Spring 2015

An Opportunistic Service Oriented Approach for Robot Search

Dan Xie
Computer Sciences

AN OPPORTUNISTIC SERVICE ORIENTED APPROACH FOR ROBOT SEARCH

A Dissertation Presented

by

DAN XIE

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2015

School of Computer Science

AN OPPORTUNISTIC SERVICE ORIENTED APPROACH FOR ROBOT SEARCH

A Dissertation Presented

by

DAN XIE

Approved as to style and content by:

To my family and friends.

ABSTRACT

AN OPPORTUNISTIC SERVICE ORIENTED APPROACH FOR ROBOT SEARCH

FEBRUARY 2015

DAN XIE

B.Sc., BEIJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

M.Sc., BEIJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

M.Sc., UNIVERSITY OF MASSACHUSETTS AMHERST

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Allen Hanson and Professor Roderic A. Grupen

Health care for the elderly poses a major challenge as the baby boomer generation ages. Part of the solution is to develop technology using sensor networks and service robotics to increase the length of time that an elder can remain at home. Since moderate immobility and memory impairment are common as people age, a major problem for the elderly is locating and retrieving frequently used “common” objects such as keys, cellphones, books, etc. However, for robots to assist people while they search for objects, they must possess the ability to interact with the human client, complex client-side environments and heterogeneous sensorimotor resources. Given this complexity, the traditional approach of developing particular control strategies in a top-down manner is not suitable.

In this dissertation an opportunistic service-oriented approach is presented to address the robot search problem in residential eldercare. With the presented approach, a hierarchy of search strategies is developed in a bottom-up manner from passive object detection and retrieval performed by embedded camera sensors to context-aware cooperative search performed by a human-robot team. By opportunistically employing available sensorimotor resources, the robotic application achieves increased search performance, and has the flexibility to balance between performance goals and resource constraints. To evaluate the proposed approach, I describe several experiments with a robot-sensor network that includes the UMass uBot-5, Pan-Tilt-Zoom cameras and wireless sensors. The results of these experiments suggest that the robot search application based on the proposed approach can lead to efficient search performance and great flexibility in resource-constrained environments.

TABLE OF CONTENTS

Page

ABSTRACT . . . . **v**

LIST OF TABLES . . . . **xii**

LIST OF FIGURES . . . . **xiii**

CHAPTER

1. INTRODUCTION **1**

1.1 Motivation 1
1.2 Approach 3
1.3 Contributions 8
1.4 Document Overview 9

2. LITERATURE REVIEW **11**

2.1 Software Architecture for Robotics Applications 11

2.1.1 Distributed Object Architecture (DOA) 12
2.1.2 Component Based Architecture (CBA) 12
2.1.3 Service-Oriented Architecture (SOA) 13
2.1.4 Cloud Robotics (CR) 14

2.2 Residential Assisted Living 15
2.3 Object Search Systems 17

2.3.1 Object Search Systems 17
2.3.2 Heuristic Approaches for Object Search 19
2.3.3 Cooperative Search Strategies 20
2.3.4 Human-Robot Cooperative Search 21

2.4 Object Recognition with Embedded Camera 22

2.4.1 Multi-tier sensor network: 22

2.4.2 Object recognition: 23
2.4.3 Semi-supervised clustering: 23

2.5 Summary 23

3. OPPORTUNISTIC SERVICE-ORIENTED ARCHITECTURE 25

3.1 Specifications 25

3.1.1 Emergency (vital sign) Monitoring. 25
3.1.2 ADL (Activities of Daily Living) Analysis 26
3.1.3 Assistive Services 26

3.2 Challenges 27

3.2.1 Heterogeneous hardware/software components. 27
3.2.2 Multi-tasking and resource contention. 28
3.2.3 Reliability. 28

3.3 Architectural Overview 28

3.3.1 Service Oriented Architecture 29
3.3.2 Opportunistic Service Oriented Approach (SOA) 30

4. TRACKING-BASED OBJECT RETRIEVAL 35

4.1 Dual-Camera Structure 35

4.1.1 Hardware Components 36
4.1.2 Overview of System Operation 37

4.2 Object Detection 39

4.2.1 Object detection in ultra-low-power tier 40
4.2.2 Sensor Triggering and ROO mapping 40

4.3 Semi-supervised object recognition 41

4.3.1 Feature Extraction 42
4.3.2 Object Recognition Procedure 43

4.3.2.1 Recognize by Color Histogram 43
4.3.2.2 Combine the Color and SIFT Features 44

4.3.3 Constrained Cluster Updating 46

4.3.3.1 Constraint Definition 46
4.3.3.2 Constrained k-means 47

4.4 Object Retrieval in Proxy Node 50

4.4.1 Event Database 50
4.4.2 Tag-based object retrieval 51

4.5 System implementation 52

4.5.1 Hardware implementation 52
4.5.2 Software environment 53

4.6 Experimental results 53

4.6.1 Energy Cost of Object Detection 54
4.6.2 Accuracy of object mapping 55
4.6.3 Comparison between color and SIFT 56
4.6.4 Benefits of using constraints 57
4.6.5 System performance on object retrieval 58

4.6.5.1 Impact of Tagging Threshold 59

4.7 Summary 60

5. SINGLE-AGENT ACTIVE SEARCH 62

5.1 Bayesian Searching Problem 62
5.2 Search Strategy 63

5.2.1 Subtask Plan 64
5.2.2 Subtask Move 66
5.2.3 Subtask Observation 66
5.2.4 Localization 66

5.3 Evaluations 67

5.3.1 *LPR-1* Scenario 67
5.3.2 Human Data in LRP1 Scenario 67
5.3.3 Simulated Experiments 69
5.3.4 Real-World Experiments 71

5.4 Summary 72

6. MULTI-AGENT COOPERATIVE SEARCH 75

6.1 Message Sharing-Based Cooperative Search Schema 76

6.2 How to Compensate for the Limitations of Teammates 79

6.2.1 Compensating for Viewpoint Limitations 80
6.2.2 Compensating For Limitations on Observation Reliability 82

6.3 Evaluations 84

6.3.1 Performance of Basic Message Sharing Strategy 84
6.3.2 Benefit of Considering Teammates' Limitations 86

6.3.2.1 Cooperative Search Strategies with Viewpoint Limitations 87
6.3.2.2 Cooperative Search Strategies with Predicted Observation Reliability 87

6.3.3 Overall Search Performance of the Robot Team 88

6.4 Summary 90

7. HUMAN-ROBOT COOPERATIVE SEARCH 92

7.1 Search Schema with Human Teammates 94
7.2 Human Peer Modeling 95

7.2.1 Dwell State Detection 96
7.2.2 Activity Modeling with HMM 97

7.2.2.1 Hidden Markov Model (HMM) 97
7.2.2.2 Activity Modeling and Training 99
7.2.2.3 Decoding 100
7.2.2.4 Prediction 101

7.2.3 Robot Search Strategy 102

7.3 Interface Design for Implicit Cooperation 103
7.4 Guided Search with Human Gesture 105
7.5 Learning from Demonstration 107
7.6 Evaluation 108

7.6.1 Simulated Experiments 108

7.6.1.1 Efficiency of Human-Robot Cooperative Search 109
7.6.1.2 Interface Design Experiments 111

7.6.2 Real World Experiments 115

7.6.2.1 Activity-Based Human-Robot Joint Search 115

7.6.2.2 Gesture-Based Human-Robot Joint Search 115

7.7 Summary 116

8. CONCLUSIONS AND FUTURE WORK 119

8.1 Conclusions 119
8.2 Directions For Future Work 120

BIBLIOGRAPHY 121

LIST OF TABLES

Table **Page**

4.1 Database in the proxy node 51

4.2 Object detection error analysis. Errors are measured in pixels. 56

4.3 Comparison of recognition methods (The power consumption during object recognition is 681mW) 57

4.4 Improvements from using constraints (The power consumption during object recognition is 681mW) 58

4.5 Object retrieval performance (The tagging threshold TH_{tag} is fixed to 0.3) 59

7.1 Average time cost with different search strategies. (sec) 111

LIST OF FIGURES

Figure **Page**

1.1 Robots are being used to search and rescue in emergency response scenarios. (a) The Nexi robot developed by MIT Media Lab and LPR (Laboratory for Perceptual Robotics) UMass Amherst is searching for victims in a post-disaster rescue scenario. (b) The PackBot developed by iRobot Corp was used in response to 9/11, and helped explore overheating nuclear plants in Japan after earthquake in 2011. (c) Other search and rescue robots. 4

1.2 Robots are being used to search for house-hold objects in assisted-living scenarios. The robots must possess the ability to interact with complex and dynamic environment and human clients to achieve successful search. (a) The uBot-5 robot [36] was searching for objects. (b) Developed by Department of Computer Science, University of British Columbia, Curious George is a visual search robot, which won the robot league of the 2007 Semantic Robot Vision Challenge (SRVC) 5

1.3 Heterogeneous hardware resources are employed in the robot assisted-living system 6

1.4 The hierarchy of the proposed object search application. 7

3.1 Problem domains and the supported applications in a residential assisted-living system. 27

3.2 System infrastracture consists of sensor, motors, kinematic devices, computational services and system interfaces. Local and remote (cloud) servics are considered. Local services support robotic devices and functional units. RN denotes ROS Node and DNS denotes the Distributed Naming Service in ROS. 30

3.3 Relationship between the hardware/computational services and the applications in the assisted-living system 31

3.4 A road map for developing search strategies in multiple levels. 32

4.1 The dual-camera sensor node 37

4.2 Software and hardware architectures of the tracking-based object retrieval application 38

4.3 An example of Field-of-View and ROO mapping: (a) Control points found by SIFT. (b) The ROO mapping result. 42

4.4 An example of clustering under constraints. Color is used to identify different clusters. To represent clusters, two dominating dimensions are calculated by Principal Components Analysis and used as x, y coordinates. 49

4.5 Power consumption analysis. (a) Effect of the sampling interval. (b) Effect of object detection interval. 54

4.6 Deployment of the dual-camera sensor nodes. 59

4.7 Effect of the tagging threshold TH_{tag}. (a) Retrieval rate. (b) ROO transmitted. 60

5.1 The search strategy for a single agent. 64

5.2 *LPR-1* testing space. 68

5.3 Trajectories (orange and blue dots) and dwell locations (black circles) when human subjects search. (a) Search for object *BOOK*, (b) Search for object *CLAMP*. Shadowed areas represent obstacles (desks and shelves). 69

5.4 Human search performance when searching for the BOOK and the CLAMP 70

5.5 (a) The physical uBot-5 and the search space. (b) The simulated uBot-5 and the simulated environment. 71

5.6 Single robot search strategies. *S1: Expected Information Gain. S2: Expected Information Gain + Travel Cost.* 72

5.7 uBot-5 robot. (a) An 11-inch MacBook Air computer is mounted on the back of the robot and used as the central processing computer. (b) An ASUS Xion Pro sensor and a Skyetek M9 UHF RFID reader are mounted on the robot. 73

5.8 Performance comparison of the human subjects and the uBot-5 when searching the *BOOK*. 74

6.1 PTZ camera agents performing object search. 76

6.2 PTZ camera agents performing human tracking and object search simultaneously, with connection failure to one searching camera. 77

6.3 The cooperative search strategy for a single agent. 78

6.4 Local PDF update for search agents. 79

6.5 An example of viewpoint limitations: (a) without knowledge of reachability constraints, the overall number of steps to achieve complete coverage is 3; (b) if agent B can compensate for viewpoint limitations in agent A, the number of steps can be reduced to 2. 81

6.6 An example of observation reliability in joint search. (a) Not considering the limitation on observation reliability, the overall number of steps to achieve complete coverage is 4. (b) The agent B compensates the limitation on observation reliability of the agent A, the overall number of steps to achieve complete coverage is 3. 83

6.7 The system console when the camera array is searching. The left four windows show the camera views, and the four coloured areas in the map represent the camera's field-of-view that is projected on the floor. 85

6.8 Search performance of the camera array for cooperative and non-cooperative coordinate modes. 86

6.9 Comparison of search strategies with viewpoint limitation. *S1: Non-cooperative search; S2: Cooperative search; S3: Cooperative search, complement teammate's viewpoint limitations.* 88

6.10 Standard deviation (a) S1; (b) S2; (c) S3. 89

6.11 Comparison of search strategies with limitation on observation reliability. 90

6.12 Standard deviation (a) S1; (b) S2; (c) S3. 91

6.13 Comparison of cooperative and single-agent search strategies (searching for the *BOOK*) 91

7.1 A scenario of robot assisted search in a home environment 94

7.2 Cooperative search strategy for a single agent when working with human teammates 95

7.3 Clustering result and human dwell states for BOOK search. 97

7.4 A graphical representation of Hidden Markov Model with three states. 98

7.5 Posterior probability distribution describing the location of the target object update for a search agent: (a) before observation. (b) after current observation 103

7.6 Skeleton tracking in the camera view of the robot. 106

7.7 Human data (a) searching for a book; (b) searching for a screwdriver 109

7.8 Search efficiency (for BOOK) 112

7.9 Search efficiency (for SCREWDRIVER) 113

7.10 Accuracy of search activity prediction 114

7.11 Comparison in search efficiency 114

7.12 Performance comparison of search strategies. The left bar represents human search performance. The right bar represents the performance of the human-robot cooperative search strategy with human activity modeling 116

7.13 Performance comparison of search strategies. The left bar represents human search performance. The right bar represents the performance of the human-robot cooperative search strategy with human gesture guidance. 117

CHAPTER 1

INTRODUCTION

1.1 Motivation

Technological progress over the past decade suggests that we are closer to personal robots that can assist human clients in the activities of daily living than ever before. However, a number of challenges have to be addressed before robotics really impacts the lives of a vast number of people in need. First, for robots to operate and assist people, they must possess the ability to interact with the complex client-side environment and the human clients. The traditional approach of developing particular control strategies in a top-down manner is not suitable for multi-task applications due to the complexity and the cost. For wide spread use and commercialization, personal robots must be packaged in a manner that pushes price/performance downward. Second, another challenge is the recruitment of heterogenous sensorimotor resources from which services can be derived in the client-side environment. Adequate organization of hardware and computational resources to form and deliver services is a current research interest.

This dissertation argues that the key to address these problems is to construct robotic applications in a distributed and service-oriented approach. One of the technical breakthroughs that puts these goals in reach is the availability of middleware to handle messages between modular sensory and motor resources (e.g., ROS –Robot Operating System [46], Yarp –Yet Another Robot Platform [43], MRDS –Microsoft Robotics Developer Studio [54], NDDS [44], etc). Human-centric applications require dynamic adaptation of services, so in this dissertation I propose using an **oppor-**

tunistic service-oriented approach to design robot applications for assisted-living. The proposed architecture builds applications using a hierarchy of services and tries to satisfy performance requirements at the lowest level possible where minimum resources are engaged and flexibility is limited. If performance specifications cannot be satisfied, then control percolates up the hierarchy to achieve better performance. The applications also have the flexibility to collapse to lower tiers and to release resources to other applications when resource contention happens.

In this dissertation, the proposed opportunistic service-oriented approach is used to address the robot search problem, where the robotic system assists a human client to find common objects in a residential environment. Despite some relevant research, robot search in unstructured environments is still an open problem. To achieve effective search, the robot must acquire skills that model the dynamic of the environment, recruit additional resources as necessary, and cooperate with the client who may themselves contribute resources.

There has been considerable recent interest in robot search and rescue in emergency response. The unprecedented number and scales of natural and human-induced disasters in the past decade has urged the emergency search and rescue community around the world to seek newer, more effective equipment to enhance their efficiency. Emergency respond robots have been proposed to help search and rescue survivors in collapsed or compromised structures, mining accidents, hostage situations, and explosions. Some examples are illustrated in Figure 1.1. A good overview of rescue robotics is in Disaster Robotics by Robin Murphy [80].

It is likely that applying search and rescue technologeis for residential assisted living settings can help to extend the period of time that elders can live independently. (some examples are shown in Figure 1.2). As the baby boomer generation ages over the next decade, health care for the elderly poses a major economic and pratical challenge. In addition to monitoring for illnesses and potentially life-threatening situations, an equally important challenge in home healthcare for the elderly is providing assistance in day-to-day life. Since moderate immobility and memory impairment are common as people age, a major problem for the elderly is locating and retrieving frequently used "common" objects such as keys, cellphones, books, etc [116, 88]. Impaired performance on this everyday task can lead to safety issues and potentially to institutionalization. However, assisted-living systems today are still unable to reliably find objects. Compared to the systems in emergency response, assisted living robots should have a high degree of autonomy and learning capability to adapt to the behavior of the human subjects living in the residential environment. Furthermore, an important aspect of assistive technology is that technology delivered into residential environment must adapt to special needs, lifestyles, preferences, residential geometry and environment. Therefore, it is important to study the object search problem and to develop effective and efficient approaches for automated object search for residential assited living.

1.2 Approach

An Opportunistic Service-Oriented approach for the design of robot search applications for residential assisted-living is proposed. The Service-Oriented approach I proposed organizes the behavior of distributed sensorimotor and computational resources to support many applications in many situations. As shown in Figure 1.3, heterogeneous sensorimotor resources are employed. The major components of the system include an array of 4 Pan/Tilt/Zoom (PTZ) cameras and mobile robots (uBot-5). Microsoft Kinect sensors, embedded wireless camera sensors and RFID (Radio-Frequency Identification) readers are also deployed to sense the environment. Primitive computational services including object recognition, feature detection and tracking. These services can migrate in the distributed architecture with the client. Client applications (object/event/activity recognition, tracking) are realized using

(a)

(b)

(c)

Figure 1.1. Robots are being used to search and rescue in emergency response scenarios. (a) The Nexi robot developed by MIT Media Lab and LPR (Laboratory for Perceptual Robotics) UMass Amherst is searching for victims in a post-disaster rescue scenario. (b) The PackBot developed by iRobot Corp was used in response to 9/11, and helped explore overheating nuclear plants in Japan after earthquake in 2011. (c) Other search and rescue robots.

compositions of primitive services informed by prior models of the context dependent interaction patterns

Robotic systems using service-oriented architectures are not new [40, 64]. The cost of a particular configuration in the SOA is defined in terms of the time a resource commitment required. The goal is to respond to the run-time context in a manner that produces the best quality result per unit cost. In a multi-tasking system that deals with resource contention and node/communication failure, services must be capable of re-configuring to optimize the expected net performance of the entire system. This dissertation advocates composing services from re-usable primitives to accumulate skills that make services robust. When executing, the application starts from running in the lowest most primitive tier, and attempts to employ more resources when there

(a) (b)

Figure 1.2. Robots are being used to search for house-hold objects in assisted-living scenarios. The robots must possess the ability to interact with complex and dynamic environment and human clients to achieve successful search. (a) The uBot-5 robot [36] was searching for objects. (b) Developed by Department of Computer Science, University of British Columbia, Curious George is a visual search robot, which won the robot league of the 2007 Semantic Robot Vision Challenge (SRVC)

is an opportunity to achieve better performance. The applications also have the flexibility to collapse to lower tiers and to release resources to other applications when resource contention happens.

In this thesis, robot search strategies are developed in an incremental manner. A bottom-up approach is adopted to develop a hierarchy of search strategies from passive object detection and retrieval performed by embedded camera sensors to context-aware cooperative search performed by a human-robot team (see Figure 1.4). This approach starts from the smallest use of resources, and incrementally and opportunistically recruits more sensorimotor resources for higher performance. Novel approaches are proposed in this dissertation to address the important questions in robot search.

Question 1: How to achieve energy-efficient yet effective search with the minimum resource in the lowest tier so that a persistent and basic functionality is maintained?

To achieve energy-efficieny, a tracking-based object retrieval application (Tier-1 in Figure 1.4) is adopted, which doesn't employ any motor resource. By passively

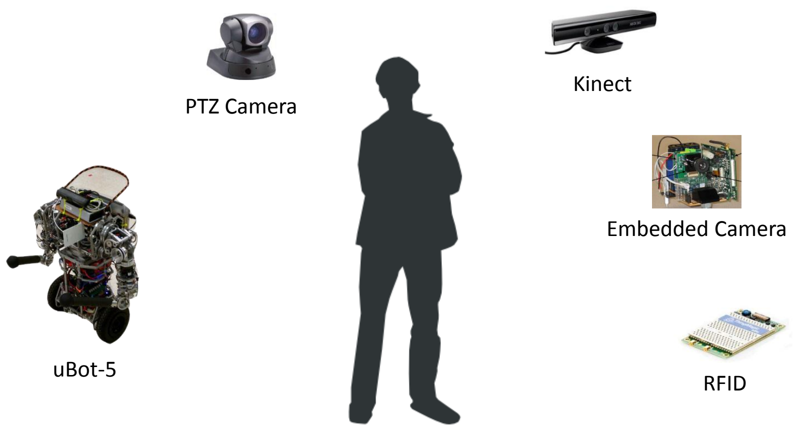


Figure 1.3. Heterogeneous hardware resources are employed in the robot assisted-living system

monitoring the environment and recording the movement of the target objects, the tracking-based method can provide object location information to the user on demand. An energy-efficient object detection and recognition algorithm is developed and the design of a dual-camera platform is presented.

Question 2: How does a robot interact with complex client-side environment to achieve efficient search?

Tracking-based object retrieval suffers from incomplete coverage and may not be sufficient in some cases. A mobile robot performing active search is required. A probabilistic framework is introduced to address the single agent search problem. Consider the case when the system recruits resources to assist a human client who has lost his/her book. The Probability Distribution Function (PDF), $Pr(x|book)$, where $x \in \mathbf{R}^3$ is the location of the book, is given a priori and updated iteratively as observations are accumulated. Search agents select the next place to look in $Pr(x|book)$ in order to maximize the expectation of finding the target object.

Machine learning is used to acquire expert domain knowledge regarding a single client in a complex client-side environment. The way to model the expert domain knowledge is twofold. (i) The human is the teacher. A Learning-by-Demonstration

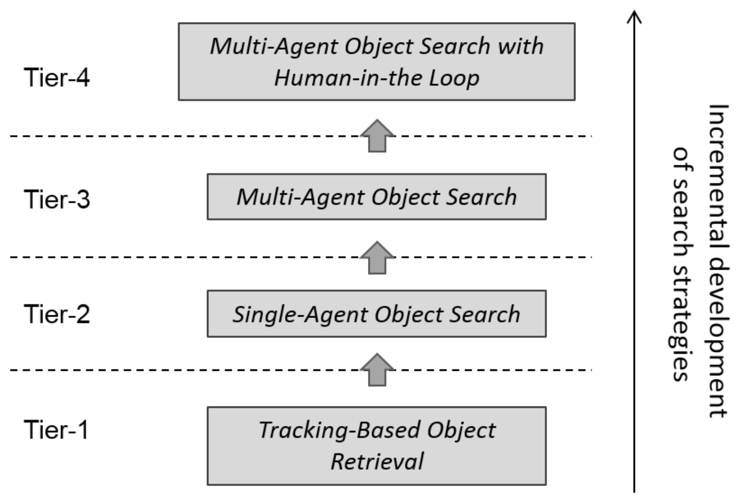

Tier-4
Multi-Agent Object Search with Human-in-the Loop
Tier-3
Multi-Agent Object Search
Tier-2
Single-Agent Object Search
Tier-1
Tracking-Based Object Retrieval
Incremental development of search strategies

Figure 1.4. The hierarchy of the proposed object search application.

(LbD) approach is used to break the programming barrier of the robot. By observing and analyzing human search activity, the robot mimics how the human searches to form an initial prior PDF as the initial problem domain knowledge, which is discussed in Chapter 7. (ii) Subsequently, contextual information is used to refine the initial domain knowledge. This dissertation investigates (in Chapter 5) how context such as the detection history and human activities influence the estimation of the prior PDF of the target and the use of this information to improve the search efficiency. To the best of our knowledge, no work has been done that uses human activity information to help reduce the search space of the robot agent.

Question 3: How does a search robot interact with other search agents and the human teammates to achieve efficient cooperation?

In search operations, a team of intelligent agents can provide a robust solution with greater efficiency than can be achieved by single agents, even with comparatively superior mobility and sensors. Some synchronous approaches have been proposed for multi-agent search [26, 37]. However, for a multi-tasking system that must deal with resource contention, pre-emption, and node/network failures, a decentralized and asynchronous version is preferred. A cooperative search strategy is proposed that employs inter-agent messages to share posterior distributions that summarize where search agents have already looked and where they are likely to look next to coordinate multiple asynchronous search agents.

It is important for the robot system to cooperate with the human clients to perform search. The cooperative search strategy is extended to model the human as an agent teammate without explicit message transmission mode. The robot agents infer the current state and intention of the human peer using a probabilistic model of human search activity acquired in the learning session. By inferring human search states, independent search activity of the robot agents are scheduled to search goals that complements the human client's activities to achieve efficient cooperation.

1.3 Contributions

The contributions of this dissertation can be elaborated as follows:

An Opportunistic Service Oriented approach is presented to address the robot search problem in residential eldercare. The resulting system provides a systematic illustration on how services can be composed in a incremental manner for accomplishing complex tasks in a resource-constrained environment. Experimental results are presented to show how increasingly efficient search performance is acquired as more sensorimotor resources are employed. The proposed framework can be applied to more than robot search applications, e.g., mobility aids and ADL (Activity of Daily Living) analysis.

In the design of robot search system, novel approaches and algorithms are presented to make it a robust system for interacting with both the complexity of the environment and the human clients.

Firstly, a **probabilistic search** strategy for a single robot agent is presented in this dissertation. Human search behavior is studied and the robot search performance is compared to the human performance. Our work also investigates how the contextual information such as the detection history and human activity influence the estimation target priors and the use of this information to improve search efficiency.

Secondly, a **decentralized approach for multi-agent cooperative** search in which agents exchange information to be complementary to each other is demonstrated. In our approach, each autonomous search agent maintains separate estimates of the spatial probability distributions for the target object and makes independent decisions about its search process. Asynchronous cooperative search is achieved by transmitting perceptual information among the agents. A novel utility function is proposed for the search agent to complement the limitation of the mobility and viewpoint and recognition reliability of the teammates in an unstructured environment.

Thirdly, a unique approach for **human-robot cooperative** search is described in which a human is modeled as a cooperative teammate whose activity pattern can be learned by robot agents using stochastic models. We also present an implicit interface design framework for robot assisted tasks, which allows the robot to infer the intention of the user and to provide assistance autonomously. It reduces the cognitive workload of the user and therefore is useful for elder care applications. The effectiveness and the efficiency of these systems and approaches are demonstrated in the experimental results.

Last but not least, this dissertation also proposes a novel **object detection and recognition algorithm for low-power cameras**, and demontrates the design and implementation of a dual-camera sensor platform that can be used to track humans and search for objects.

1.4 Document Overview

Chapter 2 offers a review of the literature to provide the background of the approaches taken in this dissertation. Chapter 3 defines the object search problem that needs to be addressed. An architectural overview of the robot search system is presented to show how increasingly efficient search behavior is developed as more sensorimotor resources are recruited. The remainder of the document focuses on individual components of the overall approach for robots search in a human environment, and elaborates on each separately.

Chapter 4 presents the tracking-based object retrival approach. An object detection and recognition algorithm for embedded cameras is introduced in this chapter. Chapter 5 presents the skill acquisition for a single search agent that recruits both the sensor and the motor resources. The performance is demonstrated by different types of agents, mobile robot and PTZ camera agents. In Chapter 6 a decentralized multi-agent cooperative search algorithm is developed. The chapter assumes that teammate agents have communication capability and a message sharing algorithm is presented.

Chapter 7 presents the approach in which human factors are considered to improve search performance. The multi-agent search scheme presented in Chapter 6 is extended to support the recognition of human activities and, thus, learning methods for cooperative human-robot interaction. This chapter also presents an implicit interface design framework for robot assisted tasks, which allows the robot to infer the intention of the user and to provide assistance autonomously.

Finally, Chapter 8 provides conclusions of the work presented in this document and discusses areas of future investigation.

CHAPTER 2

LITERATURE REVIEW

Research in several disciplines has had an impact on the approach reported in this dissertation. The rapid advancement of distributed and service-oriented software architectures (Section 2.1) has made a significant impact on all forms of robotics applications. In particular, publish-subscribe architectures allow resources to be allocated and reconfigured at run time. Section 2.2 discusses technologies in residential assisted-living from both the computer science and the social science literature. In Section 2.3, existing work on robot search is reviewed, including single robot search, multi-robot cooperative search, heuristic search strategies and human-robot cooperative search; Section 2.4 reviews work in object recognition with embedded camera sensor. Finally, the chapter concludes with a brief summary in Section 2.5.

2.1 Software Architecture for Robotics Applications

Robot systems are becoming computing intensive, especially when they must interact with the unstructured client-side environment. It is no longer practical to develop particular control strategies in a top-down manner and to carefully craft the entire software structure of the robot. New architectural paradigms have to be developed for robot systems. The research on software methods and systems for robotics has increased considerably in recent years. The application of good software practices to handle the complexity of robot systems has led to the adoption of software architectures well established in computer science. Three of those architectural paradigms are classified in [16], the DOA (Distributed Object Architecture), CBA (Component

Based Architecture) and SOA (Service Oriented Architecture). As an extension of SOA, the paradigm of Cloud Robotics (CR) was proposed in recent years and has attracted considerable attention.

2.1.1 Distributed Object Architecture (DOA)

Distributed Object Architecture paradigm is based on the concept of merging object-oriented design techniques with distributed computing systems. According to the definition provided by the OMG (Object Management Group) [7], DOA applications are "composed of objects, individual units of running software that combine functionality and data", and run on multiple computers to act as a scalable computational resource. DOA systems rely on the definition of interfaces to support the interaction between server-side and client-side objects. Among the several DOA proposals of the latest fifteen years, the Common Object Request Broker Architecture (CORBA) [2] has achieved the highest level of maturity, which is a vendor-independent specification promoted by the OMG. CORBA has been widely used as a well-proved architecture for building and deploying significant robotics systems [70, 59, 104].

The problem with DOA paradigm is that it requires a tight coupling among entities, which causes a system to be hard to modify, because each change will usually result in other required changes, in a domino effect [16]. Currently, the main area of applications using DOA paradigm is the development of real-time and embedded systems.

2.1.2 Component Based Architecture (CBA)

Component-Based Architectures (CBA) are built upon the concept of software component. W3C [8] defines a component as a software object, meant to interact with other components, encapsulating certain functionality or a set of functionalities. A component has a clearly defined interface and conforms to a prescribed behavior common to all components within an architecture. The goal of CBA is to increase productivity and quality in software development. CBA approaches define a model that the component developers have to follow in order to allow graceful composition. This model specifies the creation, use, and lifecycle management of components and includes a programming model for their definition, assembly, and deployment. Interactions can follow several schemes (synchronous, asynchronous, event-driven, etc.) and they are usually not statically defined but can be manipulated at runtime. The most mature and generally applicable CBA is CORBA Component Model (CCM) [86] of OMG. Additional details about CBA and robotics can be found in [28].

CBA requires a "medium" coupling among components, which is better than DOA's objects. But the component extensibility is often limited because mainstream class-based object-oriented programming languages do not meet a number of important requirements [16].

2.1.3 Service-Oriented Architecture (SOA)

The concept of "Service Oriented Architecture" is that an application consists of a collection of services that are started on demand. There is no core controller, but an assembly of services that are selected by the user for a specific type of scenario. One of the key advantage for SOA is that it provides loosely coupled applications, therefore increases the reusability and extensibility of the system. Recent years some mature robotic middleware systems based on SOA paradigm have been introduced. One of the outstanding work is Robot Operating System (ROS) [46], which is being adopted at a very rapid pace in the robotics research community. It provides an excellent collection of robotic algorithms and operating system type functionality for communication between distributed nodes. Microsoft Robotics Developer Studio (MRDS) [54] is another initiative in applying SOA to robotic systems. MRDS relies on the Microsoft .NET standard and also offers limited support for Unix-based systems. Yet Another Robot Platform (Yarp) [43] is another famous SOA-based middleware system that was started to support research on humanoid robots. The attempts to develop robotic applications in SOA can be found in the literature (a recent survey is presented in [92]). For example, in [40, 72] the authors presented the lessons learned from six years of experiments with planetary rover prototypes running the service oriented middleware developed by the Intelligent Robotics Group (IRG) at NASA Ames Research Center.

The natural loose coupling among software components and the native support of Web service interfaces (SOAP, RESTful, etc.) make it easy to virtualize and expose some computing services in SOA through a cloud. This comes as a result of extending the SOA paradigm to support the concept of Cloud Robotics (CR).

2.1.4 Cloud Robotics (CR)

The application of the cloud computing concept to robots is called Cloud Robotics, which has been attracting a lot of interest in the last four years. CR uses the help of the Internet to increase a robot's capabilities by reducing on-board computation and providing cloud computing services on demand. With CR, robot information can be stored in the Internet and new abilities can be learnt easily and application for robots can share common features. In [31], the authors defined the concept of Robot as a Service (RaaS) based on SOA. The design complies with the common Web service standards, development platforms, and execution infrastructure, following the Web 2.0 principles and participation. Kehoe et al. [63] illustrated a system architecture for Cloud-based robot grasping using a variant of the Google Goggles proprietary object recognition engine. An implemented prototype and initial experiments and analysis are presented in their work.

SOA are considered to be the most appropriate architecture for many robotics applications because it provides loose coupling, high extensibility [16] and compatibility with Cloud Robotics. However, in spite of recent work in this field, research on how to design the control strategy and robot application in a flexible manner in SOA is in its infancy. Such a design would adjust behavior in response to changing run-time contexts by *opportunistically* recruiting available sensory and motor resources. Opportunistic Computing has been investigated and used mainly in Wireless Sensor Networks, Communication, Networking [34]. With opportunistic computing, the execution of applications is supported by spare computational resources available somewhere in the network [18]. Opportunistic Computing paradigm has been introduced to other fields. The work in [119] proposes an approach that implements an autonomic manager as an opportunistic composition of loosely-coupled service oriented management components. Each management component implements a simple administrative task, such as monitoring a parameter, detecting a problem type, planning a specific solution or modifying a managed resource. By opportunistically integrating specialised autonomic management resources, complex and adaptable management strategies are obtained. In [68] the authors proposed an opportunistic activity recognition paradigm, with which the human activity recognition system always uses the available resources and keeps working when the sensor configuration changes. In this dissertation the concept of Opportunistic Computing is introduced into the design of SOA-based robotics applications, and an opportunistic service oriented approach for the robot search system for residential assisted living is presented.

2.2 Residential Assisted Living

As the baby boomer generation ages, health care for the elderly poses a major challenge over the next decade. The growing numbers of elderly individuals in need of support to live in the community will severely test the current services infrastructure. Part of the solution is to develop technology to increase the length of time elders can remain at home. The ultimate goal is to “consumerize” these technologies and make it practical and affordable to incorporate them into existing homes and lifestyles.

Sensor networks and ubiquitous robotics system have often been referred to as the technology that can provide an affordable solution to this problem [76, 11, 35, 114, 115]. Consequently, many research projects have explored the use of sensor systems for medical care at home including a combination of wearable and ambient sensors for vital sign, gait, and fall monitoring. The ASSIST system developed at UMass Amherst proposed the use of PTZ (pan-tilt-zoom) cameras for fall detection and object finding [111]. The AlarmNet system [11] combines wearable vital sign sensors with other stationary sensors placed within the living environment to provide home or assisted-living health monitoring services. In the CareMedia project [22] a large number of high-resolution cameras are used to provide constant monitoring of the public spaces in a dementia unit in order to reduce the burden on human caregivers. Intel's Long-Term Care project [101] focuses on the use of RFID technology to detect activities of daily living among the elderly. The CodeBlue project [76] focuses on developing a scalable software infrastructure for discovering and connecting wireless medical sensors, PDAs, and PCs. Some smart living environments for research on sensor organization and activity recognition have also been developed, such as the Aware-Home [12], the PlaceLab [96] initiative and Tiger Place [108].

Although many works in this field have been proposed, the key issue that how to design complex applications on the base of ubiquitous and heterogenous sensorimotor resources, especially the robotic resources is still open. The work in this dissertation is part of the ASSIST project developed in Computer Science Department of UMass Amherst that uses mobile robots and sensor network to solve the problems in residential assisted living [111]. In this dissertation an opportunistic SOA framework is presented to address the object search problem, and the proposed framework can be extended to the design of other applications in assisted living.

2.3 Object Search Systems

In addition to monitoring for illnesses and potentially life-threatening situations, an equally important challenge in residential assisted living is providing assistance in clients' day-to-day life. Moderate memory impairment is common as people age, hence a major problem for the elderly is locating frequently used "common" objects such as keys, cellphones, books and others.

2.3.1 Object Search Systems

There has been considerable recent interest in addressing the problems of "object search" both in academia and industry. The work in [88] investigates the real-world nature of what losing an object means and general strategies that can be used to find those objects. Some of the existing services seek to attach wireless tags on the object, such as RFIDs [24, 71, 112], Bluetooth chips [65], or 802.15.4 radios [82, 87], making it easier to track and localize the object. While this offers a feasible solution for objects such as car keys and cellphones that might already have wireless tags on them, the solution is cumbersome since it requires that every possible object that may be misplaced needs to be tagged a priori. For some kinds of sensors like RFID that have short sensing range, the receivers must be deployed densely in the residential space or carried by the user, neither of which is convenient in the eldercare context.

Other approaches utilize visual information [83, 111], e.g., the ASSIST project proposed the use of PTZ cameras for object finding [111]. Some research has been done on combining RFID and vision sensors to improve the performance of object detection and search. The work in [100, 53] discussed the basic idea of using RFID to roughly localize an object and then applying vision to refine the location. In [58], the RFID system estimates a rough position of each object. Then each object that is attached to an RFID tag is visually recognized using color histograms obtained by ceiling mounted cameras. However, these systems employ image recognition methods that are not robust, and the cooperation of RFID and vision sensor was not investigated. McDaniel [107] presents a conceptual framework where RFID and computer vision are integrated for the task of remote object perception in a wearable system for blind people. In this work, visual information aids the RFID detection system in that it enables only the object in front of the user to be detected. Furthermore, this system can be used in untagged environments. In [32], the robot obtains predefined CAD models via RFID tags placed on each object and uses the models to recognize and localize targets. In a more recent approach, Kim et al. [66] developed a robotic system for object recognition and localization. The authors proposed the use of smart tags that have an active landmark (IRED) and a data structure consisting of geometrical, physical and semantic information. When a tagged object is read, its IRED is activated, and the robot searches the scene for the active landmark (a flickering light). When the light is found, stereo cameras on a pan-tilt mechanism are used to find the object's depth, size and pose. The work in [30] presents a method for object recognition in complex scenes combining vision-based techniques applied to the 3D data obtained using range sensors, and object identification coming from RFID. Although some progress has been achieved, prior efforts to combine RFID and vision sensors are still not good enough: most of them handle the sensor information separately (e.g., RFID is responsible for detection and vision is responsible for recognition).

Our work differs from these approaches in that we investigate the approach that builds object search applications in multi-tier manner, by incorporating heterogeneous sensorimotor and computational resources and services in an opportunistic SOA framework. Particularly, this dissertation considers opportunities for collaboration between the automated search procedure and the human client. This contribution has received far less attention in the literature than purely automated systems.

2.3.2 Heuristic Approaches for Object Search

Efficiency is an important concern when evaluating technology for residential healthcare. Object search can be seen as a sensor planning problem in which an appropriate sensor configuration must be selected in order to allow a proper recognition. Sensor planning is formulated as an optimization problem in which the goal is to maximize the target detection probability while minimizing the energy, distance traveled, and time to achieve the task. However, the sensor planning problem is NP-complete [118] and thus a heuristic strategy is needed to overcome predictable computational issues. A fruitful strategy to overcome computational complexity is to introduce object detection subtasks for identifying objects that are probabilistically associated with the target to refine the spatial search. Wixon [113] uses the idea of indirect search, in which one first finds an object that commonly has a spatial relationship with the target, and then restricts the search in the spatial area defined by that relationship. The problem with indirect search is that the spatial relation between the target and intermediate object may not always exist. In addition, the detection of the intermediate object may not be easier than the detection of the target. Sujan [103] proposes an iterative planning approach driven by an evaluation function based on Shannon's information theory. The camera parameter space is explored and each configuration is evaluated according to the evaluation function. The work in [39, 97] proposed a visual attentional framework developed for the humanoid robot HRP-2 in order to implement object search behavior. The problem is formulated as an optimization problem. The concept of a visibility map is introduced to constrain the sensor parameter space according to the detection characteristics of the recognition algorithm. By this means, the dimension of the sensor parameter space is reduced. In [117] the search agent's knowledge of object location is encoded as a discrete probability density which is updated after each sensing action performed by the detection function.

People interact with objects in the course of many tasks associated with daily living. A novel idea in this dissertation is leveraging user activity to improve the cost efficiency in search tasks. User activity density can be analyzed from the vision-based people tracker, and can be used to infer the region where object use may happen. To the best of our knowledge, no work has been done that uses human activity information to help reduce the search space of the robot agent.

2.3.3 Cooperative Search Strategies

A multi-agent system is well suited for search operation, especially when in a complex environment and the mission is time sensitive. Multi-agent search systems have been proposed to locate the fire ignition point [77], find an intruder of a building [14] or search source of radiation, mines [13], victims [60] and the odor of waste [75], regarding its carried sensor and manipulator. In search operations, a team of intelligent agents can provide a robust solution with greater efficiency than can be achieved by single agents, even with comparatively superior mobility and sensors. The key to exploiting this observation in a multi-agent framework is to develop a cooperative decentralized control strategy that allows each agent to determine its actions independently while optimizing the team's performance. A synchronized coordinated search strategy was developed in a Bayesian framework in [26]. DeLima et al. [37] proposed a rule-based search method with which multiple unmanned aerial vehicles can cooperatively search an area for mobile target detection.

Although there have been considerable works on multi-agent cooperative search, most of them focus on achieving optimal planning in a single search trial without considering using accumulated knowledge to achieve efficient cooperative behaviors. Prior knowledge can be used to represent teammates' search capabilities. Ideally, an agent should learn the limitations of its teammates by observing their performance and select actions to compensate for these limitations. In this dissertation, we investigated how to achieve better performance by considering these limitations. Two types of limitations are considered, (i) viewpoint limitations, which represents the ability of an agent to reach certain locations in the search space, and (ii) limitations on observation reliability, which describe the ability of an agent to make a true-positive detection in certain locations.

2.3.4 Human-Robot Cooperative Search

Our vision for search problem solving is that humans and robots will work as partners, leveraging the capabilities of each. Human-robot teams are used in Urban Search and Rescue (USAR) [81, 84], but these applications typically use robots as "drone" under direct control of human teleoperators. In these works Search and Rescue robots are studied as near-ideal application for studying HRI (Human Robot Interaction), where human intervention enters in at some level to help ensure robustness in planning and perception, e.g., via manual tuning of heuristics and model parameters by expert human programmers; mixed-initiative/supervisory control by trained operators [102, 41]; or natural interactions with untrained non-experts [21].

Some works treat human as a teammate to accomplish the search task instead of a centralized commander, planner or manager of the system, and explore how human observation can be combined with robot sensor data to improve autonomous state estimation and model learning. Here human observation can be eye observation or carried sensor inputs. Early work by [62] and [25] showed how certain human sensor inputs could be formally characterized and fused with robotic sensor data for augmented physical perception through the Bayesian paradigm. In [25] peer-to-peer collaboration between human-computer augmented nodes and autonomous mobile sensor platforms is achieved by sharing information via wireless communication network. The individual controllers iteratively negotiate anonymously in the information space to find cooperative search plans based on both observed and predicted information that explicitly consider the human motion model, its sensors detection functions, as well as the target arbitrary motion model. These works usually limit the expected complexity and scope of human sensor inputs, largely for the sake of analytical and computational tractability. For instance, [62] assume that humans provide numerical range and bearing measurement data for target localization ("The object is at range 10 m and bearing 45 degrees"), while [25] assumes that humans provide binary "detection/no detection" visual observations for a 2D multi-target search problem. To incorporate broader range of information that can be provided by human teammates, such as using natural language semantic information, [15] proposed a Probabilistic Semantic Human Sensor Model.

All the existing work on human-robot cooperative search neglect the learning of human teammate's behavior pattern. By observing the search behavior of human teammates and learning their behavior pattern, the robot's capability and experience can be accumulated. In this dissertation I propose to use stochastic model to learn the human search pattern, which is used by the robot to perform complementary search actions and improve the search performance of the H-R team.

2.4 Object Recognition with Embedded Camera

In this dissertation I propose the design and implementation of an indoor object retrieval system using a network of wireless camera nodes. This section presents the related work on this topic.

2.4.1 Multi-tier sensor network:

The multi-tier structure for wireless sensor network has been considered in prior work to achieve energy-efficient computer vision tasks. Tenet [47] argues for a multi-tier design and SensEye [67] proposes a three-tier camera sensor network for surveillance. Our work uses two tiers, but they are tightly coupled as part of a single platform. In addition, we propose novel techniques for splitting an object recognition task between a low-power and high-power camera.

2.4.2 Object recognition:

Many different approaches to object recognition have been proposed in computer vision, including model-based and appearance-based approaches [99]. In recent years, methods using local appearance features [73, 79] have come more popular. In [109] the SIFT descriptor is combined with color histograms. The work in [90] discusses fusion methods for SIFT and LUV color moments descriptors. In our work we exploit combination of SIFT and color features for energy-efficiency as opposed to recognition accuracy.

2.4.3 Semi-supervised clustering:

Clustering is traditionally viewed as an unsupervised method for data analysis. Based on the widely used k-means algorithm, some constrained versions have been developed [110, 19, 45] to incorporate the information about the problem domain that is available in addition to the data instances themselves. As an extension of the model proposed in [19], our work incorporates hard and soft constraints together, which achieves cluster refinement for accurate object classification.

2.5 Summary

In summary, the approaches proposed in this dissertation is an advance in the related fields in robot search on two fronts: (1) It propose an opportunistic service oriented architecture for assist robot applications and uses object search problem as a case study to investigate the potential of the opportunistic SOA. The robot builds object search capability in a dynamic and hierarchical manner, which gives the robot capability to interact with environment, human and resources as well as the ability to perform efficiently in a resource constrained situation. (2) In each hierarchy of the search application, this dissertation propose novel approaches to achieve effective and efficient search, ranging from passive tracking-based (yet energy-efficient) object detection and retrieval approach to robot-human cooperative object search strategies. In the following chapters the proposed architecture and approaches will be introduced and experimental results and analysis will be presented.

CHAPTER 3

OPPORTUNISTIC SERVICE-ORIENTED ARCHITECTURE

This chapter focuses on the service-oriented architecture of the proposed object search system. In first section I present a description of the assisted living problem as the context of the object search problem. It helps us to understand the requirements and challenges of the object search system. The second section covers the details of the architectural design.

3.1 Specifications

Robot systems designed for assisted-living have been the focus of considerable research in both academic and industry. Important examples of successful applications of this research involve monitoring the health and activity of clients and mechanisms for providing cognitive and physical assistance for independent living.

3.1.1 Emergency (vital sign) Monitoring.

Time plays a determined role in health care. Some common life-threatening medical emergencies include stroke, cardiac arrest (myocardial infarction or heart attack), and seizure. Time-critical treatment usually makes a tremendous difference in these cases. Assisted living robots should provide emergency monitoring functions such as automatic blood pressure readings, heart rate monitoring and fall detection. Once the robot detects an emergency or deterioration in the user's health, it must be able to communicate this information to the relatives, care takers, or health care providers responsible for the user. It is important that this action is carried out in real time, particularly in an emergency situation. To achieve real-time response, in an assisted-living system, emergency monitoring applications usually have the highest service priority.

3.1.2 ADL (Activities of Daily Living) Analysis

ADL analysis monitors and records the user's daily activities, which can be accessed and used by the caregiver to determine the health condition of the client. In ADL analysis, different levels are involved. It includes human trajectory analysis to provide information to higher level activity recognition modules that identify activities such as sitting, walking, reading and cooking. This data, in turn, is used to estimate the health of the client. In this way, assisted living systems would provide not only an immediate, reactive response to health care shortages, but also a long-term, proactive solution to the health problem of the human clients. ADL Analysis usually doesn't require a high service priority and is tolerable to temporary and short-term data shortage, since it focuses on finding patterns from long-term data instead of a single event.

3.1.3 Assistive Services

In addition to monitoring for illnesses and potentially life-threatening situations, an equally important challenge in residential assisted living is providing assistance in user's day-to-day life. Assistive services include applications that promote the independent living of elderly clients by assisting them in daily tasks. Examples of this type of service include personal information managers that can remind the client of important appointments or mobile robots that provide physical assistance. Object search also fails in this category, which can help locate misplaced items such as keys and cell phones. The service priority for assistive services usually is lower than that for Emergency Monitoring and higher than that for ADL Analysis, and are provided in an on-demand manner.

We focus on developing object search application that helps the client to find common objects efficiently in the assisted-living context. The object search application is part of the assisted-living system ASSIST [111] developed at UMass Amherst. Figure 3.1 illustrates the problems and applications related to this assisted-living system. It can be seen that in this system the supported applications include object finding, fall alerts and client trajectory analysis. Considering the above analysis on the specifications of the assisted living system, we can see that to develop object search applications in such a system, some challenges must be addressed.

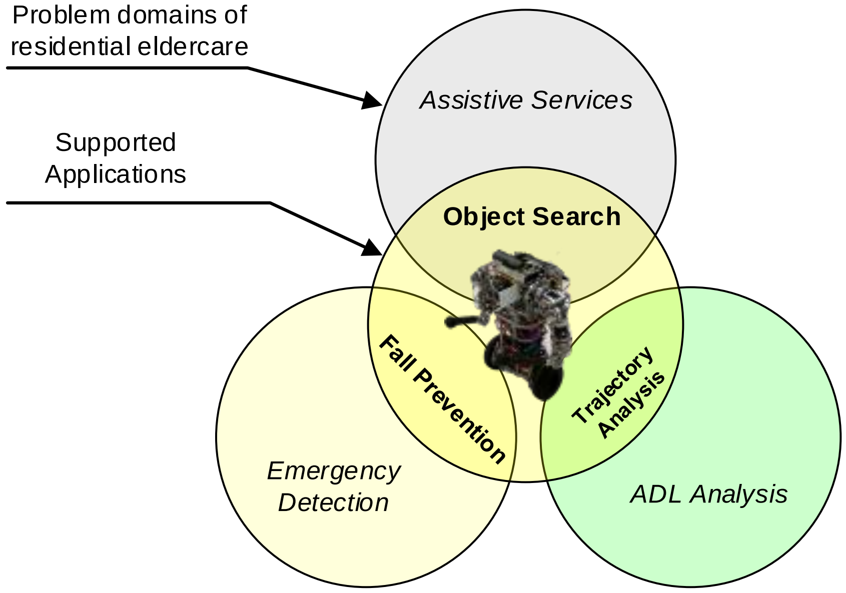


Figure 3.1. Problem domains and the supported applications in a residential assisted-living system.

3.2 Challenges

3.2.1 Heterogeneous hardware/software components.

In residential assisted-living environment, the robot system must interact with the unstructured and dynamic client-side environment, and must cooperate with other agents and human clients. Heterogeneous sensorimotor hardware/software components must be adopted to perceive and interact with this complex and dynamic environment. A novel robot system architecture that facilitates sensorimotor resource organization needs to be developed.

3.2.2 Multi-tasking and resource contention.

For wide spread use and commercialization, personalized robotic services must be packaged in a manner that pushes the price/performance ratio downward. In residential assisted-living systems, sensorimotor resources are limited. On the other hand, multiple tasks with different service priorities need to be executed simultaneously. For example, in our assisted-living system, fall prevention and object search applications may run at the same time. Since they all share monitoring hardware (Pan-Tile-Zoom camera) and computational resources (e.g., the "human tracking" component), resource contention will occur.

3.2.3 Reliability.

Reliability is essential for assisted-living applications. The sensors may disconnect from the system due to hardware, software, and network failures or become unavailable due to resource contention. Assisted-living applications must be able to handle these situations and provide effective and reliable results to the client.

Since none of the current related work fully fulfilled our requirements and vision, a number of scenarios and activities were studied.

3.3 Architectural Overview

Our goal is to develop robot search approaches that satisfy the requirements of assisted-living applications. An opportunistic service-oriented approach is advocated.

3.3.1 Service Oriented Architecture

Service oriented design has been heavily investigated [51, 55]. It aims at separating tasks by breaking a computer program into distinct modules with minimum overlap in functionality [9], provides a design framework for rapid, low-cost system development, and includes mechanisms for total system quality improvement.

Service-oriented architectures decompose the infrastructure of a personal robot system into distributed elements composed of heterogeneous resources that can be federated in many ways to support many applications. As shown in Figure 3.2, heterogeneous sensorimotor resources including Pan/Tilt/Zoom (PTZ) cameras, mobile robots, and RFID devices are employed in the assisted living system. Computational services are provided as local services or as cloud services through a privacy protection mechanism. In our system, local computational services include object recognition, human tracking, and activity recognition. Cloud computational services, such as object detection and recognition were considered, but were not used in the prototype system presented in this dissertation. Although not being used, the cloud computational services are natively supported by the presented system thanks to SOA's compatibility to cloud computing.

Multiple client applications including object search, fall prevention and ADL recognition are composed of basic services. The hardware and computational services involved in object search applications are illustrated in Figure 3.2.

As illustrated in Figure 3.3, assisted living applications contend for basic services. The same service is used by different applications with different service priorities. To deal with the resource contention, an application must be flexible enough to adapt to the limited service resources.

The possibility of developing service oriented robot systems lies in the availability of middleware to handle messages between modular sensory and motor resources (e.g., MRDS [54], ROS [46], NDDS [44], Yarp [43]). In this work, the proposed approach is

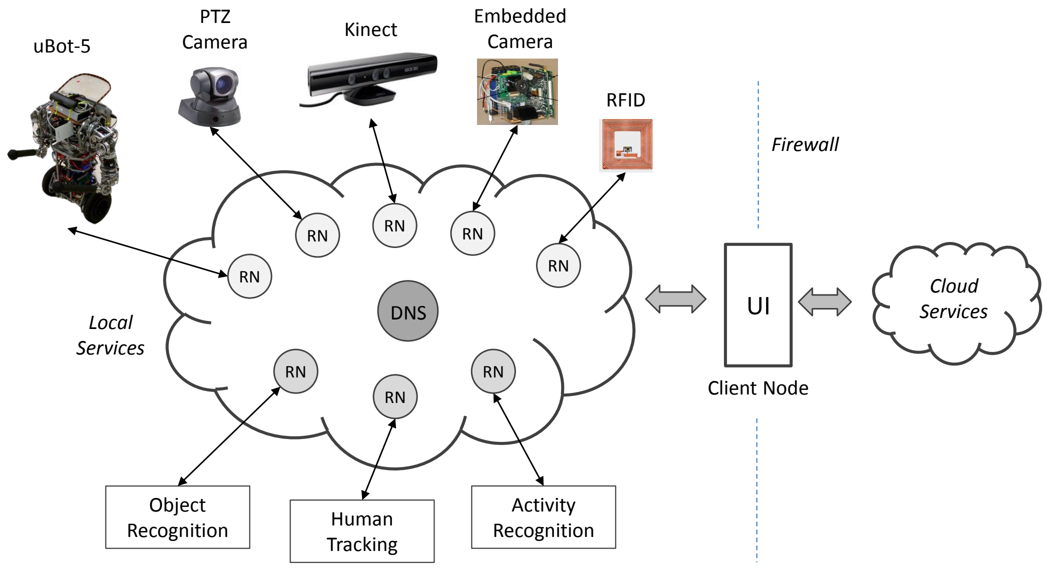


Figure 3.2. System infrastructure consists of sensor, motors, kinematic devices, computational services and system interfaces. Local and remote (cloud) servics are considered. Local services support robotic devices and functional units. RN denotes ROS Node and DNS denotes the Distributed Naming Service in ROS.

independent of the choice of the middleware implementation. In practice both MRDS and ROS were used to develop the proposed object search application. ROS was used for the experiments in this dissertation.

3.3.2 Opportunistic Service Oriented Approach (SOA)

Well-functioned robotic systems using the SOA principle have been demonstrated [40, 64]. However, design principles for single applications in a resource-constrained environment are still actively being investigated. In this dissertation, we argue that in a multi-tasking system that deals with resource contention and node/communication failure, the applications need to be able to execute at different running levels according to different resource conditions and performance requirements. The opportunistic service-oriented approach is a good fit for this problem.

The concept of opportunistic service architecture was introduced in the sensor network area [9]. Opportunistic Service-Oriented Architectures (OSOA) consist of

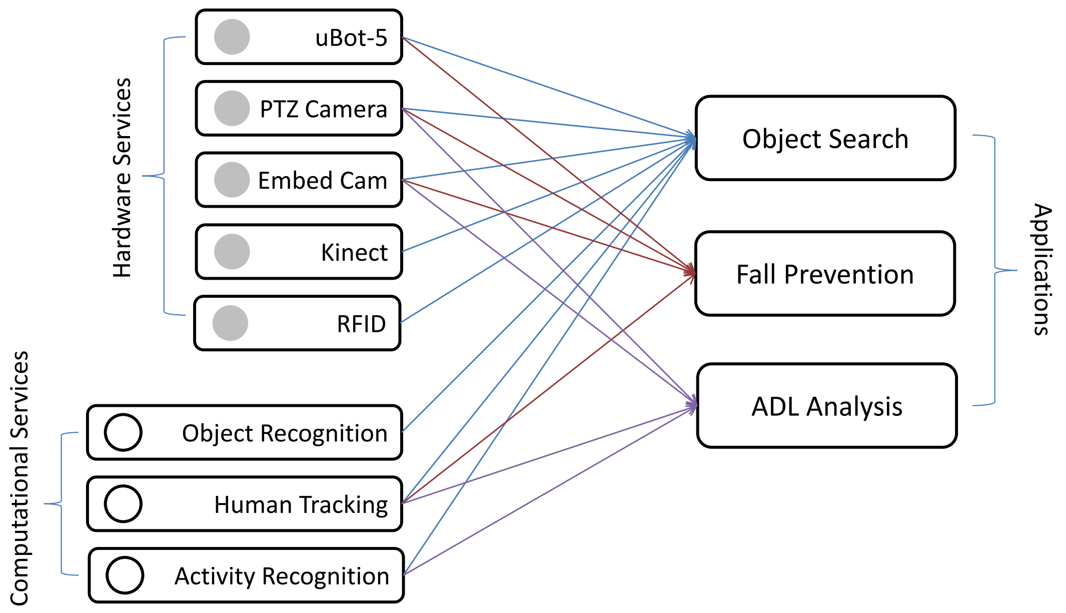


Figure 3.3. Relationship between the hardware/computational services and the applications in the assisted-living system

changing constellations of services and nodes that, for a limited amount of time, work together to achieve a common goal. The OSOA applications are designed in a hierarchical manner. Each tier in the hierarchy is composed of a subset of the sensorimotor services and provides certain performance guarantees with respect to the functionality required. Applications have the flexibility to adapt at different levels in the hierarchy by trading resources when there is an opportunity so as to balance the performance of a task against the overall value of a suite of tasks.

This dissertation proposes an opportunistic service-oriented approach for an object search application in the assisted-living system. Figure 3.4 illustrates the design hierarchy of the object search application. The application is built in an incremental manner in different tiers, each of which comprises a set of services. A bottom-up approach is adopted to develop a hierarchy of search strategies from tracking-based search (lowest tier) to human-robot cooperative search (higher tier). This approach starts from the least use of resources (tier-1), and incrementally recruits more sensorimotor resources for achieving higher performance.

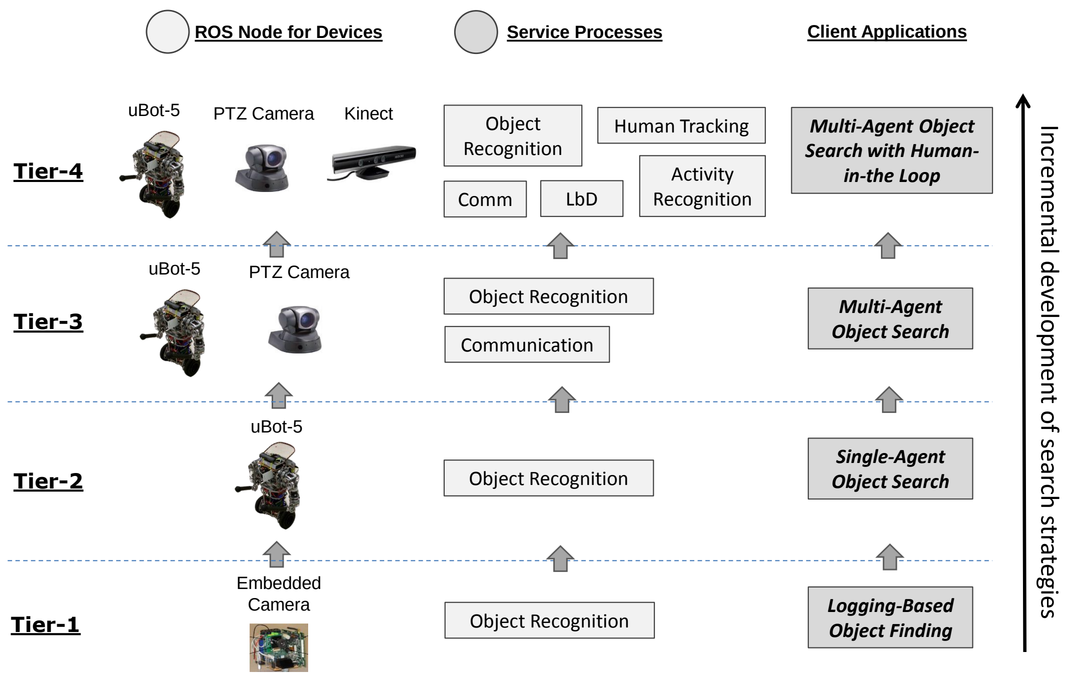


Figure 3.4. A road map for developing search strategies in multiple levels.

Tier-1: Tracking-based object retrieval. The basic search strategy (the one that lies in the lowest tier with minimum resource requirement) must be very energy efficient to maintain a persistent and basic functionality. As shown in Figure 3.4, in Tier-1 there is a tracking-based object retrieval strategy that satisfies this requirement. By passively monitoring the environment and recording the movement of the target objects, the tracking-based method can provide location information of a set of known objects to the user on demand. The tracking-based strategy consumes the minimum set of the service resources, which only includes the low-power embedded camera sensors.

Tier-2: Single agent object search. Although being effective and energy-efficient, the tracking-based object retrieval method still needs considerable effort to deploy the camera sensors to achieve complete coverage. This may be hard to achieve when the room structure is complex and dynamic, in which case the sensors must be deployed more densely, or relocated frequently according to the change of the room structure. To address this problem, a mobile robot performing active search is involved in the proposed search system. By performing active search, the mobile robot is able to reach most of the area and achieve a more complete coverage than the tracking-based strategy, even when the room structure is complex and dynamic. In our system, a robot agents (the uBot-5 mobile manipulator or an elevated, immobile PTZ camera) can be employed opportunistically when available to help the client to find objects.

Tier-3: Multi agent object search. In search operations, a team of intelligent agents can provide a robust solution with greater efficiency than can be achieved by single agents. When more sensorimotor resources are available, our system recruits multiple robot agents to perform search cooperatively. The cooperation can be achieved in a team with the uBot-5 robot and PTZ camera nodes.

Tier-4: Joint search in human-robot team. It is important for the robot system to cooperate with the human clients to perform search. In the proposed framework, the cooperative search strategy is extended in a way that models the human as an agent teammate without a direct means of message transmission. Given that human tracking and activity recognition services are available, the robot search system can work with human teammates by inferring their search intention or by interpreting their pointing gestures, which can lead to a more efficient search.

When executing, the object search application will opportunistically employ available service resources to get to the highest tier possible so as to achieve more efficient search performance. When resource contention occurs, the application degrades to a lower tier to release the resources to the applications with higher service priorities. For instance, when the application is running in tier-3, the robot and PTZ cameras search for the object jointly. However, the fall prevention application is configured at a higher priority and requires all PTZ cameras to track the human client. The object search application will release the cameras and will run in Tier-2, where only the uBot-5 continues to search for the object. By this means, the system guarantees the performance of the applications with high service priorities but still maintains best-effort performance on the application with lower service priority like object search.

In the following chapters, the design and implementation of the search strategies in each execution tier is discussed. Experimental results are presented to evaluate the performance of the proposed search strategies. A performance improvement is expected when the system recruits more sensorimotor services and executes in a higher tier.

CHAPTER 4

TRACKING-BASED OBJECT RETRIEVAL

In this chapter, the lowest tier in the aforementioned application hierarchy is described, and performance evaluation results are presented. The basic idea underlying the tracking-based approach for object search is the opportunistic identification and tracking of objects within the set of target objects. It is the lowest-tier option for search (Figure 3.4) and when the target is discovered, it records movements and retrieves historic images containing the object. The tracking-based object retrieval module uses energy-efficient embedded camera sensor nodes to perform consistent and non-intrusive monitoring.

There are two contributions arising from the method proposed in this chapter: (i) an energy-efficient object recognition approach is proposed for object tracking and retrieval using low-power wireless cameras. The system employs a technique for splitting an object recognition task into a low-power and a high-power camera part; and (ii) a dual-camera structure that comprises different types of small wireless cameras is designed.

4.1 Dual-Camera Structure

In this work we explore how low-power camera sensors can be distributed in a home environment to facilitate retrieval of objects. Small, battery powered cameras are portable, easy to deploy, and can be densely arrayed for greater coverage. While there have been many efforts in recent years to design low-power smart camera networks for surveillance, object tracking, and object detection [50, 106, 67, 17], our work is fundamentally different in that we focus on achieving energy-efficient recognition of common objects in the home.

The design of an object recognition system using low-power cameras poses significant technical challenges. The first challenge is that state-of-art image matching techniques (e.g. SIFT [73]) are expensive computationally and energy intensive, making them ill-suited for use with embedded processors. This makes it essential to develop techniques that are less complex and consume less energy but can still enable robust image recognition. The second challenge is that image recognition typically requires a high-end sensor device with a high-resolution camera, and substantial computation and memory resources. However, the use of higher-end sensor platforms (e.g. iMote2 [5]) comes at the cost of energy-efficiency, and consequently reduces the utility of the sensor due to the short battery life. One commonly proposed approach is to use a multi-tier network [67], where a low-power and low-resolution wireless camera node (e.g. Cyclops [10]) is used for object detection, and wakes up a higher power camera to perform object recognition only when needed. The problem is that the high-power camera still needs to wakeup and take images periodically to update its background model. The system described here addresses both these problems in a dual-camera structure.

4.1.1 Hardware Components

The tracking-based object retrieval application described in this chapter involves a network of dual-camera nodes, each of which comprises a low-power and high-power part that are physically connected as shown in Figure 4.2. The low-power part (Part-1) is a MICAz mote [3] equipped with a low fidelity Cyclops camera sensor (CyclopsCam) [10, 94]. The high-power part (Part-2) is a more-capable platform, the Intel Mote2 (iMote2) [5] equipped with a high fidelity Enalab camera (EnalabCam) [4]. The two cameras in the dual-camera structure are placed close enough so that they have a similar field-of-view (FOV) (alignment is a single affine transformation). A proxy node is used to organize the information from the dual-camera sensor nodes, as well as to process user queries. The proxy node runs on a Linux computer that is connected to an 802.15.4 wireless radio, and can communicate with the MICAz and iMote2 nodes.

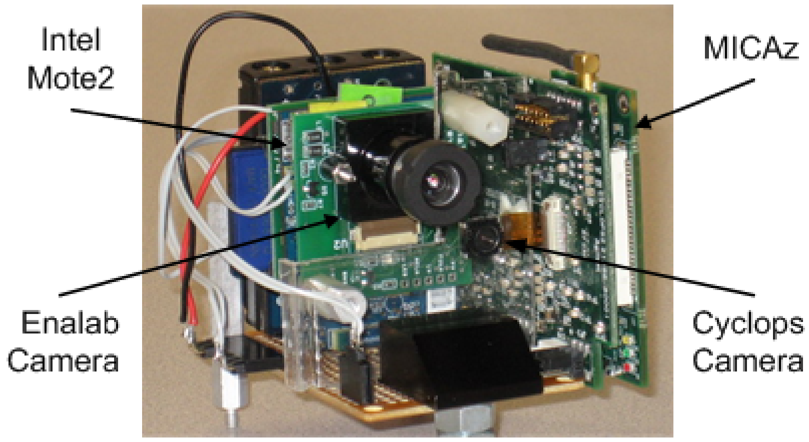


Figure 4.1. The dual-camera sensor node

4.1.2 Overview of System Operation

The operation of our system can be divided into three main components: object detection, object recognition, and object retrieval. The low-power camera in Part-1 takes an image every few seconds, and performs still object detection, i.e., it determines if an object that is detected is likely to be a newly placed object as opposed to a moving object (discussed in Section 4.2). If a still object is detected, the Part-1 camera stores the location and size of this object in its local flash memory. Once a batch of still objects have been detected, the Part-1 camera wakes up the Part-2 camera node and transfers the stored images together with information about the region in the image where the object was detected.

The Part-2 camera node uses an inter-camera region mapping function to map the Part-1 ROOs (Region of Object) to its own camera co-ordinates. This enables

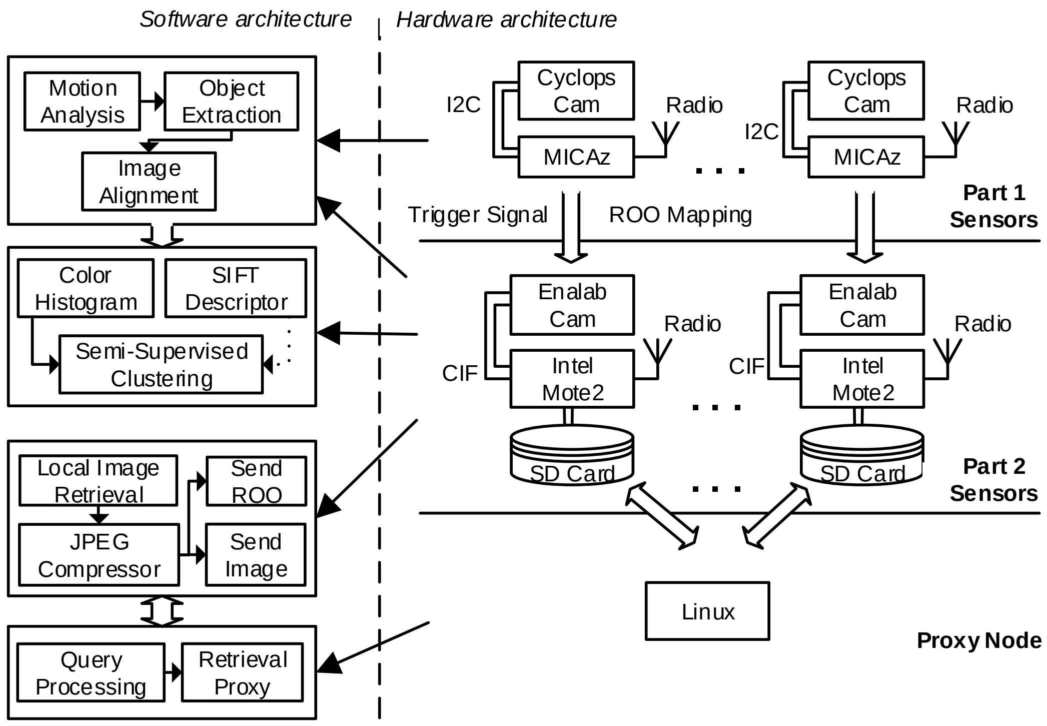


Figure 4.2. Software and hardware architectures of the tracking-based object retrieval application

it to determine which regions in its own view correspond to the new objects. Next, the Part-2 node takes an image using its high-resolution camera, extracts the ROOs corresponding to new objects, and obtains the color histogram corresponding to each ROO. The object recognition procedure first tries to recognize the object in each ROO by using the color histogram together with a semi-supervised k-means clustering. For objects that cannot be classified correctly using color features, the SIFT recognition algorithm is used as additional evidence.

A proxy node is used in this system to organize the information from the dual-camera sensor nodes, as well as process user queries. The tagged classification results from the Part-2 node together with a detection timestamp are transmitted to the proxy node, and the raw image data is locally stored on the flash memory in Part-2 node. The Part-2 camera node then goes back into sleep mode.

The user can query the system by specifying an object tag that needs to be located. An approximate time frame of interest can also be provided by the user to further refine the search. The proxy node locates the most recent event corresponding to the requested object, and queries Part-2 sensors that have reported the object. Since the Part-2 sensor is asleep, this query is first received by the Part-1 node which wakes up the Part-2 sensor and forwards the query over the serial connection. The matching image ROOs are retrieved, and displayed on a GUI (Graphic User Interface) to the user, who can mark images to be "Valid" or "Invalid". If the required object is not found, the proxy retrieves the previous event matching the query, and repeats the same procedure. This continues either until the object is located, or until no more objects are detected in the time frame of interest. Periodically, the proxy also returns user feedback to the appropriate sensors, which use this information to refine their clusters, thereby enabling more efficient and more robust recognition.

The detailed algorithms for object detection, object recognition and object retrieval are presented in the following sections. Then the system implementation and experimental results are given.

4.2 Object Detection

The object detection procedure in our system involves two steps. The first step is detecting the presence of a still object at each Part-1 CyclopsCam. This procedure aims to filter out transient motion in the field of view of the camera such that only objects that stay relatively motionless are detected. The second step involves triggering the Part-2 EnalabCam, and mapping from the ROO in the CyclopsCam to the EnalabCam. This procedure aims to address the fact that the EnalabCam is woken up infrequently and cannot maintain a reliable background model locally. Hence the EnalabCam needs to be told approximately where the detected object is located in its image.

4.2.1 Object detection in ultra-low-power tier

For object detection, the Cyclops node maintains the background using an average update model, which is computational-efficient. The background image B_m is updated by integrating the new frame I_c into the current background with a first order recursive filter: $B_m^{k+1} = (1-\alpha)B_m^k + \alpha I_c$. The blobs that represent potential objects are extracted by background subtraction. To detect that a candidate blob represents a new object placement event as opposed to a transient motion event, we need to check if this blob has been detected before and has been still for a sufficiently long time. To achieve this, we compare all the blobs in the current frame with those in the previous frame. If the size and center of a blob is similar enough to those of a blob in the previous frame, these two blobs are considered to be the same object. When the detection duration of an object blob becomes longer than a pre-defined threshold, it is classified as a still object. The object blob is then extracted and saved on the local flash memory of the MICAz mote.

4.2.2 Sensor Triggering and ROO mapping

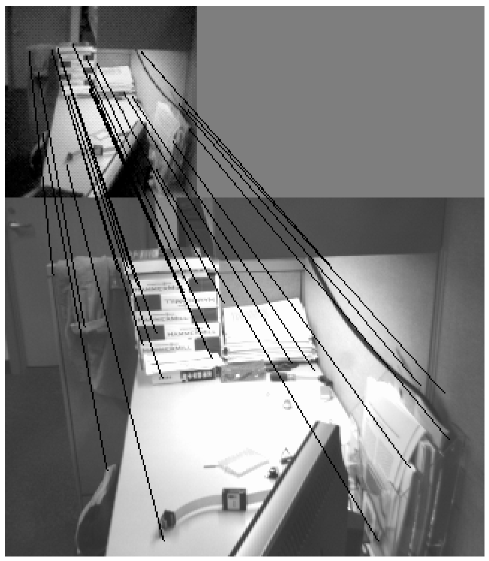
After the still objects are detected, the Part-1 node needs to wake-up the Part-2 camera in the dual camera node from deep-sleep mode. Since the wake-up of the iMote2 node from this state has long latency and consumes significant energy, the Part-1 node triggers the Part-2 node after a batch of still objects are detected. In this manner, the energy consumed to wakeup the Part-2 camera is amortized across multiple detections. Note that the wakeup delay is not a problem for our application since we are trying to detect still objects that remain in the scene for a significant duration. Our system will not be as effective in a tracking scenario since the latency of wakeup needs to be low in order to track motion. Although the FOVs of two cameras are similar, there may be slight translation, rotation and considerable scaling between them due to installation bias and imprecision in the mechanical mounts. To achieve robust region mapping from the pixels $\mathbf{P}_j = [u_x, u_y, 1]^T$ in CyclopsCam image to the pixels $\mathbf{P}_j^{'} = [v_x, v_y, 1]^T$ in EnalabCam image, a motion model is defined based on the affine transformation.

$$\mathbf{P}_j^{'} = \mathbf{D}_{x_0,y_0}\mathbf{S}_{s_x,s_y}\mathbf{R}_\theta\mathbf{P}_j$$

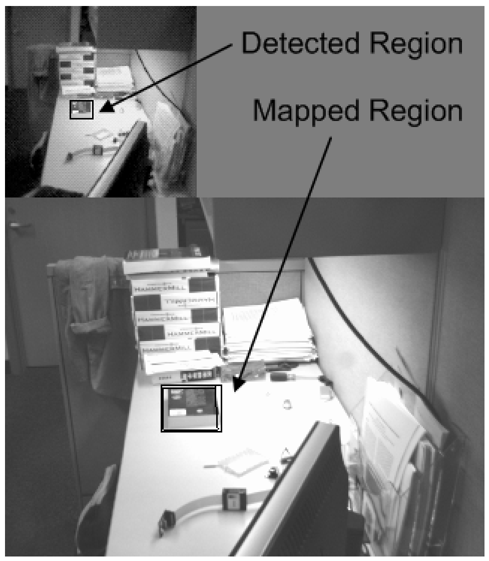
where $\mathbf{D}_{x_0,y_0}$ is the translation matrix, $\mathbf{S}_{s_x,s_y}$ is the scale matrix and $\mathbf{R}_\theta$, the rotation matrix. In order to solve the motion model, the dual-camera node executes a calibration procedure at system deployment time. Both cameras take an image simultaneously, and the Part-1 node transfers its image over the serial port to the Part-2 node. The Part-2 node then extracts SIFT descriptors [74] from the two images. Since the SIFT descriptors are invariant to the image rotation, scale, and translation expected in this application, they provide a consistent set of local descriptors to match between the two images. This calibration procedure is typically more computationally intensive than object recognition since it needs to be performed on the entire image as opposed to just the ROO. However, this is a one-time computation, hence its overhead is a very small fraction of overall energy resources. After the inter-tier calibration is done, any ROO in CyclopsCam image can be mapped to the EnalabCam image efficiently using the motion model. Figure 4.3 (b) shows a mapping result, in which the bounding box around a book that is detected on the desk in the upper CyclopsCam image is mapped to the appropriate rectangle in the lower EnalabCam image.

4.3 Semi-supervised object recognition

Given the Region-of-Object extracted by the object detection module, the next task of the Part-2 node is to efficiently recognize the object. Our approach includes three procedures: feature extraction, object recognition and constrained cluster update.



(a)



(b)

Figure 4.3. An example of Field-of-View and ROO mapping: (a) Control points found by SIFT. (b) The ROO mapping result.

4.3.1 Feature Extraction

We use two kinds of features to classify objects. The first type of feature is the 32-bin hue histogram that represents the global color information of the object region. The second type of feature that we use is the SIFT descriptor, which represents an image as a collection of local feature vectors that are invariant to image translation, scaling, rotation, and partially invariant to illumination changes and affine or 3D projection [74]. Given two images, a matching algorithm is performed to calculate the number of matching points between them, which represents the similarity.

Both color and SIFT features have their advantages and limitations in object recognition. Color features can be calculated in a computationally inexpensive manner, and are invariant to severe scale, rotation and 3D projection; however, they are not invariant to illumination changes. SIFT tolerates illumination changes and is widely considered to be one of the best feature representation methods, but it is computationally intensive and does not perform well for deformable objects or objects that have no consistent texture.

Our method uses a combination of the two features in a cascading manner. The idea is to use color features to filter out irrelevant images and to classify images that have distinctive color hues, since color histograms are computation efficient. The SIFT features are then used to recognize the remaining images that are hard to classify using only color. This combination enables us to tradeoff between efficiency and robustness, since SIFT matching is performed only on a small set of unclassified but relevant images. We note that the idea of combining two kinds of image features for more robust detection has been considered in the image processing literature [98, 91]. However, we exploit this technique for energy-efficiency as well.

4.3.2 Object Recognition Procedure

The object recognition procedure initially gathers a small set of object images as training samples and uses them to generate a set of clusters using the standard k-means algorithm. During this phase, clusters may be tagged by a user. For instance, one cluster might correspond to a cup whereas another might correspond to a cellphone. After this training phase, the system can be used to monitor the scene continually, using this initial cluster model to detect a possible object/region of interest. We now describe the recognition approach on an object o_i. The pseudocode of the algorithm is shown in Algorithm 1.

4.3.2.1 Recognize by Color Histogram

The recognition procedure first tries to classify a new object using color histogram clustering since this can be done efficiently. Since color histograms are the least precise of the two features, it is used in two ways: (a) to filter irrelevant images that are unlikely to be an object of interest and hence can be immediately discarded, and (b) to determine if an object can be recognized solely using the color histogram, in which case the SIFT descriptor based matching need not be performed. The procedure is shown in the first four steps of Algorithm 1.

The algorithm first finds the distance between the color histogram of the new object o_i and each cluster centroid, d_{ij}. In step 2, this distance is used to determine a normalized cluster membership metric, R_{ij}, which represents the likelihood that object o_i belongs to cluster j. If the minimum distance of o_i to all clusters is q[1] times larger than the maximum of the distances between all samples to their cluster centroids, o_i will be considered to be an irrelevant object and will be discarded. Otherwise, in Step 4, the algorithm checks to see if the best matching cluster (i.e. the cluster with maximum membership metric) exceeds a pre-determined threshold, T_R. If so, o_i is considered to uniquely belong to the cluster. In this case, the algorithm matches the object to the tag associated with the cluster (e.g. cup, or cellphone), and terminates.

4.3.2.2 Combine the Color and SIFT Features

While the color feature-based classification removes irrelevant images and identifies images that have clear membership in one cluster, there may be a number of images that are close to multiple clusters and cannot be classified accurately. We use the SIFT features to identify such cases. Classification using SIFT features involves three steps. First, a previously observed image that is closest to the centroid of each "nearby cluster" (the cluster j such that R_{ij} is larger than a threshold T_R^L) is chosen as the "representative" image for the cluster. This is done because the clusters were built using color features, not SIFT descriptors, hence we cannot directly use the clusters for SIFT-based classification. Second, the SIFT descriptors for the new image are compared to the representative images for each cluster, and a SIFT cluster membership score is assigned based on the similarity. Finally, a combined score is assigned to the new object based on a weighted combination of the color-based and the SIFT-based membership metric. The object is considered to belong to all clus-

[1]q is an empirically determined constant; for all results reported here, q=2

ters, and assigned all tags for which the weighted score is greater than a pre-defined threshold.

Algorithm 1: Pseudocode of object recognition procedure.

```
Input: object $o_i$
Current model: Sample set $\mathcal{X}$ and $k$ clusters $\{\mathcal{X}_h\}_{h=1}^k$
Parameters: $T_R$, $T_R^L$, $\alpha$, $TH_{tag}$
Method:
1. Calc histogram $x_i$ for $o_i$.
2. Calc membership of $o_i$ to all cluster centroids $\mu_j$:
   $R_{ij} = 1/(d_{ij}^2 \sum_{j=1}^k (1/d_{ij}^2))$, where $d_{ij} = \|x_i - \mu_j\|$.
3. If $\min(d_{ij}) > 2\max(d_{lj})_{l=1,j=1}^{l=n,j=k}$, $o_i \ni \{\mathcal{X}_h\}_{h=1}^k$, exit.
4. If $\max(R_{ij}) > T_R$, $similarity_{ij} = R_{ij}$, goto step 6.
5. (1) For each $\mu_r$ such that $R_{ir} > T_R^L$, do:
        Calc/save SIFT descriptors for $o_i$ and the closest sample $s_r$ to
        $\mu_r$; Calc number of SIFT matching points $M_{ir}$ between $o_i$ and
        $s_r$.
   (2) For the rest $\mu_r$ such that $R_{ir} < T_R^L$, let $M_{ir} = 0$.
   (3) Calc $R_{SIFT}^{ir} = M_{ir}/\sum_{j=1}^k M_{ij}$ for all $\mu_r$.
   (4) $l = \max_r(R_{SIFT}^{ir})$. Let $o_i$, $s_l \to \mathbb{S}$.
   (5) $similarity_{ij} = R_{ij} + \alpha R_{SIFT}^{ij}$.
6. Let $k = \max_j(similarity_{ij})$, $o_i \in \mathcal{X}_k$.
7. For all $j$ that $similarity_{ij} > TH_{tag}$,
   $stringof(\mu_i) \to Tags$.
```

This procedure is shown in Steps 5-7 of Algorithm 1. In step 5, we calculate the number of SIFT matching points between o_i and the representative image s_j corresponding to each cluster. For cluster j, s_j is the closest sample to the centroid μ_j. We can then calculate the number of SIFT matching points M_{ij} between o_i and s_j in each cluster. R_{sift}^{ij} represents the SIFT similarity between o_i and the sample s_j in cluster j. Thus R_{sift}^{ij} represents the evaluation score of the membership of o_i to cluster j given by SIFT features.

The overall evaluation score of the identity of o_i is calculated by combining color and SIFT features: $similarity_{ij} = R_{ij} + \alpha R_{sift}^{ij}$, where α is a weight that reflects the importance of SIFT features. If $similarity_{ij} > TH_{tag}$, then the object is associated with a specific tag, where TH_{tag} is a predefined threshold that balances the false positive and false negative of object recognition. After an object is recognized, the ROO and the entire scene image is stored in the flash memory of Tier-2 node, and metadata about the recognized object (node id, timestamp, tags) is transmitted to the proxy.

4.3.3 Constrained Cluster Updating

Until now, we have discussed how the color-based clusters can be used for classification. We will now describe how the clusters themselves can be evolved in a dynamic manner, by taking into account both information gleaned from the SIFT feature-based matching as well as from user feedback. SIFT feature-based matching provides links between two ROOs. For instance, if the matching score R_{SIFT}^{ij} of two samples i, j is high, it is likely that sample i and j are the same object. In addition, since our object retrieval application interacts with users, we can even get more constraints from user feedback. For example, a user can directly label the class of a sample, or denote if two samples are the same object. We assume in this work that user feedback is always accurate.

4.3.3.1 Constraint Definition

We define two broad classes of constraints - hard-label constraints and pair-link-constraints. The former captures constraints provided by user feedback, where the user labels an image ROO, for instance, as a PDA. The latter captures constraints between pairs of images—for instance, based on SIFT image matching. We now formally define these constraint classes.

Hard-labeled-constraint (HLC) indicates a definite match between a sample and a certain cluster. $\mathbb{H}$ denotes the set containing the HLCs.

Pair-link-constraint (PLC) represents the constraint between pairs of examples. There are three subclasses:

(i) A Must-link constraint indicates that two samples should belong to the same cluster. $\mathbb{M}$ denotes the set containing the must-link constraints.

(ii) A Cannot-link constraint indicates that two samples must belong in different clusters. $\mathbb{C}$ denotes the set containing the cannot-link constraints.

(iii) A Soft-link constraint indicates two samples are probably in one cluster. The evidence of each soft-link constraint is computed by SIFT descriptor matching, and its weight is assigned based on the SIFT matching score R^{ij}_{SIFT}. $\mathbb{S}$ denotes the set containing the soft-link constraints.

4.3.3.2 Constrained k-means

Although clustering algorithms like k-means have the ability to handle hard-label constraints, it is difficult for them to handle pair-link constraints. A few techniques [110, 19, 45] have been suggested in semi-supervised k-means algorithms to address this problem, upon which our approach is based.

Since standard k-means cannot handle pairwise constraints explicitly, the goal of clustering is formulated as minimizing a combined objective function which is the sum of the total distance between the samples and their cluster centroids and the cost of violating the pair-link constraints. The clustering problem can be formulated as minimizing the following objective function, where x_i is assigned to the partition $\mathcal{X}_i$ with centroid μ_{l_i}.

$$\begin{aligned}\Phi = \beta \sum_{x_i \in \mathcal{X}} \|x_i - \mu_{l_i}\|^2 \quad &+ \sum_{(x_i,x_j)\in\mathbb{S}} w^{ij}_S \perp[l_i \neq l_j] \\ + \sum_{(x_i,x_j)\in\mathbb{M}} w_M \perp[l_i \neq l_j] &+ \sum_{(x_i,x_j)\in\mathbb{C}} w_C \perp[l_i = l_j]\end{aligned}$$

in which $\perp$ is the indicator function, with $\perp[true] = 1$ and $\perp[false] = 0$. β is a parameter to trade off the importance of the data set itself with that of the constraints.

The cost of violating a pair-link constraint is given by the weight of this link: w_S^{ij} denotes the weight of the soft-link constraints based on SIFT matching; and w_M and w_C denote the weights on must-link and cannot-link constraints. Since explicit user feedback is more precise than the SIFT-based matching result, we use higher value for w_M and w_C than that for w_s^{ij}.

Algorithm 2: Constrained cluster updating algorithm.

```
Input: A set of old samples $\mathcal{X} = \{x_i\}_{i=1}^{n}$
        The old clusters: disjoint $k$ partitioning $\{\mathcal{X}_h\}_{h=1}^{k}$
        A set of new samples: $\mathcal{X}_{new} = \{x_i\}_{i=1}^{m}$
        Constraint sets: $\mathbb{S}$, $\mathbb{M}$, $\mathbb{C}$, $\mathbb{H}$
Parameters: $\beta$, $w_M$, $w_C$, $w_s$
Method:
1. Load the cluster configuration $\{\mathcal{X}_h\}_{h=1}^{k}$
2. Repeat until convergence:
  (1) Assign all sample with HLC: For the sample $(x_i \to j) \in \mathbb{H}$ ,
      directly assign $x_i$ to cluster $h_j$. For the sample $(x_i \nrightarrow j) \in \mathbb{H}$,
      assign it to the closest cluster $h$ such that $h \neq j$.
  (2) Assign each other sample $x_i$ to the cluster $h_L$, for
      $$h_L = \arg\min_h(\beta\|x_i - \mu_h^{(t)}\|^2 + \sum_{(x_i,x_j)\in\mathbb{S}} R_{SIFT}^{ij}\perp[h \neq l_j] + \sum_{(x_i,x_j)\in\mathbb{M}} w_M\perp[h \neq l_j] + \sum_{(x_i,x_j)\in\mathbb{C}} w_C\perp[h = l_j])$$
  (3) Estimate and update means:
      $\{\mu_h^{(t+1)}\}_{h=1}^{k} \leftarrow \{\frac{1}{|\mathcal{X}_h^{(t+1)}|}\sum_{x\in\mathcal{X}_h^{(t+1)}} w_s x\}_{h=1}^{k}$
  (4) $t \leftarrow (t+1)$
3. Delete a set of the oldest samples from the clustered data.
```

Algorithm 2 shows the cluster update algorithm. The algorithm alternates between the cluster assignment and centroid estimation steps. When doing the cluster assignment, every sample x_i is assigned to a cluster such that it minimizes the sum of the distance of x_i to the cluster centroid and the cost of constraint violations caused by that assignment. The centroid re-estimation step is the same as standard k-means algorithm.

The proof of the convergence property of our algorithm is similar to the proof in [20]. In our algorithm, the pairwise constraints are given only by SIFT features and user feedback, which are not explicit functions of the centroid, so in re-estimating the cluster centroid μ_h, only the component $\sum_{h=1}^{k} \sum_{x_i \in \mathcal{X}_h} \|x_i - \mu_h\|^2$ is minimized. Hence the objective function decreases after every cluster assignment and centroid re-estimation step. Therefore our algorithm will finally converge to a local minimum of Φ. We give samples with hard constraints more weight in the centroid re-estimation step.

The computational complexity of k-means is $O_{(nkd)}$, where n, k, d represent the number of data points, number of clusters, and dimensionality respectively. The algorithm is computational efficient since the complexity is linear in the size of the input.

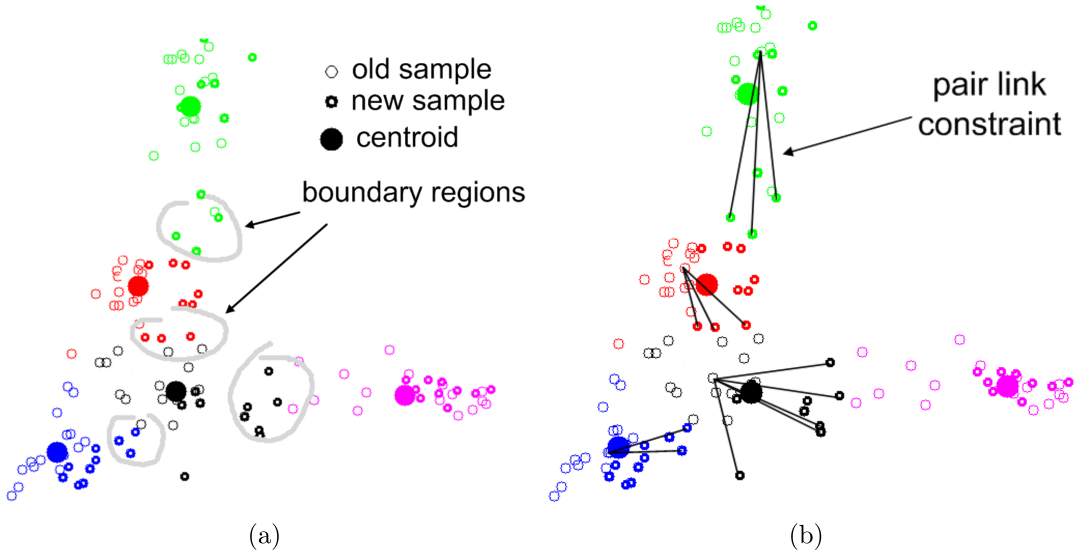


Figure 4.4. An example of clustering under constraints. Color is used to identify different clusters. To represent clusters, two dominating dimensions are calculated by Principal Components Analysis and used as x, y coordinates.

An illustration of constrained cluster updating is shown in Figure 4.4, in which each color represents an actual cluster. Due to slight illumination change, there is a small shift between new samples and old samples in each cluster. As seen from

Figure 4.4 (a), the new samples in the "boundary regions" between clusters may be incorrectly assigned due to the shift. Figure 4.4 (b) shows that by using pair-link constraints to update clusters, the centroids shift towards the new samples so that the cluster model represents the new samples better. In this way, our system is reactive to changes in illumination. The cluster update algorithm is performed infrequently on iMote2 when a sufficiently large number of constraints have been accumulated, hence the computational overhead of the approach is not significant.

4.4 Object Retrieval in Proxy Node

Our system employs a proxy node to organize the information from the dual-camera sensor nodes, as well as process user queries.

4.4.1 Event Database

The proxy node maintains a database of event messages sent by sensor nodes. Each time a sensor node detects and recognizes an object, it sends an event notification to the proxy. In the event that multiple overlapping cameras are placed to cover an area of interest, it is possible that multiple nodes can detect and recognize the same object, thereby suppressing false negatives. To merge the recognition results from multiple nodes, the proxy combines event messages with similar timestamps, and stores it in a local database for future retrieval. Note that consistent timestamps can be obtained by using a network-wide time synchronization protocol such as FTSP [78]. Table 4.1 shows an example of the stored items in the database in the proxy-node, where "Global ID" represent the global sequence number of the event, "Node-Addr (Local-ID)" indicates the address of the nodes detecting this event along with the local event index in the detecting node. "Tags" is a set that is the intersection of the recognition results of those nodes that see the same object. "Timestamp" is the average time of the same event detected by multiple sensors.

Table 4.1. Database in the proxy node

Global ID	Node-Addr (Local-ID)	Tags	Timestamp
n	2 (14)	Key; PDA	2007-10-3-22:25:50
n+1	2 (15)	Book	2007-10-3-22:31:27
n+2	2 (16); 3 (15)	Book; Cup; PDA	2007-10-3-22:38:54

4.4.2 Tag-based object retrieval

Our system provides a tag-based object retrieval capability. The user can provide the name of a tag or class as the command to retrieve the latest location of this object. The retrieval process is performed in an interactive manner. The proxy first searches for the query string (object name) in the *Tags* field of the local database. The proxy locates the latest item whose *Tags* field contains the query string, and sends an "ROO Request" message to the appropriate sensors in the NodeID field to retrieve the ROOs of interest. Each node that receives this request compresses the candidate ROO image in JPEG format and transmits it over the wireless radio to the proxy.

The candidate ROO sent back by the sensor node is shown to the user using an easy-to-use GUI. If the user marks this ROO as "Valid", i.e., confirms that it is, in fact, the queried object, the proxy sends an "Image Request" command to the sensor node and a full image containing the ROO will be transmitted back to the proxy and shown to the user. Otherwise if the user marks this ROO as "Invalid", it means the ROO is not the queried object due to a false positive. The proxy will continue to search through its database to locate an older item that matches the user query. This process is repeated until an ROO is accepted by the user or there are no more entries in the database. Such an interactive retrieval approach ensures that we don't transfer an entire scene image unless we are sure that it contains the queried object, thereby saving time and energy.

The user feedback also provides constraints that can be exploited for better clustering, as described in Section 4.3.3.2. In addition to the "Valid/Invalid" marking, users also have the option of correctly labeling a candidate ROO, or indicating if two candidate ROOs are the same object or not. This information is periodically fed back from the proxy to the appropriate sensor nodes, which use them to update the cluster model.

4.5 System implementation

This section describes the implementation details of our system based on the design discussed in previous sections.

4.5.1 Hardware implementation

Part-1: Part-1 comprises of a Cyclops camera [94] connected to a MICAz [3] mote. The Cyclops is constructed from an Agilent ADCM-1700 CMOS camera module, a Xilinx FPGA and an ATMega128 microcontroller. The Cyclops communicates with MICAz via I2C bus and uses a 2.4GHz CC2420 radio chip as the wireless component.

Part-2: Part-2 node is a combination of an Enalab camera [4] and an iMote2 [5]. Enalab camera module comprises an OV7649 Omnivision CMOS camera chip, which provides color VGA (640x480) resolution. The iMote2 is assembled from an 18-400MHz Xscale PXA271 processor and a CC2420 radio chip. The Enalab camera is connected to the Quick Capture Interface (CIF) on iMote2. To support large image data storage, a 1GB external flash memory is attached.

The Part-1 node and the Part-2 node are connected with a trigger circuit for wakeup, and communicate through the serial port. The Cyclops camera and the Enalab camera are mounted close to each other in order to increase the accuracy in inter-tier ROO mapping. The sensor nodes are powered by batteries.

Proxy node: In the prototype system a computer running Linux is used as the proxy node. An iMote2 node is connected to the proxy node and acts as the network gateway.

4.5.2 Software environment

The software environments in our system are different on the different tiers. In Part-1, both the MICAz and the Cyclops run TinyOS 1.1.14. We enhanced the object detection software available for the Cyclops to perform still object detection. The Part-2 iMote2 runs Arm-Linux. The OpenCV library [6] is used on the iMote2 to facilitate the basic image computations, such as image conversion, transformation and color histogram computation. Our SIFT algorithm is based on the SIFT++ lib, which is a lightweight C++ implementation of SIFT descriptors. The Intel Integrated Performance Primitives library (IPP) is used to accelerate data processing. A JPEG compressor was also developed using IPP lib to compress images. The IEEE 802.15.4 radio protocol is used to communicate among all nodes in the system.

4.6 Experimental results

We evaluated the performance of our object retrieval system through an extensive set of experiments. We first evaluated the benefits of using a dual-camera sensor node instead of a single camera node. We then evaluated the power consumption and performance of the object detection, and object recognition algorithms individually, and finally provide a full system evaluation using multiple cameras in a realistic environment.

There are a number of key parameters in our system most of which are empirically determined: T_R, T_R^L, α, TH_{tag} for Algorithm 1, and β, w_M, w_C, w_s for Algorithm 2. In all our experiments, T_R, T_R^L were fixed and set to 0.7 and 0.2 respectively. These values excluded samples in the "boundary regions" (as seen in Figure 4.a). α

is set to 1.2 to tune the trade-off between color and SIFT features. TH_{tag} is set to 0.3 in all experiments except in Section 7.5 where we evaluate the impact of tuning this parameter. For algorithm 2, β is set to 0.02 so as to give more weight to the constraints. The experimental results are not very sensitive to the parameters w_M and w_C, as long as they are assigned a value larger than 10. In our experiment we set $w_M = w_C = 10$. The parameter w_S is also not a sensitive parameter and is set to 2.

4.6.1 Energy Cost of Object Detection

In order to provide a better intuition for the energy gains offered by our system, we compare our system with a single-part design that keeps the Part-2 node (iMote2 + EnalabCam) always on to perform the detection. In our system, the Part-1 node (MICAz + CyclopsCam) will wake up the Part-2 node (iMote2 + EnalabCam) after every 4 still objects are detected. We also present the power consumption for two operational modes of the CyclopsCam - a "duty-cycle" mode and an "always on" mode.

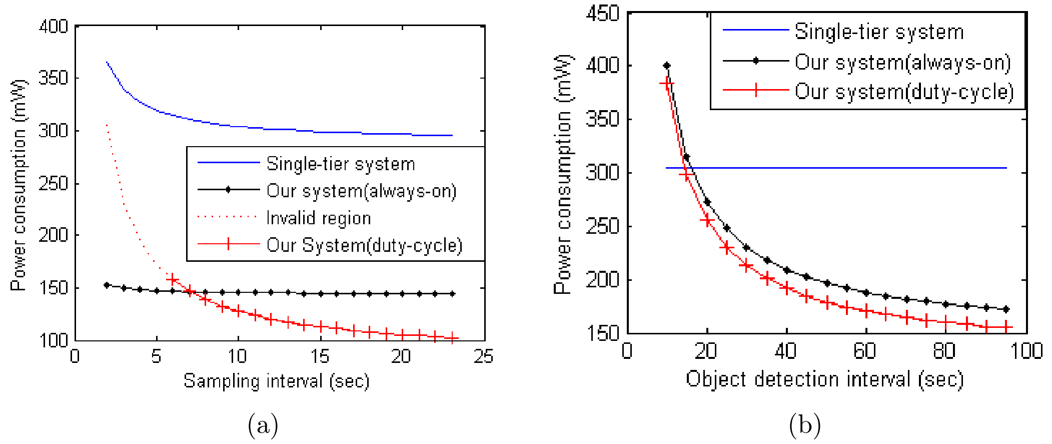


Figure 4.5. Power consumption analysis. (a) Effect of the sampling interval. (b) Effect of object detection interval.

Figure 4.5 (a) shows the power consumption for continuous monitoring as a function of the sampling interval. In this experiment, no object is detected, and consequently the iMote2 does not need to be woken up. As seen from the figure, both operation modes of our system consume less energy than the single-tier version, clearly demonstrating the benefits of using a tiered system for object detection. The experiment also reveals that the critical point for choosing between the two modes of the CyclopsCam is around 6 seconds. Thus, when the sampling frequency is less than 6 seconds, the always-on mode of the CyclopsCam is more efficient since the energy consumption for transitioning the camera from sleep to wake state dominates the total power consumption.

Figure 4.5 (b) evaluates the effect of object detection interval (i.e. the average time between two consecutive object detection events) on the power consumption of the three schemes. In this experiment, the sampling interval is fixed to 10 seconds. As shown in the figure, if the still object is detected very frequently (less than 20 seconds between detections), the power consumption of our system may be a little larger than that of the single-part version, because of the frequent wake-up overhead of the iMote2 node. For most reasonable inter-object intervals, the power consumption of the two versions of our system is considerably less than that of the single-part version.

4.6.2 Accuracy of object mapping

Obtaining an accurate mapping between the CyclopsCam's ROO and the EnalabCam's ROO is essential to the performance of our system. We compare the error in ROO estimation for two schemes: (a) an always-on single-part system that uses the iMote2 and EnalabCam, and (b) our system using inter-part wakeup and ROO mapping. The errors are calculated by comparing the object region produced by the algorithms to those labeled manually. The experimental results in Table 4.2 show that our system has only marginal higher error (less than two pixels along each axis) than a single-part system that uses a high resolution camera. In addition, the absolute error is less than three pixels on each axis.

Table 4.2. Object detection error analysis. Errors are measured in pixels.

	Center Error	Width error	Height Error
Test	Avg (Var)	Avg (Var)	Avg (Var)
EnalabCam	1.04 (0.88)	1.70 (4.90)	1.30 (4.46)
DualCam	2.47 (2.08)	2.84 (4.13)	2.91 (6.07)

4.6.3 Comparison between color and SIFT

In this section, we demonstrate the accuracy and energy benefits of using the Color and SIFT features in a cascading manner for object recognition (described in Section 4.3.2.1). Table 4.3 compares the performance of three different recognition methods: only color features, only SIFT features, and the cascading combination of the two. We use two metrics to evaluate the schemes—rate and latency. The "rate" metric represents the percentage of correctly recognized objects, and the "latency" metric shows the amount of time taken by the iMote2 node for object recognition, which in turn corresponds to the energy consumption for recognition. Five common objects are used in this test: book, cup, keyring, PDA, and TV remote control. These are all common objects that are easy to lose.

We tested the three methods with a training set and a test set. The training set contains 50 samples. The test set, FIXED-ILLUM, contains 100 samples under the same lighting conditions as in the training set. The set VARY-ILLUM contains 100 samples where illumination changes were introduced by switching off one of the three ceiling lights in the room. Samples are collected by placing the objects in different locations and with different poses on a table. We first train the clusters by using the training dataset, and then classify the two test sets respectively.

Table 4.3 shows that the recognition rate by using a combination of Color and SIFT features has higher accuracy than using the features individually. When illumination changes are present, the improvement in accuracy over color alone is 17%. The computation time for processing is significantly greater than for a method that just uses color features, but is only a third of the time required when only the higher accuracy SIFT descriptors are used. This is because our algorithm needs to run the SIFT algorithm only on roughly 20-40% of the samples. Note that cluster updates were not performed in this experiment.

Table 4.3. Comparison of recognition methods (The power consumption during object recognition is 681mW)

	FIXED-ILLUM		VARY-ILLUM	
Methods	Rate	Latency	Rate	Latency
1. Color	84%	0.0034s	63%	0.0032s
2. SIFT	83%	6.49s	75%	6.17s
3. Color + SIFT	91%	2.07s	80%	2.96s

4.6.4 Benefits of using constraints

We now evaluate the benefits of using pair-link constraints to improve clustering results. In this experiment, we collect another test dataset, VARY-ILLUM-1, that contains 100 samples under the same illumination conditions as in VARY-ILLUM. We then evaluate whether the recognition rate on VARY-ILLUM-1 improves as a result of refinement of the clusters with constraints that are obtained from VARY-ILLUM in the previous experiment. Table 4.4 shows the results of this experiment. The "No constraint" column shows the results using only standard k-means without constraints, and the "Using constraint" column shows the results of our constrained k-means algorithm with refinement using constraints derived from VARY-ILLUM. As seen from Table 4.4, the use of constraints improves the recognition results by 10% when only color features are used, and by 6% when both color and SIFT features are used. The refinement of clusters also improves the latency required to perform the object recognition by about 20%, since the clusters are more accurate and hence the SIFT recognition algorithm is invoked fewer times.

Table 4.4. Improvements from using constraints (The power consumption during object recognition is 681mW)

Recognition methods		VARY-ILLUM-1	
		No constraint	Using constraint
Color	Rate	68%	78%
Color + SIFT	Rate	77%	83%
	Latency	2.89s	2.24s

The recognition results above are all produced by a single camera node. We also evaluate the benefits of placing multiple sensor nodes with overlapping coverage for object recognition. In our experiment we evaluated the recognition rate using two camera views and found that the recognition rate improves from 82% (single view) to 86% (two views).

4.6.5 System performance on object retrieval

We now evaluate the overall performance of our object retrieval system using an experiment in a real room environment with multiple sensor nodes. In this experiment, we placed 5 dual-camera nodes in a room so that the FOVs of these nodes cover most of the area in which human activity may happen. The cameras are placed in an ad-hoc manner, so some cameras have overlaps in their field of view. Figure 4.7 shows the deployment of the camera network. We use the same object set as previous experiments: book, cup, key ring, PDA, and TV remote control. In this experiment, objects are randomly placed and removed from the monitored area. Queries for each object are generated after roughly every 20-25 object placements events.

Table 4.5 gives the results from this experiment. The first column labeled "Correct/Total" stands for the ratio of number of correctly retrieved images to the number of queries. A correct retrieval is the case where the system returns the latest scene

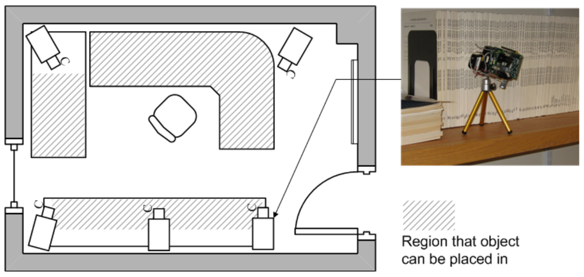


Figure 4.6. Deployment of the dual-camera sensor nodes.

image containing the queried object. "Average ROO Transmitted" denotes the average number of candidate regions that need to be transmitted to get the retrieval result. As seen from the table, in the absence of user feedback, our system achieves 90% accuracy in ROO retrieval, with less than four candidate regions retrieved per correct retrieval. If users provide additional feedback by labeling returned candidate ROOs, the accuracy increases to 95%, and the average number of ROOs need to be transmitted for each query is reduced from 3.6 to 2.5 (a reduction of 7KB in bytes transmitted).

Table 4.5. Object retrieval performance (The tagging threshold TH_{tag} is fixed to 0.3)

	Correct/Total	Average ROO (data bytes) Transmitted
No user feedback	18/20	3.6 (22.5KB)
With user feedback	19/20	2.5 (15.6KB)

4.6.5.1 Impact of Tagging Threshold

The correct rate and the number of ROOs that are transmitted are sensitive to the value of threshold TH_{tag}. As described in Section 4.3.2.2, TH_{tag} influences the number of false positives and number of false negatives in object recognition, and also determines the number of category tags saved for each ROO. Figure 4.7 illustrates the effect of changing TH_{tag} and shows that the threshold provides a tradeoff between

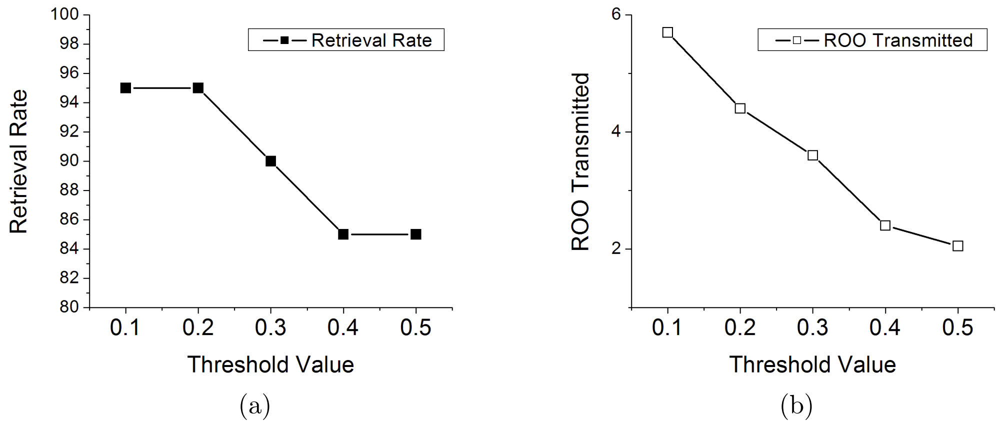


Figure 4.7. Effect of the tagging threshold TH_{tag}. (a) Retrieval rate. (b) ROO transmitted.

the number of ROOs transmitted against the accuracy of retrieval. If available energy in the system is limited, a higher value can be used for TH_{tag}, which will reduce the number of ROO images transmitted, but will also decrease the correct rate of retrieval. The experiment result shows that our system functions well in real world settings with multiple camera nodes, and can be tuned to tradeoff recognition accuracy for amount of energy expended for communication.

4.7 Summary

This chapter presents the design and implementation of an indoor object retrieval system using a network of dual-camera wireless camera nodes, each of which combine multiple cameras with complementary capabilities. Our system proposes a number of novel techniques including: (a) the use of the low-power camera both for still object detection as well as region-of-object estimation, (b) the use of two different visual features—color histogram and SIFT descriptors—for energy-efficient yet accurate object recognition, and (c) refinement of clusters for more accurate object classification using pairwise constraints from SIFT matching and user feedback. Our experimental results demonstrate that the system is energy efficient, computationally efficient, and accurate.

Although the proposed tracking-based approach achieves reasonable results and has been demonstrated as effective in assisted-living, it still suffers a success-rate problem. As described in the experimental session, it in average has a failure rate of 10%, which may cause problems in an assisted-living scenario. More importantly, it usually suffers from incomplete coverage. To improve the effectiveness of the object search application, other sensorimotor resources, like PTZ cameras, RFID sensors and mobile robots can be employed; this is addressed in the next chapter.

CHAPTER 5
SINGLE-AGENT ACTIVE SEARCH

In Chapter 4, a tracking-based object retrieval approach is proposed. In spite of its advantages of low power consumption and persistent monitoring, the tracking-based approach suffers from detection failures and incomplete coverage. To cover a larger area and to achieve more effective object search performance, the system can employ more resources. One such resource is a mobile robot embedded in a sensor array to perform active search.

In this chapter, an object search approach for a single agent is presented and its performance is evaluated. Building a smart agent with efficient search capability is a prerequisite to building a cooperative robot search team (which will be presented in the following chapters). In our system, each search agent can be seen as an expert when searching alone and is able to independently perform a complete exploration of the environment, acquire observations and decide if the target is detected in a certain region. In this chapter, a general framework is described that can be adapted by different types of agents. Two types of agents are considered and studied: (i) PTZ camera nodes and (ii) the uBot-5 robot [36].

5.1 Bayesian Searching Problem

The search problem for a single agent can be represented in a Bayesian framework [27]. For a target r, the state vector of its location $\vec{x}_r \in \mathcal{X}_r$ in Cartesian space can be expressed in the form of a Probability Density Function (PDF) $p_r(\vec{x})$. Given a prior PDF $p_r(\vec{x}_0|z_0) \equiv p_r(\vec{x}_0)$ of the target and the independent observations z, the

PDF at time step t can be constructed recursively using Bayes' theorem. In the application of object search to living space, it is reasonable to assume that when the search process starts, the target object is stationary and not allowed to move until the search finishes. So we have $p_r(\vec{x}_t|z_{1:t-1}) = p_r(\vec{x}_{t-1}|z_{1:t-1})$. After each observation, the PDF will be updated according to the observation,

$$p_r(\vec{x}_t|z_{1:t}) = Kp_r(\vec{x}_{t-1}|z_{1:t-1}) \cdot p_r(z_t|\vec{x}_t) \tag{5.1}$$

where K is the normalization factor and is given by,

$$K = 1/\int [p_r(\vec{x}_t|z_{1:t-1})p_r(z_t|\vec{x}_t)]d\vec{x}_t \tag{5.2}$$

The search system is designed to maximize the chances of finding the target given a restricted amount of time. Although using a longer time horizon can achieve a better solution, planning with a "one-step-lookahead" strategy [27, 118] that maximizes $p(Z_t|z_{1:t-1})$, where Z_t represents a "detection" event at time $t-1$, can provide reasonable performance with very low computational overhead.

5.2 Search Strategy

The single agent search task consists of four subtasks: (i) the subtask Plan is the selection of the next action and the corresponding position in $\mathcal{R}^2$ for the mobile robot or pan-tilt-zoom parameters for the PTZ cameras so as to bring a potential search sub-area into the sensing range of the agent; (ii) the subtask Move involves controlling the hardware to realize the planned state; (iii) the subtask Observation involves the procedure for detecting the target, and (iv) when the target is detected in subtask Observation, the subtask (Localization) is performed to drive the agent to approach the target, to localize the target and to broadcast the detection event. The overall local search process is illustrated in Figure 5.1. The agent repeats the subtasks of action planning, manipulation, and observation to explore all the visible area.

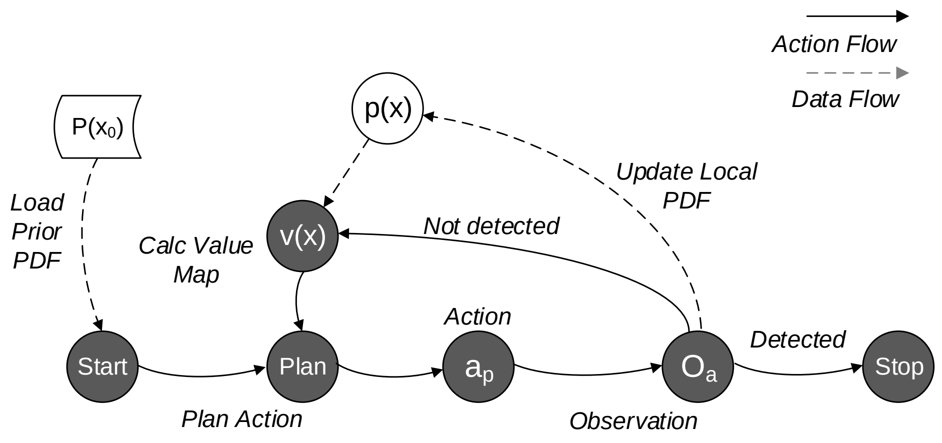


Figure 5.1. The search strategy for a single agent.

5.2.1 Subtask Plan

We assume that the geometric configuration of the search space is $\mathcal{C}_W$, and objects can be placed only on the floor or on tables of uniform height. The horizontal planes of the search region are tessellated into a two-layer occupancy grid $\mathcal{G}$ where the centers of the grid nodes $g_i \in \mathcal{G}$ are candidate positions to be observed.

Given $\mathcal{C}_W$, each agent c calculates a local visibility map $M_c(\vec{x}) = \{0, 1\}$ which indicates if a grid node g_i is visible to the camera agent or not (e.g., g_i is invisible for a PTZ camera node if g_i is located outside of the limitation of the pan/tilt/zoom parameters or is blocked by an occluding object). The local PDF map $p_{c,r}(\vec{x_0})$ for agent r is initialized by re-normalizing $p_r(\vec{x_0})$ in all visible areas.

$$p_{c,r}(\vec{x_0}) = N p_r(\vec{x_0}) \cdot M_c(\vec{x}) \tag{5.3}$$

where N is a normalization factor. The local PDF map is the core data maintained by each agent. After $p_{c,r}(\vec{x_0})$ is calculated, the agent is ready to perform search.

In our approach each agent uses a “one-step-lookahead” strategy to plan the next action. The action space of an agent consists of all the manipulations that bring a visible grid node to the center of the camera image. To do action planning, a utility value u_i is calculated for each grid node indicating the benefit of visiting this node. Given the local PDF of the target, u_i can be calculated by two different strategies,

Strategy 1 (S1): Using Expected Information Gain.

The expected information gain $u_{info}(i)$ is calculated for each grid node i indicating the possibility of detecting the target object by visiting this node. Given the local PDF of the target, $u_{info}(i)$ can be calculated by,

$$u_{info}(i) = \sum_{g_k \in \mathcal{V}_i} p_{c,r}(x_k) \tag{5.4}$$

where $\mathcal{V}_i$ is the set of all grid nodes in the *observation field*, i.e., that can be observed when agent is visiting grid node g_i.

Strategy 2 (S2): Using Expected Information Gain and Travel Cost.

In this dissertation we propose to use this strategy to consider not only the information gain but also the travel cost.

$$u_{cost}(i) = f_h(x_c, x_i) \tag{5.5}$$

where $f_h(x_a, x_b)$ is the length of the harmonic function path [33] from x_a to x_b.

Given the two criteria, the utility function $u(p)$ is calculated as:

$$u(p) = w \cdot u_{info} + (1 - w)u_{cost} \tag{5.6}$$

The set of actions selected is that set which results in the agent visiting the grid node with highest value v_i.

5.2.2 Subtask Move

After the grid node i with highest utility value is calculated, the agent drives itself to the grid node i. For PTZ camera agents, the Pan/Tile/Zoom parameters $(\alpha, \theta, \tau) = f(\vec{x}_{g_i})$ are calculated according to the 3D position of the selected grid node and the camera. To get similar observation fields, the zoom value τ is proportional to the distance between the visited grid node and the camera. For the mobile robot (uBot-5 in this dissertation), harmonic function navigation [33] is used to drive it in $\mathcal{R}^2$ space to the grid node i.

5.2.3 Subtask Observation

Object detection in the image can be achieved using a variety of methods, from vision based methods to RFID based detection. The active search part (Chapter 5, 6, 7) of this dissertation is focused on describing a generalized approach and the discussion of particular object detection and recognition algorithms that might be appropriate is beyond the scope of this dissertation. In this dissertation two object detection approaches are considered and used for evaluation, the vision-based approach that detects the target object using the camera sensors, and the RFID-based approach that detects the target object using the RFID receiver mounted on the robot.

5.2.4 Localization

When a target is detected as a result of successful execution of the second and third subtasks, the last subtask (LOCALIZATION) is performed to approach the target, localize it and to announce the detection.

There are many ways to localize the detected object, ranging from visual triangulation methods to RFID active localization methods (e.g., Sherlock [4] proposed a technique to refine the RFID localization of objects in an office environment). Here, we only consider and use visual methods. After the object is detected, the position of the object can be located using a depth-sensing camera (e.g., Microsoft Kinect sensor) that is mounted on the robot, or can be triangulated using a pair of the PTZ cameras. If located by a mobile robot, the robot converts the object's position in its coordinate system to the world coordinate system. If a precise 3D location of the object is required, the calibrated PTZ camera array can be used to localize the target object accurately.

As the result of a query, the system can simply return an image containing the queried object to the user with the bounding box labeled.

The search process terminates when the agent performs the LOCALIZATION or completes the exploration of its entire action space.

5.3 Evaluations

In this section we present the experimental results of the proposed methods. The first experiment scenario is called *LPR-1*, in which human search behavior is collected and studied. After this, a robot search (both in simulation and in the real world) experiment is conducted in this same environment and compared to the human search statistics.

5.3.1 *LPR-1* Scenario

To evaluate the performance of the active search strategies, an experimental scenario *LPR-1* is defined. In *LPR-1*, the search space is a mock apartment, which is 42×28 square feet. The actual search space is in the Laboratory of Perceptual Robotics (LPR) at UMass Amherst. A photo of the search space is shown in Figure 5.2.

5.3.2 Human Data in LRP1 Scenario

To evaluate the search performance of the robot system, we need to understand how humans search for objects. For this human search experiment, 10 subjects were recruited. Among these participants, 5 were colleagues of the author and are familiar

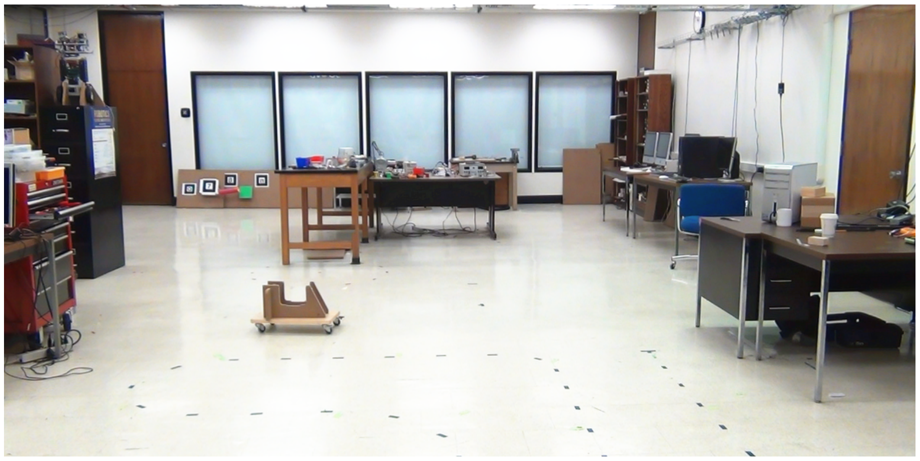


Figure 5.2. *LPR-1* testing space.

with the lab environment. The remaining 5 subjects were unfamiliar with the search room and received a short description of the room configuration and furniture. Their ages range from 26 to 32. Two search activities were considered, (1)searching for a *BOOK* and (2)searching for a *CLAMP*. The participants are asked to search for a designated target objects in each trial. Each subject performs each search task 10 times (5 for the *BOOK* and 5 for the *CLAMP*).

The furniture in this search space include desks and shelves. In each trial of the experiment, the target object was placed in a new location. Different objects have different prior distributions. The target objects were placed locations where one might reasonably expect to find them; the *BOOK* was placed on random shelves while the *CLAMP* was placed randomly on the tables. This information was not exposed to the subjects. They only use common sense to search for the object. The human search trajectories were recorded, and are shown in Figure 5.3. Figure 5.3 (a) illustrates the search trajectories from the 10 subjects when they were searching for the *BOOK*. The black circles are places where the human subjects moved slowly and spent long time to search carefully, these are called “dwell positions”. The state when the subject is in the dwell position in the search process is called the “dwell state”. Figure 5.3 (a) shows that the dwell positions overlap the places where the shelves are, which are the actual position the *BOOK* is hidden. It shows that it is very efficient for human to search using expectations. It can also be seen from the figure that different subjects have similar search behavior. The same results can also be seen in Figure 5.3 (b), which shows that the subjects dwelled around the desks to search for the *CLAMP*.

The statistics of the human search results are shown in Figure 5.4. The average time cost for searching the *BOOK* is 109 seconds and the average time cost for searching the *CLAMP* is 135 seconds.

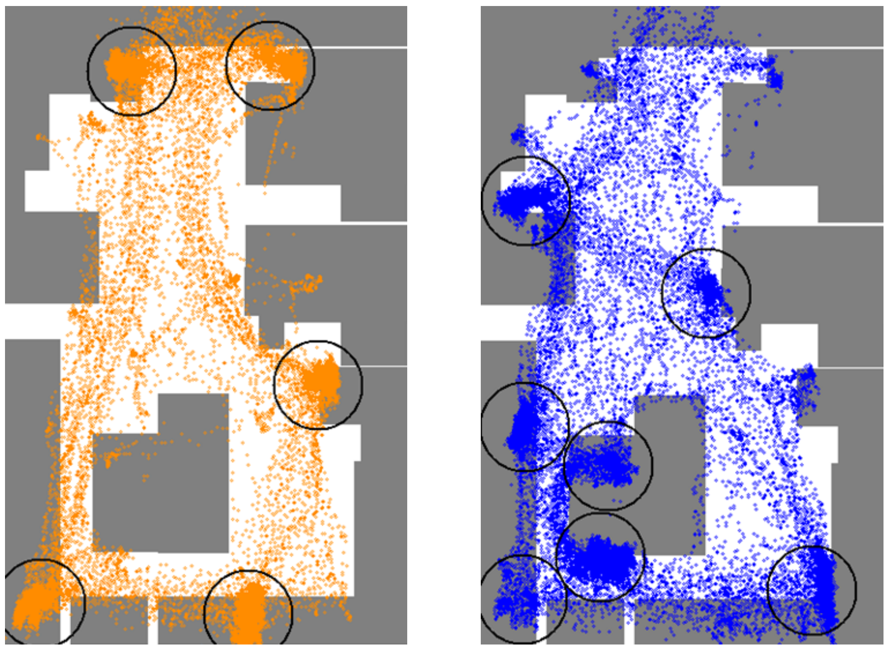


Figure 5.3. Trajectories (orange and blue dots) and dwell locations (black circles) when human subjects search. (a) Search for object *BOOK*, (b) Search for object *CLAMP*. Shadowed areas represent obstacles (desks and shelves).

5.3.3 Simulated Experiments

A set of simulations were conducted to evaluate the performance of the single agent search strategy. The experiments were performed using a simulated uBot-5 in a simulated LPR environment developed with Microsoft Robotics Development Studio

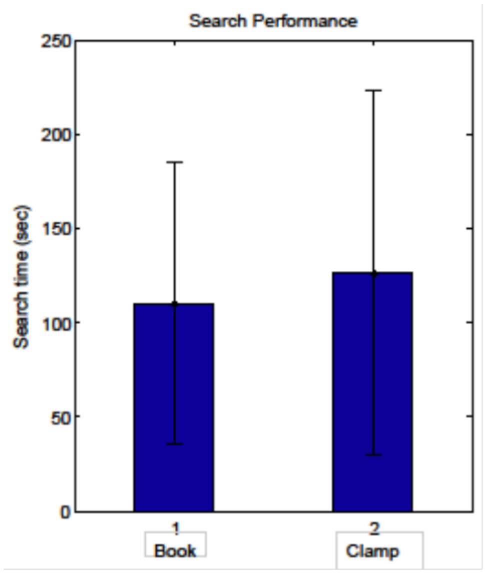


Figure 5.4. Human search performance when searching for the BOOK and the CLAMP

(MRDS) [54]. The simulated environment is an replica of the real world environment. Objects are placed in the search space randomly according to the distribution derived from human search data. The physical and simulated uBot-5 are shown in Figure 5.5. To accelerate the experiments, the moving and observation speed of the robot was accelerated so that the robot searches faster than in the real world and hence these results should be be compared to the real world performanace directly.

In this section two search strategies for a single robot are compared. (1) *Strategy 1 (S1)*: Utility function contains Expected Information Gain. (2) *Strategy 2 (S2)*: Utility function contains Expected Information Gain and Travel Cost.

It can be seen in Figure 5.6, the strategy that considers the travel cost (*S2*) has better search performance. In Figure 5.6 a step is equivalent to a second. The uncertainty reduces faster with *S2* than that it does when using only *S1*. To achieve an uncertainty threshold of 0.1, *S2* uses 22 steps and *S1* uses 37 steps. The experiment demonstrates that using the factor of travel cost in the utility function effectively

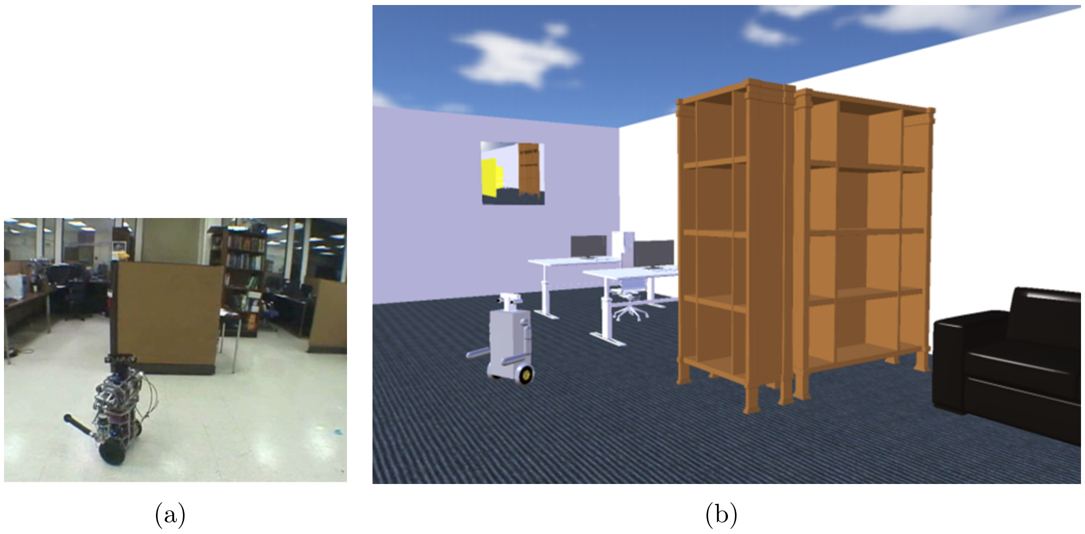


Figure 5.5. (a) The physical uBot-5 and the search space. (b) The simulated uBot-5 and the simulated environment.

improves the search efficiency. Therefore *S2* is used as the basic single agent search strategy in the following discussions.

5.3.4 Real-World Experiments

The real world experiments were conducted using the physical uBot-5 robot. uBot-5 is equipped with two types of sensors, an RFID reader and a depth camera. The depth camera is the ASUS Xion Pro sensor, which is similar to the Microsoft Kinect sensor. In this experiment, the RFID sensor is used to detect the target object. The uBot-5 carries a Skyetek M9 UHF reader and a directional antenna (shown in Figure 5.7 (b)). RFID tags were attached on the target objects. The uBot-5 navigates using harmonic function navigation [33] in $\mathcal{R}^2$ space.

To compare the performance of the physical robot search, the trials were performed in the *LPR-1* experimental scenario. The target object *BOOK* was put in the same location as in the human search experiments. The data was collected from 10 trials of the real world search. The average time cost is 193.7 seconds. Figure 5.8 illustrates the time cost comparison of the human search and the robot search. It can be seen

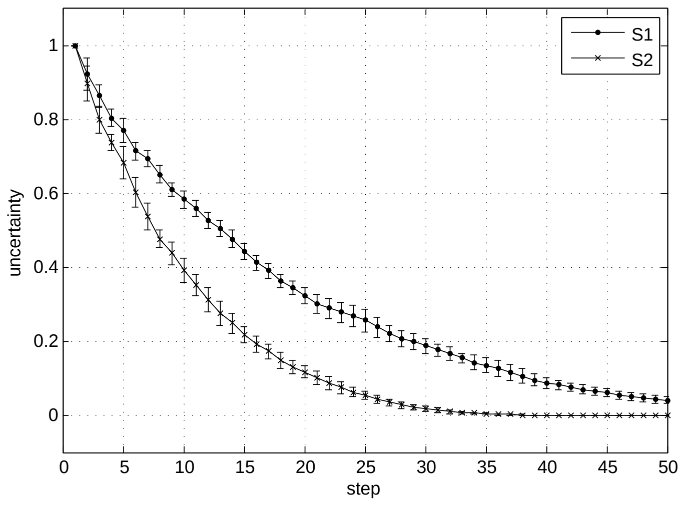
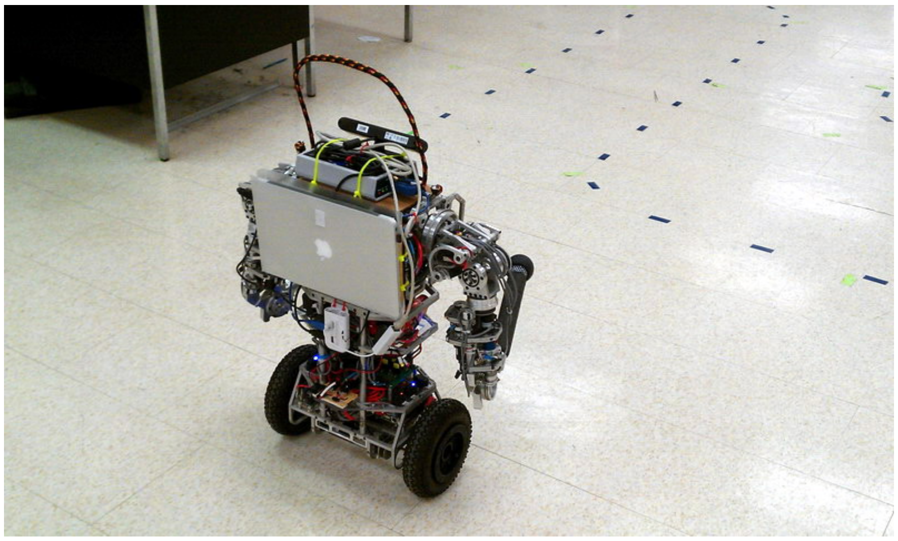


Figure 5.6. Single robot search strategies. *S1: Expected Information Gain. S2: Expected Information Gain + Travel Cost.*

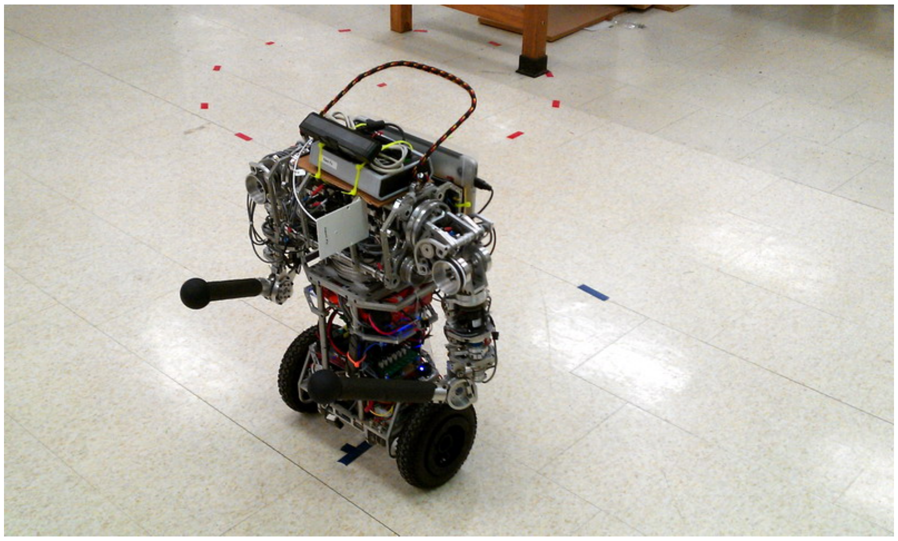
that on average the single robot search is slower than the human search, but is still useful. In many cases in residential assisted-living, it is tolerable for the robot to spend about 3 minutes to search the whole room.

5.4 Summary

In this chapter the search strategy of a single agent robot is presented. A PDF based Bayesian schema is proposed. While our study demonstrates that the single robot search strategy is efficient and useful, there is still room for improvement. By accommodating more sensorimotor resources, an agent team can be formed (tier-3 in the search application hierarchy in Figure 1.4) and better performance in the object search task can be obtained. In the next chapter multi-agent cooperative search strategies are proposed and discussed.



(a)



(b)

Figure 5.7. uBot-5 robot. (a) An 11-inch MacBook Air computer is mounted on the back of the robot and used as the central processing computer. (b) An ASUS Xion Pro sensor and a Skyetek M9 UHF RFID reader are mounted on the robot.

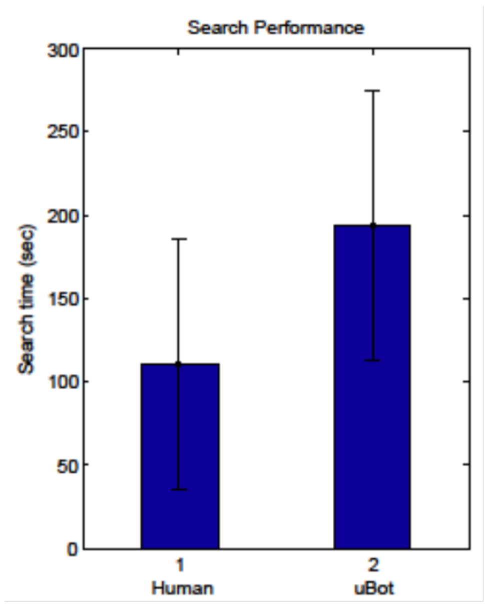


Figure 5.8. Performance comparison of the human subjects and the uBot-5 when searching the *BOOK*.

CHAPTER 6
MULTI-AGENT COOPERATIVE SEARCH

As described in the last chapter, single agent search performance has been demonstrated as efficient and (tolerably) low performance than human search. Subject to limited background modeling for objects and rooms, the uBot-5 is an accomplished autonomous searcher that can be useful in the assisted-living environment. In search operations, a team of intelligent agents can provide a robust solution with greater efficiency than can be achieved by single agents. In our framework, the functionalities of all the agents, including the uBot-5 and the PTZ camera nodes, are wrapped in services and can be invoked to join the search team opportunistically. This chapter focuses on cooperative search strategies in a robot team.

There has been successful demonstrations of synchronized control in multi-agent search [26, 37]. However, for a multi-tasking system that deals with resource contention, pre-emption, and node/network failures, a decentralized and asynchronous version has clear advantages. For example, in our application the PTZ camera nodes perform multiple tasks including object search and people tracking. Figure 6.1 and Figure 6.2 show two examples of our system. In Figure 6.1, all PTZ camera agents were performing object search, while in Figure 6.2, two camera agents were performing human tracking and the other two cameras were searching for an object. One of the searching camera agent even lost its signal due to connection failure. In the assisted-living scenario, people tracking sometimes has a higher service priority than object search, since people tracking is the basic service for vital applications such as fall prevention. In this situation, the search process can be interrupted. This problem, along with the possible node and network failures, supports a decentralized search design. In the next section, we propose an asynchronous cooperative search strategy in which autonomous search agents share perceptual information but maintain separate probability distribution for targets, and make independent decisions about their search strategy. Multiple asynchronous search agents coordinate activity in a cooperative search strategy by using inter-agent messages to share posterior distributions summarizing where search agents have already looked and where they are likely to look next.

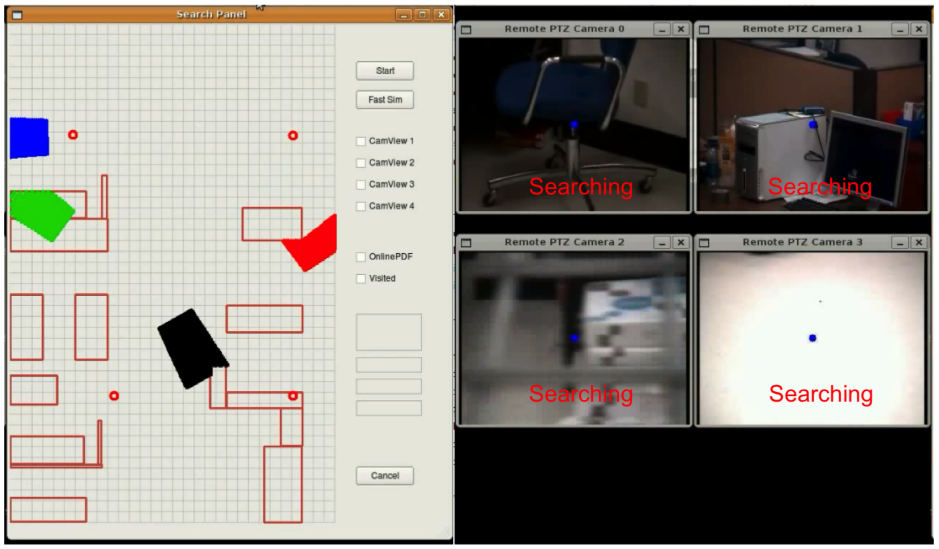


Figure 6.1. PTZ camera agents performing object search.

6.1 Message Sharing-Based Cooperative Search Schema

Rather than maintaining identical PDFs [26], in the proposed search strategy, cooperative agents maintain separate search knowledge concerning the probable observability of the target and make independent decisions about their search process. Two kinds of messages are considered to help coordinate asynchronous search agents as illustrated in Figure 6.3.

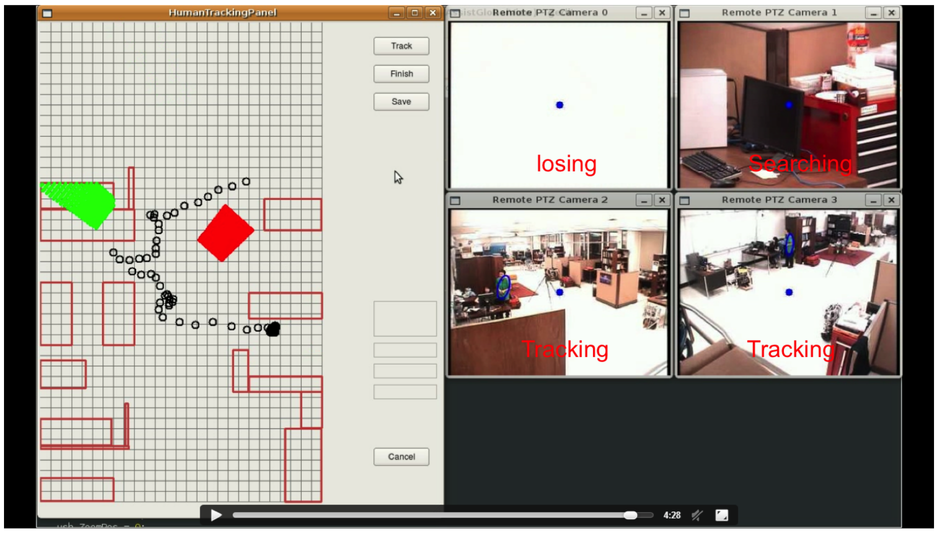


Figure 6.2. PTZ camera agents performing human tracking and object search simultaneously, with connection failure to one searching camera.

Observation Inhibition Message (POIM). Avoiding replicated observations (a form of collision avoidance) is non-trivial. As illustrated in Figure 6.3, a POIM is broadcast by an agent after it selects its next action, but before the control and observation steps are executed. A POIM contains an inhibition map $I(\vec{x})$ that identifies all the grid nodes in the observation field of the next action that has been selected. The agent receiving a POIM combines it with its current local PDF to avoid overlapping observations. As illustrated in Figure 6.4, at time $t + 2$, agent 2 receives the POIM from agent 1. The area that agent 1 is planning to observe is ablated in the local PDF of agent 2, which in turn plans its next action to cover the peak indicated on the right side.

Observation Result Message (ORM). ORM is broadcast after an action is taken. It contains the observation result $p_r(z_t|x_t)$ produced by agent r at time t. The observation result from another agent is considered to be equivalent to observations by the agent itself and, therefore, Equation 5.1 is used to update the search probability distribution. As illustrated in Figure 6.4, the local PDF of target 1 at time $t + 10$

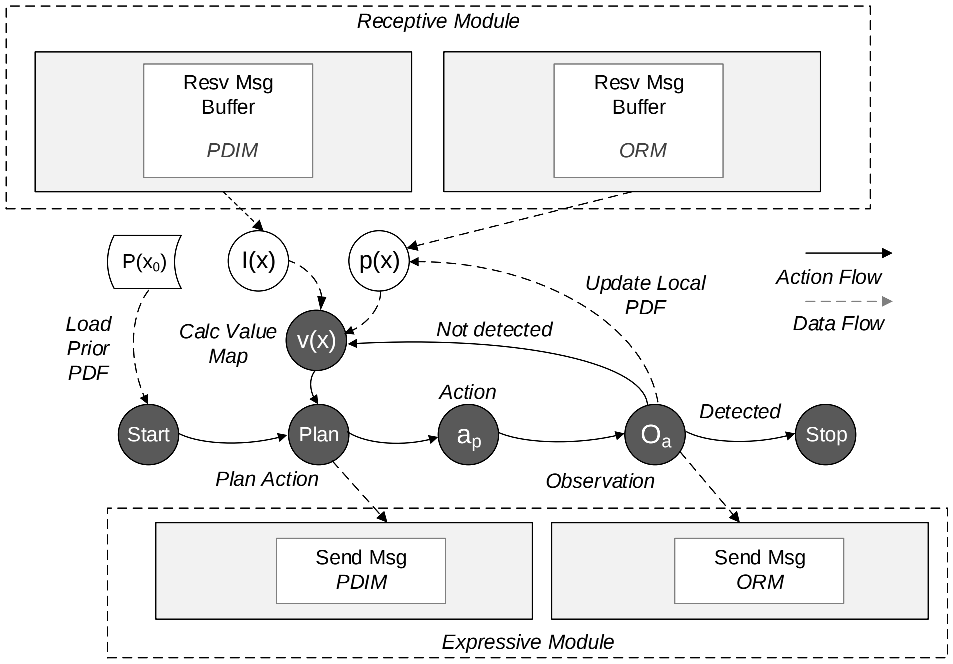


Figure 6.3. The cooperative search strategy for a single agent.

was modified by the previous observation taken by itself, the observation results sent by agent 2, and the POIM message sent by agent 2 at time $t + 5$.

To maintain the correctness of message transmission, message buffers are used to store the POIM and ORM. A consistent network timestamp is used to correct for situations where ORM_t^r arrives later than $POIM_t^r$ due to network latency. In addition, a POIM is discarded if the corresponding ORM hasn't come within time interval T_{POIM}. Therefore, if the agent sending the POIM fails or is interrupted by another task, the area masked by POIM will eventually be observed by other agents.

The approach described in this section is a basic message sharing strategy for multi-agent search. In this strategy, the agents adopt the same utility function as in single agent search, and share messages containing their search intentions and search results. The next section explores how to incorporate more prior knowledge in the utility function to improve the cooperative search efficiency.

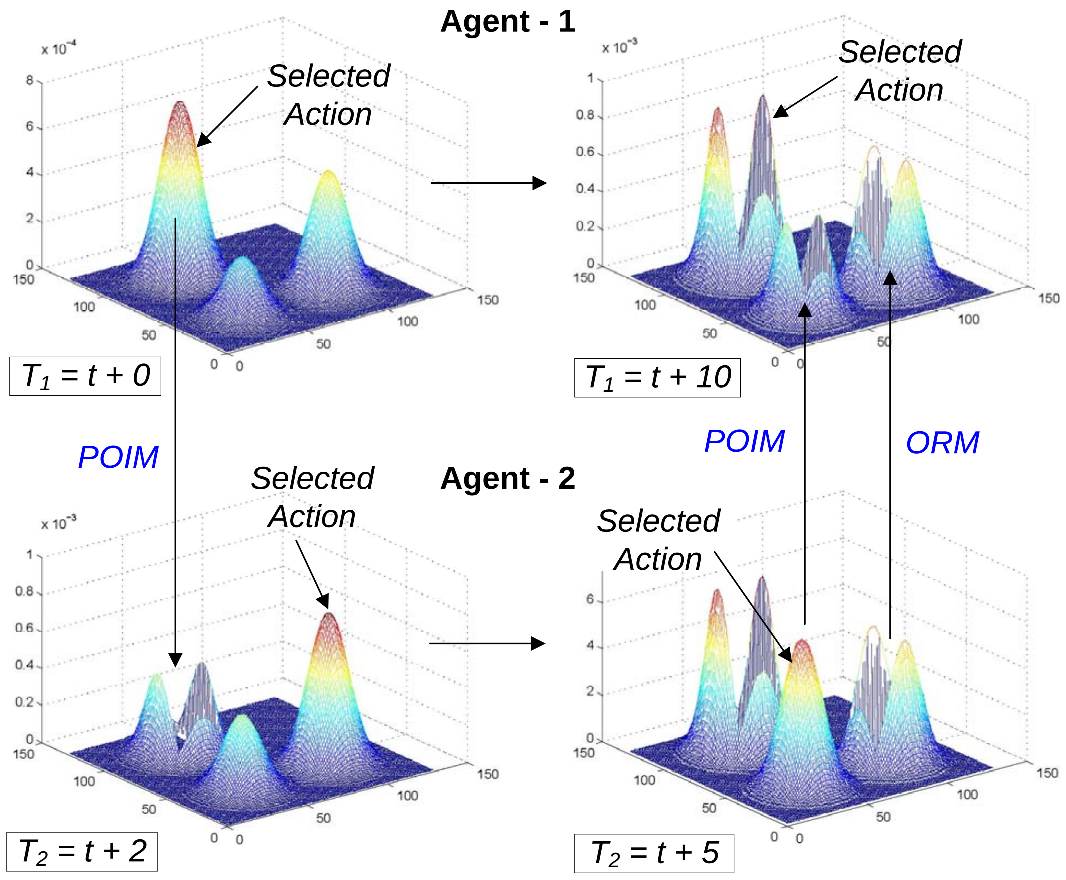


Figure 6.4. Local PDF update for search agents.

6.2 How to Compensate for the Limitations of Teammates

Intelligent agents help teammates to cover the search space efficiently. The basic message sharing-based strategy described above allows the agents to cooperate with their teammates by updating the PDF in the search. So far, prior knowledge only represents the distribution of probable object locations, without considering teammates' search capabilities. Ideally, an agent should know the limitations of its teammates and try to behave to compensate for these limitations. Two types of limitations are considered, (i) viewpoint limitations, which represents the ability of an agent to reach certain locations in the search space, and (ii) limitations on observation reliability, which describe the ability of an agent to make a true-positive detection in certain locations.

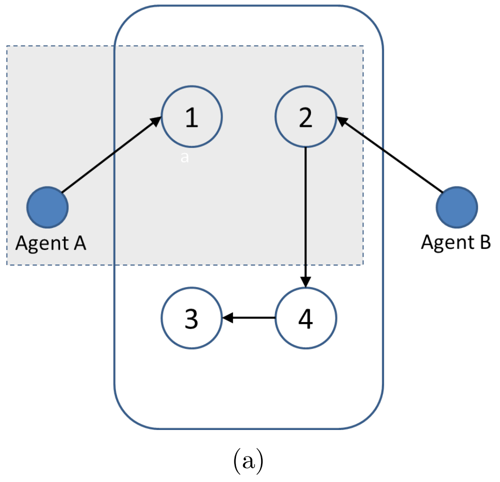
6.2.1 Compensating for Viewpoint Limitations

Robot agents have limited ability to deliver sensors to the viewpoints required by a task. The "reachable" area is defined as the inspection area that is accessible to a particular agent while it is searching. For instance, a mobile robot may not be able to check the areas on the upper levels of a shelf because the robot is not tall enough. This region in the environment may be accessible to the elevated PTZ camera, which is subject to other constraints (i.e., floor areas under tables) due to motor limitations.

To coordinate efficient multi-agent search given these agent-specific constraints, agent should be capable of covering the viewpoint limitations of their teammates. For example, in a search team consisting of a low-profile mobile camera node and a ceiling mounted PTZ camera node, the PTZ camera is well-suited for searching desk surfaces and the upper shelves of bookcases and the mobile robot can provide complementary information on the floor under desks and tables.

Appropriate treatment of asymmetric reachability constraints can improve efficiency as well. For example, in Figure 6.5, two agents start searching from their initial position and enter the search space (the rectangle with rounded corners). The gray rectangle represents the area accessible to agent A. Agent B has the ability to cover the whole space. In Figure 6.5 (a), agent B doesn't know the viewpoint limitations of agent A. To achieve a complete coverage, agent A visits location 1, and the agent B visits location 2, 4, 3 in sequence. The overall number of search steps for the team to achieve complete coverage is 3. If agent B is aware of the viewpoint limitations of agent A (Figure 6.5 (b)) then the total number of search steps can be reduced to 2.

Viewpoint limitations can be coded manually but this is an extremely tedious process and would have to be repeated each time the reach capability of an agent changes. Instead, our approach is to consider mechanisms that allow an agent to acquire knowledge of the viewpoint limitations of its teammates based on observations.



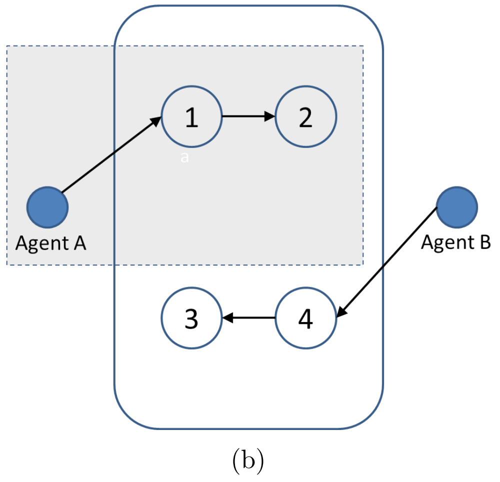


Figure 6.5. An example of viewpoint limitations: (a) without knowledge of reachability constraints, the overall number of steps to achieve complete coverage is 3; (b) if agent B can compensate for viewpoint limitations in agent A, the number of steps can be reduced to 2.

For an agent to compensate for its teammates' viewpoint limitations, utility factor, u_{reach}, is defined in addition to the existing utility factors (u_{info} and u_{cost}) presented in Chapter 5.

In the multi-agent search algorithm, an agent a_k receives Observation Result Message (ORM) from its teammates $a_0, ..., a_{k-1}, a_{k+1}, ..., a_N$. These ORMs are used to accumulate knowledge of viewpoint limitations of its teammates. The travel capability $c_{reach}(\vec{x})$ at a grid location $\vec{x}$ is defined as,

$$c_{reach}(\vec{x}) = N_d \sum_{n=1}^{N} \sum_{i=1}^{D} K_n(\vec{x} - \vec{x}_i^n) \tag{6.1}$$

where $\vec{x}_i^m$ is the inspection location reported by agent a_n in the i^{th} observation. D is the total number of the ORMs, and N denotes the number of teammates. $K_n(\cdot)$ is a suitable kernel function that is determined by the observation model of a_n (here, a Gaussian). Given $c_{reach}(\vec{x})$, $u_{reach}(\vec{x})$ can be defined as,

$$u_{reach}(\vec{x}) = 1 - \alpha \cdot c_{reach}(\vec{x}) \tag{6.2}$$

where α is the factor that balances the trend for an agent to cover its teammates' viewpoint limitations. Now, three criteria are used to calculate the utility function of this strategy:(1)u_{info}, the expected information gain; (2) u_{cost}, the travel cost; and (3)u_{reach}, the viewpoint limitation of the teammates. Then, the utility function at location $\vec{x}$ is defined as weighted combination,

$$u(\vec{x}) = \gamma_1 \cdot u_{info}(\vec{x}) + \gamma_2 \cdot u_{cost}(\vec{x}) + \gamma_3 \cdot u_{limit}(\vec{x}) \tag{6.3}$$

6.2.2 Compensating For Limitations on Observation Reliability

Observations are subject to uncertainty that depends on a wide variety of environmental contexts. For example, a vision-based object detection algorithm may be not functioning well when searching in a dark corner due to the lighting conditions. As a result the agent will probably miss the target in that position. To make an efficient search, the agent with the limitation on observation reliability should avoid untrustworthy observations that cause target detection failure. As a compensation, the other more capable agent should have a higher priority to visit the blind locations of its teammate.

An example is given in Figure 6.6, where the target object is in location 3. Agent A and B are jointly searching in the space. Location 3 is a blind location for agent A. Without considering the limitation on observation reliability, agent A may visit 3 but will miss the target. The total search steps to cover the whole area is 4. However, the joint search would have performed better with complementary behavior if both agents are aware of the observation reliability of agent A in location 3. In this case, agent A would tend to skip location 3 and agent B would tend to visit location B with a higher priority. Complementary joint search take one step less than the previous case.

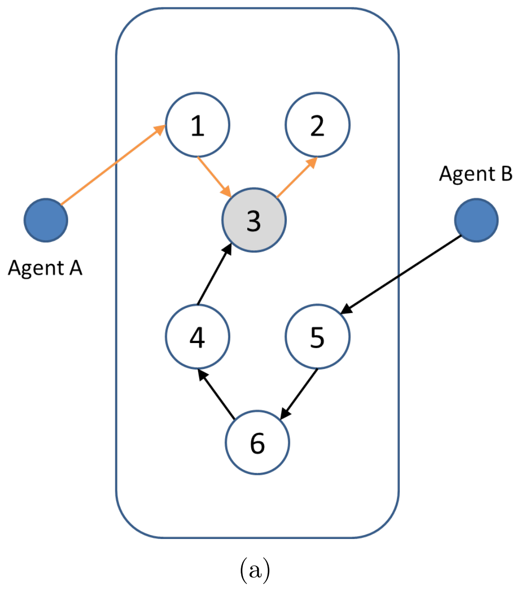


Figure 6.6. An example of observation reliability in joint search. (a) Not considering the limitation on observation reliability, the overall number of steps to achieve complete coverage is 4. (b) The agent B compensates the limitation on observation reliability of the agent A, the overall number of steps to achieve complete coverage is 3.

It's difficult if not impossible to develop an algorithm that works perfectly in all circumstances. To manually label all the places that cause object detection algorithm failure is also tedious for humans. However, autonomous agents that have access to the same environment over a prolonged period of time can model those reachable locations in which detection results are not reliable.

For agent a_i and observation sequence $z_0, z_1, ..., z_N$ in a search trial, and given that finally, the target is detected by agent a_j in location $\vec{x}$. The limitation on observation reliability of agent a_i in location $\vec{x}$ is modeled as an accumulation of the failed observations,

$$l^i_{obs}(\vec{x}) = \begin{cases} N_d \sum_{n=1}^{N} p(\vec{x}|z_n), & j \neq i \\ 0, & j = i \end{cases} \tag{6.4}$$

where $p(\vec{x}|z_n)$ is given by the observation model of agent a_i. N_d is a normalization factor.

Given $l_{obs}(\vec{x})$, and assume there are M agents, the complementary factor c_{obs} for agent k in location $\vec{x}$ is represented as,

$$c^k_{obs}(\vec{x}) = \sum_{m=1}^{M} l^m_{obs}(\vec{x}), m \neq k \tag{6.5}$$

For an agent to successfully avoid the positions with observation failure and to complement its teammates, it needs to consider a utility factor $u_{obs}(\vec{x})$,

$$u_{obs}(\vec{x}) = c_{obs}(\vec{x}) - \alpha \cdot l_{obs}(\vec{x}) \tag{6.6}$$

Three criteria are used to calculate the utility function of this strategy: (1)u_{info}, the expected information gain; (2)u_{cost}, the travel cost; and (3)u_{obs}, the limitation on observation reliability. The utility function at location $\vec{x}$ is defined as weighted combination,

$$u(\vec{x}) = \alpha_1 \cdot u_{info}(\vec{x}) + \alpha_2 \cdot u_{cost}(\vec{x}) - \alpha_3 \cdot u_{obs}(\vec{x}) \tag{6.7}$$

6.3 Evaluations

This section presents the experimental results of the proposed cooperative search strategies.

6.3.1 Performance of Basic Message Sharing Strategy

The effectiveness of the basic message sharing-based strategy in a real world scenario was evaluated using 4 PTZ camera nodes to perform joint search. The viewpoint limitation and observation reliability are not considered in this experiment.

In this experiment, four fixed Sony EVI-D100 PTZ cameras search for a target that was placed randomly in the search space. Each camera is attached to a small

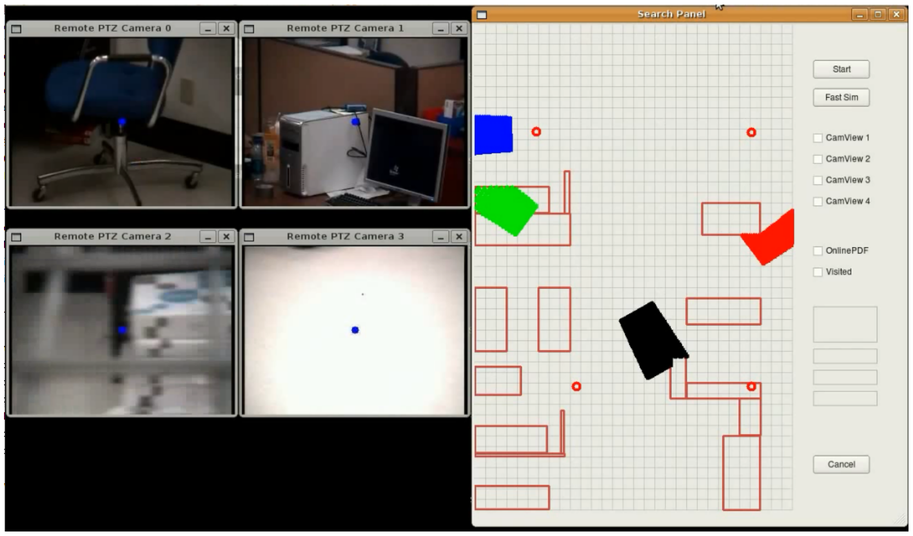


Figure 6.7. The system console when the camera array is searching. The left four windows show the camera views, and the four coloured areas in the map represent the camera's field-of-view that is projected on the floor.

local computer containing an Intel 2.5GHz dual-core processor to form a search agent. The local computer is used to process the captured images and communicate to the central PC and other agents over an 802.11g wireless network.

The deployment of the cameras is shown in the right part of Figure 6.7, in which four circles represent cameras placed 2.4 meters above the floor. The search space setting in this experiment uses the same room as that in *LPR-1*, but two horizontal levels are considered, namely the floor plane (z=0 feet) and table surface planes (z=2.4 feet). We tessellate the horizontal plane into grid nodes with a resolution of 2×2 cm. The target object is a ball with 20cm diameter and solid green color. Object detection is achieved using mean shift algorithm in color space [1]. The prior PDF for each agent is a uniform distribution.

In this test all four cameras are used to search for the object. Figure 6.7 shows a snapshot of the console screen when the camera array was searching. To evaluate the benefit of using the cooperative search strategy, we compare the efficiency of the proposed message sharing-based cooperative search strategy to the method without the cooperation mechanism, in which all agents perform independent search without exchanging message. We measured the “time to detect” cost of the search in 10 independent tests. In each test the target object was placed randomly in the environment. Figure 6.8 shows the time when the first camera detects the target. In 8 out of 10 trials, the proposed cooperative search strategy is more efficient than the approach without cooperation.

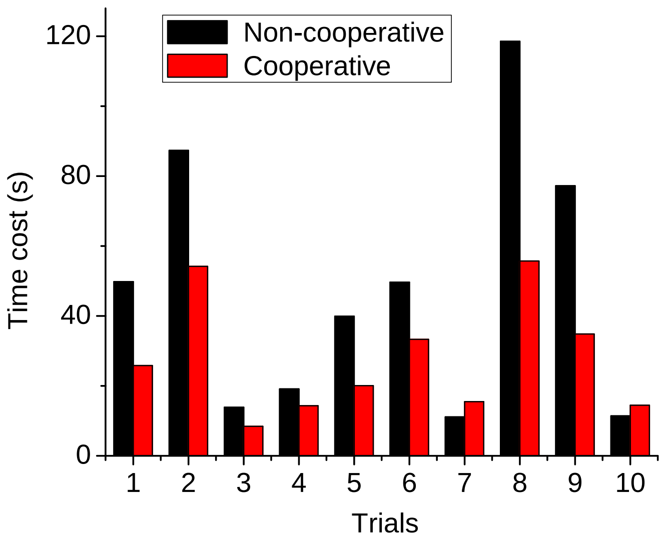


Figure 6.8. Search performance of the camera array for cooperative and non-cooperative coordinate modes.

6.3.2 Benefit of Considering Teammates’ Limitations

A simulated experiment was conducted to evaluate the impact of reachability and reliability predictions on the performance of the team. The simulated environment used in Chapter 5 was used in this experiment as well for a pair of simulated uBot-5 robots. A harmonic function path planner [33] was used for each robot to navigate in the environment $\mathcal{R}^2$, but mutual collisions between robots were not considered.

6.3.2.1 Cooperative Search Strategies with Viewpoint Limitations

In this experiment three search strategies were compared. For each strategy 50 trials were performed.

Stragety 1 (S1): Two robot agents search independently, without cooperation. The utility function incorporates the expected information gain and the travel cost. The utility function is shown in Equation 5.6 (for a reminder, the equation is repeated below as Equation 6.8), where w is set to 0.6.

$$u(p) = w \cdot u_{info} + (1 - w)u_{cost} \tag{6.8}$$

Stragety 2 (S2): Two robot agents search cooperatively using the basic message sharing framework without compensating for the viewpoint limitations.

Stragety 3 (S3): Two robot agents search cooperatively using the basic message sharing framework. They compensate predicted viewpoint limitations in the other agents. Equation 6.3 is used for the utility function with $\gamma_1 = 0.3$, $\gamma_2 = 0.2$ and $\gamma_3 = 0.2$.

Figure 6.9 summarizes the results of the experiment. It shows that Strategy 3 reduces the uncertainty fastest compared to the other two strategies. (Figure 6.10 illustrates the standard deviation for each strategy). It reveals that by considering the viewpoint limitation, each agent is able to compensate for the limitations (here reach) of its teammates and the search efficiency is improved.

6.3.2.2 Cooperative Search Strategies with Predicted Observation Reliability

This experiment also compares three search strategies but focuses on the limitation on observation reliability. For each strategy 50 trials were performed. Strategy 1 (*S1*) and strategy 2 (*S2*) are the same as those in last experiment (Section 6.3.2.1). Strategy 3 (*S3*) is defined as below:

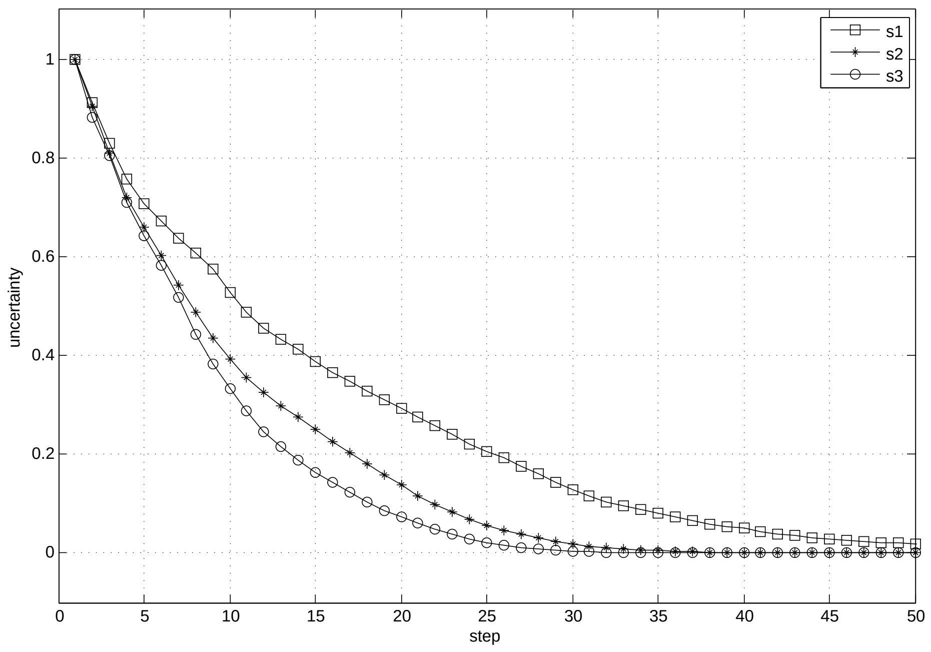


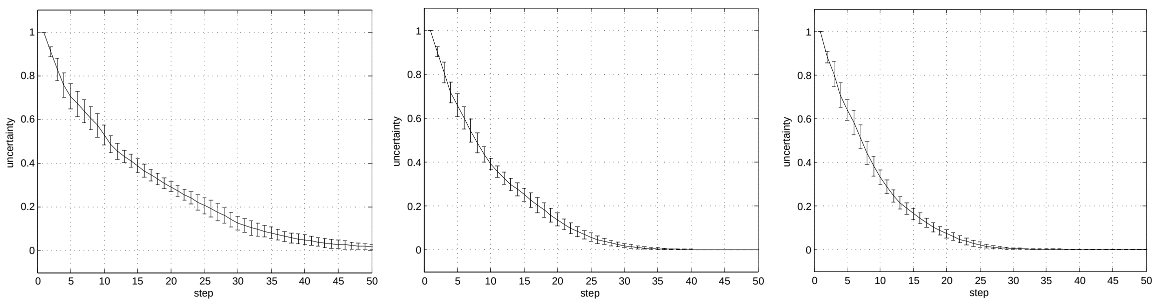
Figure 6.9. Comparison of search strategies with viewpoint limitation. *S1: Non-cooperative search; S2: Cooperative search; S3: Cooperative search, complement teammate's viewpoint limitations.*

Stragety 3 (S3): Two robot agents search cooperatively using the message sharing framework compensating for information gain, search cost, and predicted observation reliability. The utility function is shown in Equation 6.7, where $\alpha_1 = 0.3$, $\alpha_2 = 0.2$ and $\alpha_3 = 0.5$.

Figure 6.11 shows that strategy 3 outperforms the other strategies since it incorporates the limitation on observation reliability (of both itself and its teammates) into the prior knowledge. Figure 6.12 illustrates the standard deviation for each strategies.

6.3.3 Overall Search Performance of the Robot Team

Real world experiments were conducted to evaluate the search performance in a team consisting of 2 PTZ camera nodes and the uBot-5 robot. Since we want to compare the joint search performance with the single robot search and the human search performance presented in Chapter 5, we use the *LPR-1* experimental scenario



(a) (b) (c)

Figure 6.10. Standard deviation (a) S1; (b) S2; (c) S3.

in this test. The target object is the *BOOK*. Please note that (*BOOK*) is much smaller than the one (*BALL*) used in the last section (Section 6.3.1), and the PTZ cameras' zoom value is different. As a result the search speed of the PTZ cameras is slower than that in the last section. The cover of the *BOOK* is a solid color (green) and can be detected by the cameras using mean shift algorithm [1] in color space. Both the ubot-5 robot and the PTZ camera nodes use camera sensor to detect the target object.

In this experiment two strategies were used for testing. (i) S2: Strategy 2 in Section 6.3.2.2; (ii) S3: Strategy 3 in Section 6.3.2.2. For each strategy 5 trials were performed.

The results (shown in Figure 6.13) reveals that cooperative search (both S2 and S3) outperforms the single agent search and achieves relatively comparable performance with the human search. This result is consistent with the simulated results in that S3 is more efficient than S2, though not by very much. It is partly because there were only 5 trials performed and the prior knowledge of the team limitations had not been established well. In the future, more trials are needed to better evaluate the benefit of observation reliability in the real world, which will be discussed in Future Directions (Chapter 8).

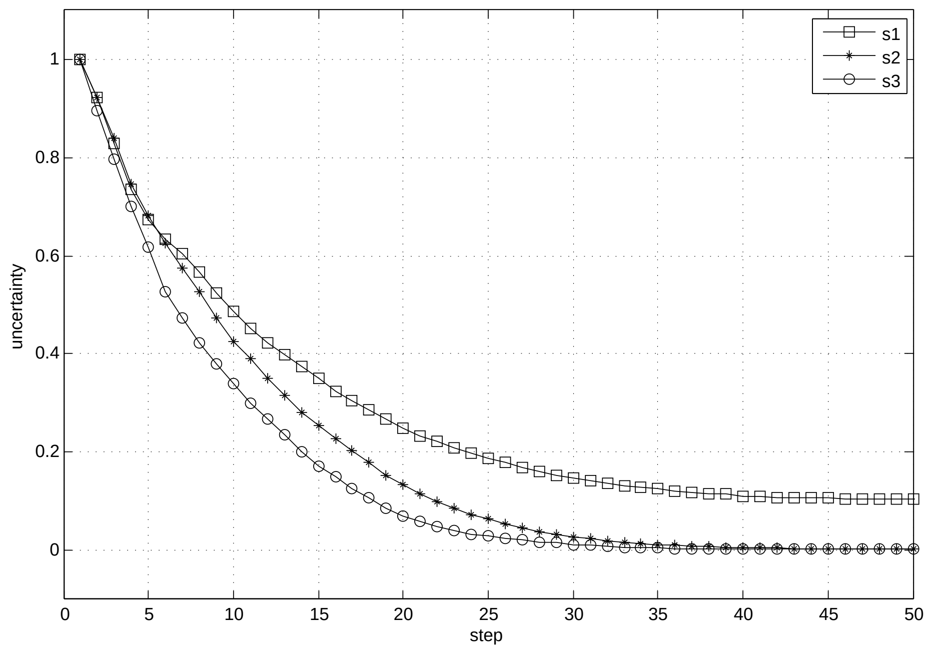
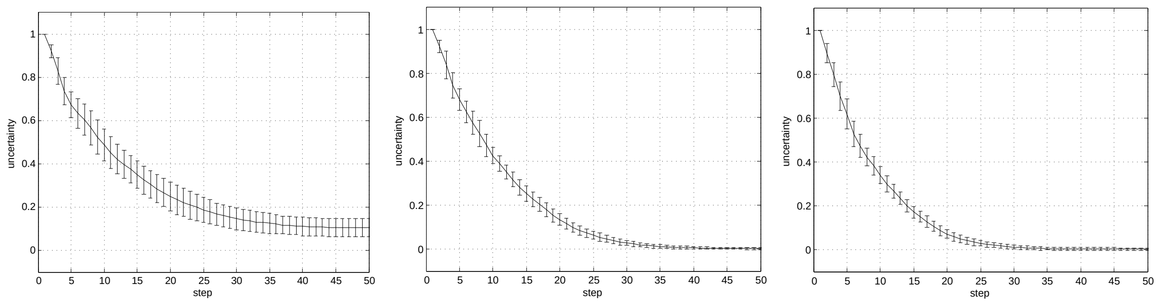


Figure 6.11. Comparison of search strategies with limitation on observation reliability.

6.4 Summary

In this chapter a decentralized and asynchronous cooperative search strategy is developed so that the system is tolerant to failures and interruptions of the search agents. Each autonomous search agent maintains separate estimates of the spatial probability distributions for the target object and makes independent decisions about its search process. Asynchronous cooperative search is achieved by transmitting perceptual information among the agents. Limitations on viewpoint and observation reliability of robot agents are also considered in order to achieve cost-efficient joint search.



(a) (b) (c)

Figure 6.12. Standard deviation (a) S1; (b) S2; (c) S3.

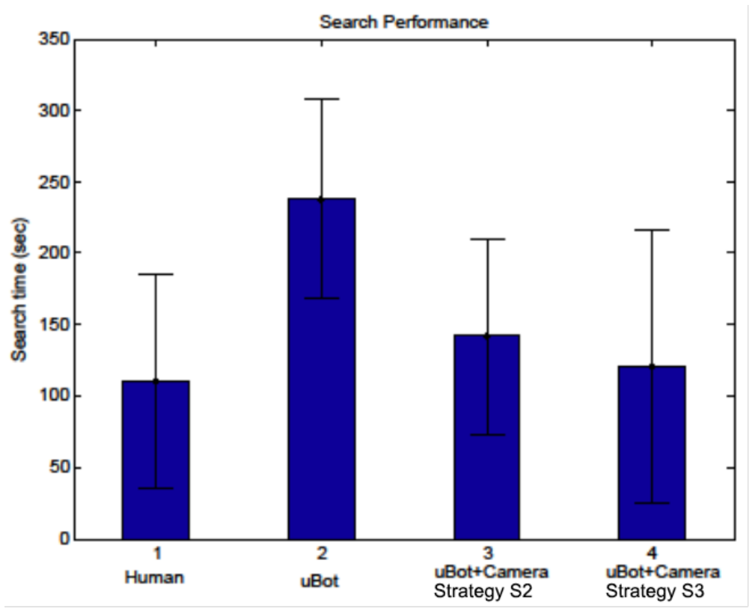


Figure 6.13. Comparison of cooperative and single-agent search strategies (searching for the *BOOK*).

CHAPTER 7
HUMAN-ROBOT COOPERATIVE SEARCH

In the previous chapter a multi-agent cooperative search strategy was introduced. Our vision for search problem solving is that humans and robots will work as partners, leveraging the capabilities of each. Human-robot teams are used in Urban Search and Rescue (USAR) [81, 84], but these applications typically use robots as "drone" under direct control of human teleoperators. There is inadequate study for coordinating human and robots as peers in search tasks, which is the main focus of this dissertation. In our approach, the robot learns to recognize and complement a human's search activity.

In the multi-agent cooperation strategy introduced in the last chapter, the autonomous search agent maintains estimates of the probability density function (PDF) for the object location and makes independent decisions about its search process. Cooperation is achieved by sharing perceptual information and intention among cooperating agents. For robot teammates, explicit message transmission is used. A human teammate is modeled as an agent without this mode of communication. In this case, the robot agents infer the current state and intention of the human peer using a model of human search activity acquired in the learning session. By inferring human search states, the robot chooses compensatory actions to achieve efficient cooperation with human peers.

Recently intention estimation for robot-human interaction has been investigated as a means for the robot to assist humans. Along this line, there is recent work aimed at recognizing human activity as a means of inferring intention. In [105], a vision based approach is used to infer the intentions of other agents. In [85], a hierarchical hidden Markov model is used to recognize a set of complex indoor activities. However, the approach of integrating intention estimation with robot planning is still in need of investigation, especially in search tasks.

A second area of the work presented in this chapter is the problem of user interface design for robot assistance. Cognitive load is an important factor for human-robot interaction and has been studied considerably in the work on interface design [56, 52, 57]. Most of this work is based on explicit message transmission without considering the potential for using predictions of human intentions. To reduce the mental stress in H-R interaction, collaborative control [29, 42] was developed for mobile autonomous robots. The robots work autonomously until they run into a problem they can't solve. At this point, the robots ask the remote operator for assistance, allowing human-robot interaction and autonomy to vary as needed. In this dissertation, we discuss how collaborative control mechanisms can be used for service robots in elder care applications. An implicit interface design for robot assisted tasks is proposed, which allows the robot to infer the intention of the user and to provide assistance autonomously. It reduces the cognitive workload of the user and therefore is useful for elder care applications.

The robot assisted search scenario is illustrated in Figure 7.1 where some of the information that the robot needs to know for efficient assistance is listed. The first two questions are related to user interface design, which is used to inform the robot what assistance is needed by human. The user interface can be interactive, which means that the robot could learn the answer to question 1 in Figure 7.1 by observing the human's action and then asking what the target object is. The third question is related to search coordination, where the robot needs to plan the next move to cooperate with its human peer in order to achieve the goal. In this chapter, our approach

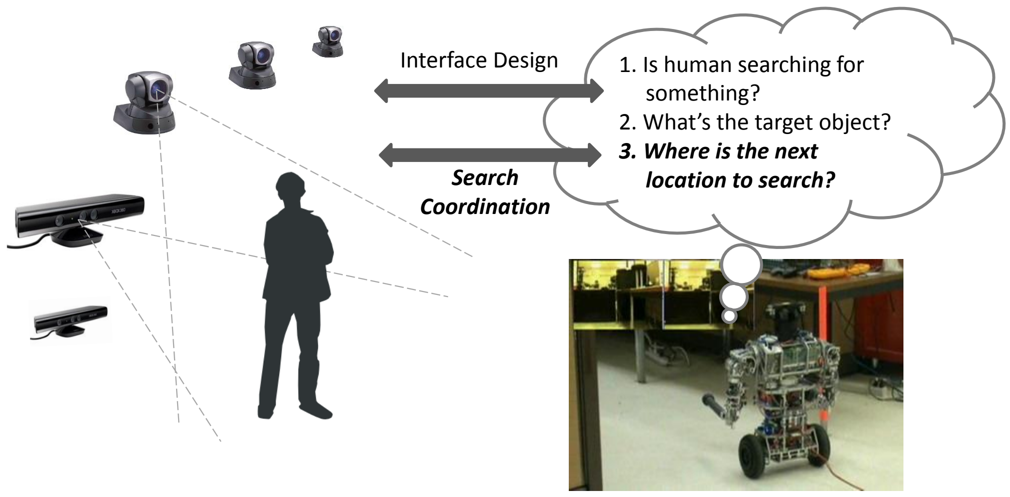

Interface Design
1. Is human searching for something?
2. What's the target object?
3. Where is the next location to search?
Search Coordination

Figure 7.1. A scenario of robot assisted search in a home environment

to human-robot cooperative search is presented first, followed by the problem of user interface design.

7.1 Search Schema with Human Teammates

In previous chapters, Bayesian framework for search tasks is introduced in which messages are used to convey observations and intentions between agents to coordinate otherwise autonomous search activities. These messages are used to update probability distributions in each agent that influence its choice of search actions. To incorporate human agents into this framework, we have adopted a strong commitment not to impose cognitive overhead on the human (eldercare) client that might diminish his or her attention to the primary search task. Therefore, while the human client will be informed if the team locates the target object, we do not require that human searchers announce their state or their intentions to the rest of the team. Instead, in addition to searching, robot agents must infer the current state and intention of the human peer using a prior model of human search activity. By inferring human search states, the robot chooses compensatory actions to achieve efficient cooperation with human peers.

A schematic for the proposed distributed architechture with a human-in-the-loop is shown in Figure 7.2, which is an elaboration of Figure 5.1 and 6.2.

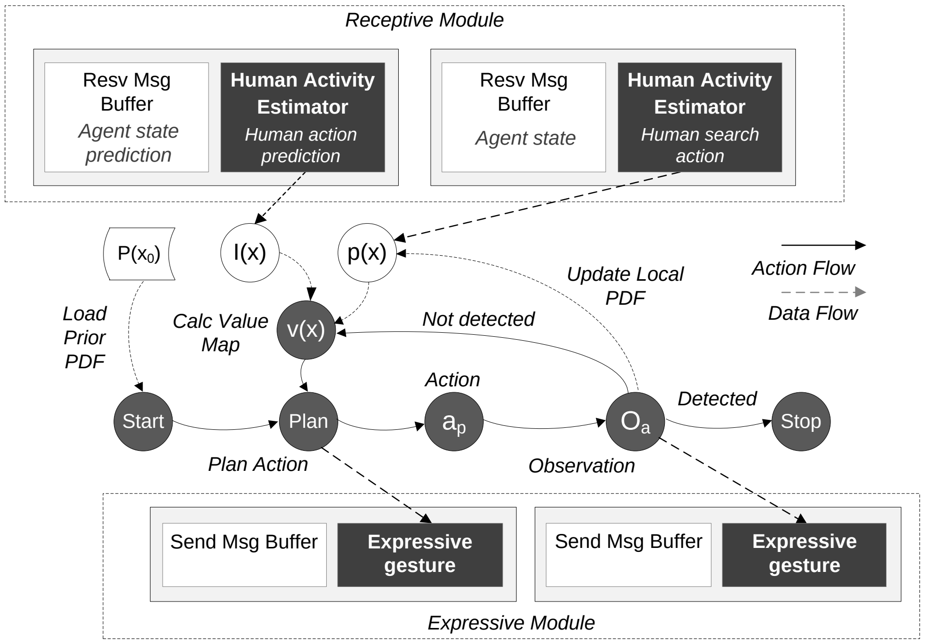


Figure 7.2. Cooperative search strategy for a single agent when working with human teammates.

7.2 Human Peer Modeling

We model human peers as agents without explicit message transmission. As shown in Figure 7.2, the *Human Activity Estimator* and *Expressive Gesture* are used to replace message sharing modules when communicating with human peers. In this chapter, the focus is on the receptive module that enables the robot to estimate human activity and to provide complementary behavior. At time t, the human activity estimator provides two types of information: (i) a human search location sequence $X_t = (x_0, x_1, ..., x_t)$, which describes the human search history till t, and (ii) a prediction x_{t+1}, which is the predicted search location of the human at time $t + 1$.

7.2.1 Dwell State Detection

A reasonable human observation model is necessary for the robot to estimate the impact of human search actions. In Section 5.3.2, the experimental data reveals that a human searcher will dwell in places where objects are likely to be found, which suggests using human observation models with different kernel sizes for dwelling and walking states. A K-means algorithm [48] is used to cluster the state vectors for all human tracking points $o(t) = [x, y, \dot{x}, \dot{y}]$ in order to distinguish dwell and walking states and to find out all possible dwell states. During training, human subjects were asked to search for various objects and tracking points and velocities were logged and subsequently clustered in batches to reveal dwell states and transitions (walking states) that connect dwell states for each target object. Figure 7.3 illustrates the clustering results.

The appropriate number of clusters, k, depends on the domain. In this work the value of k is manually selected. Automatic selection of k can be achieved by applying certain cluster analysis approaches, such as hierarchical clustering [49], but is beyond the scope of this dissertation.

We assume that human search activities are only related to the dwelling states (It is reasonable since 100% successful target found events happened in the dwelling states in the experiments presented in this dissertation). Dwell and walking states can be distinguished by using a simple classifier based on the tracking velocity, for cluster C_i, if $\sqrt{\dot{x}^2 + \dot{y}^2} > T$, then all tracking points $o_k^i \in C_i$ are labeled "walking" states, else they are labeled as certain "dwell" states. Walking clusters were discarded since they don't provide informative knowledge on the tracking behaviors.

By applying this clustering algorithm to the original human tracking data, human dwell location sequences $X_t = (x_0, ..., x_i, ..., x_t)$ can be obtained, in which x_i belongs to a dwell state from the clustering result. Then stochastic modeling tools can be selected and applied to the dwell location sequences X_t to modeled the human search activities.

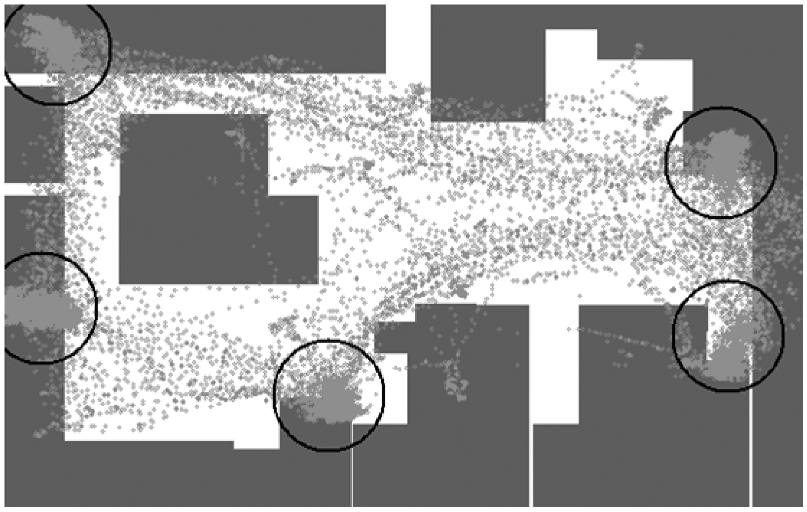


Figure 7.3. Clustering result and human dwell states for BOOK search.

7.2.2 Activity Modeling with HMM

Given the observed dwell location sequences, a stochastic model can be constructed for human searching activities. Hidden Markov Models (HMMs) are powerful tool for modeling sequential phenomena, and have been successfully used in applications involving speech signal recognition, DNA sequence analysis, handwritten characters recognition, natural language domains, etc. Recently, HMMs have been used for activity understanding, showing a significant potential for their use in activity modeling and inferring intent [93].

7.2.2.1 Hidden Markov Model (HMM)

As stated in Rabiner and Juang [69], the classical tutorial to HMMs, an HMM is defined as a doubly stochastic process with an underlying hidden stochastic process and an observed stochastic process which can produce the sequence of observed symbols. The underlying hidden stochastic process is a first-order Markov process;

that is, each hidden state depends only on the previous hidden state. Moreover, in the observed stochastic process, each observed measurement depends only on the current hidden state. An HMM inference graph is shown in Figure 7.4. The circular nodes denote the hidden state variables, and the square nodes represent the observed variables.

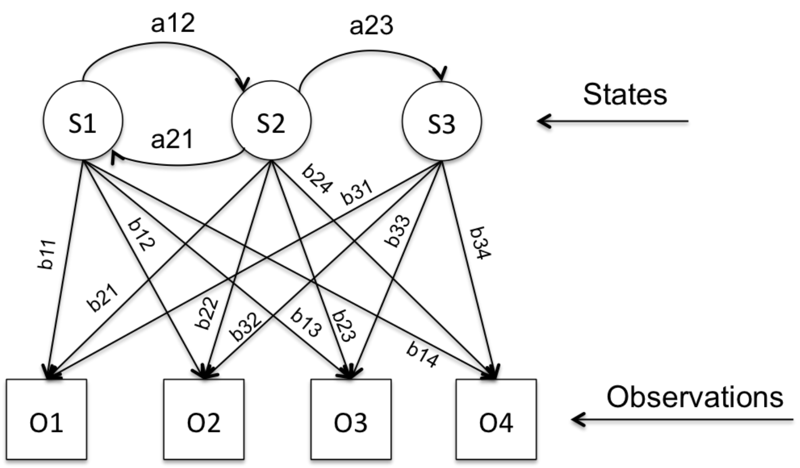


Figure 7.4. A graphical representation of Hidden Markov Model with three states.

Formally, an HMM is defined by the following entities:

- A set $S = S_1, S_2, ..., S_N$ of hidden states; where N is the number of states.
- A set $O = O_1, O_2, ..., O_T$ of individual observations; where T is the number of observations.
- A transition matrix $A = a_{ij}$, where $a_{ij}0$ represents the probability of going from state S_i to state S_j ;
- An emission matrix $B = b(O|S_i)$, where b represents the probability of emission of symbol O from state S_i;
- An initial state probability distribution $\pi = \pi_i$, representing the probability of the first state $\pi_1 = P[Q_1 = S_i]$

Generally, the HMM can be represented as: $\lambda = P(A, B, \pi)$. In real applications, there are three basic problems to be solved in the HMM model, which are:

- **Evaluation.** Given the model $\lambda = P(A, B, \pi)$ and the observation sequence $O = O_1, O_2, ..., O_T$, how do we find $P(O|\lambda)$?
- **Decoding.** Given the model $\lambda = P(A, B, \pi)$ and the observation sequence $O = O_1, O_2, ..., O_T$, how do we find a corresponding state sequence $S = S_1, S_2, ..., S_T$?
- **Learning.** Given the observation sequence $O = O_1, O_2, ..., O_T$, how do we find the model parameters $\lambda = P(A, B, \pi)$. which maximize $P(O|\lambda)$?

These problems can be solved efficiently using the Forward-Backward Algorithms, the Viterbi Algorithm and the Baum-Welch Algorithm. For more information about the algorithms typically used in hidden Markov modeling problems, please refer to the tutorial by Rabiner [69].

7.2.2.2 Activity Modeling and Training

In our work HMMs were used to model the search activity of human beings where the hidden states denote search locations and the observed outputs are the dwell locations from human tracking and clustering described in Section 7.2.1. One HMM is generated for each activity to be learned, e.g., searching for $BOOK$ and searching for $CLAMP$.

HMM parameters $\lambda = (A, B, \pi)$ are trained by observing the sequence of dwell locations traversed over a number of search demonstrations. The Baum-Welch algorithm [23] is used to train model parameter λ_i corresponding to the search activity class a_i. Baum-Welch algorithm is a generalized expectation-maximization algorithm defined by Equation 7.1 that modifies transition weights and the statistics of the models.

$$P(O|\lambda) = \sum_{S} P(O, S|\lambda) \tag{7.1}$$

For more detailed information about training HMM see [23] or [69].

7.2.2.3 Decoding

Given a trained HMM with parameters $\lambda = (A, B, \pi)$ and an observation sequence, Viterbi algorithm [69] is used to find an optimal sequence of states, which in our case indicates the search locations the human subject has visited. Given a dwell location sequence with length t' as observation $O = O_1, O_2, ..., O_{t'}$, the algorithm proceeds as follows:

Initialization:

$$\delta_1(i) = \pi_i b_j(O_1), 1 \leq i \leq N \tag{7.2}$$

Induction: For $2 \leq t \leq t', 1 \leq j \leq N$

$$\delta_{t+1}(j) = \left[\max_{1\leq i\leq N} \delta_t(i)a_{ij}\right] b_j(O_{t+1}) \tag{7.3}$$

$$\psi_{t+1}(j) = \left[\arg\max_{1\leq i\leq N} \delta_t(i)a_{ij}\right] \tag{7.4}$$

Termination:

$$p^* = \max_{1\leq i\leq N} \delta_{t'}(i) \tag{7.5}$$

$$q^*_{t'} = \arg\max_{1\leq i\leq N} \delta_{t'}(i) \tag{7.6}$$

The state path can be read backward as:

$$q^*_t = \psi_{t+1}(q^*_{t+1}), t = T-1, ..., 1 \tag{7.7}$$

The algorithm give us the optimal state sequence stored in vector q^*. p^* is the probability of q^* generating O. At time t', the calculated state sequence $q = q_1, q_2, ..., q_{t'}$ is used in Section 7.2.3 for the robot agents to update target's probability distribution based on the estimated human activities.

7.2.2.4 Prediction

In addition to estimating the optimal human search history, the system also needs to predict the next search location of the human teammate, so that the robot agents can act complementary to human search. The prediction at time $t+1$ is based on the probability estimations on all states $S_i (1 \leq i \leq N)$ at time t, which can be represented as the forward probability in the forward algorithm [69] of HMM. Given a trained HMM with parameters $\lambda = (A, B, \pi)$, the forward probability is defined as

$$\alpha_i = P(O_1, O_2, ..., O_t, q_t = S_i | \lambda) \tag{7.8}$$

This is the probability of a partial observation sequence with length t and state S_i at time t, given the model λ. $\alpha_t(i)$ can be calculated inductively by the following equations:

$$\alpha_1(i) = \pi_i, b_i(O_1), 1 \leq i \leq N \tag{7.9}$$

For $t = 1, ..., T-1$, calculate $\alpha_{t+1}(i)$:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(O_{t+1}), 1 \leq j \leq N \tag{7.10}$$

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{7.11}$$

If we calculate the forward probability $\alpha_t(i)$ for a partial observation sequence of length t, we get for each state S_t the probability of being in that state. To get a prediction about the next state q_{t+1}, the joint probabilities of forward probability $\alpha_t(i)$ and transition probability a_{ij} have to be summed up. So for each state q a next step probability $\sigma_{t+1}(q)$ can be calculated as:

$$\sigma_{t+1}(q) = \sum \alpha_t(i) a_{ij}, \forall i, j \tag{7.12}$$

The most probable state q^*_{t+1} at the next step is:

$$q^*_{t+1} = argmax[\sigma_{t+1}(q)], \forall q \tag{7.13}$$

At time t, the next human search location is predicted as the state q^*_{t+1}, which is used by the robot agents to choose search action to complement human search. The robot's search strategy is described in the following section.

7.2.3 Robot Search Strategy

Given the estimated state sequence $Q_t = (q_0, q_1, ...q_t)$ which represents the locations where human subject paused in the search and the predicted state q^*_{t+1} which indicates the location where a human subject is likely to make the next observation, the cooperative search can be performed by robot agents to complement human's search. We assume that the human observation model $p_h(z_t|x_t)$ is represented by a Gaussian distribution for simplicity. Using Q_t, q^*_{t+1} and p_h, the robot updates the probability distribution describing the likely location of the target object using Bayes' theorem (Equation 5.1). The robot selects actions in the following way:

(1) At time t, the robot infers the hidden search sequence Q_t and predicts the next human search location q_{t+1}.

(2) The robot selects the action a_t to complement the search behavior of the human. Applying Bayes' theorem, a temporary PDF $P_t^{'}$ is constructed by updating robot's PDF P_t with the estimated human location sequence Q_t along with the human observation model p_h. q_{t+1} is used to trigger an Inhibition Message $I_t(x)$ [114] to update $P_t^{'}$ as well to bias the robot to avoid searching locations where the human subject is likely to go in the next step.

(3) The robot uses $P_t^{'}$ to plan the next action. A value v_i is calculated for each grid node indicating the benefit of visiting this node. Given $P_t^{'}$ for an agent, v_i is calculated by,

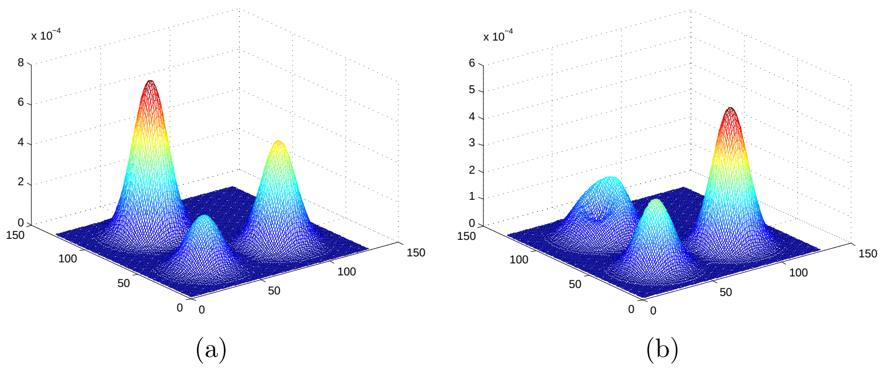


Figure 7.5. Posterior probability distribution describing the location of the target object update for a search agent: (a) before observation. (b) after current observation.

$$v_i = \sum_{g_k \in V_t^i} p_t^{'}(x_k) \tag{7.14}$$

where V_i is the set of all grid nodes in the observation field, i.e., that can be observed when the agent is visiting grid node g_i.

(4) The robot moves to the planned location and performs an observation. The PDF P_t of robot is updated in Bayes' theorem using its observation result and Bayes Theorem. Figure 7.5 illustrates the update of the PDF map using Bayes' theorem. The steps (1)-(4) iterate until the target object is found by the robot or human.

7.3 Interface Design for Implicit Cooperation

To cooperate with human peers autonomously, the robot also needs to know when to search and what is the target object (first two questions in Figure 7.1). This information is usually conveyed through a GUI or voice control and may increase the cognitive load of the user. This problem becomes more severe in elder care.

In collaborative control frameworks, robots work autonomously until they run into a problem they can't solve. At this point, the robots ask a remote operator for assistance. A similar mechanism is used in our work, where the robot estimates the human's intention and autonomously provides assistance without explicit command instructions to do so. We call this approach an *Implicit Interface* for a service robot. The explicit and implicit user interfaces are categorized as follows:

(i) an explicit interface conveys explicit tasks to the robot to implement a coordination plan conceived by the operator.

(ii) an implicit interface infers robot tasks from the observations of human activity. The robot asks questions to verify the inference results when the recognition confidence is low. The autonomy of the robot is adjusted dynamically according to the recognition confidence and the cognitive load of the user.

Using the Hidden Markov Model described in the last section, it is possible to evaluate the partial observation sequence to identify what target object the human subject is searching for (e.g., searching for BOOK rather than CLAMP). Our experiments show that search activities for different objects are distinguishable since human search patterns are different for different objects.

Given a trained HMM $\lambda = \{A, B, \pi\}$, the probability of an observation $O_t = O_1, O_2, ..., O_t$ can be calculated using the forward algorithm (iteratively using Equation 7.9, Equation 7.10 and Equation 7.11). Given that at time t, $P(O_t|\lambda_i)$ is calculated for each model λ_i, the model with maximum likelihood is chosen as the recognized activity class. The basic formulation of the problem is given by the maximization of a conditional probability as in Equation 7.15. The classes are considered to be balanced in our experiments.

$$i^* = \arg\max_i P(\lambda_i|O_t) = \arg\max_i \left[\frac{P(O_t|\lambda_i)P(\lambda_i)}{P(O_t)}\right] \tag{7.15}$$

The processes that support the implicit interface are elaborated as follows.

(a) The robot waits and observes human activities. After an amount of data accumulated, the classification algorithm is used to classify the observed human activities.

(b) The robot claims an activity a_i is detected (e.g., human is searching for a book) if the recognition confidence is higher than a threshold. In this case the robot can choose to initiate a dialog with the human for verification. The cognitive load M_t of user at time t is to be limited to satisfy a threshold T_{ml} throughout the interaction. The robot initiates a dialog if $M_t < T_{ml}$. Otherwise, the dialog is not allowed and the robot continues to assist autonomously until better recognition results are obtained. Cognitive load can be described in several different ways. For simplicity, M_t is described in terms of verbal communication density, which is the weighted sum of questions over a period of time.

(c.1) Given positive user feedback on activity a_i, the robot starts assisting the human search with the approach described in the last section.

(c.2) Without initiating interface dialog or receiving any user feedback in step (b), the robot continues to assist human autonomously with the currently recognized search activity a_i.

In step (b), more information can be obtained from the dialog with the human, such as which objects are related to the activity a_i. When generalizing the implicit interface from search activity to other daily activities, it is important to infer the objects that participate in a_i. For search activity, we consider the case in which only one object is involved. For other activities like reading, multiple objects may be involved. Imagine applying the implicit interface to a reading activity. Two objects (a book and a light) can be associated to this activity in the form of prior knowledge. Due to the constraints on cognitive load, the robot may only be allowed to assist autonomously, in which case, it will bring the book to the user and turn on the light even it is not verified through the dialog.

7.4 Guided Search with Human Gesture

I also evaluated the potential for human directional gestures to allow human clients some direct control over the coordination policies. By pointing (see Figure 7.6), the human expresses the intention that the robot should search in the area pointed to.

In our system human gesture recognition is performed using the Microsoft Kinect sensor and the OpenNI skeleton tracking library [89]. When the robot starts a search, it drives to the center of the room and the human teammate will give it a goal area to search by pointing to it. The robot then determines the direction of the pointing gesture and drives to the desired search area. After the robot searches that area, it drives back to the center of the room to receive another pointing command from the human.

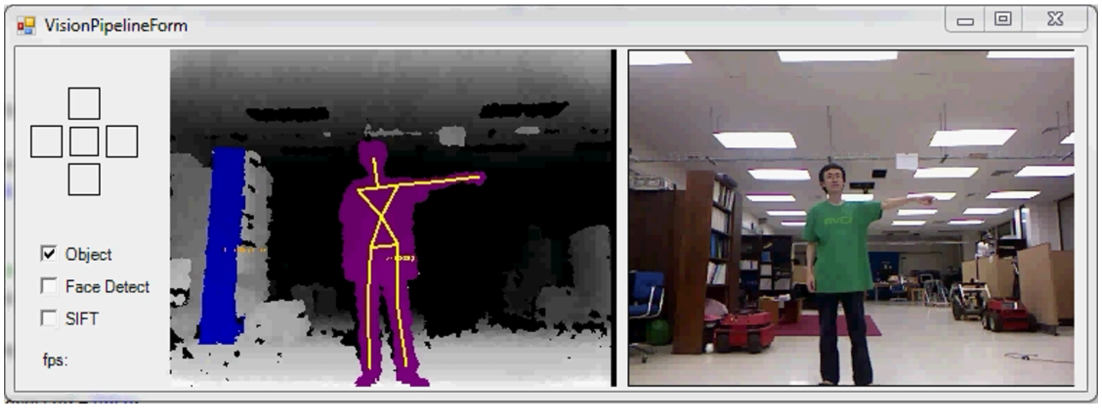


Figure 7.6. Skeleton tracking in the camera view of the robot.

The OpenNI [89] Skeleton tracking system is built in a ROS service which gives a direction vector that can be mapped into the world reference frame based on the robot's current position and heading estimate (as shown in Figure 7.6). To account for inaccuracies in human directions we adopt a probabilistic interpretation of the gesture direction [95]. The presence of a directional vector indicates a distribution $p(\vec{x}|\vec{s}_t, \vec{d}_t)$ over the target object as shown in Equation 7.16,

$$p(\vec{x}|\vec{s}_t, \vec{d}_t) = \epsilon_\sigma(dist) \tag{7.16}$$

where $\vec{s}_t$ indicates the position of the gesture giver, and $\vec{d}_t$, the indicated direction. $\epsilon_\sigma(dist)$ denotes a Gaussian centered on the direction vector.

Given $p(\vec{x}|\vec{s}_t, \vec{d}_t)$, a utility factor u_{gest} that represents the benefit of visiting a grid node in the map is obtained,

$$u_{gest}(i) = \sum_{g_k \in \mathcal{V}_i} p_{c,r}(x_k) \tag{7.17}$$

where $\mathcal{V}_i$ is the set of all grid nodes in the *observation field* (previously defined in Section 5.2.1), i.e., that can be observed when agent is visiting grid node g_i .

When searching, u_{gest} is used as a factor in the weighted combination of utility function $u(\vec{p})$,

$$u(p) = \beta_1 \cdot u_{info} + \beta_2 \cdot u_{cost} + \beta_3 \cdot u_{gest} \tag{7.18}$$

While human communication in the presented work is limited to gesture interpretation, the work extends to more sophisticated modes of interaction, such as language-based direction giving [38].

7.5 Learning from Demonstration

It is possible to refine the prior PDF over objects using the *Search Activity Density (SAD)* of human. SAD reflects the spatial probability of an object by observing the behavior of human demonstrations of search activity directed at the object.

The influence of a dwell point obtained from the teacher on the prior PDF is determined by the Cartesian observation error ellipsoid, which can be estimated by the triangulation Jacobian J for a camera pair. If D is the baseline between two cameras and γ_R and γ_L are the respective headings to the target, the uncertainty Jacobian is given as follows,

$$J = \frac{D}{\sin^2(\gamma_R - \gamma_L)} \begin{bmatrix} \sin\gamma_R \cos\gamma_R & -\sin\gamma_L \cos\gamma_L \\ \sin^2(\gamma_R) & -\sin^2(\gamma_L) \end{bmatrix}$$

The eigenvalues and eigenvectors of JJ^T define the principle directions of error amplification in stereo triangulation. The probability of observing target r at an occupancy grid $\vec{x}$ is given by:

$$p_h(\vec{x}|r) = N_h \sum_{k=1}^{D_i} K_h(\vec{x} - \vec{x}_k) \tag{7.19}$$

where $\vec{x}_k$ represents the locations where human dwell states are observed, D_i is the total number of dwell points, and $K_h(\cdot)$ is a suitable kernel function (here, a Gaussian), which is scaled and rotated using the eigenvalues and eigenvectors of JJ^T.

7.6 Evaluation

This section presents the experimental results of the proposed human-robot cooperative search strategies.

7.6.1 Simulated Experiments

An experiment that combines real world data and simulated data is presented. Human subjects search for objects in a mock apartment environment and their trajectories are recorded. The mock apartment is 42×28 square foot as shown in Figure 7.7 (a). The search space setting in this experiment is different from the setting of *LPR-1.* In this experiment we use the same room as that in *LPR-1*, but the configuration is different. The target object for search is a *BOOK* and a *SCREWDRIVER.* The PTZ camera array is used for human tracking in this experiment. The cameras perform tracking on captured frames using color and edge features [115]. Each time step, a pair of color cameras is selected to determine the 3D location of the human subject in the environment [61].

Since we focus on investigating the receptive behavior and there are no expressive gestures performed by the robot, it is reasonable to assume that human behavior is not influenced by robot actions. The real world data of the human peer is recorded

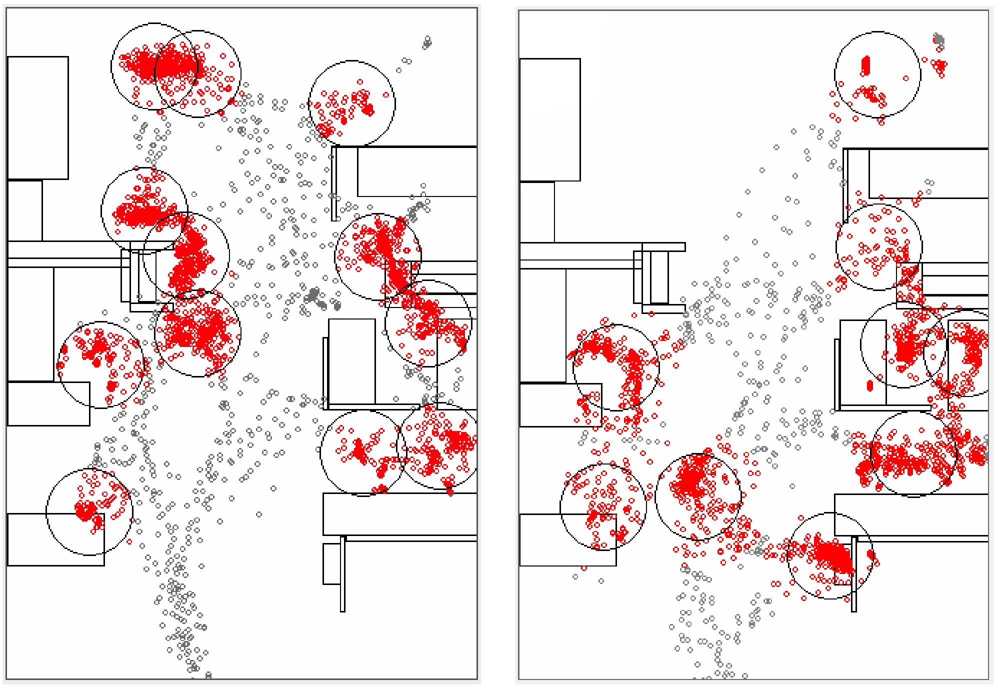

(a) (b)

Figure 7.7. Human data (a) searching for a book; (b) searching for a screwdriver.

and replayed in simulations, in which the simulated uBot-5 cooperates in search tasks to complement the human teammate. The simulated uBot-5 and environment were developed using the Microsoft Robotics Development Studio. The target object is simulated as a colored 3D sphere that is detected visually by the simulated uBot-5 using Camshift [1] algorithm.

7.6.1.1 Efficiency of Human-Robot Cooperative Search

We first evaluate the efficiency of the proposed cooperative search strategy. Eight subjects were recruited for this study. Among these participants, 3 were colleagues of the authors and familiar with the lab environment. The remaining 5 were unfamiliar with the search room and received a short description of the room configuration and furniture. Two search activities were considered, *searching for a book* and *searching for a screwdriver*. In each trial the target object was placed randomly in possible locations (e.g., a book on the shelf, table or on the couch) in the environment. The participants were asked to search for the designated target objects in each trial. As shown in Figure 5.3 and Figure 7.7, subjects presented different search patterns for different objects. For instance, the participants went to bookcases and tables to search the book, and toolboxes when searching for the screwdriver. Each participant performed 5 search trials for each object for a total of 40 trials. 24 trials were used to train the activity model and the remaining 16 trials were used to evaluate the result.

We compare the efficiency of four search strategies: (1) Single human search; (2) Single robot search; (3) Human-Robot (H-R) team search without cooperation, where human and robot search simultaneously but independently without communication; and (4) H-R team search with cooperation. For the 16 datasets used for evaluation (for each object), the "time to detect" cost of different search strategies were measured. The experiments in this part use the explicit interface, where subjects send commands to robot directly, and the team starts searching together.

Table 7.1 gives the average time cost of the search strategies. It can be seen that the search efficiency with the single robot is comparable to that of the human (the average time cost to find an object on all trials is 211.9s for single human and 219.8s for single robot), which indicates that the robot is a qualified search teammate for efficient cooperation. The time cost of cooperative search is 29.6% less on average than that of non-cooperative search for object *BOOK*, and 13.4% for object *SCREWDRIVER*, which indicates that the proposed cooperative search is efficient.

Figure 7.8 shows the comparison of search strategies in all trials for searching the book. In all trials the cooperative search is better than single human search. In 14 out of 16 trials the proposed cooperative search strategy is more efficient than the single robot search without human peer. In 14 out of 16 trials the performance of the proposed cooperative search is better than or equal to that of the non-cooperative strategy. There are some cases where cooperative search is worse than non-cooperative strategy. The reason is that sometimes the human will inspect a location, but miss the target object, which inhibits the robot from searching the same place and finding it. It suggests an improvement in our approach by learning different observation models for the human teammates with different searching capabilities, which is beyond the scope of this dissertation. Figure 7.9 shows similar results for searching for the screwdriver. The experiments show that the proposed cooperative search is efficient.

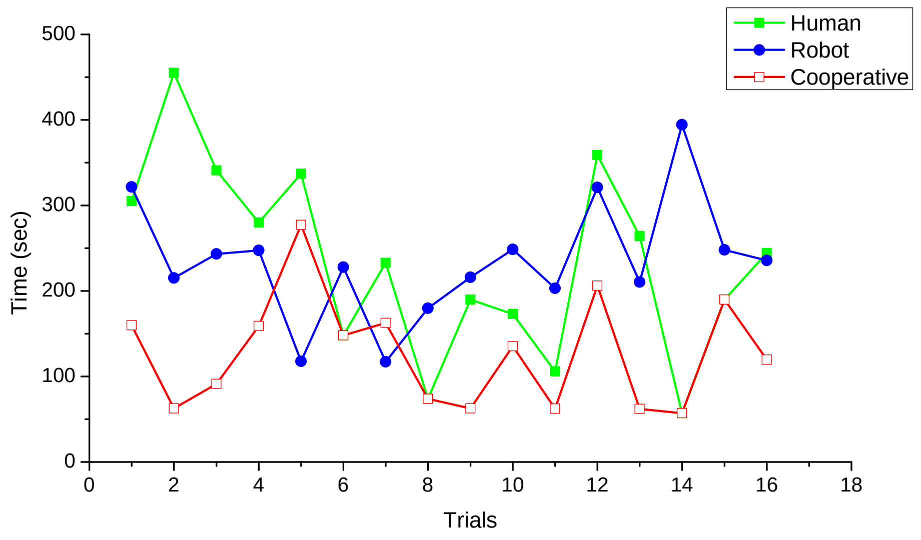
Table 7.1. Average time cost with different search strategies. (sec)

	Human	Robot	No-Co	Cooperation
Book	234.1	217.6	181.0	127.4
Screwdriver	189.7	221.9	170.5	147.2
Average	211.9	219.8	175.8	137.3

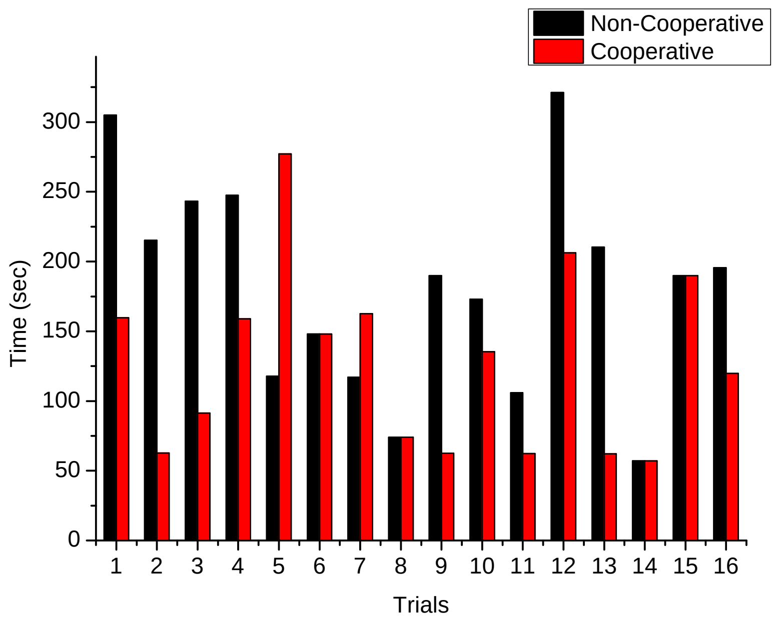
7.6.1.2 Interface Design Experiments

In this part, the search efficiency of the explicit and implicit user interface is compared. We measure the classification accuracy of the partly observed subject trajectory when searching for the book and the screwdriver. Figure 7.10 shows that the accuracy improves with the number of observations. Given 40 seconds of observation, the robot is able to predict the search activities with 75% accuracy. It can be used as the time for the robot to initiate the dialog with the human to clarify the remaining ambiguity, if the subject's cognitive load is lower than a threshold $M < T_{ML}$. In the case of $M > T_{ML}$, the robot waits longer to collect more data for better human activity recognition.

We evaluate the search efficiency when using the implicit interface design. With implicit interaction, the robot needs time to predict what human is looking for, which causes a delay for the subsequent cooperative search. We want to evaluate if this delay causes a significant degradation of the cooperative search efficiency. In all trials in this experiment, the human search was started first. The robot waited for 40 seconds and then joined the human search. The search efficiency with explicit

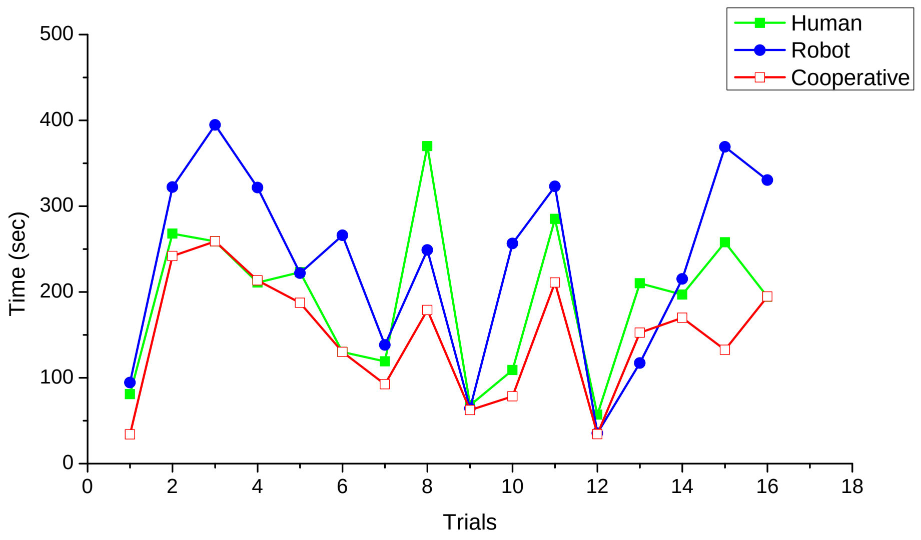


(a)

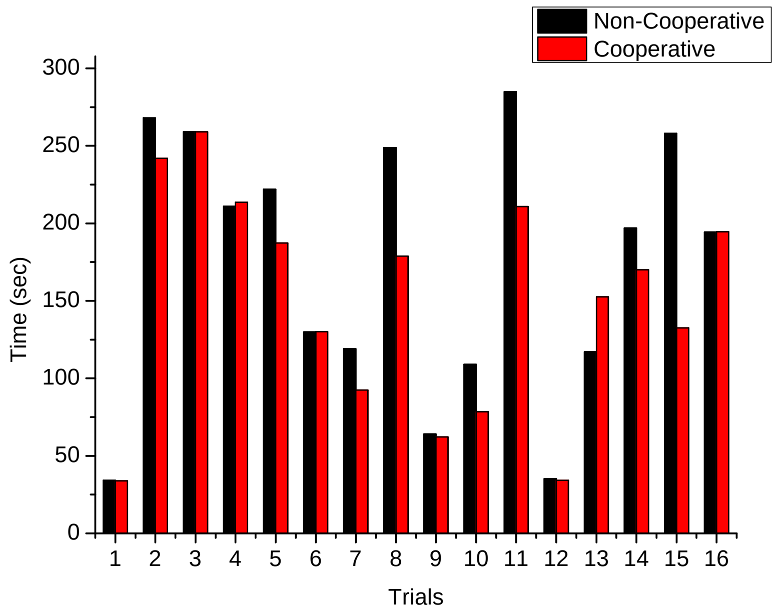


(b)

Figure 7.8. Search efficiency (for BOOK)



(a)



(b)

Figure 7.9. Search efficiency (for SCREWDRIVER)

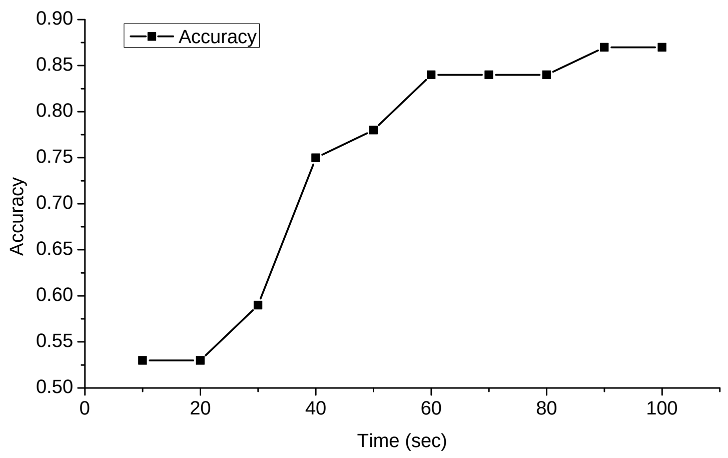


Figure 7.10. Accuracy of search activity prediction

and implicit interactions are given in Figure 7.11. It can be seen that both explicit and implicit cooperations outperforms the non-cooperative search. The average time cost with implicit interactions is only 17s (13.4%) higher than that with the explicit interactions. The dashed line in Figure 7.11 represents the actual driving time of the robot. It shows that implicit cooperation saves energy by decreasing costs associated with navigation and mobility.

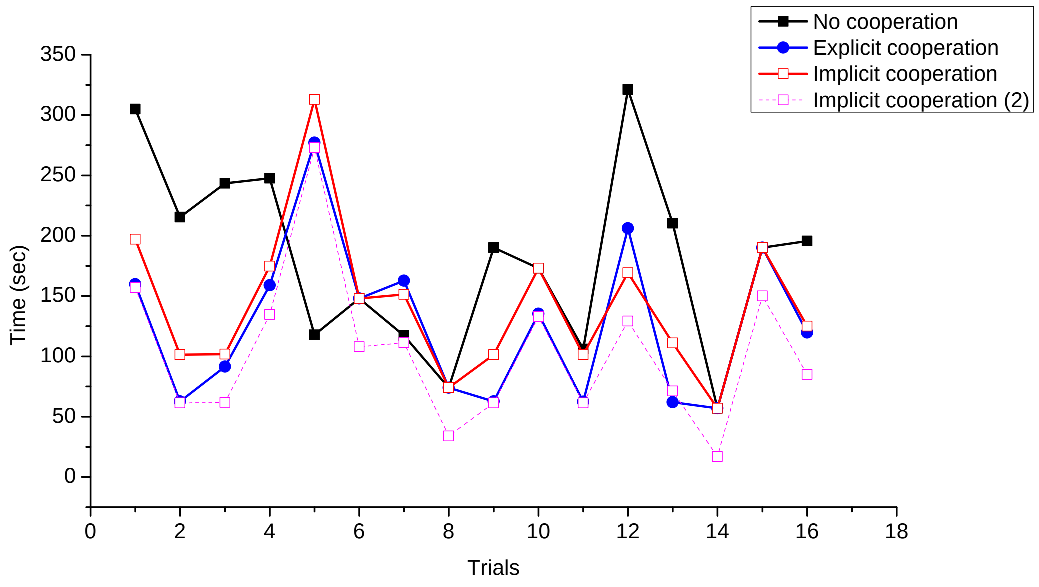


Figure 7.11. Comparison in search efficiency

7.6.2 Real World Experiments

Real world experiments were conducted in the *LPR-1* environment and the results are compared to the human search performance in *LPR-1*.

7.6.2.1 Activity-Based Human-Robot Joint Search

Figure 7.12 summarizes an experiment designed to evaluate the performance of a human-robot search team in which the robot infers the human search intention and chooses search locations to compliment the human peer. Two subjects participated in five trials each. The BOOK target was randomly placed in its possible locations (on desks or shelves). The search team consisted of the uBot-5 and the human client and the camera array was used predict the next search location for the human searcher. The uBot-5 carried an RFID reader to find a tag attached to the book. The average time cost for the human-robot team to find the target was 81 seconds. Figure 7.12 shows that the proposed H-R (Human-Robot) search strategy improves the search performance by 25% compared to search by the human alone.

Figure 7.12 shows a tendency to higher performance with the proposed strategy, but it was not statistical significant. More experiments need to be conducted in future work (Section 8).

7.6.2.2 Gesture-Based Human-Robot Joint Search

The last experiment was performed to evaluate the effectiveness of the gesture guided search. In this experiment the robot always received a pointing gesture command to search the areas that the human peer designates. Every time the robot finishes the search in a location, it returns to the center of the search space, and the human peer provides the robot the next command. Three subjects executed thirty trials each. All other experiment conditions were the same as the previous experiment.

Figure 7.13 illustrates the search performance of this strategy. The average time for the team to find the target was 94 seconds. From the result we contend that

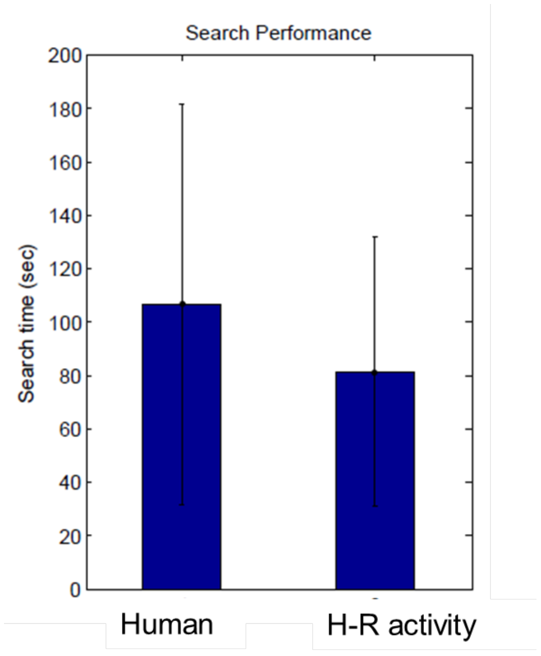


Figure 7.12. Performance comparison of search strategies. The left bar represents human search performance. The right bar represents the performance of the human-robot cooperative search strategy with human activity modeling.

gesture-guided search is efficient, but that does not represent a significant improvement over human-alone search. I believe the reason is partly because the current gesture interface is inefficient. The constraints required for recognizing gestures are costly to search performance because the Kinect sensor and OpenNI library only support kinematic reconstruction for gesture recognition at ranges between 3 and 5 meters. Longer range gesture recognition will likely improve the performance of this form of human-robot cooperative search.

7.7 Summary

In this chapter a human-robot cooperative search scheme is presented. The robot agents infer the current state and intention of the human peer using a human search activity model that was acquired prior to actual search. By inferring the human search policy, the robot chooses complementary actions to achieve efficient cooperation with

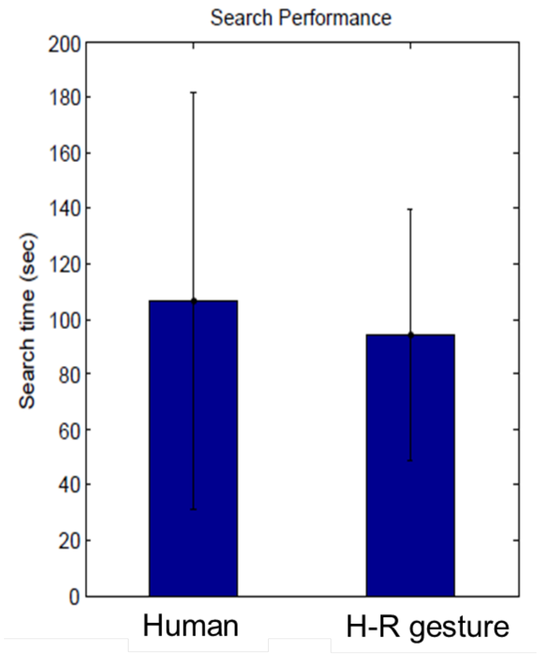


Figure 7.13. Performance comparison of search strategies. The left bar represents human search performance. The right bar represents the performance of the human-robot cooperative search strategy with human gesture guidance.

human peers. An implicit interface design for robot assisted tasks was also proposed, which allows the robot to infer the intention of the user and to provide assistance autonomously. It reduces the cognitive workload of the user and is, therefore, useful for elder care applications. The experimental results indicate that there is promise in this technology.

In this dissertation, we focus on investigating the receptive behavior of the robot in cases, where the human subject is not highly sensitive to the behavior of the robots. An example scenario is a human subject embedded in an array of stationary surveillance cameras. However, our results show that this model also applies in applications where the behavior of the human subject can be altered by the robot's presence (e.g., H-R cooperation in an emergency response team). The same model makes relevant predictions about actions that improve the performance of the team when informed by the two-agent search history. It does not depend on which agents contributed to the history given the assumption that the human and robot have similar observation capabilities. In the future, we plan to evaluate the added value (versus cost) of modeling joint team activity on the performance of search tasks.

CHAPTER 8
CONCLUSIONS AND FUTURE WORK

8.1 Conclusions

In this dissertation, I have proposed and presented an opportunistic service-oriented approach that supports applications important to service robots, particularly those involved in future residential healthcare, eldercare, and emergency response systems and specifically those that depend on human-robot cooperative search. With the proposed approach, this dissertation gives a comprehensive description on how to make an effective object search system for residential assisted-living environments. Variants of approaches and algorithms are proposed for the object search application. In this dissertation I have presented (1) a tracking-based object retrieval strategy that uses low-power embedded cameras. A novel dual-camera sensor structure and an energy-efficient object recognition approach are proposed; (2) a probabilistic search strategy for a single robot which achieves better coverage than the tracking-based approach and has comparable performance to human search; (3) a decentralized and cooperative search strategy with which the system is tolerant to failures and interruptions of the search agents; and (4) the investigation on human-robot cooperative search strategies. Experimental results are presented to show how increasingly efficient search performance is acquired as more sensorimotor resources are employed.

In this dissertation human factors on search system performance are thoroughly studied. We studied human search with real world data and revealed that patterns exist in human search behaviors. These patterns can be used as prior knowledges about the distribution of the target object to improve the search performance. With the learned human search patterns, the robot agents can also infer the current state and the intention of the human peer, and chooses complementary actions to achieve efficient cooperation with human peers.

8.2 Directions For Future Work

In this work physical robot experiments were conducted to evaluate the proposed approaches. However, the experiments presented in this thesis are necessarily limited and more experiments will be performed on the physical robot and with more human subjects. In the future, we plan to evaluate the added value (versus cost) of modeling joint team activity on the performance of search tasks. The performance of our approaches when the number of objects and human activities scale up will also be evaluated.

In addition, the robot should be able to show not only efficient but also legible behavior. Expressive behaviors of the robot to naturally communicate with human need to be studied. Furthermore, the proposed opportunistic service oriented approach can be applied to other assisted-living applications, e.g., fall prevention and ADL analysis.

BIBLIOGRAPHY

[1] Intel open source computer vision library.

[2] Common object request broker architecture (corba). http://www.corba.org.

[3] Crossbow wireless sensor platform. http://www.xbow.com/.

[4] Enalab camera. http://enaweb.eng.yale.edu/drupal/.

[5] Intel mote2 platform. http://www.xbow.com/products.

[6] Intel open source computer vision library. http://www.intel.com/technology/computing/opencv/.

[7] Object management group (omg). http: //www.omg.org.

[8] W3c. "W3C Working Group Note 11 February 2004", http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211.

[9] Wikipedia about service-oriented architecture. Available: http://en.wikipedia.org/wiki/Service-oriented architecture.

[10] Cyclops platform. http://enaweb.eng.yale.edu/drupal/.

[11] A. Wood, G. Virone, T. Doan Q. Cao L. Selavo Y. Wu L. Fang Z. He S. Lin J. Stankovic. Alarm-net: Wireless sensor networks for assisted-living and residential monitoring. *Technical Report CS-2006-11, Department of Computer Science, University of Virginia* (2006).

[12] Abowd, G. A. Bobick, I. Essa E. Mynatt, and Rogers, W. The aware home: Developing technologies for successful aging. *Proceedings of AAAI Workshop and Automation as a Care Giver* (2002).

[13] Acar, Ercan, Choset, Howie, Zhang, Yangang, and Schervish, Mark. Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods. International Journal of Robotics Research, 2003.

[14] Ahmadi, Mazda, and Stone, Peter. A multi-robot system for continuous area sweeping tasks. ICRA, 2009.

[15] Ahmed, Nisar, Sample, Eric, Yang, Tsung-Lin, Lee, Daniel, de la Garza, Lucas, Elsamadisi, Ahmed, Sullivan, Arturo, Wang, Kai, Lao, Xinxiang, Tse, Rina, and Campbell, Mark. Towards cooperative bayesian human-robot perception: Theory, experiments, opportunities. Workshop on Intelligent Robotic Systems, 27th AAAI Conference on Artificial Intelligence, 2013.

[16] Amoretti, M. & Reggiani, M. Architectural paradigms for robotics applications. Advanced Engineering Informatics, 2010. Informatics for cognitive robots.

[17] Ardizzone, E., Cascia, M.L., Re, G.L., and Ortolani, M. An integrated architecture for surveillance and monitoring in an archaeological site. *ACM Int. Workshop on Video Surveillance and Sensor Networks* (2005).

[18] Avvenuti, Marco, Corsini, Paolo, Masci, Paolo, and Vecchio, Alessio. Opportunistic computing for wireless sensor networks. Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on.

[19] Basu, S., Banerjee, A., and Mooney, R. J. Active semi-supervision for pairwise constrained clustering. *Proc. of the 4th SIAM Int. Conf. on Data Mining* (2004).

[20] Basu, S., Banerjee, A., and Mooney, R. J. Active semi-supervision for pairwise constrained clustering. *Proc. of the 4th SIAM Int. Conf. on Data Mining* (2004).

[21] Bauer, Andrea, Klasing, Klaas, Lidoris, Georgios, MÃijhlbauer, Quirin, RohrmÃijller, Florian, Sosnowski, Stefan, Xu, Tingting, KÃijhnlenz, Kolja, Wollherr, Dirk, and Buss, Martin. The autonomous city explorer: Towards natural human-robot interaction in urban environments. Intl. J. of Social Robotics, 2002.

[22] Bharucha, A.J. Caremedia: Automated video and sensor analysis for geriatric care. *29th Annual Meeting of the American Medical Directors Association* (2006).

[23] BISHOP, C. M. Pattern recognition and machine learning. Information Science and Statistics, Springer-Verlag New York, Inc., 2006.

[24] Borriello, G., Brunette, W., Hall, M., Hartung, C., and Tangney, C. Reminding about tagged objects using passive rfids. *Proc. of the 8th Ubicomp Conference* (2006).

[25] Bourgault, Frederic, Chokshi, Aakash, Wang, John, Shah, Danelle, Schoenberg, Jonathan, Iyer, Ramnath, Cedano, Franco, and Campbell, Mark. Scalable bayesian human-robot cooperation in mobile sensor networks. IROS 2008.

[26] Bourgault, Frederic, Furukawa, Tomonari, and Durrant-Whyte, Hugh F. Coordinated decentralized search for a lost target in a bayesian world. *IROS* (2006).

[27] Bourgault, Frederic, Furukawa, Tomonari, and Durrant-Whyte, Hugh F. Optimal search for a lost target in a bayesian world. *Field and Service Robotics* (2006).

[28] Brooks, A., Kaupp, T., Makarenko, A., Williams, S., and Oreback, A. Towards component-based robotics. in IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2005.

[29] Carlson, Tom, and Demiris, Yiannis. Human-wheelchair collaboration through prediction of intention and adaptive assistance. ICRA 2008.

[30] Cerrada, C. Salamanca, S. Perez E. Cerrada J.A. Abad I. Fusion of 3d vision techniques and rfid technology for object recognition in complex scenes. *IEEE International Symposium on Intelligent Signal Processing* (2007).

[31] Chen, Y., Du, Z., and Garcia-Acosta, M. Robot as a service in cloud computing. in Service Oriented System Engineering (SOSE), 2010 Fifth IEEE International Symposium on. IEEE 2010, pp. 151âĂŞ158.

[32] Chong, N.Y., Hongu H. Miyazaki M. Takemura K. Ohara K. Ohba K. Hirai S., and Tanie, K. Robots on self-organizing knowledge networks. *In Proceedings of International Conference on Robotics and Automation* (2004).

[33] Connolly, Christopher I., and Grupen, Roderic A. Harmonic control. In Proc. 1992 Int. Symp. Intelligent Control.

[34] Conti, M., and Kumar, M. Opportunities in opportunistic computing. Computer, Jan. 2010, pp. 42-50.

[35] Dan Xie, Tingxin Yan, Deepak Ganesan Allen Hanson. Design and implementation of a dual-camera wireless sensor network for object retrieval. *IPSN* (2008).

[36] Deegan, Patrick, Grupen, Rod, Hanson, Allen, Horrell, Emily, Ou, Shichao, Riseman, Edward, Sen, Shiraj, Thibodeau, Bryan, Williams, Adam, and Xie, Dan. Mobile manipulators for assisted living in residential settings. Autonomous Robots, Special Issue, 2007.

[37] DeLima, P., and Pack, D. Toward developing an optimal cooperative search algorithm for multiple unmanned aerial vehicles. *International Symposium on Collaborative Technologies and Systems* (2008).

[38] Doshi, F., and Roy, N. Efficient model learning for dialog management. In Proc. ACM/IEEE International Conference on Human-robot Interaction (HRI), pages 65âĂŞ72, New York, NY, USA, 2007. ACM.

[39] F. Saidi, O. Stasse, and Yokoi, K. A visual attention framework for a visual search by a humanoid robot. *in IEEE-RAS International Conference on Humanoid Robots* (2006).

[40] Flueckiger, Lorenzo, and Utz, Hans. Field tested service oriented robotic architecture: Case study. Published at: Publication: Other iSAIRAS 2012, Turin, Italy Date: 09/04/12.

[41] Fong, T., and Baur, C. Thorpe C. Robot, asker of questions. Robotics and Autonomous Systems, 2003.

[42] Fong, Terrence, Thorpe, Charles, and Baur, Charles. Multi-robot remote driving with collaborative control. IEEE Tran. on Industrial Electronics 50(4): 699-704, 2003.

[43] G. Metta, P. Fitzpatrick, and Natale., L. Yarp: Yet another robot platform. International Journal of Advanced Robotics Systems, special issue of Software Development and Integration in Robotics, March 2006.

[44] G. Pardo-Castellote, S. Schneider, and Hamilton, M. Ndds: The real-time publish-subscribe middleware. In Proceedings of the IEEE Real-Time Systems Symposium, 1999.

[45] Gao, J., Tan, P.-N., and Cheng, H. Semi-supervised clustering with partial background information. *Proc. of the 6th SIAM Int. Conf. on Data Mining* (2006).

[46] Garage, Willow. Robot operating system (ros). http://www.willowgarage.com/pages/software/ros-platform, 2012.

[47] Gnawali, O., Greenstein, B., Jang, K., Joki, A., Paek, J., Vieira, M., Estrin, D., Govindan, R., and Kohler, E. The tenet architecture for tiered sensor networks. *ACM Conf. on Embedded Networked Sensor Systems* (2006).

[48] Hartigan, J. A.; Wong, M. A. Algorithm as 136: A k-means clustering algorithm. Journal of the Royal Statistical Society, Series C (Applied Statistics) 28 (1): 100âĂŞ108. JSTOR 2346830, 1979.

[49] Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome. The elements of statistical learning (2nd ed.). New York: Springer. pp. 520âĂŞ528. ISBN 0-387-84857-6.

[50] Hengstler, S., Prashanth, D., Fong, S., and Aghajan, H. Mesheye: a hybrid-resolution smart camera mote for applications in distributed intelligent surveillance. *Int. Conf. on Information Processing in Sensor Networks, SPOTS* (2007).

[51] HESTAND, PAUL D. A service oriented architecture for robotic platforms. PhD Dissertation, University of Massachusetts Lowell.

[52] Hill, Susan G., and Bodt, Barry. A field experiment of autonomous mobility: operator workload for one and two robots. HRI 2007.

[53] ichiro Kumagai, Shin, and Mori, Yasuchika. Development of a search robot - sensor fusion of rfid and ccd camera. *In Proc of SICE* (2007).

[54] Jackson, J. Microsoft robotics studio: A technical introduction. Robotics Automation Magazine, IEEE, 14(4):82-87, December 2007.

[55] Jeffrey King, Raja Bose, Steven Pickles Abdelsalam Helal Steve Vander Ploeg James Russo. Atlas: A service-oriented sensor platform. In Proceedings of the Workshop on Practical Issues in Building Sensor Network Applications, Tampa, Florida, November 2006. IEEE.

[56] Johnson, Carlotta A., Adams, Julie A., and Kawamura, Kazuhiko. Evaluation of an enhanced human-robot interface. IEEE International Conference on Systems, Man and Cybernetics, 2003.

[57] Kadous, M. Waleed, Sheh, Raymond Ka-Man, and Sammut, Claude. Effective user interface design for rescue robotics. HRI 2006.

[58] Kamol, P.; Nikolaidis, S.; Ueda R.; Arai T. Rfid based object localization system using ceiling cameras with particle filter. *Proceedings of the Future Generation Communication and Networking (FGCN 2007)* (2007).

[59] Kanehiro, Fumio, Ishiwata, Yoichi, Saito, Hajime, Akachi, Kazuhiko, Miyamori, Gou, Isozumi, Takakatsu, Kaneko, Kenji, and Hirukawa, Hirohisa. Distributed control system of humanoid robots based on real-time ethernet. in: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2006.

[60] Kantor, George A, Singh, Sanjiv, Peterson, R., Rus, Daniela, Das, A., Kumar, V., Pereira, G., and Spletzer, J. Distributed search and rescue with robot and sensor teams. International Conference on Field and Service Robotics, 2003.

[61] Karuppiah, Deepak, Grupen, Rod, Hanson, Allen, and Riseman, Edward. Smart resource reconfiguration by exploiting dynamics in perceptual tasks. Proceedings of the 2005 International Conference on Robotics and Systems (IROS), Edmonton, Alberta, August, 2005.

[62] Kaupp, Tobias, Douillard, Bertrand, Ramos, Fabio T., Makarenko, A., and Upcroft, Ben. Shared environment representation for a human-robot team performing information fusion. J. of Field Robotics, 2007.

[63] Kehoe, B., Matsukawa, A., Candido, S., Kuffner, J., and Goldberg, K. Cloud-based robot grasping with the google object recognition engine. in IEEE International Conference on Robotics and Automation. IEEE, 2013.

[64] Kehoe, Ben, Matsukawa, Akihiro, Candido, Sal, Kuffner, James, and Goldberg, Ken. Cloud-based robot grasping with the google object recognition engine. IEEE International Conference on Robotics and Automation. Karlsruhe, Germany. May 2013.

[65] Kientz, J.A., Patel, S.N., Tyebkhan, A.Z., Gane, B., Wiley, J., and Abowd, G.D. Where's my stuff?: design and evaluation of a mobile system for locating lost items for the visually impaired. *ACM conf. on Computers and accessibility* (2006).

[66] Kim, J.Y., Im C.J. Lee S.W., and Lee, H.G. Object recognition using smart tags and stereo vision system on pan-tilt mechanism. *In Proceedings of ICCAS* (2005).

[67] Kulkarni, P., Ganesan, D., Shenoy, P., and Lu, Q. Senseye: A multi-tier camera sensor network. *ACM Multimedia* (2005).

[68] Kurz, Marc, Holzl, Gerold, Ferscha, Alois, Calatroni, Alberto, Roggen, Daniel, Troster, Gerhard, Sagha, Hesam, Chavarriaga, Ricardo, del R. Millan, Jose, Bannach, David, Kunze, Kai, and Lukowicz, Paul. The opportunity framework and data processing ecosystem for opportunistic activity and context recognition. International Journal of Sensors, Wireless Communications and Control, 2011.

[69] L. R. Rabiner, B. H. Juang. An introduction to hidden markov models. IEEE ASSAP Magazine, 1986.

[70] Lee, Joongjae, Jie, Minseok, Kim, Seungsu, and Jung, Munho. Balloon burster: A corba-based visual servoing for humanoid robot in a distributed environment. in Annual Conference SICE, 2007.

[71] Liu, X., Corner, M., and Shenoy, P. Ferret: Rfid localization for pervasive multimedia. *Ubicomp Conference* (2006).

[72] Lorenzo FlÃijckiger, Vinh To, Hans Utz. Service-oriented robotic architecture supporting a lunar analog test. International Symposium on Artificial Intelligence, Robotics, and Automation in Space (iSAIRAS) 2008.

[73] Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision* (2004).

[74] Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision* (2004).

[75] Lu, Qiang, and Han, Qing-Long. Designing a cooperative control system for multiple mobile robots to locate the source of odor. In Proc. of 18th IFAC World Congress, 2011.

[76] Malan, D., Fulford-Jones, T., Welsh, M., and Moulton, S. Codeblue: An ad hoc sensor network infrastructure for emergency medical care. *Int. Workshop on Wearable and Implantable Body Sensor Networks* (2004).

[77] Marjovi, Ali, Gonc, Joao, alo Nunes, Marques, Lino, and de Almeida, Anbal. Multi-robot exploration and fire searching. IROS, 2009.

[78] Marti, M., Kusy, B., Simon, G., and Ldeczi, A. The flooding time synchronization protocol. *ACM Conf. on Embedded Networked Sensor Systems* (2004).

[79] Mikolajczyk, K., and Schmid, C. A performance evaluation of local descriptors. *IEEE Trans. Pattern Analysis and Machine Intelligence* (2005).

[80] Murphy, Robin. Disaster robotics. MIT Press.

[81] Murphy, Robin Roberson. Human-robot interaction in rescue robotics. Systems, Man and Cybernetics, Part C, IEEE Tran. on 34(2): 138-153, 2004.

[82] Nakada, T., Kanai, H., and Kunifuji, S. A support system for finding lost objects using spotlight. *Int. conf. on Mobile Human Computer Interaction* (2005).

[83] Nelson, R.C., and Green, I.A. Tracking objects using recognition. *Proc. Int. Conf on Pattern Recognition* (2002).

[84] Nevatia, Yashodhan, Stoyanov, Todor, Rathnam, Ravi, Pfingsthorn, Max, Markov, Stefan, Ambrus, Rares, and Birk, Andreas. Augmented autonomy: Improving human-robot team performance in urban search and rescue. IROS 2008.

[85] Nguyen, Nam T., Phung, Dinh Q., Venkatesh, Svetha, and Bui, Hung. Learning and detecting activities from movement trajectories using the hierarchical hidden markov model. CVPR 2005.

[86] OMG. Corba component model 4.0 specification, version 4.0 (april 2006). URL http://www.omg.org/docs/formal/06- 04- 01.pdf.

[87] Orr, R.J., Raymond, R., Berman, J., and Seay, A.F. A system for finding frequently lost objects in the home. *Georgia Tech GVU Technical Report: GIT-GVU-99-24* (1999).

[88] Peters, R.E., Pak, R., Abowd, G.D., Fisk, A.D., and Rogers, W.A. Finding lost objects: Informing the design of ubiquitous computing services for the home. *Georgia Tech GVU Technical Report: GIT-GVU-04-01* (2004).

[89] PrimeSense. http://openni.org/. 2011.

[90] Quelhas, P., and Odobez, J.-M. Natural scene image modeling using color and texture visterims. *ACM Int. Conf. on Image and Video Retrieval* (2006).

[91] Quelhas, P., and Odobez, J.-M. Natural scene image modeling using color and texture visterims. *ACM Int. Conf. on Image and Video Retrieval* (2006).

[92] R. de Molengraft, van, M. Beetz, and Fukuda, T. A special issue toward a www for robots [from the guest editors]. Robotics Automation Magazine, IEEE, 18(2):20, june 2011.

[93] Rabiner, Lawrence R. A tutorial on hidden markov models and selected application in speech recognition. Proceedings of the IEEE 77 (1989).

[94] Rahimi, M., Baer, R., Warrior, J., Estrin, D., and Srivastava, M. Cyclops: In situ image sensing and interpretation in wireless sensor networks. *ACM Conf. on Embedded Networked Sensor Systems* (2005).

[95] Robbel, Philipp, Demirdjian, David, and Breazeal, Cynthia. Simultaneous localization and mapping with people. RSS 2011 Workshop on RGB-D Cameras, 2011.

[96] S. S. Intille, K. Larson, J. Beaudin E. Munguia Tapia P. Kaushik J. Nawyn, and McLeish, T.J. The placelab: a live-in laboratory for pervasive computing research (video). *Proceedings of Pervasive 2005 Video Program* (2005).

[97] Saidi, F., Stasse O. Yokoi K. Kanehiro F. Online object search with a humanoid robot. *IROS* (2007).

[98] Schgerl, P., Sorschag, R., Bailer, W., and Thallinger, G. Object re-detection using sift and mpeg-7 color descriptors. *Int. Workshop Multimedia Content Analysis and Mining* (2007).

[99] Schmid, C., and Mohr, R. Local greyvalue invariants for image retrieval. *IEEE Trans. PAMI* (1997).

[100] Seung-Ho Baeg, Jae-Han Park, Jaehan Koh Kyung-Wook Park Moonhong Baeg. An object recognition system for a smart home environment on the basis of color and texture descriptors. *IEEE/RSJ International Conference on Intelligent Robots and Systems* (2007).

[101] Sheri Reder, Gwen Ambler, Matthai Philipose Susan Hedrick. Technology and long-term care (tlc): A pilot evaluation of remote monitoring of elders. *Gerontechnology 9(1)* (2010).

[102] Sheridan, T. Humans and automation. Santa Monica, CA Wiley.

[103] Sujan, V. A., and Dubowsky, S. Efficient information-based visual robotic mapping in unstructured environments. *International Journal of Robotics Research* (2005).

[104] Takeda, Kenro, Nasu, Yasuo, Capi, Genci, Yamano, Mitsuhiro, Barolli, Leonard, and Mitobe, Kazuhisa. A corba-based approach for humanoid robot control. Industrial Robot: An International Journal, 2001.

[105] Tavakkoli, Alireza, Kelley, Richard, King, Christopher, Nicolescu, Mircea, Nicolescu, Monica N., and Bebis, George. A vision-based architecture for intent recognition. ISVC (2) 2007: 173-182.

[106] Teixeira, T., Lymberopoulos, D., Culurciello, E., Aloimonos, Y., and Savvides, A. A lightweight camera sensor network operating on symbolic information. *Proc. of 1st Workshop on Distributed Smart Cameras* (2006).

[107] Troy McDaniel, Kanav Kahol, Daniel Villanueva Sethuraman Panchanathan. Integration of rfid and computer vision for remote object perception for individuals who are blind. *Proceedings of the 2008 Ambi-Sys workshop on Haptic user interfaces in ambient media systems* (2008).

[108] Tyrer, Harry W., Alwan, Majd, Demiris, George, He, Zhihai (Henry), Keller, Jim, Skubic, Marjorie, and Rantz, Marilyn. Technology for successful aging. In *Proceedings of the 28th IEEE EMBS Annual International Conference*. IEEE Computer Society, Washington, DC, 2006.

[109] van de Weijer, J., and Schmid, C. Coloring local feature extraction. *European Conf. on Computer Vision* (2006).

[110] Wagstaff, K., Cardie, C., Rogers, S., and Schroedl, S. Constrained k-means clustering with background knowledge. *Int. Conf. on Machine Learning* (2001).

[111] Williams, A., Xie, D., Ou, S., Grupen, R., Hanson, A., and Riseman, E. Distributed smart cameras for aging in place. *Proc. of 1st Workshop on Distributed Smart Cameras* (2006).

[112] Wilson, P., Prashanth, D., and Aghajan, H. Utilizing rfid signaling scheme for localization of stationary objects and speed estimation of mobile objects. *IEEE Int. Conf. on RFID* (March 2007).

[113] Wixson, L. E. Gaze selection for visual search. *Ph.D. dissertation, Department of Computer Science, University of Rochester* (1994).

[114] Xie, Dan, Grupen, Roderic, and Hanson, Allen. Context-aware search using cooperative agents in a smart environment. IEEE Workshop on Applications of Computer Vision (WACV). December, 2009.

[115] Xie, Dan, Lin, Yun, Grupen, Roderic, and Hanson, Allen. Intention-based coordination and interface design for human-robot cooperative search. In Proceedings of the 2011 International Conference on Robotics and Systems (IROS), San Francisco, California, September, 2011.

[116] Xie, Dan, Yan, Tingxin, Ganesan, Deepak, and Hanson, Allen. *Proceedings of the 7th international conference on Information Processing in Sensor Networks* (2008).

[117] Ye, Y., and Tsotsos, J. K. Sensor planning for 3d object search. *Computer Vision and Image Understanding, vol. 73, no. 2, pp. 145âĂŞ168* (1999).

[118] Ye, Yiming, and Tsotsos, John K. Sensor planning in 3d object search. *CVIU* (1996).

[119] Yoann Maurel, Ada Diaconescu, and Lalanda, Philippe. Ceylon : A service-oriented framework for building autonomic managers.