interface like the one described in Sec. 5.5, but with the addition of a clock display and a conversation filler button.

23 undergraduate, native Japanese speakers (15 male and 8 female, average age 21.1, standard deviation 2.0 years) participated in our experiment, for which they were paid. None had participated in the other studies in this paper.

Instructions

The scenario was explained to the participants, and they were shown a demonstration of a simple interaction with the robot. The robot's response time and the importance of responding quickly were discussed, and the point was repeated several times throughout the task explanation.

Every control and map location on the interface, including the clock and conversation filler button, was explained. For the *manual-filler* conditions, the operators were instructed to manually insert conversational fillers using a button on the interface, first within 2 seconds of the end of the customer's question, and afterwards never to allow more than 5 seconds of silence. They were also told to be aware of their operation time, and to estimate it after each interaction.

A four-question training session was conducted for each operator, just as in the previous study. The same list of questions from the previous study was used for this experiment.

Trials

Predicted Response Time	Conversational Filler Strategy
less than 2 seconds	No filler
2-7 seconds	Short filler
greater than 7 seconds	Long filler

Table 5.1: Conversational filler strategies by response time

Each trial consisted of four questions, one for each of the input tasks, which were always asked in the order: *simple*, *text*, *map*, *categorized*. Note that while the customer was asking a question, the operator's screen controls were blanked, so even if an operator could anticipate the *input-type* for the next response, no pre-actuation was possible.

Four trials were conducted for each participant, one trial for each combination of *clock* and *filler* conditions. The order of *clock* and *filler* experimental conditions was counterbalanced between participants, and question sets were also counterbalanced between conditions, to ensure that results were independent of specific question content. Each participant answered each question only once.

Evaluation

After each interaction, the participants estimated the time it took them to respond to that question. Then, after each trial of four questions, the participants rated their workload for the trial. For this evaluation, we used the NASA-TLX scale (Task Load Index) [62], a tool for assessing subjective workload based on six factors: mental demand, physical demand, temporal demand, operator performance, frustration, and effort.

A total of four measurements were used in this study:

- **Operation time**, from the end of the customer's question until the operator sends a command
- **Time estimation error**, calculated by subtracting the estimated time from the actual operation time
- **Silence duration**, the maximum duration of silence between robot utterances during an interaction
- **Perceived workload**, the NASA-TLX score

5.6.2 Hypotheses

To restate the hypotheses from Sec. 5.4 in terms of the factors in this experiment, we predicted that the use of *auto-filler* would reduce *silence duration*, *perceived workload*, and *operation time*, with no effect on *time estimation error*. Furthermore, we predicted that the presence of the *clock* mechanism would reduce *time estimation error* and *silence duration*, but have no effect on *operation time* or *perceived workload*.

5.6.3 Results

The results for the four measurements are shown in Fig. 5.7. Full analysis is presented for all three factors (*clock, filler,* and *input-task*) for the measurements of “operation time,” “silence time,” and “time estimation error.” Regarding “perceived workload,” the NASA-TLX test was administered only once after each trial of four questions. As each trial contained all four input tasks, it was not possible to examine TLX scores for each input task separately. Hence perceived workload is analyzed here with respect to *clock* and *filler* only.

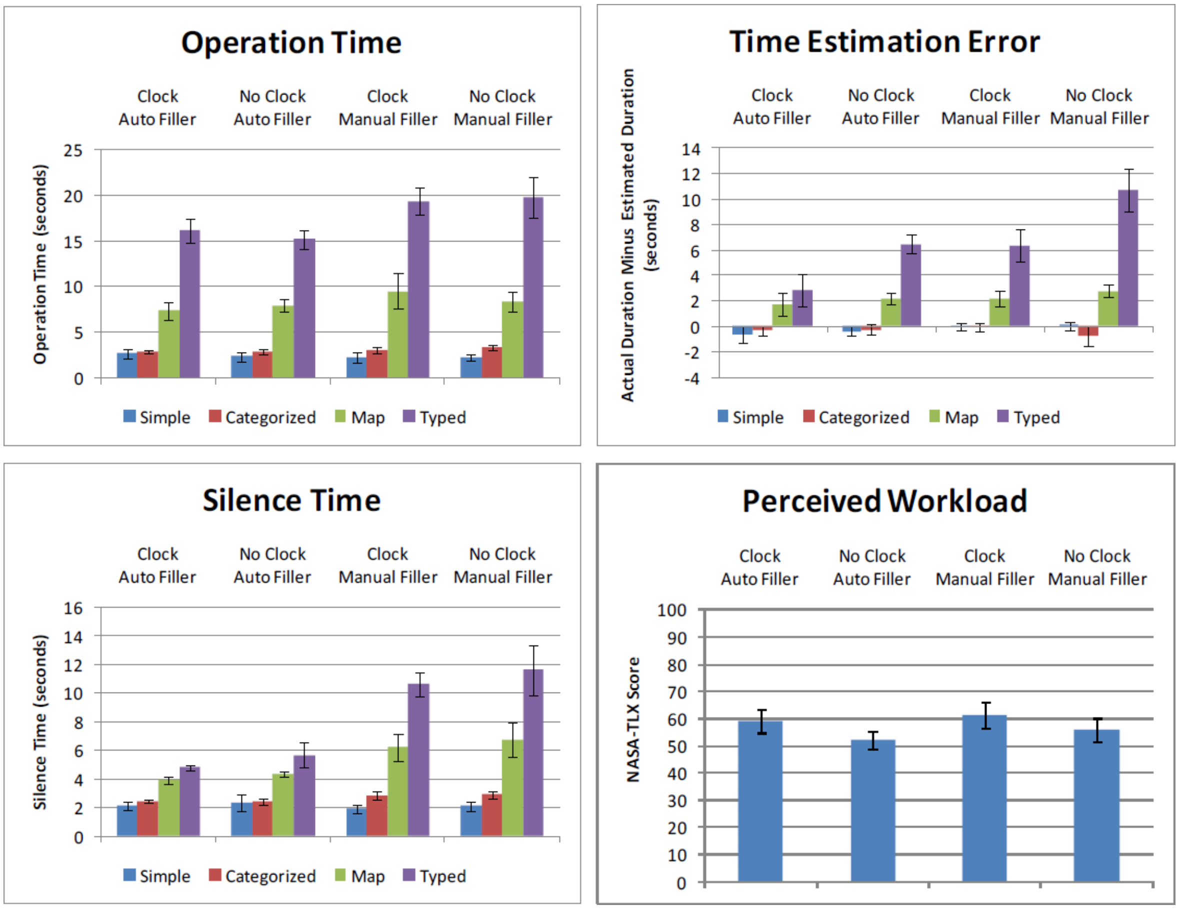


Figure 5.7: Results for the four variables measured in our experiment.

Operation time

For operation time, shown in Fig. 5.7 (upper left), a three-way repeated-measures ANOVA (Analysis of variance) was conducted with three within-subject factors: *clock*, *filler*, and *input-task*. The Huynh-Feldt ε correction was used to evaluate F ratios for repeated measures. A significant main effect was revealed in the *filler* factor (F(1,22)=13.279, p<.001, partial η^2=.376). No significance was found in the *clock* factor (F(1,22)=.069, p=.796, partial η^2=.003), or in the interaction between these factors (F(1,22)=.006, p=.937, partial η^2=.000).

For the *input-task* factor, the main effect (F(2.182,66)=121.429, p<.001, partial η^2=.847) and interaction with *filler* (F(1.984,66)=4.203, p=.022, partial η^2=.160) were significant, whereas the interaction with *clock* (F(2.316,66)=.049, p=.967, partial η^2=.002), and the interaction among the three factors (F(2.048,66)=.460, p=.639, partial η^2=.020) were not significant.

The interaction with filler indicates that the filler significantly reduced the operation time in *typed* input (p=.006), but the difference was not significant for the other input types: *simple* (p=.621)*, categorized* (p=.204), and *map* (p=.200).

The main effect and the interaction with *filler* also indicate that operation time varied for different input tasks, as we already discovered in Sec. 5.5. A multiple-comparison with the Bonferroni method was conducted for the four input tasks, which revealed significant differences in operation time as follows: for the *manual-filler* condition, *text* > *simple*, *categorized*, and *map* (p<.001), and *map* > *simple* and *categorized* (p<.001). There was no significant difference between *simple* and *categorized* (p=1.0). For the *auto-filler* condition, *text* > *simple*, *categorized*, and *map* (p<.001), *map* > *simple* (p<.001) and *categorized* (p=.001). There was no significant difference between *simple* and *categorized* (p=.119).

Predictions: *Auto-filler* will reduce operation time. *Clock* will not affect operation time.

Results: As predicted, the use of *auto-filler* reduced the time needed for operation for *text* but not significant for other input; *clock* did not contribute.

Time estimation error

For time estimation error, shown in Fig. 5.7 (upper right), a three-way repeated-measures ANOVA was conducted with three within-subject factors, *clock, filler,* and *input-task*. The Huynh-Feldt ε correction was used to evaluate F ratios for repeated measures. A significant main effect was revealed in the *filler* factor (F(1,22)= 13.415, p=.001, partial η^2=.379). The main effect in *clock* was significant (F(1,22)=19.740, p<.001, partial η^2=.473). No significance was found in the interaction within these factors (F(1,22)=.001, p=.977, partial η^2=.000).

Regarding the input-task factor, the main effect (F(1.868,66)=33.650, p<.001, partial η^2=.605) and the interaction with *clock* (F(1.559,66)=9.066, p=.002, partial η^2=.292), and the interaction with *filler* (F(2.384,66)=8.246, p<.001, partial η^2=.273) were significant, whereas the interaction among these three factors (F(1.775,66)=.223, p=.775, partial η^2=.010) was not significant.

We analyzed the interaction with the *clock* with the Bonferroni method, which revealed that in the *text* input the *clock* effect was significant (p<.001), but no significance was found in other inputs (for *simple*: p=.496, *map*: p=.418, and *categorized*: p=.218).

We analyzed the interaction with the *filler* with the Bonferroni method, which revealed that in the *text* input the *filler* effect was significant (p<.001), and almost significant in the *map* (p=.092), but no significance was found in other inputs (for *simple*: p=.359, and *categorized*: p=.781).

Predictions: *Auto-filler* will not affect time estimation error. *Clock* will reduce time estimation error.

Results: Surprisingly, *auto-filler* significantly reduced time estimation error in *text* entry; *clock* also had the effect of reducing time estimation error in the case of *text* entry.

Silence duration

For maximum duration of silence, shown in Fig. 5.7 (lower left), a three-way repeated-measures ANOVA was conducted with three within-subject factors, *clock, filler,* and *input-task*. The Huynh-Feldt ε correction was used to evaluate F ratios for repeated measures. A significant main effect was revealed in the *filler* factor (F(1,22)=18.991, p<.001, partial η^2=.463). No significance was found in the *clock* factor (F(1,22)=1.433, p=.244, partial η^2=.061) or in the interaction within these factors (F(1,22)=.011, p=.917, partial η^2=.001).

Regarding the input-task factor, the main effect (F(2.173,66)=54.385, p<.001, partial η^2=.712) and the interaction with *filler* (F(2.046,66)=15.199, p<.001, partial η^2=.409) were significant, whereas the interaction with *clock* (F(2.703,66)=.794, p=.491, partial η^2=.035), and the interaction among these three factors (F(3,66)=.006, p=.999, partial η^2=.000) were not significant. We analyzed this significant interaction with the Bonferroni method, which revealed that in the *manual-filler* conditions, max duration of silence was longer in *categorized* (p=.026)*, map* (p=.007)*,* and *text* (p<.001) input, but not for *simple* input (p=.607). Clearly, this is because *simple* input is fast enough not to require fillers, so use of *auto-filler* did not contribute to reduce max duration of silence for *simple* input.

Predictions: *Auto-filler* and *clock* will both reduce silence duration.

Results: As predicted, the use of *auto-filler* reduced the maximum silence duration, whereas interestingly, the *clock* did not affect the maximum duration of silence, even for the manual-filler condition (in fact, a separate ANOVA was conducted only for *manual-filler* conditions which did not show any significant difference).

Perceived workload

For the NASA-TLX scores, shown in Fig. 5.7 (lower right), a two-way repeated-measures ANOVA was conducted with two within-subject factors, *clock* and *filler*. A significant main effect was revealed in clock factor (F(1,22)=8.204, p=.009, partial η^2=.272). No significance was found in the *filler* factor (F(1,22)=.683, p=.418, partial η^2=.030) or in the interaction within these factors (F(1,22)=.042, p=.840, partial η^2=.002).

Predictions: *Auto-filler* will reduce perceived workload. *Clock* will not affect perceived workload.

Results: The presence of a *clock* increased the perceived workload, and using the *auto-filler* did not decrease the perceived workload as we had expected.

5.7 Discussion and Limitations

5.7.1 Summary and interpretations

The experiment results showed that when the clock was displayed, perceived workload increased. The effect on time estimation was not significant but showed a trend (p=.098) that the operator had better time estimation when the clock was shown. When the automatic filler mechanism was in use, total operation time decreased, and the length of the maximum silence interval decreased. The operator's time estimation also improved, as indicated by a decrease in estimation error.

Of the four input tasks, typing was generally the most time-consuming. The analysis of interaction with the input-task factor revealed that the clock was most helpful in time estimation for the typing tasks, and auto-filler was most effective in reducing max silence duration for the typing tasks.

These results raise some questions.

Why did the clock not help time estimation so much, while auto-filler showed a clear effect? A possible explanation is that the auto-filler simplified the operator's task, resulting in better time estimation. The literature confirms that time estimation is better in less complex situations [45, 69].

The operator's task is also more complex when the clock is visible, requiring the operator to process time information in addition to other tasks. This might explain the marginal results regarding time estimation.

Another possibility is that the robot's auto-filler behavior may have provided audible feedback to the operator, although this was not the intention of its design. This feedback may have helped the operators to estimate time, since it came at regular intervals. Furthermore, the fact that the feedback came from the auditory rather than visual channel may have decreased the operator's workload, as human factors research shows that using different sensory modalities for different tasks can improve cognitive processing efficiency [162].

Why, then, did auto-filler not reduce perceived workload, even though it actually simplified the operator's task, resulting in shorter operation time? One possibility is that, as the majority of the operator's time and attention was spent on the input tasks, those tasks more strongly influenced *perceived* workload than the manual-filler task did. Yet, the fact that both operation time and time estimation were improved by using auto-filler suggests that auto-filler may in fact reduce *actual* workload.

5.7.2 Are these findings too obvious? Not to our operators.

Interestingly, we received unsolicited complaints from two of the participants who claimed the automatic filler mechanism was frustrating, because they preferred to have complete control over the system. This seems to indicate that the operators did not always perceive a need for the automatic filler mechanism, and that its benefits are not so obvious. However, in this study the automatic filler mechanism was shown to perform much better than the operator in preventing long silences.

5.7.3 Generalizability and Limitations

These findings are specific to our teleoperation system, based on four input tasks. However, these are common operational tasks for conversational robots, so the findings may be applicable to many cases of teleoperation for social robots.

We are also interested in the teleoperation of multiple robots. These findings have not been tested in that scenario. We predict that the temporal awareness problem will be more extreme with multiple robots, since the operator's task is more complex. Our solution may thus be even more effective in that case, but this remains to be tested.

5.8 Clock Display Experiment

Based on these results, we can conclude that a clock should not be shown for the input mechanisms other than text entry, since the clock does not help time estimation in those cases but does increase perceived workload.

Text entry is an important case, however, as it requires longer actuation time than the other input methods, and thus time estimation errors carry a greater risk for excessive customer wait times. For text entry, showing a clock could be useful for reducing time estimation error. It is not clear whether it would increase perceived workload, however, as our first experiment did not measure TLX scores for separate input tasks.

We thus conducted a second experiment to focus on the effect of a clock display in text-entry tasks. Our hypothesis was that the presence of a clock during text entry would improve the operator's time estimation but might also increase perceived workload by creating a feeling of time pressure.

We also evaluated an interface design in which a clock was shown only after each text entry task was complete. Our hypothesis was that showing the elapsed time after each task would increase time estimation accuracy, whereas hiding the clock during operation would reduce time pressure and thus also reduce perceived workload.

Furthermore, preliminary studies showed that the effectiveness of the clock displays could be dependent upon the typing style of the operator. We observed that touch-typing operators who watched the screen while typing were more aware of time while a clock was being displayed than non-touch-typing operators who were looking at the keyboard. For this reason, we studied the effect of typing style as a factor in our experiment as well.

5.8.1 Conditions

For this experiment, we used a 2x3 between-participants factorial design with two factors: *typing style* and *clock type*.

The *typing style* factor was studied in two levels: *up-type*, meaning the operator looked at the screen while typing, and *down-type*, meaning that the operator looked at the keyboard some or all of the time while typing.

The *clock type* factor was studied in three levels: *no-clock*, *clock-during*, and *clock-after*. In the *no-clock* condition (NC), participants typed their answers to a customer's question into a text box, and no feedback was provided to them about the amount of elapsed time. In the *clock-during* condition (CD), a digital display of the number of elapsed seconds was provided on the screen while they typed the response. Finally, in the *clock-after* condition (CA), no clock was displayed while typing, but after typing was complete the display showed the total number of seconds elapsed.

Our hypotheses regarding *clock type* were as follows:

- The operator's time estimation will be improved when a clock is shown (*clock-during* and *clock-after* conditions).

- The operator's perceived workload will be higher for the *clock-during* condition than for the *clock-after* condition, because of the perceived time pressure.

- Operators will tend to type shorter utterances when a clock is present.

- Operators will tend to type faster in the *clock-during* and *clock-after* conditions.

- Regarding *typing style*, we made the following hypotheses:

- *Up-type* operators will have better time estimation in the

- *clock-during* condition, while *down-type* operators will have better time estimation in the *clock-after* condition.

- Better time estimation will also decrease response time and utterance length, and increase perceived workload.

5.8.2 Experimental Procedure

Scenario

The scenario we chose for this experiment was that of an operator controlling multiple information-providing robots answering questions at a university. A total of 53 paid participants, 23 female and 30 male, took part in this experiment. All were university students (average age 20.6, standard deviation 1.7 years) and all were native Japanese speakers.

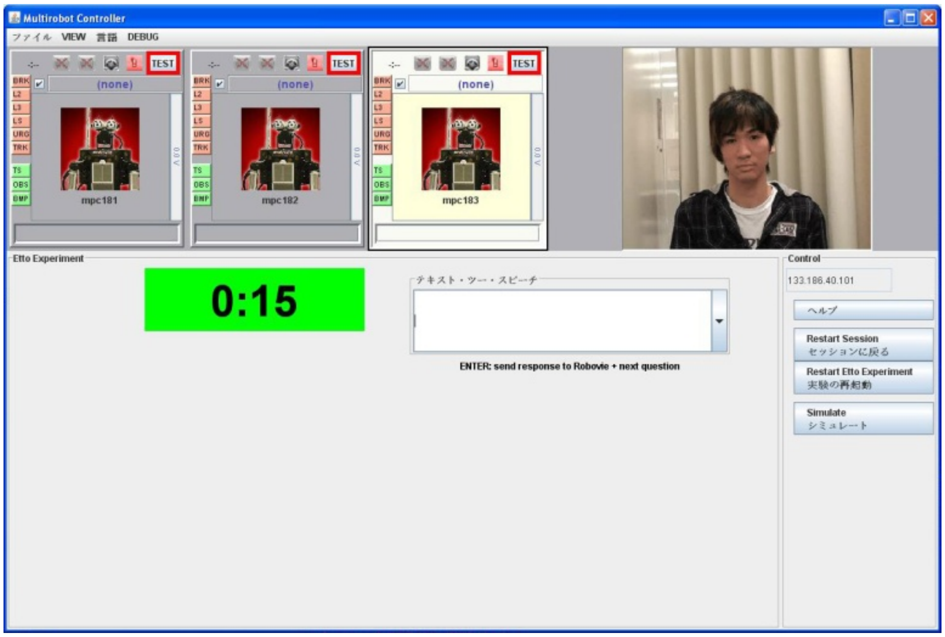


Figure 5.8: User interface for the clock experiment.

Participants performed the role of robot operator, using the interface shown in Fig. 5.8 to answer 15 simple questions about their university. They were told that the interface controlled multiple robots in other rooms, and that as soon as they had entered text for one robot to speak, the control would be switched to another robot. With this interface, they were instructed to answer the questions to the best of their ability based on their real experience.

Since the audio feedback from automatic conversation fillers could have been a confounding factor in the first experiment, we did not use them in the second experiment. However, as the auto-filler mechanism was shown to be useful, we assumed that such functionality would be present in a real teleoperation system and thus did not ask participants in the second experiment to enter manual fillers.

Questions

As reaction time was one variable of interest in this study, it was important to choose questions for which the participants would not have to look up information, but which they could answer from their background knowledge. For this reason we chose questions which we expected most students could answer about their universities, but which non-students might not know.

Three sets of 15 questions were prepared. In order to equalize difficulty between question sets, response times during preliminary trials were used to allocate questions of similar difficulty to each question set. Some examples of questions used include the following:

- Where should I go if I lose my student ID?
- How do I get to the nearest train station?
- Which courses should I take for easy A's?

For consistency of questions between trials, video and audio for all questions were recorded beforehand and then played back through the control interface during the experiments. The questions were recorded in different rooms, from the perspective of the robots' eye cameras. At least one minute of video was recorded after each question, showing the facial expressions and movements of the person waiting for the answer.

Procedure

Before the experiment, a typing speed test was administered to each participant. Their typing speed was recorded, and their touch-typing behavior was also observed and recorded. Participants were categorized as "up" if they looked up at the screen while typing, or as "down" if they looked down at the keyboard some or all of the time.

Within the *up-type* and *down-type* groups, participants were assigned to the different experimental conditions based on their typing speeds, with the goal of balancing typing speeds as much as possible across *clock type* conditions, as shown in Fig. 5.9.

The overall task was then explained to the participants. They were instructed to provide polite and complete answers to questions, but also not to make the customers wait too long. To help participants understand what a long pause would seem like to a customer, they were shown a video of a person asking questions to a robot three times. Each time, the robot paused for a different amount of time before responding. Pauses of 10, 20, and 40 seconds were shown, and the robot used conversation fillers during the pauses.

After watching the video, participants were instructed on how to use the interface, including an explanation of the clock, if one was shown. Each participant operated the interface in response to one practice question to confirm that they understood the procedure.

Participants then used the interface to answer 14 more questions in a row, all within the same *clock type* condition (*no-clock, clock-after, clock-during*). These questions measured the participants' overall performance within that condition.

Finally, participants filled out a NASA-TLX questionnaire, to evaluate their perceived workload for the task.

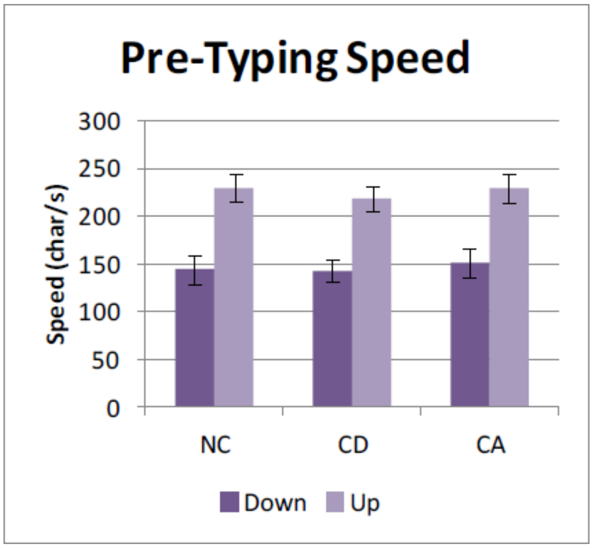


Figure 5.9: Average typing speed in pre-test.

Evaluation

In summary, the following data were also collected from each participant:

- Response time for each question
- Character length of response to each question
- NASA-TLX score for each 15-question session

5.8.3 Results

Time Estimation

For time estimation, we expected operators to underestimate the elapsed time in the *no-clock* (NC) condition, and to have more accurate estimation in the *clock-during* (CD) and *clock-after* (CA) conditions. We also expected *down-type* typists to have better estimation in the *clock-after* condition than in the *clock-during* condition, as they spent less time looking at the screen than the *up-type* typists did. Results are shown in Fig. 5.10.

The time estimates of the operators were typically shorter than the actual time durations, so in this section and in Fig. 5.10 we will express error as "actual duration minus estimated duration," so that large values represent large errors and small values represent more accurate estimation. Thus the expression "CA<NC" indicates that the time estimation in the *clock-after* condition was more accurate than in the *no-clock* condition (the error was smaller).

A two-way ANOVA with two between-subject factors, *clock type*, and *typing style*, was conducted for time estimation. A significant main effect was revealed in *clock type* (F(2,65)=

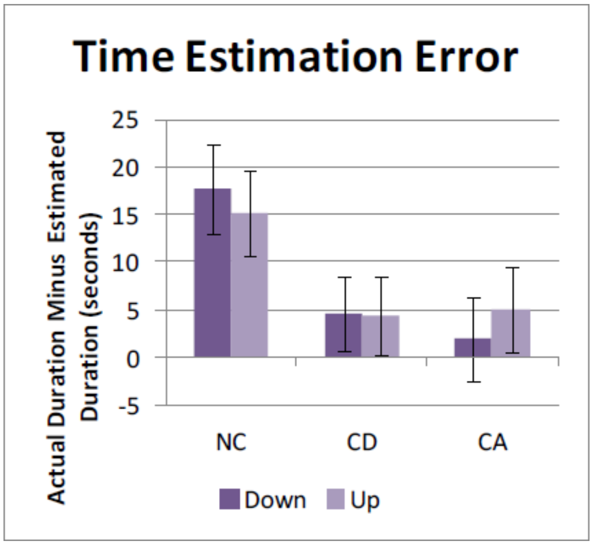


Figure 5.10: Time estimation error (actual duration minus estimated duration).

5.090, p=.009, partial η^2=.135). Multiple comparison with the Bonferroni method revealed that there were significant differences: CA<NC (p=.018), CD<NC (p=.023), but CA=CD (p=1.00). No significance was found in the *typing style* factor (F(1,65)=.001, p=.970, partial η^2=.000) or in the interaction within these factors (F(2,65)=.187, p=.830, partial η^2=.006).

These results support our hypothesis that the presentation of a clock results in better time estimation. Contrary to our expectations, however, they do not show a difference between *clock-during* and *clock-after* based on *typing style*. This may be due to the fact that *down-type* operators do look at the screen from time to time to confirm they have typed the correct phrase.

Perceived workload

Our expectation was that the presence of a clock would increase the operator's perceived time pressure and that this would be measurable by a NASA-TLX evaluation of perceived workload, shown in Fig. 5.11. Furthermore, we expected that *up-type* operators would perceive higher workload in the *clock-during* condition, whereas *down-type* operators would perceive higher workload in the *clock-after* condition.

A two-way ANOVA with two between-subject factors, *clock type* and *typing style*, was conducted for TLX score. There were no significance in the *clock type* factor (F(2,65)=1.417, p=.250, partial η^2=.042) or *typing style* factor (F(1,65)=.204, p=.653, partial η^2=.003), but the interaction within these factors was significant (F(2,65)= 4.023, p=.023, partial η^2=.110).

We analyzed this significant interaction with the Bonferroni method, which revealed that in the *up-type* condition there is an almost-significant difference between CA and CD (p=.074), but no significance found in other inputs (for up-type, CA-NC: p=1.000, CD-NC:

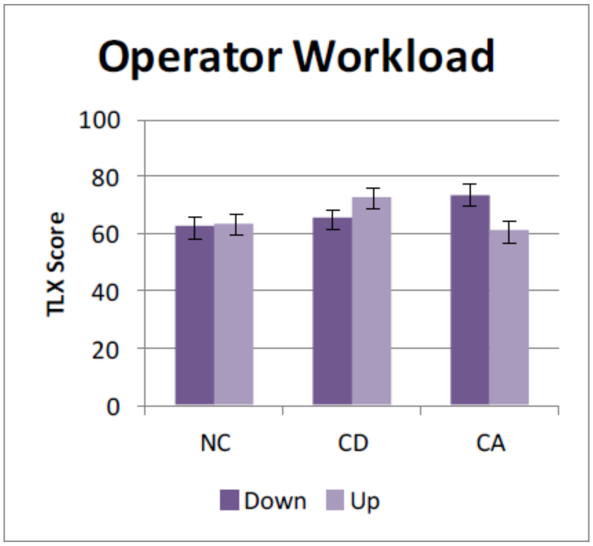


Figure 5.11: Perceived workload, as measured by NASA-TLX score.

p=.228; for down-type, CA-NC: *p*=.140, CA-CD: *p*=.299, CD-NC: *p*=1.000).

These results suggest that for *up-type* operators, the *clock-after* method may be better in terms of reducing the operator's perceived workload. Note that for most applications, it is likely that touch-typists will be employed as operators.

Response Time and Character Length

We expected that both response time and character length would be lower for the *clock-during* and *clock-after* conditions, compared with *no-clock*, and that both would be lower for *up-type* operation than for *down-type*. The results for these two measurements are shown in Fig. 5.12.

For response time, we conducted a two-way ANOVA with two between-subject factors, *clock type* and *typing style*. No significance was found in the *clock type* factor (F(2,65)=.109, *p*=.897, partial η^2=.003), in the *typing style* factor (F(1,65)=.088, *p*=.768, partial η^2=.001) or in the interaction within these factors (F(2,65)=.258, *p*=.773, partial η^2=.008).

For character length, we also conducted a two-way ANOVA with two between-subject factors, *clock type* and *typing style*. In this case, a significant main effect was revealed in the *typing style* factor (F(1,65)= 14.423, *p*=.000, partial η^2=.182). No significance was found in the *clock type* factor (F(2,65)=.292, *p*=.748, partial η^2=.009) or in the interaction within these factors (F(2,65)=.462, *p*=.632, partial η^2=.014).

Interestingly, these results do not show any significant difference in response time or character length based on *clock-type*, contrary to our expectations.

The results do show that although response time did not vary with *typing style*, the *up-type* typists provided longer (and ostensibly better) responses. We attribute this to the fact

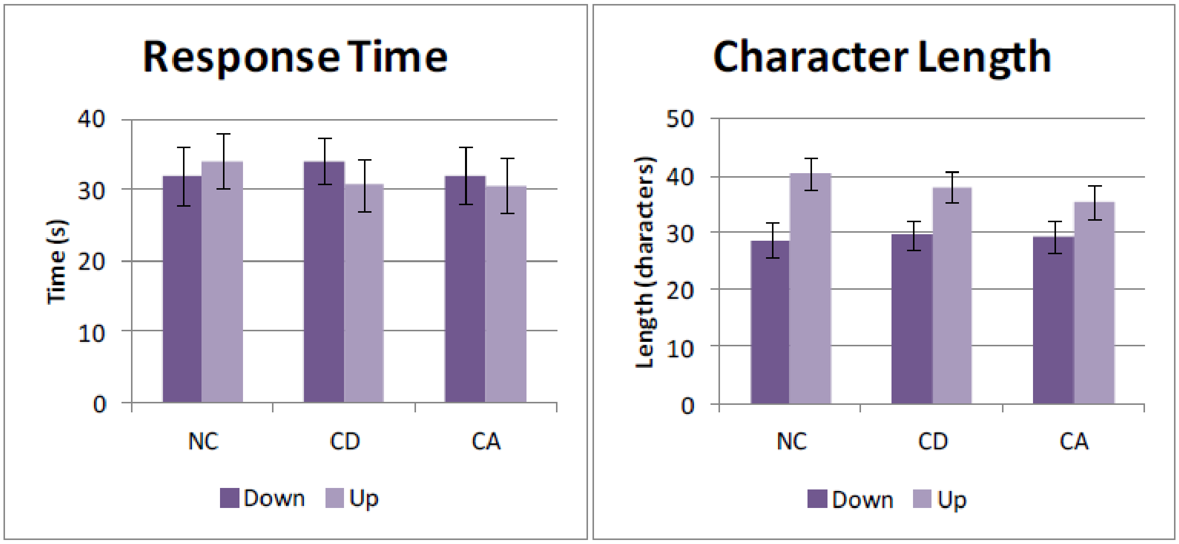


Figure 5.12: Average response time, in seconds, and average character length of responses, in Japanese characters.

that the *up-type* operators had a higher average typing speed.

5.8.4 Discussion of Results

In our previous experiment, we observed a trade-off, in which the display of a clock improved the operator's time estimation, but at the cost of an increase in perceived workload.

In this experiment, we observed that the clock display again improved the operator's time estimation, but this time the effect on perceived workload was not as evident. As this experiment only examined text entry, it is possible that the observed effect in the previous experiment mainly occurred in the other, faster, input methods.

Our results suggested that, for the case of *up-type* operators, showing the clock after the entry of each utterance (*clock-after*) resulted in lower perceived workload than showing the clock throughout the task of text entry (*clock-during*). This trend did not show strong significance (*p*=.074), but the results suggest that the *clock-after* technique might be useful for improving the time estimation of touch-typing operators without increasing their perceived workload.

We did not see a direct association between time estimation and operator performance results, either in silence time in the first experiment, or in response time in the second experiment.

We interpret this data by considering the relationship between *time estimation* and *time pressure*. The average response time was around 32 seconds. Even underestimating this time by 15 seconds, an operator would still believe that it took 17 seconds to type the response. Yet in normal conversation, a person would respond to this question far more quickly, probably within 2 or 3 seconds. It is possible that there is little difference in the *time pressure* an operator feels after 17 seconds or 27 seconds, as both of these times are far beyond what could be considered a "normal" human response time.

That would explain why little difference in response times is visible between *clock type* conditions. However, if the operators really do feel the same amount of time pressure, then why would there be a difference in character length of the operators' responses between *typing style* conditions? We believe that this could be due to touch-typists being more fluent with keyboard entry and thus accustomed to entering longer text. If slower typists are less familiar with using a keyboard, they might naturally enter shorter, less complete answers. Thus, in applications where the quality of an answer is important, we might expect faster typists to provide *better* answers than slower typists, not simply the same answers in a shorter time, although response quality was not evaluated in this study.

5.9 Discussion

Temporal awareness is important in teleoperation of conversational robots both in an immediate sense, because people have a low tolerance for long pauses in conversation, and in an overall sense, because understanding how long a customer has been waiting is important in choosing what to say. Thus impaired temporal awareness affects both utterance timing and the content of the conversation itself.

5.9.1 Partial Autonomy

In this study, partial autonomy was used to help simplify the operator's task. As technology progresses, more autonomy will become feasible. Will progress in such a direction eliminate the problem of temporal awareness? We believe it will not.

In future systems, we assume that many aspects of dialog management such as turn-taking [170] will be automated. Simple control tasks will be handled autonomously, and an operator will be responsible for handling complex, exceptional tasks that cannot be automated. As this autonomy improves over time, one operator will be able to control more and more robots.

Thus in future systems, we expect the operator's tasks to be more complex and less routine. The operator may spend less time performing direct control of minor utterances, focusing instead on high-level decisions and complex utterances. In this sense, temporal awareness will become less important in terms of immediate utterance timing, and more important in terms of choosing appropriate things for the robot to say.

5.9.2 Interaction Asymmetries

As mentioned earlier, one possible reason for the operator's poor temporal awareness is the asymmetry of the interaction, and reducing this asymmetry could help moderate the temporal awareness problem. There are two parts to be considered in this asymmetry: the task and the modality.

In terms of the task, the operator is entering data or looking up information in a map or a database, while the customer is asking a robot for information. In future systems, as the operator's task complexity increases, we expect that the task asymmetry will also increase. Both the increased complexity and the increased asymmetry may contribute to impaired temporal awareness.

In terms of the modality, the operator is interacting with a graphical computer interface, while the customer is face-to-face with a physical robot. To reduce the severity of this asymmetry, an immersive telepresence approach might help. Combining natural gesture control, as in [104] and [101], with an immersive first-person video feed [105] could reduce this asymmetry and provide the operator with a more natural sense of participating in a face-to-face interaction.

5.9.3 Limitations of this study

Customer experience

The experience of the customer interacting with the robot was not analyzed here, and the operator's performance was only examined numerically. The significance of an operator's temporal awareness as it affects the overall customer's experience is not easy to measure directly, and the importance of appropriate timing might be dependent upon the conversational context or other social factors. Considering the customer as a human element, there might be social ways to mitigate the sensitivity of customers to wait time.

Interaction complexity

Another limitation is that the interactions used in this study were simple question-and-answer exchanges. While other dialogue patterns are certainly possible, we believe that question-and-answer interactions will be quite common, particularly in the service robot domain, where interactive robots will often be providing information to people.

Another point that must be considered is that many human-robot interactions will likely extend beyond "single-round" exchanges. Our current study addresses the problem where an operator performs one input task per interaction; however, as shown in the introduction, the temporal awareness problem becomes more serious when an operator is continuously busy with many tasks. We think it likely that an operator's small judgment errors due to inaccurate temporal awareness may accumulate over several rounds of a conversation to cause significant frustration to a customer.

Multiple Robot Control

While we have seen that temporal awareness is an issue even when controlling one robot, the original problem presented in the introduction was a case of multiple-robot control, which presents new challenges. Multitasking in general has been shown to impair temporal awareness. Additionally, to enable an operator to focus on one conversation at a time, auditory information from other robots would need to be selectively muted. In such a case, the operator would be even less aware of wait time for robots that were not currently the focus of attention, and explicit mechanisms might be necessary for communicating this wait time.

Social Feedback

Another interesting issue that was raised during the final study in this paper was the effect of the operator seeing video of the customer. While some participants ignored the video feed while typing their responses, others indicated that they felt pressured by seeing the facial expressions of the impatient customer. If such social cues can be transmitted effectively, then it is possible that the operator's temporal context might more closely approximate that of a face-to-face conversation.

5.10 Conclusions

In this study, we have empirically demonstrated that the time estimation ability of operators controlling conversational robots can be impaired under high workload conditions. We have also conducted a comparison of two approaches to addressing this problem: by providing temporal information explicitly through a clock display, and by using autonomy to reduce the operator's task load. The results showed that the clock display alone did not significantly improve performance, but that it did increase the operator's perceived workload. The partial autonomy resulted in better performance as well as improved temporal awareness, without significantly affecting perceived workload.

Next, we examined the effectiveness of the clock display for text entry in particular, and found that while the clock displays significantly improved time estimation, we did not see a significant influence on the length of typed responses. The results also showed an almost-significant trend among touch-typists in which showing a clock after the finish of each operation resulted in a lower perceived workload than showing a clock throughout operation, although these two conditions yielded the same improvement in temporal awareness.

An interesting conclusion of this study is that indirectly supporting temporal awareness by simplifying an operator's task may be better in some cases than direct support, as our first experiment found perceived workload to be lower when the clock was not visible. This suggests that if better awareness can be achieved by reducing the operator's task complexity, then withholding information from the operator might be beneficial.

Finally, these findings are complemented by the technical contribution of our successful implementation of an automatic filler mechanism. Our simple approach of inferring the input task from mouse movements worked well for the tasks in this study, in that it limited silence time much more effectively than manual control. This technique was not always successful, however, and the operation task was not predicted accurately every time. For higher accuracy, it may be possible to incorporate information from interaction context or history to predict the operation task, and to extend the timing model to incorporate thinking time as well as actuation time.

Chapter 6

Interaction Design Framework

This chapter focuses on the robot itself, presenting design framework which enables the development and maintenance of interactive applications in a social robot by cross-disciplinary teams of programmers and interaction designers. By combining a modular back-end software architecture with an easy-to-use graphical interface for developing interaction sequences, this system enables programmers and designers to work in parallel to develop robot applications and tune the subtle details of social behaviors. This chapter describes the structure of the design framework and presents an experimental evaluation of the system showing that it increases the effectiveness of programmer-designer teams developing social robot applications.

6.1 Introduction

The field of social robotics is still young, and although much research has focused on details of creating humanlike interactions for social robots, little attention so far has been paid to the development process itself, which is usually performed by programmers. However, this is really a cross-disciplinary process integrating technical knowledge of hardware and software, psychological knowledge of interaction dynamics, and domain-specific knowledge of the target application.

The development of social robot applications faces not only the conventional challenges of robotics, such as robot localization and motion planning, but also new challenges unique to social robots, including new kinds of sensory-information processing, dialog management, and the application of empirical design knowledge in interaction. Examples of this design knowledge include maintaining acceptable interpersonal distance [182], approaching people from a non-frontal direction [27], and controlling the duration and frequency of eye contact [117], all of which have been shown to be important for social robots.

Applications developed in a research context are usually small-scale and engineered by small groups of highly-capable individuals. However, scaling this process to the level of real-world commercial deployment requires a collaborative design process involving people with different areas of expertise.

For example, algorithms and software modules are often developed for information-processing tasks like human tracking, social group detection, gesture recognition, prediction of human behavior, or dynamic path planning. Development of such modules fundamentally requires programming expertise.

This work is supported by the NEDO (New Energy and Industrial Technology Development Organization, Japan) project, 'Intelligent RT Software Project.'

Other tasks do not, by their nature, require programming ability. These include scripting the robot's utterances, choosing gestures, and structuring the sequence of the robot's actions. Sometimes the specialists most qualified to design the interaction flows or contents of robot behaviors are non-programming researchers or domain experts. However, these specialists are often required to rely upon programmers for development and modification of interaction flows and behavior contents.

Such a design process is inherently inefficient. To improve efficiency in the design of social robotics applications, a structured framework is necessary to enable these fundamentally distinct aspects of social robot application development to be conducted in parallel.

In this paper we propose a framework which uses clearly-defined layers of abstraction to allow this kind of parallel development. In our framework, programming specialists are free to focus on low-level programming tasks like hardware interfacing or data processing. These low-level components are then encapsulated and presented to interaction designers via an easy-to-use graphical interface for developing interaction flows and fine-tuning details of the robot's utterances and gestures.

6.2 Related Work

Related research has explored robotics development frameworks, dialogue management, and the handling of gestures and nonverbal communication.

6.2.1 Development Frameworks

Many powerful development tools exist for programming robot systems [91], and some frameworks such as ROS and Player/Stage have been adopted widely by robotics specialists. Development environments such as Choregraphe [130] enable smooth motion and behavior planning for complex operations such as dancing. Some development environments are targeted towards novice users or even young children [34]. However, all of these systems generally focus on conventional robotics problems such as navigation, mapping, and motion planning.

Some development environments are targeted more specifically towards development of robots for social interaction, including the capacity for developing dialogue management in addition to conventional robotic capability [178, 118]. However, these systems are still based on programming or scripting, and are not intuitive for nontechnical users.

For dialogue development, the CSLU RAD toolkit [108] provides a flowchart-based interface for building dialogue flows, but its inputs and outputs are limited to speech only. A framework for social robots will need to handle many kinds of sensor inputs and actuate both speech and robot motion.

6.2.2 Dialogue Management

Traditional Dialogue Management

There are three main approaches that have been used to create dialogue management systems: state-based, frame-based, and plan-based [109].

State-based systems generate utterances and recognize users' responses according to a state-transition model, like a flowchart. This approach is simple and intuitive, and thus easy to implement and often used in working systems.

Frame-based systems fit users' responses into pre-defined slots in "frames" to estimate user goals. These are often used for telephone-based dialogue systems, *e.g.* for providing weather or transportation information. Frame-based systems can handle more complex information, but involve more effort for preparation of such frames of knowledge.

Plan-based, or "agent-based," systems use a set of rules to change the internal states of an agent to navigate through conversation, *e.g.* [31]. These can handle the most complex interactions, but require very advanced natural-language processing and well defined sets of rules. This approach is often used in research but rarely used in working systems [109].

We chose a state-based approach, as it is the simplest of these three, and it is sufficient to represent the flow of a simple conversational interaction. As our system is aimed at non-programmers, simplicity and clarity are important for usability.

Although handling user-initiated interactions is one weak point of state-based approaches, it is possible to build rich interactions by designing them to be robot-initiated. This may appear to be a disadvantage of state-based modeling, but it is worth noting that robot-initiated interactions are often necessary in order to set expectations for a robot's capabilities.

Dialogue Management in Robotics

In robotics, dialogue management has sometimes been studied while taking real-world difficulties into consideration. For example, Matsui et al. integrated multimodal input for a mobile office robot [5, 102]. Roy et al. used POMDP's to take account of speech recognition errors in state-based transitions of dialogue [30, 133]. For social robots, many architectures for cognitive processing have been developed [175, 142]. The BIRON system has used state-transition models [89] and common-ground theory [97] to direct dialogue. However, the majority of such systems have been task-oriented, that is, aimed primarily at communicating commands or teaching information to a robot [118, 19, 94]. This is different from social dialogue, where the goal of the robot may be to entertain, interest, or persuade a customer.

Some research has focused on using generic dialogue patterns to generate dialogue for human-robot interaction, e.g. [128, 29]. Such research has been directed towards modeling exchanges such as factual confirmations, but not stylistic aspects such as politeness or subtlety of wording. Although such patterns enable automation of certain simple exchanges, it is not yet possible to create humanlike social interaction based on dialogue patterns alone. For applications focusing on social interaction, human knowledge is still needed at the level of implementing dialogue flow.

6.2.3 Nonverbal Communication

For embodied robots, interaction includes not only dialog management, but nonverbal communication as well. Many aspects of nonverbal behavior have been explored, such as the use of gestures and positioning [91, 178, 31, 128, 140, 84], gaze control [117, 116, 9, 157], and nodding [9, 158]. Nakano *et al.* also developed a mechanism to generate nonverbal behavior based on speech context in an embodied conversation agent [119].

Our proposed architecture allows such nonverbal behaviors to be implemented in the robot. Both implicit behaviors, such as gaze-following, and explicit behaviors, such as gestures synchronized with the dialog, are supported.

6.3 Interaction Design Framework

6.3.1 Division of Roles

The concept of division of roles drives the design of our proposed approach. Roughly speaking, we can categorize the main developers of a robot application into "programmers" and "designers." Developers in these two roles contribute in different ways to the implementation of a social robotics application. These different contributions must be reflected in the design framework and user interface.

Programmer

There are several tasks which by their nature require programming expertise.

Hardware interfacing: Adding new sensors or actuators to the system will require work at the robot driver level to enable the new components to operate with the robot's control system.

Data processing: New recognition techniques or machine learning algorithms will be necessary to help the robot understand the situation in its environment.

Behavior development: Basic interactive robot behaviors need to be developed, *e.g.* a behavior for approaching a moving person in a socially-appropriate way, based on tracking information from an external sensor network.

Interaction Designer

The tasks of an interaction designer center around the creation of content for the robot's interactions, and the creation of logical sequences of robot behaviors to be executed. Specifically, design tasks include the following:

Dialogue generation: An interaction designer will need to specify the robot's utterances and gestures. To tune the robot's performance, a designer could adjust the speed of the robot's actions or speech, or insert appropriate pauses.

Interaction flow design: By linking the robot's behaviors into sequences, a designer can create interaction flows. The designer needs to consider the order in which the robot should present information, when it should ask questions, and how it should respond to a person's actions. Non-dialogue elements could be used in these flows, such as driving to a new location or approaching a customer. An understanding of HRI design principles would be useful for an interaction flow designer.

Content entry: It may also be necessary to enter large amounts of domain-specific content, such as items in a restaurant menu, details about products in a store, directions to locations in a shopping mall, or information about seasonal events. This task might require a designer with specific domain knowledge relevant to the target application.

6.3.2 Robot Control Architecture

Our system uses a four-layer architecture, shown in Fig. 6.1. While similar to other modular architectures, the emphasis here is on the encapsulation of low-level control and processing into simple components such as behaviors and explicit gestures, which can easily be used by a non-programming designer to create social interaction flows.

Robot Driver Layer

The lowest layer is the robot driver layer, which contains hardware-specific driver modules. These modules support abstract interfaces that hide minor differences between similar robots, such as different motor or joint configurations, or size differences (*e.g.* slightly longer/shorter arms, human-size or baby-size, etc.). This enables the same applications and behaviors to be used with different robots, as long as they are functionally similar, *e.g.* wheeled humanoid robots. Our architecture currently supports four robot platforms.

The concept of modular drivers is not new, and in theory it should be possible to implement a system like ours on top of popular modular middleware frameworks such as Microsoft's Robotics Developer Studio or Willow Garage's ROS.

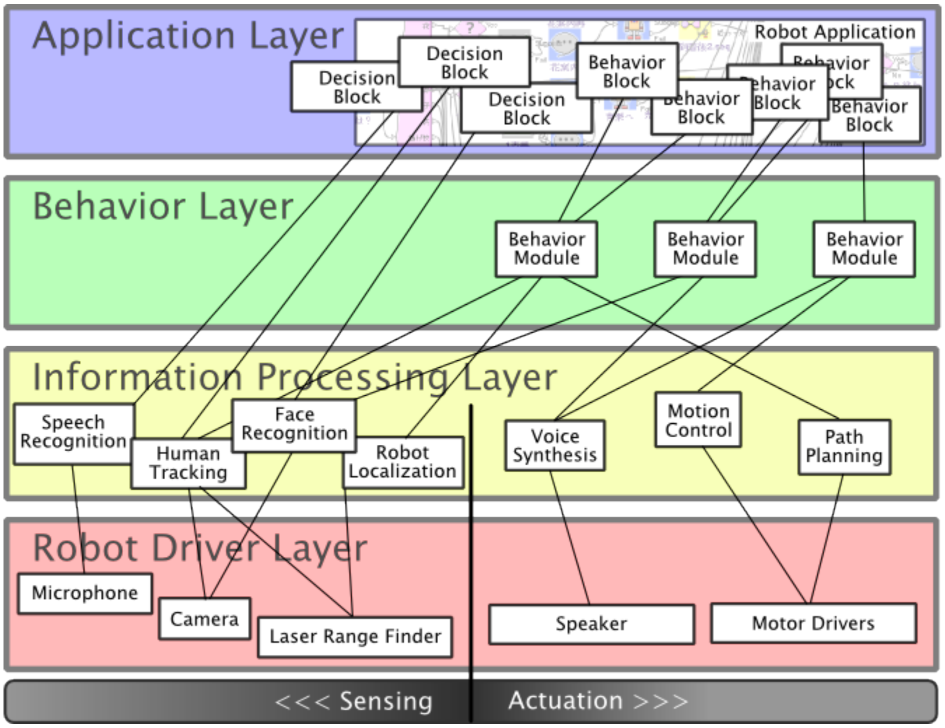


Figure 6.1: Four-layer robot control architecture.

Information Processing Layer

The information processing layer contains sensing and actuation modules. Sensing modules are components related to recognition of environments and activities in the real world. Examples include localization, human tracking, face detection, speech recognition, and sound source localization.

Actuation modules perform processing for tasks like path planning or gaze following. Some knowledge about social behavior is implemented here. Following the approach in [149], we classified non-verbal behaviors as *implicit*, which do not need to be specified by designers, and *explicit*, which need to be synchronized with utterances. Based on the state of conversation (*e.g.* talking, listening, or idling), components in this layer generate implicit behaviors such as gaze control.

Some frameworks (*e.g.* Microsoft RDS), are primarily targeted towards development at this level for robotics research or education, but our framework considers this layer mainly as infrastructure to enable the creation of higher-level behaviors.

Behavior Layer

The concept of a robot "behavior" as a combination of sensor processing and actuation is used both in behavioral robotics, *e.g.* [12], and in social robotics [84]. Examples for social robots include guide behaviors incorporating speech, gesture, and timing, or approach behaviors which react to a person's trajectory [139].

In our architecture, behaviors are implemented as software modules in the behavior layer which execute actions and react to sensor inputs. They can incorporate social knowledge, for example, by specifying gestures like tilting the robot's head to one side while asking a question [149]. It is also possible to design behavior modules to be configured by designers from the application layer. This is a powerful concept, as it enables the development of flexible, reusable behavior modules.

Application Layer

The highest layer is the application layer, where designers can develop social robot applications. Using "Interaction Composer," the graphical interaction development environment shown in Fig. 6.2, non-programmers can access behavior and sensor modules in the underlying layers. This software enables interaction flows to be built by assembling behavior and decision blocks into sequences resembling flowcharts.

6.3.3 Interaction Composer

It is important to note that Interaction Composer (IC) is not simply a graphical programming language. Its graphical representations map directly to the underlying software modules, making it a tool that bridges the gap between designers and programmers.

Behavior Blocks

Behavior blocks (the darker blocks in the flow example shown in Fig. 6.2) allow the designer to use the behavior modules defined by programmers. These can represent behaviors like asking a question or giving directions. By configuring the properties of a behavior, the designer creates a "behavior instance." A behavior flow may contain many instances of general behaviors like "Talk" and "Ask". When a programmer creates a new behavior module, a corresponding block becomes available for designers to use in IC.

A developer can also allow a designer to provide arguments for behaviors. By configuring behaviors through IC, a designer can easily use behaviors in different ways without knowing the details of the program embedded in the behavior.

Fig. 6.3 shows two possible configurations of the "Approach" behavior. The behavior itself involves complex information processing such as dynamic path-planning for approaching from the frontal direction of the person. However, the concept of "approach" is easily understood, and thus a designer can use the behavior without knowing the details of internal mechanism. There are three arguments prepared for the "Approach" behavior: the robot's speed, the distance at which to speak to the target person, and the contents of utterance. The designer could configure the behavior for "slow speed" and "social distance (1.5 m)" for

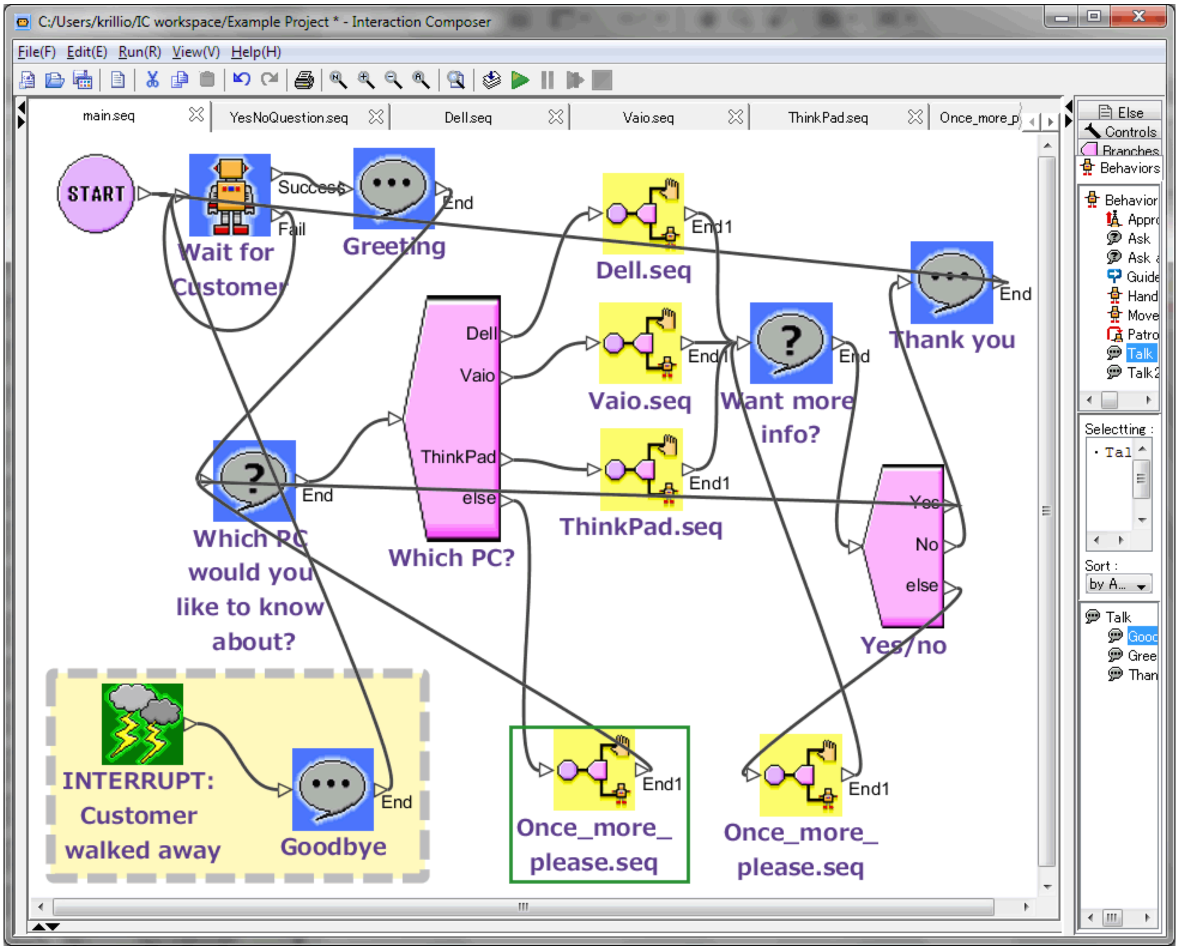


Figure 6.2: Screenshot of Interaction Composer.

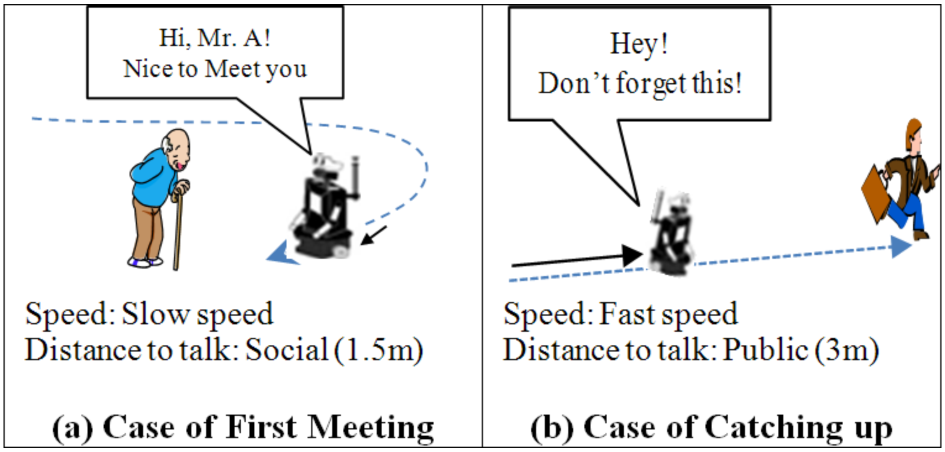
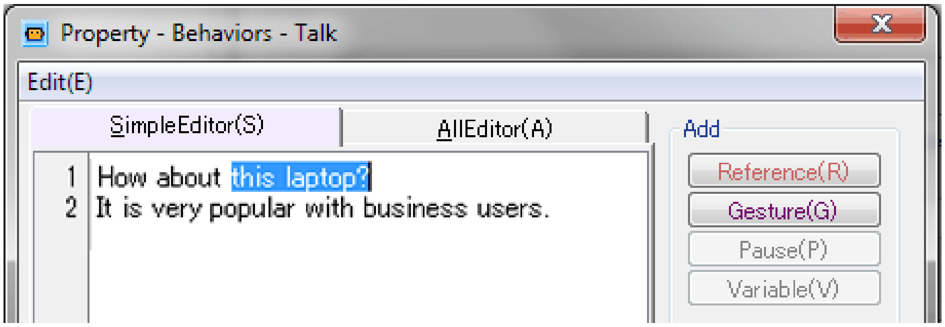


Figure 6.3: Example configurations of an “Approach” behavior.

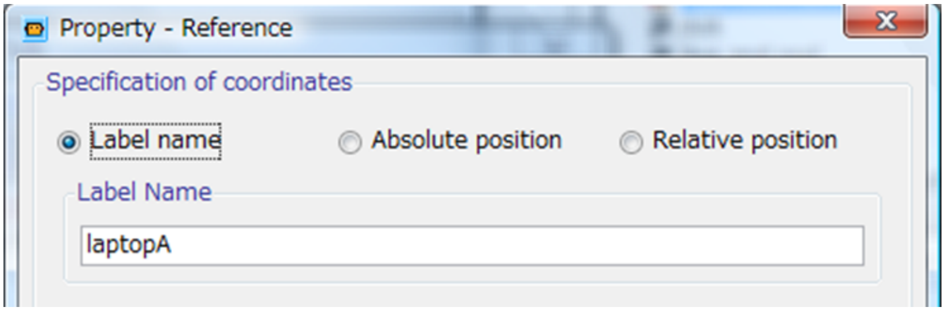
approaching a waiting person; or with “fast speed” and “public distance (3 m)” in case of catching up with a person who forgot an item.

IC also supports easy addition of robot gestures. Figs. 6.4 and 6.5 show screens for editing gestures to be associated with utterances. In Fig. 6.4, a designer inputs an utterance, (”How about this laptop?”), and then chooses a part (”this laptop”) to add a gesture. By clicking the “reference” button in Fig. 6.4, a new screen for reference (pointing) gesture appears (Fig. 6.5), in which the designer can choose a pre-defined label (“LaptopA”) for a pointing gesture. The robot will do the pointing gesture when it utters the “this laptop” phrase toward the object labeled “LaptopA”. Other gestures, such as “emphasis,” “big,” “small,” etc. can also be selected.



How about this laptop?
It is very popular with business users.

Figure 6.4: Selection of the utterance where a robot points to an object.



Specification of coordinates

● Label name ○ Absolute position ○ Relative position

Label Name: laptopA

Figure 6.5: Selection of a pointing target.

The robot also used implicit gestures, causing it to move its arms and head slowly while idling, more actively while talking, and tilting its head to the side while asking questions. When explaining the different products, explicit behaviors were also included in the utterances, such as emphasis gestures and pointing to the products being explained.

Decision Blocks

The flow of the interaction can be controlled by using decision blocks (the pentagonal blocks in Fig. 6.2) to direct the execution flow based on data from sensor inputs or internal state variables. These blocks enable the designer to work directly with human-readable data from the information processing layer (see Sec. III-B-2).

Examples of sensor inputs are the “ListenWord” (speech recognition results) and “DistanceHumanToLabel” (human position tracking) variables. For example, if a designer creates a scenario for shop assistant robot, the designer could specify that the robot should ask if the customer is looking for a desktop when “DistanceHumanToLabel(DesktopPC) <=1000”, meaning the customer is standing within one meter of the desktop PC. This information comes from the information processing layer, so the designer needs to know nothing about the implementation of the tracking algorithm.

“Sequential” and “random” decision blocks are also provided, which can be used to select a different output node each time they are executed. These blocks can be useful for adding lifelike variation to behaviors which are often repeated.

Sequences

Using only behavior and decision blocks, a program can quickly grow to be unmanageable in size and complexity. To manage this complexity, our system enables encapsulation of execution flows into subroutines, which we call “sequences.”

Sequences (the lighter blocks in Fig. 6.2) can be edited as separate execution flows, and then used as blocks within other sequences. They are a powerful tool, increasing readability and enabling structured development and debugging of interactions.

Some uses for sequences include encapsulating common tasks such as confirmation questions or delineating sections of a dialogue flow.

Interrupts

The flowchart-based representation of a dialog flow is helpful for structured, robot-driven interactions, but it does not allow us to easily react to unexpected situations. For example, if a customer walks away during an interaction, it might be best for the robot to stop speaking and begin searching for a new customer. Yet, a flow which performs such a check after every utterance would be tedious to build and hard to read.

For situations like this, we provide an “interrupt” mechanism. When the conditions of an interrupt are satisfied, the robot’s execution flow will jump to a specified sequence.

For example, an interrupt could monitor the robot’s on-board laser range finder, interrupting the flow if no human is detected in front of the robot. If this interrupt is triggered during a conversation, it means the customer has walked away, so instead of finishing a one-sided conversation, the robot could search for a new customer.

6.4 Experimental Evaluation

We conducted an experiment to evaluate the effectiveness of our parallel design approach using Interaction Composer. In our experiment, teams of one programmer and one designer collaborated to develop a small application for a shopkeeper robot at a computer store.

6.4.1 Conditions

We used a between-participants experimental design with two experimental conditions: *with-IC* and *no-IC*. In the *with-IC* condition, the designer used Interaction Composer to build the behavior flow, while the programmer worked in C to solve the programming problems. For the *no-IC* condition, Interaction Composer was not provided. Instead, the designer created interaction flows on paper, and the programmer implemented them in C, while also working on the programming tasks.

6.4.2 Experimental Setup

For this experiment we used a Robovie R-2 humanoid robot. Speech recognition was performed using the ATRASR speech recognition engine [150], and face detection was performed using a custom application written using OpenCV.

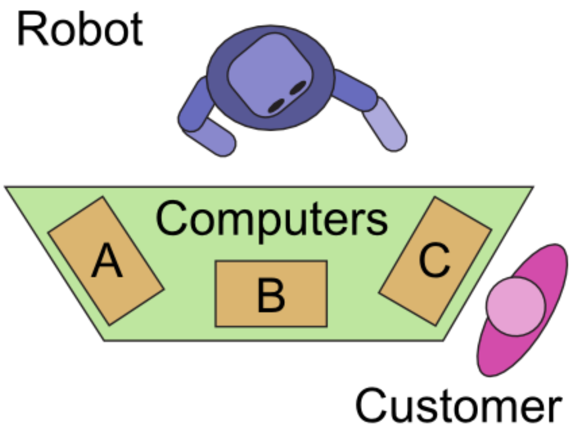


Figure 6.6: Layout of robot and computers in experiment space.

The experimental environment was laid out as shown in Fig. 6.6, with a stationary robot placed behind a table. Three laptop PC's were placed on the table, and a customer stood across the table from the robot, looking at the PC's. As the customer moved around to examine different PC's, the robot could determine the customer's position by turning its head and using face detection, and it could conduct simple conversations with the customer using speech recognition.

6.4.3 Task Specifics

To choose an appropriate balance of tasks, we considered what a typical preparations for a deployment of social robots might entail. Let us assume robots are to be deployed in a retail shop as sales associates. This would require sophisticated social interactions such as providing product information, making recommendations based on customer needs, explaining special offers, and gently encouraging customers to buy more expensive products or accessories. The robots would need to display professionalism as their actions reflect on the shop and influence customers' purchasing decisions. Assume further that the core robot system itself is a stable system for commercial use, but it has recently been upgraded with new sensors.

The design tasks to prepare for such a deployment might include developing hundreds of explanations, creating many different patterns of interaction sequences, fine-tuning the timing and gestures, and testing the smoothness of flow transitions. Programming tasks might include developing and testing recognition software for use with the new sensors. In such a situation, it would clearly be advantageous to have the interaction content developed by domain experts familiar with the products and sales techniques in the shop, enabling the programmers to concentrate on the programming tasks.

Although a real development cycle would require many people and several months, we designed this experiment to be completed within a single day. To demonstrate our proposed approach, representative design and programming tasks were chosen which could be achievable within a few hours of work. The task specifications were the same for both conditions.

Design Task

The design task for this experiment was to develop content and an interaction flow enabling the robot to explain at least two features (price, CPU speed, etc.) of each of three computers in a socially smooth conversation with a customer.

Participants were given a set of behaviors and functions, presented in Table 6.4.3 along with their C API equivalents for the *no-IC* condition. A list of available gestures was also provided. In both conditions, gestures were added by placing markup tags in the text to be spoken, as in the following example.

```
   This PC has <gesture type="emphasis"> six hours </gesture>
of battery life.
```

Programming Task

The programming task focused on the processing of sensor data, which is a common task for robotics programmers. We provided participants with a face detection application for identifying whether a customer was present and where they were standing. As real-world data is noisy, the programmer's first task was to create a simple filter to remove false face detections based on features such as height and width.

The second task was to compute the customer's position in space, based on the face detection data. This would enable the robot to turn its head towards the customer while interacting.

Behavior	**C Function**	**Description**
Talk	`void talk(string text);`	Speak and/or perform gestures.
Ask	`void ask(string text, int time, string expectedResponses);`	Speak, then listen for a spoken response within a given time limit.
LookForFace	`int lookForFace();` Returns 0 for left, 1 for center, 2 for right, 3 for none.	Look for a face to the left, center, and right of the robot.

Variable	**C Function**	**Description**
faceDetected	`int isFaceDetected();` Returns 1 if face detected, 0 if none.	True if a face is currently visible
listenWord	`int isSpeechResult(string result);` Returns 1 if the speech result was equal to *result*, or 0 otherwise.	Most recent speech recognition result

Table 6.1: Behaviors and Variables

6.4.4 Fairness of Conditions

For this experiment it was essential to provide exactly the same capabilities in both the C interface and the Interaction Composer (IC) interface. To make the interfaces as equivalent as possible, we created a single C function corresponding to each behavior template available in IC, as shown in Table 6.4.3.

For example, Fig. 6.7 shows a simple flow in IC. The same flow could be built using our C interface as follows:

```
while (lookForFace()==3) {}
talk("Welcome to my computer shop!");
```

C equivalents of the conditional, sequential, and random decision blocks were not provided, as this functionality is easily available in C, using `if` statements, `for` loops, and the `rand()` function.

The equivalent of sequences is also trivial to implement in C, simply by defining a function, and interrupts were not used for either condition in this experiment. Thus, all functionality available in IC was also easily usable in the C interface.

6.4.5 Participants

32 pairs of participants (49 male, 15 female, average age 24.8 years) took part in this experiment. Designers were required to have no computer programming experience, and programmers were required to have basic proficiency in the C language. Programmers were also given an entry-level C programming test before the experiment, and their scores were used to choose the condition for their trial. This enabled us to balance the skill levels of the programmers between conditions.

6.4.6 Procedure

After 2 hours of instruction, 3.5 hours were given for developing the robot application. Each hour, we evaluated the progress of the application using a checklist of 17 requirements, and we gave the participants feedback about missing features or serious problems.

The requirements checklist was independent of the experimental condition, and was strictly an evaluation of the robot's outward behavior, not the underlying implementation. Examples include the following:

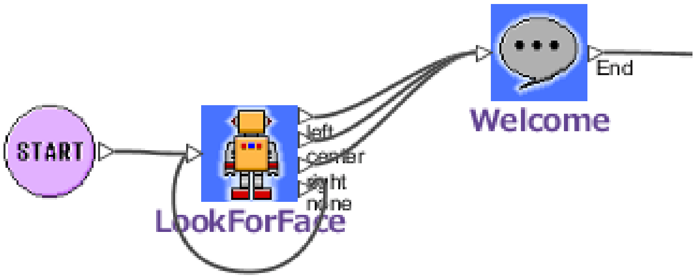


Figure 6.7: Example flow using IC.

Greet the customer when they arrive.
Introduce at least two features of each product.
Explain only features requested by the customer.
Show variety in utterances when they are repeated.
Say goodbye only when the customer has left.

6.4.7 Evaluation

We evaluated the overall quality of the completed applications and performed secondary evaluations of the individual subtasks.

Primary Evaluations

We first measured the overall quality of the completed applications using the requirements checklist, for a score from 0 to 17, where 9 points represents completion of all basic tasks.

We also conducted an interactive evaluation, in which two evaluators, blind to the experimental conditions, spent 10 minutes interacting with the robot for each application and gave subjective quality ratings on a 100-point scale. These evaluators considered things like the appropriateness of utterances, naturalness of gestures, and how the robot made them feel as a customer.

Secondary Evaluations

We measured performance on the programming tasks by testing the accuracy of the face detection filter and noting whether the second programming task had been completed, as many teams skipped this optional task due to time pressure.

To quantitatively measure the complexity of the interaction design, we counted the number of unique utterances used in each interaction flow, expecting that interactions of higher quality will display a greater variety of utterances.

6.5 Results

Results of primary evaluations regarding the overall performance of the robot application are shown in Fig. 6.8, and results from secondary evaluations are shown in Fig. 6.9.

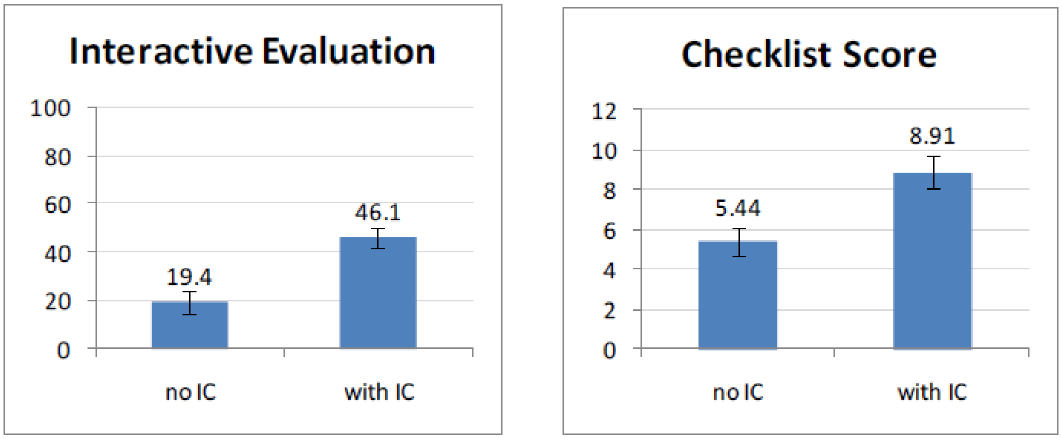


Figure 6.8: Results for primary evaluations.

6.5.1 Primary Evaluations

For the interactive evaluation, we averaged the ratings between the two evaluators. A one-way ANOVA conducted for the interactive evaluation results revealed a significant main effect ($F(1,30)=20.659$, $p<.001$, partial $\eta^2=.408$). A one-way ANOVA conducted for the checklist scores also revealed a significant main effect ($F(1,30)=9.905$, $p=.004$, partial $\eta^2=.248$). Applications in the *with-IC* condition significantly outperformed those in the *no-IC* condition in both evaluations.

6.5.2 Secondary Evaluations

Face tracking accuracy was similar between the two conditions. A one-way ANOVA conducted for face-tracking accuracy revealed no significant effect ($F(1,30)=.299$, $p=.589$, partial $\eta^2=.010$). As we intentionally balanced the ability levels of programmers between conditions, is unsurprising that we did not see a significant difference in this task. In the *no-IC* condition, programmers typically gave this task priority and completed it before working on the interaction flow. For Task 2 completion, however, a chi-square test revealed a significant difference between conditions ($\chi^2(1) = 18.286$, $p<.001$, $\phi=.758$).

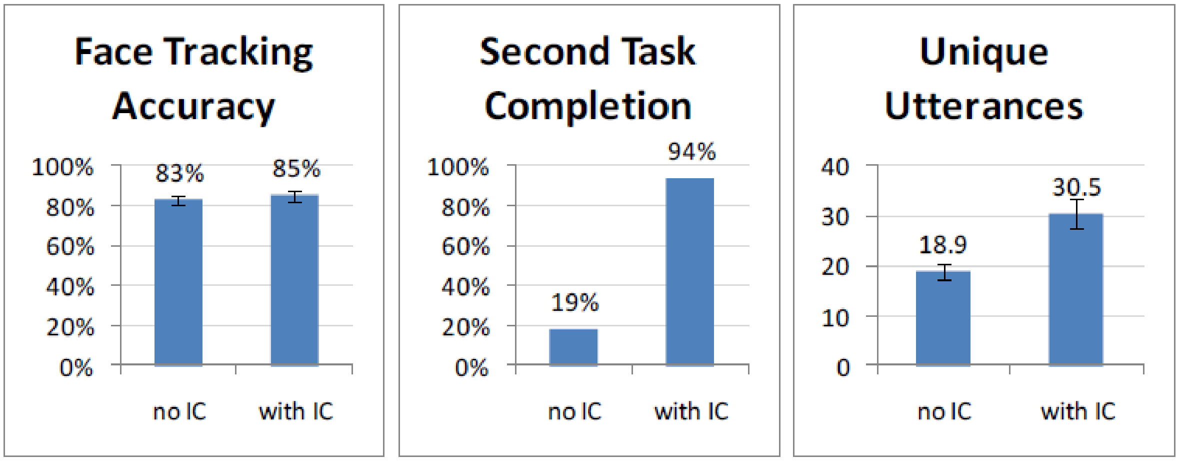


Figure 6.9: Results for secondary evaluations.

Only 19% of programmers in the *no-IC* condition completed the second task, compared with 94% in the *with-IC* condition. Programmers in the *no-IC* condition were usually too busy building the interaction flow to work on this lower-priority programming task.

Finally, a one-way ANOVA was conducted for the number of unique utterances, revealing a significant main effect ($F(1,30)=12.760$, $p=.001$, partial $\eta^2=.298$). These results showed significantly more utterances in the *with-IC* condition, indicating that the increased involvement of the designer enabled the creation of more complex interaction flows.

6.6 Discussion and Conclusions

6.6.1 Observations

During our experiment, different teams used our design framework with varying degrees of success. One thing we observed is that a clear understanding of the interface between the programming side and the interaction design is critical for productive collaboration. Some designers using IC misunderstood the functionality of the robot's behaviors, for example, confusing the *LookForFace* behavior and the *faceDetected* variable. These designers built flows with redundant or incorrect use of behaviors.

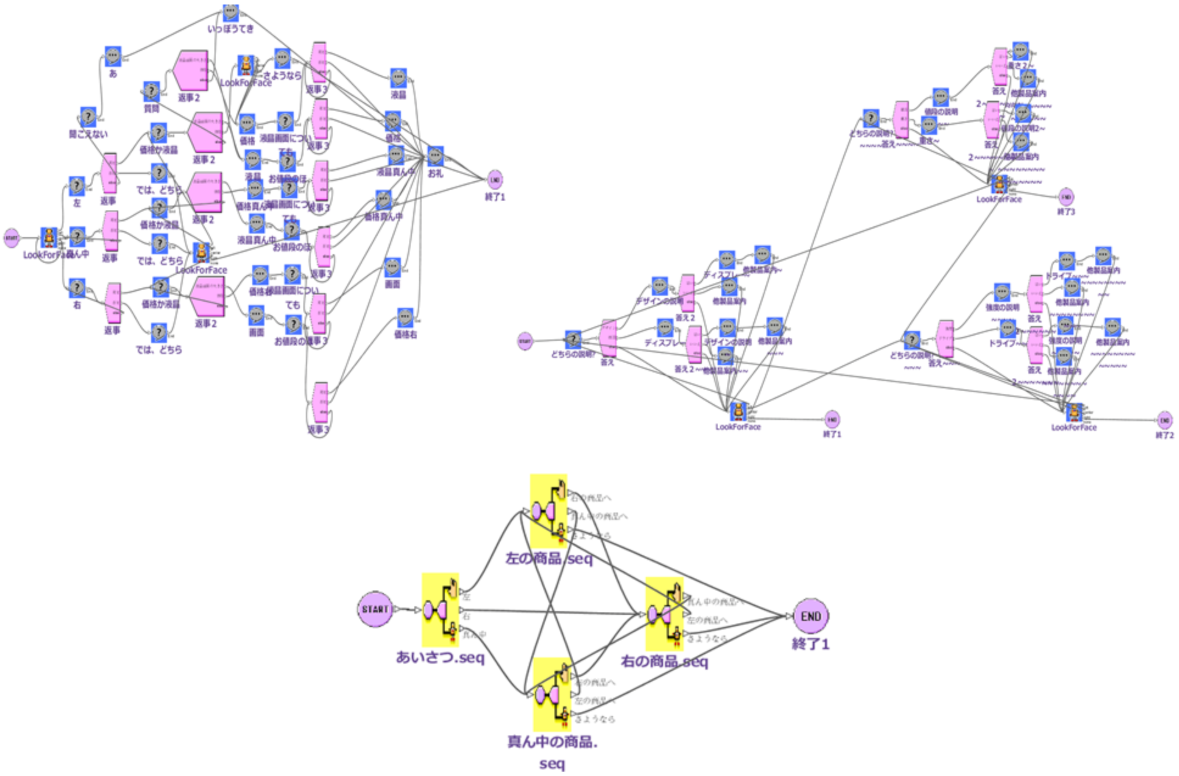


Figure 6.10: Examples of flow organization.

All teams in the *with-IC* condition used sequences, but some were more effective than others. Many designers built nearly the entire flow within a single sequence, *e.g.* Fig. 6.10 (upper left). Others took advantage of the graphical freedom in IC to arrange the elements into visual groups, *e.g.* Fig. 6.10 (upper right). More than half of the teams organized their flows by using top-level sequences, *e.g.* Fig. 6.10 (bottom). The most successful teams used sequences extensively (the best used 11, while most used 5 or fewer) to encapsulate tasks like asking confirmation questions or confirming the customer's presence.

In some of the more successful *no-IC* cases, programmers took strong initiative in the interaction design, using the designer's flow as a rough guideline since the designer did not have a clear understanding of what was difficult or easy to implement. The best programmers were able to generate logical flows equivalent to average-performance flows in the *with-IC* condition, but their flows lacked advanced features like checks to see if the customer had moved, variation in utterances, and small talk. These features were present in many *with-IC* flows.

Many unsuccessful *no-IC* cases failed because of mistakes in software design. As text is a linear medium, it can be hard to see the structure of an interaction flow just by looking at source code. Without a visualization of the structure, many programmers forgot to handle important contingencies, such as the case when no speech recognition result is received.

6.6.2 Scalability

Scalability is a concern for any programming environment, and while our results show that the state-based approach we use for dialog management is useful for simple flows, its scalability and flexibility remain important questions.

We have found this approach useful for long, mostly-linear flows, and if the robot guides the interaction by asking questions, people's responses are usually predictable. We have also found it to be effective for interactions where the robot needs to make simple responses to a large number of keywords, *e.g.* providing directions to one of 90 places in a shopping mall.

The state-based approach is not as effective when the robot needs to remember interaction history, *e.g.* offering to explain only information it has not already presented. In these situations the complexity of the flow rises exponentially with the amount of state that needs to be remembered. Such situations are fairly common in social interactions, so for some applications we have developed custom behavior modules to handle interaction history.

So far, the current balance between functionality and ease-of-use has been sufficient for our applications. No doubt this balance will change in the future as applications become more complex. However, the principle of enabling designers and programmers to collaborate through parallel development will only become more important as complexity increases.

6.6.3 Conclusions

In this paper we have presented a novel interaction design framework which enables non-programmers and programmers to work in parallel to develop interactive applications for social robots. In our experiment we have validated that this new design approach increases efficiency and application quality.

Structuring the development process to reflect the unique roles of designers and programmers should help to increase efficiency and enable both programmers and designers to produce higher quality work, making this a first step towards a scalable development process that will eventually be applicable to commercial social robotics applications.

Chapter 7

The Network Robot System

This chapter will present an overall framework designed to support deployments of multiple social robots in real public environments. In addition to the elements discussed so far, the framework provides coordination between robots operating in the same environment, manages the assignment and scheduling of robot services, and enables structured knowledge sharing between system elements.

A set of functional requirements for a network robot system are discussed, based on several years of experience in conducting field studies with social robots in public spaces. The implementation of a network robot system is then presented, and a demonstration in a shopping mall illustrates how such a network robot system framework can be used to support heterogeneous teams of robots providing services in a real public environment.

7.1 Introduction

Over the last decade, our laboratory has conducted many field trials in real-world environments, such as a science museum [152], a train station [153], and shopping malls [151, 78]. Each deployment posed challenges in recognition, decision-making, robot coordination, and information sharing. Through this experience, we have developed and refined a framework which addresses these challenges.

This framework is based on a "Network Robot System" (NRS) design approach, in which the robots themselves are merely the visible component of a network which integrates environmental sensor systems, central planning servers, cloud-based knowledge resources, and human users and supervisors. This framework has been successfully used by our research group (Fig. 7.1 (a)) and in collaboration with others (Fig. 7.1 (b)) in several field deployments. While similar systems have been developed for multi-robot task management, this is the first time this approach is being demonstrated in the domain of social robots.

In this paper, we will present our framework in the context of tasks such as guiding customers in a shopping mall. Our intention, however, is to share a general approach which can be useful in service robot deployment scenarios like those explored by other groups,

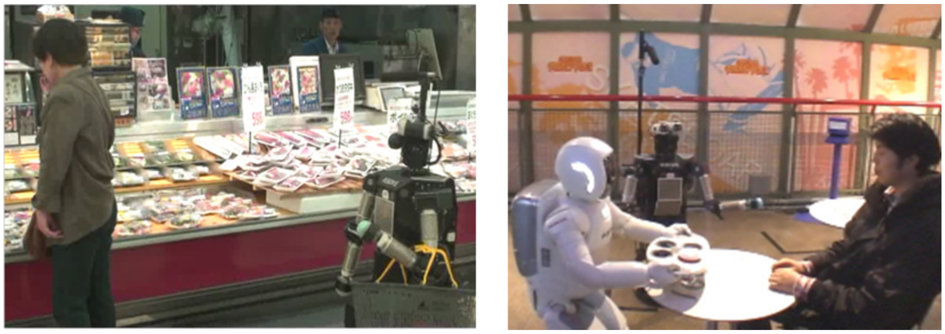


Figure 7.1: Example scenes using our Network Robot System framework. (a) helping a customer with shopping, (b) collaboration between heterogeneous robots.

e.g. trash collection [106], pedestrian guidance [46], and assisting people in hospitals [115], supermarkets [174], and offices [169].

As this framework involves many technical components, we will often refer to publications which present those subsystems in greater depth.

7.2 Related Work

7.2.1 Social Robots in the Field

Robots developed for HRI in public spaces, such as museum guides [15, 172] and shopping assistants [60] have addressed navigational and perceptual problems such as people-tracking and localization in public spaces. Other research, like Snackbot, has focused on elements such as dialog content and social appropriateness [93]. Robots have also been placed in busy public spaces to investigate social acceptance of robots [184, 88]. These are examples of the types of robot services that our proposed framework aims to support in a modular and scalable way.

7.2.2 Ambient Intelligence

To augment on-board sensing, we often use “ambient intelligence” (AI) systems embedded in the environment. Laser range finders (LRFs) located in an environment can help a robot to find and approach pedestrians from a distance [81]. Environmental sensors have also been used with mobile robots to cover large areas for surveillance [137]. The PEIS Ecology (“Physically Embedded Intelligent Systems”) project explored an architecture for integrating robots with various sensors and actuators in a smart-home environment [135]. As such systems can greatly aid robot perception and recognition, our proposed framework emphasizes integration with AI systems.

7.2.3 Multiple-robot coordination

Other research has addressed networked robots in traditional, non-social robot scenarios. Some architectures address decentralized cooperative behavior of swarm robots [126]. Other studies have focused on coordinating multiple robots, such as control of coverage [76] and formation [180, 111], and efficient exploration [64]. Centralized control approaches have also been taken, particularly when delegating heavy computation from mobile robots to servers [77].

While these approaches can produce simple behaviors such as shape formation, social robot applications require more complex behaviors than these frameworks can provide.

7.2.4 Networked Social Robot Systems

Multiple-robot networks have also been developed for HRI. The Expo.02 work of Siegwart et al. was one example, where robots shared their locations and some actions were coordinated to control the flow of visitors [159]. The DustBot project was also designed to support multiple robot types, and it included communication with beacons [37]. However, these works involved little or no coordination among robots, and did not propose a general framework that could address various methods of coordination. By contrast, we are proposing a general framework, handling communication, robot coordination, and ways to define and allocate services between a central server and individual robots.

Many studies have addressed task allocation and coordination, summarized in [47], where often the goal is task allocation for a large number of robots and a primary concern is efficiency in allocation given utility of tasks. Our NRS's task allocation is relatively simple from an algorithmic point of view, but it addresses many practical considerations required for allocating social robots to provide services to human users. Hence, our work provides a successful example of how we can instantiate such theoretical ideas in a very concrete way.

7.3 Scenarios and Design Requirements

To provide a context for understanding the functionality of our framework, we will present three simplified robot service scenarios that we have actually implemented and demonstrated. These were chosen to showcase typical functionalities used in many of our field trials. Throughout the paper, we will refer back to these scenarios to place the system elements in context.

7.3.1 Scenario I: Multi-robot Coordination

In this scenario, a customer at the mall entrance uses his mobile phone to request a robot to provide information about the shopping mall, while a nearby customer requests a robot to carry his bags (Fig. 7.2). A central server assigns robots to these tasks based on their locations and capabilities, and plans coordinated paths for them to approach the customers. A humanoid robot approaches the first customer to offer information, and a robotic shopping cart approaches the other one, offering to carry his baggage.

This scenario demonstrates the necessity of service allocation based on robot capabilities and coordinated path planning between robots.

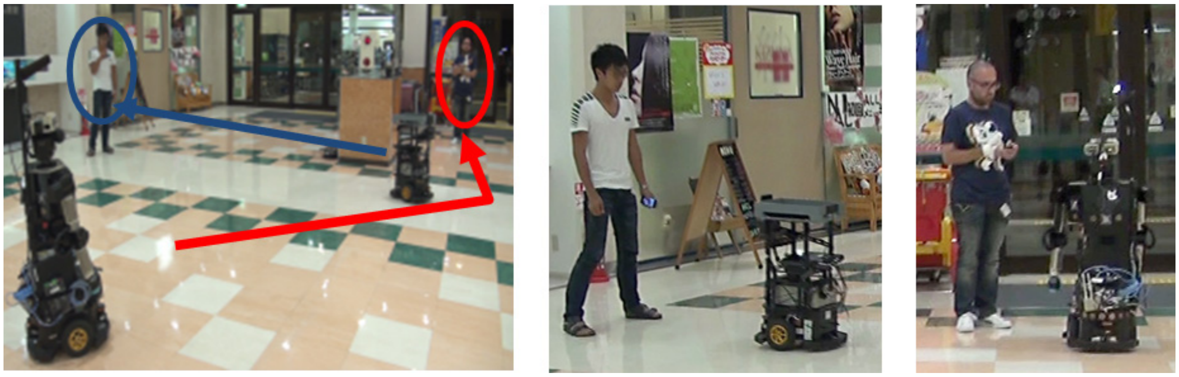


Figure 7.2: Coordinating multiple robots: (a) Different services are requested, (b) Each robot engages in conversation with a customer.

7.3.2 Scenario II: Context-Aware Service

The next scenario illustrates an example wherein robots proactively provide services to people based on their situational context, rather than responding to explicit requests. Near a large intersection in a shopping center, a robot is waiting to offer route guidance to customers. A woman stops in front of a map of the mall (Fig. 7.3 (a)). While looking at the map, she is approached and offered help by the robot: “Are you looking for a particular shop?” (Fig. 7.3 (b)). The robot then answers any of her questions by giving directions or accompanying her to a destination. This scenario illustrates the need to recognize and anticipate people’s needs, and the ability to allocate robot services accordingly.

7.3.3 Scenario III: Personalized Services

The third scenario shows how the NRS enables personalized services to be delivered to customers. From home, a registered customer uses her mobile phone to request a robot to help her with shopping and enters her shopping list (Fig. 7.4 (a)). Upon her arrival at the mall (Fig. 7.4 (b)), a robot comes to greet her and accompanies her through the supermarket carrying her basket (Fig. 7.4 (c)). Based on their location in the supermarket, the robot can remind her of items on her shopping list. This scenario illustrates how the personal information can be used in services. It also shows the need for a means of personal identification of the service recipient.

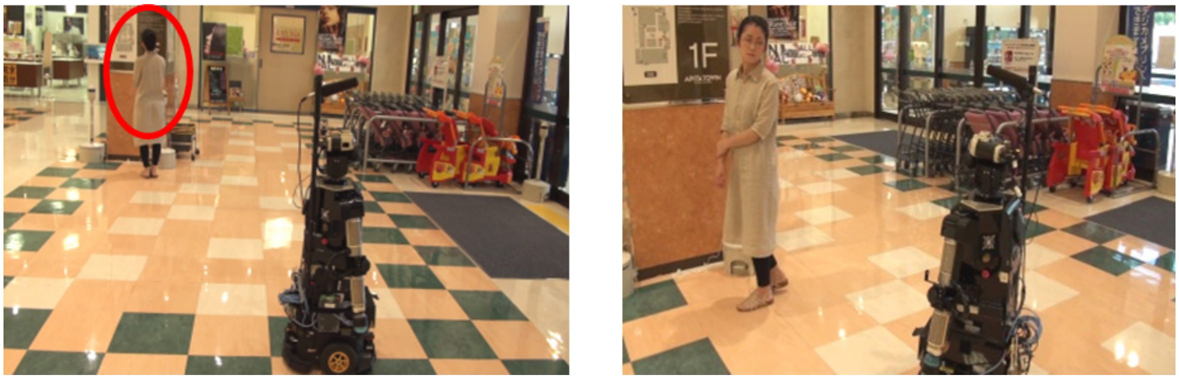


Figure 7.3: An example of *context-aware service*: (a) The sensing framework detects a woman stopping in front of a map, then (b) The robot approaches her to offer information.

7.3.4 Design Requirements

The scenarios outlined here have illustrated several important requirements for a NRS framework. Other requirements, e.g. navigational safety and supervision by a remote operator, apply to all of the scenarios. Table 7.1 summarizes the key requirements which we considered in the design of our framework.

When we describe the elements of our system, we will refer back to Table 7.1 to show how those elements help satisfy this set of requirements.

7.4 System Implementation

In the process of preparing several field deployments of robots in commercial and public spaces, we have developed a network robot system framework that has come to address these requirements.

The high-level elements of our system are shown in Fig. 7.5. These include a **sensing framework**, several **information modules**, a system for **service allocation**, a **coordination module** for navigational coordination and path planning, and support from a **human supervisor**. Table 7.2 summarizes these elements, where the numbers in parentheses correspond to the requirements presented in Table 7.1.

In this section we will present the general design of each module of our system as well as specific instances of these modules from our implementations. Requirements from Table 7.1 addressed by a module are designated in parentheses, e.g. (**Table 7.1 – N1**).

7.4.1 Sensing Framework

For robots to interact with humans, they need to be able to perceive and react to who people are and what they are doing. A sensing framework embedded in the environment can be

Category	Requirements
Navigation	N1. Localization N2. Safety N3. Path planning and spatial resource allocation N4. Socially appropriate motion near people
Conversation	C1. Recognition support C2. Expert knowledge
Anonymous Services	A1. Recognize and anticipate people's behavior A2. Assign services based on anticipated need
Personalized Services	P1. Identify individuals P2. Enable users to request services P3. Store customer data
Modularity	M1. Coordination of services M2. Support for multiple types of robots
Safety and Supervision	S1. Monitor for and correct recognition errors S2. Identify and intervene in problem situations S3. Enable one operator to supervise many robots

Table 7.1: System requirements

used to track people in real time, to assist robot localization, and to anticipate people's future behavior.

Robust tracking of people

Tracking the motion of people enables robots to react to a person's identity and behavior, and motion data can help infer socially meaningful knowledge such as direction of attention [51], which can be useful for social interactions.

Sensors embedded in the environment can provide wide-area, high-precision tracking

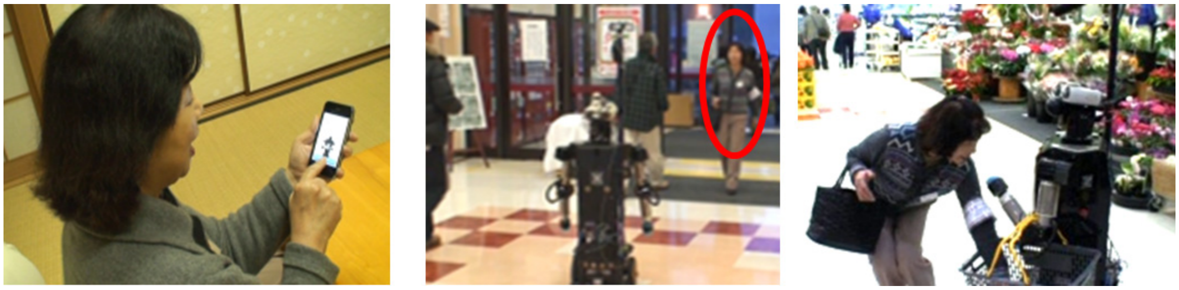


Figure 7.4: An example of *personalized service*: (a) requesting a robot from her mobile phone, (b) detecting the customer's arrival, and (c) shopping with the customer.

System Element	Functionality
Sensing framework	Robust tracking of people (N4) Recognize and anticipate people's behavior (A1) Assist in robot localization (N1) Identify individuals (P1)
Information modules	Store information about robots (M2) Store information about environments (N2,N3) Store information about customers (P3)
Path planning and spatial allocation	Coordinate robot paths to avoid conflicts and deadlock (N3) Ensure smooth locomotion near people (N4) Provide paths based on robot type (M2)
Service allocation	Coordinate robot services (M1) Enable users to request services (P2) Assign services based on anticipated need (A2)
Support from a human operator	Support for recognition (C1,S1) Direct control of robot (C2,S2) Ability to control multiple robots (S3)

Table 7.2: Key system elements and functions. Numbers in parentheses refer to requirements from Table 7.1

of people (**Table 7.1 – N4**), which can outperform people-tracking using on-board sensors in many situations: in crowded or cluttered areas, occlusions and noise can prevent stable tracking using onboard sensors, and resolution of laser or visual tracking drops off with distance from the robot. In our experience tracking pedestrians in a shopping mall with an on-board laser range finder, leg detections become indistinguishable from clutter and sensor noise beyond about 6 meters from the robot.

Sensors such as laser range finders [51] and video cameras [103] can be deployed to expand sensing coverage, minimize occlusions, and increase accuracy to track customers' locations precisely. We have also used Wi-Fi fingerprinting [6] for coarse localization of specific individuals.

Identifying individuals

Identification of individuals (**Table 7.1 – P1**) enables personalization of service, as illustrated in Scenario III, and continuity of service over time or between robots. In our work, we have used techniques such as RFID [152, 85], visual face recognition [20], and Wi-Fi-based identification using smartphones for this task. By combining these results with the high-precision position information available from the position tracking systems, we can locate known individuals with high precision [122].

In our demonstration of Scenario III, tracking was performed using a laser-based people

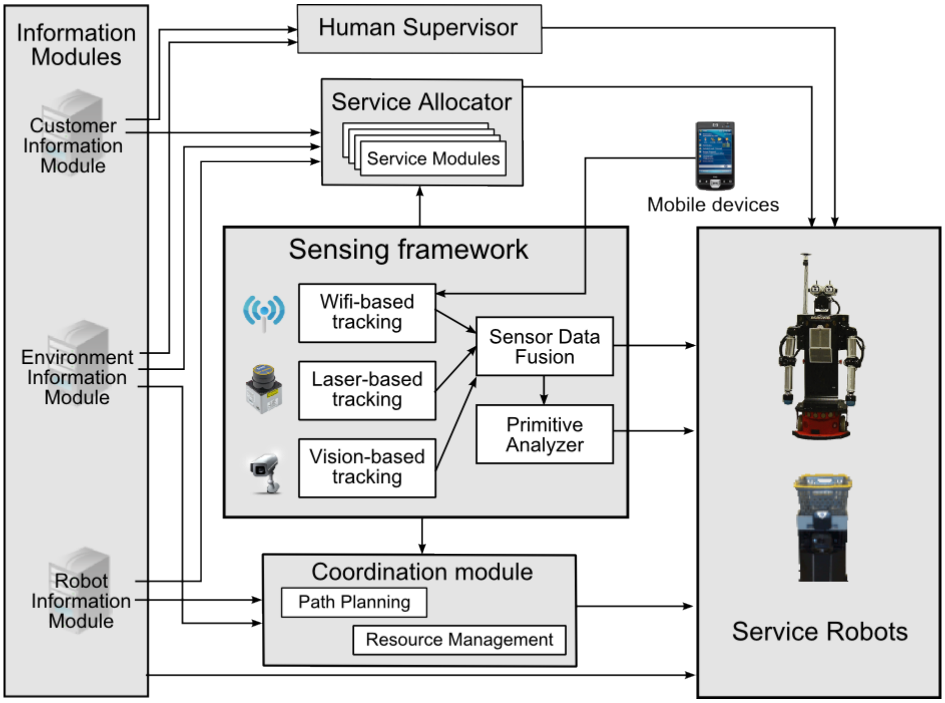


Figure 7.5: Overall architecture of our proposed Network Robot System framework.

tracker (high-precision, but anonymous), together with localization using Wi-Fi fingerprinting (low-precision, but able to identify registered individuals). On arrival at the mall, the customer's smartphone automatically connected to the Wi-Fi localization system, which estimated her rough location. By associating this location with the laser-based tracking data, the sensing framework could determine her precise location, enabling the robot to approach her and offer its service.

Recognizing and anticipating human behavior

Recognition of people's behavior is important in implementing smooth HRI. In particular, we are concerned with position and information derived from position, e.g. "walking fast" (a motion primitive) and "in front of map" (a spatial primitive). In one field trial we deployed robots to offer shop recommendations to visitors. Based on recognition of people's behavior, the robots avoided people who were hurrying and appeared busy, while approaching people who seemed to be walking slowly and might be open to talking with the robot [151].

Anticipation of human behavior is also important – if slow-moving robots can anticipate a person's future behavior, they can start moving early to approach potential customers [139]. The work of [176] has also used anticipation of human behavior to facilitate robot locomotion in crowds.

The presence of an environmental sensing framework enables us to use human motion data observed over long periods of time to generate detailed statistical models describing human behavior which makes anticipation possible (**Table 7.1 – A1**). The *primitive analyzer* performs anticipation using location-based behavior likelihood models and an individual's behavior history over time [81].

In Scenario II, this technique was used to identify that the customer was performing a "stopping" behavior in front of the map.

Assisting robot localization

Inconsistent localization between robots can cause a number of coordination problems. In the example shown in Fig. 7.6 (left), robots R1 and R2 have slight localization errors, perceiving their own poses to be R1' and R2'. Both robots believe they have identified separate people in need of help, but they have actually detected the same person, resulting in multiple robots offering services to the same person, as shown in the photo.

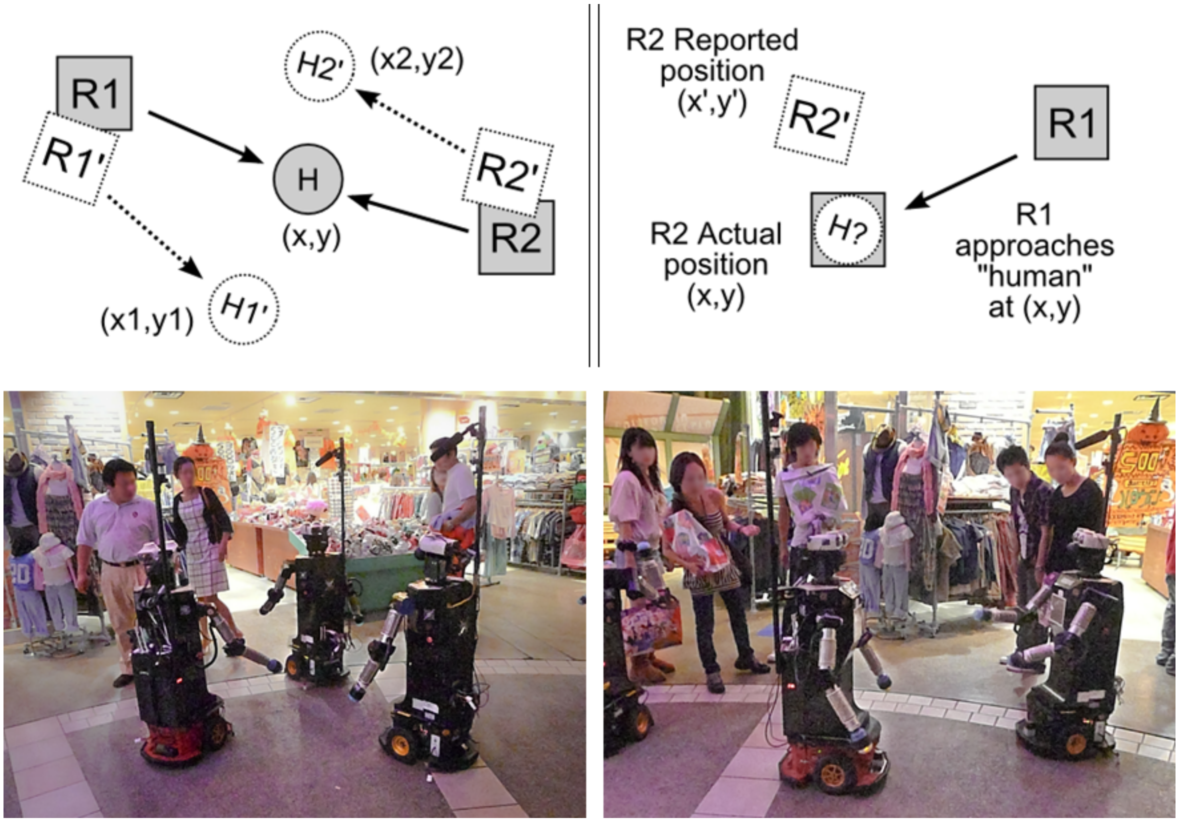


Figure 7.6: Social robot failures that can occur due to localization problems. Left: Example of multiple robots approaching the same person. Right: Example of a robot mistaking another robot for a human and talking to that robot instead of an actual customer.

A similar problem, illustrated in Fig. 7.6 (right) is that one robot could mistake another for a pedestrian and try to initiate a social interaction with it. Since our robots are humanoid in form, they can be mistakenly detected as people by some sensor systems. This has resulted in robots offering services to each other, as shown in the photo. These two problems were common in our early field trials.

Absolute localization in relation to known locations is important as well – knowing that the noodle shop is over *there* and the toilets are over *there* is important for a robot giving directions.

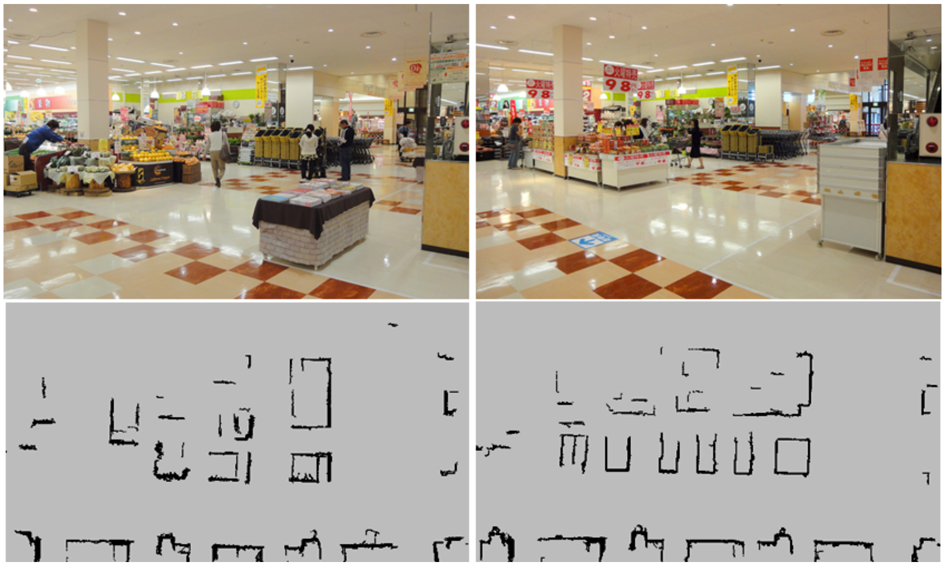


Figure 7.7: Example of an area in a shopping mall where features change from day to day. Top: photos on two different days. Bottom: laser scan maps of the area on two different days.

In public spaces such as shopping malls, features used for map-based localization can change frequently as products and temporary displays are moved, such as the supermarket entrance shown in Fig. 7.7, where we conducted field experiments during 2009-2010 [78]. In such environments, fixed references in the environment can often provide better estimates of a robot's position than the robot can obtain from map-matching. We have often used our human-tracking sensor systems to assist robots with localization by directly tracking the robots in the fixed reference frame of the sensors [53]. (**Table 7.1 – N1**).

7.4.2 Information Modules

Roughly speaking, three categories of information are needed to support a network robot system.

The *Robot Information Module (RIM)* includes information about the capabilities of each robot, which can be used by a central planner for path planning within each robot's mobility constraints and appropriate allocation of robots to perform services. (**Table 7.1 – M2**). In Scenario I, the planner used RIM information to allocate robots to services appropriate to their capabilities, e.g. assigning the cart robot to the baggage-carrying task.

The *Environment Information Module (EIM)* includes navigation and safety maps of an environment, to be used for localization and path planning. In our implementation, navigation maps are generated through offline SLAM using laser scan and odometry data recorded from robots, and safety zone maps are generated by hand. (**Table 7.1 – N2, N3**).

The *Customer Information Module (CIM)* holds personal information about customers (or other service recipients) and is necessary for applications where personalized services are to be provided (**Table 7.1 - P3**). In Scenario III, the customer provides her shopping list information using her cell phone. This information is stored in the CIM alongside her name and known device ID.

A summary of the data stored in these information modules is shown in Table 7.3.

7.4.3 Path Planning and Spatial Allocation

The *coordination module* addresses the needs of path planning and of spatial coordination between robots, considering traversability constraints which may vary by robot type. Some discussion of this module can be found in [151].

Information Module	**Data Provided**
Robot Information Module (RIM)	Services offered Navigable terrain types Maximum clearance Maximum speed Ownership
Environment Information Module (EIM)	Localization map Dangerous areas Traversability map
Customer Information Module (CIM)	Customer name Mobile device ID Application content Personalization data

Table 7.3: Information module data summary

Static space management

Multiple robots trying to go through a physically narrow space such as a supermarket aisle could encounter a deadlock situation (Fig. 7.8). Robots might also compete for spaces based on application needs, such as several cart robots waiting near the checkout counter to offer to

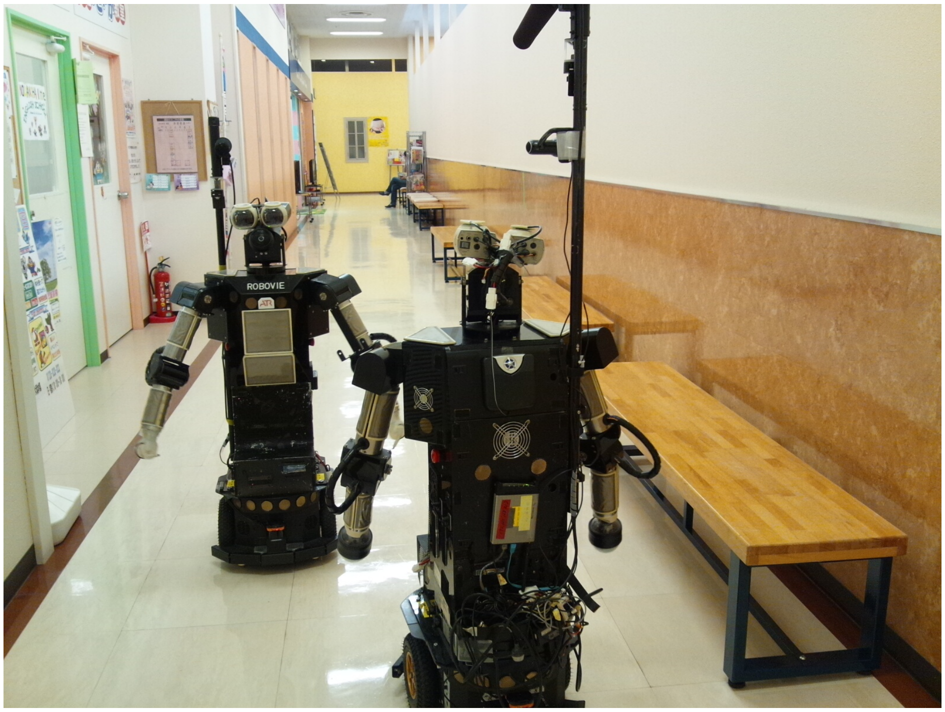


Figure 7.8: Deadlocked robots unable to pass each other in a corridor

carry shoppers' groceries, guide robots trying to occupy the space near a map, or advertising robots crowding near an entrance.

The *coordination module* helps to avoid deadlock and negotiate conflicts between robots for spatial resources (**Table 7.1 – N3**). When a robot requests use of a limited spatial resource, the *coordination module* will grant permission only if the space is not already in use.

Dynamic path planning

Robots operating in public spaces should move smoothly among people, avoiding collisions and strange behaviors like abrupt changes in direction. When information from a wide-area *sensing framework* is available, the *coordination module* can plan paths for the robot that are socially appropriate in the presence of pedestrians (**Table 7.1 – N4**).

When robots request paths to their destinations, the *coordination module* computes an efficient path appropriate for the service. The server periodically updates these paths, giving priority to robots with higher-priority services.

Social factors must also be considered in planning robot motion. For example, approaching someone from a frontal direction has been shown to be more psychologically acceptable than approaching from the side or back [160, 139]. Distance with partners has also been considered in planning [35]. A robot can also communicate intention through its locomotion, as illustrated by the example of “friendly patrolling” [66]. The *coordination module* can generate such special paths, based on the robot’s current service task.

In Scenario I, each robot received the location of its assigned customer from the sensing framework and requested a path to that destination from the *coordination module*. The *coordination module* arranged the paths to achieve best efficiency of the robots (Fig. 7.9). It computed paths for the robots in order of the priority of their services, while avoiding any potential conflict or collision among the robots, enabling each robot to safely reach its customer.

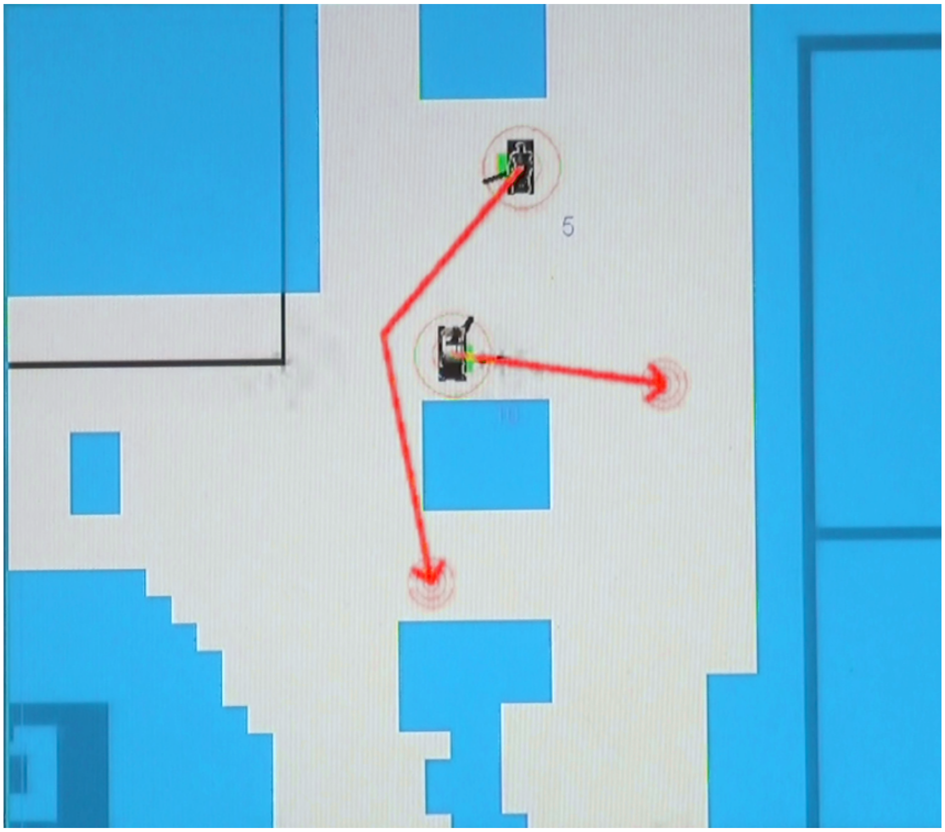


Figure 7.9: Paths planned for two robots in this example

Traversability

The path planner uses information from the *RIM* and the *EIM* to plan paths for each robot based on traversability constraints (**Table 7.1 – M2**). For example, many public spaces have “detectable warning surface” blocks on the ground with raised bumps to help guide vision-impaired people, as shown in Fig. 7.10. Some robots can traverse these uneven surfaces, while others cannot. To address this problem, we create a traversability map for each robot type to be stored in the EIM. When a robot cannot safely traverse a region, it is designated as an "obstacle" zone.

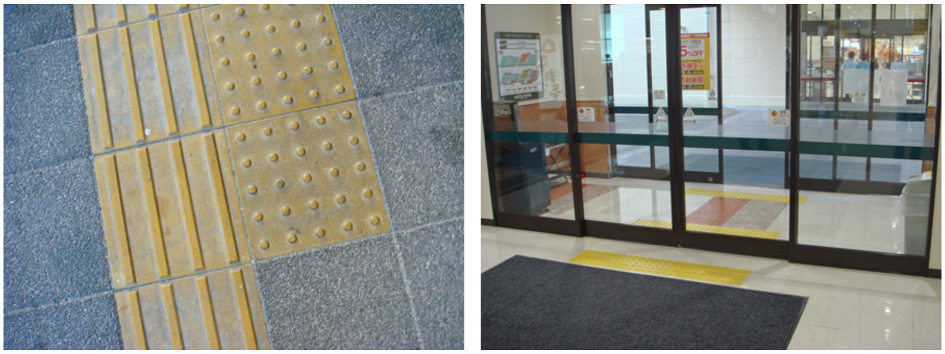


Figure 7.10: Detectable warning surfaces for visually-impaired pedestrians are common in public spaces in Japan. Some robots cannot safely traverse these surfaces.

Other areas may be physically traversable by robots but particularly dangerous – automatic glass doors are an example of this, as it is difficult for a robot to sense whether the doors are open and determine when it is safe to pass. We designate these areas as "teleoperation-only zones" through which robots may only pass if supervised by a remote operator (See Section 4.6).

Fig. 7.11 shows an example of these zones taken from a shopping mall. The left map is for a cart robot, and the right map is for a humanoid robot which is taller, less stable, and unable to traverse uneven surfaces. The corridor going from left to right goes through automatic doors in the center of the map. The map on the right contains "obstacle" areas around the automatic doors, where the floor is uneven. The *coordination module* will not plan any path through these areas for the humanoid robot. In the map on the left, however, such obstacle areas do not exist. The planner may thus plan a path for the cart robot that goes through the center of the map, but the robot will not be allowed to autonomously navigate through the "teleoperation-only areas" – a human operator must be called to assist the robot through these areas.

Safety

For robots operating in proximity to people in public spaces, safety is a very serious concern. While it is most important to avoid harming people, it is also important to avoid damaging the environment or damaging the robot.

To ensure safety from collisions with people, we use standard collision avoidance algorithms locally on the robots. Currently, we are using the "dynamic window approach" algorithm [44]. In crowded spaces, global path planning is used to direct the robots along routes which prevent them from coming too close to people detected by the sensing framework.

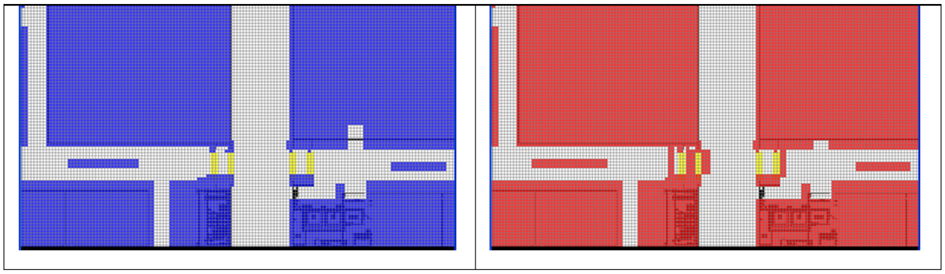


Figure 7.11: Traversability grid maps for each robot type. (Left) Grid maps for cart robot: blue areas represent obstacles, and yellow areas represent teleoperation-only zones. (Right) Grid map for Robovie-II: red areas represent obstacles, and yellow areas represent teleoperation-only zones

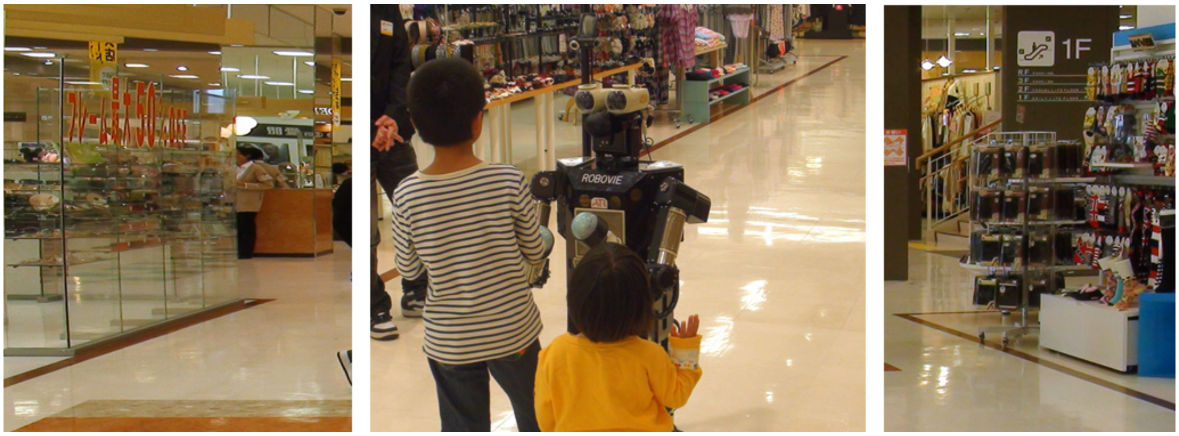


Figure 7.12: Dangerous areas in a shopping mall. Left: glass walls. Center: movable tables and shelves where only legs are visible to ground-level laser range finders. Right: movable clothing rack where only center pole is visible to laser range finder.

Additionally, public environments often contain dangerous areas that a robot cannot detect with its sensors. Transparent and reflective objects such as glass doors and mirrors, or drop-offs such as downward steps, can be difficult or impossible for robots to detect with laser range finders or cameras. Fig. 7.12 shows some examples of obstacles that are difficult for robots to detect with ground-level laser range finders (a typical way for robots to detect obstacles). For safety and planning, maps of these invisible obstacles are provided by the EIM.

7.4.4 Service Allocation

The *service allocator* is the central planning mechanism which assigns services to robots and monitors the execution of those services. It handles service requests, identifies service opportunities, handles reservations for future services, and coordinates service allocation across multiple robots by considering the priorities of services and the capabilities and physical locations of the robots in its allocation algorithm (**Table 7.1 - M1**).

Services

In our framework, each service to be provided by robots is comprised of several *service tasks*, which are execution units managed by the server. The server contains logic determining which service tasks should be executed, under which conditions, by which types of robots. Once the server assigns a service task to a robot, the robot itself handles the details of service task execution, reporting back to the server upon success or failure.

On-demand services For on-demand services, customers need a means to request services directly from the system, e.g. using a smartphone, or from the robots directly (**Table 1 - P2**). In either case, the requests are sent to the *service allocator*. Service requests can be for immediate service, as in Scenario I, or reservations for future services, as in Scenario III, where the customer schedules a shopping trip and reserves a robot to assist her at that time.

Proactive services For proactive services, such as giving directions, recommending shops, or advertising services, the robots will approach unknown people to offer their services. In such cases, the *service allocator* must identify *opportunities* for providing services, rather than responding to requests, and allocation logic must be developed to assign robots to services based on anticipation of who will need or want the service (**Table 7.1 – A2**). To do this, it uses the statistical model of customer behavior provided by the *primitive analyzer*, described in Section 4.1.3.

For example, in the case of robots advertising for a shop, the system could be configured to target customers who are exhibiting “not busy” behavior, in the spatial region in front of the shop. In the Scenario II example, the target behavior was “stopping” in the spatial region in front of the map.

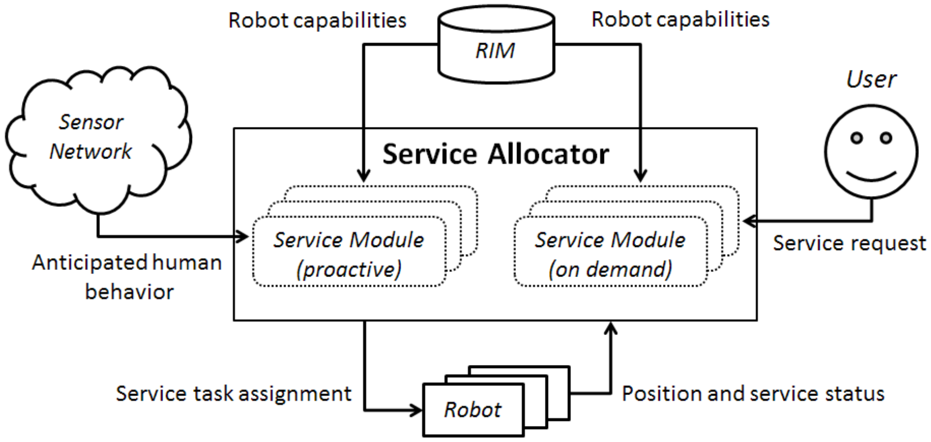


Figure 7.13: Basic flow of service task allocation

The model of global behaviors can be used to predict customers who are likely to perform this behavior in this area several seconds before they arrive, which gives the system time to allocate a robot and for that robot to move into the appropriate area. The robot is then sent to approach the person and give information about the shop.

Service allocator

Figure 7.13 shows the basic flow of service task allocation in the *service allocator*. The allocation logic depends on the specific needs of the service [152]. For instance, a time-critical application would require a robot that is immediately available, but a proactive service would need a robot that could effectively initiate interaction with a customer, e.g. able to approach from a frontal direction even if it required more time [139].

Thus, we designed the framework to enable developers to create their own algorithms for determining when a service should be performed. These are stored in the form of *service modules* within the *service allocator*. The *service allocator* uses the rules stored in its *service modules* to decide the allocation of available robots.

Concretely, the *service allocator* holds information about all available services in a list. For each service, it stores 1) the name of the service, 2) its priority, and 3) a link to a *service module* that handles the initiation of the service. For *proactive services*, the *service allocator* periodically checks the rules of each *service module* in order of priority, assigning service tasks to robots when necessary. For *on-demand service*, it calls the associated *service module* when it receives a request from a user, using the rules stored in that *service module* to determine which robot to assign.

To allocate a service to a robot, the service allocator performs a service matching algorithm, shown in Table 7.4. It queries the RIM for a list of all robots capable of providing that service. Next, the service allocator checks the location and service status (busy or available)

```
INPUT : The list of services service_list ordered by decreasing priority, the list of robots
robot_list, and the service allocator service_allocator

FOR each service in service_list DO
    IF service is proactive THEN
        initiation_rule_is_satisfied = InitiationRuleFromServiceModule(service);
    ELSE
        IF service is on demand THEN
            initiation_rule_is_satisfied = InitiationFromCustomerRequest(service);
        END IF
    END IF

    IF initiation_rule_is_satisfied THEN
        robot_capable_list = GetServiceCapableRobotsFromRMI(robot_list, service);
        final_robot_list = FilterRobotListByLocationAndStatus(robot_capable list);
        selected_robot = RequestRobotForAllocation(final_robot_list, service);

        IF selected_robot exists THEN
            AssignRobotToService(service_allocator, selected_robot, service);
        END IF
    END IF
END FOR
```

Table 7.4: Service matching algorithm

reported by each robot. It provides this list of robots to the service module, which contains logic to decide whether to ask the service allocator to assign the service to one of the robots in the list.

Service modules

When called, a *service module* (contained within the service allocator) performs its own computation, defined by developers. A *service module* for a *proactive service* typically uses information from the sensing framework. For each robot in the given list, it computes whether initiation of the service using the robot would be beneficial. If so, it requests the *service allocator* to assign the service to that robot. For example, a *service module* for an advertisement application might search for a slowly-walking person, who may be a window-shopper, and try to find a robot that can approach that person quickly.

A *service module* for *on-demand* service typically performs a simple computation checking whether robots are available nearby so the service can be provided in timely manner.

Service execution

Service tasks are then executed on the individual robots. Once a service is allocated, the server requests the target robot to perform the first service task, and the robot updates its status to “busy” so that it will be not considered for other services.

The detailed procedure for each service task is pre-installed in each robot, and the service task is executed on the robot side, often using information from the server to provide the service, e.g. the location of a person to approach, or the name of a registered user. The robot notifies the *service allocator* upon completion of each service task, and new service tasks required for the service are assigned until the entire service is complete, and the robot updates its status to “idle”.

Usage examples

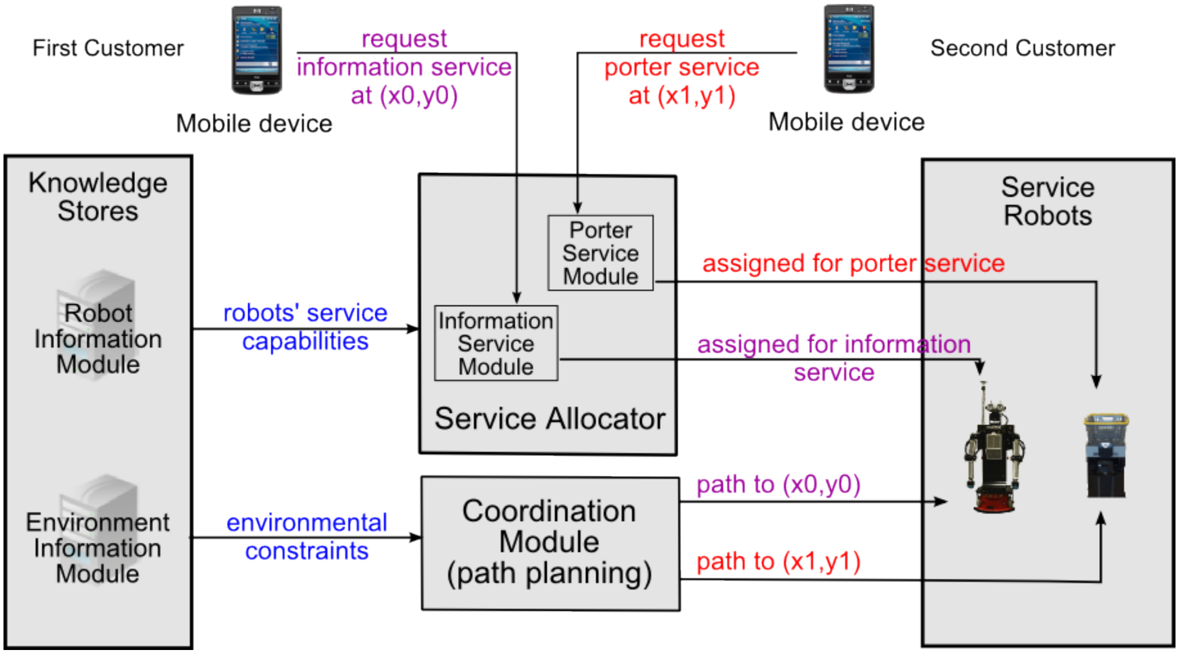


Figure 7.14: Data flow for coordinating multiple robots

In this section, we will describe the service allocation process for the three scenarios presented in Section 7.4.4.

In Scenario I, illustrated in Fig. 7.14, two customers requested different services, and so each request was handled by the corresponding service module. The request from the first customer was handled by the “information” service module. From the list of robots able to provide the “information” service (provided by the RIM), the *service allocator* selected the robot closest to that customer to perform the service, and the *coordination module* planned an appropriate path for each robot.

The request from the second customer was handled by the “baggage-carrying” service module, which likewise selected the nearest robot with appropriate capabilities. The *service allocator* then assigned the cart robot to provide service for the second customer.

In Scenario II, the “route guidance” service module contained logic designating the space in front of the map as an area where “stopped” people would be likely to need the route guidance service. Thus, when the *primitive analyzer* reported a customer “stopping” at that location, the service module selected a robot for the “route guidance” service based on its capabilities and position to approach the customer and offer its service.

In Scenario III, the customer reserved a robot for a future shopping trip. Slightly before the appointed time, the *service allocator* assigned a robot to the “shopping support” service, beginning with a “wait for customer” service task. When the sensor network detected the customer’s arrival, the *service allocator* assigned the robot its next task, “approach customer,” to begin its service.

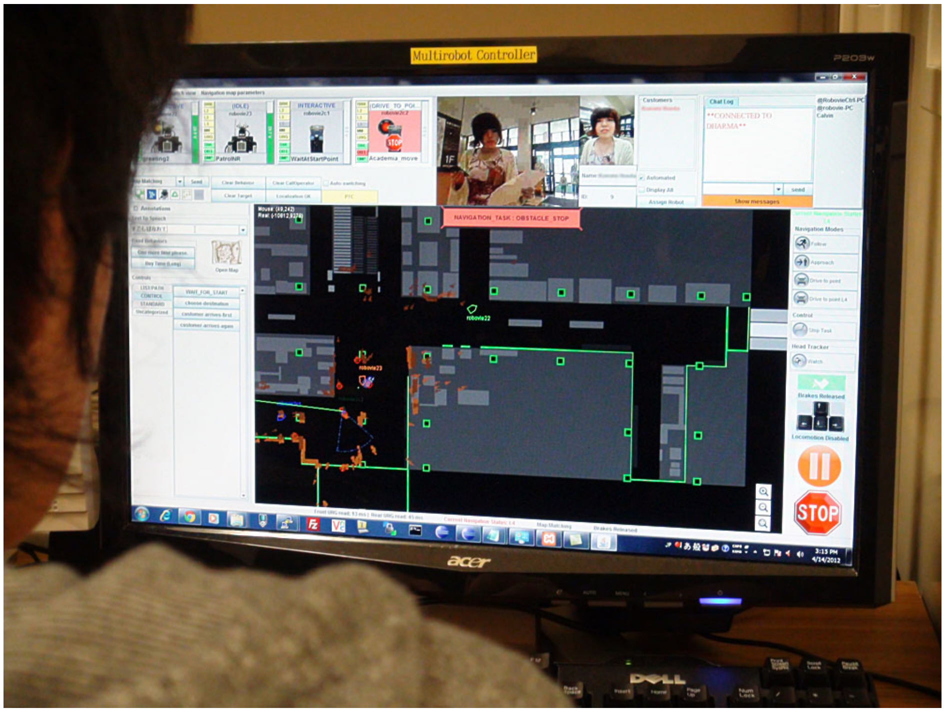


Figure 7.15: Operator using a teleoperation console to supervise four robots.

7.4.5 Support From a Human Operator

While robots today have greater capabilities for sensor recognition, dynamic planning, error detection, and error recovery than ever before, they are still far from ready to be deployed autonomously alongside humans in unstructured, public environments. For the foreseeable future, we expect that there will always be a human supervisor present behind the scenes to monitor and assist robots at some level.

In our system, supervisors typically use an interface such as that shown in Fig. 7.15 to assist the robot's recognition of sensor inputs, e.g. speech recognition or person identification (**Table 7.1 – C1**), and to monitor for and correct sensing errors, e.g. identifying dangerous situations or correcting localization (**Table 7.1 – S1**). The robot performs its own speech, gesture, and motion planning autonomously, and the role of the human is only to provide occasional sensor inputs.

In rare cases, an operator will need to control the robot directly to handle "uncovered" situations, such as an unexpected question from a customer (**Table 7.1 – C2**), or replanning the robot's path to avoid unmodeled obstacles (**Table 7.1 – S2**). In these cases, the robot cannot respond autonomously, so the human controls the robot directly.

Finally, some mechanism is needed to enable one operator to manage multiple robots (**Table 7.1 – S3**). Techniques such as *proactive timing control* (Glas et al., 2012) and *conversation fillers* (Shiwa et al., 2009) can help improve performance of semi-autonomous robot teams in social interactions. These and other issues regarding supervisory teleoperation of multiple social robots are discussed in more depth in (Glas et al., 2012).

7.5 Field Evaluation: Deployment in a Shopping Mall

Finally, to demonstrate the effectiveness of our NRS approach in enabling social robot applications in the field, we deployed four robots to provide services in a shopping mall.

7.5.1 Field Environment

The evaluation was conducted in the section of a shopping mall shown in Figure 16. We set up a laser-based human tracking system in a high-traffic area near the entrance of the shopping mall, to identify pedestrians for the robots to approach, and to assist in localization of the robots.

The experiment was performed over a span of 3.5 hours on a Saturday afternoon in April. The day was not crowded, but the flow of pedestrians was steady. Fig. 7.17 shows scenes from the field trial.

7.5.2 Robot Services

For this evaluation, we implemented both proactive and on-demand robot services to demonstrate the flexibility of our framework.

Two kinds of robots were deployed: two Robovie 2 humanoid robots and two cart robots, shown in Fig. 7.17. The humanoid robots provided conversation-based services: for adults, the robots gave directions to locations in the mall, and for children, the robots performed

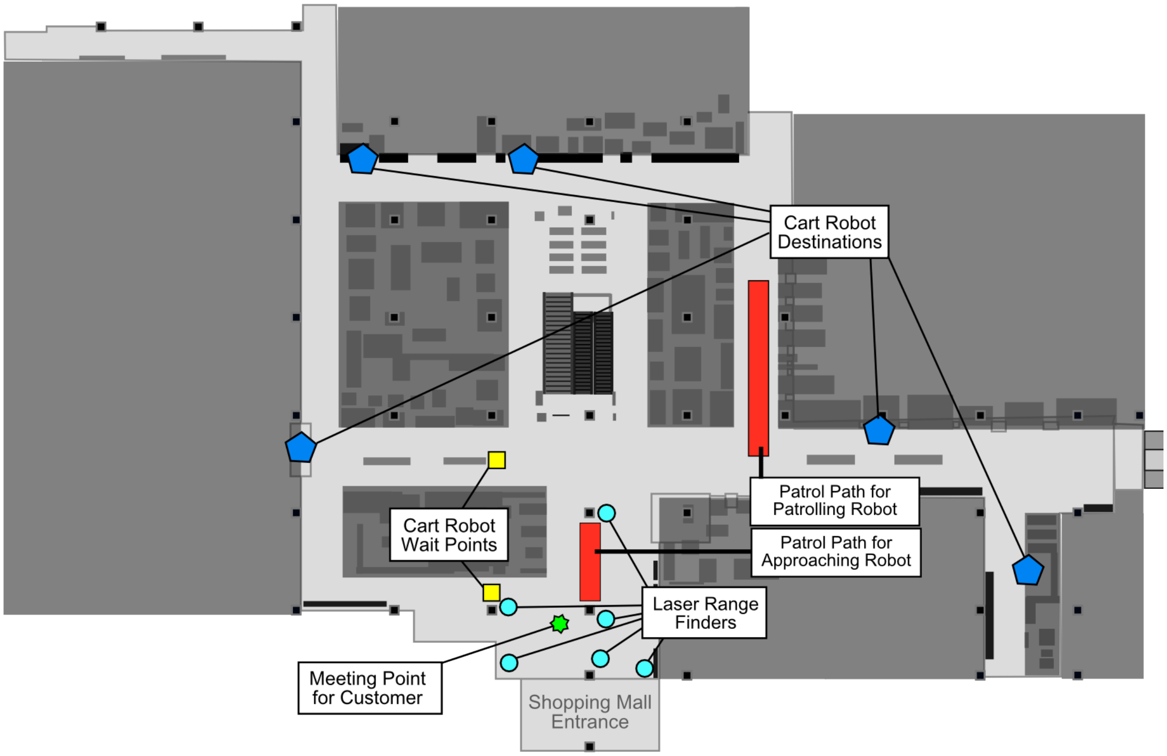

Cart Robot Destinations
Cart Robot Wait Points
Patrol Path for Patrolling Robot
Patrol Path for Approaching Robot
Laser Range Finders
Meeting Point for Customer
Shopping Mall Entrance

Figure 7.16: Map of the shopping mall where we conducted our field trial.

entertaining play behaviors such as “rock, paper, scissors” and a guessing game. The cart robots provided an on-demand baggage-carrying service, carrying a customer’s bags to various destinations in the mall.

The state transition diagram in Fig. 7.18 shows how these services were implemented in terms of “service tasks” assigned by the service allocator.

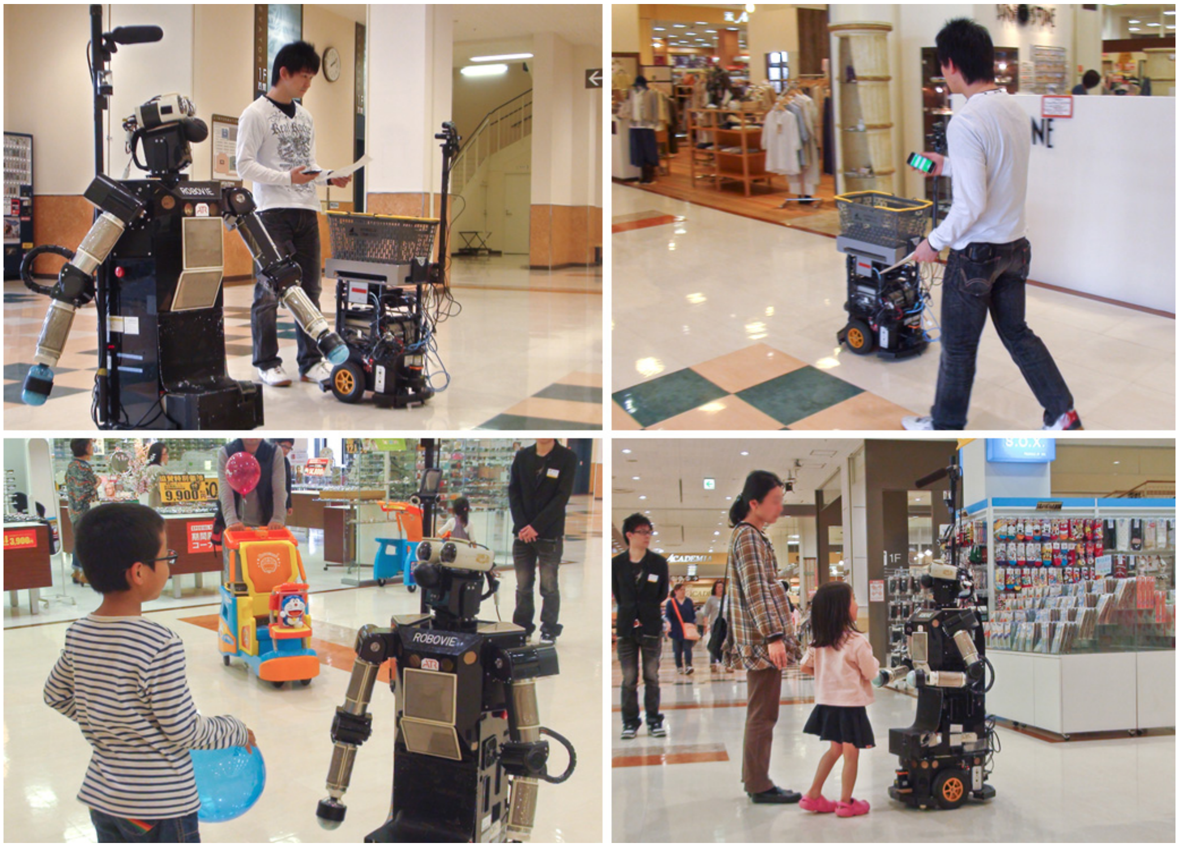


Figure 7.17: Scenes from our field trial. Top: A participant interacting with a cart robot. Bottom: Shopping mall customers interacting with Robovie.

Anonymous service without sensor network support

One of the two humanoid robots (the “patrolling robot”) was placed in the shopping area and set to patrol along a fixed route, relying only on its on-board sensors for localization and human detection. This condition was chosen to illustrate the flexibility of the NRS framework – not all service areas can be covered by sensor systems, but the other functions of the framework can still be used in these areas.

For this service, the central server provided path planning, and the human operator assisted the robot with localization and speech recognition, but service allocation was handled locally – when a customer stopped in front of the robot, it detected them and offered its services, reporting updates in its service state to the *service allocator*.

Context-aware service using the sensor network

The other humanoid robot (the “approaching robot”) was placed in the entrance area covered by the human tracking system. Based on the information from this sensor network, the robot was assigned to actively approach customers to offer its services.

The *primitive analyzer* was used to predict which customers would be good approach candidates. The *coordination module* then calculated an approach path for the robot. The human operator assisted with speech recognition and localization support, and the robot was also able to use the sensor network to assist in localization and tracking people.

Personalized service: Baggage-carrying robots

Two non-humanoid cart robots, shown in Figure 7.17 (top), were also deployed. These robots were equipped with a camera, speaker, and microphone, for simple conversational interactions. They provided a baggage-carrying service to registered customers who requested the service from their mobile devices.

Baggage-Carrying Robots	
Number of interactions	30
Total distance traveled	3564 m
Participants reporting positive impressions	73%
Conversational Robots	
Number of interactions	27
Participants reporting positive impressions	68%
Sensor network	
Anonymous people tracked	431
Registered ID’s detected	15

Table 7.5: Summary of results from the field trial.

The *coordination module* computed paths for the robots to requested destinations around the shopping mall, using safety and traversability maps from the EIM, and it dynamically allocated standby locations for the carts, for idle times between customer requests.

The CIM provided photos for the operator to use in confirming the identity of the customer, and it provided the person’s name so that the robot could greet the customer personally.

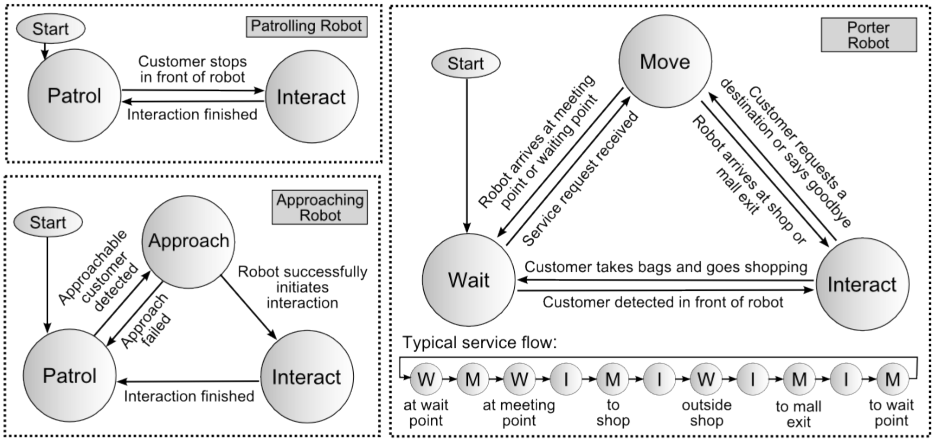


Figure 7.18: State transitions for service tasks in our field trial.

The sensor network provided identification and coarse localization for the customer carrying the smartphone, and forwarded the customer's requests to the service allocator.

The human supervisor assisted the robots with speech recognition, localization, occasional obstacle avoidance, and confirming the identity of the registered customers.

7.5.3 Procedure

Proactive services

The humanoid robots approached and interacted with real customers in the shopping mall. After each interaction finished, a member of our laboratory briefly interviewed the customer to learn their impression of the robot and the quality of its service.

Personalized services

Since use of the personalized cart service required registration in the system and installation of a mobile application, it was impractical for us to test the service with general customers for a short-term deployment. Instead, we hired 15 participants to interact with the robots and give us their opinions of the service. Participants were native Japanese speakers recruited through an online service, 8 male and 7 female, average age 25.3 years, s.d.14.2 years, paid 1000 yen per hour. Six had previously interacted with robots.

Each participant was instructed to use the cart's services twice, requesting a different destination each time, placing their baggage on the cart, and walking with it to the destination as if they were really using the service while shopping. After the participants had completed this task, they were interviewed to learn their impressions of the robot and the quality of its service.

7.5.4 Allocation of Functionality

To provide these services, functionality was distributed between the robot system, the NRS servers, and the human operator.

Server Tasks

The Service Allocator assigned services to robots. For the cart robots, this was done based on information from the participants' mobile devices, and for the humanoids it was done based on detection of people, either by the environmental sensor system in the case of the approaching robot, or by the on-board detection system in the case of the patrolling robot.

The Coordination Module computed paths for all robots, based on destination requests from the baggage-carrying robots, or based on human positions for the approaching robot. It sent a fixed path to the patrolling robot.

Robot Tasks

The robots executed on-board logic for carrying out the services assigned to them. The contents of these services were designed as conditional flowcharts, using the "Interaction Composer" software presented in (Glas, Satake, Kanda & Hagita, 2011). When a path was needed or a service phase was completed, the robot would notify the central servers.

The robots also performed on-board sensing to detect nearby people and obstacles. For the patrolling robot, when a person was detected, the robot sent a request to the Service Allocator to begin providing a service to that person. For the other robots, the detection of a person was used in on-board processing for service execution.

Operator Tasks

The tasks of the human operator were as follows:

Confirm the identities of customers. For the cart robots, the operator matched a photo of the customer to the video feed from the robot's camera, to guarantee proper user identification for personalized services.

Perform manual speech recognition. At each point where the robot's internal behavior flow expected a speech input, the operator was presented with a list of valid candidates (e.g. "yes" and "no"), and the operator would listen to the audio stream from the robot's microphone and click the button corresponding to what was said.

Select guide destinations from a map. When the humanoid robots offered directions to locations, the operator would listen to the requested location and click the appropriate spot on a map. We have found this interface to be more effective than selection from a list, which is used for other speech recognition tasks.

Correct the robot's localization. The operator periodically corrected the robot's position on a map if the robot was improperly localized.

Assist the robot in obstacle avoidance. The operator sometimes manually controlled the robot's locomotion for short periods of time if the robot became stuck due to objects in the environment, such as baggage carts or baby strollers stopped in its path.

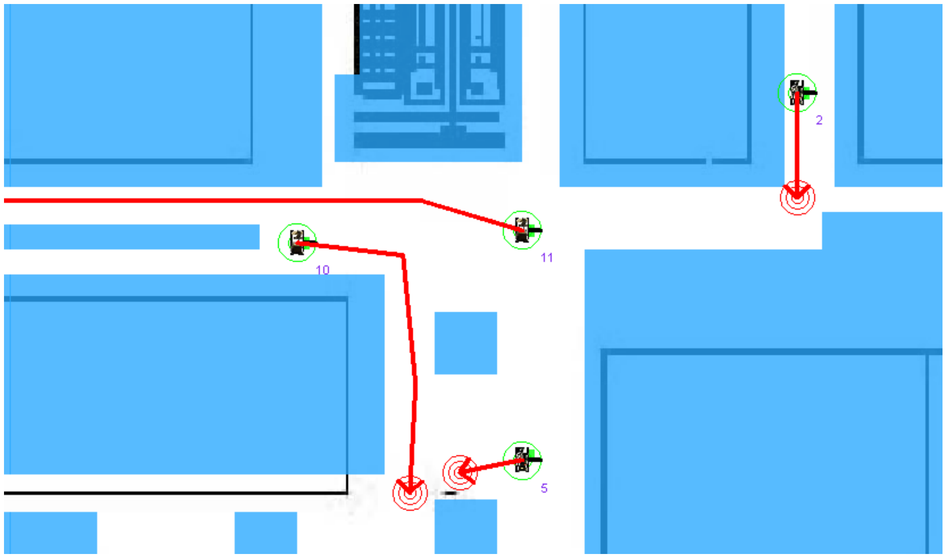


Figure 7.19: Example of paths generated for four robots during our field evaluation.

7.5.5 NRS Framework Achievements

Overall, the NRS framework successfully enabled four robots to simultaneously perform services in a commercial space, taking advantage of centralized planning, sensor networks, and the availability of a human operator. Various metrics of the system's performance are described here and summarized in Table V.

The human-tracking system and *primitive analyzer* identified 431 trajectories of people walking through the entrance area over the 3.5-hour span of our experiment – an average of about two people per minute. The Wi-Fi device tracker successfully identified the participants' mobile devices and successfully received 100% of service requests, enabling the *service allocator* to send a robot whenever a customer requested one.

In response to 30 service requests, two from each of the 15 participants, the *service allocator* successfully allocated a cart robot to the customer every time. It also successfully directed the humanoid robots to initiate 27 interactions with anonymous customers. One interaction is defined as a complete conversation with a unique customer.

The *coordination module* successfully guided the cart robots to and from 30 destinations, traveling a combined total distance of 3564 m. Fig. 7.19 shows an example of the paths generated for the four robots during our field trial. For the approaching robot, the path planner provided successful approach paths to talk with 18 people.

7.5.6 Customer Impressions

Of the 42 people who interacted with the robots (27 anonymous shopping mall customers who freely interacted with the humanoids, and 15 hired participants who interacted with the carts), we were able to interview 40 about their impressions of the robots. These interviews were recorded and transcribed, and two independent coders categorized the responses from the customers. To measure coding consistency, Cohen's kappa was evaluated for each set of responses coded, and then the coders were asked to discuss and reach agreement on any responses they had disagreed upon, to produce a final, consistent data set.

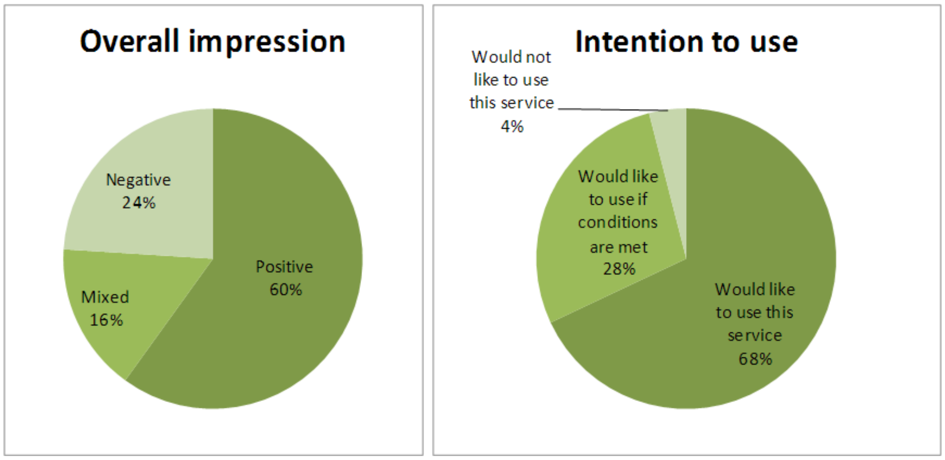


Figure 7.20: Interview results from 25 shopping mall customers who freely interacted with the humanoid robots.

Humanoid robots

For the 27 anonymous customers who interacted with the humanoid robots, we were able to interview 25. We asked these customers two questions: (1) what were their impressions of the robots, and (2) whether they would like to use such robot services if they were available. Results are shown in Fig. 7.20.

Impressions from their interactions with the robots: A majority of 76.0% (19/25) of the customers reported positive feelings, including interest, happiness, and amazement. 40% (10/25) reported negative feelings, which included frustration with the conversational interaction, and a comment from one customer that the robot's appearance was scary. Cohen's kappa for this question was 0.83, indicating very good agreement between the coders.

Aside from general positive comments like saying the robot was "fun" or "interesting," four customers commented that they liked that the robot made eye contact and followed them with its gaze. By contrast, two customers gave negative comments that the robot was not making eye contact with them but was looking somewhere else. These remarks underscore the importance of gaze control in conversational interactions. Also, one customer specifically commented on the robot's approach behavior, saying that the fact that the robot drove up to him to speak created a friendly feeling.

Five customers reported that the robot's response speed was too slow, most likely due to the operator taking a long time for some operation, e.g. finding a shop on the map.

Other negative comments included, "the robot seemed condescending," "the robot's appearance is too angular and scary," and, "I felt self-conscious while talking to the robot in front of other people, as if I was talking to myself." These comments suggest areas where the robot's behavior and appearance could be improved for better interactions.

Intention to use this robot service if it is available in the future: An overwhelming 96.0% (24/25) of the customers who interacted with the robot stated that they would like to use this type of robot service in the future, although some (7/25) said this was contingent upon improved conversational capability of the robot. Cohen's kappa was 0.65, indicating good agreement.

Three customers suggested that the robot would be particularly useful for first-time visitors to the shopping mall, and seven thought that the robot would be particularly fun for children to talk to. Two suggested that it would be useful for elderly people, although one of them raised the concern that speech recognition would be particularly difficult for elderly customers, since they tend to speak in heavy local dialects.

Baggage-carrying robots

For the 15 paid participants who used the cart robots, we asked three questions: (1) what were their impressions of the robots, (2) whether they would like to use such robot services if they were available, and (3) how they felt about using the smartphone interface to interact with the robots. These results are shown in Fig. 7.21.

Impressions from their interactions with the robots: Most participants reported both positive and negative feelings about the baggage-carrying robot service. In total, 66.7% (10/15) reported positive feelings (happiness, friendliness, amazement, and approval), and

86.7% (13/15) reported negative feelings (anxiety, frustration, and difficulty in communicating with the robot). Cohen's kappa was 0.67, indicating good agreement between the coders.

Of the negative comments, the most common complaint (6/15) was that the robot's speed (0.75 m/s) was too slow. Other complaints were about the robot's speech itself – three people said that its pronunciation was strange, and three said that its response was too slow. Two said that they enjoyed talking with the robot and wished they could have talked with it more (the cart robots spoke very little, and spent most of their time driving to the destination in silence).

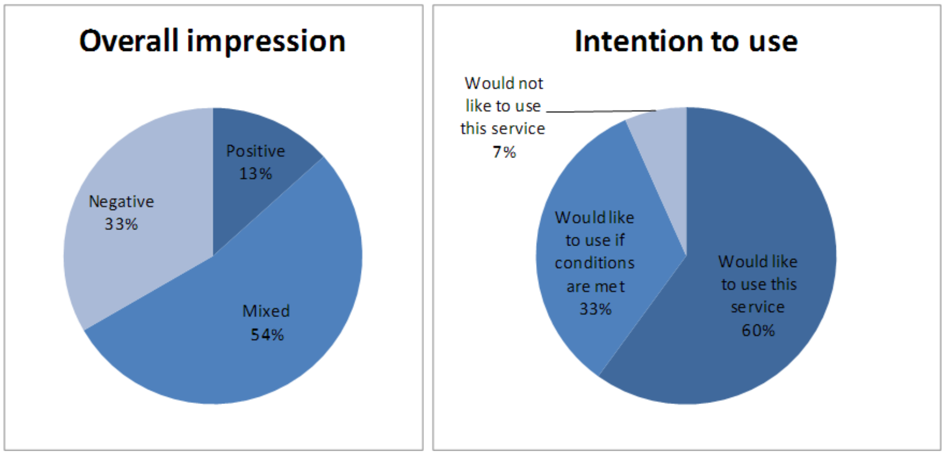


Figure 7.21: Interview results from 15 hired participants who interacted with the baggage-carrying cart robots.

We had expected that a personalized service that called the customer by name would get a positive response, and indeed, three participants did say they were happy that the robot called them by name; however, another said, "I was a bit embarrassed when the robot said my name very loudly," indicating that robot behaviors should be designed carefully not to make people feel self-conscious if other people might be listening.

Finally, three participants said the conversation seemed mechanical and lacked feeling, and that the interactions seemed artificial or awkward, e.g. "the cart has no feeling when it talks, so when it says 'thank you' I don't feel anything." It is possible that these comments reflect an effect similar to the findings in [78], where customers were more comfortable with a humanoid robot than with a cart robot as a shopping companion.

Intention to use this robot service if it is available in the future: The response of the participants to this question was overwhelmingly positive: 93.3% (14/15) of participants indicated that they would like to use such a robot in the future, although 33.3% (5/15) said so only under certain conditions, such as "if the system is improved," "only when I have heavy baggage," and "when I am elderly, but not while I am young." Many of the participants (5/15) said in particular that they thought this service would be useful for elderly people or people with disabilities. Cohen's kappa for this question was 0.76, indicating very good agreement.

Feelings about interacting with the robot using their mobile phone: For the cart robots, we asked an additional question to get feedback on people's impressions of the mobile phone interface. In this case, 73.3% (11/15) of participants reported positive feelings, including ease of use, convenience, and approval. The other 26.7% (4/15) reported negative feelings, such as frustration with waiting for the robot to arrive after requesting the service, or the necessity of owning a smartphone to use the service. Cohen's kappa for this question was 1.0, indicating perfect agreement.

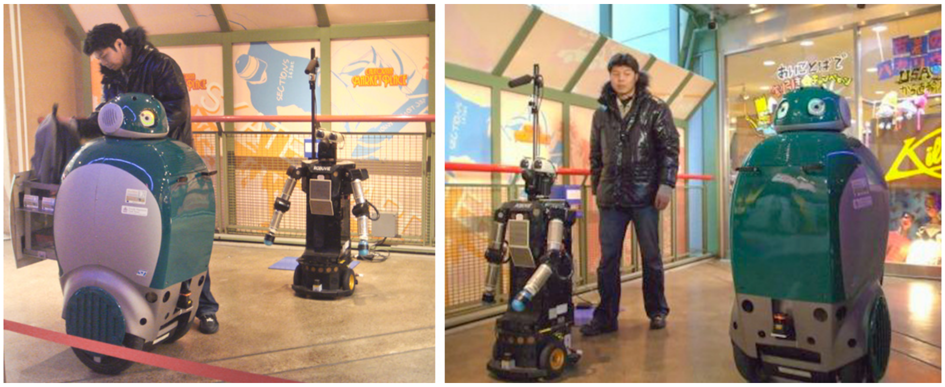


Figure 7.22: Collaboration between Robovie and DustCart, supported by the NRS.

7.5.7 Discussion

Cross-platform collaboration

One strength of our NRS framework is the ability to easily integrate new robots, new sensors, or new services to the system. Using simple protocols to connect to the planning, coordination, and localization servers, third-party robots can be integrated easily into the NRS without deep or complex software integration, enabling collaboration between very different kinds of robots.

We have conducted collaborative work with the DustCart robot from the EU Dustbot project (Fig. 7.22), and with Honda's ASIMO robot (Fig. 7.1). In these demonstrations, Robovie talked with visitors and initiated a collaborative task, wherein the other robot performed some physical task, e.g. serving a drink or carrying baggage, while Robovie continued talking to the visitor, offering chatting or verbal instruction.

In each case, only 1-2 weeks of implementation and testing were necessary to integrate the new robots with the NRS platform and prepare a collaborative robot demonstration. We have also conducted other NRS demonstrations with robots such as Mitsubishi's Wakamaru and Toshiba's ApriPoco. A ROS interface component has also been developed for the latest version of the platform.

Practical considerations

Aside from the target functionalities presented in this paper, our experiences have shown a number of practical benefits provided by the modular design of the NRS framework.

When robots experience hardware problems – hard drive crashes, electrical failures, etc., the modularity of the NRS framework makes it easy to swap a backup robot into the system, enabling the experiment or demonstration to continue in a nearly seamless way. Rather than statically specifying services, paths, etc. within individual robots, the NRS dynamically allocates paths and services, which minimizes the settings or code that need to be modified when the composition of the robot team changes.

We have even replaced robots with different robot models – in one field trial we had some hardware problems with a Robovie-R3 robot, and we were able to seamlessly replace it with a Robovie-R2 (a robot with a very different design) for an important demonstration. This was possible because differences in hardware components and internal implementations of gestures and poses are hidden beneath the abstraction layer of "service tasks," enabling the different robots to operate interchangeably within the network robot system.

The addition of new sensor types is also facilitated by our modular design. We have developed several versions of our human-tracking system, e.g. using RGBD sensors. Although developed by an independent team, these new sensor systems can be seamlessly used with our robots if they support the data protocols in our NRS framework design. This flexibility has been extremely helpful in managing the complexity of heterogeneous robot deployments in multiple environments with different sensor systems.

Finally, the EIM makes it possible to easily switch environments. We often move robots between our lab and various field trial environments, and the local NRS at each environment enables the robots to automatically make use of the latest navigation maps and receive path planning and service allocation for that environment.

7.6 Conclusions

We have presented a framework for a "network robot system," in which mobile robots, planning servers, and sensors embedded in an environment are integrated to provide robot services to people in social contexts. The requirements for this framework, motivated by our experiences in several years of field trials, primarily include the need for recognition and anticipation of people's behavior, identification of individuals, coordination of services and navigation paths between robots, and supervision by a human operator.

We presented a field experiment showcasing the capabilities of this framework by providing services with four robots in a shopping mall, and our results showed that not only was the technical framework successful in supporting the robot services, but that people who used the robots responded in a positive way, with a great majority indicating that they would like to use services like these in the future. This underscores the worth of conducting research and field experiments to investigate and develop social robot services in real-world environments, and we submit that the NRS approach is an effective and practical way to make such robot deployments a reality.

Chapter 8

Discussion and Conclusions

8.1 Summary of Achievements

This thesis has explored many aspects of the problem of how to enable the deployment of social robots in real-world environments. The essential points that have been presented are as follows:

- The use of external sensors to provide robust, high-precision tracking of people in the environment to assist robots in navigational interactions.
- Development of empirical models of social behavior in a space, to enable anticipation of people's future behavior.
- The use of a human operator to assist a team of robots with difficult recognition problems and "uncovered situations," incorporating the "Proactive Timing Control" technique to ensure smooth operation.
- A study of user interface design to assist an operator's time perception and to minimize the risk of unacceptable delays during conversational interactions.
- A modular design framework to enable design and maintenance of interactive robot service applications by teams of designers and programmers.
- A framework for robot coordination, service allocation, and knowledge sharing to support the operation of heterogeneous teams of service robots.

All of the elements in this "Network Robot System" framework have been tested through many field experiments and demonstrations, to provide many different kinds of services using several different robot platforms.

Many roboticists may disagree with this proposed design. Certainly this approach departs from what some may consider an ideal in robotics: the intelligent, autonomous, self-contained robot. I would argue that the use of external sensors and planning servers is not so different from the use of GPS for automobile navigation, and that the trend of using offboard sensors will grow, not only for robots, but for human beings in our daily lives as well.

The use of a human operator for recognition support is probably more controversial. The "Wizard of Oz" technique is seen as a tool which can sometimes be justified for research, but not as a replacement for autonomy in real robots. While I agree in principle with the idea that the operator should not be necessary, I feel that the most practical approach is to initially include an operator in the loop, to enable safe and reliable robot services, and then gradually work to increase the autonomy of the robots until the operator exists merely as a supervisor to ensure safety. This achieves the same eventual goal of having fully-autonomous robots, but my proposed approach should enable robots to be deployed usefully in the world long before all the necessary technologies are ready.

8.2 Future Directions

One of the strong motivations for this work was the fact that practical deployment of autonomous, standalone social robots is not feasible with today's technology. How, then, will the proposed framework evolve as robot technologies become more mature?

8.2.1 Operator's Role

Perhaps the most significant change may be the role of the operator. As the technology for speech recognition and robot localization improves, the responsibilities of the operator will most likely shift towards handling only rare "uncovered" situations and monitoring robots for safety and quality of service. Such a high-level supervisory operator could potentially manage large teams of 20 or more robots.

8.2.2 Ambient Intelligence

Several trends will likely affect the direction of the ambient intelligence portions of the NRS. First, sensors are becoming smaller, cheaper, and more powerful. Second, people-tracking, computer vision, and object recognition algorithms (and the hardware they run on) are becoming faster and more accurate. Third, mobile devices with embedded sensors such as accelerometers and cameras are becoming more ubiquitous, suggesting the possibility that at some point pedestrians themselves could be considered a part of a sensor network.

These advances will make it easier to collect human motion data for modeling public spaces, and vast amounts of data may become easily available in online repositories, enabling a much more detailed analysis and more accurate prediction of human behavior patterns.

Detection of features aside from position will become possible. By using sensor data to infer social relationships, identify objects being carried, and perhaps even detect a person's mood, it would be possible to much more accurately target people interested in a service, and to customize robot services specifically for those individuals.

3D mapping is another area in which great strides are currently being made. In the near future it is likely that the robots themselves will be able to collaboratively maintain dynamic three-dimensional maps of public spaces in a robust way, perhaps eliminating any need to rely on environmental sensor networks or the assistance of an operator for localization.

8.2.3 Network Reliability

One of the weakest links in a Networked Robot System is the network itself - in our field trials in shopping mall environments, this was a major source of problems and system failures. Major sources of interference came from microwave ovens, from nearby electronics stores, and from groups of children playing with wirelessly networked handheld video game consoles.

Doubtless, as wireless devices become more ubiquitous in our daily life, higher-bandwidth and more robust connectivity technologies will become available. However, it might be prudent for commercial NRS deployments to enable backup frequencies in case of network interference, as well as explicitly considering the possibility of connectivity loss in the design of the robots and their service applications.

8.2.4 Central Planning

As the scale of NRS deployments grows, it is interesting to consider how the path planning and service allocation approaches proposed in Chapter 7 scale to handle large numbers of robots or large environments.

For a very large scale system, required storage for the information modules would increase, sensor processing would need to be distributed across multiple servers, and some mechanism would need to be developed for managing large maps (similar to Google Maps for extremely large service areas). Computation for path planning is based on priority, and so it should increase linearly with N (the number of robots). Perhaps service allocation would be heaviest, increasing with the number of idle robots N and the size of the space S, *i.e.* order of SN. However, since this computation is simple, one server should be able to handle a space the size of several buildings.

Thus, the general architecture of the service allocation and coordination modules could remain essentially the same, despite expansion of service areas and number and variety of robots.

8.2.5 Application Design

It is hard to imagine how far the basic approach to application design recommended in Chapter 6 will scale. Most likely, the basic flowchart approach will continue to be useful, but in conjunction with other techniques such as learning-by-demonstration. Such combinations could capitalize on the clear readability of functional flow diagrams, while also taking advantage of the ability to specify subtleties of body movement, positioning, and timing, of which even a human performing the task might be unaware.

Another possible direction in which the application design approach might evolve is towards the model of mobile applications, in which independent program developers create robot services for download from a shared marketplace. These could take the form of entire service flows or individual behavior modules for inclusion in larger flows.

8.2.6 Cloud Resources

At the time of writing, online data stores such as RoboEarth [1] are already being prepared to provide cloud-based knowledge support for networked robots. Such resources could be incorporated into a NRS architecture, providing support for robot tasks such as object recognition or providing updatable procedure scripts.

On the other hand, some knowledge stored in the CIM, RIM, or EIM could be highly proprietary to the owners of a NRS, whereas other knowledge, such as map data, might be shareable or even outsourced to external services. Careful consideration will need to be given to levels of privacy and ownership of information when, for example, one organization licenses robots to multiple businesses.

Another online resource concept is the idea of “crowdsourcing” - outsourcing tasks to distributed groups of people on the internet. This is a growing technique which may also be an interesting option to consider as an alternative to a full-time operator for simple tasks like speech recognition.

8.3 Conclusion

To conclude, I have presented a Network Robot System approach enabling the deployment of social robots to provide services in real-world environments. This work has focused on several critical elements required to enhance the capabilities of the robots, ensure safety and stability, and coordinate teams of robots.

The contribution of this work lies not only in the theoretical proposal of such a system, but also, and especially, in the practical demonstration of its implementation and use in real field trials. Unforeseen difficulties uncovered through these practical experiences, such as the temporal awareness problem of the operators or the need for robots to anticipate people’s motion in order to approach them effectively, created the motivation for further studies and improvements to the system.

I have demonstrated stable implementations of each of the systems presented in this work through a series of robot demonstrations and long-term deployments in several field environ-

[1] RoboEarth. http://www.roboearth.org

ments over a period of 4-6 years. The effectiveness of these systems has been demonstrated in the field, and the field experiences have contributed to the direction of this research. All of the systems presented here are still in use as framework for supporting a variety of research in human-robot interaction.

Thus, this work provides a successful, concrete example of a coherent collection of systems for human tracking, behavior analysis and anticipation, supervisory teleoperation, interaction design, and robot service coordination, which together enable the practical deployment of teams of social robots to provide services in real-world environments.

Acknowledgments

I would like to thank my advisor, Dr. Hiroshi Ishiguro, who has greatly inspired me through the audacity of his ideas and his vision of pushing the boundary between science and art. I would like to thank Dr. Takayuki Kanda for his guidance and insight over the last five years. He has taught me so much about research, from experimental design through publication, and although we have had many conflicting opinions in our meetings, he has always helped me navigate the road from "interesting idea" to "journal publication." Many thanks also to Dr. Takahiro Miyashita and Dr. Norihiro Hagita for their wisdom and advice, not only in research, but in life. I would also like to thank Dr. Tatsuo Arai and Dr. Shogo Nishida for contributing their time as readers for my thesis.

Next, this work would not have been possible without the tireless work of my colleagues, Dr. Masahiro Shiomi and Dr. Satoru Satake, who have contributed enormously to the success of our field trials and the evolution of the Network Robot System into what it is today. Thanks also to the staff and management of the APiTA Town Keihanna shopping mall and Universal CityWalk Osaka, and great thanks to Dr. Satoshi Koizumi for his help in negotiating and coordinating many of our field experiments.

I would also like to thank my team: Benoit Toulmé, Phoebe Liu, Kuanhao Zheng and especially Florent Ferreri, who have been at my side for much of this journey. Thanks also to the interns and researchers who have come and gone - Greg Cole, Peace Cho, Dan Klein, Calvin Lam, Andres Mora, Kyle Sama, Yunchen Fu, Rémy Burel, and Baptiste Pernet. Although times were sometimes hard and deadlines tight, it was a pleasure to work with all of you. I would also like to express my gratitude to the assistants and planning staff of ATR, especially Sayuri Shirai, Sayaka Taniguchi, and Tomoko Honda, who have put extraordinary effort into supporting my work.

And of course I have to to thank my wonderful parents and my awesome sister. You are the best, and I couldn't have achieved any of this without your inspiration, love, and support.

The work presented in Chapters 2 and 3 was supported by the Special Coordination Funds for Promoting Science and Technology of the Ministry of Education, Culture, Sports, Science and Technology, Japan, and the work in Chapter 3 was supported in part by the Grant-in Aid for Scientific Research (A), Kakenhi, under Grant 21680022. The development of the interaction design framework presented in Chapter 6 was funded by NEDO, and the work presented in Chapters 4, 5, and 7 was supported by the Ministry of Internal Affairs and Communications of Japan.

Works Cited

[1] R. Aipperspach, T. Rattenbury, A. Woodruff, and J. Canny. A quantitative method for revealing and comparing places in the home. *UbiComp 2006: Ubiquitous Computing*, pages 1–18, 2006.

[2] A. Almeida, J. Almeida, and R. Araujo. Real-time tracking of moving objects using particle filters. In *Proc. of the IEEE International Symposium on Industrial Electronics (ISIE 2005)*, pages 1327–1332, Dubrovnik, Croatia, June 2005.

[3] M. Alvarez, R. Galan, F. Matia, D. Rodriguez-Losada, and A. Jimenez. An emotional model for a guide robot. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 40(5):982–992, 2010.

[4] G. Antonini, M. Bierlaire, and M. Weber. Discrete choice models of pedestrian walking behavior. *Transportation Research Part B: Methodological*, 40(8):667–687, 2006.

[5] H. Asoh, T. Matsui, J. Fry, F. Asano, and S. Hayamizu. A spoken dialog system for a mobile office robot. In *Eurospeech*, volume 99, pages 1139–1142, 1999.

[6] P. Bahl, V. Padmanabhan, and A. Balachandran. Enhancements to the radar user location and tracking system. Technical report, technical report, Microsoft Research, 2000.

[7] G. Ball and D. Hall. Isodata, a novel method of data analysis and pattern classification. Technical report, DTIC Document, 1965.

[8] M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun. Learning motion patterns of people for compliant robot motion. *The International Journal of Robotics Research*, 24(1):31–48, 2005.

[9] C. Breazeal, C. D. Kidd, A. L. Thomaz, G. Hoffman, and M. Berlin. Effects of nonverbal communication on efficiency and robustness in human-robot teamwork. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 708–713, 2005.

[10] C. Breazeal and B. Scassellati. How to build robots that make friends and influence people. In *Intelligent Robots and Systems, 1999. IROS'99. Proceedings. 1999 IEEE/RSJ International Conference on*, volume 2, pages 858–863. IEEE, 1999.

[11] A. Brooks and S. Williams. Tracking people with networks of heterogeneous sensors. In *Australasian Conference on Robotics and Automation (ACRA 2003)*, pages 1–7, Brisbane QLD, Australia, Dec.1–3 2003.

[12] R. Brooks. A robust layered control system for a mobile robot. *Robotics and Automation, IEEE Journal of*, 2(1):14–23, 1986.

[13] A. Bruce and G. Gordon. Better motion prediction for people-tracking. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA 2004)*, New Orleans, LA, USA, 2004.

[14] B. Brunner, G. Hirzinger, K. Landzettel, and J. Heindl. Multisensory shared autonomy and tele-sensor-programming - key issues in the space robot technology experiment rotex. In *Intelligent Robots and Systems '93, IROS '93. Proceedings of the 1993 IEEE/RSJ International Conference on*, volume 3, pages 2123–2139, 1993.

[15] W. Burgard, A. B. Cremers, D. Fox, D. Hahnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. The interactive museum tour-guide robot. In *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 11–18, Madison, Wisconsin, United States, 1998. American Association for Artificial Intelligence.

[16] X. Chai and Q. Yang. Multiple-goal recognition from low-level signals. In *Proceedings of the National Conference on Artificial Intelligence*, volume 20, page 3. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[17] C. C. Chang and C. J. Lin. Libsvm: A library for support vector machines, 2001.

[18] H.-H. Chiang, S.-J. Wu, J.-W. Perng, B.-F. Wu, and T.-T. Lee. The human-in-the-loop design approach to the longitudinal automation system for an intelligent vehicle. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 40(4):708–720, 2010.

[19] A. Clodic, R. Alami, V. Montreuil, S. Li, B. Wrede, and A. Swadzba. A study of interaction between dialog and decision for human-robot collaborative task achievement. In *Robot and Human interactive Communication, 2007. RO-MAN 2007. The 16th IEEE International Symposium on*, pages 913–918, 2007.

[20] O. Corporation. Okao vision, 2012.

[21] J. W. Crandall and M. L. Cummings. Developing performance metrics for the supervisory control of multiple robots. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 33–40, Arlington, Virginia, USA, 2007. ACM.

[22] J. W. Crandall and M. A. Goodrich. Characterizing efficiency of human robot interaction: a case study of shared-control teleoperation. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 2, pages 1290–1295 vol.2, 2002.

[23] J. Cui, H. Zhao, and R. Shibasaki. Fusion of detection and matching based approaches for laser based multiple people tracking. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006)*, pages 642–649, New York, USA, 2006.

[24] M. L. Cummings and P. J. Mitchell. Predicting controller capacity in supervisory control of multiple uavs. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 38(2):451–460, 2008.

[25] N. Dahlback, A. Jonsson, and L. Ahrenberg. Wizard of oz studies: why and how. In *Proceedings of the 1st international conference on Intelligent user interfaces*, pages 193–200, Orlando, Florida, United States, 1993. ACM.

[26] J. Dan R. Olsen and S. B. Wood. Fan-out: measuring human control of multiple robots. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 231–238, Vienna, Austria, 2004. ACM.

[27] K. Dautenhahn, M. Walters, S. Woods, K. L. Koay, C. L. Nehaniv, A. Sisbot, R. Alami, and T. Simeon. How may i serve you?: a robot companion approaching a seated person in a helping context. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 172–179. ACM, 2006.

[28] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA 1999)*, pages 1322–1328, Detroit, Michigan, USA, May 1999. ICRA.

[29] M. Denecke. Rapid prototyping for spoken dialogue systems. In *Proceedings of the 19th international conference on Computational linguistics - Volume 1*, pages 1–7, Taipei, Taiwan, 2002. Association for Computational Linguistics.

[30] F. Doshi and N. Roy. Efficient model learning for dialog management. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 65–72, Arlington, Virginia, USA, 2007. ACM.

[31] M. Dragone, T. Holz, B. Duffy, and G. O'Hare. Social situated agents in virtual, real and mixed reality environments. In *Intelligent Virtual Agents*, pages 166–177. Springer, 2005.

[32] J. L. Drury, L. Riek, and N. Rackliffe. A decomposition of uav-related situation awareness. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 88–94, Salt Lake City, Utah, USA, 2006. ACM.

[33] N. Eagle and A. Pentland. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, 2006.

[34] B. Erwin, M. Cyr, and C. Rogers. Lego engineer and robolab: Teaching engineering with labview from kindergarten to graduate school. *International Journal of Engineering Education*, 16(3):181–192, 2000.

[35] D. Feil-Seifer and M. Mataric. Distance-based computational models for facilitating robot interaction with children. *Journal of Human-Robot Interaction*, 1(1), 2012.

[36] F. Ferland, F. Pomerleau, C. T. L. Dinh, and F. Michaud. Egocentric and exocentric teleoperation interface using real-time, 3d video projection. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 37–44, La Jolla, California, USA, 2009. ACM.

[37] G. Ferri, A. Manzi, P. Salvini, B. Mazzolai, C. Laschi, and P. Dario. Dustcart, an autonomous robot for door-to-door garbage collection: From dustbot project to the experimentation in the small town of peccioli. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 655–660, 2011.

[38] A. Fod, A. Howard, and M. J. Mataric. Laser-based people tracking. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA 2002)*, pages 3024–3029, Washington, DC, USA, 2002.

[39] T. Fong and C. Thorpe. Vehicle teleoperation interfaces. *Autonomous Robots*, 11(1):9–18, 2001.

[40] C. Fortin and R. Breton. Temporal interval production and processing in working memory. *Attention, Perception, & Psychophysics*, 57(2):203–215, 1995.

[41] C. Fortin and R. Rousseau. Time estimation as an index of processing demand in memory search. *Attention, Perception, & Psychophysics*, 42(4):377–382, 1987.

[42] C. Fortin, R. Rousseau, P. Bourque, and E. Kirouac. Time estimation and concurrent nontemporal processing: Specific interference from short-term-memory demands. *Attention, Perception, & Psychophysics*, 53(5):536–548, 1993.

[43] D. Fox. KLD-Sampling: Adaptive Particle Filters. In *Advances in Neural Information Processing Systems (NIPS) 14*, pages 713–720. MIT Press, 2001.

[44] D. Fox, W. Burgard, and S. Thrun. The dynamic window approach to collision avoidance. *Robotics & Automation Magazine, IEEE*, 4(1):23–33, 1997.

[45] P. Fraisse. Perception and estimation of time. *Annual review of psychology*, 35(1):1–37, 1984.

[46] A. Garrell and A. Sanfeliu. Model validation: Robot behavior in people guidance mission using dtm model and estimation of human motion behavior. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5836–5841, 2010.

[47] B. Gerkey and M. Mataric. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954, 2004.

[48] D. F. Glas, T. Kanda, H. Ishiguro, and N. Hagita. Simultaneous teleoperation of multiple social robots. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 311–318, Amsterdam, The Netherlands, 2008. ACM.

[49] D. F. Glas, T. Kanda, H. Ishiguro, and N. Hagita. Field trial for simultaneous teleoperation of mobile social robots. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 149–156, La Jolla, California, USA, 2009. ACM.

[50] D. F. Glas, T. Kanda, H. Ishiguro, and N. Hagita. Teleoperation of multiple social robots. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 42(3):530–544, 2012.

[51] D. F. Glas, T. Miyashita, H. Ishiguro, and N. Hagita. Laser tracking of human body motion using adaptive shape modeling. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 602–608, 2007.

[52] D. F. Glas, T. Miyashita, H. Ishiguro, and N. Hagita. Laser-based tracking of human position and orientation using parametric shape modeling. *Advanced Robotics*, 23(4):405–428, 2009.

[53] D. F. Glas, T. Miyashita, H. Ishiguro, and N. Hagita. Automatic position calibration and sensor displacement detection for networks of laser range finders for human tracking. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2938–2945, 2010.

[54] R. Gockley, A. Bruce, J. Forlizzi, M. Michalowski, A. Mundell, S. Rosenthal, B. Sellner, R. Simmons, K. Snipes, and A. Schultz. Designing robots for long-term social interaction. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 1338–1343. IEEE, 2005.

[55] R. Gockley, J. Forlizzi, and R. Simmons. Interactions with a moody robot. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 186–193, Salt Lake City, Utah, USA, 2006. ACM.

[56] R. Gockley, J. Forlizzi, and R. Simmons. Natural person-following behavior for social robots. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 17–24. ACM, 2007.

[57] M. A. Goodrich, T. W. McLain, J. D. Anderson, J. Sun, and J. W. Crandall. Managing autonomy in robot teams: observations from four experiments. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 25–32, Arlington, Virginia, USA, 2007. ACM.

[58] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings-F*, 140(2):107–113, 1993.

[59] A. Green, H. Huttenrauch, and K. S. Eklundh. Applying the wizard-of-oz framework to cooperative service discovery and configuration. In *Robot and Human Interactive Communication, 2004. ROMAN 2004. 13th IEEE International Workshop on*, pages 575–580, 2004.

[60] H. Gross, H. Bohme, C. Schroter, S. Muller, A. Konig, C. Martin, M. Merten, and A. Bley. Shopbot: Progress in developing an interactive mobile shopping assistant for everyday use. In *Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on*, pages 3471–3478. IEEE, 2008.

[61] B. Hardin and M. A. Goodrich. On using mixed-initiative control: a perspective for managing large-scale robotic teams. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 165–172, La Jolla, California, USA, 2009. ACM.

[62] S. Hart and L. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. *Human mental workload*, 1:139–183, 1988.

[63] Y. Hato, S. Satake, T. Kanda, M. Imai, and N. Hagita. Pointing to space: modeling of deictic interaction referring to regions. In *Human-Robot Interaction (HRI), 2010 5th ACM/IEEE International Conference on*, pages 301–308. IEEE, 2010.

[64] A. D. Haumann, K. D. Listmann, and V. Willert. Discoverage: A new paradigm for multi-robot exploration. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 929–934, 2010.

[65] K. Hayashi, D. Sakamoto, T. Kanda, M. Shiomi, S. Koizumi, H. Ishiguro, T. Ogasawara, and N. Hagita. Humanoid robots as a passive-social medium - a field experiment at a train station. In *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, pages 137–144, 2007.

[66] K. Hayashi, M. Shiomi, T. Kanda, and N. Hagita. Friendly patrolling: A model of natural encounters. In *Robotics: Science and Systems VII*, Los Angeles, CA, 2012. MIT Press (MA).

[67] D. Helbing, F. Schweitzer, J. Keltsch, and P. Molnar. Active walker model for the formation of human and animal trail systems. *Physical Review E*, 56(3):2527, 1997.

[68] B. Hesse, C. Werner, and I. Altman. Temporal aspects of computer-mediated communication. *Computers in Human Behavior*, 4(2):147–165, 1988.

[69] R. Hicks, G. Miller, G. Gaes, and K. Bierman. Concurrent processing demands and the experience of time-in-passing. *The American Journal of Psychology*, pages 431–446, 1977.

[70] S. G. Hill and B. Bodt. A field experiment of autonomous mobility: operator workload for one and two robots. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 169–176, Arlington, Virginia, USA, 2007. ACM.

[71] B. Hillier and J. Hanson. *The Social Logic of Space*. Cambridge University Press, Cambridge, U.K., 1984.

[72] G. Hoffman and C. Breazeal. Effects of anticipatory action on human-robot teamwork: Efficiency, fluency, and perception of team. In *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, pages 1–8. IEEE, 2007.

[73] C.-W. Hsu and C.-J. Lin. A comparison of methods for multiclass support vector machines. *Neural Networks, IEEE Transactions on*, 13(2):415–425, 2002.

[74] C. Hua, H. Wu, Q. Chen, and T. Wada. A general framework for tracking people. In *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, pages 511–516. IEEE, 2006.

[75] C. M. Humphrey and J. A. Adams. Compass visualizations for human-robotic interaction. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 49–56, Amsterdam, The Netherlands, 2008. ACM.

[76] I. I. Hussein and D. M. Stipanovic. Effective coverage control for mobile sensor networks with guaranteed collision avoidance. *Control Systems Technology, IEEE Transactions on*, 15(4):642–657, 2007.

[77] M. Inaba, S. Kagami, F. Kanehiro, Y. Hoshino, and H. Inoue. A platform for robotics research based on the remote-brained robot approach. *The International Journal of Robotics Research*, 19(10):933–954, 2000.

[78] Y. Iwamura, M. Shiomi, T. Kanda, H. Ishiguro, and N. Hagita. Do elderly people prefer a conversational humanoid as a shopping assistant partner in supermarkets? In *Proceedings of the 6th international conference on Human-robot interaction*, pages 449–456, Lausanne, Switzerland, 2011. ACM.

[79] J. Jaffe and S. Feldstein. *Rhythms of dialogue*. Academic Press, New York, 1970.

[80] D. B. Kaber and M. R. Endsley. The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task. *Theoretical Issues in Ergonomics Science*, 5(2):113–153, 2004.

[81] T. Kanda, D. F. Glas, M. Shiomi, and N. Hagita. Abstracting people's trajectories for social robots to proactively approach customers. *Robotics, IEEE Transactions on*, 25(6):1382–1396, 2009.

[82] T. Kanda, D. F. Glas, M. Shiomi, H. Ishiguro, and N. Hagita. Who will be the customer?: a social robot that anticipates people's behavior from their trajectories. In *Proceedings of the 10th international conference on Ubiquitous computing*, pages 380–389, Seoul, Korea, 2008. ACM.

[83] T. Kanda, T. Hirano, D. Eaton, and H. Ishiguro. Interactive robots as social partners and peer tutors for children: A field trial. *Human-Computer Interaction*, 19(1):61–84, 2004.

[84] T. Kanda, H. Ishiguro, M. Imai, and T. Ono. Development and evaluation of interactive humanoid robots. *Proceedings of the IEEE*, 92(11):1839–1850, 2004.

[85] T. Kanda, M. Shiomi, Z. Miyashita, H. Ishiguro, and N. Hagita. An affective guide robot in a shopping mall. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 173–180, La Jolla, California, USA, 2009. ACM.

[86] T. Kanda, M. Shiomi, L. Perrin, T. Nomura, H. Ishiguro, and N. Hagita. Analysis of people trajectories with ubiquitous sensors in a science museum. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 4846–4853. IEEE, 2007.

[87] H. Kawai, T. Toda, J. Ni, M. Tsuzaki, and K. Tokuda. Ximera: A new tts from atr based on corpus-based technologies. In *Fifth ISCA Workshop on Speech Synthesis*, 2004.

[88] R. Kirby, J. Forlizzi, and R. Simmons. Affective social robots. *Robotics and Autonomous Systems*, 58(3):322–332, 2010.

[89] M. Kleinehagenbrock, J. Fritsch, and G. Sagerer. Supporting advanced interaction capabilities on a mobile robot with a flexible control system. In *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 4, pages 3649–3655 vol.4, 2004.

[90] K. Koile, K. Tollmar, D. Demirdjian, H. Shrobe, and T. Darrell. Activity zones for context-aware computing. In *UbiComp 2003: Ubiquitous Computing*, pages 90–106. Springer, 2003.

[91] J. Kramer and M. Scheutz. Development environments for autonomous mobile robots: A survey. *Autonomous Robots*, 22(2):101–132, 2007.

[92] J. Krumm and E. Horvitz. Predestination: Inferring destinations from partial trajectories. *UbiComp 2006: Ubiquitous Computing*, pages 243–260, 2006.

[93] M. K. Lee, J. Forlizzi, P. E. Rybski, F. Crabbe, W. Chung, J. Finkle, E. Glaser, and S. Kiesler. The snackbot: Documenting the design of a robot for long-term human-robot interaction. In *Human-Robot Interaction (HRI), 2009 4th ACM/IEEE International Conference on*, pages 7–14, 2009.

[94] O. Lemon, A. Bracy, A. Gruenstein, and S. Peters. The witas multi-modal dialogue system i. In *Proceedings of EuroSpeech*, volume 2001, 2001.

[95] J. Letchner, D. Fox, and A. LaMarca. Large-scale localization from wireless signal strength. In *Proceedings of the national conference on artificial intelligence*, volume 20, page 15. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2005.

[96] P. Lewis and R. Miall. Distinct systems for automatic and cognitively controlled time measurement: evidence from neuroimaging. *Current opinion in neurobiology*, 13(2):250–255, 2003.

[97] S. Li, B. Wrede, and G. Sagerer. A dialog system for comparative user studies on robot verbal behavior. In *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pages 129–134, 2006.

[98] L. Liao, D. Fox, and H. Kautz. Location-based activity recognition using relational markov networks. In *Proceedings of the 19th international joint conference on Artificial intelligence*, pages 773–778, Edinburgh, Scotland, 2005. Morgan Kaufmann Publishers Inc.

[99] L. Liao, D. Patterson, D. Fox, and H. Kautz. Learning and inferring transportation routines. *Artificial Intelligence*, 171(5):311–331, 2007.

[100] A. Madhavapeddy and A. Tse. A study of bluetooth propagation using accurate indoor location mapping. *UbiComp 2005: Ubiquitous Computing*, pages 903–903, 2005.

[101] D. Matsui, T. Minato, K. MacDorman, and H. Ishiguro. Generating natural motion in an android by mapping human motion. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3301–3308. IEEE, 2005.

[102] T. Matsui, H. Asoh, J. Fry, Y. Motomura, F. Asano, T. Kurita, I. Hara, and N. Otsu. Integrated natural spoken dialogue system of jijo-2 mobile robot for office services. In *Proceedings of the National Conference on Artificial Intelligence*, pages 621–627. John Wiley & Sons, Ltd., 1999.

[103] Y. Matsumoto, T. Wada, S. Nishio, T. Miyashita, and N. Hagita. Scalable and robust multi-people head tracking by combining distributed multiple sensors. *Intelligent Service Robotics*, 3(1):29–36, 2010.

[104] N. Mavridis, E. Machado, N. Giakoumidis, N. Batalas, I. Shebli, E. Ameri, F. Neyadi, and A. Neyadi. Real-time teleoperation of an industrial robotic arm through human arm movement imitation. In *Proceedings of the International Symposium on Robotics and Intelligent Sensors (IRIS)*, 2010.

[105] N. Mavridis, A. Tsamakos, N. Giakoumidis, H. Baloushi, S. Ashkari, M. Shamsi, and A. Kaabi. Steps towards affordable android telepresence. In *Proceedings of the ACM/IEEE conference of human-robot interaction (HRI) social robotic telepresence ws*, 2011.

[106] B. Mazzolai, V. Mattoli, C. Laschi, P. Salvini, G. Ferri, G. Ciaravella, and P. Dario. Networked and cooperating robots for urban hygiene: The eu funded dustbot project. In *Proceedings of the 5th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, 2008.

[107] M. L. McLaughlin. *Conversation: How talk is organized*, volume 3 of *SAGE Series in Interpersonal Communication*. Sage Publications, 1984.

[108] M. McTear. Modelling spoken dialogues with state transition diagrams: experiences with the cslu toolkit. *development*, 5:7, 1998.

[109] M. F. McTear. Spoken dialogue technology: enabling the conversational user interface. *ACM Comput. Surv.*, 34(1):90–169, 2002.

[110] M. Michalowski, S. Sabanovic, and R. Simmons. A spatial model of engagement for a social robot. In *Advanced Motion Control, 2006. 9th IEEE International Workshop on*, pages 762–767. IEEE, 2006.

[111] S. Monteiro and E. Bicho. Attractor dynamics approach to formation control: theory and application. *Autonomous Robots*, 29(3):331–355, 2010.

[112] M. Montemerlo, S. Thrun, and W. Whittaker. Conditional particle filters for simultaneous mobile robot localization and people-tracking. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA 2002)*, pages 695–701, Washington, DC, USA, 2002. ICRA.

[113] J. R. Movellan, F. Tanaka, I. R. Fasel, C. Taylor, P. Ruvolo, and M. Eckhardt. The rubi project: a progress report. In *Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 333–339, Arlington, Virginia, USA, 2007. ACM.

[114] J. Mumm and B. Mutlu. Human-robot proxemics: Physical and psychological distancing in human-robot interaction. In *Human-Robot Interaction (HRI), 2011 6th ACM/IEEE International Conference on*, pages 331–338, 2011.

[115] B. Mutlu and J. Forlizzi. Robots in organizations: the role of workflow, social, and environmental factors in human-robot interaction. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 287–294, Amsterdam, The Netherlands, 2008. ACM.

[116] B. Mutlu, J. Forlizzi, and J. Hodgins. A storytelling robot: Modeling and evaluation of human-like gaze behavior. In *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, pages 518–523, 2006.

[117] B. Mutlu, T. Shiwa, T. Kanda, H. Ishiguro, and N. Hagita. Footing in human-robot conversations: how robots might shape participant roles using gaze cues. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 61–68, La Jolla, California, USA, 2009. ACM.

[118] M. Nakano, Y. Hasegawa, K. Nakadai, T. Nakamura, J. Takeuchi, T. Torii, H. Tsujino, N. Kanda, and H. G. Okuno. A two-layer model for behavior and dialogue planning in conversational service robots. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3329–3335, 2005.

[119] Y. I. Nakano, G. Reinstein, T. Stocky, and J. Cassell. Towards a model of face-to-face grounding. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, pages 553–561, Sapporo, Japan, 2003. Association for Computational Linguistics.

[120] C. Nielsen, M. Goodrich, and R. Ricks. Ecological interfaces for improving mobile robot teleoperation. *Robotics, IEEE Transactions on*, 23(5):927–941, 2007.

[121] C. W. Nielsen and M. A. Goodrich. Comparing the usefulness of video and map information in navigation tasks. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 95–101, Salt Lake City, Utah, USA, 2006. ACM.

[122] S. Nishio, N. Hagita, T. Miyashita, T. Kanda, N. Mitsunaga, M. Shiomi, and T. Yamazaki. Robotic platforms structuring information on people and environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2008.

[123] P. Nurmi and J. Koolwaaij. Identifying meaningful locations. In *Mobile and Ubiquitous Systems: Networking & Services, 2006 Third Annual International Conference on*, pages 1–8, 2006.

[124] Y. Okuno, T. Kanda, M. Imai, H. Ishiguro, and N. Hagita. Providing route directions: design of robot's utterance, gesture, and timing. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 53–60, La Jolla, California, USA, 2009. ACM.

[125] D. Olsen and M. Goodrich. Metrics for evaluating human-robot interactions. In *Proceedings of PERMIS*, volume 2003, 2003.

[126] L. E. Parker. Alliance: an architecture for fault tolerant multirobot cooperation. *Robotics and Automation, IEEE Transactions on*, 14(2):220–240, 1998.

[127] S. Patel, K. Truong, and G. Abowd. Powerline positioning: A practical sub-room-level indoor location system for domestic use. *UbiComp 2006: Ubiquitous Computing*, pages 441–458, 2006.

[128] J. Peltason and B. Wrede. Modeling human-robot interaction based on generic interaction patterns. *AAAI Report on Dialog with Robots*, 2010.

[129] K. Perlin. Real time responsive animation with personality. *Visualization and Computer Graphics, IEEE Transactions on*, 1(1):5–15, 1995.

[130] E. Pot, J. Monceaux, R. Gelin, and B. Maisonnier. Choregraphe: a graphical tool for humanoid robot programming. In *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*, pages 46–51, 2009.

[131] R. M. Ratwani, J. M. McCurry, and J. G. Trafton. Single operator, multiple robots: an eye movement based theoretic model of operator situation awareness. In *Proceedings of the 5th ACM/IEEE international conference on Human-robot interaction*, pages 235–242, Osaka, Japan, 2010. IEEE Press.

[132] B. Robins, K. Dautenhahn, R. te Boekhorst, and C. L. Nehaniv. Behaviour delay and robot expressiveness in child-robot interactions: A user study on interaction kinesics. In *Human-Robot Interaction (HRI), 2008 3rd ACM/IEEE International Conference on*, pages 17–24, 2008.

[133] N. Roy, J. Pineau, and S. Thrun. Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 93–100, Hong Kong, 2000. Association for Computational Linguistics.

[134] H. Sacks, E. Schegloff, and G. Jefferson. A simplest systematics for the organization of turn-taking for conversation. *Language*, pages 696–735, 1974.

[135] A. Saffiotti, M. Broxvall, M. Gritti, K. LeBlanc, R. Lundh, J. Rashid, B. S. Seo, and Y. J. Cho. The peis-ecology project: Vision and results. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2329–2335, 2008.

[136] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 26(1):43–49, 1978.

[137] A. Sanfeliu, J. Andrade-Cetto, M. Barbosa, R. Bowden, J. CapitÃ¡n, A. Corominas, A. Gilbert, J. Illingworth, L. Merino, and J. Mirats. Decentralized sensor fusion for ubiquitous networking robotics in urban areas. *Sensors*, 10(3):2274–2314, 2010.

[138] A. Sanfeliu, N. Hagita, and A. Saffiotti. Special issue: Network robot systems. *Robotics and Autonomous Systems*, 56(10):791–791, 2008.

[139] S. Satake, T. Kanda, D. F. Glas, M. Imai, H. Ishiguro, and N. Hagita. How to approach humans?: Strategies for social robots to initiate interaction. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 109–116, La Jolla, California, USA, 2009. ACM.

[140] B. Scassellati. Investigating models of social development using a humanoid robot. In *Neural Networks, 2003. Proceedings of the International Joint Conference on*, volume 4, pages 2704–2709 vol.4, 2003.

[141] P. Scerri, D. V. Pynadath, and M. Tambe. Towards adjustable autonomy for the real world. *J. Artif. Int. Res.*, 17(1):171–228, 2002.

[142] M. Scheutz, P. Schermerhorn, and J. Kramer. The utility of affect expression in natural language interactions in joint human-robot tasks. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 226–233, Salt Lake City, Utah, USA, 2006. ACM.

[143] D. Schulz, W. Burgard, D. Fox, and A. B. Cremens. People tracking with mobile robots using sample-based joint probabilistic data association filters. *International Journal of Robotics Research (IJRR)*, 22(2):99–116, 2003.

[144] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers. People tracking with mobile robots using sample-based joint probabilistic data association filters. *The International Journal of Robotics Research*, 22(2):99–116, 2003.

[145] B. Sellner, F. W. Heger, L. M. Hiatt, R. Simmons, and S. Singh. Coordinated multiagent teams and sliding autonomy for large-scale assembly. *Proceedings of the IEEE*, 94(7):1425–1444, 2006.

[146] B. P. Sellner, L. M. Hiatt, R. Simmons, and S. Singh. Attaining situational awareness for sliding autonomy. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 80–87, Salt Lake City, Utah, USA, 2006. ACM.

[147] X. Shao, H. Zhao, K. Nakamura, K. Katabira, R. Shibasaki, and Y. Nakagawa. Detection and tracking of multiple pedestrians by using laser range scanners. In *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, pages 2174–2179, 2007.

[148] X. Shao, H. Zhao, K. Nakamura, R. Shibasaki, R. Zhang, and Z. Liu. Analyzing pedestrians' walking patterns using single-row laser range scanners. In *Systems, Man and Cybernetics, 2006. SMC'06. IEEE International Conference on*, volume 2, pages 1202–1207. IEEE, 2006.

[149] C. Shi, T. Kanda, M. Shimada, F. Yamaoka, H. Ishiguro, and N. Hagita. Easy development of communicative behaviors in social robots. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 5302–5309, 2010.

[150] T. Shimizu, H. Yamamoto, H. Masataki, S. Matsunaga, and Y. Sagisaka. Spontaneous dialogue speech recognition using cross-word context constrained word graphs. In *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, volume 1, pages 145–148 vol. 1, 1996.

[151] M. Shiomi, T. Kanda, D. F. Glas, S. Satake, H. Ishiguro, and N. Hagita. Field trial of networked social robots in a shopping mall. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 2846–2853, St. Louis, MO, USA, 2009. IEEE Press.

[152] M. Shiomi, T. Kanda, H. Ishiguro, and N. Hagita. Interactive humanoid robots for a science museum. *Intelligent Systems, IEEE*, 22(2):25–32, 2007.

[153] M. Shiomi, D. Sakamoto, T. Kanda, C. T. Ishi, H. Ishiguro, and N. Hagita. A semi-autonomous communication robot: a field trial at a train station. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 303–310, Amsterdam, The Netherlands, 2008. ACM.

[154] T. Shiwa, T. Kanda, M. Imai, H. Ishiguro, and N. Hagita. How quickly should communication robots respond? In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 153–160, Amsterdam, The Netherlands, 2008. ACM.

[155] N. Sian, T. Sakaguchi, K. Yokoi, Y. Kawai, and K. Maruyama. Operating humanoid robots in human environments. In *Proc. RSS Workshop: Manipulation for Human Environments*, Philadelphia, PA, 2006.

[156] N. E. Sian, K. Yokoi, S. Kajita, and K. Tanie. Whole body teleoperation of a humanoid robot integrating operator's intention and robot's autonomy: an experimental verification. In *Intelligent Robots and Systems, 2003. (IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1651–1656 vol.2, 2003.

[157] C. Sidner, C. Kidd, C. Lee, and N. Lesh. Where to look: a study of human-robot engagement. In *Proceedings of the 9th international conference on Intelligent user interfaces*, pages 78–84. ACM, 2004.

[158] C. L. Sidner, C. Lee, L.-P. Morency, and C. Forlines. The effect of head-nod recognition in human-robot conversation. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 290–296, Salt Lake City, Utah, USA, 2006. ACM.

[159] R. Siegwart, K. O. Arras, S. Bouabdallah, D. Burnier, G. Froidevaux, X. Greppin, B. Jensen, A. Lorotte, L. Mayor, M. Meisser, R. Philippsen, R. Piguet, G. Ramel, G. Terrien, and N. Tomatis. Robox at expo.02: A large-scale installation of personal robots. *Robotics and Autonomous Systems*, 42(3):203–222, 2003.

[160] E. A. Sisbot, R. Alami, T. Simeon, K. Dautenhahn, M. Walters, and S. Woods. Navigation in the presence of humans. In *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*, pages 181–188, 2005.

[161] F. Sparacino. The museum wearable: Real-time sensor-driven understanding of visitors' interests for personalized visually-augmented museum experiences. In *Proceedings of Museums and the Web*, Boston, MA, 2002.

[162] K. Stanney, S. Samman, L. Reeves, K. Hale, W. Buff, C. Bowers, B. Goldiez, D. Nicholson, and S. Lackey. A paradigm shift in interactive computing: Deriving multimodal design principles from behavioral and neurological foundations. *International Journal of Human-Computer Interaction*, 17(2):229–257, 2004.

[163] A. Steinfeld, T. Fong, D. Kaber, M. Lewis, J. Scholtz, A. Schultz, and M. Goodrich. Common metrics for human-robot interaction. In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 33–40, Salt Lake City, Utah, USA, 2006. ACM.

[164] A. Steinfeld, O. C. Jenkins, and B. Scassellati. The oz of wizard: simulating the human for interaction research. In *Proceedings of the 4th ACM/IEEE international conference on Human robot interaction*, pages 101–108, La Jolla, California, USA, 2009. ACM.

[165] A. Subramanya, A. Raj, J. Bilmes, and D. Fox. Recognizing activities and spatial context using wearable sensors. In *Conf. Uncertainty Artif. Intell.*, Cambridge, MA, 2006.

[166] O. Sugiyama, T. Kanda, M. Imai, H. Ishiguro, N. Hagita, and Y. Anzai. Humanlike conversation with gestures and verbal cues based on a three-layer attention-drawing model. *Connection Science*, 18(4):379–402, 2006.

[167] N. Suzuki, K. Hirasawa, K. Tanaka, Y. Kobayashi, Y. Sato, and Y. Fujino. Learning motion patterns and anomaly detection by human trajectory analysis. In *Systems, Man and Cybernetics, 2007. ISIC. IEEE International Conference on*, pages 498–503, 2007.

[168] N. Taatgen, H. Van Rijn, and J. Anderson. An integrated theory of prospective time interval estimation: The role of cognition, attention, and learning. *Psychological Review*, 114(3):577, 2007.

[169] L. Takayama, E. Marder-Eppstein, H. Harris, and J. M. Beer. Assisted driving of a mobile remote presence system: System design and controlled user evaluation. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1883–1889, 2011.

[170] K. Thorisson. Natural turn-taking needs no manual: Computational theory and model, from perception to action. *Multimodality in language and speech systems*, 19, 2002.

[171] S. Thrun. Particle filters in robotics. In *Uncertainty in Artificial Intelligence (UAI)*, pages 511–518, 2002.

[172] S. Thrun, M. Bennewitz, W. Burgard, A. B. Cremers, F. Dellaert, D. Fox, D. Hahnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Minerva: a second-generation museum tour-guide robot. In *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, volume 3, pages 1999–2005 vol.3, 1999.

[173] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[174] T. Tomizawa, A. Ohya, and S. Yuta. Remote shopping robot system - development of a hand mechanism for grasping fresh foods in a supermarket. In *Intelligent Robots and Systems, 2006 IEEE/RSJ International Conference on*, pages 4953–4958, 2006.

[175] J. G. Trafton, M. D. Bugajska, B. R. Fransen, and R. M. Ratwani. Integrating vision and audition within a cognitive architecture to track conversations. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 201–208, Amsterdam, The Netherlands, 2008. ACM.

[176] P. Trautman and A. Krause. Unfreezing the robot: Navigation in dense, interacting crowds. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 797–803, 2010.

[177] A. Turner and A. Penn. Encoding natural movement as an agent-based system: an investigation into human pedestrian behaviour in the built environment. *Environ Plann B*, 29(4):473–490, 2002.

[178] A. J. N. van Breemen. Scripting technology and dynamic script generation for personal robot platforms. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3487–3492, 2005.

[179] D. Vanhooydonck, E. Demeester, A. H #252, ntemann, J. Philips, G. Vanacker, H. V. Brussel, and M. Nuttin. Adaptable navigational assistance for intelligent wheelchairs by means of an implicit personalized user model. *Robot. Auton. Syst.*, 58(8):963–977, 2010.

[180] L. Vig and J. A. Adams. Multi-robot coalition formation. *Robotics, IEEE Transactions on*, 22(4):637–649, 2006.

[181] A. M. Villagrasa. People tracking for a personal robot. Master's thesis, Royal Institute of Technology, Stockholm, Sweden, Aug. 2005.

[182] M. Walters, D. Syrdal, K. Koay, K. Dautenhahn, and R. Te Boekhorst. Human approach distances to a mechanical-looking robot with different robot voice styles. In *Robot and Human Interactive Communication, 2008. RO-MAN 2008. The 17th IEEE International Symposium on*, pages 707–712. IEEE, 2008.

[183] M. L. Walters, K. Dautenhahn, R. te Boekhorst, K. Kheng Lee, C. Kaouri, S. Woods, C. Nehaniv, D. Lee, and I. Werry. The influence of subjects' personality traits on personal spatial zones in a human-robot interaction experiment. In *Robot and Human Interactive Communication, 2005. ROMAN 2005. IEEE International Workshop on*, pages 347–352, 2005.

[184] A. Weiss, R. Bernhaupt, M. Tscheligi, D. Wollherr, K. Kuhnlenz, and M. Buss. A methodological variation for acceptance evaluation of human-robot interaction in public places. In *Robot and Human Interactive Communication, 2008. RO-MAN 2008. The 17th IEEE International Symposium on*, pages 713–718, 2008.

[185] S. N. Woods, M. L. Walters, K. L. Koay, and K. Dautenhahn. Comparing human robot interaction scenarios using live and video based methods: towards a novel methodological approach. In *Advanced Motion Control, 2006. 9th IEEE International Workshop on*, pages 750–755, 2006.

[186] M. Yamamoto and T. Watanabe. Timing control effects of utterance to communicative actions on embodied interaction with a robot. In *Robot and Human Interactive Communication, 2004. ROMAN 2004. 13th IEEE International Workshop on*, pages 467–472, 2004.

[187] F. Yamaoka, T. Kanda, H. Ishiguro, and N. Hagita. How contingent should a communication robot be? In *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 313–320, Salt Lake City, Utah, USA, 2006. ACM.

[188] F. Yamaoka, T. Kanda, H. Ishiguro, and N. Hagita. How close?: model of proximity control for information-presenting robots. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pages 137–144, Amsterdam, The Netherlands, 2008. ACM.

[189] Y. Yoshikawa, K. Shinozawa, H. Ishiguro, N. Hagita, and T. Miyamoto. Responsive robot gaze to interaction partner. In *Proceedings of robotics: Science and systems*, 2006.

[190] H. Zhao and R. Shibasaki. A novel system for tracking pedestrians using multiple singlerow laser-range scanners. *IEEE Transactions on Systems, Man, and Cybernetics*, 35(2):283–291, 2005.

[191] K. Zheng, D. F. Glas, T. Kanda, H. Ishiguro, and N. Hagita. How many social robots can one operator control? In *Proceedings of the 6th international conference on Human-robot interaction*, pages 379–386, Lausanne, Switzerland, 2011. ACM.