Probability-based Context-aware Robotic System

by

Kun Wang, M.Sc.

A thesis submitted to
The Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
in
Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario
April 2017

Abstract

The development of ambient intelligence in the past decades calls for the smart environment that is able to respond to entities inside. The technology of context-awareness is a promising solution to this mobile distributed computing, not only in small personal electronic devices, but also in robotic systems.

A prototype of context-aware assistive robotic system is designed and implemented in this thesis on the purpose of providing assistive services to the seniors and people with disabilities. The issue of context reasoning is extensively reviewed and researched. The particle filtering algorithm is improved in terms of proactive particle sampling, fuzzy based velocity estimation, and active observation selection, for the basic tasks of robotic tracking and following. The Monte Carlo partially observable Markov decision process is enhanced in terms of belief state augmentation, value function computation based on direct sampling, and non-greedy action execution, for the task of path planning. Experimental results are carried out to validate the feasibility and effectiveness of the proposed solutions. The prototyped robotic system can be further implemented on other potential applications for the aim of advanced service provision.

Table of Contents

List of Tables

List of Figures

List of Symbols

Symbols	Definition
$\boldsymbol{b}$	histogram filter bin
$\boldsymbol{c_t}$	index of the possible transition models
$\boldsymbol{d_K}$	distance between query and reference belief
$\boldsymbol{f_t}(\cdot)$	system transition model
$\boldsymbol{h_t}(\cdot)$	system observation model
$\boldsymbol{H(X)}$	entropy of random variable x
$\boldsymbol{H(X_t \mid z_t)}$	entropy of random variable x after observation
$\boldsymbol{H(\chi)}$	entropy of the particle set χ
$\boldsymbol{H_b(x)}$	entropy of belief
$\boldsymbol{k_o}$	output gain

$\boldsymbol{L}$	dimension of the action space
$\boldsymbol{N(\mu, \sigma)}$	Gaussian distribution
$\boldsymbol{\overline{o}}$	median of the membership output
$\boldsymbol{P(H\|X)}$	posterior probability density function (PDF)
$\boldsymbol{p(b)}$	probability of b
$\boldsymbol{p(b'\|u, b, z)}$	distribution calculated from b, u, and z
$\boldsymbol{r(x, u)}$	rewards for action u at state x
$\boldsymbol{\hat{\dot{r}}_t}$	estimated horizontal velocity
$\boldsymbol{r_t}$, $\boldsymbol{\dot{r}_t}$, $\boldsymbol{\ddot{r}_t}$	horizontal position, velocity and acceleration
$\boldsymbol{s_t}$, $\boldsymbol{\dot{s}_t}$, $\boldsymbol{\ddot{s}_t}$	vertical position, velocity and acceleration
$\boldsymbol{u^*}$	optimal action
$\boldsymbol{u_t}$	system control
$\boldsymbol{V_T(x)}$	value of state x at time index T
$\boldsymbol{w_n}$, $\boldsymbol{w_u}$, $\boldsymbol{w_f}$, $\boldsymbol{w_r}$	model weights
$\boldsymbol{w_i}$	weight of the particles
$\boldsymbol{x_t}$	system state

$\boldsymbol{x_t^i}$	random set of particles
$\boldsymbol{z_t}$	System observation
$\boldsymbol{\gamma}$	discount factor
$\boldsymbol{\delta(\cdot)}$	Dirac function
$\boldsymbol{\theta}$	action set
$\boldsymbol{\lambda_u}$	parameter of the exponential distribution
$\boldsymbol{\pi_{ij}}$	probability of model transition
$\boldsymbol{\chi_k}$	reference belief
$\boldsymbol{\chi_{query}}$	query belief
$\boldsymbol{\Phi_t}$	adaptive vector
$\boldsymbol{\phi_i^t}$	model switching potential
$\boldsymbol{\Omega_t^2}$	turning rate

Chapter 1

Introduction

1.1 Background

With the development of mobile internet technology, the ambient intelligence in computing is becoming a hot research topic. There is a strong requirement that the environment with ambient intelligence be sensitive and responsive to the presence of entities inside, such as people, devices, sensors, central servers, etc. In this collaborating environment, all the devices are working together to support the users, to carry out necessary tasks for the users, and to provide required services in an automatic manner, using information and intelligence hidden and processed in the network that connects all the entities in the environment [1]. This is also conforming to the current trend of internet of things.

1.2 Concept of Context and Context-awareness

Through the past few decades, the technology of context-awareness has offered considerably new opportunities for the end users and the developers in software applications. First introduced in [2], it has been widely utilized in the mobile distribution computing systems, e.g. notebook, PDA, smart phone.

It is essential while difficult to organize a general concept of context. In some previous work, context was just referred to by enumerations of examples. For instance, in some research, context was defined as location, identity, time, users' intention and emotion, etc. In other research, context was simply provided as synonyms: environment or situation which contained three main parts: computing environment, user environment and physical environment. To the best knowledge, the current most accurate and general definition is from Dey *et al.* in [3]:

> *Context is any information that can be used to characterize the situation of entity where entity is a person, place or object considered relevant to the interaction between users and applications including users and applications themselves.*

The basic idea of this definition is that, as long as a piece of information can be used to characterize a situation, that information is context. For instance, in the area of medical care, the contexts can be users' body signals such as electrocardiogram (ECG), electroencephalogram (EEG), and electromyography (EMG) that is detected by wearable sensors, users' status and activities detected by camera or sonar sensors mounted on moving robot, or users' intention estimated using probability theory.

In general, there are four primary types of context: location, identity, time and activity. These four contexts can be categorized as the basic contexts that can be further used and processed to obtain other contexts in higher level. For example, if a user is detected to be in the bedroom at midnight with the light off, it can be inferred that the user's activity is sleeping.

As for context-aware computing, previous definitions fall into two categories: using contexts and adapting to contexts. The first category focuses on detecting, sensing, interpreting and then displaying contexts. The second focuses on how the applications can dynamically change their behaviors to adapt to the sensed contexts. Again, Dey *et al.* [3] gives the definition of context-awareness:

> *A system is context-aware if it uses context to provide the relevant information and/or services to the user, where relevancy depends on the user's task.*

This definition is more general, including both "using contexts" and "adapting to contexts", and is focusing more on the users, not just the applications.

Overall, context-awareness means that the systems can use the necessary contexts from the surrounding environment to execute required tasks and to adjust their operations accordingly. The principle here is that, applications should be aware of the contexts and should adapt to the changes of contexts to provide "anytime, anywhere, anyone" computing service for the users, with all these processes vanishing into the application background to make the users and their tasks the central focus, without users' explicit interventions [4].

1.3 Context-aware System Design

1.3.1 General Context-aware Systems

For context-aware system design, one basic rule is the separation of detecting from using contexts. Under this rule, several frameworks have been proposed in previous research with different functional ranges, locations, naming of layers, etc. Framework technique has the advantage of simplifying the design of context-aware system since it can hide the low-level details of context-aware system and allow the developers to focus on the goals of the system. The properly designed framework can be also conveniently modified and extended in accordance with specific applications. To the best knowledge, the most common approach for context-aware system design in the previous research is the classical layered and centralized structure [4], presented in Fig. 1.1.

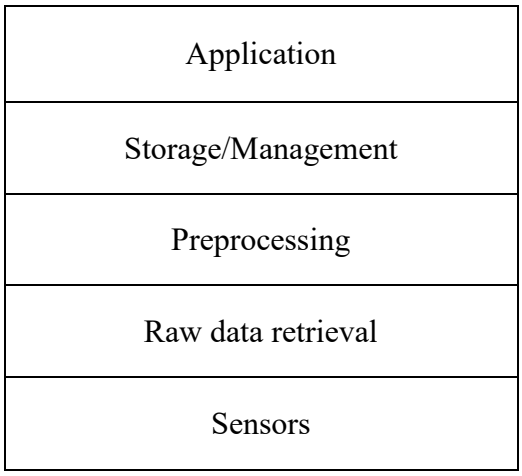


Figure 1.1: The layered structure of context-aware system

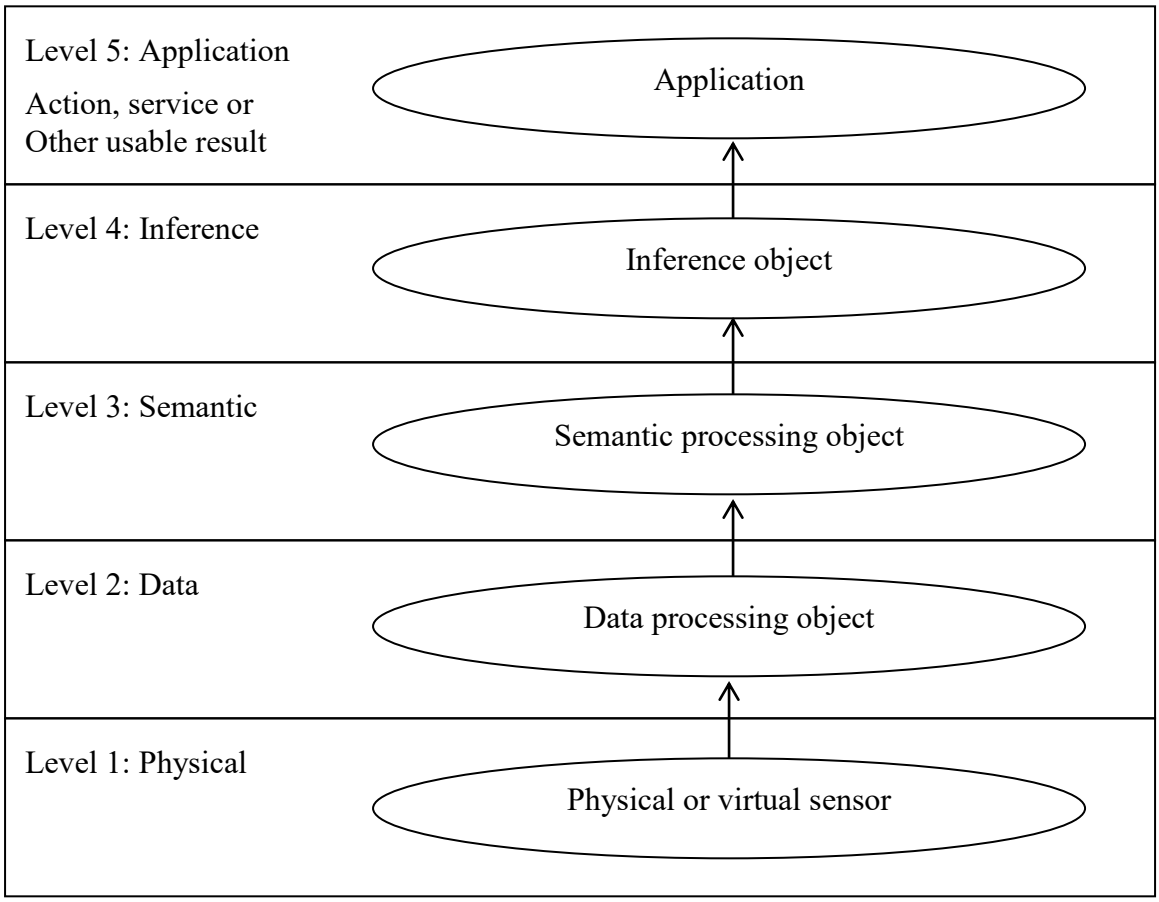


Figure 1.2: The five-layer context-aware system architecture

In order to realize the separation of detecting from using context to improve the system extensibility and reusability, the most commonly used method is the design of layered and centralized architecture. Ailisto *et al.* [5] proposed a five-layer context-aware system architecture, by adding the functions of context reasoning into the current context detecting and using. This layered architecture is shown in Fig. 1.2, in which each layer is illustrated with one typical object. Specifically, in the physical layer, physical or virtual sensor objects were responsible for detecting raw data from the surrounding environment.

These raw data were pre-processed in the data layer using signal processing techniques. Then the pre-processed data were modeled and converted into meaningful contexts in the semantic layer. In the following inference layer, the modeled contexts were reasoned for the decisions and policies making, using the pre-defined inference rules. In the end, these decisions and policies were fed into the application layer to be executed as services.

There has been extensive research in the field of context-aware system design. Korpipää *et al.* [6] developed the architecture of the Context Managing Framework, a classical hierarchical infrastructure with one centralized component, using layered structure. It consisted of four parts: context manager, resource server, context recognition service and application. In general, the context manager functioned as a central server, while other entities as clients. Two main features in this framework were the utilization of blackboard-based mechanism in the context manager and the context ontology in resource server and recognition service. The blackboard mechanism, used for communication between entities, was a data-centric model, which processed the posted messages to a shared media (blackboard) and subscribed to it to be notified when some specified event occurred. It was therefore simple to add new context resource and easy to configure.

Similarly, Service-Oriented Context-Aware Middleware (SOCAM) in [7] adopted a central server as well, called context interpreter, which collected context data from context providers in a distributed manner and processed it to applications, using the ontology mark-up language (OWL). The SOCAM architecture mainly consisted of Context Providers, Context Interpreter, Context Database, Service Location Service and Context-Aware Mobile Services. Context Providers were responsible for context

abstraction to separate the low-level sensing from the high-level manipulation. All the information from Context Provider was processed and interpreted, by Context Interpreter, into the form of OWL descriptions, and was stored in the Context Database for future retrieval. Context-aware mobile services were applications and services that consumed different level of contexts and adapted their behaviour. Service Locating Service was important for the interaction between different components in that different providers registered and advertised their services through it so that context interpreter or mobile services were able to locate a context provider and to obtain context. Context interpreter and mobile services were also able to register themselves to the Service Locating Service in order to be discovered and accessed by other context-aware systems.

The Context Toolkit developed by Dey and Abowd [8], was another example of hierarchical framework but with peer-to-peer architecture using one centralized discoverer. The distributed sensor units, called widgets, which were sensor software wrappers or agents similar to the blackboard media in [6], had the ability to standardize the sensed context, to transform the information into a uniform representation, and to provide them to components or applications that consumed the context. One feature of widget was that it could hide the low-level sensing mechanism and details. Context Interpreter was responsible for context abstraction, i.e. transforming the valued context data (temperature) into semantic context (hot, cold, etc.). Moreover, it could fuse the contexts from multiple sources to develop more complex context representation. Context Aggregator was more specialized in collecting multiple context sources and in making decision about a single entity, e.g. person, place or object, thus facilitating the access of several contexts of a single entity. The Context Toolkit structure is illustrated in Fig. 1.3.

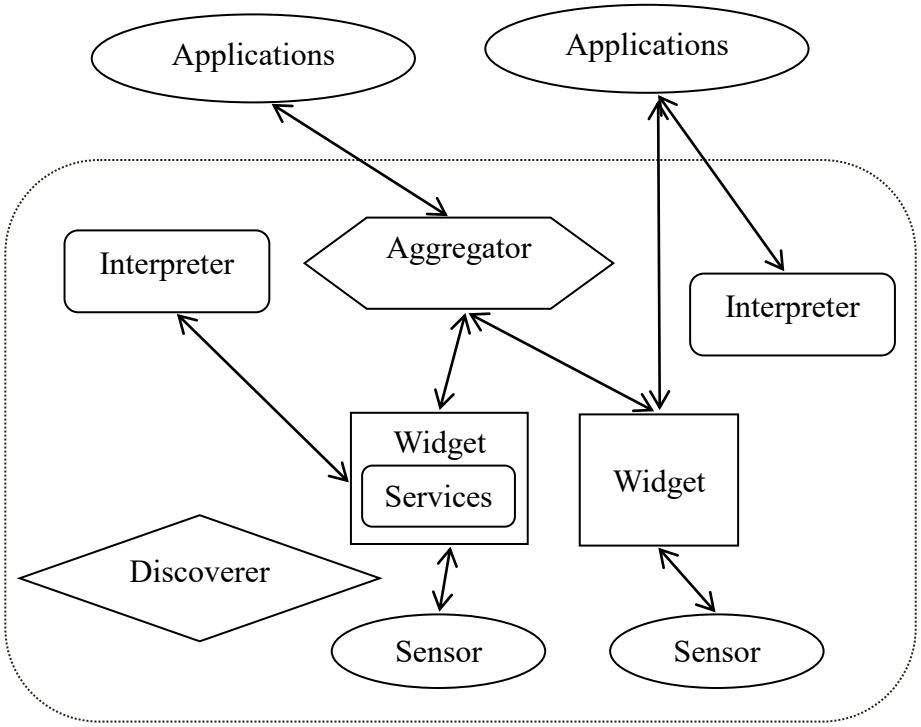


Figure 1.3: The structure of the Context Toolkit

In addition, Fahy and Clarke [9] presented the project Context-Awareness Sub-Structure (CASS) as an extensible centralized middleware approach, in which the centralized part consisted of four components: Interpreter, Context Retriever, Rule Engine and Sensor Listener. The Listener gathered context from sensors and the Retriever retrieved the gathered context. Both of them used the Interpreter services. Roman *et al*. [10] designed the Gaia project by extending typical operating system to include context-awareness. In this system, the components concerning context-awareness were Event Manager, Context Service and Context File System. Event Manager distributed events in the active space and implemented a decoupled communication model based on suppliers, consumers, and channels. With the use of the Context Service,

applications can query and register for particular context information and higher-level context objects. Context File System used application-defined properties and environmental context to execute tasks that were otherwise performed manually with additional programming, facilitating the provision of services. It presented context as directories, in which the path was used to represent context types and values.

One possible disadvantage of the centralized architecture is that the system may be over-dependent on the central context agent, creating a "bottle neck" in large applications. In order to overcome this limitation, Chen *et al*. [11] developed Context Broker Architecture (CoBrA) by adopting a fault-tolerance design. This architecture also adopted the idea of central intelligent context broker for maintaining and managing the contexts. However, in order to be fault-tolerant, a group of brokers, instead of a single broker, was created as a broker federation. The federation managed a shared contextual model for the community of agents that were corresponding to different applications, e.g. mobile wearable devices, services for auxiliary contexts, or web services for the presence of users. These brokers were responsible for each part of space and were able to exchange contexts between each other. Each broker consisted of four functional components: the Context Knowledge Base, the Context Inference Engine, the Context Acquisition Module and the Privacy Management Module.

1.3.2 Robotic Context-aware Systems

In addition to the area of mobile distribution computing, in the past decades the interests of context-aware research is increasingly falling into the robotic control of the intelligent assistive applications. This idea of context-aware robotic system conforms to the trend of

intelligent robots, particularly in the highlighted assistive applications. Lacey and MacNamara [12] developed a context-aware shared control of robot with mobility aid for the elderly blind. This proposed control solution used the Bayesian network to provide context-aware shared control. The main approach was the combination of Bayesian network and user input with high-level contexts to infer an estimate of user's navigation goal. In [13], Khan *et al*. researched on context-aware navigation using sensor association rules. Specifically, data mining methodology was adopted for retrieving significant frequent patterns and was extended to facilitate robots to learn and navigate on unknown terrain in a natural way. Röning *et al.* [14] designed an agent-based architecture for context-aware service management. This architecture was implemented on a personal tele-operated robot, in the application of intelligent home assistance and remote monitoring of the aged. In other cases, robot with the ability of following people is required in assistive service provision application. Yuan *et al.* [15] developed context-aware human-like following behavior for domestic robot by adopting three basic following behaviors: path-following, direction-following and parallel-following. Apart from the industrial robot, context-aware robot can be considered as service robot to provide intelligent service. Hong *et al.* [16] introduced the context-awareness for a network-based service robot by implementing the Context-Aware Middleware for URC Systems (CAMUS) [17, 18], based on which different types of service robots could be designed. Applications included robot greeting, home monitoring, personal aid, and human-following, etc.

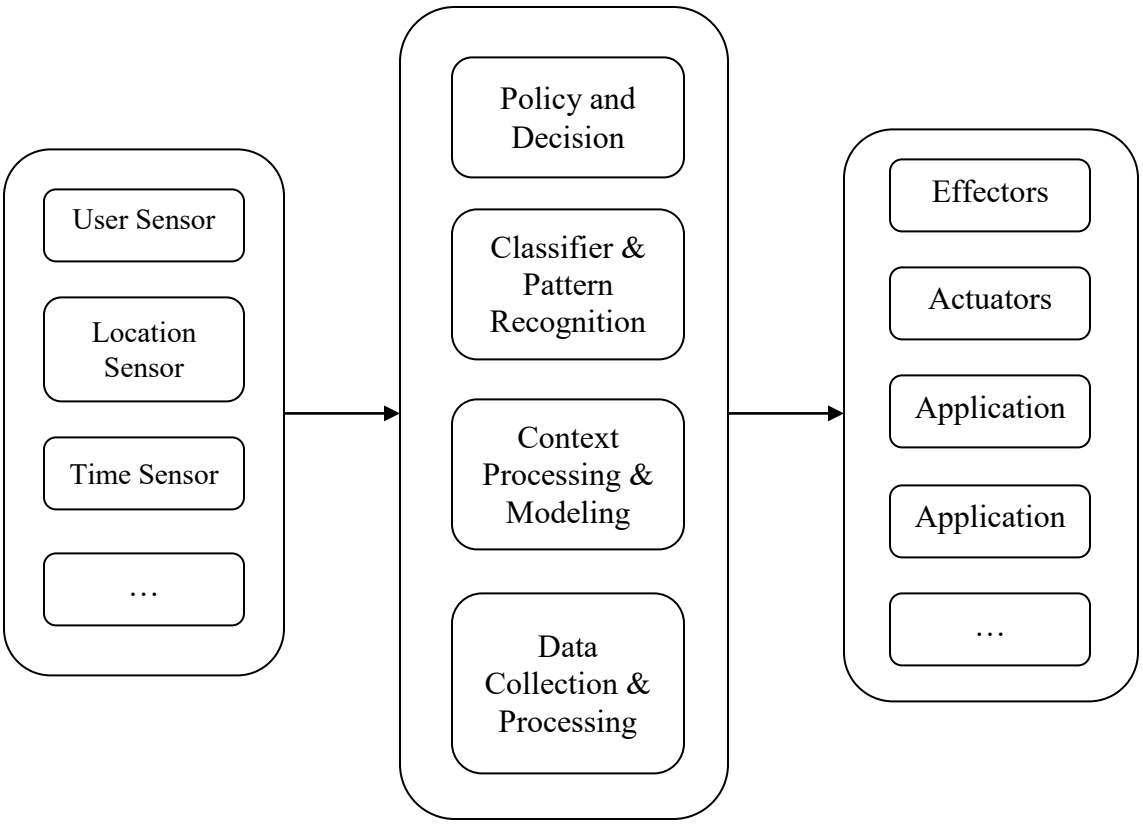


Figure 1.4: The conceptual structure of the layered and centralized context-aware robotic system

As mentioned in the above sections, the context-aware system structure can be divided into different layers in terms of the structure view. On the other hand, in terms of the function view, it can be divided into several components, each of which is responsible for its own task. In this thesis, by considering the specific functions and tasks that the robotic system should have and perform, a conceptual example of the layered and centralized context-aware robotic system structure is designed, shown in Fig. 1.4.

Similarly, in the sensor node layer, sensor components are responsible for detecting raw data from the environment, e.g. collecting signals either from locally mounted

sensors such as camera, sonar, laser, or from other remote sensors (e.g. body signal sensor) that detect user and environment conditions. These raw data are processed in the central agent layer where signal processing components are adopted. The processed data are also modeled and converted into high-level contexts, and then corresponding decisions are made based on reasoning rules. Finally, these decisions are fed into the application layer for task execution and service provision.

1.3.3 Requirements of Context-aware Robotic System

In order to develop the context-aware robotic system, its relationship with traditional context-aware system should be highlighted. Many of the previous context-aware research is in the field of mobile computing application, and there are several distinctive properties and requirements in the context-aware robotic system.

1. The previous context-aware mobile systems are concerned with various types of information, or contexts. These contexts include locations, time, user identities, environment, etc. However, although the context-aware assistive robotic system still takes the above contexts into consideration for the execution of the normal tasks, it also focuses more on the users and is more specialized in user contexts, e.g. ECG signal, blood pressure, heart pulse, etc. The context-aware assistive robot can be considered as a traditional context-aware system that is more user- and assistance-oriented.

2. Since the context-aware robotic system is more interested in assistive applications, it can focus on fewer types of contexts than in traditional context-aware mobile systems. Therefore, sensor fusion technique, which is required in case of large amount of sensor information, is not so necessary in context-aware robotic system,

making the whole system more concise and with better real-time quality, fulfilling fast-response requirement in the assistive applications.

3. A large portion of previous context-aware systems mainly has the function of displaying contexts, since the primary objective is to present the information that is possibly required by users. Thus, the users often need to complete the final tasks by themselves in the physical world. Since the context-aware robot is with the task-execution ability, it can execute the tasks required by users, especially when users' abilities are not powerful enough or even are not available. This task-execution ability also conforms to not only "using contexts", but also "adapting to, or changing contexts".

These requirements can be considered as the guidelines when designing the context-aware robotic system for assistive purpose, i.e. selecting the user related contexts and providing physical services for the users. All these will be reflected in the contexts that are researched and implemented in the later chapters of this thesis.

1.3.4 Design of Context-aware Robotic System

Considering the general features and requirements mentioned in the above sections, the general context-aware robotic system design with the layered and centralized architecture is shown in Fig. 1.5, based on the structure of Context Toolkit [8].

Local sensors contain sonar, camera, laser, etc., whose data can be directly accessed by middleware infrastructure. Remote sensors can detect the body signals from the target people, e.g. ECG, EEG, for monitoring purpose, as well as the auxiliary signals, e.g. light, humidity, thermal condition of remote areas. The direct human input can be sensed through a properly designed GUI. This input is used as the original system status, e.g. the

intention of user, or in some emergent situation, e.g. emergency stop. All of these signals are collected by software widgets, which consist of the necessary drivers and programs dedicated to the corresponding sensors.

The interpreter has the function of context processing to enhance the quality of the raw data, and inferring high-level context based on the data and making corresponding decisions. This component implements the techniques including signal filtering, context modeling, context reasoning, decision making, etc. The Context History & Storage component is acting as a database which is responsible for storing previous contexts. Therefore, the contexts can be fast accessed and retrieved in case of estimating user preferences and providing proactive personalized services. This component can be implemented based on universal database or other specific software, depending on the application requirements. The central component, aggregator, regulates all the necessary components and executes the required tasks on the mobile robot platform for applications of target tracking, object localization, auto-navigation, etc. As one example, the implementation of the visual tracking and position monitoring can be designed with the utilization of local sensor, middleware widget, interpreter of tracking and locating algorithm, aggregator of robot OS, context history of the past states of target, and corresponding applications.

Since the robotic system is user service oriented, the context-aware robotic system places more focus on the user contexts, which by nature are difficult to reliably collect due to its different features, domains, and requirements. Therefore, most research work in the current literature only focuses on a specific domain of contexts, which however requires some extra adjustment before being implemented on other domains. The

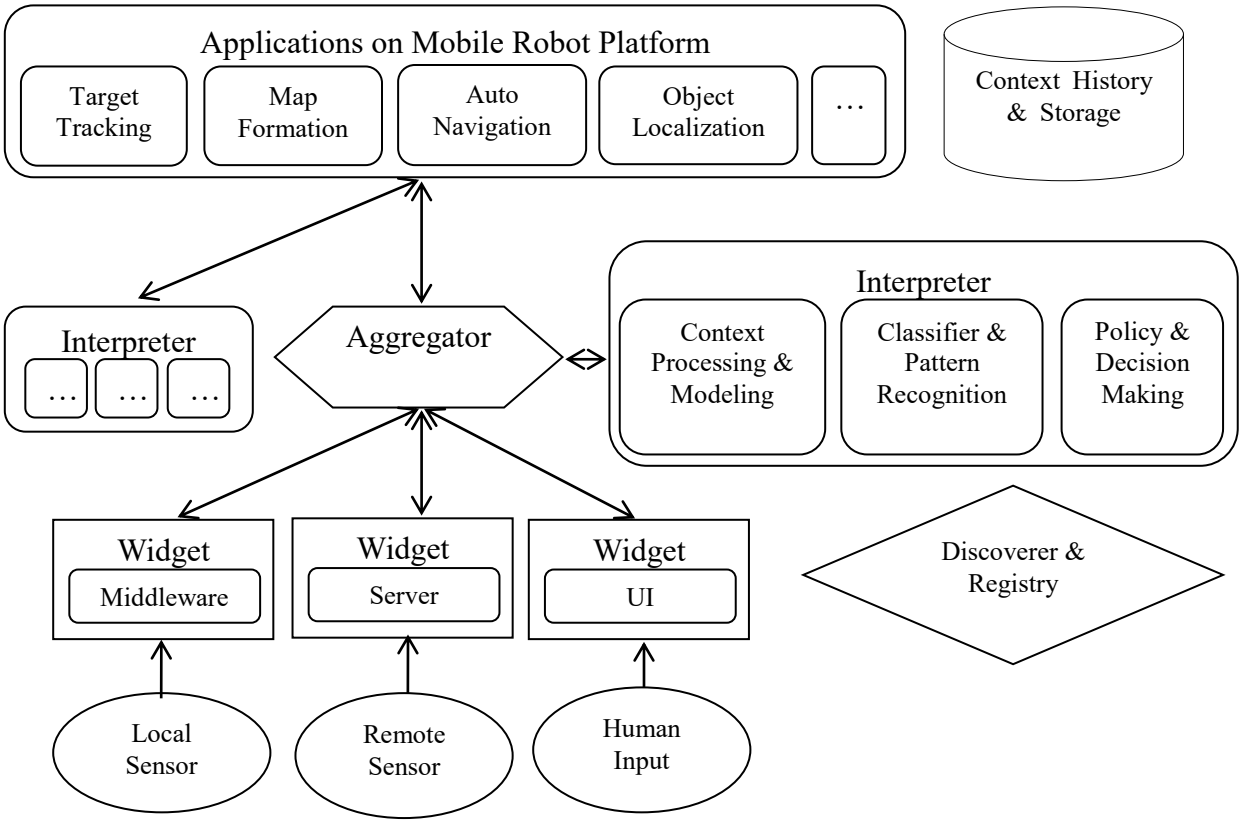


Figure 1.5: The context-aware robotic system design with the layered and centralized architecture

proposed structure provides a general structure to adapt to different domains of context, namely local, remote and human input. And more specific types of contexts can be included without large amount of extra effort. Besides, the proposed system only adopts the components and modules that are related to user service provision. It only requires the interpreter for context reasoning and the aggregator for context management, and the application layer only has user related services, thus making the proposed system more efficient in processing user contexts and proving corresponding services. Moreover, the application layer acts as the interface between context processing and context implementation. In this way, existing application can be removed or modified and

additional applications can be added with comparatively little effort. While in most current literature, the system is usually designed with respect to only a single application.

Overall, the proposed system structure provides a general guideline the design of context-aware robotic system. All the advantages are aligned with the requirements of the context-aware robotic system, which is elaborated in the previous section.

1.4 Research of Context-awareness

In the broad research area of context-awareness, a lot of investigation focuses on the approaches to the context data representation and context inference, namely, the context modeling and context reasoning.

Context modeling, which is related to the topic of knowledge representation, is required because, in the context-aware environment, all the involved entities are interconnected together in some way and there is a large amount of context information exchanged among them in a real-time manner at the background. Unfortunately, all the contexts are collected from various sources that often bear different and incompatible features with each other. Moreover, due to the vast complexity of the context-aware applications, there are various requirements as to how the context information be represented. Therefore, there should be a universal and formal scheme in which all the sensor information collected from the surrounding environment and consumed by applications can be represented as a general form that is understandable by all the involved lower and upper entities. A good context modeling scheme has the advantage of

reducing the context-aware system complexity and improving the system maintainability and evolvability, thus making the context information easy to reuse and share.

There has been extensive research in the context modeling. The approaches of modeling differ, in terms of the ease of usability by the applications, the expressive power of environmental information, the support to the system construction, the scalability of future management, etc. However, since this thesis mainly focuses on the issue of context reasoning, only a brief review of context modeling is provided. Early modeling approaches include the key-value and markup models. The key-value models use the simple key-value pair to describe the corresponding entity-attribute pair in the context-aware environment, while the markup models use more formal markup languages to perform this task, such as XML, the CC/PP standard [19]. Although these models are easy to use, they have limited capabilities to cover a variety of contexts and to represent the sophisticated relationship, timeliness and quality of contexts. Domain-focused modeling represents the various types of contexts by their application domains, such as Who, What, Where and When, the so called W4 model [20]. All contexts are then grouped and managed according to their different domains. Although this model works well in some domain-specific context-aware applications, it inevitably lacks the property of generality and cannot be utilized in other non-domain-specific scenarios.

More advanced context modeling approaches include the object-role based models, spatial models and ontology-based models. The object-role based models have their roots in database techniques and focus on the Context Modeling Language (CML) [21]. The CML provides a graphical notation to satisfy the requirements of context-aware system. The main strengths of CML are the capabilities of capturing the different classes

and sources of contexts, representing the imperfect contexts, and storing the dependencies and histories of certain contexts. However, since all the contexts in CML are represented in a flat style, the hierarchical structure, in which one domain of contexts is higher than another, cannot be well supported. The fundamental of spatial models lies in the fact that space is an important context in many applications and the contexts in this model are organized by their physical locations which can either be a geometric coordinate such as a GPS data or a symbolic coordinate such as a room number. The spatial models can also have hierarchical structure by using different semantic levels [22]. Although the location-based modeling provides many benefits to location-based and even non-location-based applications, one possible weakness is that it takes effort for the applications to collect reliable location information and keep it up to date. Ontology-based models treat the contexts as a specific type of knowledge and ontology, a type of description logic [23], is used to model the contexts and their relationships. The basic formalism of ontology-based model is OWL-DL [24], in which classes, individuals, characteristics and relations can be modeled. It has been adopted by several context-aware architectures, such as the SOCAM [7], GAIA [10] and CoBrA [11], mentioned in section 1.3.1. The OWL-DL indeed has a strong expressive power. However, it still has trade-offs. In some cases, the OWL-DL has the efficiency problem and therefore the real-time quality of the system cannot be always guaranteed.

On the other hand, the importance of context reasoning lies in the fact that, in many cases, the raw data directly collected from the sensors are basic and trivial, and are vulnerable to small disturbance. Moreover, they are normally so atomic that each of these contexts only provides a single piece of information that is usually too limited and too

specific to be efficiently utilized by the applications. For the example of location-awareness, the position data provided by the GPS device are in the form of grid, which are hardly meaningful to human users. Therefore, context reasoning or interpretation is required to abstract the high-level contexts from low-level ones. These high-level contexts are often called the situational contexts, or simply situations [25]. In the above example of location-awareness, a simple reasoning procedure can translate the grid number into a name of street in the case of outdoor location, or into a room number in the case of indoor location, where the construction of a mapping between grid numbers and names are required. Moreover, the initially reasoned context of location can be combined with other contexts to infer more advanced contexts by other reasoning techniques. For instance, after combining the location context of the user with the user's identity, schedule or preference, it can be inferred if the user is at the right location at the right time.

The topic of context reasoning, as an important research area of context-awareness, has been extensively investigated. In [26], the context reasoning techniques were surveyed along with context modeling, and the context reasoning were reviewed according to three main types, namely the object-role based context, space based context, and ontology based context. A more concurrent work is presented in [25], where the context reasoning was researched according to two streams, the specification-based techniques and the learning-based techniques. In this thesis, the context reasoning algorithms are reviewed with a newly proposed taxonomy. Specifically, the context reasoning is categorized into the non-probability-based reasoning and the probability-based reasoning, along with the argument and analysis as to why and how the

probability-based techniques are chosen as the main context reasoning in this thesis. More details will be presented in Chapter 2 as the literature review on the topic of context reasoning.

1.5 Contributions of the Thesis

The main contributions of this thesis include:

1. A new layered and centralized structure of robotic context-aware system is designed based on the discussion of the state-of-art research on general context-aware systems. The robotic context-aware system has centralized components of widgets, interpreters, aggregators and applications, which are structured in the layers of context acquisition, context processing and reasoning, and application. The designed structure can be implemented in many assistance-oriented applications, e.g., to provide support to elder people with moving disability, to tracking and monitor patient's state and location.

> [89] K. Wang and X. P. Liu, "Visual object tracking based on filtering methods," in Proc. Instrumentation and Measurement Technology Conference (I^2MTC), Hangzhou, China, May 2011, pp. 1–6.

2. The context reasoning techniques are extensively researched based on a new taxonomy proposed in this thesis. This new taxonomy classifies the reasoning techniques according to the theory base of the techniques which are either probability-based or non-probability-based.

3. A novel probability-based technique of proactive particle sampling is developed. This technique is based on the methods of adaptive multiple system state transition model and the fuzzy-logic-based velocity estimation to update the proposal distribution of the particle filter. The model transition matrix is updated in a way that model with higher potential is assigned with a higher transition probability. Meanwhile, each transition model is adjusted by the fuzzy-logic based velocity estimation to keep track of the most recent changes in target motion.

[107] K. Wang and X. P. Liu, "Particle Filtering Enhanced Human Tracking on Context-Aware Robotic System," Proc. Int'l Symp. Haptic Audio-Visual Environ. Games, Oct. 2013, pp. 92-97.

[108] K. Wang and X. P. Liu, "Particle Filtering-based tracking and localization on context-aware robotic system," Proc. Int'l Conf. Computer Science & Education, Vancouver, BC, 2014, pp. 229-234.

4. The active system observation approach is developed. In this approach, instead of treating all the system observations equal, each system observation is evaluated and only those with higher quality are incorporated into the final system belief. Therefore, the system can actively select the observations to strengthen the belief and to reduce the effects of uncertainties. Entropy is adopted as the metric to evaluate each system observation.

[109] K. Wang and X. P. Liu, "Adaptive multi-model and entropy-based localization on context-aware robotic system," IEEE Int'l Conf. Systems, Man, and Cybernetics (SMC), San Diego, CA, 2014, pp. 3755-3760.

5. An enhanced version of path planning algorithm is proposed based on the Monte Carlo partially observable Markov decision process, namely the MC-POMDP. Specifically, the belief state space, or particle set, is augmented by incorporating the index of entropy, which is further integrated into the Monte Carlo computation of the value function to alleviate the negative impact of the uncertainties in the belief space. The value function is represented by expectation of the distances between the particles that are directly sampled from the query set and the reference sets. The values of each action are represented as weight, which guides the selection of the corresponding action for next update cycle.

1.6 Thesis Outline

In this introduction of Chapter 1, the background of the study, the concept of context and context-awareness, the review of context-aware systems, and the main research issues in context-awareness are discussed. Chapter 1 also provides the objectives and outline of the thesis, as well as the main contributions.

The literature review is presented Chapter 2, focusing on the main topic of the context reasoning techniques. Several taxonomies of review of context reasoning are presented, along with a newly proposed taxonomy, namely probability-based reasoning and non-probability-based reasoning. An extensive amount of reasoning techniques are reviewed and analyzed pertaining to this taxonomy. Chapter 2 also provides the reason why probability-based approaches are selected as the main context reasoning techniques

for the probabilistic robotic system, as well as the scheme of reasoning for the implementation of context-aware assistive robotic system in the following chapters.

The two main parts of the probability-based particle filter reasoning approaches are the system dynamics and system observations. The issue of system dynamics is investigated in Chapter 3 in which a new dynamics model is proposed to closely track the system state transition. The system observation is researched in Chapter 4 to deal with the issue of observation noise and how to accommodate the system to alleviate the effects of the uncertainties.

Chapter 5 reviews the Markov Decision Processing (MDP) and the partially observable MDP, and presents and illustrates the enhanced version of Monte Carlo POMDP for the implementation of assistive robotic system. The enhancements come from the augmentation of the belief state space, value function computation, and non-greedy action selection.

Chapter 6 firstly presents and compares the performance of the standard particle filtering and the proposed particle filtering in the experiments of human tracking and self-localizing. Then the results of the traditional MC-POMDP and the proposed MC-PPOMDP in the experiments of autonomous path planning are illustrated, along with the performance analysis to prove and validate the feasibility and effectiveness of the proposed algorithms.

Chapter 7 concludes this thesis and summarizes the future works.

Chapter 2

Context Reasoning and Inference

As one of the main topic of context-awareness, the methodology of context reasoning has been extensively investigated in the past decades. Tremendous techniques of context reasoning were developed with different technical bases, and have been widely implemented in various scenarios. Therefore, at this point of view, the work of the review of most, if not all, of these techniques is a matter of necessity. This issue is covered in this chapter. First, the literature review of other researchers is briefly introduced. Then the review is performed based on a newly proposed criterion, which classifies the reasoning techniques into the two categories of probability-based and non-probability-based. At the end of the chapter, the reason why the probability-based reasoning techniques are selected for the implementations later in the thesis is provided, along with the basic scheme of context reasoning that is adopted.

2.1 Related Work of Context Reasoning Techniques Review

Bettini et al. in [26] surveyed the context reasoning techniques in conjunction with different context modeling methods. Specifically, the authors classified the reasoning techniques into three categories, namely, the object-role based, the spatial model based and the ontology based. The object-role modeling originates from the roots in database modeling techniques and supports for reasoning in the forms of assertion evaluation and database-like queries. The spatial model based reasoning focuses on the important context of space in many context-aware applications. It allows the reasoning about the context of location and spatial relationships of objects in the form of three typical queries, which are position, range, nearest neighbor. Besides, the spatial model can facilitate the partition of large amount of context information along the spatial dimension. For the ontology based reasoning, the benefit lies in that it is possible to automatically derive new knowledge using the pre-defined classes and properties, and that it is able to detect inconsistencies in the context information as well, which is crucial in the ontology definition. Overall, it is a fact that the performance of reasoning algorithms are affected by the choice of context modeling method. Therefore, the advantage of this survey methodology is that it relates the context reasoning to modeling as a single research topic and that it provides guidance regarding how to develop context reasoning algorithms depending on different modeling methods that are adopted in the system design.

In [25], the context reasoning techniques were reviewed according to two mainstreams, which are specification-based and learning-based. The former was developed and implemented in the early stages of context reasoning with few sensors that provided easy-to-interpret data information. The main reasoning approach was to design

proper reasoning engines to infer required context from sensor input, using expert knowledge in logic rules. The logic reasoning belongs to this category, as well as the more advanced ontological reasoning. When taking the issue of uncertainty into consideration with more sensors deployed, the probabilistic approaches were incorporated. The main feature is to associate the level of certainty with the corresponding inferred context to deal with the imprecision and incompleteness of sensor data, as well as their unequal contribution to reasoning the required context. However, with the boot of sensor deployment, the expert knowledge used in the specification-based approaches became less feasible. Therefore, more research focused on the incorporation of machine learning and data mining techniques into context reasoning. The Bayesian derivative models, including the naïve Bayes, Bayesian networks, HMM, etc., are in this category, as well as the grammar-based approaches, such as context free grammars. Furthermore, in order to deal with the requirement of large amount of precious training data, the data mining techniques are applied to uncover the relationships between data and contexts. These techniques include suffix-tree, Jeffrey divergence, etc. Overall, in addition to surveying the traditional reasoning methods, this review method also covers the topic of how to integrate machine-learning algorithms into context reasoning, which has a strong potential in the future development of context-awareness.

A similar survey is conducted in [27], where the reasoning techniques were reviewed in the categories of ontological reasoning, rule-based reasoning, and distributed reasoning. The key techniques in this review were those that employed Semantic Web-based representations, due to the efficient and strong capability of context description, sharing and reuse. Moreover, the survey also pointed out that the advantages of

distributed reasoning were better control, fast design and easy maintenance. The alternative techniques, including learning, offline reasoning and probabilistic reasoning, were also investigated. There are also other surveys in the context reasoning, in which the reasoning techniques were divided into case-based, logic-based, ontology-based and probability-based.

Recently, more attention has been attracted to the issue of uncertainty in context information. It is a fact that the context information inevitably contains different types of uncertainties, due to the nature of sensor imperfection, information incompleteness, ambiguity, etc. Even though the contexts were precise, there still exist other issues including fast-changing and redundant information from multiple sources, which profoundly affects the usability of contexts. Therefore, the theory of probability has been implemented to cope with the issue of uncertainty and there has been a great body of work in this area. In the following sections, the criterion of non-probability-based and probability-based is used for the context reasoning techniques review.

2.2 Non-probability Based Context Reasoning

At the traditional and early stages of context-awareness, the services that many context-aware systems provide are based on one common rule: different services are according to different contexts collected. Normally the contexts that are considered in these scenarios are atomic and independent. Taking the objective contexts (environmental contexts) as an example, these contexts, including time, temperature, light condition, etc., are usually a single unit in the system, which is stable without fast change. The detection and

collection of these simple context units is usually accurate and reliable, which is able to satisfy the requirements of the context-aware system. Under this principle, some mainstreams of context reasoning techniques are reviewed in this section, including rule-based reasoning, ontology-based reasoning, etc.

Logical rules

The logical rule based approach is one of the basic context reasoning method, with the key underlying assumption that knowledge or context is discrete and thus can be modularized. It provides a formal model for context reasoning. The objective of the logic rule is to represent logical specification of context, to check the consistency of contexts in the rule base, and to expand the intelligence of the system by incorporating more context data. The main reasoning process here is to collect sensor data and then, based on the predefined rule base, to infer new context by simple logical formulae. The newly inferred context can be further utilized for reasoning at the next level [28][29]. In [28], the author introduced how to define the status of the person by collecting several necessary contexts, specified in the following rule.

$$
\begin{aligned}
&\textit{Occupied}(\textit{person}):\\
&\quad \exists t_1,t_2,\textit{activity} \bullet \text{engaged\_in}\,[\textit{person},\textit{activity},t_1,t_2] \bullet\\
&\quad (t_1 \le \textit{timenow}() \wedge (\textit{timenow}() \le t_2 \vee \textit{isnull}(t_2)) \vee\\
&\quad (t_1 \le \textit{timenow}() \vee \textit{isnull}(t_1) \wedge \textit{timenow}() \le t_2) \wedge\\
&\quad (\textit{activity} = \text{“in meeting”} \vee \textit{activity} = \text{“taking call”})
\end{aligned}
$$

In the above example, the status "*Occupied*" is inferred if the person is detected as "in meeting" or "taking call" in the time interval (t_1, t_2) specified by the *timenow*() function, on the basis of the temporal "engaged_in" fact type/relation. Based on this basic rule reasoning, more advanced work has been performed [30][31][32]. The authors in [30] extended the logical rule reasoning to multi-level. The following example shows how the situation of "in_meeting_now" is inferred.

```
if in_meeting_now(E) then
  with_someone_now(E),
  has_entry_for_meeting_in_diary(E).
if with_someone_now(E) then
  location*(E, L), people_in_room*(L, N), N>1.
if has_entry_for_meeting_in_diary(E) then
  current_time*(T1),
  diary*(E, 'meeting', entry(StartTime, Duration)),
  within_interval(T1, StartTime, Duration).
```

Specifically, the "in_meeting_now" status is defined upon two conditions or sub-status, which are "with_someone_now" and "has_entry_for_meeting_in_diary". Each of these conditions has its own reasoning rules depending on the ultimate sensor data. In this way, the sensor data can be converted into more abstract and meaningful contexts. Furthermore, the whole reasoning process can be modularized and decoupled into several phases in different levels, which can facilitate the development of easy-to-amend and easy-to-maintain system. The inferred sub-status from the middle levels can also be re-used for multi high-level contexts inference.

The idea of multi-level context reasoning was also implemented in the situation theory [33], where an intermediate level micro situation was introduced between infons and situations. Generally, the infon represents the discrete unit of information for a single unity and the situation consists of several infons. With the introduced intermediate micro situations, the hierarchical aggregation and reuse of information is realized, where the infons can be initially processed into micro situations which are then further reasoned to create the required high-level situation of users.

Although the rule-based context reasoning has been widespread utilized in many context-aware systems, it lacks the ability to deal with the contexts that are ambiguous, uncertain and fast changing. Therefore, it is common to implement the rule-based reasoning along with other necessary mechanisms.

Temporal-spatial logic

In many scenarios, the time and location information is so essential to context reasoning that very little can be inferred without the knowledge of where and when the targeted context occurs. Therefore, tremendous research on time-spatial reasoning has been carried out. In [34], the status of the user was reasoned by the introduction of the temporal operators ANDlater and ANDsim in Even-Condition_Action rules. The ANDlater operator relates two events that are happening consecutively, while ANDsim is for concurrent events. One reasoning example is show below.

IF at_kitchen_on ANDlater tdRK_on ANDlater no_movement_detected
THEN assume the occupant has fainted

where the event of at_kitchen_on represents that the user is at the kitchen, tdRK_on that the user is passing through the door between kitchen and reception area, and the event no_movement_detected that no movements of user is detected. With the two temporal operators, the three events are related sequentially and the corresponding context can be specified as "the user has fainted".

More advanced temporal relations were developed in [35] by the adoption of Allen's temporal logic [36] and the point algebra [37]. The Allen's temporal logic dealt with time intervals instead of time points. Specifically, thirteen atomic temporal relations between time intervals were introduced as in Fig. 2.1.

These relations can be used to describe how one event is related to another. For instance, if the door is opened (I_1) before the user enters the room (I_2) and the presence sensor is triggered (I_3) during the user stays in the room, we can have the following event relations: $I_1 < I_2$ and I_2 di I_3, which can further infer that $I_1 < I_3$, meaning the door is opened before the sensor is triggered. On the other hand, the point algebra uses time point to describe event relationships in a similar way, to deal with some scenarios where the time interval logic is intractable.

In terms of spatial reasoning, there are a number of calculi, including topology, where the Region Connection Calculus (RCC) is used to describe the spatial relationships between different regions, as well as the spatial arrangements in each region. For example, the binary predicate C*(x,y)* indicates whether two regions or objects *x* and *y* are connect or not [38], and TPP(*x,y*) indicates *x* is tangential proper part of *y* [39]. Based on these binary predicates, one simple reasoning rule can be realized as below.

Relations	Illustration	Interpretation
$I_1 < I_2$ $I_1 > I_2$	I_1 ---------- ---------- I_2	I_1 before I_2 I_2 after I_1
I_1 m I_2 I_1 mi I_2	I_1 ---------- ---------- I_2	I_1 meet I_2 I_2 met by I_1
I_1 o I_2 I_1 oi I_2	I_1 ---------- ---------- I_2	I_1 overlap I_2 I_2 overlapped by I_1
I_1 s I_2 I_1 si I_2	I_1 ---------- ---------- I_2	I_1 start I_2 I_2 started by I_1
I_1 d I_2 I_1 di I_2	I_1 ---------- ---------- I_2	I_1 during I_2 I_2 started by I_1
I_1 f I_2 I_1 fi I_2	I_1 ---------- ---------- I_2	I_1 finish I_2 I_2 finished by I_1
$I_1 = I_2$	I_1 ---------- ---------- I_2	I_1 equal I_2

Figure 2.1: Thirteen atomic temporal relations

$$\text{NTPP}(\textit{chair, light-cone}) \wedge \text{TPP}(\textit{light-cone, living room}) \rightarrow \text{NTPP}(\textit{chair, living room})$$

where NTPP means “not tangential proper part of”.

The reliable spatial reasoning also depends on the context of distances and sizes of object. For instance, in a smart home environment, it may be required to provide direction to the elderly to get to the nearest seat. In [40] a simple distance metric such as *far* or *close* was used to provide distance information. And the delta calculus was implemented to describe the scale information between two objects, which introduced a triadic relation $x(>,d)y$ indicating that x is larger than y by an amount d. For example, *daylight*(>,*1.2*)*light-cone* indicates that *daylight* is stronger than the light from *light-cone*. This reasoning is important to the systems that are expected to adapt the light of light-cone accordingly.

The temporal logic can also be extended into the spatial reasoning. In [41], the Allen's temporal logic was generalized to reason the spatial arrangement in a region by describing the order of object along a certain dimension. For instance, based on the relation of $I_1 < I_2$, it is then possible to infer the similar relationships that object I_1 is behind object I_2, or object I_1 is inside (or on top of) object I_2 if I_1 d I_2.

Ontology

The advantage of ontology lies in its expressive ability to specify and represent the domain knowledge into well-structured terminology, providing a formal way to make them understandable, sharable and reusable. The theory of ontology is similar to the object-oriented concept in that it defines classes, individuals, attribute properties and object properties. For instance, two domain concepts "dining room" and "eating activity space" are related by the "is-a" property [42]. The properly built ontological models can be integrated into different types of reasoners to derive new knowledge from the low-level sensor data. In [43], the following simple forward-chaining reasoning rule was deployed to infer the user's status.

```
(?user rdf:type socam:Person),
(?user, socam:locatedIn, socam:Bedroom),
(?user, socam:hasPosture, 'LIEDOWN'),
(socam:Bedroom, socam:lightLevel, 'LOW'),
(socam:Bedroom, socam:doorStatus, 'CLOSED')
-> (?user socam:status 'SLEEPING')
```

As shown above, the sensor data of location, posture, light level and door status were ontologically modeled, and the status of the user was inferred as "SLEEPING". However, one possible drawback of this reasoning is that, with the same sensor data, the user's status may be "READING" instead of "SLEEPING". Apparently the above method of forward-chaining rule is unable to handle this type of ambiguity. Fortunately, the ontology can be used to deal with this issue by validating the results inferred from reasoning techniques. In [44], a similar case was researched in a smart home environment, where "RestRoom" and "LivingRoom" were the subclasses of "Room" with the property of "hasArtifact.Sink" and not "hasArtifact.WaterFixture", respectively, while "Sink" was a subclass of "WaterFixture". The user activity of "BrushingTeeth" was a subclass of "PersonalActivity" which was only possible in a location with "Sink". This pre-defined knowledge can be ontologically shown as below.

$$\text{RestRoom} \subset \text{Room} \cap \exists \text{hasArtifact.Sink}$$

$$\text{LivingRoom} \subset \text{Room} \cap \neg \exists \text{hasArtifact.WaterFixture}$$

$$\text{BrushingTeeth} \subset \text{PersonalActivity} \cap \forall \text{performedIn.}(\exists \text{hasArtifact.Sink})$$

Based on these ontological definitions, the user's location that is inferred from other reasoning techniques can be further validated. For example, if the user's activity is determined as "BrushingTeeth" and the possible detected locations are "RestRoom" and "LivingRoom", the above ontological definitions can be used to validate the two locations, which will probably filter out "LivingRoom" and keep "RestRoom".

Another significant usage of ontology is to detect the inconsistencies in the contexts [45]. In the structure of class hierarchy, the ontological reasoner can check if a class is a

subclass of tow disjoint classes, or if two classes are contradictory to each other, e.g. a car cannot be driven by two people at the same time, or a person cannot be two disjoint places at the same time. Ontology can also serve as the sematic base for some knowledge-centric software, as in [46], where the central layers of the system is constructed by using ontology to provide a homogeneous representation of heterogeneous data, thus facilitating the context interoperability and automatic high-level reasoning.

One main benefit of ontological reasoning is that it is well integrated with the ontology models that have been widely utilized in context representation, with low computational burden. However, the benefit comes with trade-off. As in rule-based reasoning, one weakness of ontology is that a pre-constructed knowledge base is expected, which requires a considerable amount of knowledge engineering effort and is difficult to update once designed. Besides, it can only deal well with simple ambiguity. When complexity grows, the ontological validation is not powerful enough to filter out all the invalid inferred results. A possible treatment to these issues is to leverage the strong representation capability of ontology and then to incorporate with other reasoning techniques, instead of only using ontology itself as a standalone solution.

Contextual graph

Another modeling approach based context reasoning is the contextual graph, which is the directed acyclic graph to represent the necessary actions to perform in accordance with the contexts. In the graph, the action nodes denote the actions to achieve the required objective and the context nodes describe the possible contextual issues of a given event.

The main advantages of contextual graph is that, it allows for a clearer representation of multiple contexts and considers the context dynamics. Therefore, it is able to deal with contexts with large structure in industrial implementations. In [47], the contextual graph based reasoning was used in the application of subway line control, where there was a great amount of actions to deal with multiple objectives, in comparison with the traditional decision tree based reasoning. The author pointed out that, compared with a decision tree, the adoption of the contextual graph was beneficial not only to the graphical simplification, but also to the ability of self-evolution to accurately represent the dynamics of context reasoning. It was also demonstrated that the contextual graph can deal with the issue of parallel sequences of actions, which is on the other hand another concern in the decision tree models.

The objective of contextual graph is to solve the issue of the mutual relationships among contexts, which is caused by the context mobility. One character of context mobility is that sometimes only part of the context changes. For example, taking the whole room as a single context, the context of people's activities in the room may change fast, while the context of temperature only exhibits small fluctuation and the context of room location remain static. In [48], this issue of context partial change was researched. Specifically, the changing part of contexts was grouped as the dynamic context and the static part as shared context. The exemplar structure of contextual graph is shown in Fig. 2.2, where the round and rectangles shapes denote the action nodes and the context nodes, respectively. The shared contexts are described in the grey rectangles that are connected to the action nodes by dotted lines.

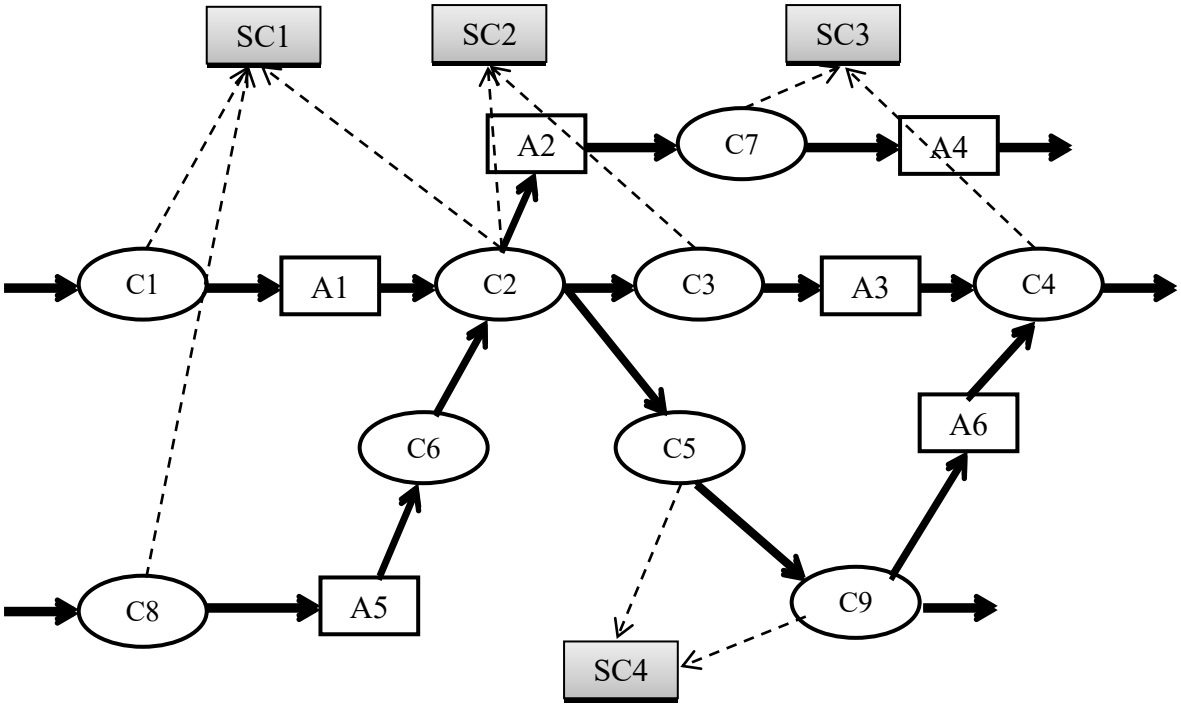


Figure 2.2: Context framework based on contextual graph

The basic process of reasoning is to determine a path from a start node to an end node and this path is corresponding to the specific way for problem solving. If an action is linked to the context that has been detected, then this action will be executed and generates a new successive context, and another possibly action linked to this new context will be carried out. This process continues until the final goal is achieved or the required level of context is inferred. The whole graph is also updated once there is new context or action detected.

Other related research works include [49] where the contextual graph was used for modeling the agent activity in the agent-based context-aware system, and [50] which described the implementation of contextual graph in decision support system in order to solve the common user interaction problems of irrelevancy and redundancy.

2.3 Probability Based Context Reasoning

The non-probability based techniques have constructed a valid base for context reasoning. They are feasible in many simple implementations where only low-level or raw contexts are considered. However, when it comes to high-level contexts, especially where the user subjectivity (personal context) is related, more advanced usage of context reasoning is required. The most distinguished character of user subjectivity is that this context cannot be detected by sensors in a direct way. For example, the motion sensor can accurately detect that a person is walking through the hall way and even detect what object this person is possibly holding, but it can hardly detect where the destination is. Moreover, in many cases, only detecting the person's movement is far from enough to generate the meaningful contexts for the system. Hence, in order to make decisions in an intelligent way, the system usually is more interested in high-level contexts, such as where the person is headed, or what task this person may be performing. Unfortunately, these contexts cannot be detected by sensor devices with full certainty. In other words, there is no single solid high-level context that can be reliably detected. For instance, with rule-based reasoning techniques, there can be many possible destinations the person is headed to, which is a confusing factor for the system to make decisions. As a result, the system sometimes has to predict the required context in some manner. Moreover, even for the objective contexts mentioned in the previous section, there exists an implicit assumption that the contexts detected are precise and complete, which is not always true in practical cases. Taking temperature as an example, even if being pre-processed, the sensed results are sometimes still not fully accurate in some adverse environmental conditions due to the noise, fluctuations, as well as the quality of sensing device. All the above factors

result in the issue of uncertainty. Uncertainty implies that in a certain situation a person does not dispose about information which is quantitatively and qualitatively appropriate to describe, prescribe or predict deterministically and numerically a system, its behavior or other characteristics [51]. Uncertainty is so common an issue that the reasoning techniques to handle it are required. This is also where the theory of probability is integrated. In this section, some main types of probability-based context reasoning techniques are reviewed.

Fuzzy logic

The fuzzy logic theory has been widely utilized to deal with uncertainty due to vagueness, by introducing the fuzzy set and the corresponding membership function. The uncertainty is reflected by the degree to which an element belongs to a fuzzy set. This type of degree can be expressed as the similarity between the element and the prototype element in the fuzzy set or as the preference of one choice over some other possible ones. In this way, the raw and imprecise numerical sensor data can be translated into linguistic variables, or fuzzy sets, that make sense to human's natural understandings, with a probability of a certain value to represent its reliability, and vice versa. For example, the specific temperature degree provided by sensor device can be translated into "cold" with a fuzzy value of 0.6, "cool" with 0.3 or "warm" with 0.1, the specific period of time into "long" or "short". Similarly, the knowledge expressed by a human observer "cold" can be translated into a corresponding numerical value. The fuzzy logic is viewed as one of the

methods that can deal with uncertainty [51] and support several operations of fuzzy sets, e.g. intersection, union, complement and modifier.

Another application of fuzzy logic for the numerical-linguistic mapping is in [52], where the fuzzy logic was integrated into the Context Spaces model to present the Fuzzy Situation Inference (FSI) for situation modeling and reasoning under uncertainty. Two basic reasoning methods were analyzed. The first one was based on weights of the contexts and level of confidence of the values, as in (2.1).

$$Certainty = \sum_{i=1}^{n} w_i \mu(x_i) \tag{2.1}$$

where $\mu(x_i)$ denotes the membership degree of the element x_i, w_i the corresponding weight assigned and n the number of elements. The result of *Certainty* represents a weighted membership degree of x_i. The second method took into consideration the inaccuracies of sensors, which calculates the *Certainty* as follows.

$$Certainty = \sum_{i=1}^{n} w_i \mu(f(x_i, e_i)) \tag{2.2}$$

where w_i is still the corresponding weight assigned while $\mu(f(x_i, e_i))$ denotes the membership degree of element x_i which also considers the sensor reliability or error rate e_i. The above two fuzzy functions were then implemented in a heath monitoring application to reason about patients' situation of "normal", "hypotension", or "hypertension".

A general model to represent semantics of uncertainty is provided in [53] at different levels, e.g. sensors, low-level contexts and high-level contexts. By modeling

context as ((*s, p, o*), *t*, *conf*), where (*s, p, o*) is a RDF triple to evaluate uncertainty, *t* is the life span and *conf* is the context confidence, the situation was abstracted as the invariant characteristics of contexts which was interpreted as constraints on context values and life span. In order to identify the possible situation, a set of contexts was evaluated to verify if they satisfied the corresponding constraints by intuitively computing the match degree in the framework of fuzzy set. This proposed method was implemented in an office environment for demonstration and evaluation. Besides, the authors in [54] involved situation awareness as an abstraction of context awareness, and introduced the use of fuzzy logic for inferring and reasoning about the users' current situation as an extension of classificatory problem solving. Specifically, ontology was used to model the situation and similarities among diverse situations were measured and reasoned about to identify users' situations. In addition, two metrics, the degree of situation involvement, which referred to the degree of belief that a user was involved in a situation, and the degree of system pervasiveness, which denoted whether the system was capable of reasoning about the situation, were introduced and evaluated by the fuzzy logic to make proper system decisions.

Naive Bayes and Bayesian networks

The Bayesian techniques are based on the well-known Bayes' theorem that can be generalized in (2.3).

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)} \tag{2.3}$$

where H is the target hypothesis, X the evidence to support the hypothesis, $P(X)$ and $P(H)$ the prior probability density function (PDF) for X and H, $P(X|H)$ is the likelihood function, and $P(H|X)$ the required posterior PDF. It can be basically interpreted as that, the probability of hypothesis H conditioned on evidence X can be obtained based on the probability of X conditioned on H, multiplied by the prior of H and normalized by prior of X. When there are multiple evidences supporting the same hypothesis, or the evidence has multiple attributes that are independent of one another, the above likelihood can be reduced to the form of recursive Bayes:

$$P(X|H) = \prod_{k-1}^{n} p(x_k|H) \tag{2.4}$$

where x_k is one of the multiple evidences or attributes.

Bayes' theorem was extensively researched and analyzed in [55], in which the authors exploited the potential of the naive Bayes Probability Estimation (NBE) for general probabilistic estimation tasks, and provided a thorough analysis and evaluation on its learning time, inference time and accuracy. The authors also argued that the NBE had a very attractive real-time quality, which was linear in the number of components and the query variables, and allowed for fast inference. There has been a broad utilization of Bayes' theorem in reasoning about contexts, e.g. users' activities and users' availabilities. In [56], the naive Bayes classifier was adopted to detect activities using the "tape-on" sensor system. It was pointed out that, although the naïve Bayes had the strong assumptions that each variable were independent and observable and this assumption sometimes made it poorly perform in real domains, the naive Bayes had good results,

compared to some more complex algorithms as decision tree, due to its low variance of classifier that offset the effect of the high bias from the assumptions. The authors demonstrated that the proposed method was capable of recognizing activities of interest such as toileting, bathing, and grooming with accuracies ranging from 25% to 89%, depending on the evaluation criteria adopted. Similarly in [57], the users' activities, such as using PC, meeting or discussing were inferred based on the evidences such as PC and phone usage, ambient sound, time and location. Other related research includes [58].

As mentioned above, the feasibility of naive Bayes relies on the strong assumption that all the evidences are conditionally independent of one another. When this assumption is broken, its performance might degrade. In cases where there indeed exist dependencies among evidences, the Bayesian (belief) network can be applied as the substitution. The Bayesian network is generally a directed acyclic graph where each node, which represents a variable, is connected to another by an arc, which represents the casual relationship between the two nodes from parent to descendent. The core component is the conditional probability distribution (CPD) which gives all possible combinations of the parent nodes of each node, $p(x|parent(x))$. The Bayesian network has been applied in many context-aware applications. In [59], by leveraging the similarity between the structures of situation lattice and Bayesian network, the authors converted the lattice to a Bayesian network, where each node and arc is corresponding to a situation and a dependence edge, respectively. The maximum likelihood estimation was applied for the CPD that was continuously updated when new observations were available. In the prototype pervasive computing infrastructure, Gaia [10], the authors also implemented Bayesian network to deal with context uncertainties.

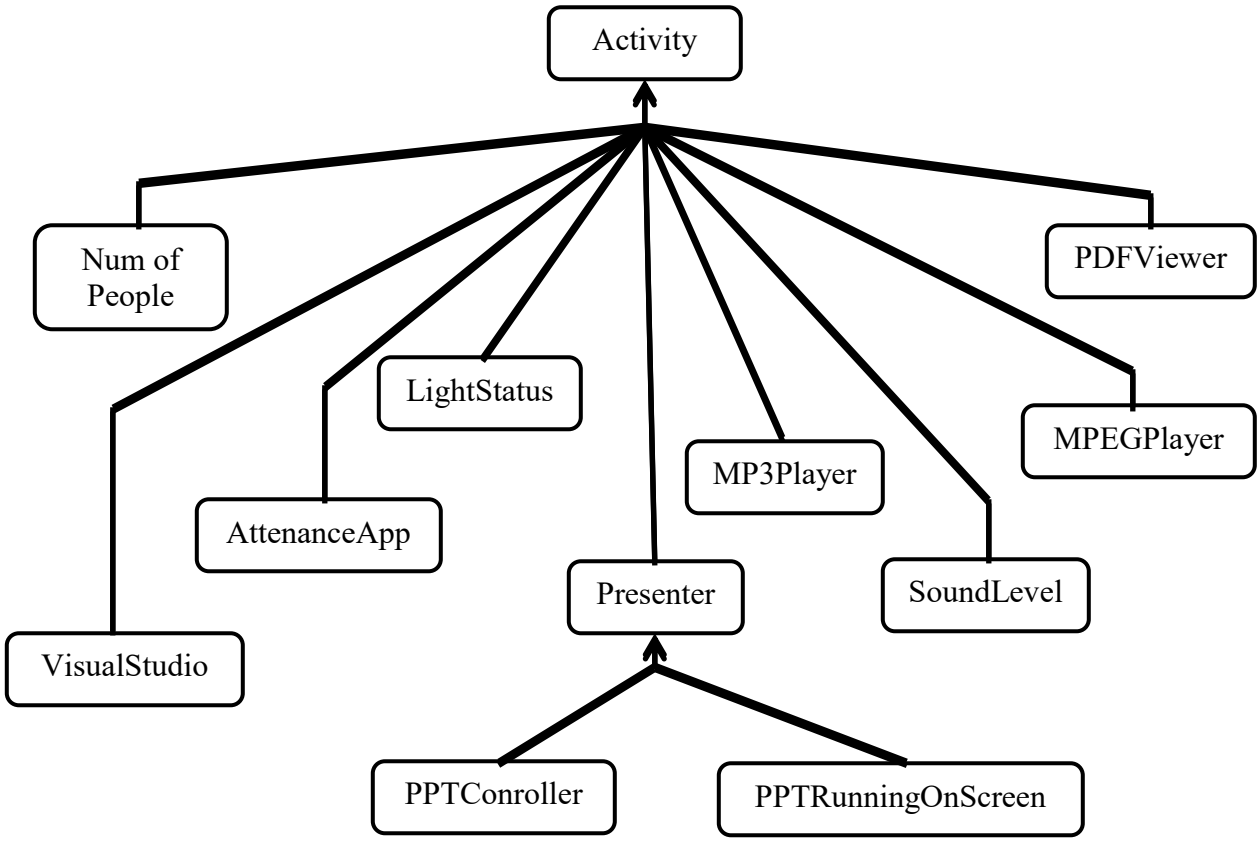


Figure 2.3: Bayesian network of inferring the activity in a room

An example is illustrated in Fig. 2.3 for inferring the activity in a room, where the root node represents the context to be inferred and the leaf nodes are sensor data. The root node "Activity" can take values of "meeting", "presentation", "idle" and so on, which are represented in the context predicates, *activity*(*room*, *meeting*), *activity*(*room*, *presentation*), *activity*(*room*, *idle*). The corresponding probabilities of these predicates are based on what the network deduces from the leaf nodes. However, in this network, there exists causal relationship between the node of "Presenter" and the nodes of

"PPTController" and "PPTRunningOnScreen", which breaks the independence assumption and cannot be modeled in naive Bayes.

Similar research was conducted in [60], where the ontology was extended by incorporating uncertainty and the Bayesian network was constructed. Other related work includes [61][62]. The main benefit of Bayesian network is that it provides a clear and well-structured framework to represent the relationships and likelihood among all possible events.

Evidence theory

The evidence theory, or specifically the Dempster-Shafer Theory (DST), is a framework of mathematical representation of uncertainty, where the key components are the mass function, the frame of discernment and the combination rule. The mass function allocates beliefs of sensor data in the frame of discernment and the combination rule fuses the multiple beliefs with the corresponding level of certainty. The most significant innovation of DST compared with the normal probabilistic methods is that, it allows probability to be assigned not only to a single element, but also to a set of elements when there is not enough information to distribute belief more precisely to individual element [63]. In contrast, the normal probabilistic methods only equally divide belief among all elements.

The principle of using DST is that, the low-level sensor data is used as the evidence of high-level contexts or states in the pre-defined activity model, and the high-level contexts are then fused to infer more complex contexts until the top level of context is reached or the required level is satisfied [64].

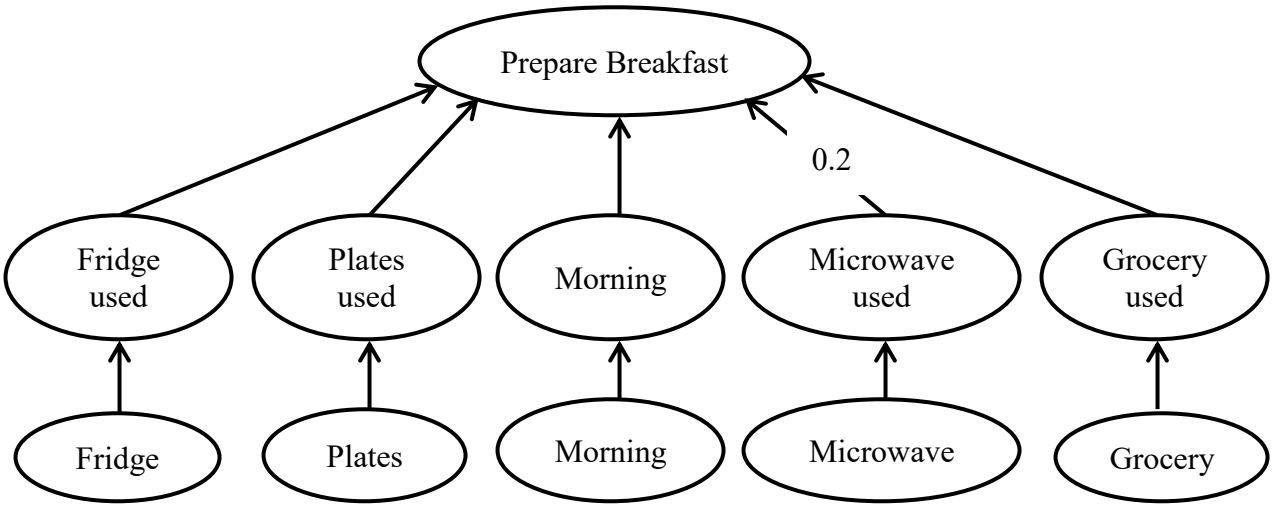


Figure 2.4: Pre-defined evidential network based on expert knowledge

There are two basic procedures in DST. First, a pre-defined evidential network based on expert knowledge is required to describe how sensor data lead to higher contexts. One example of this network in [64] is illustrated in Fig. 2.4 as a tree hierarchy, which described how the user's activity of "prepare breakfast" was inferred based on five contexts of "fridge used", "plates used", "morning time", "microwave used" and "groceries used". These five contexts were resulted from their specific sensors deployed in the smart home environment. The value of "0.2" represented the occasional use of microwave in breakfast preparation. At the bottom layer, the mass functions of the deployed sensors assigned mass to the context values as follows:

$\{FirdgeUsed = 1, \neg FridgeUsed = 0\}$
$\{PlatesUsed = 1, \neg PlatesUsed = 0\}$
$\{MicrowaveUsed = 1, \neg MicrowaveUsed = 0\}$
$\{GroceryUsed = 1, \neg GroceryUsed = 0\}$
$\{Morning = 1, Evening = 0\}$

For the situation of “Prepare Breakfast”, each context value belief was propagated above as evidence as follows:

$\{FirdgeUsed = 1, \neg FridgeUsed = 0\} \rightarrow \{Breakfast = 1, \neg Breakfast = 0\}$
$\{PlatesUsed = 1, \neg PlatesUsed = 0\} \rightarrow \{Breakfast = 1, \neg Breakfast = 0\}$
$\{MicrowaveUsed = 1, \neg MicrowaveUsed = 0\} \rightarrow \{Breakfast = 0.2, \neg Breakfast = 0, \theta = 0.8\}$
$\{GroceryUsed = 1, \neg GroceryUsed = 0\} \rightarrow \{Breakfast = 1, \neg Breakfast = 0\}$
$\{Morning = 1, Evening = 0\} \rightarrow \{Breakfast = 1, \neg Breakfast = 0\}$

Then the total belief of the situation was obtained via evidence combination. The authors compared two approaches, the simple evidence averaging and the Dempster’s combination rule, by which the belief of “prepare breakfast” was inferred as 0.64 and 0.98, respectively. A similar work can be found in [65].

There are also extensive researches on improving and extending the original DST. In [66], the authors extended the DST with the general UML-based context quality model and the aggregation model. These models provided a flexible and generalized way for context-aware system to incorporate the context quality into the design process, and identified quality issues as well as their aggregation across different context layers. Besides, the time extended mass function was introduced in [67], where the duration of an inferred activity was used to extend the lifetime of low-level evidence that were then fused to infer the high-level situations. In [68], the authors focused on how to improve the performance of DST approach by reducing the computational overhead. The method was to use the K-L evidence selection strategy to filter out the less important evidence and only selected those with highest belief values.

However, there are also some weaknesses of DST. One is that it largely depends on the availability of two main factors, which are the pre-defined structure of evidence

hierarchy (activity model) and the quantified uncertainty of sensor data. Both of the two factors heavily rely on expert knowledge, which undermines the performance of DST if this knowledge is incomplete or unavailable.

Statistical classification

In order to satisfy the fast growing of context complexities, methods that are able to deal with large quantities of data are required. Amongst them, the feasibility of utilizing the classification algorithms in context data reasoning is researched in many works. By means of classification, the dimension of the large quantity of context data can be reduced to be tractable so that they can serve as the final required reasoned results to trigger the corresponding system actions or as the intermediate knowledge to higher level of reasoning.

There has been a lot of research on classification algorithms, including linear classifier, support vector machines, kernel estimation, decision trees, etc., as well as their implementation on context reasoning. The decision tree is a predicative model with the structure of leaf and branch, where a leaf represents a classification and a branch describes a conjunction of features that lead to the target classifications. Due to its capability of generating classification rules that are easy to understand and explain, it is useful in analyzing sensor data and feature extraction, e.g. the human activity classification and recognition [69][70][71]. In [69], several features were extracted from the data of accelerometers worn by users, including mean, energy, frequency-domain entropy and correlation of acceleration data, by means of FFT-based feature computation.

The recognition of users' physical activities was performed by using different classifiers, including decision table, instance-based learning, C4.5 decision tree, and naïve Bayes classifiers. By comparison, the authors proved that the decision tree yielded the recognition results with the highest rate of accuracy. However, the training time for the decision tree was relatively slow, especially when compared with its short running time. The authors also pointed out that although the decision tree was able to recognize the activity, e.g. "walking", but it might not readily recognize the activity style, e.g. "walking slowly" or "walking fast". In [70], to recognize real-time human movement, the authors developed a hierarchical and modularized structure of decision tree, where the broad classifications were made on the top level and more detailed sub-classification were on the lower level. Moreover, the author suggested the validation of the classification sequences by using rule logic or statistical modeling. Another research on the decision tree based human movement classification from body acceleration data was conducted in [71], where a generic framework of classification was based on a hierarchically structured binary tree with classes and sub-classes in different levels. Specifically, the two main classes of movements were activity and rest. Activity included waking, running, and falls, and rest included standing, lying, and sitting. Then movements were sub-classified until the required level of details was satisfied. The author argued that the advantage of the modularized binary decision tree was that the classes or sub-classes can be added or removed without affecting other parts of the tree and individual algorithms can be modified without the need to change others.

When both linear and nonlinear data is concerned, the support vector machine (SVM) can be applied, which firstly uses a nonlinear mapping to transform the original

data into a higher dimension, and then searches for an optimal linear hyper plane to separate the data into different classes. It has been approved that different classes of data can always be separated if the nonlinear mapping is with sufficiently high dimension. In [72], the location and activity of people in a smart home environment was inferred by detecting the usage of certain electrical devices such as light, TV set, fan, stove, etc., where the SVM was constructed based on the toolkit in [73]. The authors proved that this approach was able to learn and classify various electrical events with 85%-90% accuracy. It was also pointed out that the advantage of SVM was its ability to deal with the cases where the target feature space was fairly large compared to the training set, thanks to the over-fitting protection employed. In [74], human behavior anticipation was researched based on the analysis of motion primitives such as walking, running, going straight. Specifically, the trajectories were tracked by particle filtering. Then, according to velocity, direction and shape features, these trajectories were classified by SVM into fast-walk, idle-walk, wandering, stop, etc., to further infer a higher level understanding of people's global behavior, e.g. entering through the north door, walking across the room, or sitting at the desk.

Probabilistic filtering

The probabilistic filtering, or more specifically the Bayesian filtering, is the class of filtering methods that adopt the Bayesian theory to continuously and recursively detect and update the system's belief about the surrounding environment. It is usually implemented on the low-level context reasoning which directly deals with the raw sensor

data. For example, it has been widely utilized in the applications of robotic self-localization by reasoning about the sonar or range-laser sensor data. Generally, it can be formulated as a technique of probabilistic inference. It is also known as the so-called Markov estimation, since the dynamic system in the Bayesian filters is usually assumed to be Markov in order to make the computation tractable. The basic structure of the probabilistic filter is mainly consisted of two components, the system transition and sensor observation, where the system transition is for estimating the belief and the sensor observation is for correcting the belief, respectively. This process can be formulated as below.

$$\begin{aligned} x_t &= f_t(x_{t-1}, \epsilon_t) \\ z_t &= h_t(x_t, \gamma_t) \end{aligned} \tag{2.5}$$

where, x_t represents the system state or belief and z_t the observation at time step t. $f_t(\cdot)$ and $h_t(\cdot)$ are the system transition model and observation model. ϵ_t and γ_t are the processing noise and observation noise, respectively.

Traditional methods of probabilistic filtering include the well-known Kalman filter. Although it has been widely implemented in many scenarios, it can only have good performance in the systems where the state transition is a linear process and the noises are Gaussian distributed [75]. To deal with this issue, the extended Kalman filter (EKF) and unscented Kalman filter (UKF) are developed to solve the non-linear problems by linearizing the non-linear system functions [16]. However, this linearization also has limited approximation and cannot handle multi-model distributions due to the assumption of Gaussian parametric form of posterior [75].

Fortunately, the alternative stream of filtering is introduced to deal with the Gaussian assumption. This stream is the non-parametric filter, specifically the techniques of sequential-importance-sampling (SIS), or the particle filter (PF). The particle filter is a non-parametric technique with no functional form that instead uses a set of weighted samples (particles) to estimate the posterior distribution of the target state. There is no strong parametric assumption on the posterior density and thus a complex multi-model distribution can be represented. The two main steps in PF are prediction and correction, corresponding to system transition model and observation model, respectively. The prediction step increases uncertainty and the correction step decreases uncertainty. There is intensive research in PF that focuses on how to develop more advanced system transition models and how to refine observation data. The standard PF, also known as the Conditional Density Propagation [76], used system dynamic model with visual observations to propagate the sets of target state over time. As the pioneer research of particle filter, the drawback of this approach is that it is not robust in strongly cluttered environments since the proposal distribution is solely determined by the system dynamic model and inevitably neglects the most recent observations, resulting in many wasted particles in low likelihood areas. To deal with this issue, the authors in [77] constructed a new proposal distribution based on the state partitioning (SP) technique and a bank of parallel extended Kalman filters (PEKF). In details, all the samples of the state were partitioned into two sets, the nominal part and residua part, respectively. After partitioning, these samples were propagated through a bank of parallel filters and the estimated state was then calculated as the weighted sum of these samples. The authors also adopted the multi-model technique to deal with the target that has complex motion

patterns. In [75], the unscented particle filter (UPF) was presented, where the unscented Kalman filter was used to generate sophisticated proposal distributions that seamlessly integrated the current observations. Specifically, the unscented transformation was used to approximate the mean and covariance of the state after nonlinear transformation that not only outperformed the extended KF but also was computationally efficient since the Jacobian or Hessians was not required. The authors in [78] developed a robot-based tracking system using particle filter with the color visual cue and sonar cues. In order to deal with the sample impoverishment, the particle re-sampling process was proposed to draw samples in the filter from the neighborhoods of previous samples and the state space model was augmented with acceleration variables. In [79], by considering the target motion, the authors proposed a motion-based proposal density, where the Adaptive Block Matching (ABM) was used to calculate the estimated motion vector and to deduce the motion-based proposal distribution. The superiority was proved by using the Kullback-Leibler (KL) performance measure. The experimental results showed that this approach is able to handle sudden and unexpected motions.

2.4 Scheme of Context Reasoning Mechanism

2.4.1 Hybrid Model of Context Reasoning

As reviewed in the last sections, different types of context reasoning techniques have their own feasibilities for different applications, as well as advantages and weaknesses. In the current development of context-aware systems, the systems are required to be designed in a way that multiple objectives should be handled at the same time. For

example, a location-aware system should not only be able to detect and track the user's current position in real time, but also provide necessary services based on the integration of other related contexts, e.g. displaying the near point of interests based on the user's preference, navigating the fast route based on the user's destination and the current traffic conditions. Moreover, the sources of context or sensing data are rarely unique and simple, but usually heterogeneous. Even for a single sensed element, there are normally multiple sources with different qualities, thus introducing the issue of fusing inconsistent and non-uniform data. Both of the above issues need to be addressed. Therefore, the context reasoning based on only one single reasoning technique is often, if not always, far from enough to satisfy the requirements of applications. At this point of view, it is necessary to design and implement different reasoning techniques according to the specific scenarios and the corresponding requirements. As a solution, the hybrid model of reasoning method is designed and adopted by many state-of-art context-aware systems. In [80], in addition to comparing and contrasting the performance of different reasoning schemes, the author presented a conceptual architecture of hybrid context reasoning and identified the suitable scheme to be employed by each component of the architecture. This architecture is depicted in Fig. 2.5.

This schematic architecture of dynamic situation reasoning can be divided into three steps. The first step involves the acquisition of atomic or primitive environmental contexts by directly deployed sensors, and the brief pre-processing on these contexts for further use, such as de-noising procedures, primitive abstraction, etc. For example, the specific readings from thermometers can be abstracted into different meaningful states: cold, lukewarm, warm, or hot, by the adoption of fuzzy logic.

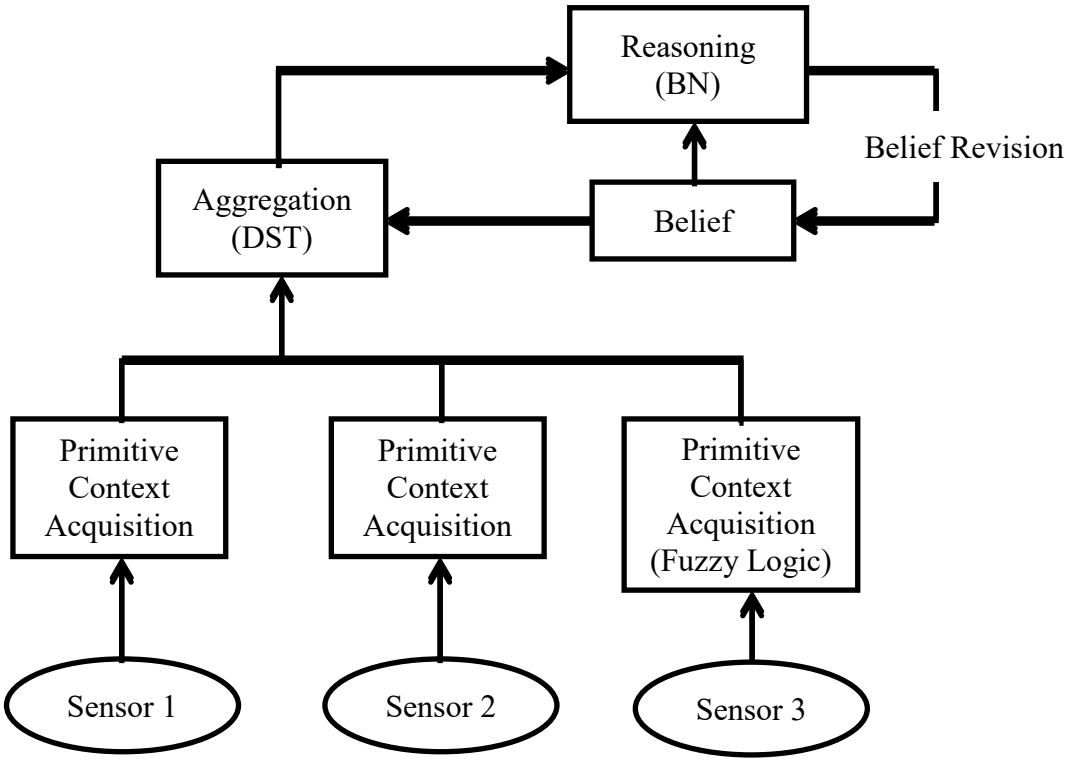


Figure 2.5: The conceptual architecture of hybrid context reasoning

Then these processed data will be the input of the second step, where multiple contexts towards the same entity but from heterogeneous sources are aggregated to resolve any possible conflicting observations. Taking the temperature example again, it is probable that in the same room (entity) the readings from different thermometers are inconsistent, depending on their different resolutions and accuracies, as well as their positions in the room. Therefore, an aggregation technique is required to appropriately deal with this issue. In Fig. 2.5, the Dempster-Shafer Theory of evidence is shown as the tool for combining different evidences.

The third and final step of context computing is the implementation of high-level inferring techniques, including Bayesian Networks (BN), Hidden Markov Model (HMM),

etc., to reason about the ultimate user required context. For instance, after acquiring all the necessary contexts (indoor temperature as in the previous example, as well as outdoor temperature, number of people in the room, time of the day, etc.) with satisfactory quality, the air-conditioning system can make the decision of "increasing the indoor temperature", "decreasing the indoor temperature", or "no actions". All the ultimate reasoned contexts are stored and reflected in the system as beliefs, which in turn guides the processes of aggregation and reasoning.

2.4.2 Probabilistic Robotic Context Reasoning

It can be seen that the author in [80] is mostly concerned with the role of various probabilistic schemes to reason about the uncertainties in corresponding contexts. In fact, uncertainty is also the first and foremost topic in robotics, which is resulted from the noise and error in the environment, sensors, actuators, as well as robot's software and algorithmic approximation. The effects of uncertainty on robot performance are also related to the application domain. In some well-modeled environment such as the product assembly pipeline, the system can be well engineered so that uncertainty only plays a minor role. In contrast, in the cases of complex and hard-to-model environments such as residential area or unknown space exploration, the uncertainty is so substantial that it is one of the most important factors in the design of a capable robotic system. Being assistance-oriented, the context-aware robotic system in this thesis falls into the second category and consequently, the process of uncertainty is the major issue.

As a promising approach, probabilistic robotics is a relatively emerging method that deals with uncertainties in both robot perception and action. The key idea is the

probability theory. Instead of solely relying on the "best guess" of the acquired contexts, the probabilistic algorithms adopt the probability distributions to cover the whole space of guesses. By representing the ambiguity and degree of belief in a mathematical way, the system is able to maintain a panoramic awareness of what the surrounding environment is and then to select the next optimal action in a robust manner according to different goals. These goals can be the ultimate goal to obtain a single required context, or the intermediate goal to simply reduce the system's uncertainty, etc. Compared with the traditional robotic techniques, which are model-based or reactive behavior-based, the probabilistic approach is able to better cope with sensor limitations and has a weaker requirement on the accuracy of robot models. Another benefit is that the probabilistic approach also deems it with no necessity to maintain the complete state assumption, or the Markov assumption, which is usually not practically preferred or tractable. With only the interested states modeled, all other unmodeled states will be considered to be with close-to-random effects, which will be digested by the probabilistic approaches.

The probabilistic robotics approach can be considered as a state estimated problem, which mainly consists of two stages, robot action and robot perception. Generally, being equipped with an internal state to reflect the surrounding environment, the robot action is corresponding to the internal state change resulted from the robot control and actuation that is applied to the environment. The stage of robot action will inevitably increase the uncertainty of the system. On the other hand, the stage of robot perception involves the acquisition of sensor data from the environment to correct the internal state, thus decreasing the uncertainty in the system by reducing the difference between the internal state and the real state of the environment.

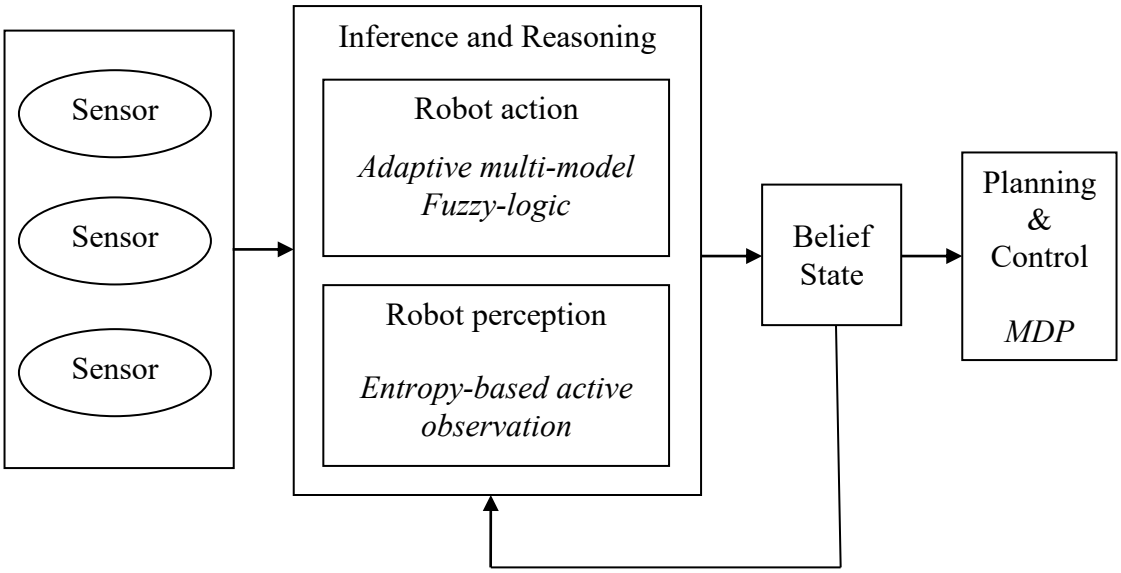


Figure 2.6: The conceptual architecture of hybrid context reasoning

As for the context-aware robotic system in this thesis, different reasoning techniques are developed and implemented at different stages of context processing in the robotic system. In the aspect of robot action, a revised system dynamics is adopted to closely track the change of system state, and in the aspect of robot perception, a newly designed observation method is proposed to actively minimize the uncertainties in the sensed data. The reasoned belief state is then used for the task of plan and control using an improved version of sequential decision-making based on Markov decision processing algorithm. This hybrid context reasoning scheme is illustrated in Fig. 2.6.

2.5 Summary and Conclusion

In this chapter, firstly the previous review work of the context reasoning techniques was briefly introduced. Secondly, a new taxonomy of context reasoning techniques was proposed based on the whether the approaches are probability-based or non-probability-based, along with an extensive literature review based on this new taxonomy. In the end, the hybrid scheme of context reasoning was presented, and its implementation on the probabilistic robotics is illustrated.

Chapter 3

Prediction of System Dynamics in Bayesian Filtering

3.1 Introduction

The contexts for the robotic systems have multiple types, including visual, acoustic, haptic, location contexts, as well as other human-related types, e.g. body signals, and some basic ones including time, temperature. Among all these, the visual and location contexts are the basic while important ones due to the fact that, after obtaining these information of the target object, the relationship between the robot itself and the environment, as well as their interactions, can be partially determined. Another important reason of the choice of visual and location contexts is because of the context-aware robotic system requirements specified in Chapter 2. The visual and location contexts are closely related to the conditions of the users. Focusing on these contexts, the system is able to effectively and efficiently provide more advanced user related services and assistance. Moreover, although the visual and location contexts of the users is not

significant for displaying, yet they are the prerequisites of executing physical tasks, i.e. adapting system behaviors or changing contexts. In general, the visual and location awareness has been widely utilized in the applications on the purpose of safety surveillance [81], human machine interaction [82], healthcare monitoring, defense fields [4], etc. However, despite of the extensive research and wide implementations, there are still many problems that need to be addressed. This is resulted from that fact that, besides the traditional tracking and localization challenges (e.g. sensor data noises, stochastic control effects, interfering signals), the target object often shows complicated motion types, data deformations and occlusions. Furthermore, the environment is likely to be complex, containing background clutters.

Generally, the process of robotic tracking and localization can be formulated as that, given a certain representation or priori of the surrounding environment and the ability of sensing, the robot itself should maintain a constant and ongoing awareness about the target and environment, as well as of itself, in order to perform necessary future tasks and to interact with related objects or users. Here, the key issues to the successful robotic tracking and localization are the correct priori and sensor observations. There are in general two primary approaches to tracking and localization problems [12]. One is known as the deterministic method based on target representation and localization techniques. The other is known as the stochastic method based on filtering and data association technique [13], which is usually referred to as filtering technique. Both of these two approaches have been widely utilized in practical applications.

3.1.1 Deterministic Target Representation

Theoretically, with all the correct and sufficient information, the robot can accurately maintain tracking and localization in real time. For example, a simple and traditional method uses the angle and distance measurement based on the standard geometry. Specifically, for the deterministic target representation technique, a cost function, whose variables are target motion, is often designed based on the target features extracted from the information of visual data [13]. With proper calculation of target motion parameters, the cost function will converge correspondingly to an ideal value. Therefore, in the target representation and localization technique, the tracking problem can be formulated as an optimization process. The objective is how to calculate the target state in order to minimize the value of the cost function. As a result, mathematical optimization techniques are usually required. The authors in [14] used the classical gradient descent method, in which, a flow deformed the initial curve towards the minimal value of the designed objective function to detect and track multiple moving objects in image or video sequences. In [75], the mean shift algorithm, which was able to find the highest value of the visual density through utilizing the mean shift vector obtained from the density gradient, was used to track moving object with radical color changes. Besides, in [16], a robust human head tracking system was developed. In this system, the human head was modeled as an ellipse and two tracking modules. One module was concentrating on the intensity gradient around the ellipse perimeter while the other focusing on the color histogram of the ellipse interior.

However, in many cases, the robot operates in a partially known or unknown environment thus no or only little prior information about the environment is available.

Even worse, the sensing information is inevitably harsh for one to work with due to the nature of signals, inference, and noise [83], making the tracking and localization extremely difficult to be performed in a deterministic way.

3.1.2 Filtering and Data Association

Due to the reason mentioned in the above section, the robotic tracking and localization is usually considered as a problem of probabilistic inference where the filtering and data association techniques are applied. It is also known as the so-called Markov estimation, since the state dynamics in the system is usually assumed to be Markov in order to make the computation tractable. The main difference between the probabilistic inference and deterministic inference is that the probabilistic inference provides the concept of degree of belief, a numerical value between 0 and 1 assigned to each of the estimations, instead of either 0 or 1 as in the deterministic inference.

The filtering and data association technique deals with the dynamics of the tracked and localized target. So the state space model is used. Meanwhile, the model of system state measurement is also considered. Therefore, the tracking problem can be formulated as a state estimation problem based on the system state transition probability and observation probability. This estimation problem is usually categorized as a dynamic Bayesian network (DBN) problem to represent the temporal probability model of the system. To be specific, the stochastics of the system can be considered as a hidden Markov model (HMM), where the state of the process is described by a single discrete random variable. The basic temporal evolution of the system is shown in Fig. 3.1, in which x_t represents the system state at time index t, z_t the observation of the system state

and u_t the system control. Generally, the system state transition is governed by the conditional probability $p(x_t|x_{t-1}, u_t)$, which specifies how system state evolves over time as a function of previous state and control (or without control) as illustrated by the dotted circle in Fig. 3.1, and the system measurement is governed by $p(z_t|x_t)$ which specifies how the observation z_t is generated from the current system state x_t.

As reviewed in Chapter 2, the traditional solutions here include the Kalman filter, the extended Kalman filter (EKF), the unscented Kalman filter (UKF), whose performances rely on either Gaussian assumption or simple model distributions of system dynamics. The good alternative is the non-parametric filter, specifically the techniques of sequential-importance-sampling (SIS), or the particle filter (PF) which is a non-parametric technique with no functional form that instead uses a set of weighted samples (particles) to estimate the posterior distribution of the target state. The particle filtering has been widely implemented and also been proved to have good performance when dealing with the complex global and dynamic tracking and localization [75][76][77][78][79].

As shown in Fig. 3.1, there are mainly two main steps in the state estimation problem. The system transition model, or the system dynamics, is researched in this chapter, while the observation model, or the system measurement, will be investigated in Chapter 4.

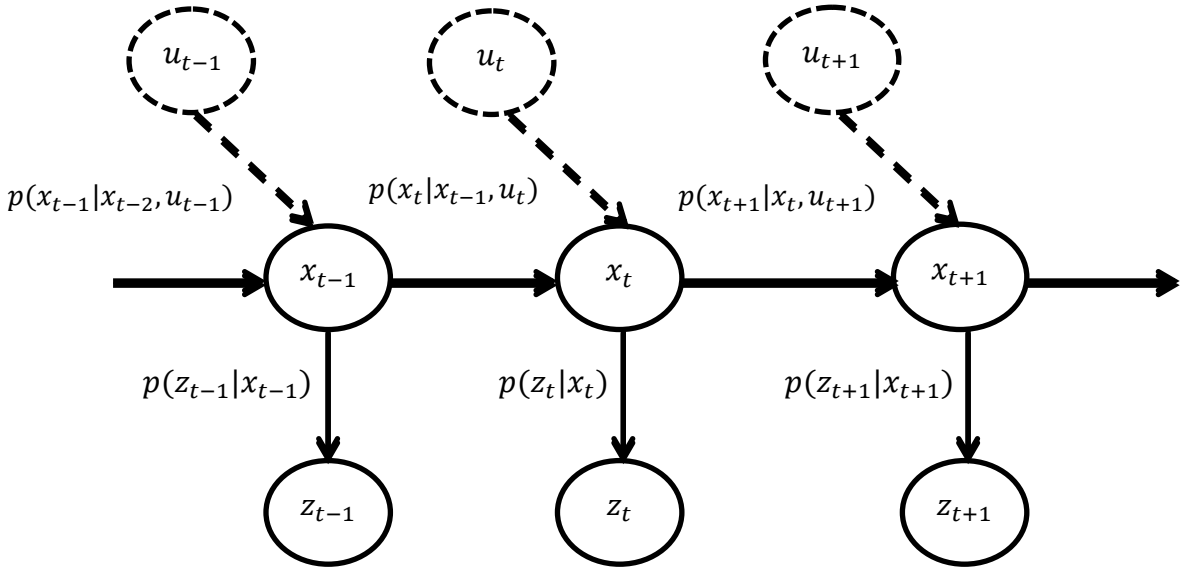


Figure **3.1: The temporal evolution of system states, observations and controls**

3.2 Principle of Particle Filtering

The basic idea of particle filters is to represent the posterior distribution PDF $p(x_t|z_{1:t},u_{1:t})$ of the target state by a set of weighted-samples or particles. Without loss of generality, consider the following state space model:

$$x_t = f_t(x_{t-1},u_t,\varepsilon_t) \tag{3.1}$$

$$z_t = h_t(x_t,\gamma_t) \tag{3.2}$$

where, x_t, z_t and u_t represent the target state, observation and control at time index t, respectively. The arbitrary functions of $f_t(\cdot)$ and $h_t(\cdot)$ are the system evolution and

observation, which can be nonlinear and non-Gaussian. At last, ε_t and γ_t are processing and observation noise.

The primary objective is to construct the PDF $p(x_t|z_{1:t}, u_{1:t})$ under the rule of Chapman–Kolmogorov equation and Bayes formula:

$$p(x_t|z_{1:t-1}, u_{1:t}) = \int p(x_t|x_{t-1}, u_t)p(x_{t-1}|z_{1:t-1}, u_{1:t-1})dx_{t-1} \tag{3.3}$$

$$p(x_t|z_{1:t}, u_{1:t}) = \frac{p(z_t|x_t)p(x_t|z_{1:t-1}, u_{1:t})}{\int p(z_t|x_t)p(x_t|z_{1:t-1}, u_{1:t})dx_t} = \eta p(z_t|x_t)p(x_t|z_{1:t-1}, u_{1:t}) \tag{3.4}$$

where $p(x_{t-1}|z_{1:t-1}, u_{1:t-1})$ is the state at time index $t-1$, $p(x_t|x_{t-1}, u_t)$ the system transition model, $p(x_t|z_{1:t-1}, u_{1:t})$ the priori at time index t, $p(z_t|x_t)$ the likelihood function of observation, $p(x_t|z_{1:t}, u_{1:t})$ the desired posterior density at time index t, and η the normalization factor.

To be more generic, the PDF is usually denoted by the belief over the state variable as $bel(x_t) = p(x_t|z_{1:t}, u_{1:t})$. The probability distribution before incorporating the measurement z_t and after system transition $p(x_t|x_{t-1}, u_t)$, is denoted by the priori belief $\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t})$. Therefore, (3.1) is the priori probability distribution representing the prediction of the system state, whereas (3.2) represents the required posterior probability distribution after measurement update or correction by incorporating system observation.

Theoretically, the particles maintained by the particle filter represent the discrete estimation of the belief, or system state. However, in some cases, the continuous belief representation is required. In other words, the points in the belief state that are not

represented by the particles need to be estimated as well. This issue is also known as the problem of density estimation. Approaches include K-MEANS algorithm, density tree, kernel density estimation, etc. One common method is to approximate the PDF by the sum of weighted Dirac functions as follows:

$$p(x_t|z_{1:t}, u_{1:t}) \approx \sum_{i=1}^{m} w_t^i \delta(x_t - x_t^i) \tag{3.5}$$

where $\delta(\cdot)$ is Dirac function, $x_t^i, i = 1, \dots, m$ is set of random particles generated from a certain distribution, and each particle with a particular weight $w_t^i, i = 1, \dots, m$, satisfying:

$$\sum_{i=1}^{m} w_t^i = 1 \tag{3.6}$$

$$w_t^i \propto p(z_t|x_t^i), \qquad i = 1, \dots, m. \tag{3.7}$$

However, the continuous density estimation requires more powerful computational capability, which is usually unavailable in many robotic applications. Therefore, in this thesis, the discrete belief representation is preferred and the continuous density estimation is mainly used for simulation and verification when necessary.

3.3 Adaptive Multi-Model based Proactive Particle Sampling

Traditionally, the standard PF assumes that the system dynamics is governed by a single transition model. Although feasible, it is unable to tackle the scenarios where system switches between different types of operation, e.g. in tracking and localizing the target object with complex motions, thus making the estimation inaccurate and unstable [84]. The multi-model technique is introduced in [85] as a solution to this issue, and the multi-

model particle filter (MMPF) is developed in [77][86] to deal with the object tracking with complex motion. Generally, the MMPF technique adopts multiple models to represent the system dynamics and the selection of each model is governed by the model transition matrix. However, in many of the literatures, the probability value in the model transition matrix is usually constant and is set empirically. Therefore, although being able to handle pure complex and random target motion, this solution is not optimal when the target presents some comparatively stable motions for a certain period of time amongst randomness, or the target changes the motion patterns. This issue is resulted from the fact that the model transition matrix is designed as fixed and cannot be modified even when the target motion changes. In order to deal with this issue, a novel multi-model technique is proposed in this section to adaptively select the suitable transition model to proactively sample the particles, according to the potential of the target motion.

3.3.1 Multi-Model System Dynamics

The principle of multi-model technique is to assume the system operates according to one model from a pre-defined finite set of hypothetical models, which is also known as regimes. According to (3.1), the system transition process with multiple models is given as

$$x_t = f_t^{c_t}(x_{t-1}, u_t) + \Gamma \cdot \varepsilon_t \tag{3.8}$$

where c_t is the index of the possible transition models governed by the following model transition matrix

$$\Pi = \begin{bmatrix} \pi_{11} & \cdots & \pi_{1N} \\ \vdots & \ddots & \vdots \\ \pi_{N1} & \cdots & \pi_{NN} \end{bmatrix} \tag{3.9}$$

in which $\pi_{ij} = p(c_{t-1} = i|c_t = j)$ is the probability of model transition from regime i to j, under the condition that $\sum_{j=1}^{N} \pi_{ij} = 1$, and N is the total number of possible regimes.

To the best knowledge, the most common hypothetical models adopted in the current literatures with the absence of the control factor u_t are the constant velocity model, clockwise/counter-clockwise turn model, and constant acceleration model. These models correspond to the target with complex and unpredictable motion, as well as robot that is in a state of random walking or wandering.

1. Constant velocity transition model ($c_t = 1$)

$$\begin{bmatrix} r_t \\ s_t \\ \dot{r}_t \\ \dot{s}_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{t-1} \\ s_{t-1} \\ \dot{r}_{t-1} \\ \dot{s}_{t-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \varepsilon_t \tag{3.10}$$

where the horizontal and vertical position r_t and s_t, as well as the velocity $\dot{r}_t$ and $\dot{s}_t$, form the state vector x_t, ε_t is the 2-by-1 random noise, each component of which is conforming to the zero-mean normal distribution, to represent the uncertainty in system state transition, and T is the sampling interval.

2. Clockwise/counter-clockwise turn transition model ($c_t = 2,3$)

$$\begin{bmatrix} r_t \\ s_t \\ \dot{r}_t \\ \dot{s}_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{\sin\Omega_t^{c_t} T}{\Omega_t^{c_t}} & \frac{\cos\Omega_t^{c_t} T - 1}{\Omega_t^{c_t}} \\ 0 & 1 & \frac{1 - \cos\Omega_t^{c_t} T}{\Omega_t^{c_t}} & \frac{\sin\Omega_t^{c_t} T}{\Omega_t^{c_t}} \\ 0 & 0 & \cos\Omega_t^{c_t} T & -\sin\Omega_t^{c_t} T \\ 0 & 0 & \sin\Omega_t^{c_t} T & \cos\Omega_t^{c_t} T \end{bmatrix} \begin{bmatrix} r_{t-1} \\ s_{t-1} \\ \dot{r}_{t-1} \\ \dot{s}_{t-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \varepsilon_t \tag{3.11}$$

where $\Omega_t^2 = -a/\sqrt{\dot{r}_t{}^2 + \dot{s}_t{}^2}$ and $\Omega_t^3 = a/\sqrt{\dot{r}_t{}^2 + \dot{s}_t{}^2}$ denotes the turning rate for clockwise and counter-clockwise turn, respectively, and a is the typical maneuver acceleration of target. Since the turning rate is time variant and is involved in the function of target velocity, the clockwise/counter-clockwise turn transition model is highly non-linear.

The above two models both assume that the horizontal and vertical velocities are constant. In order to consider the scenario where velocities change, the following model is considered.

3. Constant acceleration transition model ($c_t = 4$)

$$\begin{bmatrix} r_t \\ s_t \\ \dot{r}_t \\ \dot{s}_t \\ \ddot{r}_t \\ \ddot{s}_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 & T^2/2 & 0 \\ 0 & 1 & 0 & T & 0 & T^2/2 \\ 0 & 0 & 1 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{t-1} \\ s_{t-1} \\ \dot{r}_{t-1} \\ \dot{s}_{t-1} \\ \ddot{r}_{t-1} \\ \ddot{s}_{t-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \varepsilon_t \qquad (3.12)$$

where $\ddot{r}_{t-1}$ and $\ddot{s}_{t-1}$ represent the horizontal and vertical acceleration of the target motion.

4. Velocity motion transition model

The above three motion models correspond to the random target motion with no presence of the control u_t. Nevertheless, it is necessary to consider the scenario where the robot motion is under control, e.g. following a pre-defined path or executing explicit motion commands for some specific tasks. Generally, since many commercial robots can be controlled through two variables, the translational velocity v and the rotational velocity ω, the control u_t can be incorporated into (3.1) by these two variables to

formulate the control transition model.. Therefore, the control u_t at time index t is usually denoted as

$$u_t = \begin{pmatrix} v_t \\ \omega_t \end{pmatrix} \tag{3.14}$$

This model is referred to as the velocity motion transition model defined in (3.15), corresponding to the robot motion under a specific controlling sequence, e.g. to move forward at 1*m/s* for 5*s*, to rotate at 0.1π *rad/s* for 5*s* (turning right), and then to move forward at 2*m/s* for 10*s*.

$$\begin{bmatrix} r_t \\ s_t \end{bmatrix} = \begin{bmatrix} r_{t-1} - \frac{\hat{v}_t}{\hat{\omega}_t}\sin\theta + \frac{\hat{v}_t}{\hat{\omega}_t}\sin(\theta + \hat{\omega}_t T) \\ s_{t-1} + \frac{\hat{v}_t}{\hat{\omega}_t}\cos\theta - \frac{\hat{v}_t}{\hat{\omega}_t}\cos(\theta + \hat{\omega}_t T) \end{bmatrix} \tag{3.15}$$

where $\begin{bmatrix} \hat{v}_t \\ \hat{\omega}_t \end{bmatrix} = \begin{bmatrix} v_t \\ \omega_t \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \varepsilon_t$ in order to incorporate the noise in control, and θ is the robot orientation, which is illustrated in a global coordinate system in Fig. 3.2, along with the robot position at (*x,y*).

All these models can be applied in the particle filter for the step of system state estimation. Fig. 3.3 to Fig. 3.5 illustrates the sampling from the models without control in the *x*-*y* plane. For the constant velocity and acceleration model, the initial position, horizontal and vertical velocities and acceleration are set as (0,0), 1*m/s*, and 1m/s^2, respectively. For the counter-clockwise mode, the initial position and maneuver acceleration is (10,10) and -2m/s^2, respectively. For all the models, the sampling interval *T* is 0.5*s*. In each of the sampling, 300 particles are deployed, represented by the dots in the figure, and the arrow denotes the path in an ideal noise-free environment. Fig. 3.6 illustrates the sampling from the velocity motion model. The robot is executing a control

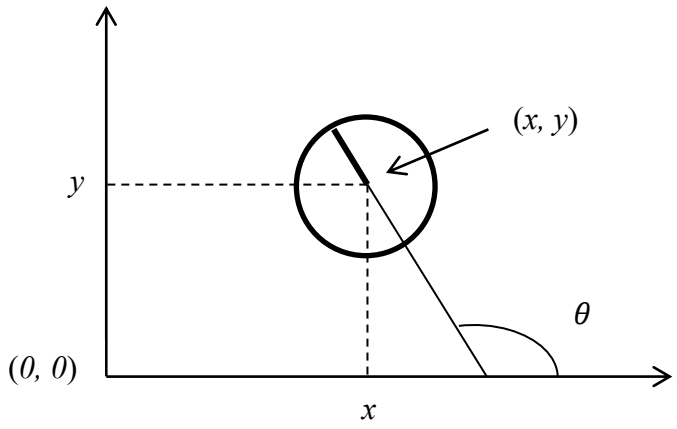


Figure **3.2: Robot pose in a global coordinate system**

sequence of three phases:

$$u_t = \begin{cases} \binom{1}{0} & 0s < t * T < 1s \\ \binom{1}{2} & 1s < t * T \leq 2s \\ \binom{1}{0} & 2s < t * T \leq 3s \end{cases} \tag{3.16}$$

where the solid line again represents the noise-free path, the robot initial position is at (3,3), the sampling interval T is $0.1s$, and 300 particles are deployed. It can be clearly shown from all these figures that, the uncertainty in the system is increasing along with the state evolution, represented by the large dispersion of the samples in the space area.

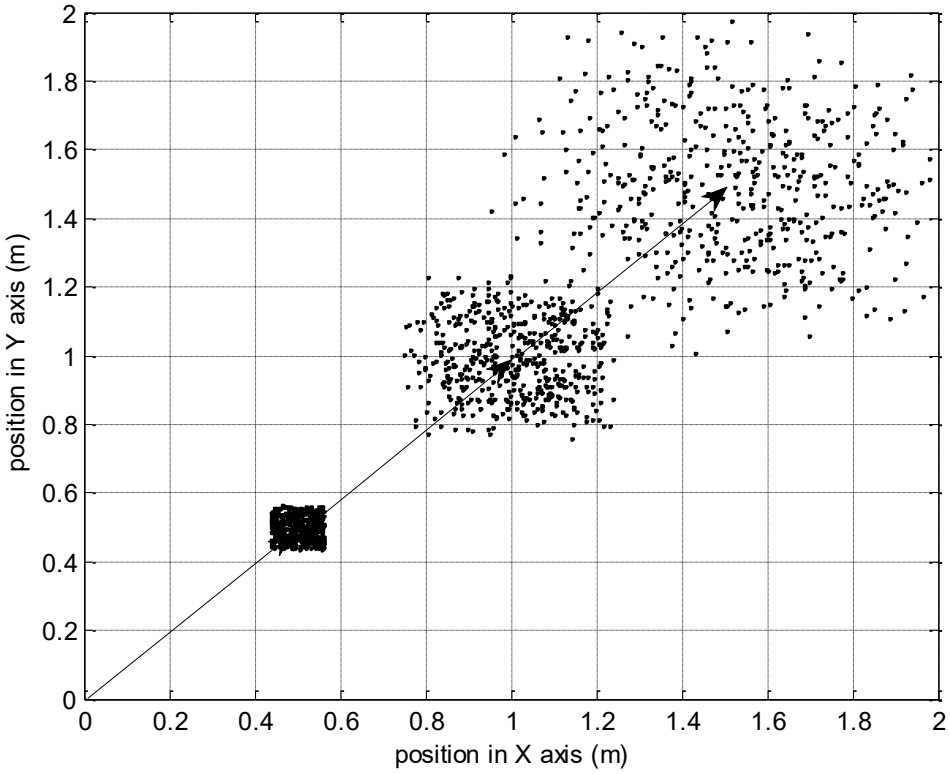


Figure 3.3: Particle sampling from the constant velocity transition model

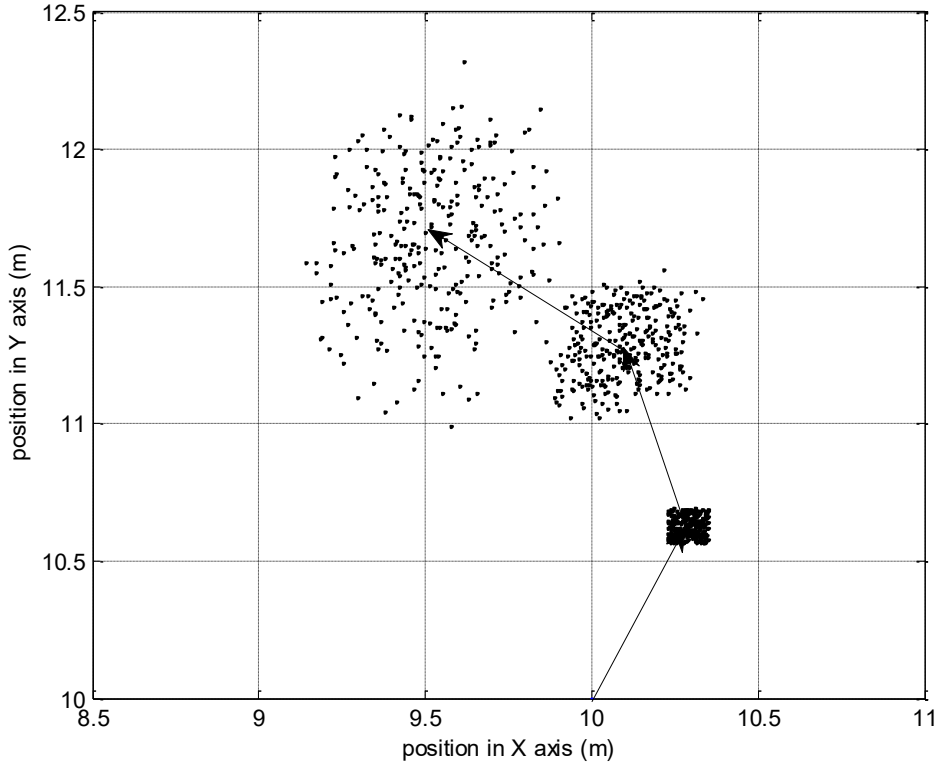


Figure 3.4: Particle sampling from the counter-clockwise turn transition model

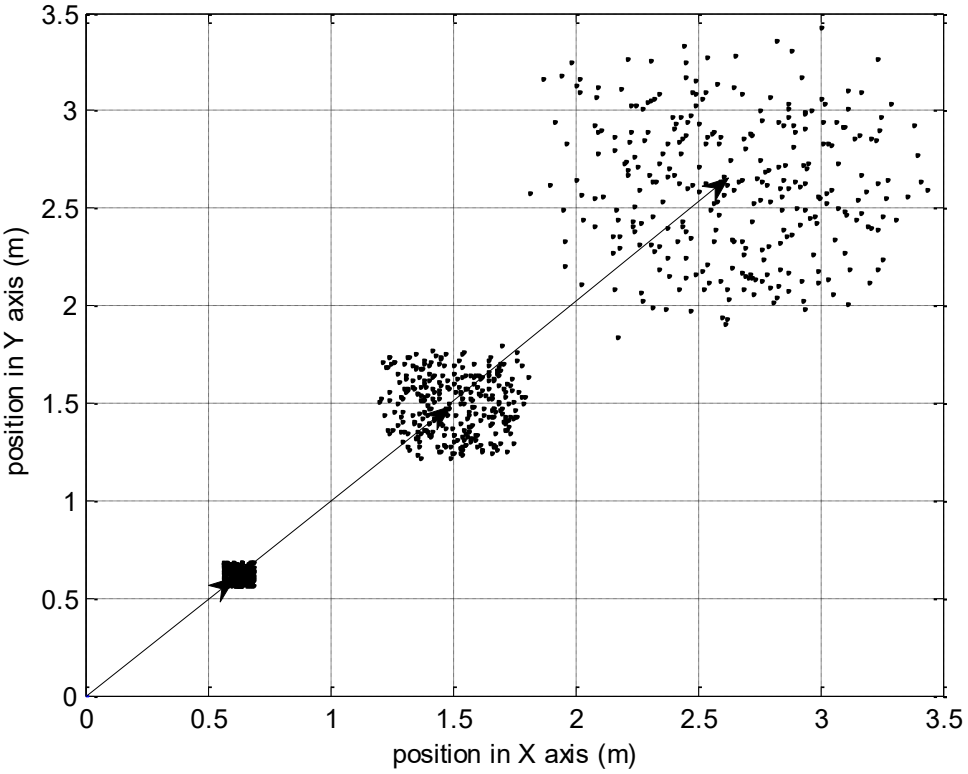


Figure 3.5: Particle sampling from the constant acceleration transition model

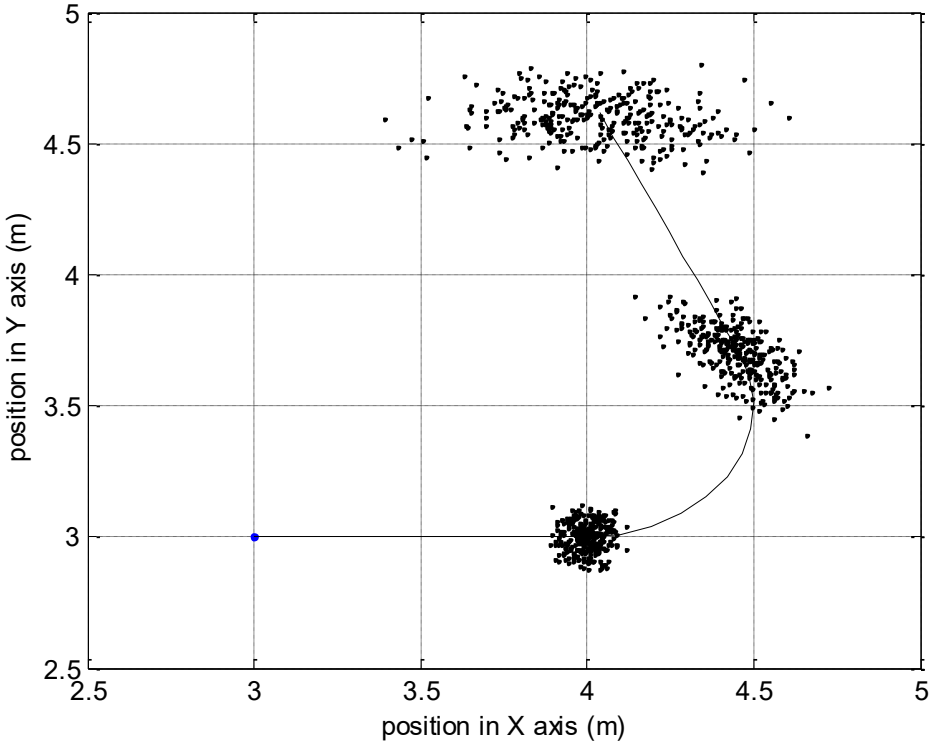


Figure 3.6: Particle sampling from the velocity motion transition model

3.3.2 Proactive Particle Sampling

The technique of multi-model is able to deal with complex target motions. The key factor is the model transition matrix Π defined in (3.9), which governs how different models are selected in each time index. Normally, the model transition matrix is usually set according to the following rules:

1) the probability of maintaining the current model (the value of the diagonal elements) is larger than switching to other models,
2) the probability of switching to constant velocity model otherwise (the value of the first column) is larger,
3) the probability of switching to other models otherwise are then equal.

At the beginning of each cycle of the particle filtering, each particle is sampled by one of the four hypothetical models. The first sampling process operates according to the initial probability $p_0 = [\pi_1\ \pi_2\ \pi_3\ \pi_4]$, whereas in the following sampling process, the model switching is governed by the model transition matrix.

As argued before, in many literatures, the model transition matrix is usually constant, which is unable to optimally deal with target with changing motion patterns. In this thesis, a novel model transition matrix is proposed, in which the probability of model transition can be adaptively modified according the current target motion pattern. Specifically, an adaptive vector $\Phi_t = [\phi_1^t\ \phi_2^t\ \phi_3^t\ \phi_4^t]$ is introduced, where ϕ_i^t represents the sample's current potential of switching to model i at next time index. The value of Φ_t is set according to Gaussian distribution centered at ϕ_i^t, and ϕ_i^t is selected according to the following process. After each observation is incorporated, the weights of some

particles are increased. With all the particles generated by the same model i, the adaptability of model i is calculated as the proportion of the particles with increased weights to all the particles generated by this model. And the model i with highest adaptability (largest proportion) is selected as ϕ_i^t, the center of the Gaussian distribution. Correspondingly, the model transition matrix Π is adaptively updated at each filtering cycle as

$$\Pi_t' = \begin{bmatrix} \pi_{11}^{t-1} & \cdots & \pi_{14}^{t-1} \\ \vdots & \ddots & \vdots \\ \pi_{41}^{t-1} & \cdots & \pi_{44}^{t-1} \end{bmatrix} \begin{bmatrix} \phi_1^{t-1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \phi_4^{t-1} \end{bmatrix} \tag{3.17}$$

and each row is normalized to satisfy that $\sum_{j=1}^{4} \pi_{ij}^t = 1$ to obtain Π_t, which is used in the next cycle of particle filtering.

3.4 Fuzzy Logic-based Estimated Velocity Difference

In the standard particle filter, all the system transition models are used to propagate the state vector from time index $t-1$ to t. However, one drawback can be seen from the system dynamic equation (3.1) that the estimation of object state are all based on time step up to $t-1$, thus neglecting the information in the current state at t. The drawback is theoretically inevitable in the standard particle filter due to the fact that the system transition model $p(x_t|x_{t-1}, u_t)$ is often used as the proposal distribution to predict $\overline{bel}(x_t)$ from $bel(x_{t-1})$, and observation is incorporated into the system belief only after this state prediction. Consequently, the standard particle filter is unable to take into account the most recently observed state, making the system prone to loss of tracking and

localization of object with unpredictable motions. To be specific, the constant velocity transition model and the clockwise/counter-clockwise turn transition model both assume that the velocity of the target remains stable with only some minor errors. Although the constant acceleration transition model takes velocity change into consideration, it is unable to handle the change of target direction, e.g. a target moving forward might abruptly turn to other directions.

To deal with this issue, many research of the particle filter has been conducted on how to design a better proposal distribution to account for the most recent observation [77][79][87][88]. In [89], an approach of target velocity estimation was proposed using the first-order differentiation value to update the proposal distribution of the particle filter. The basic idea is that, the abruptness of the target motion change can be alleviated by estimating the current state of position or velocity before incorporating the observation data. Specifically, the target current velocity can be derived from the previous position and velocity. This is also known as the velocity estimation problem, which exists in many control-related applications [90], including motion control, manipulator control. In general, the velocity estimation method in [89] was based on the finite difference method with window size of 1. Although simple and fast, this method only incorporates samples in two time indexes. It can be easily verified that this method is more precise with the window size growing, which will produce a smaller variance of the velocity estimation but unfortunately will also introduce delay and reduce reliability [91]. A promising alternative solution is thus to adjust the window size adaptively. Based on this idea, a novel velocity estimation approach is proposed in this section.

3.4.1 Finite Difference Method of Euler Approximation

In general, velocity estimation algorithms can be classified into two classes: fixed-time and fixed-position, with respect to time or space that is used to estimate the velocity, respectively [92]. A typical velocity estimation method adopts the finite difference method (FDM) given in (3.18), taking the horizontal velocity $\dot{r}_t$ as an example, without loss of generality.

$$\widehat{\dot{r}_t} = \frac{r_t - r_{t-n}}{nT} \tag{3.18}$$

where $\widehat{\dot{r}_t}$ is the estimated velocity, r_t is the horizontal position at time index t, T is the sampling time, and n is the window size.

3.4.2 Fixed Window Size versus Varied Window Size

The simple and easy-to-implement method uses the window size $n = 1$ [89]. However, as argued above, there is a trade-off between a large and a small window size. Specifically, a small window size is with good real time response and is able to capture the most recent change in velocity, but is inevitably more vulnerable to noise and error. On the other hand, a large window size is more robust to random fluctuations, but it also decreases the sampling rate, thus having a delay-response and unable to reflect the most recent change in velocity, as well as consuming more computational resources which is often not well available on robotic platform. Therefore, setting the windows size as a constant for all cases is usually not an optimal solution. The alternative solution here is to adaptively change the window size according to different conditions.

The principle of adaptive windowing is to change the size of the windows according to actual motions of the target. Specifically, if the target is moving with high velocity, the windows size should be small to increase the sampling rate in order that any fast changes in velocity can be promptly captured. On the other hand, if the target is moving slowly, the windows size can be large to decrease the sampling rate so the effect of noise can be minimized. In [91], the velocity estimation approach was proposed based on the criterion whether the samples inside the windows size was covered within an uncertainty band, which was pre-defined by the noise in velocities. Although the effectiveness of this approach has been proved, it requires an additional step of auxiliary smoothing to be more robust.

In this section, the fuzzy logic is adopted [93] for the velocity estimation to update the proposal distribution of the particle filter, because it is well suitable for dealing with approximation rather than accuracy, and has no requirement of system parameters. Specifically, the fuzzification is the process of changing a real scalar value into a fuzzy value. This is achieved with the different types of fuzzifiers, or membership functions, which are used to associate a grade to each fuzzy value. The simplest membership functions are formed using straight lines, for example, the triangular membership function which is a collection of three points forming a triangle, and the trapezoidal membership function which has a flat top and is a truncated triangle curve. These straight-line membership functions have the advantage of simplicity. Two other popular types of membership functions are the generalized bell membership function and Gaussian membership function, due to their smoothness and being nonzero at all points. The Gaussian membership function is built on the Gaussian distribution curve. The bell

membership function has one more parameter than the Gaussian membership function, so it can approach a non-fuzzy set if the free parameter is tuned. Because of their smoothness and concise notation, Gaussian and bell membership functions are popular methods for specifying fuzzy sets. However, although the Gaussian membership functions and bell membership functions can achieve smoothness, they are unable to specify asymmetric membership functions, which is important in certain applications. The sigmoidal membership function, which is either open left or right, is an asymmetric and closed membership function that can be synthesized using two sigmoidal functions. Besides, the polynomial curves based membership functions include the Z, S, and Pi curves, all named because of their shape. The Z based function is the asymmetrical polynomial curve open to the left, S based is the mirror-image function that opens to the right, and Pi based is zero on both extremes with a rise in the middle.

In this section, the triangular function is adopted due to its simple form and high efficiency of implementation. Another reason is that it can well adapt to the range of the target velocity researched in this thesis.

3.4.3 Adaptive Windowing with Fuzzy Logic

The target horizontal velocity $\dot{r}_t$ at time index t is estimated according to (3.18). In order to detect the high velocity and low velocity, the variable of the difference of the target position between $t-1$ and $t-2$, namely Δr_{t-1}, is introduced. Meanwhile, in order to detect the potential change of the velocity, another variable of the difference of the velocity between $t-1$ and $t-2$, namely $\Delta \dot{r}_{t-1}$, is used. These two variables are defined

as below in (3.19). The adaptive setting of the window size n depends on these two values of Δr_{t-1} and $\Delta \dot{r}_{t-1}$.

$$\begin{aligned} \Delta r_{t-1} &= r_{t-1} - r_{t-2} \\ \Delta \dot{r}_{t-1} &= \dot{r}_{t-1} - \dot{r}_{t-2} \end{aligned} \tag{3.19}$$

A normal fuzzy logic is implemented for setting the windows size n, and the Δr_{t-1} and $\Delta \dot{r}_{t-1}$ are selected as the two input of the fuzzy logic. The membership function μ of the fuzzy logic is shown in Fig. 3.7, which is designed as the triangular function in interval between 0 and 1. In this triangular membership function, there are 5 fuzzy values adopted, namely NL, NM, ZE, PM and PH, which means negative low, negative medium, zero, positive medium and positive high, respectively. The choice of 5 fuzzy values can reasonably represent the changing of the target velocity, while less fuzzy values are not able to and more fuzzy values will affect the efficiency and effectiveness of the velocity estimation. The details of the fuzzy rules are shown in Table 3.1, according to the following criteria.

1) If Δr_{t-1} and $\Delta \dot{r}_{t-1}$ are PH, the target velocity is high and with the potential of greatly increasing, thus the windows size should be small in order to promptly capture the change.
2) If Δr_{t-1} and $\Delta \dot{r}_{t-1}$ are NL, the target velocity is low and barely changes, thus the windows size should be large to minimize the effect of noise and error.
3) If Δr_{t-1} and $\Delta \dot{r}_{t-1}$ are ZE, the target velocity and its potential are both mild and comparatively stable, thus the window size should be medium to keep the balance between precision and reliability.

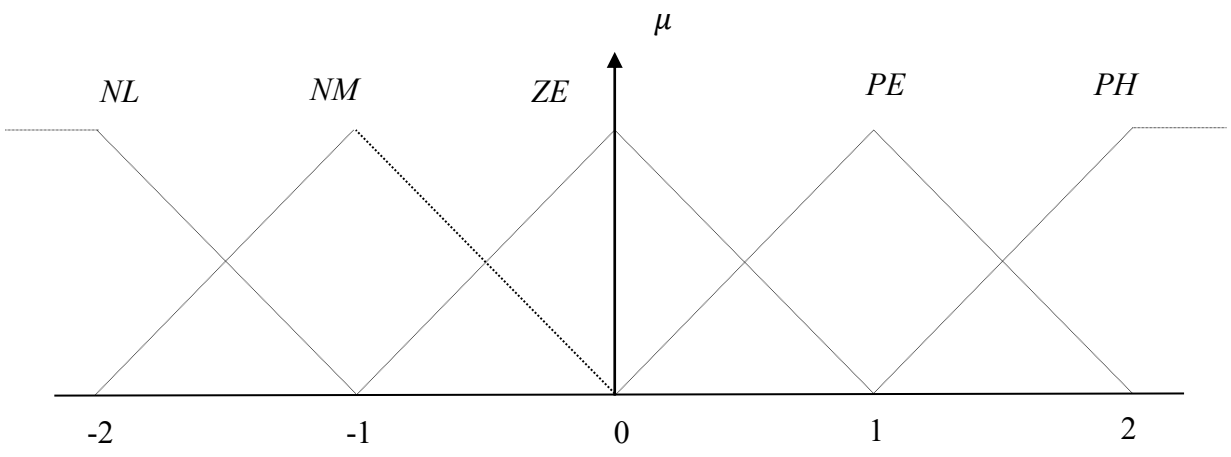


Figure 3.7: Membership functions of fuzzy-logic

4) If Δr_{t-1} is PH while $\Delta \dot{r}_{t-1}$ is NL, the velocity is high but comparatively stable, the window size could be medium to weaken the noise effect while keeping a close track of the high velocity.
5) If Δr_{t-1} is NL while $\Delta \dot{r}_{t-1}$ is PH, the velocity is low and just about to increase, the window size should be medium to be ready to grow, in order to capture the possible increase of the target velocity.

The fuzzy output NL corresponds to a large window size, PH to a small window size, and all others to a medium window size in between. The typical Weighted Average Method for defuzzification is chosen to calculate the windows size n, expressed as:

$$n = n_{max} - k_o \frac{\sum \mu(\bar{o})\, \bar{o}}{\sum \mu(\bar{o})} \tag{3.20}$$

where n_{max} is the upper bound of the window size which is case-specific, k_o is the output gain, $\bar{o}$ is the median of the membership output, μ is the membership function.

Table 3.1 Fuzzy rules of velocity estimation

		Δr_{t-1}				
		NL	NM	ZE	PM	PH
$\Delta \dot{r}_{t-1}$	NL	NL	NL	NL	PM	ZE
	NM	NL	NL	PM	ZE	PM
	ZE	NL	PM	ZE	PM	PH
	PM	PM	ZE	PM	PH	PH
	PH	ZE	PM	PH	PH	PH

Once the window size is obtained, the estimated horizontal velocity is calculated according to (3.18), the estimated vertical velocity can be calculated in a similar way. Then the velocity estimator can be calculated as

$$v_{e_t} = \begin{bmatrix} \dot{r}_{est_t} \\ \dot{s}_{est_t} \end{bmatrix} = \begin{bmatrix} \widehat{\dot{r}_t} - \dot{r}_t \\ \hat{\dot{s}}_t - \dot{s}_t \end{bmatrix} \tag{3.21}$$

which is then incorporated into the multi-model system dynamics (3.8) as:

$$x_t = f_t^{c_t}(x_{t-1}, u_t) + \Gamma \cdot \varepsilon_t + v_{e_t} \tag{3.22}$$

thus the system transition models (3.10) ~ (3.12) are updated as:

$$\begin{bmatrix} r_t \\ s_t \\ \dot{r}_t \\ \dot{s}_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{t-1} \\ s_{t-1} \\ \dot{r}_{t-1} \\ \dot{s}_{t-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \varepsilon_t + \begin{bmatrix} 0 \\ 0 \\ \dot{r}_{est_t} \\ \dot{s}_{est_t} \end{bmatrix} \tag{3.23}$$

$$\begin{bmatrix} r_t \\ s_t \\ \dot{r}_t \\ \dot{s}_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{\sin\Omega_t^{c_t} T}{\Omega_t^{c_t}} & \frac{\cos\Omega_t^{c_t} T - 1}{\Omega_t^{c_t}} \\ 0 & 1 & \frac{1 - \cos\Omega_t^{c_t} T}{\Omega_t^{c_t}} & \frac{\sin\Omega_t^{c_t} T}{\Omega_t^{c_t}} \\ 0 & 0 & \cos\Omega_t^{c_t} T & -\sin\Omega_t^{c_t} T \\ 0 & 0 & \sin\Omega_t^{c_t} T & \cos\Omega_t^{c_t} T \end{bmatrix} \begin{bmatrix} r_{t-1} \\ s_{t-1} \\ \dot{r}_{t-1} \\ \dot{s}_{t-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix} \varepsilon_t + \begin{bmatrix} 0 \\ 0 \\ \dot{r}_{est_t} \\ \dot{s}_{est_t} \end{bmatrix} \tag{3.24}$$

$$\begin{bmatrix} r_t \\ s_t \\ \dot{r}_t \\ \dot{s}_t \\ \ddot{r}_t \\ \ddot{s}_t \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 & T^2/2 & 0 \\ 0 & 1 & 0 & T & 0 & T^2/2 \\ 0 & 0 & 1 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & T \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{t-1} \\ s_{t-1} \\ \dot{r}_{t-1} \\ \dot{s}_{t-1} \\ \ddot{r}_{t-1} \\ \ddot{s}_{t-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \varepsilon_t + \begin{bmatrix} 0 \\ 0 \\ \dot{r}_{est_t} \\ \dot{s}_{est_t} \\ \ddot{r}_{est_t} \\ \ddot{s}_{est_t} \end{bmatrix} \quad (3.25)$$

Note that for the acceleration model, the acceleration estimator $\ddot{r}_{est_t}$ and $\ddot{s}_{est_t}$ can be calculated following the same fuzzy-logic-based method. In addition to the linear velocity estimation of the target motion, the fuzzy based adaptive windowing velocity estimator is integrated into system dynamics as an additional information. Therefore, the effect of the sensing noise can uncertainties can be reduced and the tracking of the target motion can be more accurate, and the prediction of the state of the system is more reliable.

3.5 Algorithm of Adaptive MM with Fuzzy-based Estimation

The proposed algorithm of adaptive multi-model with fuzzy-logic velocity estimation is illustrated in Fig. 3.8 and the pseudo code is shown in Algorithm 1. Overall in this chapter, the system dynamics in the proposed PF is enhanced in two main aspects, namely the multi-model with adaptive transition matrix and the fuzzy-based windowing velocity estimation. It differs with and improves against the standard PF in that, in the standard PF, the system dynamics is either single-model or multi-model with constant transition matrix which is usually empirically set, and the velocity estimation is simply based on the window size of 1, with no consideration of the actual conditions of the system environment.

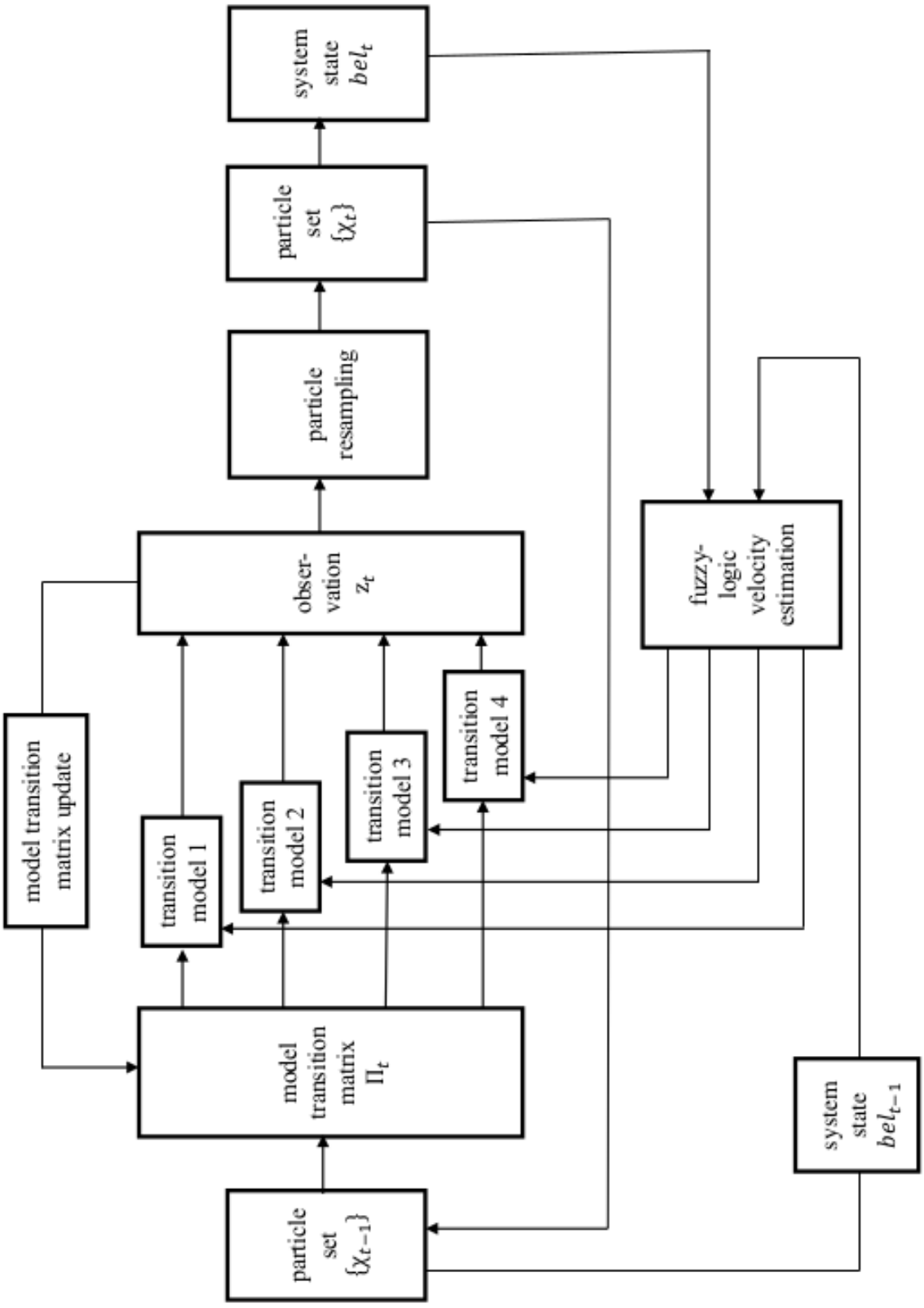


Figure 3.8: The proposed algorithm of adaptive multi-model with fuzzy-logic velocity estimation

Algorithm 1: Adaptive Multi-Model and Fuzzy-Logic--based Velocity Estimation

Inputs: the previous particle set $\{\chi_{t-1}\}$, the current model transition matrix Π_t, observation z_t, control u_t,
Variables: particle weight w_t^i, number of particles m
Outputs: the current particle set $\{\chi_t\}$

set $\{\chi_t\}$=$\{\emptyset\}$, $w_t^i = 0$,
for $i = 1$ to m, do
  generate random x_{t-1}^i from $\{\chi_{t-1}\}$ according to weights in w_{t-1}^i,
  sample from the motion models: $x_t^i = f_t^{c_t}(x_{t-1}^i, u_t) + \Gamma \cdot \varepsilon_t + ve_t$ according to Π_t,
  calculate the weight: $w_t^i = p(z_t | x_t^i)$,
  add $\{x_t^i, w_t^i\}$ to $\{\chi_t\}$
end for
normalize w_t^i,
calculate the potential of each model and update Π_t to get Π_{t+1},
resample $\{\chi_t\}$,
estimate the density of $\{\chi_t\}$ to get the current system state bel_t,
estimate the velocity based on bel_{t-1} and bel_t, and update the transition models,
return $\{\chi_t\}$

3.6 Summary and Conclusion

This chapter introduced the technique of the proactive particle sampling using the methods of adaptive multiple model of system state transition and the fuzzy-logic-based velocity estimation to update the proposal distribution of the particle filter.

In details, instead of setting the model transition matrix as constant, the matrix was updated in each cycle of the particle filtering by calculating the transition potential of each model, and the potential of each model was obtained by the proportion of the number of particles generated by this model to the total number of particles. As a result, the matrix was updated in a way that model with higher potential was assigned with a

higher transition probability. Meanwhile, each transition model was adjusted by the fuzzy-logic based velocity estimation to keep track of the most recent changes in target motion, while maintaining good robustness to the effect of noise and error.

Chapter 4

Active System Observations in Bayesian Filtering

4.1 Introduction

The system observations, or the environment measurement, comprise of the second major domain in probabilistic robotics, functioning as the correction to the system state estimation from the system dynamic models in Chapter 3. Generally, the observation model describes the process of how the sensor data are generated from the surrounding physical world and how they can be processed and incorporated in to the system beliefs by probabilistic approaches.

Nowadays in robotic applications, there are a variety of different sensing modalities, including range sensors, image sensors, tactile sensors, object detectors, etc. Each sensor is applicable to its specific scenario and the modeling approach depends on the sensor type. Generally, the image sensor, e.g. the camera, is best modeled by projective

geometry, whereas the range sensor is best modeled by emitting signal and detecting its reflection on the surface of the object in the environment.

As for the target tracking and localization in this thesis, the most important factor is the context of distance, e.g. the distance between robot and human user for tracking, the distance between robot and the reference objects in the environment for localization. Therefore, the main type of sensor selected in this thesis is the range sensor. The range sensors are among the popular sensors in robotics and there exists a wide variety of range sensors. The sonar sensor works by emitting a signal and recording its echo, under the law of wave reflection. It is easy to deploy and is widely available since many commercial robots are equipped with sonar sensor. Moreover, recently in many robotic applications, the WLAN signal is commonly used due to its vast availability [94], and the received signal strength (RSS) is selected as the measurement cue. According to the path loss law, the detected RSS can be mapped as the distance between signal sender and receiver.

4.2 Probabilistic Sensor Observation Models

In order to explicitly incorporate the noise of the sensor measurements into Bayesian filtering, specific observation models are required. As mentioned in Chapter 3, for probabilistic filtering methods, the sensor observation model can be defined as the conditional probability distribution $p(z_t|x_t)$, where x_t is the system state (target location) and z_t is the measurement. This distribution specifies the probability of measurement z_t at the location x_t, or in other words, how the observation z_t is generated from the current

system state x_t. Although this approach is addressed with the range sensors in this thesis, the underlying principles are also applied to other types of sensors, such as camera or landmark detector.

There are two main reasons to model the sensor in the probabilistic way. Firstly, it is inevitable for the sensor signal to be noise-free. The noise can either from signal generation or from signal processing. Furthermore, the response characteristic of a sensor depends on some variables that cannot be modeled explicitly, such as the material of the surface that affects the signal reflection, and the density of air that affects the signal strength. By modeling the sensor measurements as the probability distribution $p(z_t|x_t)$, instead of a deterministic function $z_t = f(x_t)$, these uncertainties can be well accommodated.

4.2.1 Basic Observation Models

The main benefit of modeling sensor measurements in the probabilistic way is the avoidance of extremely crude model to cover all details of the physical world. Instead, the task is how to design the specific models to accommodate different types of uncertainties that may affect the sensor measurements. One simple and basic model is to consider the observation as a Gaussian distribution centered at the sensing data:

$$p(z_k|x_k) \sim N(\mu, \sigma) \tag{4.1}$$

where the mean μ denotes the sensing data and σ^2 is the variance. When referring to two dimension, μ is the mean vector and the variance matrix Σ replaces the variance. Fig. 4.1 illustrates the Gaussian model in both one and two dimensions, assuming that the measurement is at the position (75, 75) in the *x*-*y* plane.

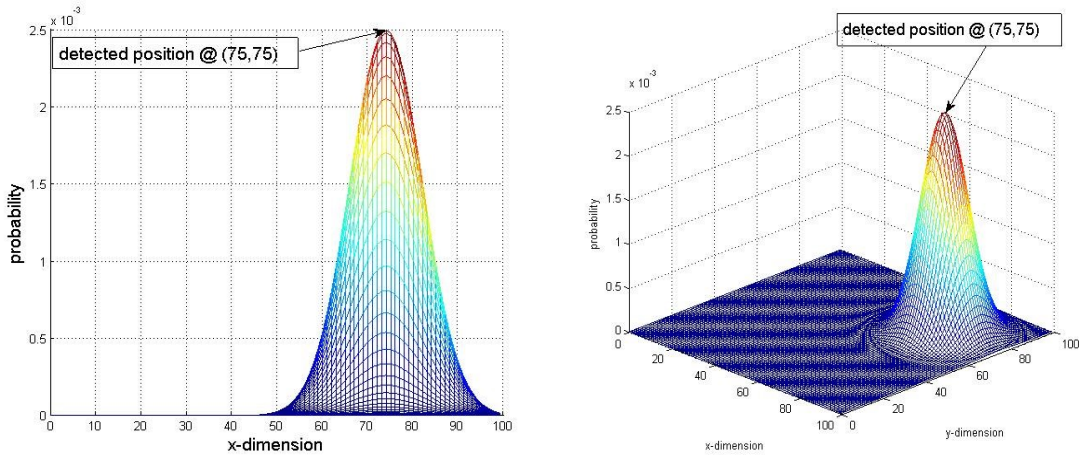


Figure 4.1: Gaussian-based observation model in one and two dimension

Although the proposed technique in this thesis has no requirement of Gaussian assumption, the Gaussian models are still adopted and researched in this section. This is due on the fact that many independent random processes can be well modeled using the Gaussian-based distribution, including the important classes of signal and noise. Besides, the Gaussian distribution is the root of various probabilistic theories and filters. Therefore, the Gaussian represented system observations can collaborate well with other Gaussian-based techniques and also be seamlessly incorporated into the system beliefs. Moreover, the multiple Gaussian models can be conveniently fused to formulate a distribution of multi-sensor observation, due to its efficient computation.

However, merely modeling the observation uncertainties as the measurement noise in the form of Gaussian is unable to cover the types of noise and error that are caused by other factors, e.g. measurement failure, unexpected object, unexplained random noise, etc. Hence, in order to incorporate these factors, four observation models are adopted in this thesis and the ultimate desired distribution $p(z_t|x_t)$ is the mixture of these four densities.

1. Measurement noise model

The measurement noise model corresponds to the case where the sensor detects the correct range but inevitably subject to natural noise and error. This model is simply denoted as a Gaussian distribution centered at the detected range, as show before:

$$p_n(z_t|x_t) = \begin{cases} \eta\mathcal{N}(z_t; z_t^*, \sigma_n^2) & if\ 0 \le z_t^* \le z_{max} \\ 0 & otherwise \end{cases} \tag{4.2}$$

where z_t^* is the detected range, z_{max} the maximum sensor range, $\mathcal{N}$ the normal distribution, and η the normalization factor since the distribution only covers the state space $[0, z_{max}]$. The measurement noise model is illustrated in Fig. 4.2.

2. Unexpected object model

Since the system operational environment is dynamic, the range sensor is likely to detect unexpected object. For example, instead of detecting the wall, the sonar may detect people passing by or other moving objects, making the measurement incorrectly shorter than expected. The probability in such situations can be modeled as an exponential distribution (4.3) since the likelihood of detecting unexpected object decreases with range.

$$p_u(z_t|x_t) = \begin{cases} \eta\lambda_u e^{-\lambda_u z_t^*} & if\ 0 \le z_t^* \le z_{max} \\ 0 & otherwise \end{cases} \tag{4.3}$$

where λ_u is the parameter of the exponential distribution. This model is illustrated in Fig. 4.3.

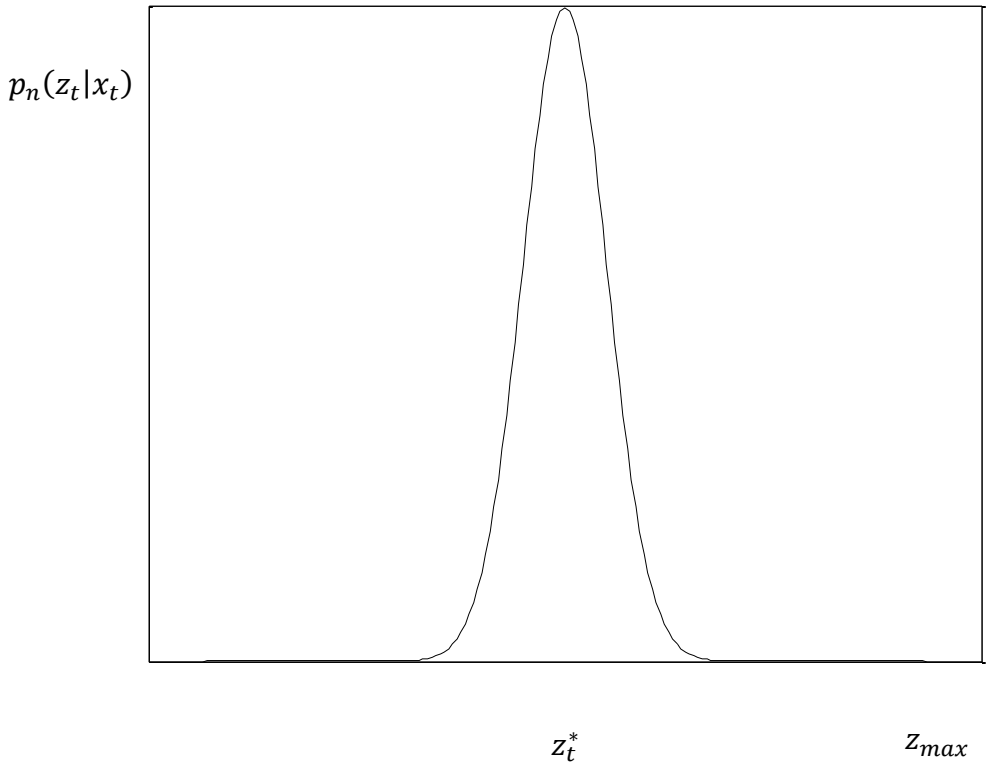


Figure 4.2: Measurement noise observation model

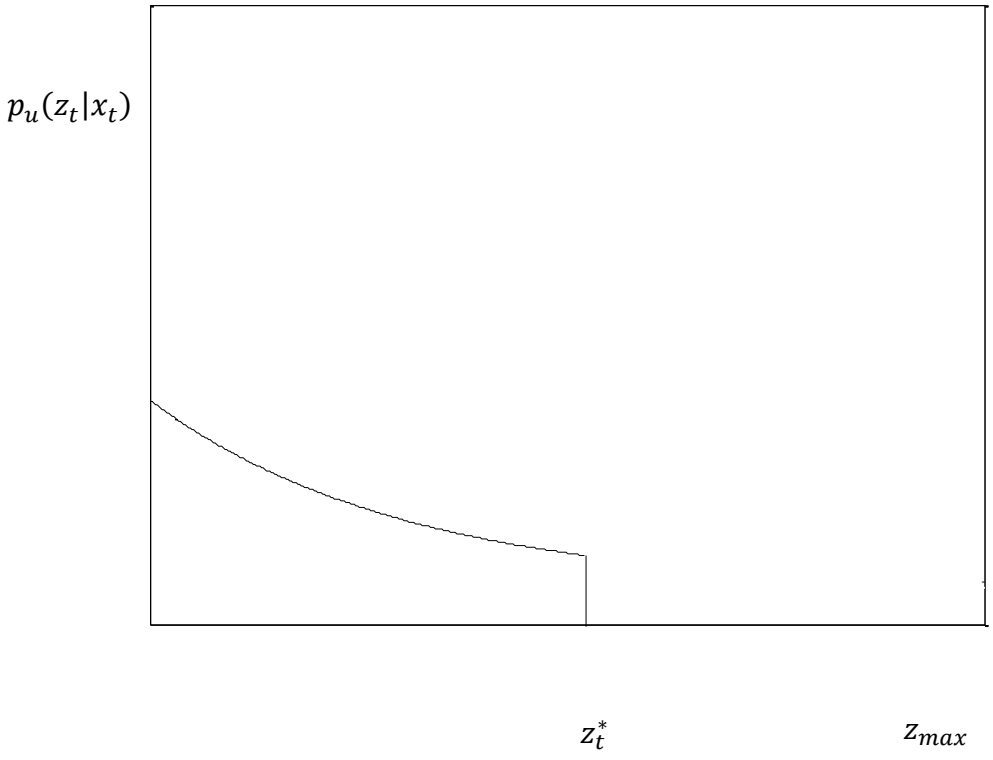


Figure 4.3: Unexpected object observation model

3. Measurement failure model

Sometimes the sensor entirely misses the target and simply returns a maximum range measurement z_{max}. This situation is called "measurement failure", which happens frequently. The measurement failure can be modeled as a point-mass function centered at the maximum range z_{max}, given as (4.4) and illustrated in Fig. 4.4.

$$p_f(z_t|x_t) = \begin{cases} 1 & if\ z_t^* = z_{max} \\ 0 & otherwise \end{cases} \tag{4.4}$$

4. Random noise model

The last model is to cover the unexplainable noise in the measurement. These noises are due to some factors that cannot be modeled explicitly, e.g. phantom readings of sonar. For simplicity, the random noise model is designed as a uniform distribution spread over the entire range space $[0, z_{max}]$ in (4.5) and Fig. 4.5.

$$p_r(z_t|x_t) = \begin{cases} 1/z_{max} & if\ 0 \le z_t^* \le z_{max} \\ 0 & otherwise \end{cases} \tag{4.5}$$

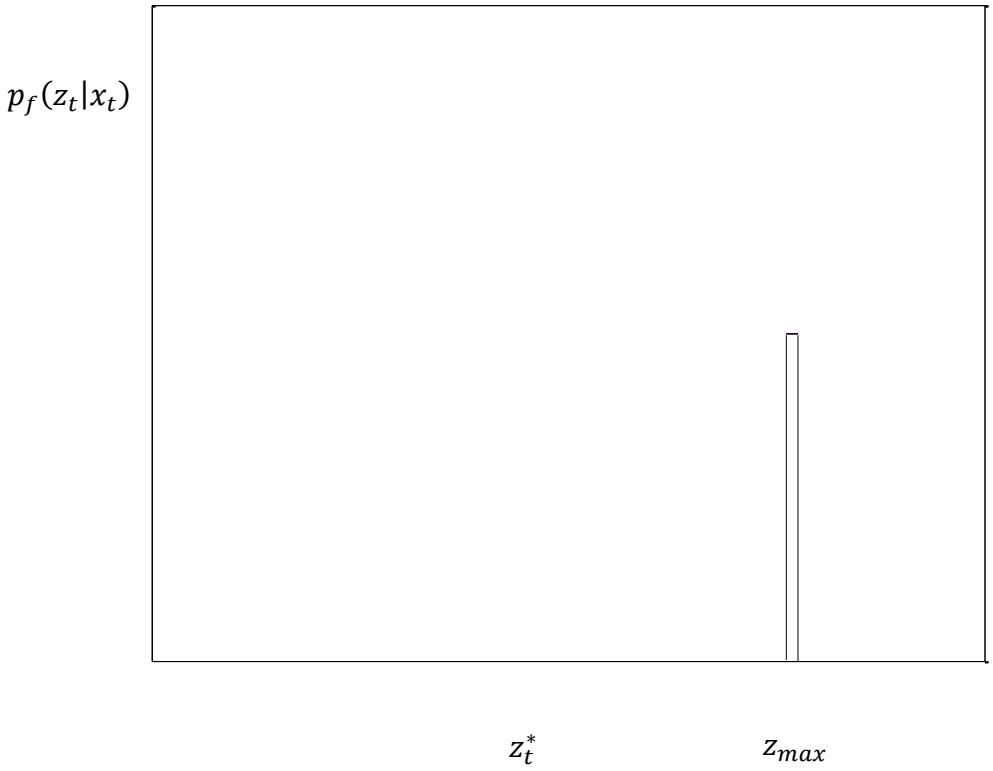


Figure 4.4: Measurement failure observation model

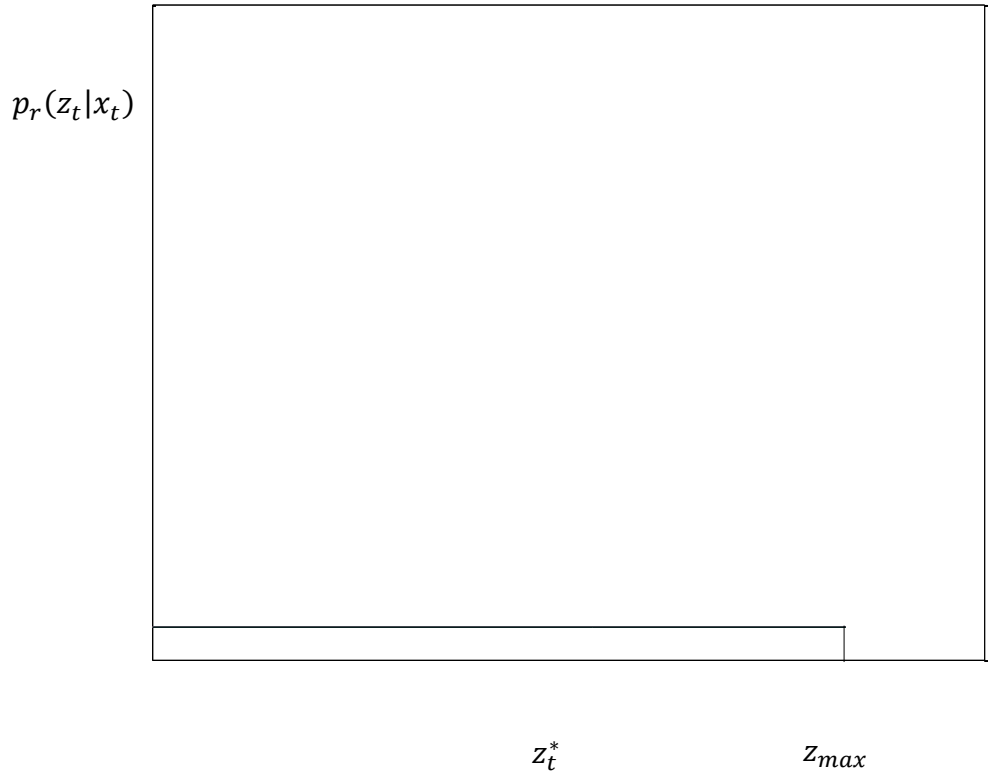


Figure 4.5: Random noise observation model

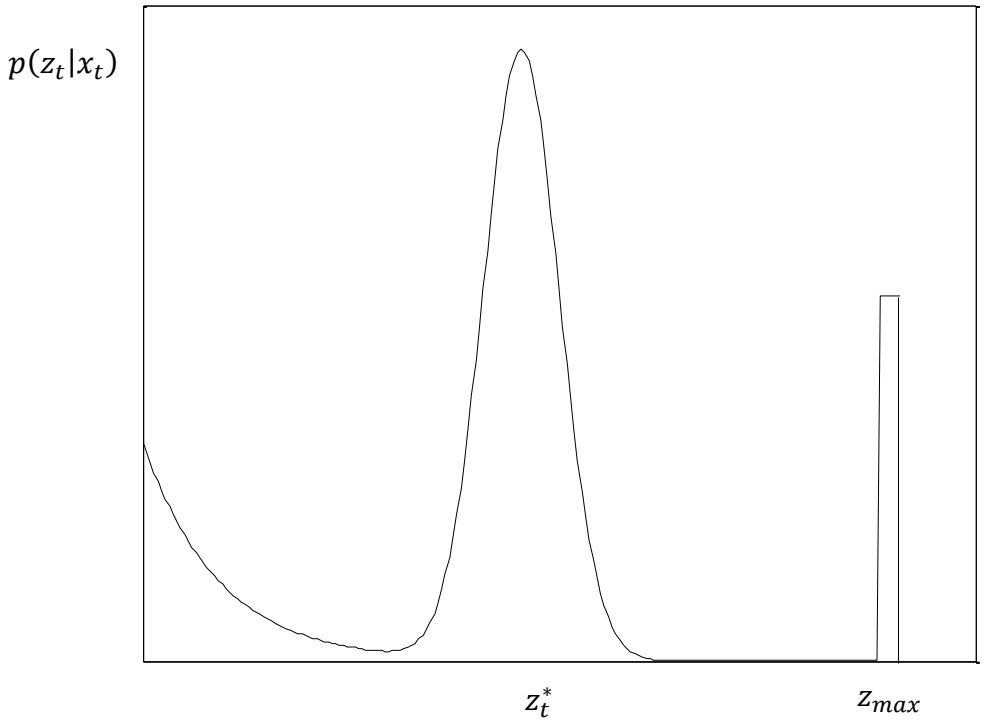


Figure 4.6: Typical weighted-average observation mode

Finally, the above four models are merged together by a weighted average to represent a single sensor measurement, which is given in (4.6) and illustrated in Fig. 4.6.

$$p(z_t|x_t) = \begin{pmatrix} w_n \\ w_u \\ w_f \\ w_r \end{pmatrix}^T \cdot \begin{pmatrix} p_n(z_t|x_t) \\ p_u(z_t|x_t) \\ p_f(z_t|x_t) \\ p_r(z_t|x_t) \end{pmatrix} \tag{4.6}$$

where w_n, w_u, w_f and w_r denotes the weight of the corresponding models, satisfying that $w_n + w_u + w_f + w_r = 1$.

4.2.2 Multiple Sensor Measurement Fusion

Usually a single sensor measurement is far from enough for reasoning and filtering. Hence, many sensors generate one measurement with multiple information, e.g. the

picture returned by camera contains brightness, saturation, color, etc., or more than one numerical value in each cycle, e.g. many range finders generate entire scans of ranges. In other cases, multiple sensors are deployed in the environment to measure the same context. As for location estimation, there are often more than one sensors to detect the target distance. This approach requires the sensor fusion method. Specifically, the multiple sensor measurements can be denoted as:

$$z_t = \{z_t^1, \dots, z_t^K\} \tag{4.7}$$

where K is the total number of measurements and z_t^k is used to refer to an individual measurement at time t.

One simple fusion method adopted in this thesis is to obtain the final likelihood of observation $p(z_t|x_t)$ as the product of the individual measurements:

$$p(z_t|x_t) = \prod_{k=1}^{K} p(z_t^k|x_t) \tag{4.8}$$

Fig. 4.7 shows this fusion method in the case where four sensors are measuring the RSS to detect the target location in a two-dimensional *x*-*y* plane. Note that only the measurement noise model is considered here, without loss of generality.

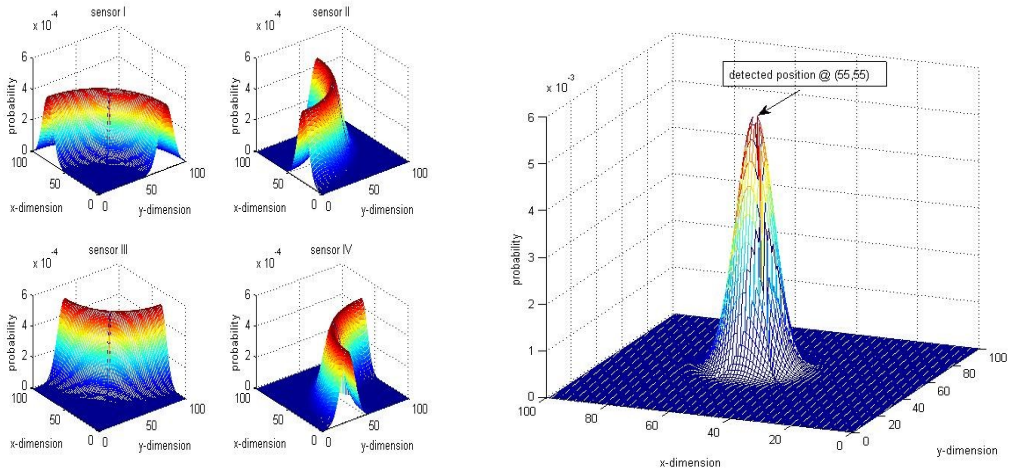


Figure 4.7: Two dimensional Gaussian-based multi-sensor fusion

4.2.3 Analysis of the Basic Models

The use of integration of multiple observation models to take into account different scenarios has a strong robustness to many of the processing errors and signal noise. However, due to the randomness of error and noise, each sensor measurement is contaminated with different level of uncertainty. This fact results in one possible drawback of the multiple observation models, which is, there is no mechanism to differentiate or evaluate the uncertainty in each measurement. Instead, the filtering process passively incorporates all measurements in an indiscriminate manner. This issue needs to be considered since not all sensor data are comparatively accurate and reliable, which is usually the case in the physical world.

Another weakness of the passive and indiscriminate observation is that, it is based on the assumption that the state of the target will not greatly change in a short period of time. Although feasible for many of the cases, this assumption fails in some harsh

scenarios. For example, in the robotic kidnapping problem, the location of the robot will be greatly and suddenly change, which makes the system observation vastly change as well. This situation also applies when the target is transported in an offline period during which the observation is unable to capture this change. The vast change of observation in this case is not resulted from error but from real fact. Therefore, this issue cannot be simply treated in a pure probabilistic way.

To deal with the above two issues, firstly a mechanism of evaluation of sensor measurement quality is introduced in the following section. Then based on this evaluation mechanism, the method of active observation selection is proposed.

4.3 Entropy-based Active Observation

Theoretically, the system transition increases the uncertainty in system state estimation while observation decreases it. Due to the noise in the sensing process, each observation has different performance on decreasing and increases the uncertainty. In this section, entropy is used as the main measurement to evaluate the uncertainty in the state estimation, and furthermore to actively select the sensor observations with higher quality.

4.3.1 Concept of Entropy

Information entropy, or Shannon entropy [95], is a measure of the uncertainty associated with a random variable x and its corresponding probability $p(x)$ in the state space X:

$$H(X) = \sum_{x \in X} p(x) \frac{1}{\log p(x)} = -\sum_{x \in X} p(x) \log p(x) \tag{4.9}$$

with $p(x) = 0$ that $0 \cdot \log {}^{1}/_{0} = 0$. The properties of entropy can be given as follows.

- $H(X) \geq 0$, with equality holds iff $p(x)$ is a point-mass distribution,
- $H(X) \leq log(A),$ with equality holds iff $p(x)$ is a uniform distribution over $[0, \mathrm{A}]$.

Fig. 4.8 shows the extremes where the two equalities hold. From this figure, it can be seen that if the probability distribution is more centered at a specific range, meaning that the system is more certain about its state, the corresponding entropy is smaller, and vice versa. Hence, the value of entropy can be utilized to measure the level of uncertainty in system belief. In other words, in order to make the system more confident about its state, the corresponding entropy should be minimized. Therefore, the task here can be formulated as an entropy minimization problem.

4.3.2 Entropy Enhanced Observation in Particle Filter

In order to compute the entropy of the particle filter, the density of the particles need to be extracted at first. Several techniques are able to estimate the density from the particle set, among which the histogram filter is an extremely efficient option. The discrete histogram filter is superimposed over the particles in the state space and the probability of each bin is computed as the sum of all weights of the particles covered by that bin. Therefore, the entropy of a particle set is calculated according to the corresponding entropy of its density estimated by the histogram filter.

$$H(X) = \sum_{b \in B} p(b) \frac{1}{\log p(b)} = \sum_{x \in X, b \in B} \left(\left(\sum\nolimits_{x^i \in b} w^i \right) * \frac{1}{\log \sum_{x^i \in b} w^i} \right) \tag{4.10}$$

where b denotes the bin, $p(b)$ is the probability of b.

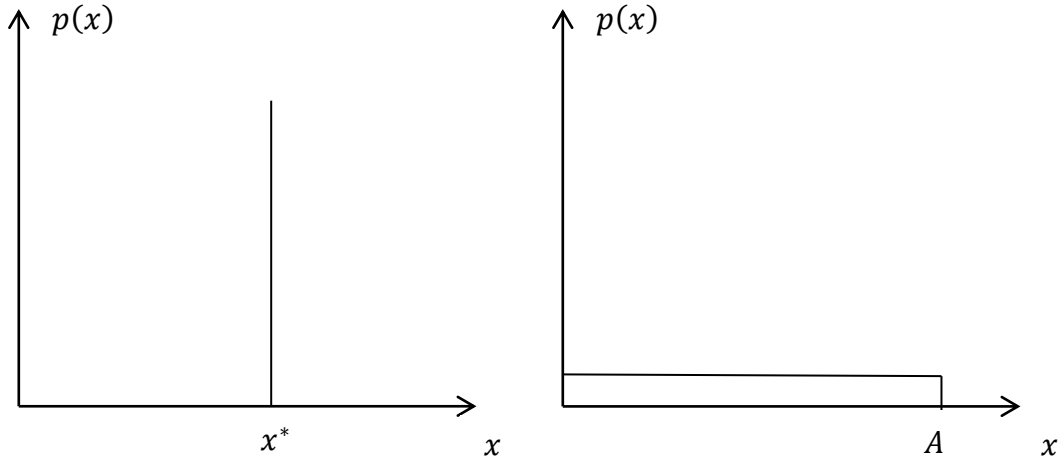


Figure 4.8: Entropy extremes for point-mass and uniform distribution

Theoretically, after each integration of the observation, the uncertainty in the system state estimation decreases, that is $H(X_t|z_t) < H(X_t)$. Inevitably, due to the sensing uncertainty, if $H(X_t|z_t) > H(X_t)$, then this observation is probably incorrect and hence is defined as unacceptable. The entropy-based filtering only uses the observation z_t with which $H(X_t|z_t) < H(X_t)$. If $H(X_t|z_t) > H(X_t)$, then the observation z_t is discarded and another observation z_t at the same time index is collected, making the system observation highly active.

Moreover, as argued before, one possible drawback of the entropy-based localization technique is that this approach is unable to reliably handle the localization failure [96], which is also known as kidnapping problem. This phenomenon can be explained straightforwardly that after localization failure or the robot being kidnapped and teleported to another unknown location in the off-line phase, a large portion of the

sensor readings, including the correct ones, will be largely different from the previous readings. Although these readings are not due to error or uncertainty, they will still be naturally discarded, due to the function of the above entropic observation selection, thus making it difficult to reflect the real location. To deal with this issue, the entropic observation selection is modified in the way that, if all the observations z_t in n consecutive collections yield larger entropy, then the system state jump is inferred and the mean of the three z_t is selected as the observation at time index t. The number of n is set according to specific applications, considering the trade-off between accuracy and efficiency.

4.4 Algorithm of Entropy-based Particle Filter

The proposed algorithm of entropy-based active system observation is shown in Algorithm 2. Overall in this chapter, the proposed PF is mainly enhanced in the aspect of system observations. The most significant difference between the proposed PF and the standard PF is the adoption of entropy to measure the quality of the system observations. In the standard PF, all system observations are treated equal, which is unable to differentiate the abnormally corrupted measurements due to sensing failure or robot kidnapping, from the normally noise-corrupted ones. While in the proposed PF, entropy is introduced to measure the observation quality and the quality is used to calculate the weight of each corresponding observation. The principle is that, normally corrupted observations are assigned larger weights, and vice versa.

Algorithm 2: Entropy-based Active System Observation

Inputs: the particle set $\{\overline{\chi_t}\}$ sampled by system transition model, number of observations n, the current observation z_t^n,
Variables: particle weight w_t^i, number of particles m,
Outputs: the current particle set $\{\chi_t\}$ after incorporating observation

set $\{\chi_t\}$=$\{\emptyset\}$, $w_t^i = 0$, $n = 1$,
calculate the entropy of $\{\overline{\chi_t}\}$ as $H(\overline{\chi_t})$,
do
  collect observation z_t^n
  for $i = 1$ to m, do
    calculate the weight: $w_t^i = p(z_t^n | x_t^i)$,
    add $\{x_t^i, w_t^i\}$ to $\{\chi_t\}$,
  end for
  normalize w_t^i,
  calculate the entropy of $\{\chi_t\}$ as $H(\chi_t)$,
  $n = n + 1$
while $H(\chi_t) > H(\overline{\chi_t})$ and $n < 3$
end do-while
resample $\{\chi_t\}$,
return $\{\chi_t\}$

4.5 Summary and Conclusion

In this chapter, the system observation model was modified in a way that it was able to active select sensor measurements, according to the corresponding entropy in the system belief. Specifically, only the sensor measurements that decreased the belief entropy were incorporated, whereas those increase the entropy were treated unacceptable and discarded. The case of system jump was also considered in a way that, when several consecutive measurements all increased the entropy, the state jump was inferred.

The advantage of the proposed PF based on the techniques introduced in this and previous chapter is that it is able to predict the system state more accurately using adaptive multi-model and fuzzy-based velocity estimator, and to correct the system

prediction with less noise-corrupted observations. The result is the ultimate system belief which is able to more closely reflect the actual environment situations and to provide the more reliable contexts for the higher-level context reasoning in applications and services.

Chapter 5

Monte Carlo Planning and Control

The previous chapters are mainly about how to process the information input into the robotic system to reason about the high-level contexts. However, in many cases where assistive service is required, the capability of processing the input data and estimating the quantities of interests from the surrounding environment is not fully satisfactory. The ultimate objective in many assistive context-aware robotic applications is how to choose the right action according to the situations and the tasks. This issue is also known as the automatic and autonomous planning and control. Some of the typical examples include the unknown environment exploration, coastal navigation, etc. In this chapter, the scenario of the autonomous path planning in the indoor environment is considered and a corresponding Monte Carlo based planning and control algorithm is proposed.

5.1 Introduction

5.1.1 Markov Decision Process

For a robotic agent, the issue of choosing the right action can be formulated as the problem of sequential decision making. The corresponding well-known underlying mathematical framework dealing with this issue is the Markov Decision Process, or MDP [97]. By definition, an MDP is the specification of a sequential decision making problem for a fully observable environment with a Markov transition model and additive rewards. The three main components of a typical MDP are the initial state, state transition model and reward function. The solution of an MDP case is a control policy that guides the agent to choose the right action according to the agent's state in each time index. In other words, a policy is a mapping from states to actions. With a policy, the agent will know what to do next, no matter what state the agent currently resides in. The objective of an MDP is to find the optimal policy with which the agent will collect maximum reward in finite or infinite time horizons by choosing the actions that are mapped from states by the policy. Since the consequences of the actions are inevitably corrupted by uncertainty, the objective is then to find the maximum expected reward.

The structure of the MDP can be summarized as a dynamic decision network, with a strong resemblance to that of the dynamic Bayesian network that is illustrated in Chapter 3. The specifics of the MDP is shown in Fig. 5.1, in which x_t still represents the system state at time index t, z_t the observation of the system state. Compared with Fig. 3.1, here, u_t, the system control or agent action, is made explicit since the choice of action plays an

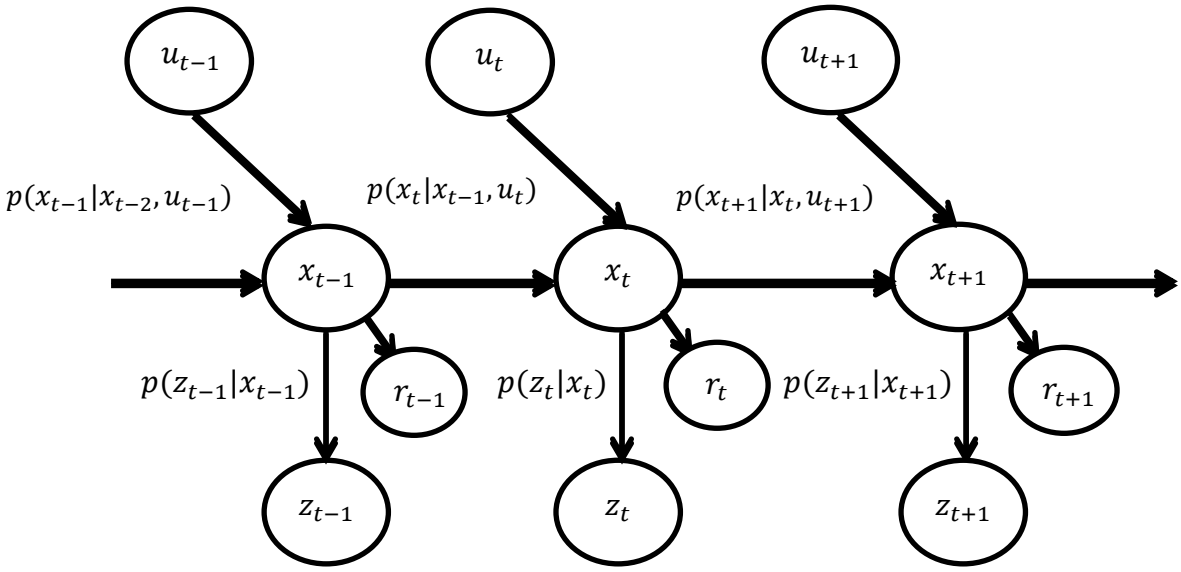


Figure **5.1: The generic structure of dynamic decision network**

important role. Moreover, r_t, the reward of each action if executed at time index t, is incorporated into this structure. According to the assumption that the system state is fully observable, the observation model $p(z_t|x_t)$ is deterministic. However, it allows for stochastic effects of actions, so the state transition model $p(x_t|x_{t-1},u_t)$ is still a probability distribution. In general, the underlying scheme of MDP is a Markov Chain, combined with the choices of actions.

5.1.2 Partially Observable Markov Decision Process

The MDP framework can be utilized to solve the problems of sequential decision making in an exact manner for finite world, where the states, actions, observations and horizons are all finite. However, the main drawback is that the successful implementation of the MDP relies on the usually false assumption that the environment or state is fully

observable, i.e., it denies any uncertainties in the observation model of the system, which is not applicable in the real world. Therefore, the MDP framework that can incorporate the uncertainties in the observed information is developed, namely the partially observable Markov decision process (POMDP).

The POMDP can be considered as a combination of the general MDP and the hidden Markov model. It allows for the uncertainties in both the actions and observations. Specifically, at each time index, the agent receives a probabilistic observation that is a noisy projection of the agent's state. As a result, the observation model $p(z_t|x_t)$ in Fig. 5.1 is a probability distribution. The uncertainties in the observations result in that the agent cannot directly observe the exact current state, thus unable to rely on the policy mapping from states to actions. In order to make a decision, the agent instead holds a belief, which is the probability distribution over all the states the agent might find itself in, given the sequence of observations and actions so far. It is straightforward to think that the expected value of the belief can be used as the estimate to the current state for the control policy. This method, however, only works well when there is no sudden change in the state of the agent, which is not always true in practical applications. Therefore, conditioned on the mode of the posterior is not a viable solution. Instead, conditioning on the belief state is necessary. Hence, the corresponding policy is the mapping from beliefs to actions.

Similar to MDP, the main components of POMDP is the initial belief, belief transition model and reward function for belief. The general decision making cycle in POMDP is 1) to execute the action according to current belief and policy, 2) to receive a new observation, and 3) to update the current belief, e.g., using the filtering techniques.

Due to the characteristic of incorporating the uncertainties in both action and observation, POMDP is applicable for many of the cases in real world. In the aspect of path planning of the context-aware assistive robotic system, the objective is to find the optimal or proper paths for specific tasks in a certain environment. Here, the state is the location of the robot which, as researched in previous chapters, cannot be directly sensed but only approximated in the form of a belief. Therefore, the framework of POMDP is utilized for the task of path planning in this chapter.

5.1.3 Value Iteration Algorithm

One popular and well-understood algorithm to solve the MDP case is the classical value iteration [98]. The basic idea of the value iteration algorithm is to calculate a value function for all the states and then to use these values to select the action optimally in each time index. The importance of defining the value function over all state spaces in non-deterministic environment is that it is capable of dealing with the stochastic effects of the actions that can lead the agent to any state in the state space.

In the case of MDP, where state is fully observable, the central update equation in value iteration is called the Bellman equation, or Bellman update [99], given as

$$V_T(x) = \gamma \max_u \left[r(x,u) + \int V_{T-1}(x') p(x'|u,x) dx' \right] \tag{5.1}$$

where $V_T(x)$ is the value of state x at time index T, u the action, γ the discount factor for the additive rewards, $r(x,u)$ the rewards for action u at state x, $p(x'|u,x)$ the transition probability of state x to state x' with action u executed. Since the update process of the states is in the reverse temporal order, the value iteration is also referred to as the backup

step. For practical purpose, the integration over the continuous state space is usually approximated by discrete decomposition, where the integration is replaced by the finite sum and the value $V_T(x)$ is implemented as a look-up table.

With γ, $r(x,u)$ and $p(x'|u,x)$ known, the procedure of the value iteration is first to assign a random value to each state and then to use the Bellman equation to calculate the value of each state in an iterative manner. It is proved that if the Bellman equation is applied infinitely often, i.e., $T \to \infty$, convergence is guaranteed and a unique equilibrium will be reached. The resulted value $V_T(x)$ for each state is the unique solution and the corresponding optimal policy is then obtained according to the value $V_T(x)$. In practical applications, an error bound is adopted to control the runtime of the algorithm to avoid the infeasible infinite iterations.

As mentioned in previous parts, for the more general case of POMDP where the state x is not fully observable, the agent has to select actions according to the belief instead of the concrete state. The belief is a posterior distribution over states, and the belief space is the space of all the posterior beliefs that the agent might hold about the environment. The corresponding Bellman equation is therefore implemented over the belief space, shown as

$$V_T(b) = \gamma \max_u \left[r(b,u) + \int V_{T-1}(b') p(b'|u,b) db' \right] \tag{5.2}$$

where the state x in (5.1) is replaced by the belief b, which is not a concrete value but a probability distribution.

The theoretically correct replacement of state by belief brings practical problems. In many cases, the belief space is continuous so there are infinite numbers of different values of $V_T(b)$. Moreover, instead of integrating over a finite number of values in (5.1), (5.2) is an integration over the belief which is a probability distribution. There is no guarantee that this integration can be carried out exactly, or proper approximation can be found. The key factor in (5.2) is $p(b'|u,b)$, the probability of reaching state b' from state b after executing action u. This is a distribution over distribution that cannot be calculated in an exact manner. To avoid this complexity, it is beneficial to re-express the term $p(b'|u,b)$ by explicitly showing the observation z

$$p(b'|u,b) = \int p(b'|u,b,z)p(z|u,b)dz \tag{5.3}$$

where $p(b'|u,b,z)$ is a distribution calculated from b, u, and z using the Bayes filter, given as

$$B(x) = p(x'|z,u,b) = \eta p(z|x') \int p(x'|u,x)b(x)dx \tag{5.4}$$

Consequently, (5.2) can be rewritten as

$$V_T(b) = \gamma \max_u \left[r(b,u) + \int \left[\int V_{T-1}(b')p(b'|u,b,z)\, db' \right] p(z|u,b)dz \right] \tag{5.5}$$

where the inner integration $\int V_{T-1}(b')p(b'|u,b,z)\, db'$ has only one non-zero term since $p(b'|u,b,z)$ is a distribution with a single belief calculated from u, b and z, so the integration over the belief b' is degraded into a single-value term and the whole update (5.5) only depends on the integration over the observations z, which is more convenient and efficient than the original update in (5.2). The last term, $p(z|u,b)$, the probability of

receiving observations z after executing action u with belief b, can be further factorized, which will be examined with more details in the following sections.

5.2 General Algorithms of POMDP

The assumption of partial observability brings more complexity to the algorithm of POMDP. It has been proved that, for the special case of finite world, where the state space, the action space, the observation space and the planning horizon are all finite, the original value iteration can be applied in an exact manner and the corresponding value functions can be represented as the piecewise linear functions over the belief space. However, the value iteration algorithm in this case is computationally cumbersome, and even intractable for the more general POMDP cases. Hence, the main principle of solving the general POMDP is to approximate the planning process in order to yield computational improvements, while still maintaining good performance in practical applications. As a result, the best-known performance of the general POMDP solutions comes from the results produced by various ways of approximations to the original value iteration algorithm.

5.2.1 QMDP

The QMDP [100] algorithm can be considered as a hybrid of the solutions of MDP and POMDP, which shortcuts the probabilistic planning by assuming that at some point in the future, the states become fully observable, so that the MDP-optimal value iteration can be implemented in a direct and exact manner. The advantage of QMDP is that it is as

computationally efficient as the MDP, but still returns a policy that is defined over the belief space instead of the state space.

In MDP case, an optimal policy can be found by computing the values for all the states using (5.1). However, the resulting values are defined over states, which is not applicable to the case of partial observability. The QMDP algorithm extends the values of states to the values of belief as shown below.

$$V_T(b) = E_x[V_T(x)] = \sum_{i=1}^{N} p_i V_T(x_i) \tag{5.6}$$

where the value of belief is generalized as the expected value of state, N the number of states and $p_i = b(x_i)$ the probability of state x_i. Therefore, with values of all states calculated using the original value iteration (5.1), the values of beliefs can be represented as the linear combination of the values of states, and the optimal policy can be found by iterating all possible actions u.

5.2.2 Augmented MDP

The principle of the Augmented MDP algorithm, or AMDP [101][102], is to compress the continuous and high-dimensional belief space into a low-dimensional representation and then to use this compact representation for efficient planning. Specifically, the AMDP relies on the assumption that the belief space can be summarized by a lower-dimensional sufficient statistic f, which is the mapping between the high and low dimensional space. The optimal policy is then obtained from $f(b)$ instead of b.

In AMDP, the compact representation is chosen as the tuple

$$\bar{b} = f(b) = \begin{pmatrix} \underset{x}{\operatorname{argmax}}\, b(x) \\ H_b(x) \end{pmatrix} \tag{5.7}$$

where $H_b(x) = -\int b(x) \log b(x)\, dx$ is the entropy of the belief. This compact representation is called the augmented state space. In order to perform value iteration in this augmented state space, in AMDP the augmented state space is discretized and the values of the states are represented as a look-up table. Then the transition probabilities and reward functions over the augmented state space are constructed by learning from Monte Carlo simulations through a sampling procedure, in which the frequency statistic is used to calculate how often an augmented belief $\bar{b}$ transits to another belief $\bar{b}'$ under an action u, and what is the average reward induced by this transition.

Although the AMDP algorithm relies heavily on the assumption that f is a sufficient statistic of the original belief b and that the system is Markov with respect to the augmented belief $\bar{b}$, it is highly practical due to its efficiency and approximation. There are also some similar algorithms to AMDP which are developed by adding more features into the statistic f, or by using different statistics other than the entropy, e.g., the moments of the belief distribution (mean, variance), the eigenvalues and eigenvectors of the covariance, etc. However, the underlying principles are the same.

5.2.3 Monte Carlo POMDP

The Monte Carlo POMDP algorithm, or MC-POMDP [103], approximates the general POMDP by condensing the corresponding belief space using the particle filter and machine learning approaches. The advantage of the MC-POMDP is that it is able to represent the value functions over any arbitrary state space and has no requirement for the

state space to be finite, due to the characteristics of particle filter. In order to use the sampling algorithm, $p(z|u,b)$, the probability of receiving observations z after executing action u with belief b in (5.5), further resolves as

$$p(z|u,b) = \iint p(z|x')p(x'|u,x)b(x)dxdx' \tag{5.8}$$

which, taken into (5.5), gives

$$V_T(b) = \gamma \max_u \left[r(b,u) + \iiint V_{T-1}(B)p(z|x')p(x'|u,x)b(x)dxdx'dz \right] \tag{5.9}$$

where $V_{T-1}(B)$ is the value of the integration $\int V_{T-1}(b')p(b'|u,b,z)\,db'$ and B is the belief calculated from u, b and z. The multi-variable sampling procedure is then performed as $x \sim b(x)$, $x' \sim p(x'|u,x)$ and $z \sim p(z|x')$ to calculate the belief B, which is similar to the general particle filter algorithm.

Furthermore, the representation of the value function is required. In the MC-POMDP algorithm, the value function is defined and calculated over the particle set, and the control policy is a mapping from particle sets to actions, i.e. $\chi \to V(b)$, where χ is the particle set propagated by belief b. The local learning algorithm reminiscent of nearest neighbor is used to compute the value functions that can be updated by particle set. Specifically, this algorithm maintains a set of reference beliefs χ_k with associated values V_k, and the value of the query belief with unseen particle set χ_{query} is then calculated on its K nearest neighbors $\chi_1 \ldots \chi_K$, using Shepard's interpolation, shown as

$$V(\chi_{query}) = \eta \sum_{k=1}^{K} \frac{1}{d_k} V_k \tag{5.10}$$

where $d_1 \dots d_K$ are the distances between the query belief and the reference beliefs, calculated by the KL-divergence.

Using particles, the advantages of the MC-POMDP are thus similar to those of the particle filter. It can maintain a relatively small set of beliefs and is applicable to continuous states, actions, and measurements. However, it inevitably has its own drawbacks and limitations, which will be examined in the following sections, along with the corresponding improvements in accordance with the characteristics and requirements of the assistive context-aware robotic applications and tasks.

5.3 Enhanced MC-POMDP

The MC-POMDP algorithm relies on several approximations. The use of particle set constitutes one such approximation. The second one comes from the learning algorithm of nearest neighbors for the value function representation. The last approximation is due to the Monte Carlo style value iteration, which by nature relies on convergence. In this section, the drawbacks and limitations resulted from these approximations are examined, and the corresponding enhancements of the traditional MC-POMDP algorithm are explored and researched.

5.3.1 Entropy Weighted Monte Carlo Backup

In the traditional MC-POMDP, the normal value iteration backup step (5.1) for the general MDP case is reconstructed as a Monte Carlo simulation in (5.9). Meanwhile, in

order to compute the value function, it is necessary to split the maximization over actions in (5.9) to explicitly specify the action u in the value function.

$$V_T(b,u) = \gamma\left[r(b,u) + \iiint V_{T-1}(B)p(z|x')p(x'|u,x)b(x)dxdx'dz\right] \quad (5.11)$$

$$V_T(b) = \max_u [V_T(b,u)] \quad (5.12)$$

The value function $V_T(b,u)$ is computed in a Monte Carlo simulation instead of the exact integration. In each simulation for a certain action u, a sample state x is randomly selected from the current belief b, or particle set χ. This sample is processed through particle filter to propagate a new particle set χ' and its value $V(\chi')$ is computed using the nearest neighbor algorithm. This process is carried out repeatedly for *N* times, where *N* is a pre-defined parameter, and the corresponding value $V_T(b,u)$ for the action u is computed as the expectation

$$V_T(b,u) = V_T(\chi,u) = \sum_{n=1}^{N} \frac{1}{n}\gamma[r(x_n,u) + V_{T-1}(\chi'_n,u)] \quad (5.13)$$

Although the Monte Carlo simulation converges to the true value if $N \to \infty$ or N is sufficiently large, the expectation in (5.13) treats all particle sets χ'_n with equal probability or weight, $1/n$. However, the sets resulted in each simulation are with different qualities, because some of the sets are closer to the true state than the others. Therefore, it is beneficial if these sets can be differentiated or evaluated. As mentioned in previous chapters, each of the resulting particle set obtained by the particle filter should theoretically contain less uncertainty, which is sometimes not true due to the stochasticity in the model transition and sensor observation. The particle set that is more corrupted by

noise should be able to be detected and processed with special attention. In order to achieve this aim, a technique to measure the uncertainty in the particle set using entropy is proposed. This technique is similar to that of the active selection of sensor observation in Chapter 4 but different in the implementation.

In order to evaluate the particle set in each simulation, the particle set, or belief state, is modified in a way similar to the AMDP that the entropy of each set is integrated as a new index to construct an augmented particle representation

$$\bar{\chi} = \binom{\chi}{H(\chi)} \tag{5.14}$$

in which $H(\chi)$ is the entropy of the particle set χ, computed as the sum of the entropy of the density of each bin $b \in B$ estimated by the superimposed histogram filter, and the density of each bin is computed as the sum of all weights of the particles covered by that bin, shown as

$$H(\chi) = \sum_{b_j \in B} p(b_j) \frac{1}{\log p(b_j)} = \sum_{x \in \chi, b_j \in B} \left(\left(\sum\nolimits_{x^i \in b_j} w^i \right) * \frac{1}{\log \sum_{x^i \in b_j} w^i} \right) \tag{5.15}$$

The entropy is then integrated into the value function computation (5.13) as the weight of the corresponding particle set χ'_n. The principle is that a large entropy signifies more uncertainty and a particle set with a comparatively large entropy needs to be less weighted. Consequently, the new form of the value function computation is shown as

$$V_T(b, u) = V_T(\chi, u) = \sum_{n=1}^{N} w_{\chi'_n} \gamma [r(x_n, u) + V_{T-1}(\chi'_n, u)] \tag{5.16}$$

where $w_{\chi'_n}$ is the weight of the particle set χ'_n given as

$$w_{\chi'_n} = \frac{1/H(\chi'_n)}{\sum_{n=1}^{N} 1/H(\chi'_n)} \tag{5.17}$$

With the augmented particle representation, each set is weighted by its associated entropy in the process of the Monte Carlo backup. Implemented in this way, this algorithm is able to actively alleviate the negative impact of the particle sets that are with high entropy or uncertainty in the Monte Carlo simulations, thus enhancing the accuracy and efficiency of the approximation.

5.3.2 Direct-sampling Computation of Value Function

Another key issue of the MC-POMDP is how to represent the value function. It is a common approach to represent the states and value functions using discrete decomposition, similarly to the histogram representation. In general MDP cases and some POMDP cases where decomposition methods of approximation are adopted, the value function can be computed from states or discrete beliefs. However, this technique only works well in low dimensional state. In MC-POMDP with high dimension, the belief is represented by particle set and the value function is thus computed using particle set. Although this method can represent the value functions over any arbitrary state space, the particle set with usually large size of *M* is of *M*-dimensional and the probability of any particle set to be observed twice is zero, due to the stochastic nature of the particle propagation.

In the traditional MC-POMDP, the value function is represented using a local learning algorithm reminiscent of nearest neighbor as shown in (5.10). However, the adoption of the KL-divergence as the metric of distance has a couple of issues. Strictly speaking, the KL-divergence $D_{KL}(P||Q)$ is not a true metric and is not a distance in the technical sense. It fails to obey the triangle inequality and it is not symmetric since in general $D_{KL}(P||Q) \neq D_{KL}(Q||P)$. Another issue is that, although the KL-divergence between two continuous probability distributions is well defined, it still requires special attention when applied to discrete particle set. In the traditional MC-POMDP, the particle set is converted into continuous distribution by convolving each particle using a Gaussian kernel with small fixed covariance. Although practically feasible, this method inevitably introduces additional approximations. These issues are ignored in the traditional MC-POMDP.

In order to alleviate the approximations introduced by the Gaussian kernel convolution and KL-divergence metric, the proposed algorithm in this section measures the distance between two particle sets in a direct manner using Monte Carlo sampling. Specifically, for the query particle set χ_{query} and one reference set χ_k, the associated measure of distance d_k is computed by the expectation of all the Euclidean distances between two corresponding particles, each of which is directly sampled from the set χ_{query} and χ_k.

$$d_k = E\left(||\chi_{query} - \chi_k||\right) = \frac{1}{M}\sum_{m=1}^{M}(||x_{query}^m - x_k^m||) \tag{5.18}$$

where M is the number of the particles in each set, x_{query}^{m} is one instantiation sampled from the query set $x_{query}^{m} \sim \chi_{query}$ and x_{k}^{m} from the reference set $x_{k}^{m} \sim \chi_{k}$. The expectation is simply calculated with equal probabilities. In this way, the distance of two particle sets is directly computed in a Monte Carlo manner, avoiding the necessity of approximate Gaussian kernels, as well as easing the burden of the convolution computation.

5.3.2 Probability-based Action Selection

In the general sequential decision making process, the action in each update cycle is usually selected according to the greedy principle, i.e., the algorithm traverses all applicable actions in each update and selects the action that yields the largest reward as the target action for the following system transition. The greedy principle is shown as

$$u^{*} = \underset{u}{\operatorname{argmax}} \left[r(b,u) + \int V_{T-1}(b') p(b'|u,b) db' \right] \tag{5.19}$$

where u^{*} is the target action.

The greedy action selection is optimal at horizon 1. The reason for relying on greedy policy is that the enormous branching factor renders the multi-step planning nearly impossible, due to the unpredictable nature of practical implementations. When the agent acquires a new belief state, the action will be adjusted accordingly, which makes the previously decided action inapplicable.

However, it is not always true that the greedy action selection is always the optimal policy. In practice, it may be advantageous to occasionally select a random action, especially in some special cases where the stochasticity is not overwhelming or the

environment is not harshly complex, e.g. in the indoor environment. Therefore, it is possible that proactive planning for the horizon $T > 1$ is more optimal than the greedy action selection. However, random action selection is by nature not always reliable. In order to solve this issue, the proposed MC-POMDP implements the action selection according to the evaluation of the resulted value $V(b, u)$, instead of simply selecting the action with the highest reward or randomly. Specifically, after traversing all actions in each update cycle and obtaining all the values, each action is paired with its own value to form a tuple $\{u_l, V(b, u_l)\}$. Then all the tuples are gathered to formulate an action set θ in which the values of the actions are normalized and represented as the weights of the actions. The next step is similar to the sampling procedure in the particle filter algorithm, where the action for the next update cycle is selected in accordance with the weights as follows.

$$u^* \sim \theta\{u_l, w_{u_l}\} \tag{5.20}$$

where u^* is the optimal action, θ the action set, $w_{u_l} = \frac{V(b,u_l)}{\sum_{l=1}^{L} V(b,u_l)}$ the normalized weight of the associated action u_l, and L the dimension of the action space. This method of action selection can be considered as a balance between the greedy policy where only one certain action is selected and the random policy where no action is selected with certain.

5.4 Algorithm of the Proposed MC-POMDP

The algorithm of the proposed Monte Carlo POMDP is shown in Algorithm 3.

Algorithm 3: Monte Carlo POMDP

Inputs: initial belief b and value function V
Variables: system transition model, observation model, control of action u, particle set χ, χ', particle entropy $H(\chi)$, action weight w_χ, action value $V(b,u)$, action set θ
Outputs: ultimate converged value function V

sample state $x \sim b(x)$
initialize χ with M samples from $b(x)$
repeat until convergence of value function V
  for $l = 1$ to L, do
    $V(b,u_l) = 0$
    for $n = 1$ to N, do
      select random $x_n \in \chi$
      sample from the motion models: $x'_n \sim p(x'_n|u_l, x_n)$
      sample from the observation model: $z_n \sim p(z_n|x'_n)$
      propagate the new particle set: $\chi'_n = \mathbf{particle\_filter}(\chi, u_l, z_n)$
      calculate the entropy of the new set $H(\chi'_n)$ according to (5.15)
      calculate the associate value function $V(\chi'_n, u_l)$ according to (5.18) and (5.10)
    end for
    calculate the weight of particle set $w_{\chi'_n}$ by normalizing $\frac{1}{H(\chi'_n)}$ according to (5.17)
    calculate the value of action $V(b,u_l)$ according to (5.16)
    add $\{u_l, V(b,u_l)\}$ into the action set θ
  end for
  update value function: $V(\chi) = \max_{u_l} V(b,u_l)$
  normalize the values of actions: $w_{u_l} = \frac{V(b,u_l)}{\sum_{l=1}^{L} V(b,u_l)}$
  select the optimal action by sampling the action set: $u^* \sim \theta\{u_l, w_{u_l}\}$
  sample from the motion model: $x' \sim p(x'|u^*, x)$
  sample from the observation model: $z \sim p(z|x')$
  propagate the new particle set: $\chi' = \mathbf{particle\_filter}(\chi, u^*, z)$
  update the particle set: $x = x'$, $\chi = \chi'$
end repeat
return V

5.5 Summary and Conclusion

In this chapter, an improved Monte Carlo POMDP algorithm was proposed in the aspects of the Monte Carlo backup computation, value function representation and action selection. The particle set was augmented by incorporating the index of entropy, which was further integrated into the Monte Carlo computation of the value function to alleviate the negative impact of the uncertainties in the set. The value function was represented by expectation of the distances between the particles from query set and the reference sets using the direct sampling instead of the KL-divergence to avoid the randomness that was introduced by Gaussian kernel convolution. The values of each action were normalized and represented as weights, which guided the action selection for the next update cycle, whereas in the traditional MC-POMDP, the action was selected simply based on the greedy principle or random principle.

Chapter 6

Results and Analysis of Experimentation

First, the proposed particle filtering method, including the adaptive multi-model, fuzzy-logic velocity estimation, and the active sensor observation selection, is implemented in this chapter for the experimentations of human tracking and self-localization. Then the proposed Monte Carlo POMDP algorithm with entropy backup, sampling-based value function representation, and action selection, is implemented for two different autonomous path planning tasks in the indoor environment. All the experiments of tracking, self-localization and path planning are chosen and designed according to the context-aware robotic system requirements in Chapter 2, i.e. changing the system behaviors to adapt to the users visual and location contexts. The corresponding results are compared and analyzed.

6.1 Experimental Setup

The mobile robot PeopleBot^{TM} from MobileRobots Inc., shown in Fig. 6.1, is used for experimentations and analysis. Performance PeopleBot^{TM} is an intelligent mobile robot specially designed and equipped for human-robot interaction research and applications.

Figure 6.1: Experimental platform of Peoplebot^{TM}

The basic components of the PeopleBot^{TM} include an aluminum body, balanced drive system (two-wheel with differential with casters), reversible DC motors, motor-control and drive electronics and high-resolution motion, all managed by an onboard microcontroller and mobile-robot server software [104]. The height of the robot is 1,115 mm with an adult people-height upper deck for sensing and interaction accessories, including a pan-tilt-zoom robotic color camera that is the input device for visual context. The designed and proposed particle filter method for tracking and localization and Monte Carlo POMDP algorithm for path planning is implemented on this platform. The proposed algorithms are programmed in the microcontroller.

For the circular robot that can turn on a spot, it is common to consider the two-dimensional Cartesian coordinates only, ignoring the factor of rotation in order to keep the computational load manageable. The velocity of the robot is also ignored, assuming that the robot is able to move into any state of the environment in a reasonable time.

In this chapter, firstly, the results of human face detecting and tracking method based on the proposed particle filtering are shown. The reason the human face is selected as the measurement cue is that in many assistive applications, the targets are human themselves. Next, the human-machine tracking and following experiments are carried out on the mobile robot in two types of real time scenarios: straight-line tracking and circular motion tracking. Then the experiments of robotic self-localization are performed for three types of motions: straight-line trajectory, circular trajectory, and random walking. In the end, the proposed Monte Carlo POMDP algorithm is used for the autonomous path planning task in the indoor environment for the task of automatic objects transportation for human users, as one type of assistive services.

6.2 Face Tracking

The human face detection algorithm is based on the algorithm developed in [105], which is able to detect human face in each frame of the video stream. Once the human face is detected in the frame, it is modeled as a Gaussian distribution according to (4.1).

6.2.1 Results and Analysis of Face Tracking

The visual context input is the position of the detected human face, collected in real-time from the mounted camera. Two sets of human face detection results are shown in Fig. 6.2 and in Fig. 6.3, with the standard PF and the proposed PF, respectively. In both sets, the human face is detected at a speed of 20 *fps*. All the frames in both figures are from videos with the similar target and environmental conditions.

As indicated at each frame in Fig. 6.2, with complex background and conditions of light, the face detection with the standard PF exhibits several false detection regions. These false regions inevitably affect the detection performance and probably lead to tracking failure. On the other hand, in Fig. 6.3, under the same background and light conditions, the proposed PF method still identifies the human face and is able to track it correctly. Meanwhile, the proposed PF method also discards the background clutter and changes of light, thus remaining stable and accurate.

The main reason of the false detections is that there is no perfect face detection algorithm that can reliably differentiate all non-face objects. In the proposed PF, these non-face detections can be effectively eliminated by the designed system dynamics and observations. Specifically, the effects of the system state jump, i.e. the case of the human face jumping from one place to a farther place in a short period of time, and the uncertainty-corrupted sensor measurements are reduced. While in the standard PF, all these effects are taken into the system belief. The outcome is that some particles are allocated to the incorrect positions or assigned with the incorrect weights, which produce the false detections in the final tracking results.

Figure 6.2: Human face tracking with standard PF

Figure 6.3: Human face tracking with proposed PF

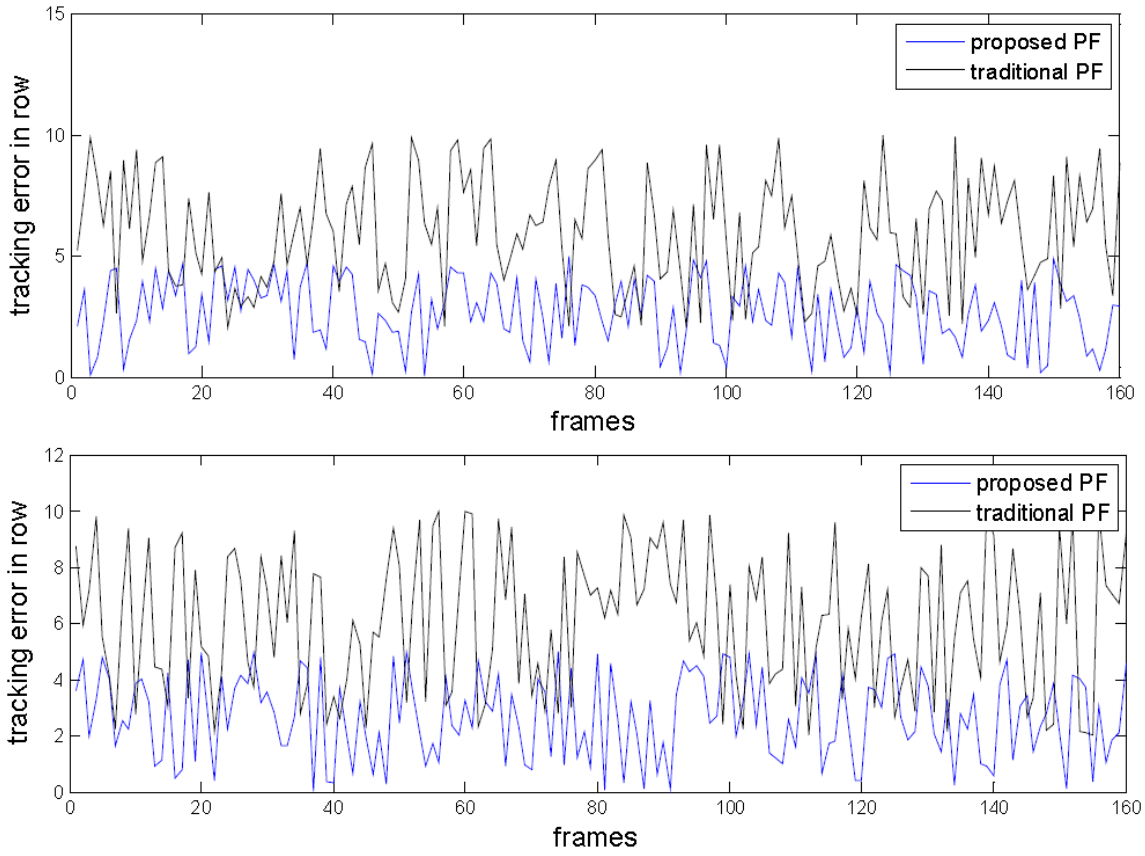


Figure 6.4: Human face tracking errors in row and column

Fig. 6.4 compares the tracking errors in row and column of the proposed particle filter and the standard particle filter. The error is defined as the distance between the true position and the detected position of the human face, and is measured in the unit of centimeter. Moreover, if there are multiple detections in a single frame as Fig. 6.2 shows, the detected position is simply calculated as the mean of these positions. It can be shown from the figures that under mild tracking conditions with simple background and conditions of light, both PFs have similar tracking errors. However, in case of hard tracking conditions, the proposed PF enhances the tracking performance by around 50%, both in row errors and column errors.

6.3 Human-Robot Tracking and Following

In this section, the robot is set to track and follow the human movement in real time, based on both the standard particle filter and the proposed particle filter.

6.3.1 Straight-line Tracking and Following

Fig. 6.5 shows the results of straight-line tracking of forward and backward motions with abrupt changes, under two different tracking error tolerances and moving speeds.

The black curve and blue curve represents the proposed PF and the standard PF, respectively. The moving speeds are selected according to the study of human walking speeds [106], especially considering the elders and people with disabilities. Therefore, a normal speed at 100 mm/s and a relatively slower speed at 60 mm/s are considered. The circular motion speeds in the following part are selected in a like manner. Moreover, in both cases there is no dramatic change in the moving speed, due to the consideration that the elders and people with disabilities usually prefer and exhibit constant moving. However, the proposed technique can be also applied to the scenarios with fast-changing moving speeds.

Specifically, in this experiment, the action of person is 1) to remain still (error=0, ideally), 2) to move forward (error<0), 3) to stop (error=0), 4) to move backward (error>0), 5) to stop (error=0), 6) to move forward (error<0), 7) to abruptly move backward (error>0), and 8) to stop at last. The objective of tracking is to maintain the human face in the center of the visual field by robot motion. In addition, the relative error is the ratio of the distance between the target position and the center point of visual field

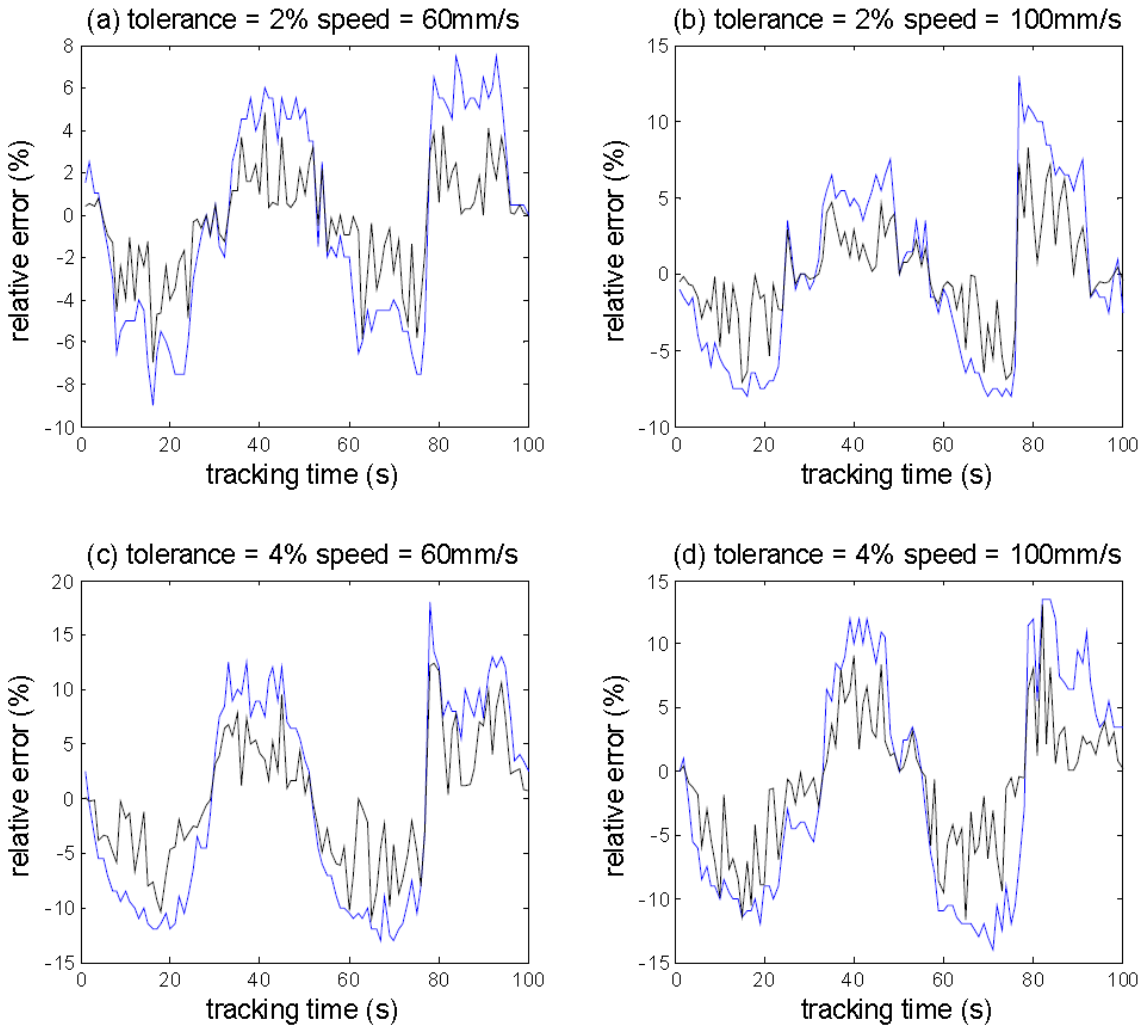


Figure 6.5: Straight-line following with standard and proposed PF

to the length or width of the visual field, and the tolerance represents how far the target position can deviate from the center point of visual field before the robot moves. The value 0 of the relative error means the ideal distance between the robot and the user, and the positive and negative values represent the scenarios in which the robot is too close to or too far away from the user. The maximal values of the relative errors depend on the actual environmental situations, including system setups, application requirements, etc.

On the other hand, higher speed makes the robot follow the user with better real time quality whereas comparatively less stable, which is shown from the error in (b) and (d). The best tracking performance happens in (a), with low tolerance and low speed. In addition, when abrupt motion change happens at around 80*s* in (a), (c) and 45*s* in (b), (d), the robot still maintains a good and fast response using the proposed PF.

The advantage of the proposed PF with respect to the standard PF is that, with a more closely and accurate tracked human face, the motion of the robot can be better synchronized with that of the user. With a single correctly detection target, the robot has a more dedicated goal of moving. While in the case of using the standard PF, when there are multiple detections, the goal is not well determined and the tracking is unstable and more error-prone. The advantage is further clarified in the following section of circular tracking.

Note that although only two speeds are considered, the tracking system performed well with other speeds less than 180 mm/s, which satisfies many cases of normal and slow human walking.

6.3.2 Circular Tracking and Following

The results of real-time circular motion tracking with two error tolerances and moving speeds are shown in Fig. 6.6.

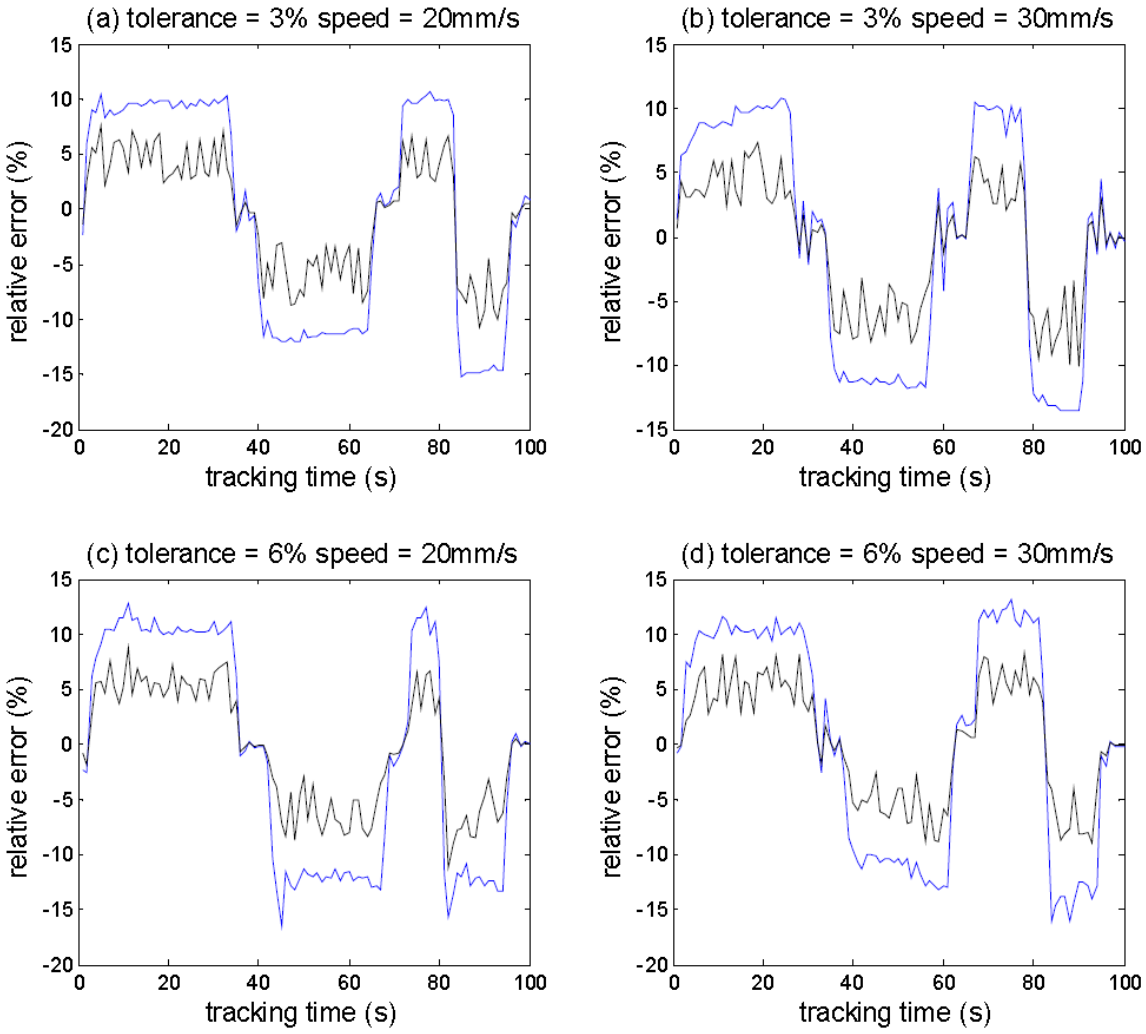


Figure 6.6: Circular following with standard and proposed PF

Again, while both the solutions have stable tracking, the proposed PF has a better performance than the standard PF with smaller tracking error. As shown in the results, the maximal value of the error is reduced from around 10% to 5%. And the maximal values still depend on the actual setup in this specific experiment and can hold different values in other experiments and implementations. The effect of different tolerance is that low tolerance makes the robot following more stable whereas in high tolerance, the robot acts

slowly, thus making the tracking errors less consistent. Similar to the case of straight-line tracking, the higher speed makes the robot act rapidly. However, one drawback in the higher speed is that when user stops, the robot exhibits oscillatory action, as shown in (b) and (d) at around 30s, 60s and 90s. That means the robot rotates clockwise and counter-clockwise alternatively for a while until the user is fixed in the center of the visual field. The best tracking performance happens in (a), with low tolerance and low speed. In the circular motion tracking, the robot also has a good and fast tracking performance in the presence of abrupt motion change from counter-clockwise to clockwise, at around 80s in each case.

6.4 Robotic Self-localization

The experiments of robotic self-localization are performed in a two-dimensional plane with magnitude of 10m-by-10m which is decomposed into a 100-by-100 grid. Four wireless senders are set at each of the four corners and the robot can detect the signal strength from each sender.

6.4.1 Static Self-localization

In this section, the robot is programmed to localize itself while remaining still. Fig. 6.7 shows the results of the static localization at the location with the grid of (67, 48).

In this experiment, the robot remains at the same position in an indoor environment and locates itself repeatedly for 80 times, based on both the proposed PF and the standard PF for comparisons. However, in order to keep the diversity of the particles and to avoid

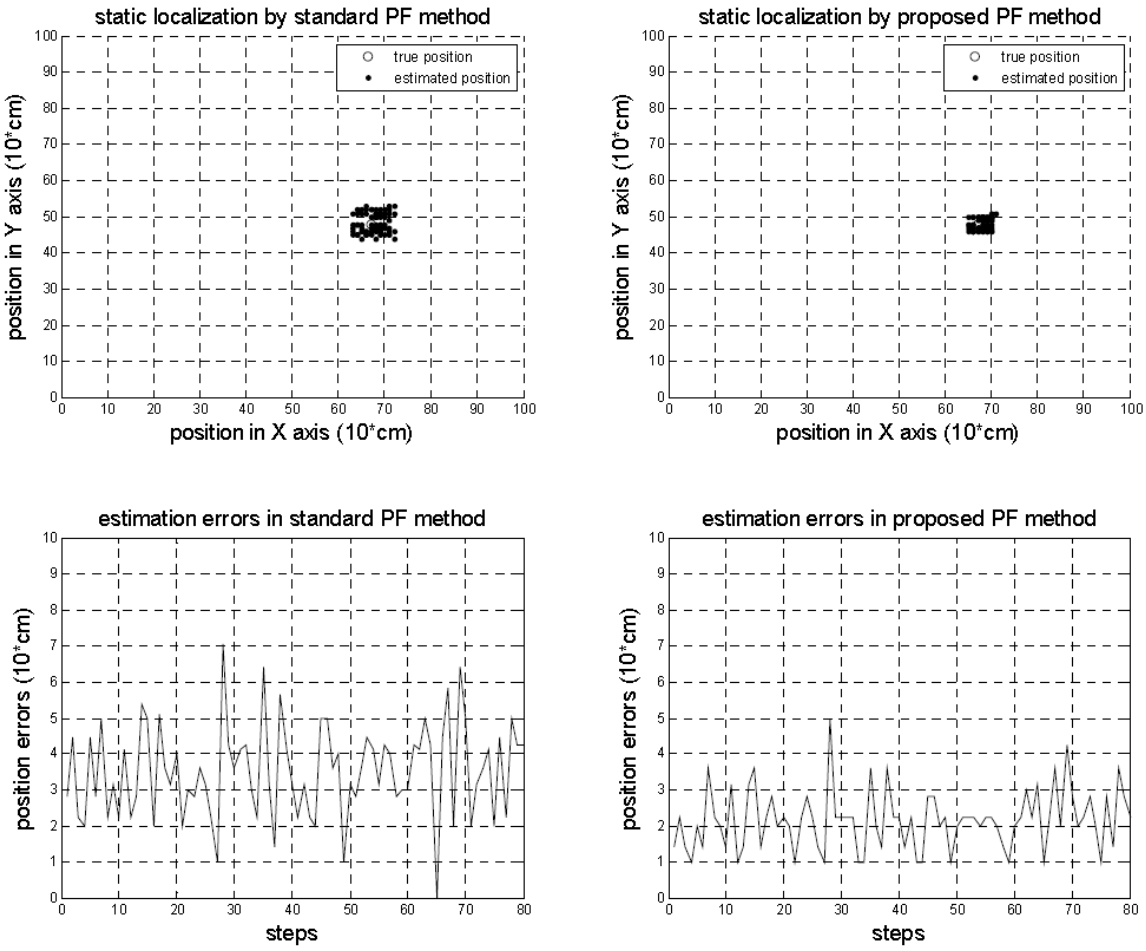


Figure 6.7: Static localization with standard and proposed PF

the issue of sampling variance, the system dynamics in this case is only simulated as a small fluctuation, instead of absolutely staying static. From the results, it can be seen that the proposed PF improves the performance of the standard PF by approximately 30% in the localization error. Since the dynamic model in this case only has minor changes, the localization errors are largely from the uncertainties of observations. Therefore, the entropy plays an important role in this case and the decrease of the entropy in observation is mainly due to the correctly sensed data. The improvements of the performance are mostly contributed by the proposed entropy-based active selection of observation, which

is able to retain the sensing data that decrease the entropy, thus decreasing the system uncertainties accordingly.

6.4.2 Localization in Pre-defined Straight-line Trajectory

In this section, similarly the standard and proposed PF are both implemented on the typical localization of the straight-line motion in the indoor environment. The results of are shown in Fig. 6.8.

In this scenario, the robot is moving in an indoor environment following a pre-defined trajectory with different features, e.g. straight-line motion, abrupt turning motion. As seen from the results, in the straight-line movement, both the standard and proposed PF are able to maintain a stable localization, while the proposed PF still has improved the performance of localization with smaller errors. The largest errors happen around step 50, 100 and 150, when the robot executes the abrupt turning motions. However, the proposed PF recovers quickly and still maintains relatively smaller errors than the standard PF, thus remaining stable, while the standard PF has relatively large errors around those steps and recovers relatively slowly. The results demonstrate that the proposed PF method has a better performance than the standard PF method, in that the proposed PF is able to alleviate the effect of uncertainties introduced by noise to maintain the good localization. Moreover, when the uncertainties are brought by complex and unexpected target motions, the proposed PF also has a robust and stable performance. Also by comparing Fig. 6.7 and Fig. 6.8, it can be seen that the performance of static localization is better than that of dynamic localization, since there is no effect of the motion uncertainties in the static scenario.

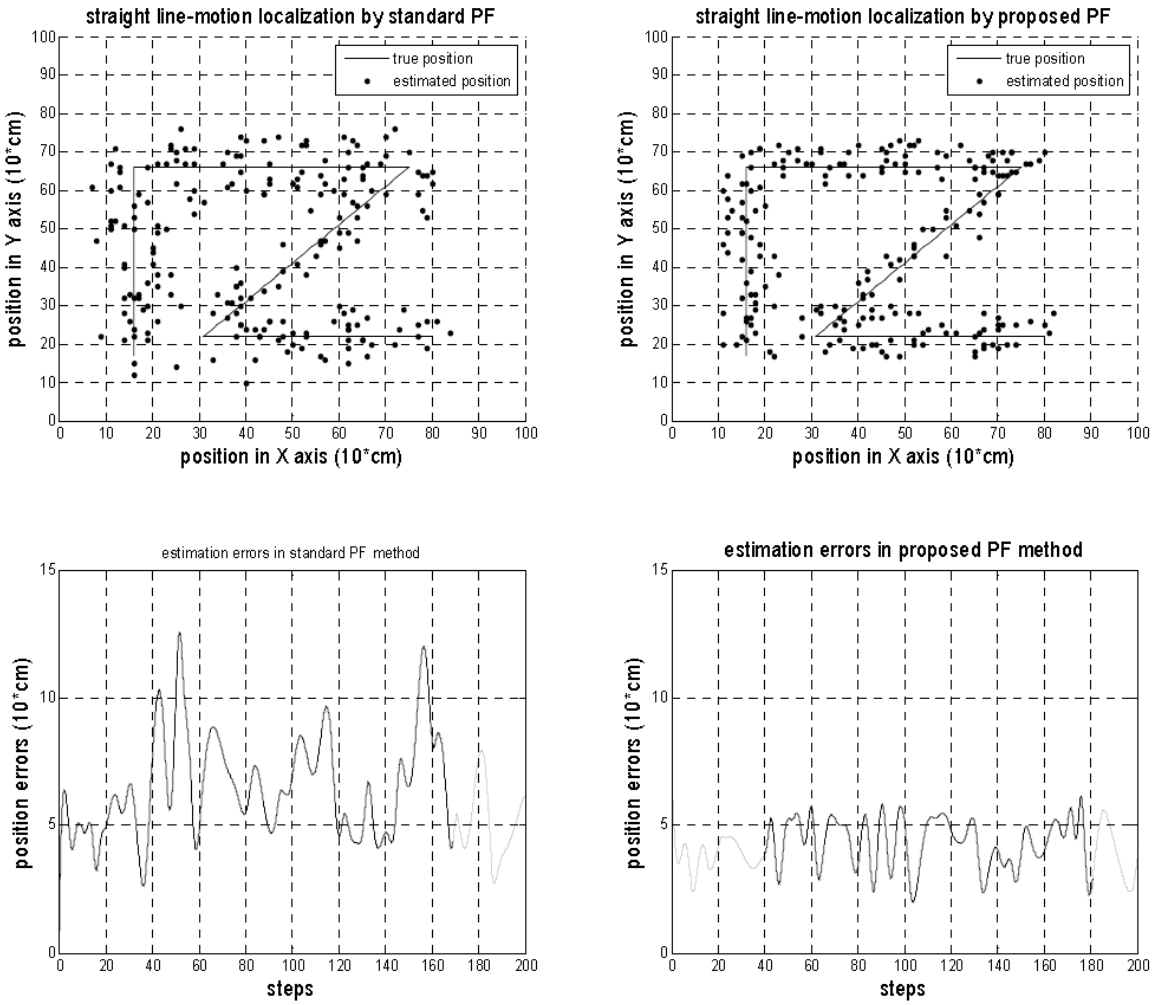


Figure 6.8: Localization with standard and proposed PF in straight-line trajectory

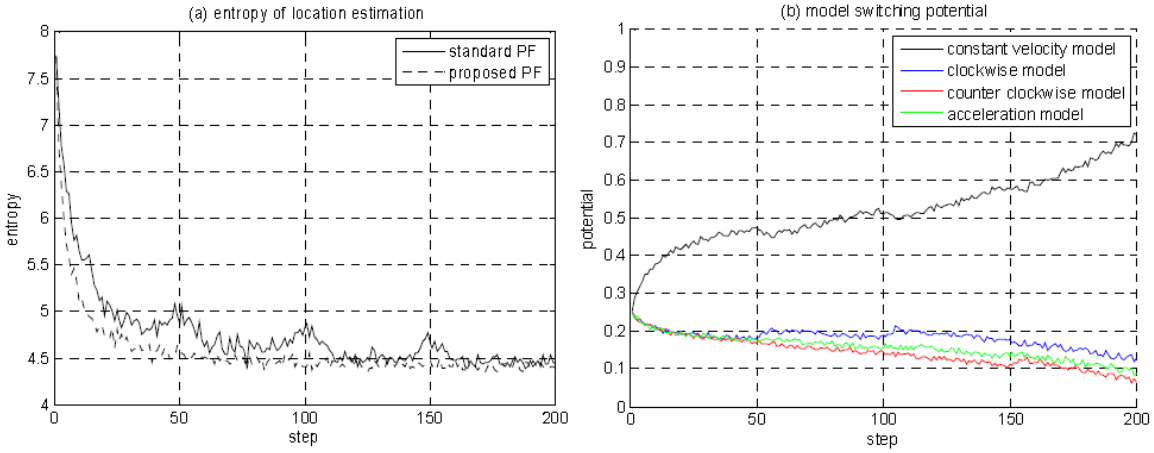


Figure 6.9: Entropy and model switching potential

To further demonstrate the effectiveness of the proposed PF, the entropy and the model switching potential in each localization step is shown in Fig. 6.9 (a) and (b), respectively. From (a) it can be seen that although the entropies in both the standard and proposed PF converge eventually, the entropy in the proposed PF drops faster. This phenomenon is because of the entropy-based system observation which only selects the measurements that strengthen the system belief. Moreover, it is also illustrated that the entropy in the standard PF is affected by the three abrupt turns of the trajectory, while the entropy in the proposed PF is more robust to the robot motion since the multi-model and fuzzy-based system prediction is able to alleviate the effect of abrupt system state change. In (b), the switching potential of the four models are shown to better clarify the mechanism of adaptive model transition. In this figure, the constant velocity model has the largest potential, while the potentials of the other three models all decrease gradually, except at the three turns, when the robot executes the motions of slowing down, turning, and accelerating. Similar conclusions can be made based on the results in the following sections of localization.

6.4.3 Localization in Pre-defined Circular Trajectory

To further test the feasibility of the proposed PF solution in other common situations, the experiments of circular motion localization are performed and the results are presented in Fig. 6.10. The results show that although both of the PFs have stable location estimations, again the proposed one exhibits better performance with relatively smaller errors.

Furthermore, by comparing Fig. 6.8 and Fig. 6.10, it can be seen that the standard PF has comparatively smaller errors in the case of straight-line motion localization than in the case of circular motion localization, except when the robot executes the abrupt turning motions. This phenomenon can be explained by the fact that the circular motion is relatively more complex than the straight-line motion, thus more difficult to be localized accurately. However, in the proposed PF, there are no such obvious discrepancies in the location estimations and the performances of localization are stable and reliable in both scenarios.

Similarly, the entropy and model switching potential is shown in Fig. 6.11 (a) and (b). It can be seen that the entropy in the proposed PF still drops faster than that in the standard PF. Since the robot in this scenario only moves in a circle without any abrupt change of motion, the droppings of the entropies in both cases are smoother than they are in Fig. 6.9 (a). However, the entropies in both cases converge to a larger value than in the straight-line case, simply because the circular motion is more complex. Another consequence of the circular motion is shown in (b), where the potential of the clockwise model is dominant, whereas the potentials of all the other three decrease gradually.

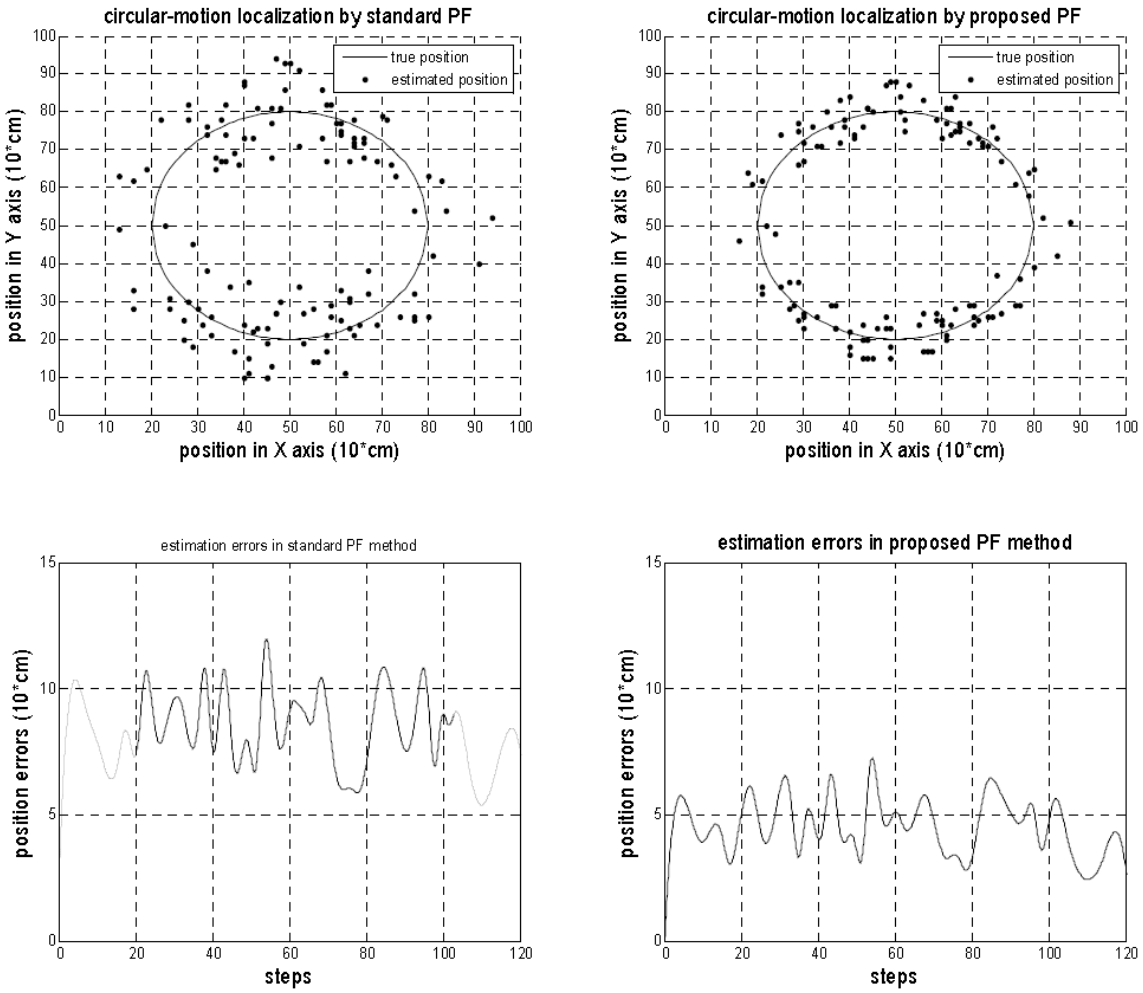


Figure 6.10: Localization with standard and proposed PF in circular trajectory

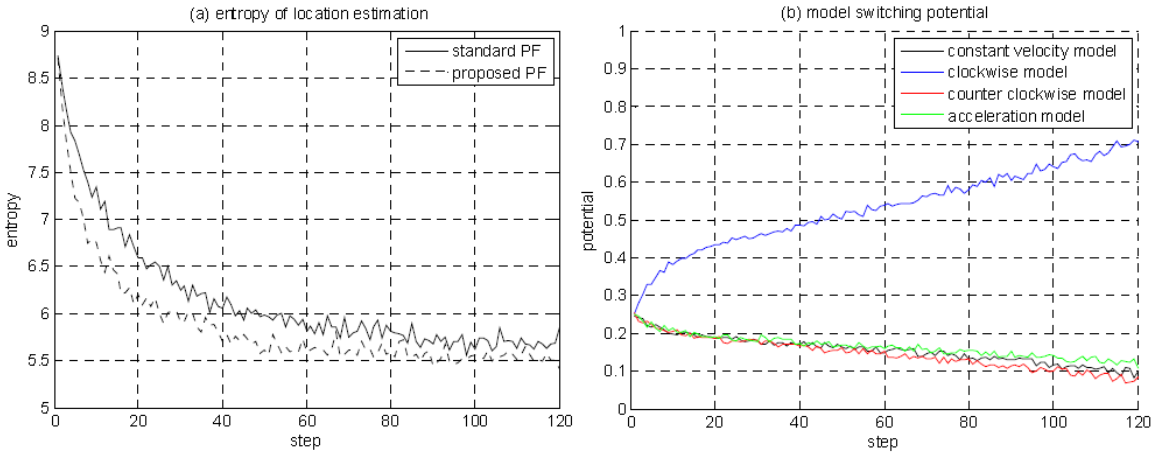


Figure 6.11: Entropy and model switching potential

6.4.4 Localization in Random Walking

In the more general case of random walking, the localization results are shown in Fig. 6.12 using both the standard and the proposed PF. In this scenario, there is no pre-defined trajectory and the robot is moving randomly with no strong correlation to a specific single motion model. This type of randomness complicates the system dynamic in that the target has no preference to a specific model. Therefore, all models should be considered all the time. Again, the results still show the superiority of the performance of the proposed PF that is able to closely track the random trajectory, especially at the moment when moving direction changes. Another feature is that the performance of the standard PF is also inferior compared with that in the pre-defined scenario, due to the presence of all models in the random-walking scenario instead of almost only one constant velocity model or clockwise/counter-clockwise model.

Again, the entropy and models switching potentials is shown in Fig. 6.13 (a) and (b). Since the random walking motion is the most unpredictable in all the three scenarios, the entropies of both standard and proposed PF have the largest convergence value, although the entropy of proposed PF still drops faster. Similar to Fig. 6.9 (a), the entropy of standard PF is again affected by the two turnings in the robot motion, whereas the proposed PF remains robust. As for the model switching potential, since there is no dominant model in the random walking, there is no obvious increasing or decreasing in the potential values of a specific model, as shown in Fig. 6.13 (b), except that the potential of clockwise model increases by a small volume due to the two turnings of the robot motion.

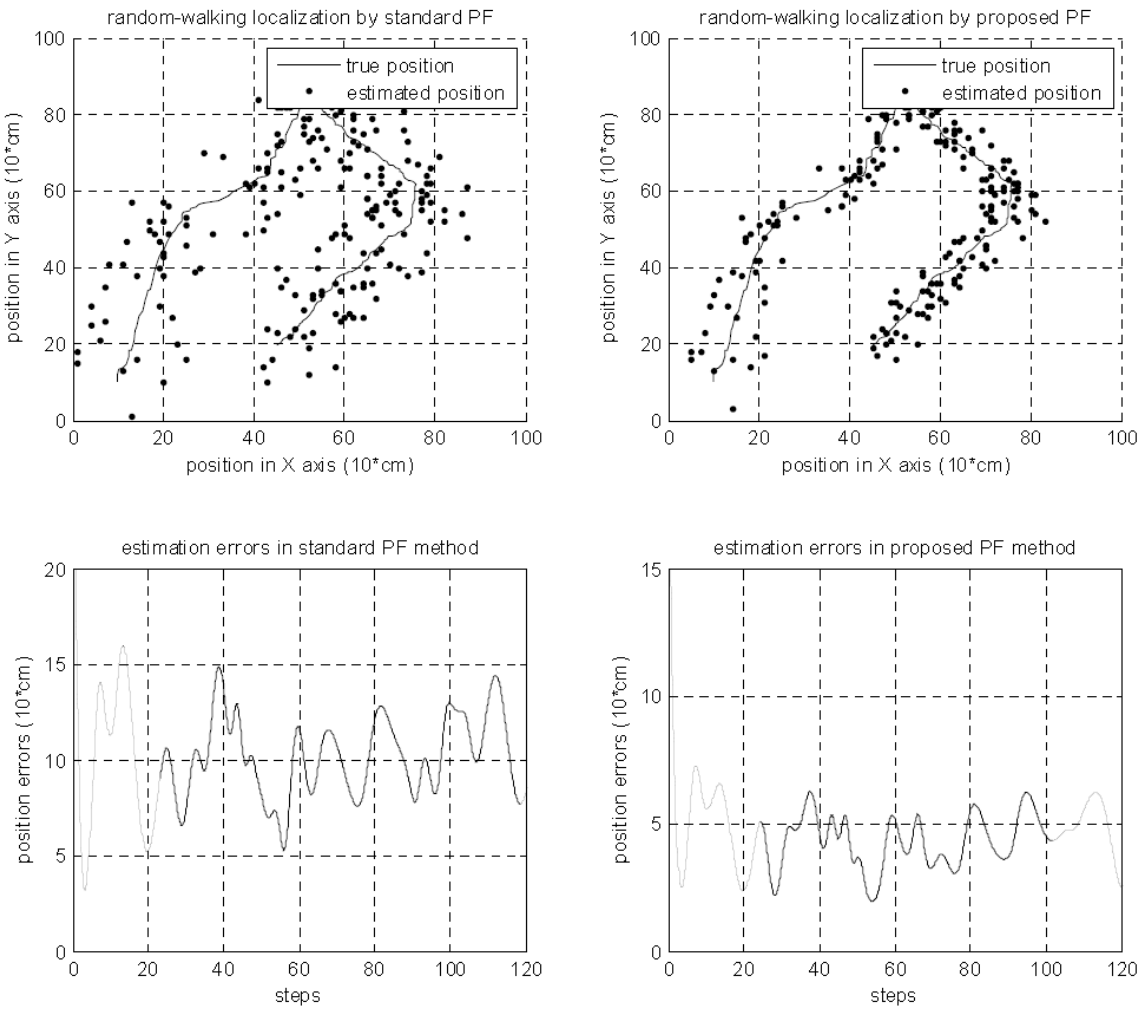


Figure 6.12: Localization with standard and proposed PF in random walking

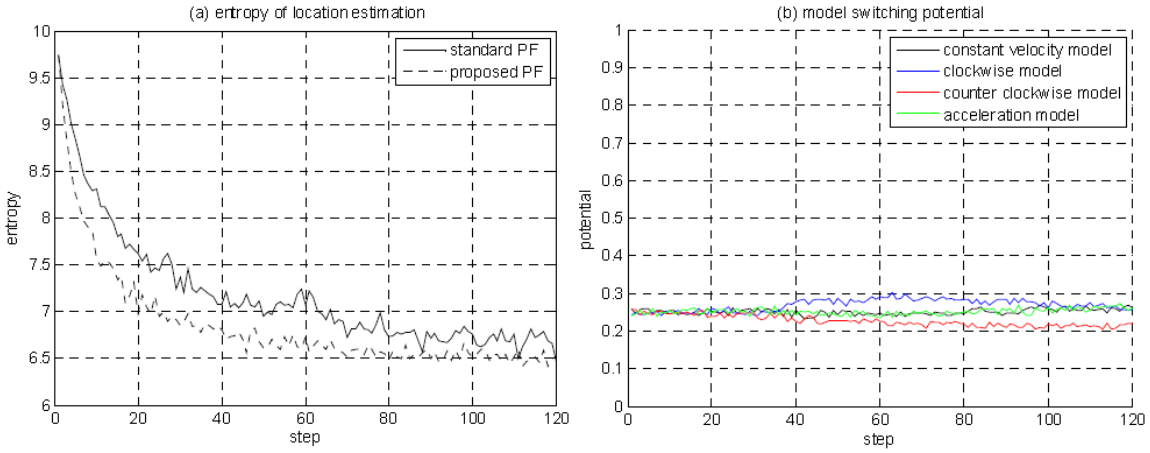


Figure 6.13: Entropy and model switching potential

6.5 Autonomous Path Planning

In this section, the proposed Monte Carlo POMDP algorithm is implemented on the mobile robot to carry out two path planning tasks in the indoor environment. One is the direct path planning in which the robot is required to find a path from the starting point to the destination in an environment with a pre-defined layout. This situation is for the normal autonomous task execution, e.g., transporting objects from one place to another without user intervention. The other task is for information gathering, in which the robot only has limited knowledge about the destination. Therefore, the robot has to actively gather necessary information, e.g., moving to the users at a known location to obtain commands or requests, and then planning the path accordingly.

6.5.1 Direct Path Planning

The proposed MC-POMDP is implemented for an normal path planning task. The basic setup of the experiment is firstly introduced along with the corresponding general MDP policy. Then the results of the traditional and proposed MC-POMDP are illustrated, compared, and analyzed.

6.5.1.1 Experiment Setup and MDP Policy

The basic layout of the ordinary path planning is an indoor environment with a magnitude of 10m-by-10m, which is decomposed into a 10-by-10 grid. The robot is initially placed at the lower left corner and the destination is the lower right corner. In MDP, the robot is assumed to have noiseless observations. However, there is 20% randomness rejected into four stochastic actions, which are going up, down, left and right.

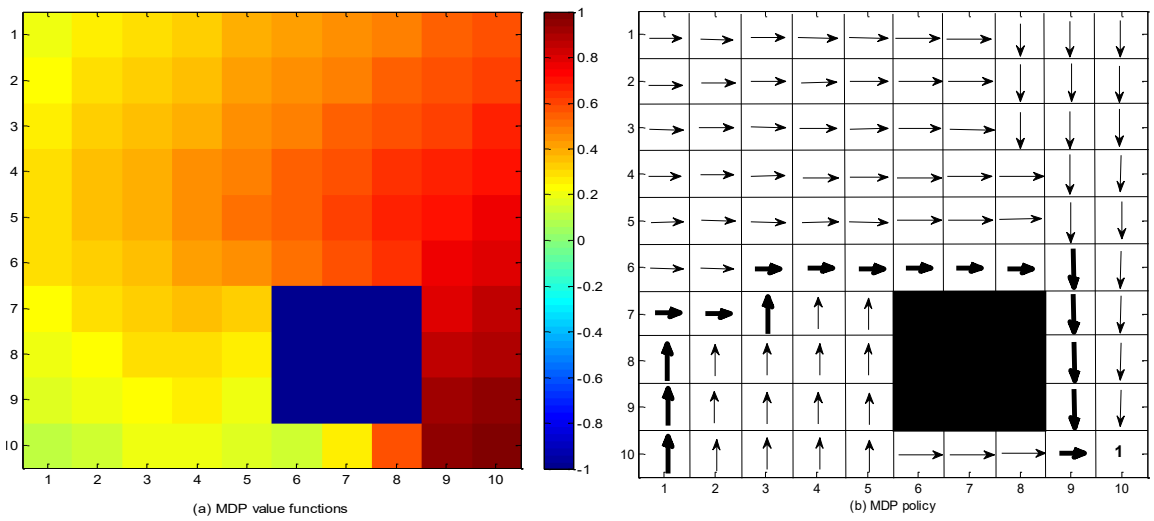


Figure 6.14: MDP value functions and policy of direct path planning

A pillar is close to the destination, which makes one upper wide corridor and one lower narrow corridor. The boundary of the room and the pillar needs to be avoided. This experiment is designed to validate the algorithm for some general assistive purposes, e.g., transporting a specific object for the users from one place to another.

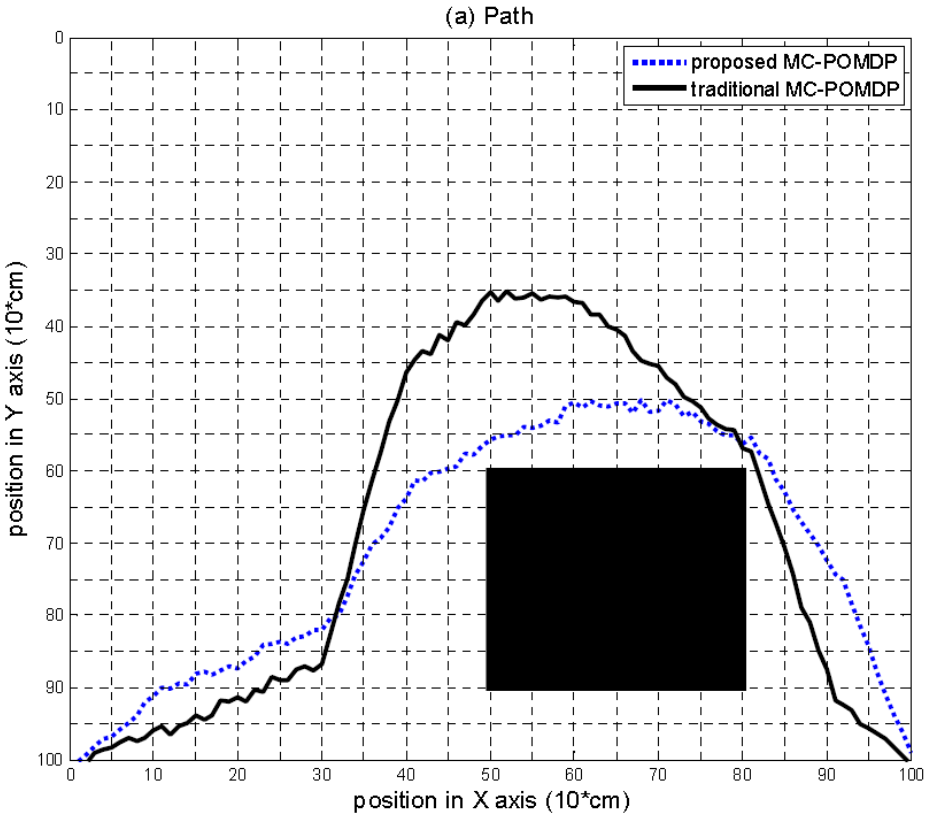
Fig. 6.14 (a) shows the layout of the indoor environment, as well as the color map of the MDP value functions with the range from -1 to 1, where only the stochasticity in the robot motion is considered and the observation is assumed to be noiseless. The destination at the lower right corner is assigned with the reward of 1 and the pillar is assigned with the reward of -1 since it is not accessible. Apparently, there are two possible paths in this situation, through the upper and the lower corridors, respectively. The difference of the two paths is due to the MDP policies iterating with different negative values of action reward $r(x,u)$. When the value of action reward is comparatively smaller, the robot needs to arrive at the destination as soon as possible in

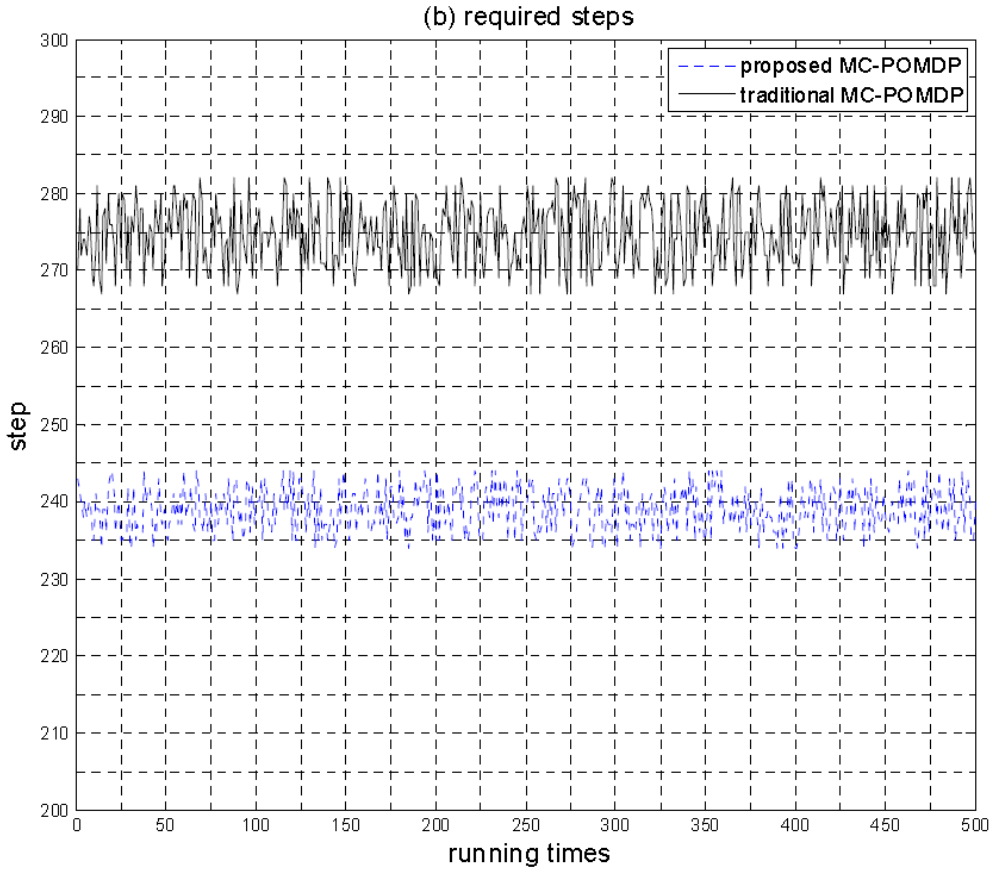
order to maximize the total rewards. Therefore, the corresponding MDP policy is to take the lower narrow corridor that is shorter but makes it more risky for the robot to collide with the wall and the pillar. On the other hand, when the reward is comparatively larger, the agent will take the upper path that takes more time but is safer. Fig. 6.14 (b) shows the policy with a comparatively larger reward $r(x,u) = -0.035$ and the agent prefers the upper path by taking the action of going up if at state (10,5). One possible path starting from the lower left corner is shown by the bold arrows, which takes 18 steps for the robot to arrive at the destination.

6.5.1.2 Results of Proposed MC-POMDP

The traditional and the proposed MC-POMDP are implemented on the above situation, respectively, assuming that both the action and sensor observation cannot be represented in an exact manner but only as a probability distribution. Moreover, the environment layout is further decomposed into a 100-by-100 grid for better illustration and performance of path planning.

Due to the effect of the stochasticity in action and observation, not all the MC-POMDP executions are able to result in a successful policy. A successful policy can be defined as the one that can lead the robot to the required destination through the path without colliding the wall or the pillar. Fig. 6.15 (a) shows two exemplary paths that are planned by the standard and the proposed MC-POMDP, respectively. It can be seen that although both paths are planned to avoid the narrow lower corridor and to lead the robot to the destination, the path of the proposed one can effectively maintain a good distance from the pillar for collision avoidance while still keeping a reasonable total path length.


(a) Path
proposed MC-POMDP
traditional MC-POMDP
position in Y axis (10*cm)
position in X axis (10*cm)
0
10
20
30
40
50
60
70
80
90
100


(b) required steps
proposed MC-POMDP
traditional MC-POMDP
step
running times
200
210
220
230
240
250
260
270
280
290
300
0
50
100
150
200
250
300
350
400
450
500

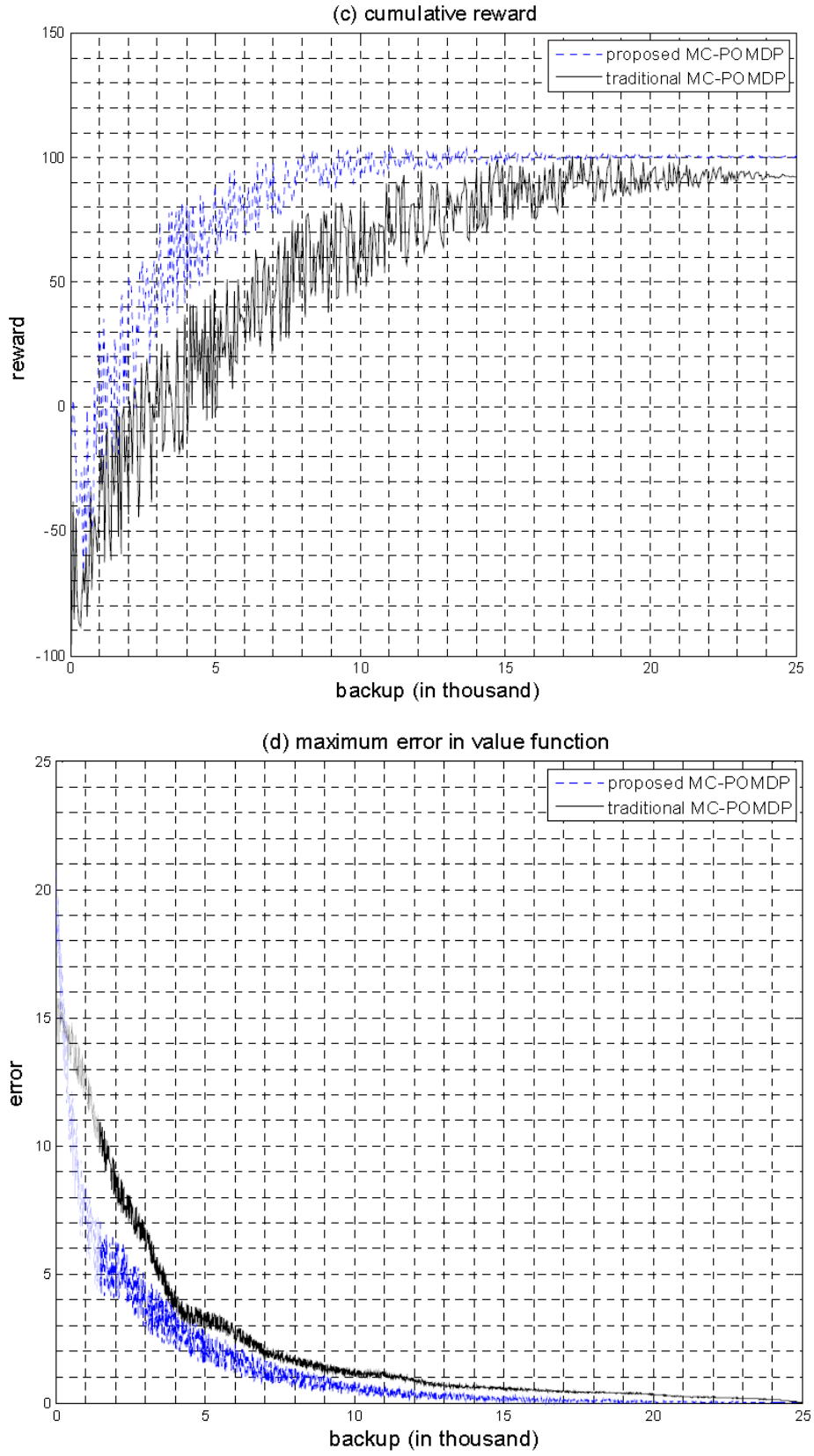


Figure 6.15: Direct path planning of the proposed and traditional MC-POMDP

On the other hand, the path of the traditional one moves unnecessarily farther from the pillar around the state (80,30) but closer to the pillar around (60,80). Moreover, the path of the proposed MC-POMDP takes fewer steps and a shorter route than the traditional one in which the robot frequently makes unnecessary detours. Fig. 6.15 (b) shows the general steps that the successful policy requires by planning the path for 500 repetitions. The average number of steps in the proposed MC-POMDP is around 12.3% less than that in the traditional one. Note that since the layout is more decomposed in this scenario, more steps are required than in the MDP case.

In the aspect of algorithm efficiency, Fig. 6.15 (c) shows the averaged cumulative rewards with respect to the number of backups in the two MC-POMDP algorithms. The results show that although the rewards in both algorithms can reach nearly equal convergence values, the reward collected in the proposed MC-POMDP converges after 15,000 backups, which is faster than the traditional one in which the reward only starts to converge after 20,000 backups. The errors of the planning cycles in the two MC-POMDP algorithms given at Fig. 6.15 (d), with respect to the number of backups (in thousand). The error is computed as the difference between the cumulative rewards of two consecutive iterations. This error has no unit since it only reflects the quality of the corresponding policy. It can be seen that the maximal error in the proposed MC-POMDP starts higher and drops faster than that in the traditional one, thus again proving the efficiency of the proposed MC-POMDP.

6.5.2 Information Gathering

The task of information gathering is carried out using the traditional and the proposed MC-POMDP in this section. Again, the basic setup and MDP policy is shown, followed by the results that are obtained using the two MC-POMDP algorithms.

6.5.2.1 Experiment Setup and MDP Policy

The environment is set up with the same layout and the robot is again initially placed at the lower left corner. However, there are two possible destinations, at the middle of the left wall or at the middle of the lower wall. In order to know which choice is the right one, the robot has to first go to the user or command hub at the upper right corner, and then heads to the required destination. This situation can be considered as the information gathering, e.g., coastal navigation, map exploration, etc. For the specific case in this section, in order to arrive at the ultimate destination, the robot has to make a detour or to move to places other than the destination first to gather some necessary information, e.g., the user's commands or the type of the object that is being transported, before heading to the ultimate destination. Fig. 6.16 shows the value functions and the associated policy of the stage I of resorting to the upper right corner to gather information. Fig. 6.17 shows the following stage II, where the robot takes either of these two paths after it is informed with the required location of destination.

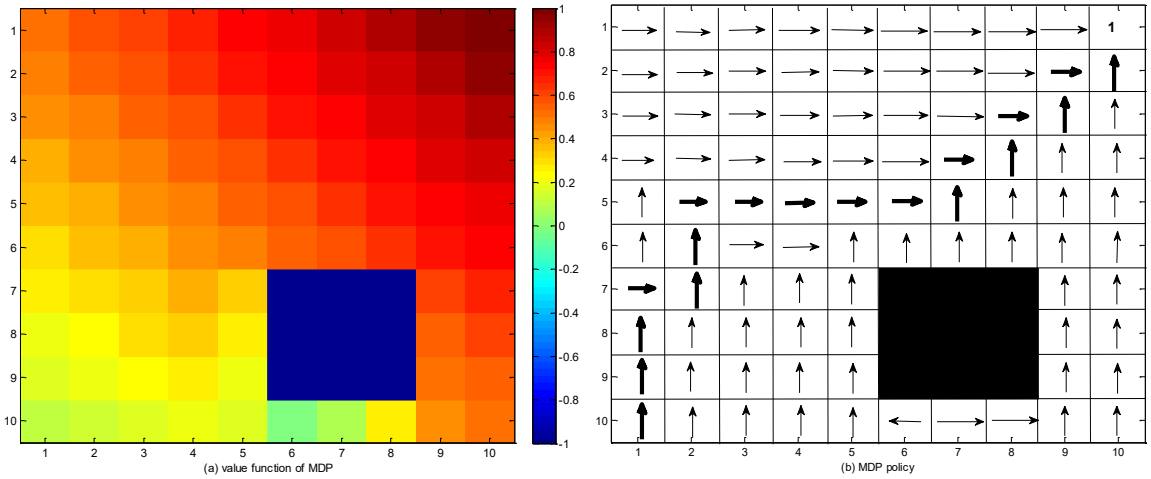


Figure 6.16: MDP value functions and policy of information gathering-stage I

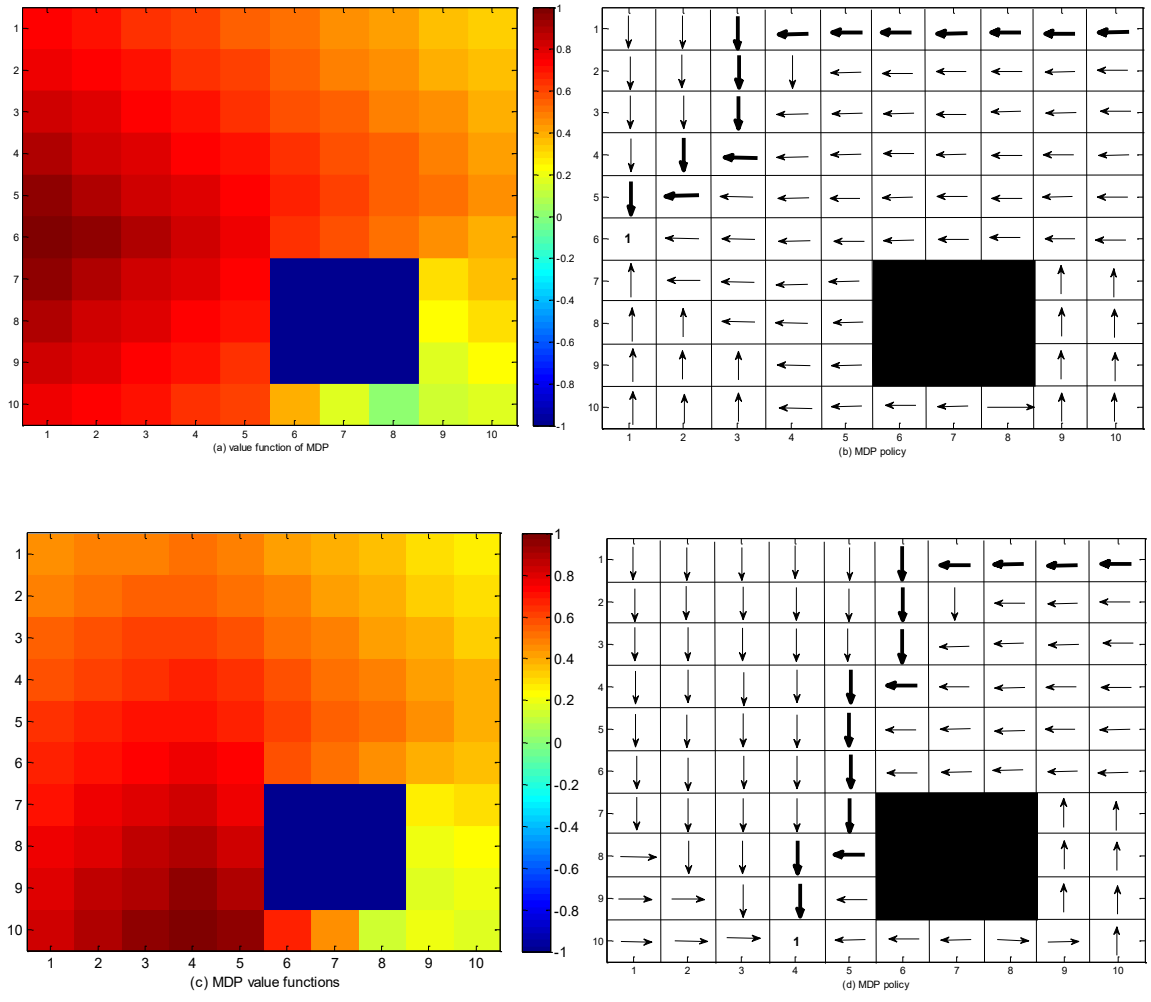


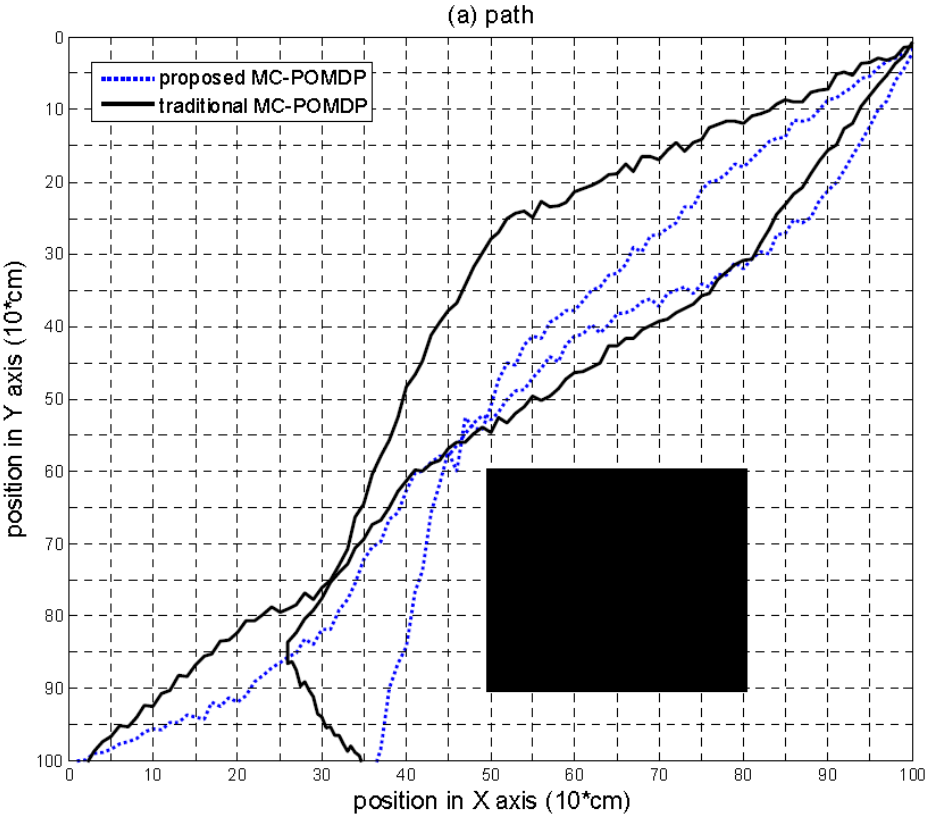
Figure 6.17: MDP value functions and policy of information gathering-stage II

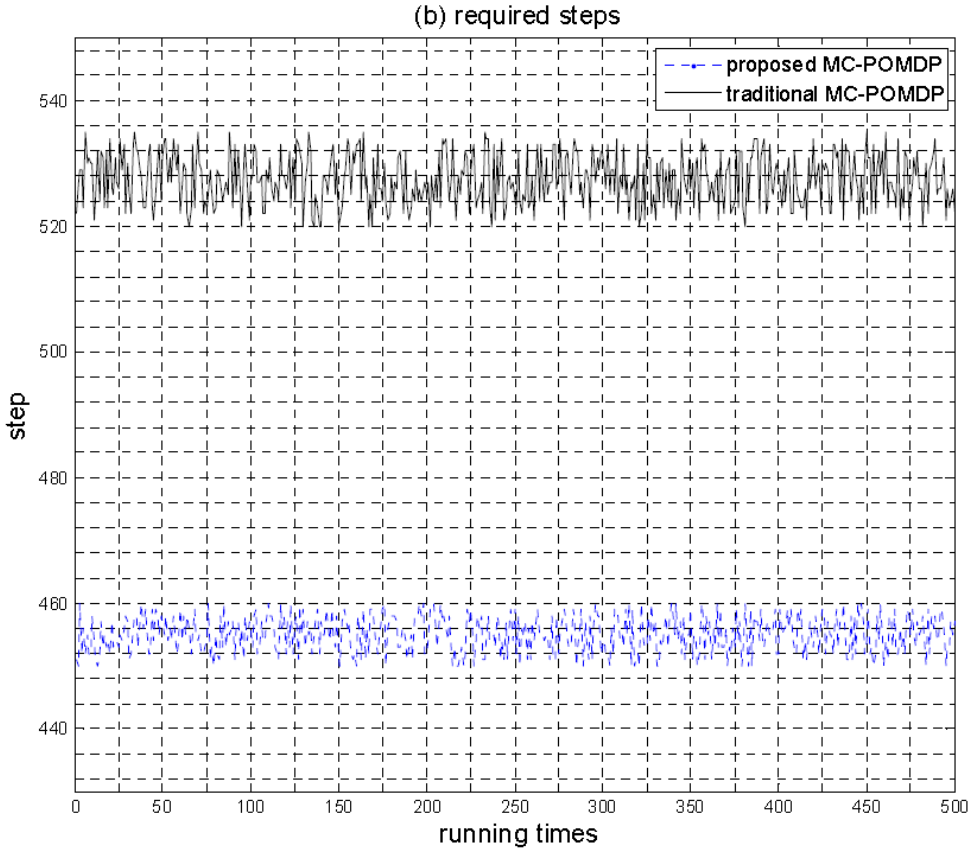
6.5.2.2 Results of Proposed MC-POMDP

The information gathering task is carried out by the proposed MC-POMDP along with the traditional one in the same layout and grid decomposition, assuming that the target destination is at the middle of the lower wall.

The exemplary successful two-stage paths from both two planning algorithms are illustrated in Fig. 6.18 (a). Same as the scenario of direct path planning, the trade-off between the total distance of path and the avoidance of obstacle needs to be considered first. As shown in the results, the proposed MC-POMDP is able to make a reasonable balance between the path distance and the avoidance of the wall and pillar. In other words, the obstacle can be proactively avoided while at the same time efficiency is considered, i.e., the distance and time the path takes are well treated. In the path of the traditional one, despite of the successful obstacle avoidance, some of the movements are unnecessary, e.g., farther away from the pillar in stage I and too close to the pillar in stage II. As a result, the path of the proposed MC-POMDP takes fewer steps, as shown in Fig. 6.18 (b). Note that since the scenario of information gathering tasks is consisted of two stages, more steps are required than in the previous direct path planning task.

Similarly, the averaged cumulative rewards and maximum error with respect to the number of backups are illustrated in Fig. 6.18 (c) and (d). Again, it is shown that the proposed MC-POMDP has a faster increasing cumulative rewards and faster dropping errors than the traditional MC-POMDP.


(a) path
proposed MC-POMDP
traditional MC-POMDP
position in Y axis (10*cm)
position in X axis (10*cm)
0
10
20
30
40
50
60
70
80
90
100


(b) required steps
proposed MC-POMDP
traditional MC-POMDP
step
running times
440
460
480
500
520
540
0
50
100
150
200
250
300
350
400
450
500

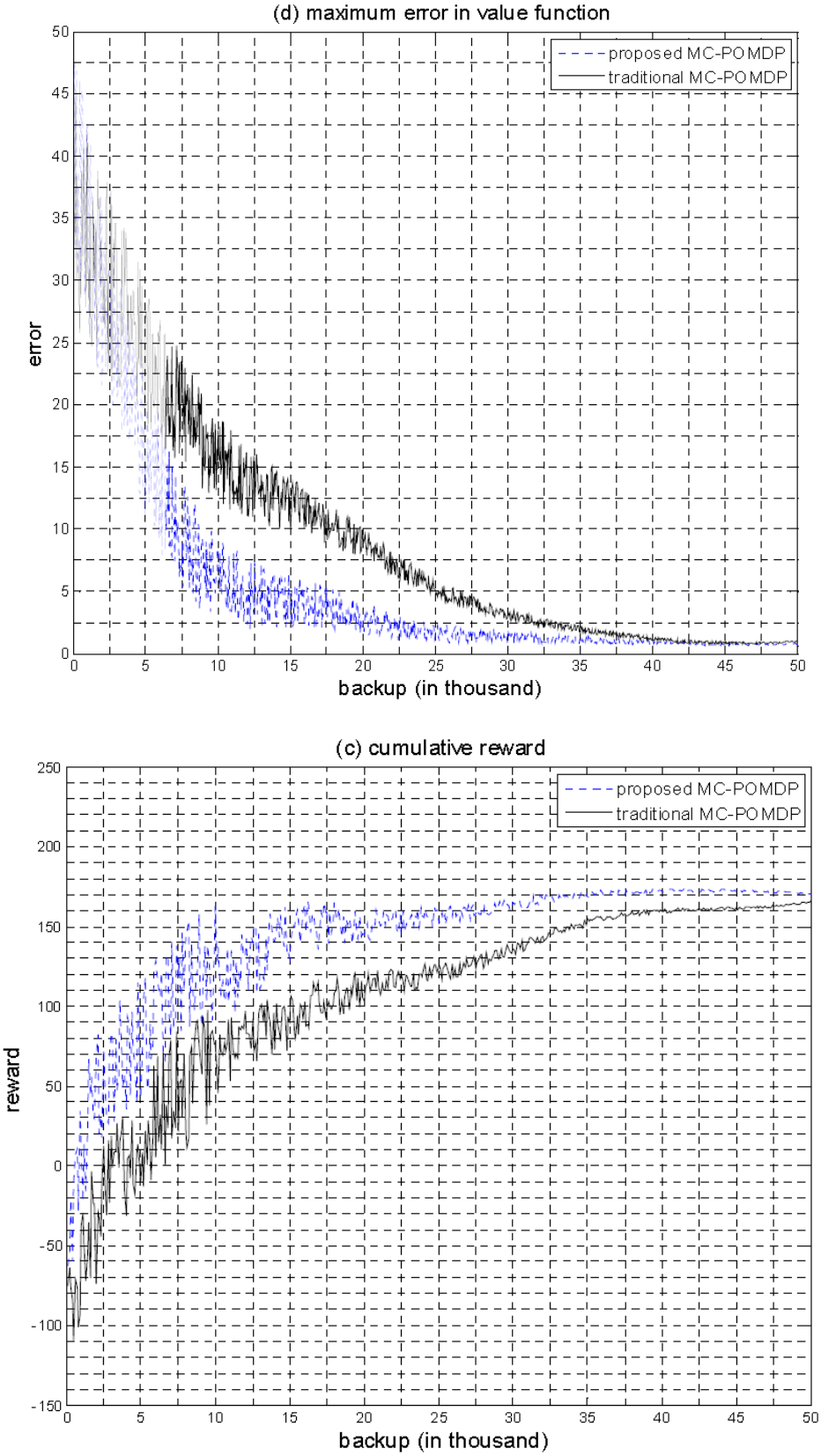


Figure 6.18: Information gathering of the proposed and traditional MC-POMDP

6.6 Summary and Conclusion

In this chapter, firstly the proposed particle filtering method based on adaptive multi-model, fuzzy logic velocity estimation and active observation selection was experimented and validated in the cases of human face tracking, human-machine tracking and following, and robot self-localization. The experimental results were analyzed and compared with those of the standard particle filter to prove the better performance of the proposed particle filtering algorithm. Then the proposed MC-POMDP was implemented for two path planning tasks in the indoor environment, namely the direct path planning and information gathering. It was shown from the experimental results that the proposed MC-POMDP performed better, compared with the traditional MC-POMDP, in terms of efficiency and accuracy.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this thesis, a prototype of the assistive robotic system was designed in accordance with the context-aware system architecture, and the corresponding context reasoning techniques based on the particle filtering method and Monte Carlo partially observable Markov decision process algorithm were developed. The prototyped robotic system was capable of tracking and following human users, performing self-localization, and planning path automatically for specific tasks in an indoor environment. The possible applications include supporting the elder people or people with disabilities, providing location-related services, monitoring user status, etc.

The main contributions of this thesis are summarized as follows:

1. A new layered and centralized structure of robotic context-aware system was proposed. This robotic context-aware system can be implemented in many assistance-oriented applications, e.g., to provide support to elder people with moving disability, to tracking and monitor patient's state and location.

2. A new taxonomy of context reasoning technique was proposed based on the whether the approaches were probability-based or non-probability-based. An extensive literature review based on this new taxonomy was presented.

3. A novel technique of proactive particle sampling was developed. This technique was based on the methods of adaptive multiple system state transition model and the fuzzy-logic-based velocity estimation to update the proposal distribution of the particle filter. The model transition matrix was updated in a way that model with higher potential was assigned with a higher transition probability. Meanwhile, each transition model was adjusted by the fuzzy-logic based velocity estimation to keep track of the most recent changes in target motion.

4. The active system observation approach was designed. In this approach, entropy was used as the uncertainty metric to evaluate each system observation. Instead of taking all the observation data, only those with higher quality were incorporated into the final system belief. As a result, the system can actively accommodate its sensing behaviors to alleviate the effects of uncertainties.

5. An enhanced version of path planning algorithm is proposed based on the Monte Carlo partially observable Markov decision process, namely the MC-POMDP. Specifically, the belief state space, or particle set, was augmented by incorporating the

index of entropy, which was further integrated into the Monte Carlo computation of the value function to alleviate the negative impact of the uncertainties in the set. The value function was represented by expectation of the distances between the particles from the query set and the reference sets by direct Monte Carlo sampling. The values of each action were represented as weights, which were then used to select the corresponding action for next update cycle.

7.2 Future Work

The proposed assistive robotic system and the corresponding context reasoning techniques are in the prototype stage. There is no doubt that more research works are still required for further development. Some of these works include:

1. In the current research work, the map of the environment is always assumed available, while in some cases this prerequisite is not always fulfilled. Next stage of research will consider the scenario in which the map is unavailable, and integrate the algorithm of the map exploration, or simultaneously localization and mapping (SLAM), into the system implementation.

2. The multi-sensor fusion method adopted in this thesis was performed by simple multiplication. In the following research, more advanced techniques, including the Dempster-Shafer theory, will be investigated and implemented in the process of data fusion to further reduce the effects of uncertainty.

3. In order to further verify the feasibility and effectiveness of the proposed system and algorithms, more experiments and more advanced scenarios need to be considered, e.g., the environment in which there are multiple and dynamic obstacles and/or users, or the task that requires more complicated and multiple stages of planning.

References

[1] “Ambient intelligence,” https://en.wikipedia.org/wiki/Ambient_intelligence.

[2] B. Schilit, N. Adams, and R. Want, “Context-aware computing applications,” *in Proc. Wksp. Mobile Comp. Sys. App.*, Santa Cruz, CA, 1994, pp. 85-90.

[3] A. K. Dey and G. D. Abowd. “Towards a Better Understanding of context and context-awareness,” *Technical Report GIT-GVU-99-22*, Georgia Institute of Technology, College of Computing, June 1999.

[4] M. Baldauf, S. Dustdar, and F. Rosenberg, “A survey on context-aware systems,” *Int. J. Ad Hoc and Ubiquitous Computing*, vol. 2, no. 4, pp. 263-277, 2007.

[5] H. Ailisto, P. Alahuhta, V. Haataja, V. Kyllnen, and M. Lindholm, “Structuring context aware applications: five-layer model and example case,” *in Proc. Ubicomp Wksp. Concepts and Models for Ubiquitous Computing*, Gteborg, UK, 2002.

[6] P. Korpipää, J. Mantyjarvi, J. Kela, H. Keranen, and E-J. Malm, “Managing context information in mobile devices,” *IEEE Pervasive Computing*, vol. 2, no. 3, pp.42–51, 2003.

[7] T. Gu, H.K. Pung, and D. Q. Zhang, “A middleware for building context-aware mobile services,” *in Proc. IEEE Conf. Vehicular Technology*, Milan, Italy, 2004, pp. 2656-2660.

[8] A. K. Dey, and G. D. Abowd, “The context toolkit: aiding the development of context-aware applications,” *in Proc. Wksp. Software Engineering for Wearable and Pervasive Computing*, Ireland, 2000.

[9] P. Fahy, and S. Clarke, “CASS–a middleware for mobile context-aware applications,” *in Proc. Wksp. on Context Awareness, MobiSys*, 2004.

[10] M. Roman, C. Hess, R. Cerqueira, A. Ranganathan, R.H. Campbell, and K. Nahrstedt, “A middleware infrastructure for active spaces,” *IEEE Pervasive Computing*, vol. 1, no. 4, pp.74–83, 2002.

[11] H. Chen, “An intelligent broker architecture for pervasive context-aware systems,” *PhD dissertation*, University of Maryland, Baltimore County, ML, 2004.

[12] G. Lacey, and S. MacNamara, “Context-aware shared control of a robot mobility aid for the elderly blind,” *Int. J. Robotics Research*, vol. 19, no.11, pp. 1054-1065, 2000.

[13] T. Khan, S. Gao, and W. S. Lee, “Context-aware robot navigation based on sensor association rules,” *Int. Conf. Convergence and Hybrid Information Technology*, 2009, pp.217-220.

[14] J. Röning, and J. Riekki, “Mobile robots: part of a smart environment,” *Robotics and Machine Perception, SPIE's Int. Technical Group Newsletter*, vol. 9, no. 1, March 2000.

[15] F. Yuan, M. Hanheide, and G. Sagerer, “Spatial context-aware person following for a domestic robot,” *Int. Wksp. Cognition for Technical Systems*, Munich, Germany, 2008.

[16] C.-S. Hong, K.-W. Lee, H.-S. Kim, and H. Kim, “The Context-Awareness for the Network-based Intelligent Robot Services, Advances in Service Robotics,” *Ho Seok Ahn (Ed.), InTech*, Available: http://www.intechopen.com/articles/show/title/the_context-awareness_for_the_network-based_intelligent_robot_services.

[17] H. Kim, Y.-J. Cho, and S.-R. Oh, “CAMUS: A middleware supporting context-aware services for network-based robots,” *IEEE Wksp. Advanced Robotics and Its Social Impacts*, Nagoya, Japan, 2005, pp. 237-242.

[18] H. Kim, M. Kim, K.-W. Lee, Y.-H. Suh, J. Cho, and Y.-J. Cho, "Context-aware server framework for network-based service robots," *SICE-ICASE Int. Joint Conference*, Busan, Korea, Oct., 2006.

[19] "Composite Capability/Preference Profiles (CC/PP): Structure and Vocabularies 1.0," Available: https://www.w3.org/TR/CCPP-struct-vocab/

[20] G. Castelli, A. Rossi, M. Mamei, F. Zambonelli, "A simple model and infrastructure for context-aware browsing of the world," *in Proc. 5th IEEE Conference on Pervasive Computing and Communications,* IEEE Computer Society, 2007.

[21] K. Henricksen, J. Indulska, "Developing context-aware pervasive computing applications: Models and approach," *Pervasive and Mobile Computing*, vol. 2, no. 1, pp. 37-64, 2006.

[22] A. Frank, "Tiers of ontology and consistency constraints in geographical information systems," *International Journal of Geographical Information Science*, vol. 15, no. 7, pp. 667-678, 2001.

[23] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, P. F. Patel-Schneider (Eds.), *The Description Logic Handbook: Theory, Implementation, and Applications*, Cambridge University Press, 2003.

[24] I. Horrocks, P. F. Patel-Schneider, F. van Harmelen, "From SHIQ and RDF to OWL: The making of a web ontology language," *Journal of Web Semantics*, vol. 1, no. 1, pp. 7-26, 2003.

[25] J. Ye, S. Dobson, and S. McKeever, "Situation identification techniques in pervasive computing: A review," *Pervasive and mobile computing*, vol. 8, no. 1, pp. 36-66, 2012.

[26] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan and D. Riboni , "A survey of context modelling and reasoning techniques," *Pervasive Mob. Comput.*, vol. 6, pp.161-180, 2010.

[27] A. Bikakis, T. Patkos, G. Antoniou, D. Plexousaki, M. M, F. A and A. E, “A survey of semantics-based approaches for context reasoning in ambient intelligence,” *Workshops Ambient Intelligence*, 2007.

[28] K. Henricksen, J. Indulska, A. Rakotonirainy, “Developing Context-aware Pervasive Computing Applications: Models and Approach,” *Journal of Pervasive and Mobile Computing*, vol. 2, no. 1, pp. 37-64, Feb. 2006.

[29] A. Ranganathan, J. Al-Muhtadi, R. H. Campbell, “Reasoning about uncertain contexts in pervasive computing environments,” *IEEE Pervasive Computing*, vol. 3, no. 2, pp. 62-70, 2004.

[30] S. W. Loke, “Representing and reasoning with situations for context-aware pervasive computing: A logic programming perspective,” *Knowledge Eng. Rev.*, vol. 19, no. 3, pp.213 -233, 2005.

[31] S. W. Loke, “Logic programming for context-aware pervasive computing: Languages support, characterizing situations, and integration with the web,” *Web Intelligence*, pp. 44-50, 2004.

[32] M. Bhatt, S. Loke, “Modelling dynamic spatial systems in the situation calculus,” *Spatial Cognition and Computation 8 (1&2)*, pp. 86–130, 2008.

[33] A. Kalyan and V.S. Gopalan, “Hybrid context model based on multilevel situation theory and ontology for contact centers,” *in Proc. Third IEEE International Conference on Pervasive Computing and Communications Workshops*, 2005, pp. 3–7.

[34] J. C. Augusto, J. Liu, P. McCullagh, H. Wang, J.-B. Yang, “Management of uncertainty and spatio-temporal aspects for monitoring and diagnosis in a smart home,” *International Journal of Computational Intelligence Systems*, vol. 1, no. 4, pp. 361–378, 2008.

[35] B. Gottfried, H. W. Guesgen, S. Hübner, “Spatiotemporal reasoning for smart homes,” *Designing Smart Homes 4008*, pp. 16–34, 2006.

[36] J. F. Allen, “Maintaining knowledge about temporal intervals,” *Communications of the ACM*, vol. 26, no. 11, pp. 832-843, 1983.

[37] M. Vilain, and H. A. Kautz, “Constraint Propagation Algorithms for Temporal Reasoning,” *in Aaai*, vol. 86, pp. 377-382. 1986.

[38] D. A. Randell, Z. Cui, and A. G. Cohn, “A spatial logic based on regions and connection,” *KR 92*, pp. 165-176, 1992.

[39] M. J. Egenhofer, and R. D. Franzosa, “Point-set topological spatial relations,” *International Journal of Geographical Information System*, vol. 5, no. 2, pp. 161-174, 1991.

[40] A. U. Frank, “Qualitative spatial reasoning about distances and directions in geographic space,” *Journal of Visual Languages & Computing*, vol. 3, no. 4, pp. 343-371, 1992.

[41] H. W. Guesgen, *Spatial reasoning based on Allen's temporal logic*, Berkeley: International Computer Science Institute, 1989.

[42] N. Yamada, K. Sakamoto, G. Kunito, Y. Isoda, K. Yamazaki, S. Tanaka, “Applying ontology and probabilistic model to human activity recognition from surrounding things,” *Information and Media Technologies*, vol. 2, no. 4, pp. 1286-1297, 2007.

[43] T. Gu, H. K. Pung, D. Q. Zhang, “A service-oriented middleware for building context-aware services,” *Journal of Network and Computer Applications*, vol. 28, no. 1, pp. 1–18, 2005.

[44] D. Riboni, C. Bettini, “Context-aware activity recognition through a combination of ontological and statistical reasoning,” *in Proc. international conference on Ubiquitous Intelligence and Computing*, Springer, Berlin, Heidelberg, 2009, pp. 39–53.

[45] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, D. Riboni, “A survey of context modelling and reasoning techniques,” *Pervasive and Mobile Computing*, vol. 6, no. 2, pp. 161–180, 2010.

[46] L. Chen, C. Nugent, M. Mulvenna, D. Finlay, X. Hong, “Semantic smart homes: towards knowledge rich assisted living environments,” *Intelligent Patient Management*, pp. 279–296, 2009.

[47] P. Brezillon, L. Pasquier, and JC Pomerol, “Reasoning with contextual graphs,” *European Journal of Operational Research*, vol. 136, no. 2, pp. 290-298, 2002.

[48] T. V. Nguyen, L. Wontaek, H. A. Nguyen, and D. Choi, “Context awareness framework based on contextual graph,” *in Wireless and Optical Communications Networks, 2008. WOCN'08. 5th IFIP International Conference on*, pp. 1-5, 2008.

[49] M. LARABA and P. BREZILLON., “Using Contextual Graphs for Supporting Qualitative Simulation Explanation,”

[50] JV. Nguyen, B. C. Becker, and A. J. Gonzalez, “Contextual Graphs for a Real-World Decision Support System,” *in FLAIRS Conference*, pp. 643-648. 2006.

[51] H. J. Zimmermann, “An application-oriented view of modeling uncertainty,” *European Journal of Operational Research*, vol. 122, no. 2, pp. 190–198, 2000.

[52] P. D. Haghighi, S. Krishnaswamy, A. Zaslavsky, M. M. Gaber, “Reasoning about context in uncertain pervasive computing environments,” *in EuroSSC’08: Proceedings of the 3rd European Conference on Smart Sensing and Context,* Springer-Verlag, Berlin, Heidelberg, 2008, pp. 112–125.

[53] J. Ye, S. McKeever, L. Coyle, S. Neely, S. Dobson, “Resolving uncertainty in context integration and abstraction,” *in: ICPS’08: Proceedings of the International Conference on Pervasive Services*, ACM, Sorrento, Italy, July 2008, pp. 131–140

[54] C. B. Anagnostopoulos, Y. Ntarladimas, S. Hadjiefthymiades, “Situational computing: an innovative architecture with imprecise reasoning,” *Journal of System and Software*, vol. 80, no. 12, pp. 1993–2014, 2007.

[55] D. Lowd, P. Domingos, “Naive bayes models for probability estimation,” *in ICML’05: Proceedings of the 22nd International Conference on Machine Learning*, *ACM*, Bonn, Germany, 2005, pp. 529–536.

[56] E. M. Tapia, S. S. Intille, K. Larson, "Activity recognition in the home using simple and ubiquitous sensors," *in Pervasive'04: Proceedings of the international conference on Pervasive Computing*, 2004, pp. 158–175.

[57] M. Mühlenbrock, O. Brdiczka, D. Snowdon, J.-L. Meunier, "Learning to detect user activity and availability from a variety of sensor data," *in PERCOM'04:Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications, IEEE Computer Society*, Orlando, Florida, March 2004, pp. 13–22.

[58] T. van Kasteren, B. Krose, "Bayesian activity recognition in residence for elders," *in IE'07: Proceedings of the third IET International Conference on Intelligent Environments*, September 2007, pp. 209–212.

[59] J. Ye, L. Coyle, S. Dobson, P. Nixon, "Using situation lattices to model and reason about context," *in MRC 2007: Proceedings of the Workshop on modeling and reasoning on context (coexist with CONTEXT'07)*, Roskilde, Denmark, August 2007, pp. 1–12.

[60] T. Gu, H. K. Pung, D. Q. Zhang, "A Bayesian approach for dealing with uncertain contexts," *in Proceedings of Advances in Pervasive Computing, coexisted with Pervasive'04. Austrian Computer Society*, April 2004, pp. 205–210.

[61] W. Abdelsalam, Y. Ebrahim, "A Bayesian-networks-based approach for managing uncertainty in location-tracking," *in Proceedings of the Canadian Conference on Electrical and Computer Engineering*, vol. 4., May 2004, pp. 2205–2208.

[62] B. A. Truong, Y.-K. Lee, S.-Y. Lee, "Modeling uncertainty in context-aware computing," *in ICIS'05: Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science*, 2005, pp. 676–681.

[63] K. Sentz, S. Ferson, "Combination of evidence in dempster-shafer theory," *Tech. Rep. SAND 2002-0835*, Sandia National Laboratories, 2002.

[64] S. McKeever, J. Ye, L. Coyle, S. Dobson, "Using dempster-shafer theory of evidence for situation inference," *in P.M. Barnaghi, K. Moessner, M. Presser, S. Meissner, EuroSSC 2009: Proceedings of the 4th European Conference on Smart Sensing*

and Context, in: Lecture Notes in Computer Science, vol. 5741, Springer, 2009, pp. 149–162.

[65] X. Hong, C. Nugent, M. Mulvenna, S. McClean, B. Scotney, S. Devlin, "*Evidential fusion of sensor data for activity recognition in smart homes*," Pervasive and Mobile Computing, vol. 5, pp. 236–252, 2009.

[66] S. McKeever, J. Ye, L. Coyle, S. Dobson, "A context quality model to support transparent reasoning with uncertain context," *in K. Rothermel, D. Fritsch, W. Blochinger, F. Drr (Eds.), QuaCon'09: Proceedings of the 1st International Workshop on Quality of Context, in: Lecture Notes in Computer Science*, vol. 5786, Springer, 2009, pp. 65–75.

[67] S. Mckeever, J. Ye, L. Coyle, C. Bleakley, S. Dobson, "Activity recognition using temporal evidence theory," *Journal of Ambient Intelligence and Smart Environment*, vol. 2, no. 3, pp. 253–269, 2010.

[68] D. Zhang, J. Cao, J. Zhou, M. Guo, "Extended dempster-shafer theory in context reasoning for ubiquitous computing environments," *in CSE'09: Proceedings of the 2009 International Conference on Computational Science and Engineering, IEEE Computer Society*, Washington, DC, USA, 2009, pp. 205–212

[69] L. Bao, S.S. Intille, "Activity recognition from user-annotated acceleration data," *in Pervasive'04: Proceedings of the Second International Conference on Pervasive Computing*, Vienna, Austria, April 2004, pp. 1–17

[70] D. M. Karantonis, M. R. Narayanan, M. Mathie, N. H. Lovell, B. G. Celler, "Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring," *IEEE Transactions on Information Technology in Biomedicine*, vol. 10, no. 1, pp. 156–167, 2006.

[71] M. J. Mathie, B. G. Celler, N. H. Lovell, A.C.F. Coster, "Classification of basic daily movements using a triaxial accelerometer," *Medical and Biological Engineering and Computing,* vol. 42, no. 5, pp. 679–687, 2004.

[72] S. N. Patel, T. Robertson, J. A. Kientz, M. S. Reynolds, G. D. Abowd, “At the flick of a switch: detecting and classifying unique electrical events on the residential power line,” *in UbiComp’07: Proceedings of the 9th International Conference on Ubiquitous Computing, Springer-Verlag*, Berlin, Heidelberg, 2007, pp. 271–288.

[73] “Weka. Weka 3: Data Mining Software in Java (2007),” Available: http://www.cs.waikato.ac.nz/ml/weka/.

[74] T. Kanda, D. F. Glas, M. Shiomi, H. Ishiguro, N. Hagita, “Who will be the customer?: a social robot that anticipates people’s behavior from their trajectories,” *in UbiComp’08: Proceedings of the 10th International Conference on Ubiquitous Computing*, *ACM*, Seoul, Korea, 2008, pp. 380–389.

[75] F. Yuan, M. Hanheide, and G. Sagerer, “Spatial context-aware person following for a domestic robot,” *Int. Wksp. Cognition for Technical Systems*, Munich, Germany, 2008.

[76] M. Isard and A. Blake, “CONDENSATION - conditional density propagation for visual tracking,” *Int. J. Comput. Vis*., vol. 29, no. 1, pp. 5–28, 1998.

[77] Y. Zhai, M. Yeary, S. Cheng and N. Kehtarnavaz, “An object tracking algorithm based on multiple model particle filtering with state partitioning,” *IEEE Trans. Instrum. Meas*., vol. 58, no. 5, pp. 1797-1089, May. 2009.

[78] L. Jing, P. Vadakkepat, “Improved particle filter in sensor fusion for tracking random moving object”, *Proc IEEE Conf. Instrum. and Meas. Tech*., vol. 1, 2004, pp. 476-481.

[79] N. Bouaynaya, Q. Wei and D. Schonfeld, “An online motion-based particle filter for head tracking applications,” *Proc. IEEE ICASSP*, Mar. 2005, vol. 2, pp.225–228.

[80] W. Dargie, “The role of probabilistic schemes in multisensor context-awareness,” *In Pervasive Computing and Communications Workshops, 2007. PerCom Workshops' 07. Fifth Annual IEEE International Conference on*, pp. 27-32. IEEE, 2007

[81] M. Greiffenhagen, D. Comaniciu, H. Niemann and V. Ramesh, “Design, analysis, and engineering of video monitoring systems: An approach and a case study,” *Proc. IEEE,* vol. 89, no. 10, pp. 1498–1517, Oct. 2001.

[82] P. Perez, J. Vermaak and A. Blake, “Data fusion for visual tracking with particles,” *Proc. IEEE*, vol. 92, no. 3, pp. 495–513, Mar. 2004.

[83] A. Chakraborty, “A distributed architecture for mobile, location dependent applications,” Master thesis, MIT, Cambridge, MA, 2000.

[84] B. Ristic, S. Arulampalam, and N. Gordon, “Beyond the Kalman Filter: Particle Filters for Tracking Applications,” *Norwood, MA: Artech House*, Feb. 2004.

[85] S. McGinnity and G. Irwin, “Multiple model bootstrap filter for maneuvering target tracking,” *IEEE Trans. Aerosp. Electron. Syst*., vol. 36, no. 3, pp. 1006–1012, Jul. 2000.

[86] R. Guo, Z. Qin, X. Li, and J. Chen, “Interacting multiple model particle type filtering approaches to ground target tracking,” *J. Comput*., vol. 3, no. 7, pp. 23–30, Jul. 2008.

[87] Y. Rui and Y. Chen, “Better proposal distributions: Objects tracking using unscented particle filter,” *in Proc. IEEE Conf. Comput. Vis. Pattern Recog*., Dec. 2001, pp. 786–793.

[88] Y. Zhai, M. Yeary, J. Noyer, J. Havlicek, S. Nemati, and P. Lavin, “Visual target tracking using improved and computationally efficient particle filtering,” *in Proc. IEEE Conf. Image Process.*, Oct. 2006, pp. 1757–1760

[89] K. Wang and X. P. Liu, “Visual object tracking based on filtering methods,” *in Proc. Instrumentation and Measurement Technology Conference (I^2MTC),* Hangzhou, China, May 2011, pp. 1–6.

[90] R. Kavanagh, “Number-theoretic approach to optimum velocity decoding given quantized position information,” *IEEE Transactions on Instrumentation and Measurement*, vol. 50, pp. 1270 –1276, Oct. 2001.

[91] F. Janabi-Sharifi, V. Hayward, and C. . J. Chen, “Discrete-time adaptive windowing for velocity estimation,” *IEEE Transactions on Control Systems Technology*, vol. 8, no. 6, pp. 1003–1009, 2000.

[92] L. Bascetta, G. Magnani, and P. Rocco, “Velocity estimation: Assessing the performance of non-model-based techniques,” *IEEE Transactions on Control Systems Technology*, vol. 17, no. 2, pp. 424–433, 2009.

[93] F. Yusivar, D. Hamada, K. Uchida, S. Wakao, and T. Onuki, “A new method of motor speed estimation using fuzzy logic algorithm,” *in Electric Machines and Drives, 1999. International Conference IEMD ’99*, pp. 278 –280, May 1999.

[94] H. Liu, H. Darabi, P. Banerjee and J. Liu, “Survey of wireless indoor positioning techniques and systems,” *IEEE Transactions on Systems, Man, and Cybernetics Part C*, vol. 37, no. 6, pp. 1067–1080, Nov 2007.

[95] D. MacKay, *Information theory, inference and learning algorithms*, Cambridge university press, 2003.

[96] Dieter Fox, Wolfram Burgard, Sebastian Thrun, and Armin B. Cremers. “Position estimation for mobile robots in dynamic environments,” *In AAAI/IAAI*, pp. 983-988, 1998.

[97] K. J. Astrom, “Optimal control of Markov decision process with incomplete state estimation,” *J. Math. Anal. Appl.*, vol. 10, pp. 174-205, 1965.

[98] R. E. Korf, “Real-time heuristic search: New results,” *Proc. Nat. Conf. Artif. Intell.*, 1988, pp. 139-144.

[99] R. E. Bellman, *Dynamic programming*, Princeton university press, 1957.

[100] M. L. Littman, A. R. Cassandra and L. P. Kaelbling, “Learning policies for partially observable environments: Scaling up,” *in Proc. Twelfth Int’l Conf. Machine Learning*, 1995, pp. 362-370.

[101] N. Roy, W. Burgard, D. Fox and S. Thrun, "Coastal navigation: Robot navigation under uncertainty in dynamic environments," *in Proc. IEEE Conf. Robotics Automation (ICRA)*, 1999, pp. 35-40.

[102] N. Roy, G. Gordon and S. Thrun, "Finding approximate POMDP solutions through belief compression," *J. Artif. Intell. Res.*, vol. 23, pp. 1–40, Jul. 2005.

[103] S. Thrun, "Monte Carlo POMDP," *in Advances in Neural Information Processing Systems*, vol. 12, pp. 1064-1070, 2000, MIT Press.

[104] *Performance PeopleBot™ Operations Manual*. MOBILEROBOTS Company, 2007.

[105] P. Viola, and M. Jones, "Rapid object detection using a boosted cascade of simple features," *in Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2001, vol. 1, pp. 511-518.

[106] I. Transafety, "Study compares older and younger pedestrian walking speeds," Available: http://www.usroads.com/journals/p/rej/9710/re971001.htm.

[107] K. Wang and X. P. Liu, "Particle Filtering Enhanced Human Tracking on Context-Aware Robotic System," *Proc. Int'l Symp. Haptic Audio-Visual Environ. Games*, Oct. 2013, pp. 92-97.

[108] K. Wang and X. P. Liu, "Particle Filtering-based tracking and localization on context-aware robotic system," *Proc. Int'l Conf. Computer Science & Education*, Vancouver, BC, 2014, pp. 229-234.

[109] K. Wang and X. P. Liu, "Adaptive multi-model and entropy-based localization on context-aware robotic system," *IEEE Int'l Conf. Systems, Man, and Cybernetics (SMC)*, San Diego, CA, 2014, pp. 3755-3760.