EXTRACTION OF SALIENT FEATURES FROM SENSORY-MOTOR SEQUENCES FOR MOBILE ROBOT NAVIGATION

By

Jian Peng

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Electrical Engineering

May, 2004

Nashville, Tennessee

Approved:

Professor Richard Alan Peters, II

Professor Mitch Wilkes

Professor Joseph S. Lappin

Professor Kazuhiko Kawamura

Professor David C. Noelle

DEDICATION

This work is dedicated with love and appreciation to

my parents, Huixian Peng and Yuming Yu

my brother, Jun Peng, my sister, Li Peng

and my fiancée, Fen Jiang

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Richard Alan Peters II, for supporting me during my years at Vanderbilt. Without his support and ideas it would not be possible for me to obtain my Ph.D. degree. I would like to thank CIS Director Dr. Kazuhiko Kawamura for supplying me and other students such a wonderful research facilities. I would like to thank Dr. Mitch Wilkes for many wonderful discussions, and his family, Jane and Daniel, for giving me another home in Nashville. I would like to thank Dr. David Noelle for several inspiring discussions and his meticulous review of my dissertation draft. I would like to thank Dr. Joe Lappin for sitting on my committee. I thank the Department of Electrical Engineering and Computer Science, School of Engineering, Vanderbilt University for providing me a state-of-art research facility and necessary financial support.

Many thanks to my previous and current colleagues in the Center for Intelligent Systems, among them are Dr. Carlotta Johnson, Dr. Tamara Rogers, Kim Hambuchen, Dr. Anthony Alford, Dr. Bugra Koku, Dr. Siripun Thongchai, Dr. Nilas Ponchai, Dr. Mark Cambron, Dr. Steve Northrup, Dr. Atit Srikaewa, Turker Keskinpala, Hande Keskinpala, Hiko Muguruma, Ed Brown, Xinyu Ao, and others that I don't write your name here. Special thanks to Ms. Flo Fottrell for her excellent administrative work.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Overview

The ultimate goal of intelligent robotics research is to make machines behave intelligently, that is to learn from previous experience and adapt to the ever-changing environment. A robot estimates the environment changes through sensing, and responds to these changes through actions. The aim of robot control is to construct an appropriate structure between sensing and action to perform a task. Nehmzow and Mitchell proposed that the following two questions need to be answered for a robot control system [1]:

- What happens inside the robot, after sensor signals have been received and before actuator signals have been generated, independent of the particular control strategy adopted?
- Which functions ought to be computed by the controller in order to make the robot achieve a specified task?

The answer to the first question is a robot control architecture (cf. Chapter II). In the short history of robotics research and artificial intelligence (AI), many different control paradigms have been tried and have shown both their strengths and weaknesses to accomplish this goal [2]. An early attempt was the SMPA (sense-model-plan-act) architecture, which is highly based on traditional artificial intelligence (AI) [3]. SMPA based robots are capable of executing tasks in highly structured environments, however, they are not able to adapt to quick and/or wide variations in a given task or in the environment. In the late 1980's, Brooks and others introduced the behavior-based approach (cf. Chapter II), and they have shown that behavior-based robotics systems perform well in unstructured environments [4].

In behavior-based systems, the second question becomes the problem of how to construct behavior. There are at least two approaches to constructing behaviors: hard-wire and learning. In pure behavior-based systems, behaviors are hard-wired, i.e., the designer must choose the basic behaviors and must define the behavior's starting and stopping conditions [4]. In effect, the designer must anticipate which sensory events should precede which action. This is feasible only in simple applications. When the system complexity increases, however, it is difficult, if not possible, for the designer to anticipate the sensory information available to the robot which, in an unforeseen situation, will be salient for action selection. Learning is the preferred method in these applications.

Many researchers have introduced different approaches to learn behaviors for a robot. Mataric's TOTO robot learns while acting in the environment and forms a spatio-temporal sensory-motor description of the environment [5]. The map indicates the sensory and motor status of the robot at a particular point in space, and the robot learns how to sequence basic behaviors from sensory input. However, the robot does not determine the sensory signals for the given task; the robot learns the sequencing of basic behaviors rather than the salient sensory-motor features that define the basic behaviors themselves.

Pfeifer's seminal work on sensory-motor coordination (SMC) [6] demonstrated that SMC could be used to form categories of objects and to discriminate between them. SMC principle states that perception and action are not two independent processes; instead, they are highly coupled. Pfeifer's experiments (cf. Chapter IV) showed that perception and classification were greatly simplified if the agent's motor ability was taken advantage of.

However, his experiments were limited to two categories using a limited number of sensors. While his approach was believed to scale up, it was not demonstrated.

Rosenstein and Cohen have developed a method for agents to learn categories through interaction with the environment using the real-time detection of events [7]. In order to recognize events, Rosenstein and Cohen used simple, conspicuous patterns (for example, rising edge from a bump switch) as event markers (cf. Chapter V, section III). Using a five second window, the most recent sensor readings are correlated with one of four templates (sharp edge, long sharp edge, slope-then-plateau, and plateau-then-slope). A learning algorithm is then initiated to form categories from the correlation results.

All these learning methods are unsupervised; categories self-organize from the sensory-motor couplings experienced by the robot. A second approach takes advantage of the fact that a robot can be controlled by a person. When a person controls a robot by initiating and terminating each of its actions in sequence, the person's SMC causes the robot to act purposefully. That is, the operator watches the robot and controls the motion of the robot based on his or her own observations. If during such operation the robot records its own sensory information, it could, perhaps, associate its own sensory signals with particular actions. This approach would bootstrap the process of acquiring SMC and is analogous to a parent helping a child to walk by holding his or her hand while giving encouragement. The child still must learn how to control his or her actions, but the parent's assistance and encouragement aids in the process.

It is likely that within any task sequence initiated and controlled by a person, certain sensory events will always precede certain changes in motor activity. These events would be approximately similar over repeated performances and therefore detectable through signal analysis. If a given set of sensory events uniquely precedes a given motor activity in a task context, then these events can be used to trigger that activity. Moreover, certain sensor events are almost always going to occur after a specific environmental interaction. For example, the state of the proximity sensor will change as it nears an object. These salient sensory, coupled with the association motor activity, can be used to construct robot behaviors. Also, detection of these events indicates that a certain interaction has occurred, and this could be used to sequence behaviors to carry out a certain task.

Based on these hypothesis, Cambron developed a grasp pose estimation and control method for a humanoid robot at Vanderbilt University [8]. Peters introduced a task learning method for Robonaut, NASA's space-capable, dexterous humanoid robot [9]. This dissertation research extends this approach to mobile robot navigation tasks, that is, to extract features that are salient to navigation tasks and to use these features to navigate autonomously in the learned environment.

Document Organization

The remainder of this dissertation is divided as follows:

Chapter II to V comprise the literature review section. As stated above, robot control has two aspects, architecture and control function. Chapter II discusses architectures. First a generic traditional architecture is presented, then Brooks' seminal behavior-based robotics architecture, subsumption, is discussed. Arkin's schema is presented next. Finally, the current trend of emigrating from behavior-based robotics to cognitive robotics is discussed as well as its impact on robot architecture.

Chapter III gives the definitions of machine learning and robot behavior learning. The reason why learning is necessary is also given. It also discusses the difference between traditional AI learning and robot behavior learning. A tentative robot learning curriculum, i.e., the sequence from low-level reflexive behavior learning to high-level goal-oriented behavior learning is presented.

This dissertation will only survey current approaches in the middle level of this curriculum, namely, attentive competencies and planning capabilities. This is because, firstly, low level reflexive action learning is being researched by others and has a number of plausible solutions, and secondly, high level learning is still very primitive at its current stage. Although middle level behavior learning is less explored, solving this problem could lead to solutions at the higher level. Throughout the learning methods reviewed is a common theme: sensory-motor coordination (SMC). The centrality of SMC is where these current approaches differ most from traditional AI approaches.

Chapter IV presents the sensory-motor coordination principle and describes how this principle can simplify the categorization problem. Traditional approaches to categorization and their problems are discussed first. Then, categorization via sensory-motor coordination is described, with emphasis on how salient features are detected, and why it can simplify the categorization problem when system-environment interactions are taken into account.

Three learning approaches to feature detection are discussed in Chapter V. First, the sensory-motor coordination principle is presented in the context of categorization, and its application in mobile robot categorization. Second, imitation-based learning is presented as a way of learning basis behaviors. Third, episode-based learning is discussed. The relationship between all these approaches is shown at the end. The use of computer vision for detection of salient features is also discussed in the experiment description.

Chapter VI presents the salient feature detection approach used in this research. First, the hypotheses and assumptions employed in the research are presented. Then the procedure to extract salient features is discussed. A formal mathematical description of the extraction procedure then follows.

Chapter VII discusses the experiment platform, a mobile robot named Scooter. First, Scooter's hardware is presented. Second, all the sensors onboard are discussed. Third, the software architecture is introduced.

Chapter VIII presents the experimental results. Two sets of experiments, involving indoor and outdoor environments, were performed. The purpose of these experiments is to validate the salient feature detection method and to evaluate its performance. The results of these experiments and system performance are discussed. A failed method is also presented.

Chapter IV concludes the dissertation, discusses the significance of the results, and suggests some future research directions.

CHAPTER II

ROBOT CONTROL ARCHITECTURE

This chapter will discuss the robot control problem and robot control architectures. It is organized as following. Section I presents two problems that robot control has to solve: architecture and behavior construction. Section II discusses the tradition control architecture and why it has failed in unstructured environment. Section III introduces a new approach, behavior-based robotics. Two such architectures, subsumption and schema, are then presented in Section IV and V. Last section discusses the new trend, from behavior-based robotics to cognitive robotics.

Robot Control Problem: Background

In this dissertation, the term "robot" is used according to [1]. A robot is a physical device that interacts with the real world and that has the following properties:

1. the ability to measure properties of the environment through sensors
2. the ability to perform physical actions of an object-manipulation or locomotion type in the environment
3. the presence of a control structure that determines the action to be performed at any given moment.

Figure 1 shows such a mobile robot, equipped with various sensors to perceive its environment, and actuators that allow it to perform actions in the world. The robot has a particular task to achieve; this is what the box labeled "task description" signifies.

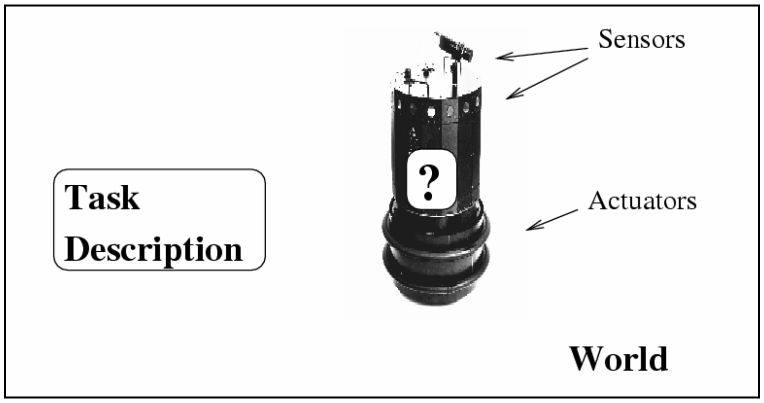


Figure 1: The robot learning scenario [1]

The aim of robot control is to replace the question mark in Figure 1 with some appropriate structure. In order to do this, Nehmzow and Mitchell proposed that the following two questions need to be answered [1]:

1. What happens inside the robot, after sensor signals have been received and before actuator signals have been generated, independent of the particular control strategy adopted?
2. Which functions ought to be computed by the controller in order to make the robot achieve a specified task?

The answer to the first question is a robot control architecture: the focus of this chapter. Chapter III to V survey current answers to the second question. In the short history of robotics research, many different control architectures have been tried and have shown both their strengths and weaknesses. This chapter presents some representative architectures, a generic SMPA architecture followed by two behavior-based architectures, Brook's subsumption and Arkin's schema. The last section discusses the transition from behavior-based robotics to cognitive robotics.

Traditional Robot Control Architecture

Traditional robot control architectures are derived from traditional artificial intelligence (AI) paradigms. In such an architecture, a central controller/planer fuses all sensor readings, builds a world model, plans the next action, and finally gives the control commands to the actuators. Figure 2 depicts such an architecture.

Early robots, such as Shakey [3], adopted this type of architecture. In an SMPA architecture, a programmer pre-selects behavior decisions through rules. Salient sensory information and responses have to be anticipated. This can work in a structured environment, like an assembly line, when everything is predictable *a priori*. However, such architectures tend to fail in an unstructured environment or even in a loosely structured environment due to their inability to adapt to the environment. This is exacerbated by the conflict between 1) SMPA is not robust and sensitive to noise, and 2) the inherently noisy sensory data.

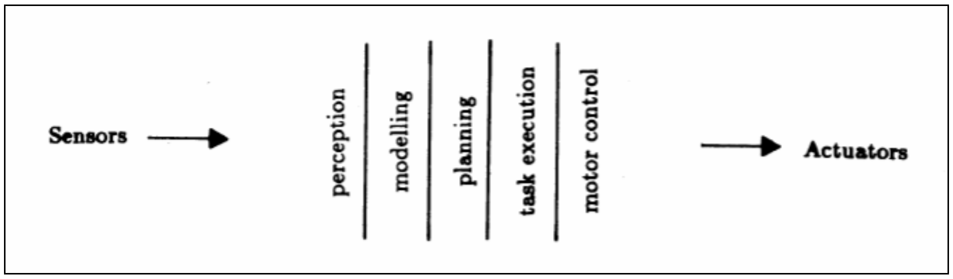


Figure 2: Traditional Architecture: sense-model-plan-act (SMPA) [4]

Behavior-Based Robotics

In the late 1980's, Rodney Brooks at the MIT AI Lab introduced the concept of behavior-based robotics [4]. In this new paradigm, basic behaviors, which are motor reactions to sensory stimuli, are the building blocks of more complex behaviors. He introduced ***subsumption architecture*** which embodied this idea. He rejected the idea of a central planner that has collective knowledge of the system. A central planner introduces a *representation bottleneck* which blocks real-time response. He argues that the SMPA paradigm is detrimental to the construction of real, working robots. A postulate of Brooks' philosophy is that "the world is its own best model" [10]. Given an appropriate model, an SMPA system can solve complicated problems within a world that strictly adheres to the model. However, Brooks noted that storing models of the world is dangerous in dynamic, unpredictable environments because such representations can become incorrect or outdated.

Brooks believes that robots must be fundamentally reactive. In the subsumption architecture, the behavior-based approach entails the horizontal decomposition of planning into a collection of concurrent layers; each connected to its own sensory inputs. A set of behaviors defines the control system. Behaviors are implemented as real-time processes that take inputs from sensors or other behaviors and send output commands to effectors or other behaviors. The controller, then, is a distributed network of concurrently executing behaviors. The interaction of the behaviors with the environment results in the desired overall system performance. Figure 3 shows a parallel behavior-based approach to robot control. This approach has produced some of the best autonomous functions to date, for example, the behaviors of Ghengis [11].

Brooks and others have argued that true intelligence in a robot will be an emergent property of a behavior based system rather than the direct result of the clever programming of an SMPA [10-12]. This statement is a working hypothesis that is supported by research in cognitive science [13]. The control systems of behavior-based robots are more closely akin to the structure and function of animal systems than are SMPA architectures.

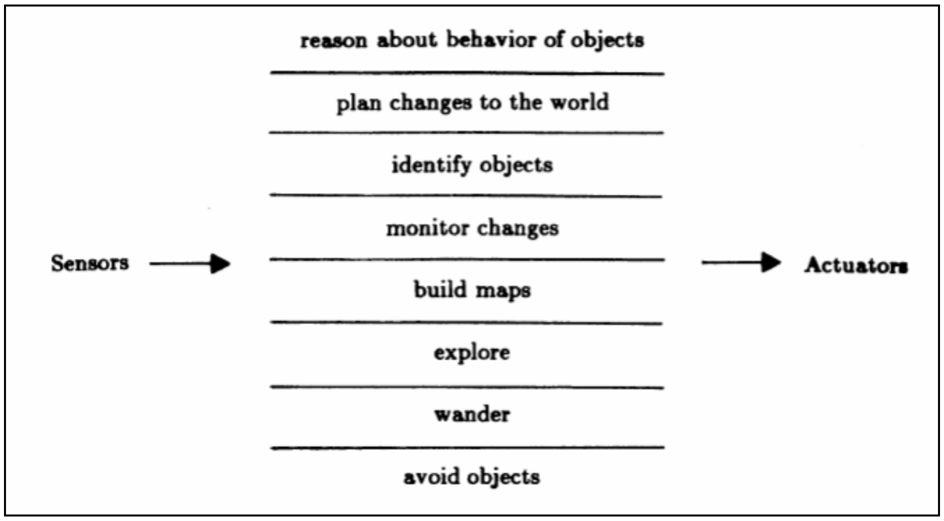


Figure 3: Behavior-based controller [4]

Brooks' Subsumption Architecture

Brooks' subsumption architecture controls a robot through a collection of augmented finite state machines (AFSM) organized into layers [4, 11]. Each AFSM, as shown in Figure 4, can be activated by sensory inputs and can produce outputs that drive actuators or are passed to the inputs of other modules. These modules do not create and pass on explicit detail. Instead, each layer is itself a complete route from input to action. The communication between layers is restricted to some signal passing. One layer can encourage, interrupt, or override the activity of another.

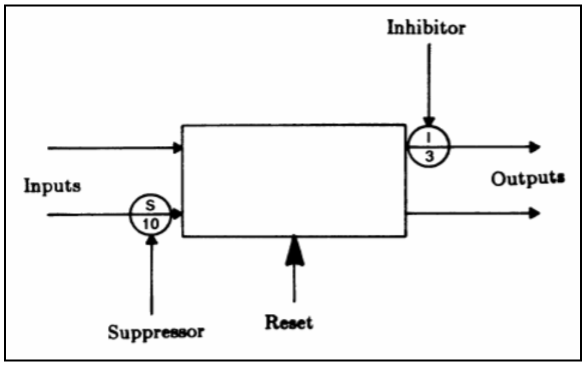


Figure 4: AFSM used within the Subsumption Architecture [4]

Figure 5 shows a simple mobile robot with three behavioral layers [4, 11]. The lowest layer is composed of simple behaviors for avoiding obstacles. The middle layer, the explore layer, permits the robot to move in the absence of obstacles and to cover large areas. The highest layer, "back-out-of-tight-situation", is used to help the robot remove itself from restrictive situations where simple avoidance and exploration behaviors fail to free the robot.

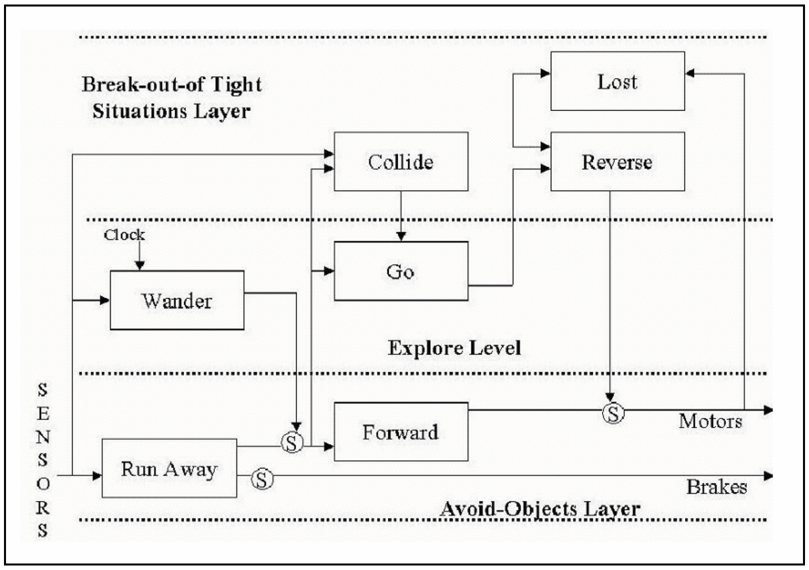


Figure 5: Subsumption Architecture used in Ghengis [4, 11]

Brooks showed the application of this architecture in several mobile robots [12], among them, Genghis [4, 11]. Arkin evaluated the subsumption architecture in Section 4.3.5 of [14]. He noted that subsumption has the following strengths:

- Hardware retargetability: Subsumption can compile down directly onto programmable-array logic circuitry.
- Support for parallelism: Each behavioral layer can run independently and asynchronously.
- Niche targetability: Custom behaviors can be created for specific task-environment pairs.

It also has the following weaknesses:

- Run time flexibility: The priority-based coordination mechanism, the *ad hoc* flavor of behavior generation, and the architecture's hard-wired aspects limit the ways the system can be adapted during execution.
- Support for modularity: Since upper layers interfere with lower ones, layers cannot be designed completely independently [15]. Also behaviors cannot always be prioritized, leading to artificial arbitration schemes [15].

Kirsh likewise pointed out some limitations on subsumption [16]. He stated that since concept and representation are abandoned in subsumption, it can only achieve limited intelligence. Concepts are either necessary for certain types of perception, learning, and control, or they make those processes computationally simpler. Once a creature has concepts, its capacities are vastly multiplied. This issue will be further discussed in the last section of this chapter, ***From Behavior-Based Robotics to Cognitive Robotics***. The next section discusses another behavior-based robotics architecture, "schema".

Arkin's Schema Architecture

Neurophysiological schema theory emerged early in the twentieth century and has been used in many different fields. Among them, Arbib used schema theory as a framework for the rigorous analysis of behavior [13]. In his definition, a schema is an adaptive controller that uses an identification procedure to update its representation of the object being controlled [17]. Arbib was also the first to consider the applications of schema theory to robotics systems [17].

One advantage of schemas in neurophysiology is that they requires no prior commitment to a hypothesis on the physical location of each schema but can be linked to a neural structure as and when it becomes appropriate [13]. Arbib described two types of schemas, perceptual schemas and motor schemas. Perceptual schemas are used for sensory analysis. Motor schemas provide the control systems that can be coordinated to effect a wide variety of movements. Schema instances may be combined to form schema assemblages to generate complex behaviors. Arbib used the term coordinated control program [17] for a schema assemblage that processes input through perceptual schemas and delivers its output via motor schemas. The assemblage interweaves the activations of these schemas in accordance with the current task and sensory environment to mediate more complex behaviors. Schema theory provides a distributed model of computation. It supports many concurrent activities for object recognition and planning or control of different activities.

In the late 1980's Ronald Arkin introduced a behavior-based robotics architecture based on Arbib's schema theory [14, 18], and called it the "schema architecture". In this architecture, as in schema theory, two types of schemas are defined, perceptual schemas and motor schemas. Unlike Arbib's schema theory, in which a perceptual schema is an independent unit, a perceptual schema in Arkin's architecture is embedded within each motor schema. These perceptual schemas provide environmental information specific to that particular behavior. Unlike Arbib's schema theory, in which an activation signal selects a motor schema to be activated, each motor schema in Arkin's architecture outputs an action vector (consisting of both orientation and magnitude components) that defines the way the robot should move in response to the perceived stimuli, as show in Figure 6. The end result is the vector sum of all the motor schema outputs.

A robot controlled by Arkin's motor schema follows gradients in a vector field map of its environment. Figure 7 shows how different motor schemas are combined to generate desired final behaviors on two-dimensional planes. This approach has been used for ground-based navigation where each vector is two-dimensional, for generating three-dimensional vectors for flying or underwater navigation, and for use in mobile manipulators with many additional degrees of freedom [14].

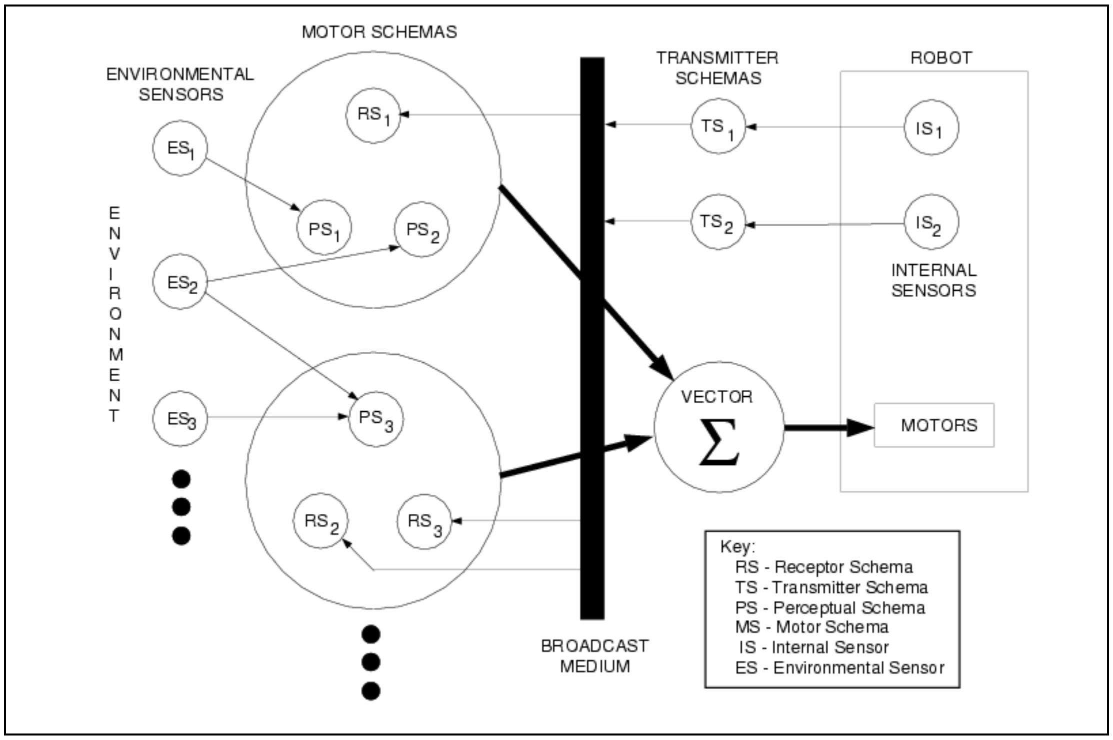


Figure 6: Perception-action schema relationships [14]

Arkin himself gave an evaluation of the schema architecture in Section 4.4.5 of [14]. It has the following strengths:

- Support for parallelism: Schema theory is a distributed theory of computation involving multiple parallel processes. Motor schemas are naturally parallelizable.
- Run time flexibility: Since schemas are software agents, it is simple to reconfigure the behavior control system at any time.
- Timeliness for development and support for modularity: Schemas are essentially software objects and are modular by definition.

It also has the following weaknesses:

- Niche targetability: the generic modular nature of the primitive schemas discourages the design of very narrowly focused components.
- Hardware retargetability: Schema-based systems do not provide the hardware compiler as the subsumption architecture does.

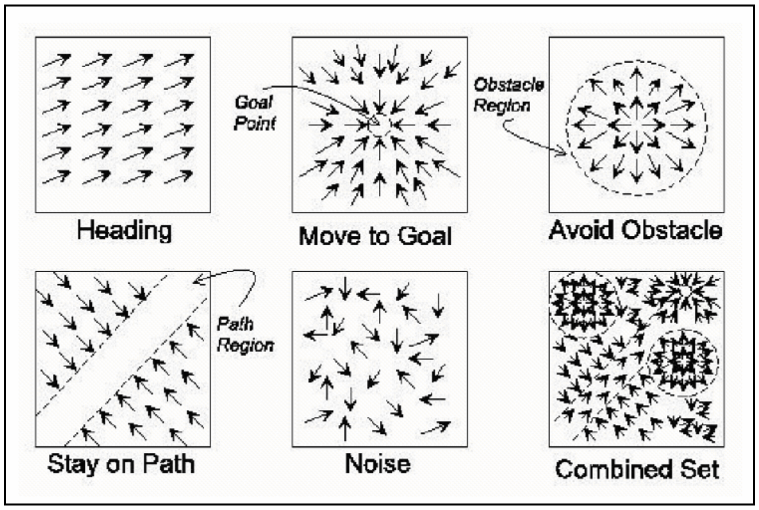


Figure 7: Motor schema [19]

Kirsh's criticism of Brook's subsumption architecture also applies here. Schema can only achieve limited intelligence because concept/representation and deliberation/planning are omitted. This omission has led the second shift in robotics, from behavior-based robotics to cognitive robotics, which is the topic of next section.

From Behavior-Based Robotics to Cognitive Robotics

Previous sections have discussed the shift from traditional to behavior-based robotics, which happened in the mid and late 1980's, and was seen mainly in mobile robotics research. A new shift, from behavior-based to cognitive robotics, started around the mid 1990's. This shift has been gradual and continues. In Rodney Brooks' words, this second shift "is necessary if we are to build robot systems that have behavior that is of similar complexity to that of humans" [20].

As Kirsh pointed out, since concept and representation are abandoned in subsumption, it can only achieve limited intelligence. In order to overcome this, many architectures proposed in the early and mid 1990's tried to unite the traditional AI approach and behavior-based approach in the same architecture. Among them were hybrid deliberative/reactive three-layer architectures [21]. These consist of three layers: a controller, a sequencer, and a deliberator. The Controller is the bottom layer. It implements one or more feedback control loops, tightly coupling sensing to actuation. The sequencer, the middle layer, selects which primitive behavior the controller should use at any given time, and supplies parameters for that behavior. The deliberator is the top layer and is the locus of the time-consuming computations, such as planning. One example of a three-layer architecture is AuRA, proposed by Arkin. It combines motor schemas with a traditional AI spatial planner. The planner configures the reactive control system prior to execution and reconfigures it in the event of task failure [14, 22].

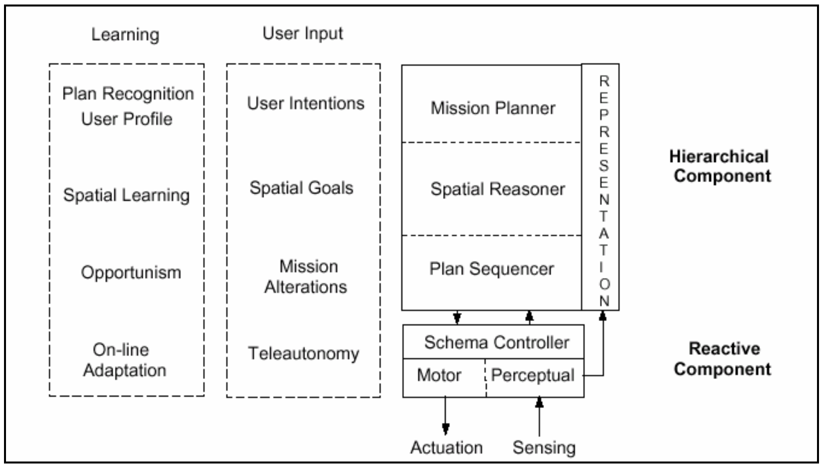


Figure 8: High-level AuRA Schematic [22]

Brooks argued in the late 1990's that to build a full human-level intelligence that is able to operate and to interact in the world requires an architecture decomposition that is different from both traditional AI approaches and behavior-based approaches [20]. A hybrid architecture is simply not enough. This new architecture decomposition needs to solve problems at different levels. First, to act like a human an artificial creature needs a rich set of abilities in gaining sensor information and a diverse set of motor actions. Besides this significant behavioral repertoire there are also a number of key issues that have to be considered in building a humanoid robot [20]:

- Bodily form

There are two reasons one might build a robot with humanoid form. First, the form of human bodies is critical to the representations that humans develop for both internal thoughts and language. Second, a humanoid form would be both easy and natural for humans to interact with in a human-like way.

- Motivation

A full humanoid robot would be confronted with many choices of what or whom to interact with, and how that interaction should take place. The system needs to have some sort of motivations, and these motivations must enable the humanoid to act properly.

- Coherence

A humanoid robot has many different subsystems, low level reflexes, and behavioral patterns. How all these should be orchestrated is a central problem.

- Self adaptation

The system must continuously adapt itself to cope with the changes in the world. The challenge is to identify the appropriate signals that can be extracted from the environment in order to have this adaptation happen seamlessly behind the scenes.

- Development

While in principle it might be possible to build a fully-formed intelligence, it does not seem to be practical. Another approach is to build a baby-like level of intelligence, and then to recapitulate human development to gain adult human-like understanding of the world and self. This is a completely new challenge for robotics.

- Historical contingencies

To understand something of the human system to build something that has human-like behaviors, one should consider what is essential and what is accidental in the human system.

- Inspiration from the brain

Since the human brain is the seat of intelligence, one might consider approaches that have functional decompositions based on what is known about the brain and try to apply that structure directly to the design of the processing system for a humanoid robot. However, the brain does not have the modularity that a carefully designed system might have. Thus one should be skeptical of these approaches.

Rod Brooks initiated the Cog humanoid robot [20, 23] in 1993 to foster a shift from behavior-based robotics toward cognitive robotics. Cog has been constantly evolving since then and other humanoid robots have been built in the MIT AI Lab. Figure 9 shows the basic control architecture for these robots. This is a complex architecture [23], consisting of six subsystems.

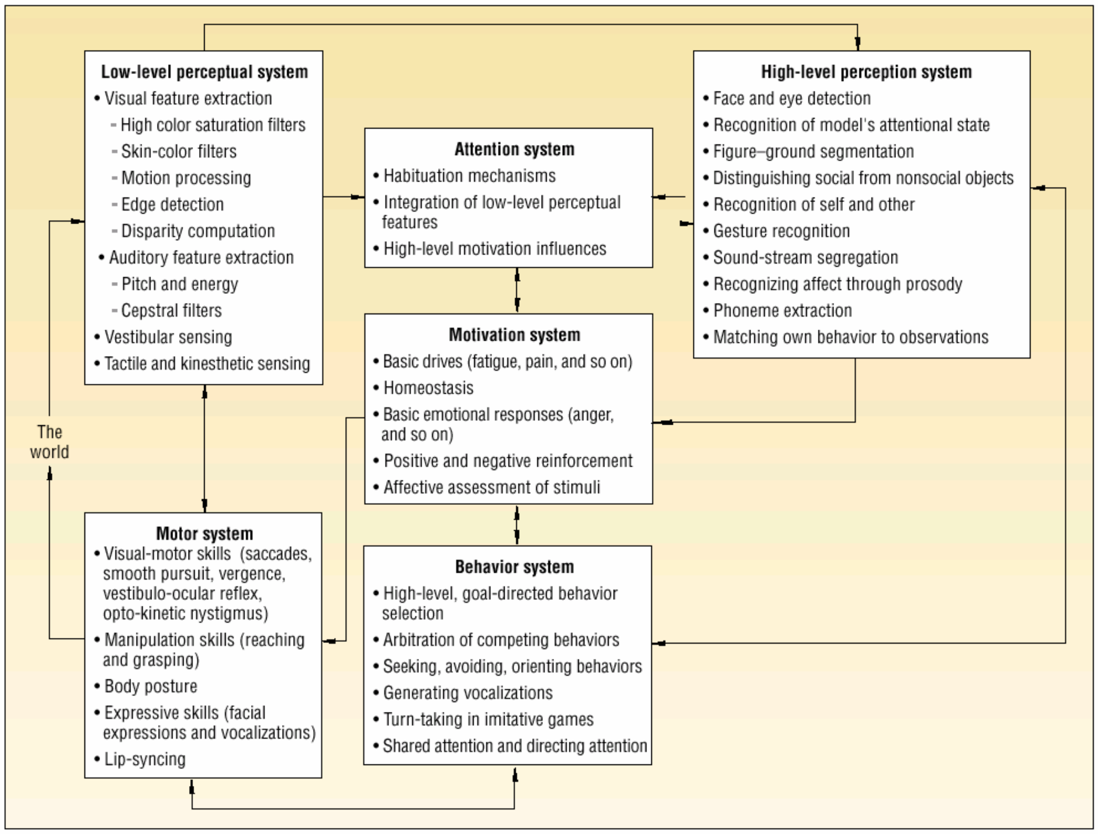

Low-level perceptual system
• Visual feature extraction
- High color saturation filters
- Skin-color filters
- Motion processing
- Edge detection
- Disparity computation
• Auditory feature extraction
- Pitch and energy
- Cepstral filters
• Vestibular sensing
• Tactile and kinesthetic sensing
Attention system
• Habituation mechanisms
• Integration of low-level perceptual features
• High-level motivation influences
High-level perception system
• Face and eye detection
• Recognition of model's attentional state
• Figure–ground segmentation
• Distinguishing social from nonsocial objects
• Recognition of self and other
• Gesture recognition
• Sound-stream segregation
• Recognizing affect through prosody
• Phoneme extraction
• Matching own behavior to observations
Motivation system
• Basic drives (fatigue, pain, and so on)
• Homeostasis
• Basic emotional responses (anger, and so on)
• Positive and negative reinforcement
• Affective assessment of stimuli
The world
Motor system
• Visual-motor skills (saccades, smooth pursuit, vergence, vestibulo-ocular reflex, opto-kinetic nystigmus)
• Manipulation skills (reaching and grasping)
• Body posture
• Expressive skills (facial expressions and vocalizations)
• Lip-syncing
Behavior system
• High-level, goal-directed behavior selection
• Arbitration of competing behaviors
• Seeking, avoiding, orienting behaviors
• Generating vocalizations
• Turn-taking in imitative games
• Shared attention and directing attention

Figure 9: A generic control architecture for MIT humanoid robots [23]

Rolf Pfeifer also discussed what is required for a "complete autonomous agent" [24, 25]:

- Self-sufficiency

A complete autonomous agent must maintain its energy supply without external human intervention.

- Autonomy and Situatedness

An autonomous agent must actually exhibit autonomy. Situatedness means that the agent acquires information about its environment only through its sensors in interaction with the environment. A situated agent interacts with the world on its own, without requiring human intervention.

- Embodiment

Autonomous agents must be real physical agents; in other words, they have physical bodies. Embodiment implies that the agent is continuously subjected to physical forces, to energy dissipation, to damage, and to any influence in the environment.

- Adaptivity

Adaptivity is a consequence of self-sufficiency. If an agent is to sustain itself over extended periods of time in a continuously changing, unpredictable environment, it must be adaptive.

In summary, due to (a) slow reaction and (b) sensitive to noise, SMPA architecture is only limited in well-structured environment. Behavior-based robotics rejects the idea of a central planner that has collective knowledge of the system. In pure reactive systems, intelligence emerges from carefully designed reactive behaviors. Such systems are fast and robust to noise, yet the intelligence is limited. Hybrid architectures combine the benefits of both systems, and they are the prevailing choices in robotics research nowadays. However, such architectures are usually complex. Many issues have to be taken into account during the design of such a system [14].

CHAPTER III

BEHAVIOR LEARNING

The last chapter gives current answers to the architecture problem. This chapter will give current answers to the problem of constructing behaviors. There are at least three approaches to construct behaviors: hard-wired, modeling biological agent, and learning. In hard-wired (pre-programmed) approach, a designer chooses the basic behaviors and defines the behavior's actions, starting and stopping conditions. In effect, the designer must anticipate which sensory events should precede which action. This is feasible only in simple, reactive situation. In modeling-biological-agent approach, a designer models as closely as possible the known aspects of the control strategies used by biological agents. This can be used to verify models of animal cognition. However, such model does not always exit. Even when such model exits, usually it is too complicated for current hardware technologies. Learning approach does not require direct programming and adapts well to the environment changes, thus, it is the preferred method. This chapter will discuss what is robot behavior learning and what to learn for a robot.

What is Robot Behavior Learning

Learning is often viewed as an essential part of an intelligent system, yet there is no universal definition. Webster's Dictionary defines it as "modification of a behavioral tendency by experience". Ronald Arkin gives an operational definition: learning produces changes within an agent that over time enable it to perform more effectively within its environment [14]. Tom Mitchell defines machine learning as the study of computer algorithms that improve automatically through experience [26]. In the robotics field, the robot behavior learning problem is to design a robot so that it improves its performance through experience [1]. The key here is "improve performance through experience". To be precise, we must specify what performance and experience are.

Looking from the state space point of view, a behavior changes a robot from one state to another state. First, the robot is in an initial state with an initial stimulus. After a corresponding behavior is carried out, the robot is in a new state. This new state has a new stimulus, and a new behavior should be carried out. This process continues until either the robot reaches the final goal or being interrupted. A robot's experience and performance can be formally defined in this state space with rewards and/or penalties assigned to states and state transitions. Nehmzow and Mitchell have elaborated this in [1].

What to Learn

For a robot to behave autonomously, it needs to learn many different behaviors to cope with the environment at different scales. These behaviors cannot be learned in an arbitrary order. Instead, low-level behaviors should be learned before high-level behaviors. This is the development issue discussed in last chapter. A curriculum is therefore required. This curriculum covers two aspects of learning [1]:

- An agenda of what is to be learned,
- The order in which it is to be learned.

Generally speaking, at the bottom level of the learning curriculum, there is the learning of reflexive sensor-motor associations, expressed in competencies such as obstacle avoidance, wall following, and following attractors. Competencies at this level are state of the art in mobile robot research [27] [28]. At the next level, there is learning of how to combine several fundamental competencies to achieve more complicated tasks, how to identify components of plans and how to assemble them to form complete plans.. Nehmzow and Mitchell give a more detailed curriculum of robot learning [1]:

- Reflexive actions
- Attentive competences
    - Detection of salient features of the environment
    - Detection of correlation in perceptual patterns over time
    - Ability to focus attention
    - Differentiation between relevant and irrelevant events with respect to the current goals
- Identification of sequences of actions as plan components
- Planning capabilities
    - The assembly of plan components to plans
    - Planning of actions independent of their actual execution
    - Ability to modify or abandon plans
- Identification of macro-plan components, and
- Their assembly to form a complex macro plans
- Cumulative learning
- Forming abstract concepts
- Integration of several of previous levels into one system.

This dissertation will only survey current approaches in the middle level of this curriculum, namely, attentive competencies and planning capabilities. This is because, firstly, low level reflexive action learning is being researched by others and has a number of plausible solutions, and secondly, high level learning is still very primitive at its current stage. Although middle level behavior learning is less explored, solving this problem could lead to solutions at the higher level. Before presenting different learning methods in detail, a common theme in all these learning methods will be discussed. This common theme is sensory-motor coordination (SMC), and this SMC is where the current approaches differ radically from traditional AI approaches.

CHAPTER IV

SENSORY-MOTOR COORDINATION PRINCIPLE

The last chapter gives a definition of behavior learning and a robot learning curriculum. This chapter reviews literature on the self-organization of sensory-motor coordination (SMC) descriptions into categories. Any basic behavior defines a category of action in response to sensory input – a stimulus-response category. Because any agent in the real world has to be able to make distinctions between different types of objects in order to react appropriately, it must have the competence of categorization.

Pfeifer pioneered sensory-motor coordination research in robotics [6, 25]. This chapter summarizes his work in [25]. First, it reviews some of the traditional AI approaches to the problem of determining SMC. Next, it discusses why these approaches have failed in real world applications. This discussion leads to the sensory-motor coordination principle and why it is promising to solve the categorization problem.

Categorization: Traditional Approaches

Making distinctions in the real world comes very naturally to humans: we recognize objects and find our way around with great ease. However, getting machines to do the same thing has turned out to be an enormously difficult problem, one of the hardest in the study of intelligence.

For the most part, research on categorization has been studied within an information processing framework. This is manifested in both cognitive psychology and AI/computer vision. In some schools of traditional cognitive psychology (Figure 10), categorization in humans and higher animals is assumed to involve the following steps [29]. First, a structural description of the object to be categorized is formed. This description provides information about the object's primitive perceptual features, such as horizontal and vertical lines and the relations between these basic features. Next, category representations with similar structural descriptions are searches in memory, and the category representation most similar to the structural description is selected. Based on the selected representation, inferences about the object are drawn, and finally information about the categorization is stored in the memory.

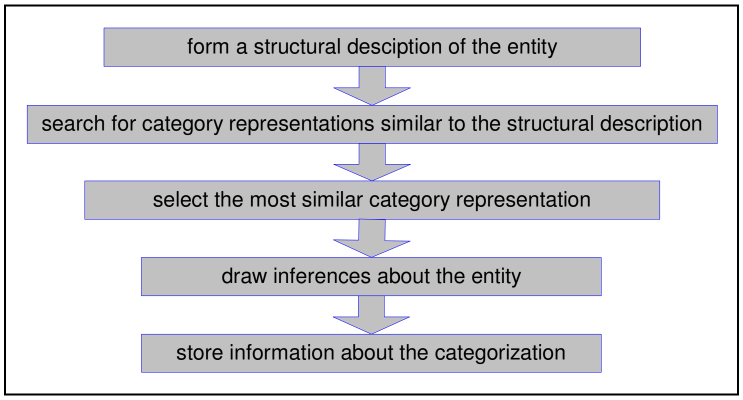

form a structural desciption of the entity
search for category representations similar to the structural description
select the most similar category representation
draw inferences about the entity
store information about the categorization

Figure 10: The information processing approach to categorization [29]

Many computer vision problems require pattern recognition, and thus categorization. The core idea in most traditional machine vision approaches to object recognition is similar to the one mentioned above: An input---an image of an object---has to be mapped onto an internal representation---in this case stored templates. Due to different lighting conditions and size variances, direct template matching usually gives unsatisfactory results and more elegant matching methods are required.

Ullman distinguished three major approaches to object recognition [30], all of which are based on the idea that in order for the agent to match an image of an object to the stored internal representation, regularities across different views of one object have to be exploited.

These approaches differ in their specific methods and assumptions of how these regularities can be extracted. These three approaches are: invariant properties based methods, parts decomposition methods, and alignment methods.

Methods based on invariant properties assume that certain observable properties remain invariant under the transformations an object is allowed to make. Thus by calculating this invariant and comparing it with the stored models, a certain object can be recognized.

Parts decomposition methods rely on the decomposition of an object into its constituent parts or generic components. For example, a face might be decomposed into the eyes, nose, and mouth, each of which is assumed to be recognizable on its own. The recognition task is to locate the part (e.g., a view of a mouth), find the corresponding generic component (e.g., mouth), and finally describe the object in terms of its constituent parts. This method is heavily influenced by traditional AI: first break a question into small parts, then use search and inference to find the solution.

Alignment methods are based on the idea of aligning an acquired image of the object with a corresponding stored model. For example, if the image of the object and the corresponding stored model are very similar except for a difference in size, then aligning the two involves scaling either image or model, thereby reducing the discrepancy and improving the match. More generally, the idea is to store not only a model of an object but also a set of "allowed transformations" that the object may undergo (e.g., changes in size, position, orientation). Object recognition then becomes a matter of searching for a particular model and a particular transformation that increase the match between the model and the image.

All these approaches have had limited success in specific situations. However, they fail quite often in real-world, real-time robotics applications. Why?.

Problems with Traditional Categorization Approaches

Categorization is a notoriously difficult problem. This is mainly due to the ***object constancy problem***, that is, the problem of determining what parts of the input imagery belong to one and the same object. This problem is hard because the same object can lead to a very large number of different input patterns depending on the viewing angle relative to the object, the lighting conditions, noise in the sensors and so forth. Why the above mentioned three computer vision approaches fail is a manifestation of this problem:

- The invariant properties method was inspired by the famous Hu's moment invariant in 2-dimensional objects [31]. However, looking for such an invariant in 3-dimensional objects has been unfruitful, and it is now generally considered that such an invariant does not exits in 3D cases.
- The part decomposition method is based on classical AI approaches and hence suffers from the same classical AI problems: intolerant to changes and noise, while changes and noise are abundant and inevitable in real-world. Also this method usually fails abruptly rather than gracefully in real applications.
- The alignment method is also based on classical AI approaches and suffers the same set of problems.

Yet there is a common theme in all these methods: the recognition problem is solved by means of internal processing only. No system-environment interaction involved. In Wechsler's words, "recognition is the ultimate goal of any visual system" ([32] p. 303). Categorization is a perception ability, and perception in classical AI has also omitted the system-environment interaction. Arkin pointed out significant problems with this approach [14]:

- Perception considered in isolation: Perception is not a disembodied process; it is a synergistic process deeply involved with the complete agent's cognitive and locomotion systems.
- Perception as king: Perception needs to be viewed as only one of the many requisite needs for a functioning intelligent agent.
- The universal reconstruction: Much perceptual research has focused on creating three-dimensional world models. These models are often built without regard for the robot's needs. A deeper question is whether these reconstructive models are really needed at all.

Active perception is the first step to involving motor behaviors into perception. Pahlavan et al defined an active visual system as " a system which is able to manipulate its visual parameters in a controlled manner in order to extract useful data about the scene in time and space" [33]. Bajcsy's [34] seminal paper on active perception characterizes it as the application of intelligent control to perception using feedback from both low-level sensory processes and high-level complex features. Active perception focuses primarily on the needs of perception, rather than the needs of action. Also, in most active perception systems, only camera head control is utilized for better perception. The agent's ability to move around and to manipulate objects is still ignored.

From an evolutionary perspective, the ultimate goal of vision is not recognition *per se*, but rather to enable an agent to behave efficiently in the real world. When looking for mechanisms of "categorization", we have to take the frame-of-reference problem into account. The description of a behavior in terms of categories that an agent apparent "has" does not imply that the mechanism behind this behavior actually employs explicit category representations, such as descriptive graphs or prototype models. Categorization should be studied in the context of system-environment interaction. In an autonomous agent this interaction is grounded in the sensory-motor coordination.

Categorization via Sensory-Motor Coordination

The idea of sensory-motor coordination can be traced back to John Dewey, an American philosopher. He wrote, "We begin not with a sensory stimulus, but with a sensorimotor co-ordination…. In a certain sense it is the movement which is primary, and sensation which is secondary, the movement of the body, head, and eye muscles determining the quality of what is experienced" ([35] pp. 127-128). Dewey implies that perception and action are tightly coupled, and he called this coupling "sensory-motor coordination". J. J. Gibson holds the same idea in his direct perception theory. According to his view, perception is not the result of a passive module that processes information; rather, "Perceiving is an act, not a response, an act of attention, not a triggered impression, an achievement, not a reflex" ([36] pp. 68). Pfeifer has demonstrated that sensory-motor coordination in robotics leads to categorization [6, 25] and his work will be summarized in the next chapter.

Sensory-motor coordination does not mean simply "behavior". For example, a robot that is turning about its own body axis is not necessarily engaged in a sensory-motor coordination if the sensory patterns generated by the behavior are not necessary to guide its behavior. Sensory-motor coordination involves object-related actions that effectively structure the sensory space for the learning about an object. This is an active process whereby the agent's motions constrain its sensory input. The association of characteristic sensory signals with specific motions in a specific context leads to categorization. The principle has two main aspects.

- The analysis and the design of behaviors are best understood in terms of how sensory and motor systems are coordinated.
- Through physical action, an embodied agent can structure its own sensory input and thereby induce regularities in the sensory signals that significantly simplify learning.

These two aspects are more evident in the survey presented in the next chapter. In summary, by interacting physically with an object while sensing and response, an agent generates spatio-temporal correlations in the sensor space. This constraints the number of possible sensor states, thus eases the constancy problem. An agent can categorize the environment by exploring it in such a way that sensor readings are correlated with each other and with the exploratory actions.

Sensory-motor coordination forms a basis for categorization because parametric descriptors of SMC self-organize. Self-organization was first studied in thermodynamic systems. In a thermodynamically open system, one in which there is energy throughput, order can emerge from chaos in a phenomenon know as self-organization [37]. Self-organization appears in biological and social systems, and emerges in human and animal perception and action. In SMC, patterns emerge from seemingly incoherent sensory-motor interaction, form categories of sensory coupled actions.

CHAPTER V

SALIENT FEATURE DETETECTION AND BEHAVIOR LEARNING USING SMC

The last chapter discussed some traditional approaches to categorization and their limitations in real-world applications. The sensory-motor coordination principle was then introduced. This chapter surveys current approaches to salient feature detection that depend on this principle, and how SMC can facilitate middle level behavior learning. The first section demonstrates how to use SMC can solve the categorization problem and contrasts it to traditional approaches. The second section presents imitation-based learning which uses imitation to construct sensory-motor couplings. Many approaches are discussed here, both on humanoid and on mobile robots. The third section presents episode-based learning and how it can construct sensory-motor couplings to form concepts. There are critiques of each approach at the end of each section. The last section summarizes the whole chapter.

Categorization: SMC approaches

Categorization: Pfeifer Group's Approach

Pfeifer and his group built two robots, SMC I and SMC II, and showed in their experiments that the SMC approach can indeed simplify the categorization problem.

SMC I: Basic Categorization

The SMC I agent was developed to investigate basic categorization behavior in the real world in a first attempt to study how autonomous agents can make sense of an unlabeled world on their own [6, 25, 38]. A miniature robot, Khepera, was the test platform. It has small wheels, 8 IR-sensors, ambient light sensors, and a gripper with two degrees of freedom, shown in Figure 11. The goal of the agent is to bring the small objects to location A (with a light source), while leave the large ones where they are, as shown in Figure 12.

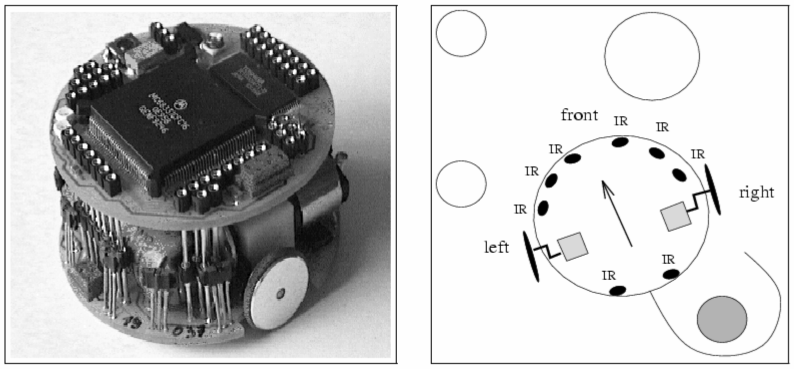


Figure 11: Khepera robot [38]

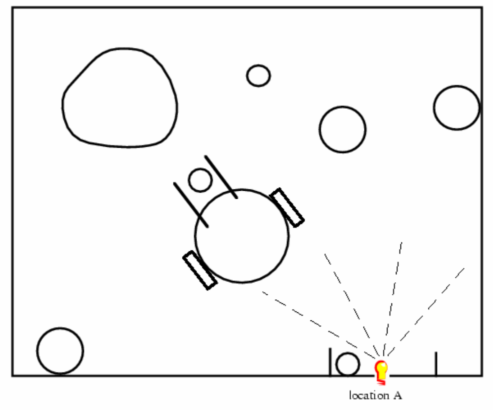


Figure 12: The ecological niche of the SMC I agent

The robot has to categorize different size objects in order to pursue its goal. It has been shown that this categorization is not possible using only the IR sensors on Khepera [39]. However, Pfeifer proved that by using SMC, this categorization problem can be solved. In his experiment, the robot control architecture is EBA, Extended Braitenberg Architecture [40], shown in Figure 13. This architecture is similar to Arkin's motor schema. Eight EBA processes (similar to Arkin's motor schemas) are used: move-forward, avoid-obstacle, turn-towards-object, sense object, grasp-object, push-object, turn-away, and phototaxis (another name is bring-to-nest).

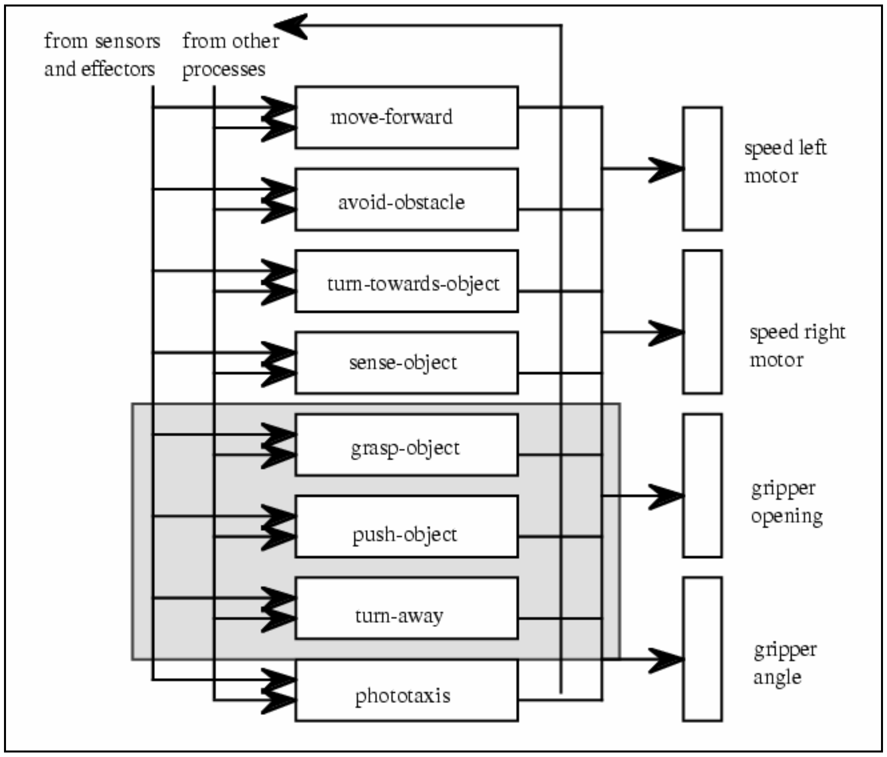


Figure 13: Architecture of SMC I agent [6]

The Move-forward behavior, turn-towards-object behavior, and avoid-obstacle behavior combined generate a circling behavior, i.e., the robot circles around an object. This circling behavior induces high spatiotemporal correlations in the sensor patterns. Figure 14 shows the correlation between subsequent input vectors as the agent approaches a large object. Vectors are 10-dimensional: eight IR sensors and two motor speeds. Theoretically, when the agent moves in the open, the correlation should have a value of 1 because the IR sensors have a limited range, so that there should be no stimulation. Noise, however, makes the correlation drops to roughly 0.5. As the agent approaches an object, the correlation drops, because sensory activation is changing rapidly. Once the agent is near the object, the dynamics of the processes come into play and there is a time-locked activity in the sensory-motor space, and the correlation rapidly jumps to the maximum. Note that these correlations are induced by the agent's own movements, or in other words, by means of sensory-motor coordination.

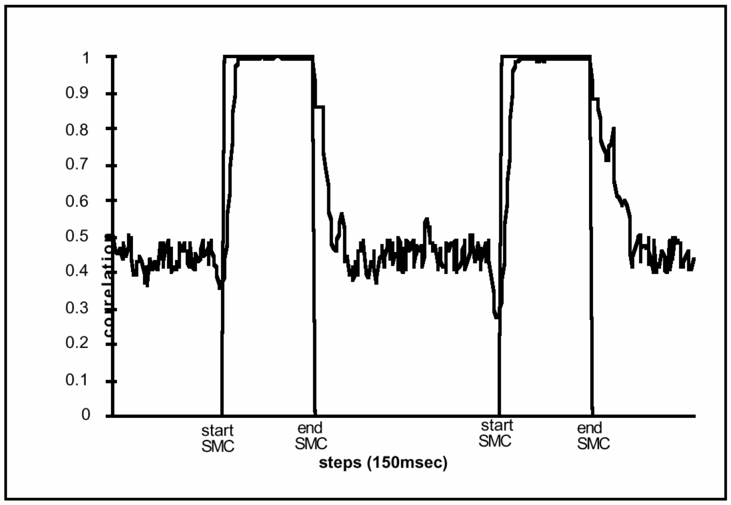


Figure 14: Change in correlations between subsequent input vectors [6]

Now a learning mechanism is required to learn these sensory-motor coordinations. A Kohonen map [41] is used for this purpose, shown in Figure 15. This Kohonen map receives input from the wheel encoders---measuring the speed of the motors---and the IR sensors. The Kohonen map is connected to the grasp, push, and avoid networks, which in turn write their outputs to the motors and the gripper. The gripper sensors provide input to the value system, which modulates the weight change in the connections between the Kohonen map and other processes. At first, the robot behaviors toward small and large objects do not differ. But after several trials, the SMC I agent can associate different behaviors to different sizes of objects as in Figure 16.

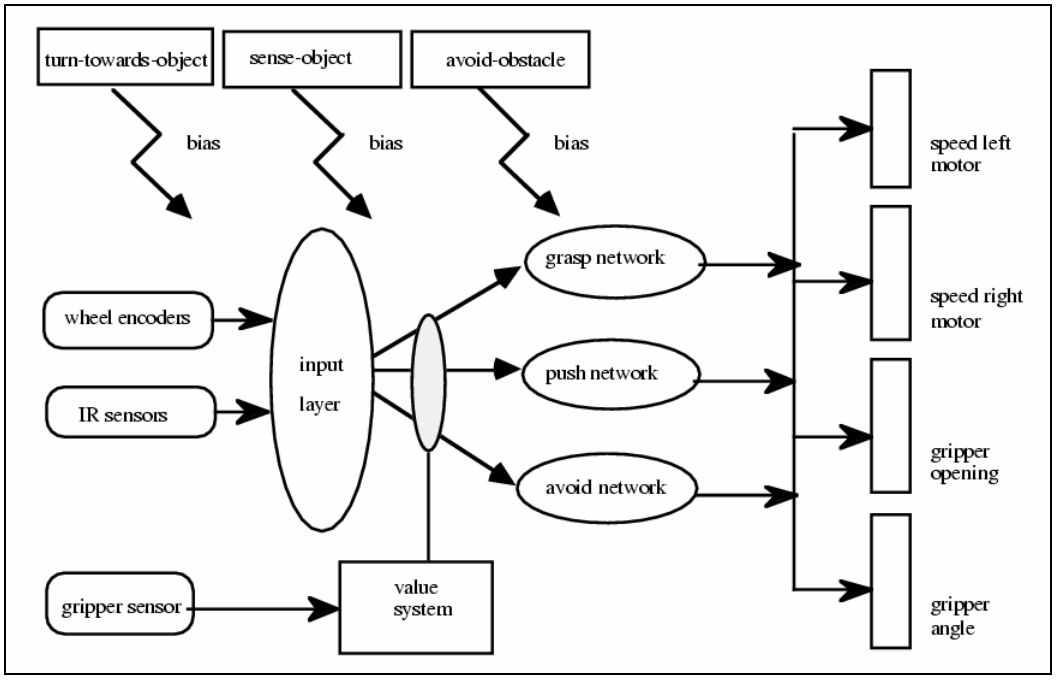


Figure 15: Learning in SMC I Agent [6]

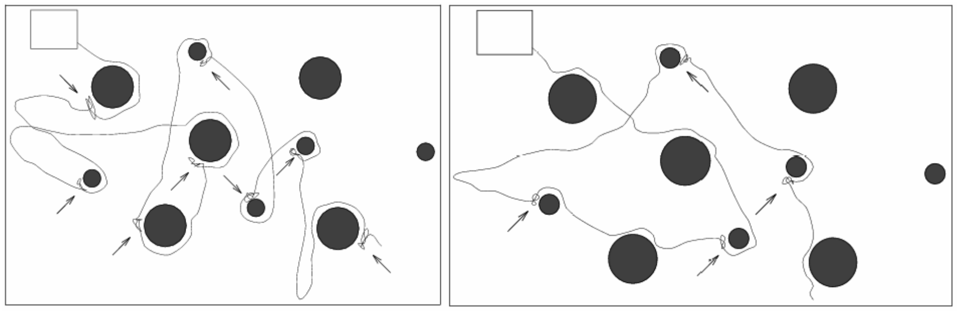


Figure 16: A typical trajectory of the robot before learning and after it has encountered a large number of pegs [38]

SMC II: Increasing the Complexity

SMC II is an extension of the SMC I Agent [42, 43]. SMC II was again implemented on Khepera. Besides IR sensors, a CCD camera is now added. Instead of passively scanning the camera image, an artificial eye has been mounted, which actively moves a fovea to interesting parts (e.g. bright spots, texture, movement) of the image. This foveation process is an instance of a sensory-motor coordination. In addition, the control of the arm-gripper system is considerably improved.

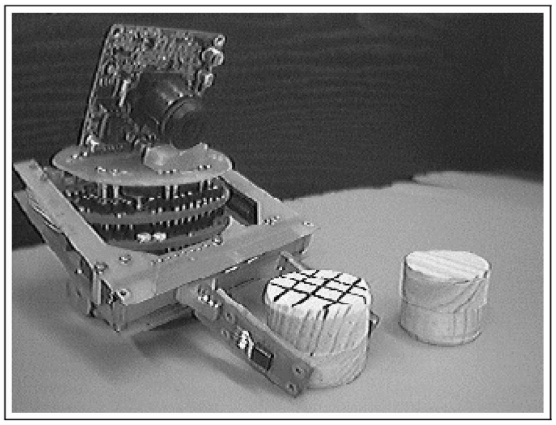


Figure 17: SMC II agent [42]

As before, the task of the robot is to collect objects and to bring them to a home base. The objects were cylindrical and all of the same size. They were either conductive or non-conductive. The conductive objects had a strongly textured surface while the non-conductive ones had a white or only slightly textured surface. The robot's task was to collect the conductive objects. The control architecture of SMC II is again based on the EBA, shown in Figure 18.

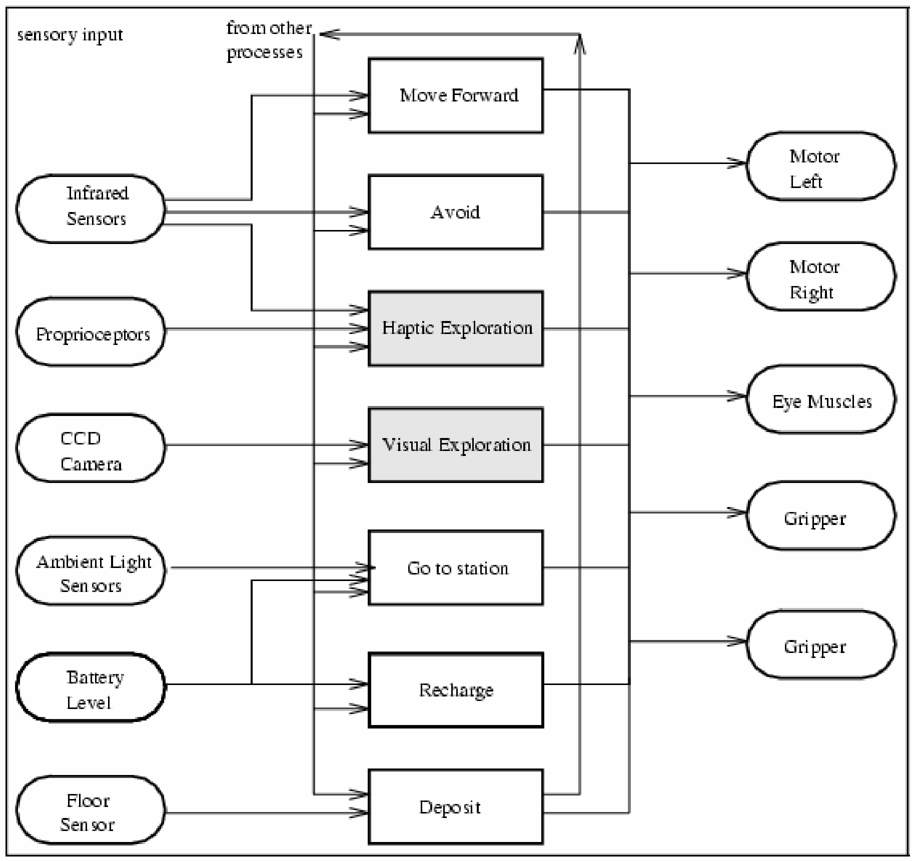


Figure 18: SMC II control architecture [43]

The shaded processes in the above figure, haptic and visual exploration, replaced the various processes in SMC I that implemented object-related behavior. The agent can interact with its environment in two ways. First, there are attentional sensory-motor loops which are directly coupled to the arm, gripper and wheel motors (i.e. the effectors of the robot). As a result the agent moves its body and the arm into a position from which it can explore the object. Second, both haptic and visual data which result from the exploration process are used for learning and categorizing objects. As shown in Figure 19 , in both systems there are sensory maps that are connected to feature maps. Feature maps respond to properties of objects such as texture (visual) or conductivity (haptic). The interaction between the two modalities is implemented via modifiable weights between these feature maps. The correlation of signals of the haptic and the visual feature maps by these reentrant connections forms the basic mechanism of categorization. These visual and haptic feature maps are connected via modifiable feedback connections to the corresponding visual and haptic attention maps. The main idea here is to link the correlated activity in the feature maps to the attentional sensory-motor loop. Value signals modulate the learning process, just as in SMC I. The value map receives input from the conductivity sensor and the gripper proprioceptors. The basic motivation behind these connections is that the robot should learn only when it explores an object. Activity in the value map acts like a gating function and is used as a reinforcement signal for the synaptic modifications between the two feature maps. Figure 20 shows a typical trajectory of the robot initially and after it had encountered 10 objects of each type. Obviously the agent learned how to differentiate these two types of objects.

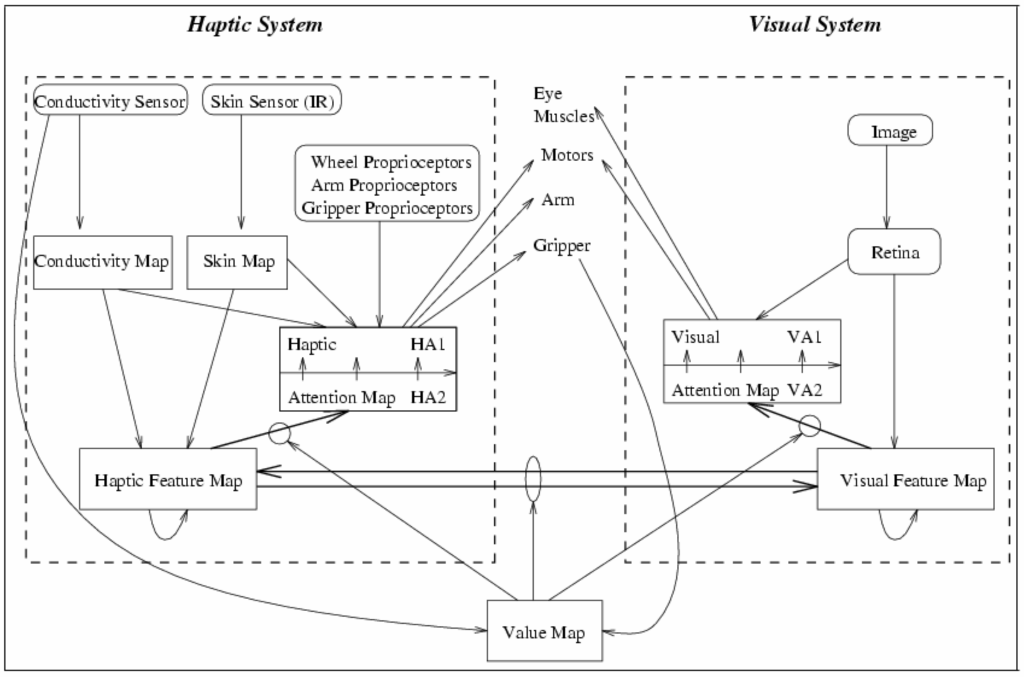


Figure 19: Overview of the visual and the haptic exploration systems in SMC II [43]

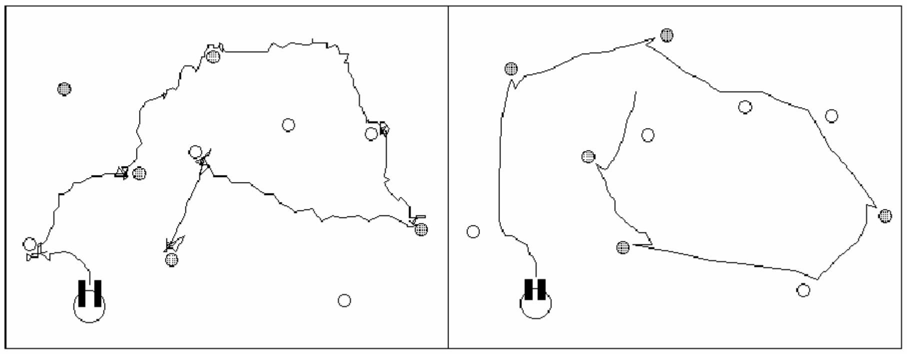


Figure 20: A typical trajectory of the robot initially (left) and after it had encountered 10 objects of each type (right) [43]

Categorization: Grupen Group's Approach

Roderic Grupen of University of Massachusetts Amherst and his group designed a humanoid torso, Magilla [44, 45], as shown in Figure 21. It consists of two Whole Arm Manipulators, two multi-fingered Barrett hands, and a TRC BiSight stereo head.

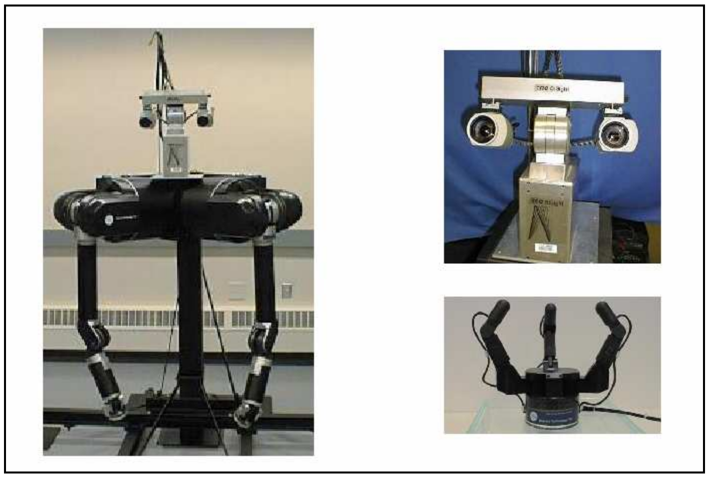


Figure 21: Magilla, the UMASS humanoid torso [44]

Figure 22 is a sketch of a computational framework that addresses the development of manual skills with robot hands. One dimension of development is viewed as a scheduling problem in which robotic resources are engaged to satisfy a task. Primitive actions are closed-loop control processes constructed by combining an artificial potential - the reward - with a subset of the available sensors and effectors. As these controllers interact with objects and tasks, a set of prototypical dynamic models are constructed that identify haptic categories during grasp formation. Incremental learning (i.e., development), in which important control knowledge is accumulated over time, was used to learn the optimal grasp strategies. This is summarized in their conjecture 1 [44]:

Conjecture 1 (Reflexive Basis for Motor Development). Reflexes are scheduled in a manner consistent with other development mechanisms, in a sequence that leads an agent through a progression of incremental and environmentally-mediated learning tasks. These tasks acquire critical knowledge structures in an appropriate order.

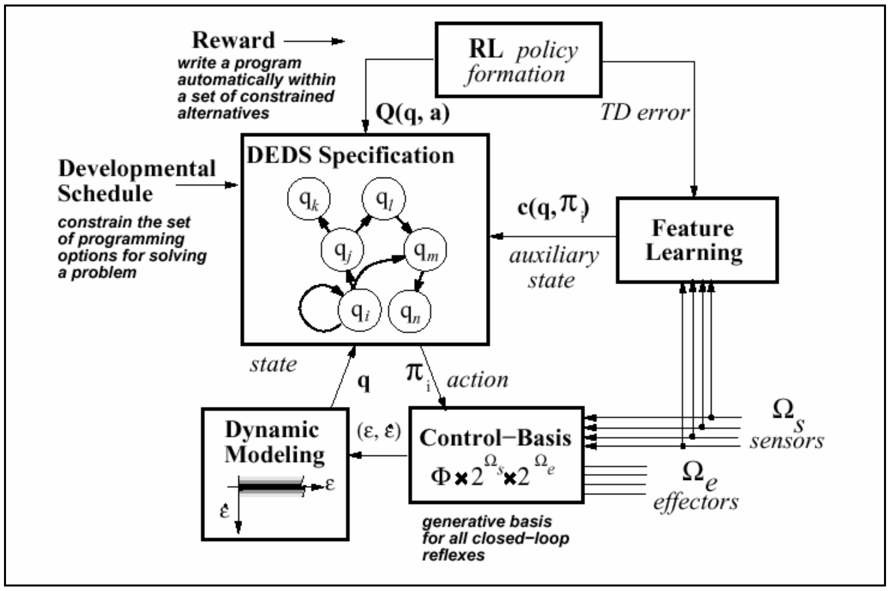


Figure 22: Native structure, learning, and development in an integrated architecture [44]

Once the utility of haptic categories and grasping policies are learned and compiled into value functions, these mature haptic value functions can be used as the basis for visual discrimination tasks. This is summarized in their conjecture 2 [44, 45]:

Conjecture 2 (*Haptic-then-Visual Development*). A haptic subsystem is employed first to discover useful grasping strategies at the expense of perceptual acuity and efficiency. Having identified such haptic models they form the basis for the acquisition of high-precision, efficient visual operators that recover important control contexts. This results in a powerful associative multi-modal model of interaction with objects in which haptic experience can be predicted by visual features and vice versa.

The objective here is to develop a plausible scheme for learning visual features that robustly correlate with the orientation of the hand during a successful grasp. Then, these features can be used to recommend a hand orientation and a native grasp control strategy that should be engaged before the first tactile contact occurs, bootstrapping the haptically-driven grasp and eliminating the need for expensive and inefficient haptic context recognition. Each type of object may require a dedicated visual feature to fully capture the haptic context. Object identities are not known to the system, so the need for dedicated features must be discovered by grasping experience. Visual learning is entirely driven by the utility of the features to the haptic system.

Local appearance-based features are employed to represent visual context. Two types of primitive features are used: texel and edgel. A *texel* is a vector consisting of filter responses from Gaussian-derivative operators of the first three orders; An *edgel* uses an orthogonal pair of first-order derivatives only. Spatial combination of these primitives can express a wide variety of shape and texture characteristics at various degrees of specificity. An incremental, on-line learning procedure assembles such compound features in a simple-to-complex manner, as the need for increasingly distinctive features arises [46-48]. Figure 23 illustrates a geometric arrangement of oriented primitives that has been generated by such an approach to form a useful distinction. This particular feature is composed of three primitives, defined by the angles ϕ, the distances d, and the orientations θ. Visual distinctions in this framework need not be universal because they are tagged to particular states and tasks in the behavior of the system, sometimes inexpensive combinations of features are adequate for discriminating locally between important visual contexts.

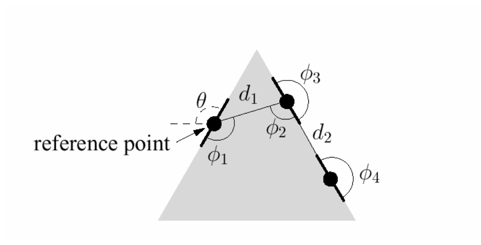


Figure 23: A geometric feature composed of three primitives [44, 45]

Features are learned as follows. A reward function for successful grasping is constructed. Given an image, the responses of the reward function for all features are measured. The best feature is selected by expectation maximization (for detail of this process, please refer to [44]), and is used to recommend a hand orientation. The robot then executes the grasp, starting with the recommended hand orientation. If the hand orientation turns out not to be appropriate, then all mixture models are re-estimated based on a case list of previous experiences. A new prediction is made based on the new models. If this new prediction is still wrong, then two new features are generated: A primitive feature is randomly sampled from the image, and a new compound feature is generated by randomly expanding an existing feature by adding a new point. If a feature performs well over time, it will increasingly be employed. If it performs poorly, it will eventually cease to be used at all. Unused features are discarded periodically.

Critiques of SMC Approach

Pfeifer and Scheier summarize the benefit of the SMC approach in [6]:

(1) It provides the basis for physical control over objects. Needless to say, having control over objects like food items or building materials has many advantages for survival and therefore has a strong functional character.

(2) A second purpose is of a perceptual nature: sensory motor coordination implies that both sensory and motor processes play an integral part in perception. From this it follows that in ontogenetic development the two should co-evolve since otherwise there could be no coordination. For robots this means that sensory and motor abilities should "match" in complexity, i.e. they should obey the principle of ecological balance [49].

(3) The third purpose is more of an information-theoretic nature: sensory-motor coordination induces correlations, thus reducing the high-dimensional sensory-motor space to a low-dimensional sub-space. These correlations are largely due to the agent's self-motion. They are essential enablers of categorization and learning, and they provide a natural focus-of-attention mechanism.

(4) It allows for the integration of several sensory modalities (e.g. visual, haptic, or auditory). This can be achieved by forming associations according to spatio-temporal correlations that exist between various sensory modalities. Thus, sensory-motor coordination has an exploratory function: new relationships can be discovered.

(5) And finally, the most obvious purpose of sensory-motor coordination is learning to master sensory-motor coordination itself, e.g. hand-eye coordination which is essential in reaching behavior and grasping behavior. This can be learned from the correlation of sensory signals produced by self-motion.

The limitation of sensory-motor coordination is that it is a low level behavior learning technique, so it cannot be used for high-level learning. It also requires an elegant and powerful learning mechanism to learn the SMC. Another drawback is that usually it takes a long time and many trials for the system to converge into a stable configuration.

Sensory-motor coordination has a close relationship with schema architecture [14]. In Arkin's schema architecture, two types of schemas are defined, perceptual schemas and motor schemas. A perceptual schema is embedded within each motor schema. These perceptual schemas provide the environment information specific for that particular behavior. So every behavior is a tight sensory-motor coupling.

Imitation-Based Learning

Imitation was succinctly defined by Thorndike in 1911 to be: ***From an act witnessed learn to do an act.*** ([50] p. 79, cited in [51]) Yando et al defined it in a psychological context as: ***Imitation is the motoric or verbal performance of specific acts or sounds that are like those previously performed by a model.*** ([52], cited in [51]) Bakker and Kuniyoshi stated it in terms of the mechanism involved: ***Imitation takes place when an agent learns a behavior from observing the execution of that behavior by a teacher*** [51].

Imitation-based learning can drastically speed up the learning process because it reduces the learning search space comparing to other learning methods. Schaal argues that imitation is the route to humanoid robot learning [53]. Imitation in robots (or in animals, or humans) is composed of three fundamental processes, described by Kuniyoshi et al [54] as "seeing, understanding, and doing". It was reformulated in [51] that for an agent to imitate an action by a teacher, it must at least:

1. *Observe* the action.

2. *Represent* the action.

3. And *reproduce* the action.

Action representation is the key for all three steps. Parallel to the evolution from traditional AI to behavior-based robotics, symbol systems were first used in imitation, such as Kuniyoshi's first imitation system [54]. In this experiment, a robot agent watches a human teacher performing a simple assembly task in a tabletop environment. Figure 24 shows the overview and hardware configuration of this system. In this early paper, traditional AI-based symbolic logic was used as the foundation.

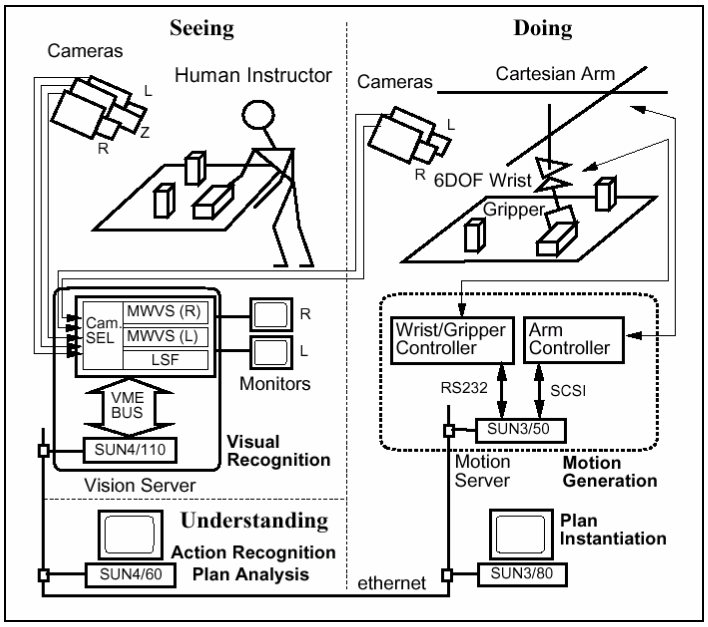


Figure 24: Overview and hardware configuration of the learning system [54]

In the "seeing" mode, the vision system perceives the teacher's hand movements, determines the start and finish points of actions, and tracks the human hand. Then these movements are recognized as the actions it already knows, e.g., pick, move, place, etc. The robot was explicitly told when to reproduce the observed actions by switching to the "doing" mode. The initial state of the table is analyzed, and the parameters of the action sequence are changed to conform to the change of the state, e.g., if the position of the objects on the table has changed between the seeing and doing modes.

Hayes and Demiris showed in [55] an imitation-based learning system for mobile robot navigation in simulation. A robot agent follows a teacher robot through a maze, detecting significant teacher actions such as turning, and physically copying those actions. The agent also learns to associate the environment – the position in the maze (in terms of local wall position) – with the teacher's actions. As was common at that time, a symbolic representation is used. Actions are represented as symbolic rules. The antecedent of such a rule is a description of the environment in which an action occurred, and the right-hand-side is the action to be executed in that environment. For example, right-hand turns become associated with an environment in which there is one wall to the left and one straight ahead. The authors didn't finish the real implementation in that paper.

Traditional AI logics have been abandoned in the new imitation based learning. Primitives, or basis behaviors (discussed later), have been used to represent actions. This section reviews recent work in this field. First, imitation-based learning methods in humanoid robots are presented, including Mataric's work, Kuniyoshi's work, and the dynamic brain project. Then, imitation in mobile robots is dicussed, mainly Gaussier's work. Special emphasis is put on how to detect salient features to facilitate the agent in forming the sensory-motor primitives. At the end are the critiques of imitation-based learning.

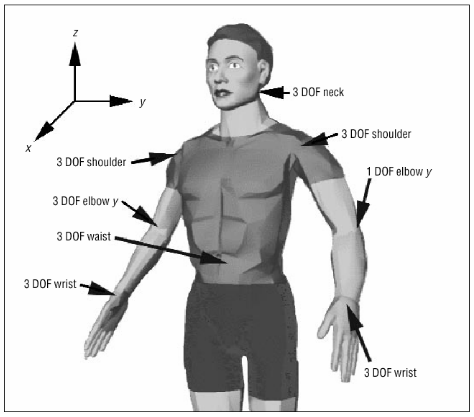
Imitation: Mataric Group's Approach

Recently, Maja Mataric and her group have been exploring the role of basis behaviors as sensory-motor primitives in imitation-based learning, i.e., using basis behaviors to represent the action. This is not surprising, as she introduced the notion of basis behavior in 1992 [56]. She was the Ph.D. student of Rodney Brooks, who proposed behavior-based robotics [4]. In Brooks' subsumption architecture, multiple asynchronous behaviors are subsumed and sequenced to produce intelligent behavior. The basis behavior set of a system provides elements that are not further reducible to each other and that, when composed by sequential or concurrent execution, produce the complete behavior repertoire for the system. If the total behavior is regarded as the total vector space, then the basis behavior set can be thought of the canonical supporting basis. In her Ph.D. thesis [57], she outlined desirable criteria for selecting and evaluating basis behaviors.

She later generalized the notion of basis behavior to multi-robot interaction and demonstrated how a small set of basis behaviors per robot can be used to demonstrate a rich repertoire of individual and group-level behaviors, including the following: flocking, homing, herding, aggregation, dispersion, and formations [58-60]. Basis behaviors are constructed, learned, or evolved from stable, robust interaction dynamics between the agent/robot and its environment, and serve as a substrate for the system's more complex behaviors. Among these methods, learning is definitely the preferred choice.

Published Mataric group's research projects on imitation have been done on two simulation platforms, Adonis and Cosimir. Adonis is a 20 DOF (degree-of-freedom) 3D rigid-body simulation of a human with full dynamics developed at the Georgia Institute of Technology Animation Lab, as shown in Figure 25(a). Cosimir is a 65 DOF 3D humanoid avatar developed at the University of Dortmund, as shown in Figure 25(b).

Her group has experimented with two types of primitives: innate and learned. Innate primitives are user-selected and preprogrammed. Learned primitives are derived directly from human movement data. They have gathered different types of such data using different methods [61, 62]: vision-based motion tracking of the human upper body, magnetic markers on the arm using the FastTrak system, and full-body joint angle data using the Sarcos Sensuit.



(a) Adonis

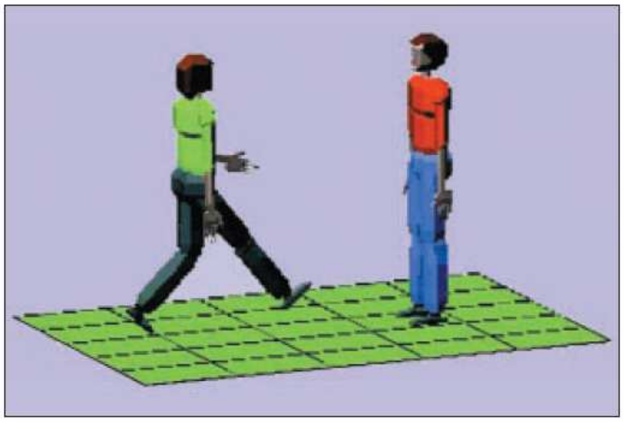

(b) Cosimir

Figure 25: The Adonis and Cosimir simulation platforms[63]

In [64], Mataric's group has demonstrated the effectiveness of a basis set consisting of three types of movements:

1. discrete straight-line movements of sub-sets of DOF (degrees of freedom), accounting for reaching-type motions;
2. continuous oscillatory movements of sub-sets of DOFs, accounting for repetitive motions;
3. postures, accounting for large subsets of the body's DOFs.

Figure 26 shows the data flow of the imitation system used in [62]. As stated before, imitation involves three steps: seeing, understanding, and doing. In this figure, thin lines represent the flow of data in the training phase. Thick lines represent the flow of data in the testing phase. All dashed lines represent the use of previously computed data. The three layers of the imitation architecture are shown vertically. In the training/learning phase, training data are first filtered to remove incidental random zero-velocity points caused by noise. Next, the data are segmented, and principal component analysis (PCA) is applied to obtain a set of eigenvectors, thus greatly reducing the dimensionality of the movement data. Then, clustering techniques are used to extract patterns of similar movements in the data. These clusters or patterns form the basis for the primitives.

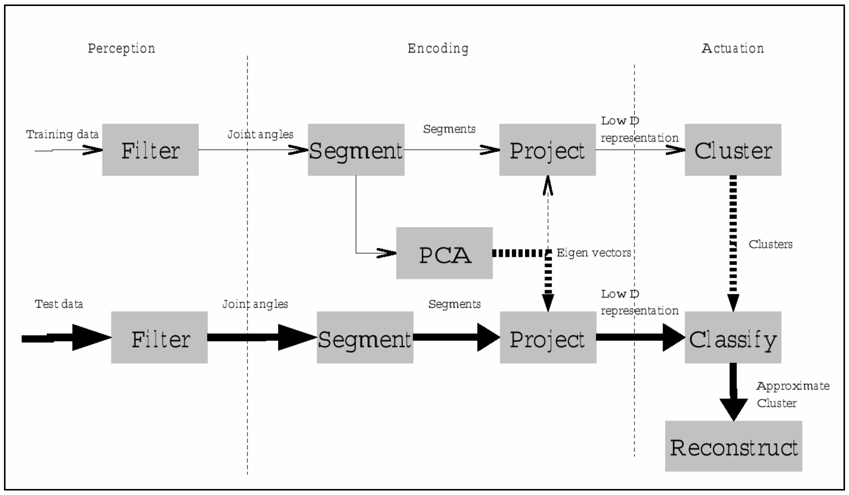


Figure 26: Data flow of the imitation system [62]

Mataric's group demonstrated another imitation based learning system in [65]. Two Cosimir humanoid avatars [66] were designated as the imitator and the imitatee. The imitator learns different sequences of limb movements, initially demonstrated by the imitatee avatar. The control architecture of this system is inspired by neurological models of visuo-motor processing [65]. The architecture is divided into three parts, visual recognition, motor control, and learning (Figure 27). Three types of sequence are learned: 1) repetitive patterns of arm and leg movements; 2) oscillatory movements of shoulder and elbows, using video data of a human demonstration; and 3) precise movements of the extremities: grasping and reaching.

Currently, none of their experiments have been done on the real humanoid robot. This violates the embodiment principle of cognitive robotics. They plan to use physical humanoid robots as the ultimate test beds for evaluating the imitation architecture.

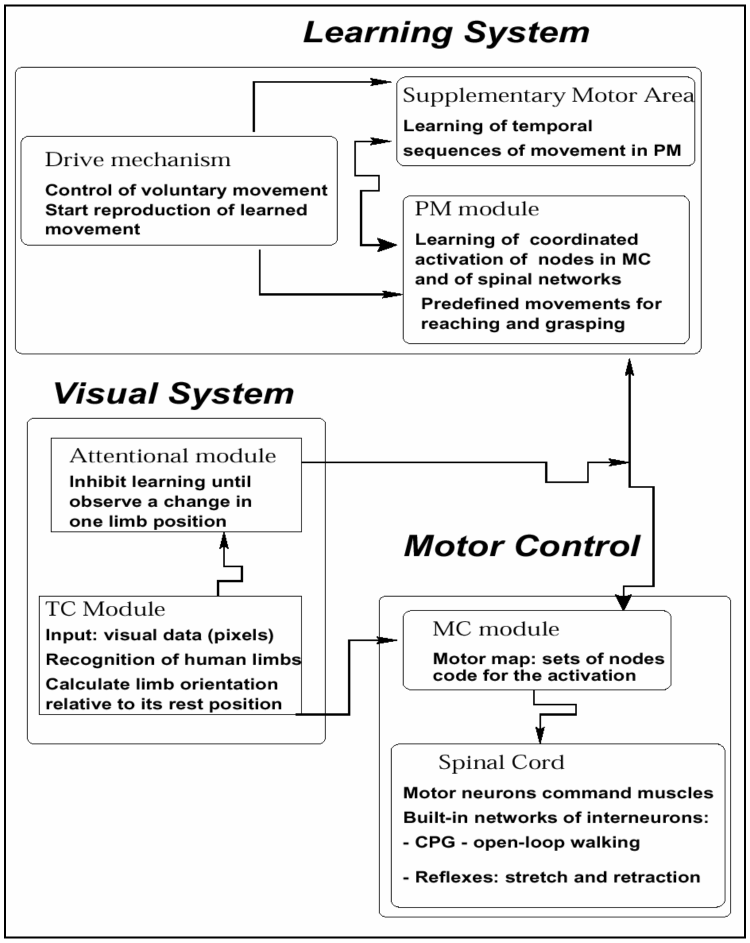

Learning System
Supplementary Motor Area
Learning of temporal sequences of movement in PM
Drive mechanism
Control of voluntary movement
Start reproduction of learned movement
PM module
Learning of coordinated activation of nodes in MC and of spinal networks
Predefined movements for reaching and grasping
Visual System
Attentional module
Inhibit learning until observe a change in one limb position
TC Module
Input: visual data (pixels)
Recognition of human limbs
Calculate limb orientation relative to its rest position
Motor Control
MC module
Motor map: sets of nodes code for the activation
Spinal Cord
Motor neurons command muscles
Built-in networks of interneurons:
- CPG - open-loop walking
- Reflexes: stretch and retraction

Figure 27: Imitation-based learning architecture [65]

Imitation: Kuniyoshi Group's Approach

Kuniyoshi has been working on imitation for some time. His early work using traditional AI logics has been discussed in the beginning of this section. In his later research, traditional AI approaches were abandoned and new cognitive approaches were used. Kuniyoshi and Berthouze showed how to let an active stereo head learn several visual behaviors using imitation in [67].

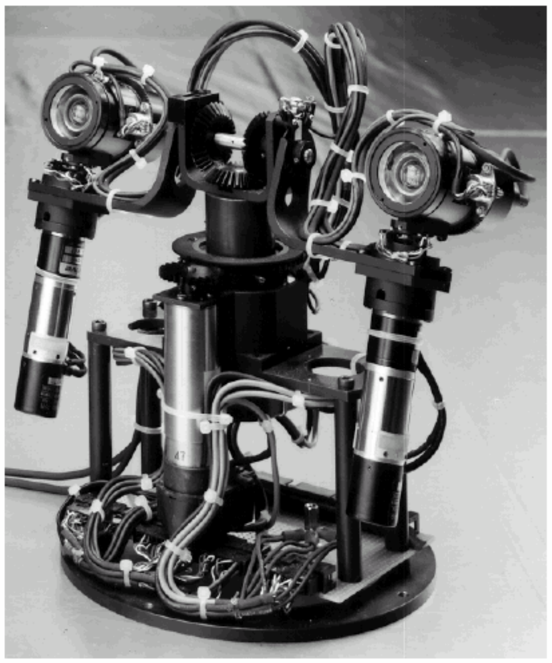


Figure 28: ESCHeR: ETL stereo head for robot vision [67]

ESCHeR (Etl Stereo Compact Head for Robot Vision) is the platform for the experiments, as shown in Figure 28. It's a 4 DOF binocular active vision head equipped with foveated wide angle lens (i.e., the projection curve of the optical system is nonlinear.) Three case studies are presented in the paper [67]. Case study I takes up the issue of self-exploration and emergence of a quasi-stable embodied interaction dynamics. The goal is to learn foveation, i.e., turn the camera head to the target. This is done in simulation using a dynamic neural network model with a temporal learning rule. The learning is embedded in the operation of the network itself, which is suitable for quick on-line adaptation. Case study II takes up on-line adaptation and emergence of coordination issue. It shows Emergent VOR (vestibulo-ocular reflex) behavior out of the coordination of redundant DOFs. This is implemented on the ESCHeR. The distributed FEL (Feedback Error Learning) controllers independently learn the sensory-motor loop between ESCHeR and the environment [68]. Case study III investigates the issue of observing and categorizing the self-interaction dynamics. This issue is a step towards building a neural mechanism which "senses" the emergent higher-order dynamics structure and then creates an internal structure. This is implemented in simulation using a similar neural network structure as in case study I. Fully implemented imitation-based learning is not presented in this paper.

Imitation: Kawato Dynamic Brain Project and CyberHuman Project

The Kawato Brain Project is a joint five year project being undertaken at the ATR Labs in Kyoto, Japan, and at the University of Southern California (USC) in Los Angeles. The project was started in October 1996 and ran until October 2001. After that, it evolved into the CyberHuman Project with collaborators from around the world, including USC, CMU, and Georgia Tech. The aim of these projects is to understand the computational problems of learning and motor control in the human brain. Learning processes in the living human brain are first studied. Models are then developed and simulated using artificial neural networks. These models are then tested using a full sized humanoid robot, DB.

Imitation-based learning is the main learning method adopted in these projects. In [69, 70], the humanoid robot DB learned how to reach out. In [71], which received the ICRA2002 best paper award, DB learnt a tennis forehand swing from a human demonstration, as shown in Figure 29. In this research, the representation of movement plans is based on a set of nonlinear differential equations with well-defined attractor dynamics. These sets of nonlinear differential equations form control policies (CP) that are robust to strong external perturbations and also can be modified on-line by additional perceptual variables. The attractor landscape of the control policy can be learned rapidly with a local weighted regression technique.

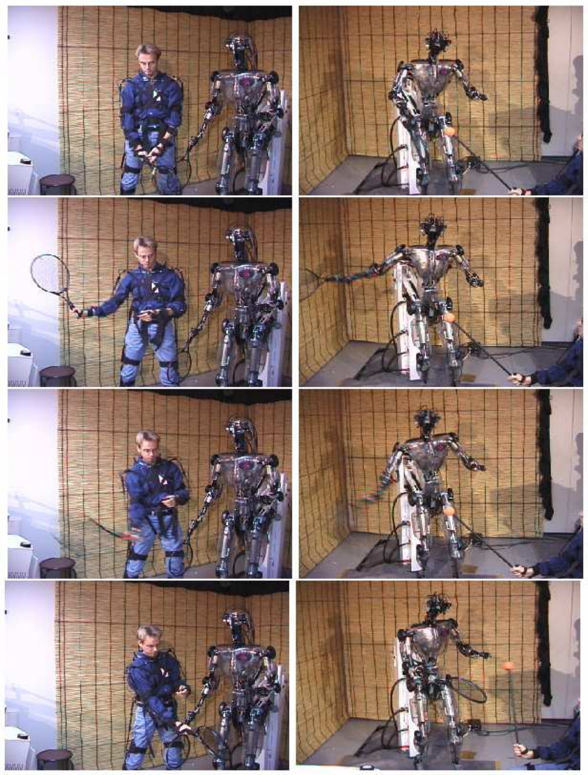


Figure 29: DB learning a forehand swing from a human demonstration [71]

In [72, 73] DB is taught to play air hockey with a person, as shown in Figure 30. They also use primitives, same as basis behaviors, for their representation. Figure 31 shows their framework of imitation-based learning. A human, using domain knowledge, designs the candidate primitives. The primitive-recognition module segments the observed behavior into chosen primitives. This segmented data are then used to provide the training data for the primitive selection, sub-goal generation, and action generation modules. The primitive-selection module provides the agent with the primitive type to perform for the observed state of the environment. The desired outcome of performing that primitive type is specified by the subgoal-generation module. The action-generation module finds the needed actuator commands. The learning-from-execution module provides information to the agent that can be used to improve its performance while operating in the environment.

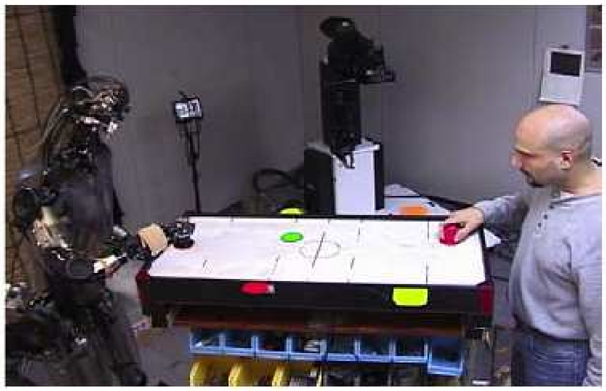


Figure 30: DB playing air hockey [73]

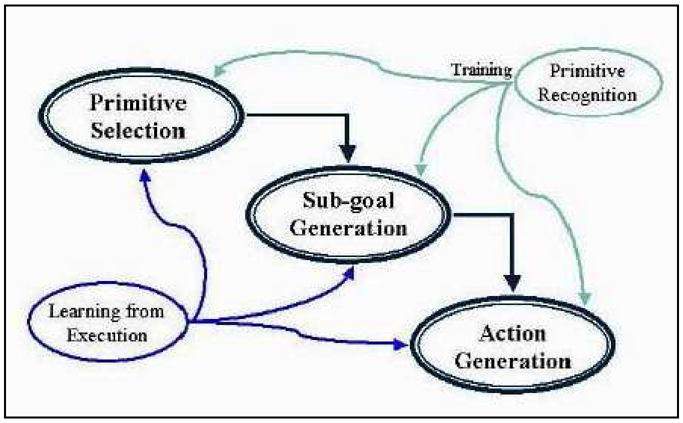


Figure 31: Framework for learning from observation using primitives [73]

Imitation: Gaussier's Approach

Gaussier and Zrehen introduced PerAc (Perception-Action), a neural network architecture to control mobile robots [74]. A PerAc network is the basic building block of this neural network. As shown in Figure 32, each PerAc block consists of two pathways, a reflex pathway and a conditioning pathway. The reflex pathway encodes the inborn, unlearned association between a particular perception and action, while the conditioning pathway encodes the association between a particular perception and the realization of a particular action. This learned association is conditioned by a reinforcement signal that represent the internal motivation of the robot. A winner-take-all (WTA) mechanism select the final action output. This architecture was tested on a mobile robot for scene recognition and landmark-based navigation [74].

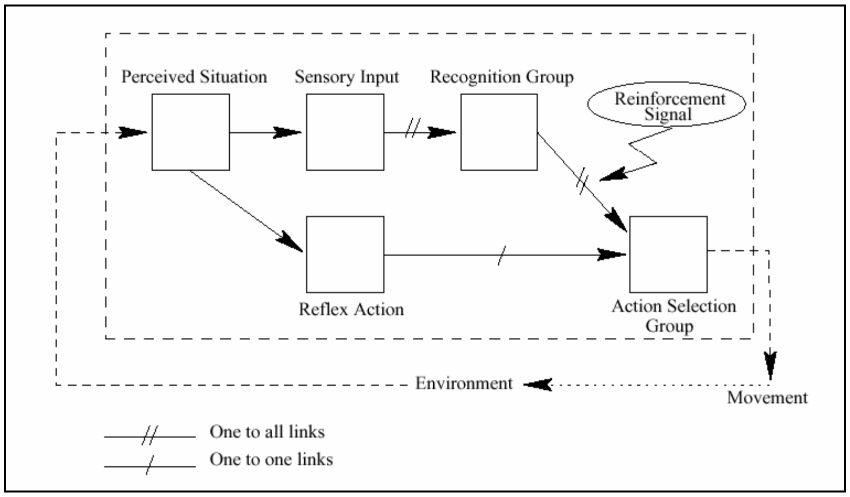


Figure 32: Architecture of a PerAc block [75]

This architecture was then used in imitation-based learning [75, 76]. First, the student robot tries to follow the teacher robot. Following behavior is implemented using optical flow. During following, the temporal sequence of the trajectory is learned by the PerAc architecture, as shown in Figure 33. The TD (Time Derivator) neurons detect the movement changes characterized by OFF-ON transitions of MO (Motor Output) neurons, and these detected changes are used as the input for a bank of spectral neurons called time filter batteries. Time filter batteries (TB) are delay neurons having different time constants. They perform a spectral decomposition of the signal that will allow the neurons in the Prediction Output group (PO) to register transition patterns between two events in the sequence. Figure 34 shows an experimental trajectory learned in the same experiment.

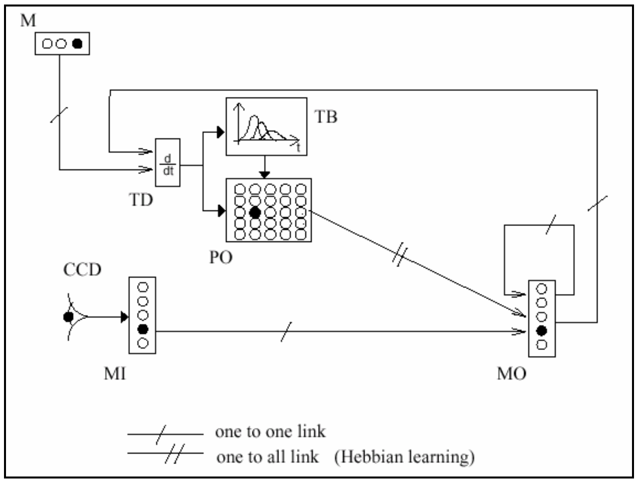


Figure 33: The PerAc architecture for movement sequence learning [75]
In this figure, CCD – CCD camera, M – Motivations, MI – Movement Input,
MO – Motor Output, TD – Time Derivator, TB – Time Battery, PO – Prediction Output

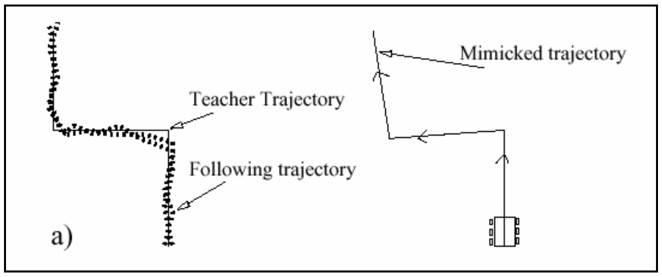


Figure 34: An experimental trajectory learned in the same experiment [75]

Critiques of Imitation-Based Learning

Using a teacher's demonstration as the starting point of student learning can significantly speed up the learning process: imitation drastically reduces the size of the sate-action space that needs to be explored [53]. Bakker and Kuniyoshi listed the promise of imitation in [53]:

- Adaptation

An agent with the ability to imitate has an excellent mechanism for adapting to its environment. By observing other agents' action, the agent can quickly learn new behaviors that are likely to be useful because those behaviors are already being used by agents successfully operating in the same environment.

- Efficient communication

Imitation provides agents with an efficient non-verbal means of communication. Because it is non-verbal, it does not require the teacher and the agent to 'speak the same language'. The agent can learn from other agents that are of different species or are built from different hardware. Also, imitation has the advantage of being an undemanding and unobtrusive communication method because a teacher does not have to go 'off-line' to transfer a behavior to an agent: the agent can learn by observing the teacher without interfering with the teacher's performance.

- Compatibility with other learning mechanisms

Imitation can be used as a learning mechanism in conjunction with existing learning schemes for agents, such as reinforcement learning or symbolic induction schemes. This is evident in the previous survey. For example, Kuniyoshi et al [54] and Hayes et al [55] used symbolic schemes; while Ijspeert et al [71] and Gaussier et al [75] used reinforcement learning techniques.

- Efficient learning

The learning agent does not need to explore a huge space; instead, by observing other successful agents, it can find the stable configuration very quickly. Especially in a society of agents, such learning ensures the survival of useful behaviors: such behaviors will rapidly spread and may even be passed on to following generations.

However, imitation does not come without a price. It requires a leaning agent which has the abilities of "seeing, understanding, and doing" [54], and these abilities are not trivial. Schaal listed some outstanding questions in this field [53]:

- Learning perceptual representations: How can appropriate representations of the identity and movement of others be developed in an automated fashion from visual input?
- Movement primitives: How can new primitives be learned, and old primitives be combined to form higher level movement primitives? How is sequencing and the recognition of sequences of movement primitives accomplished?
- Understanding task goals: How can the intention of a demonstrated movement be recognized and converted to the imitator's goal?

So for in all the surveyed systems, "seeing" – the first stage of imitation-based learning – is accomplished through student's observation of teacher's action. However, in many robotic systems, an easier approach exists. That is, a teacher can directly control the robot via teleoperation. This usually simplifies the perception problem usually encountered in student's observation of teacher's action. Cambron used this approach to let a humanoid robot recognize different grasp poses [8]. Peters et al also used this approach to let Robonaut, a space-capable humanoid robot, learn grasping tasks [9].

Episode-Based Learning

Episode-based learning focuses on the origin of conceptual knowledge. Dr. Paul Cohen and his group currently constitute the main group working on this approach. The origin of conceptual knowledge, in Cohen's words, are "the earliest distinctions and classes, the first efforts to carve the world at its joints" [77]. Cohen's work is highly influenced by the interactionist philosophy of Mark Johnson and George Lakoff [78] and the developmental psychology of Jean Mandler [79]. The goal is an unsupervised learning mechanism for extracting concepts from time series of sensor readings.

This section only discusses Dr. Cohen's work. It is organized as follows. The first part discusses the definition of concept in Cohen's view. Parts II and III discuss how a concept is formed. This is done in two steps, first from activities to prototypes, then from prototypes to concepts. An example of a mobile robot forming categories is presented next. At the end is the critique of this method.

What are Concepts

In [80], Rosenstein and Cohen give "concept" a working definition: Concepts are abstractions of experience that confer a predictive ability for new situations. The meaning of a concept is the prediction it makes. This definition applies equally well to both objects and activities and depends upon a notion of "category". Objects and activities are dual – linked by sensorimotor experience – with a category of one connected to a category of the other.

Epistemically, a category is simply a collection of instances (of objects or activities) and a concept is a category plus its entailments (or consequences of category membership). A prototype is a representative category member that may or may not correspond to any observed instance. One can think of "concept" as a category prototype plus its meaning or predictive inferences. This is illustrated in Figure 35.

In this framework [80], the concept discovery begins with time series data. Time series form clusters of points that have something in common, and the subsequent outcomes, i.e. the future properties of the cluster members, can then be observed. Since clusters are equivalent to categories and outcomes to entailments, the corresponding acts of clustering and observation are equivalent to the discovery of concepts. Concept use then follows a similar two-step procedure: (1) Recognition. Given a new time series, find the cluster prototype most like the new instance. (2) Prediction. Report the most likely outcome for the cluster referred to by the matching prototype.

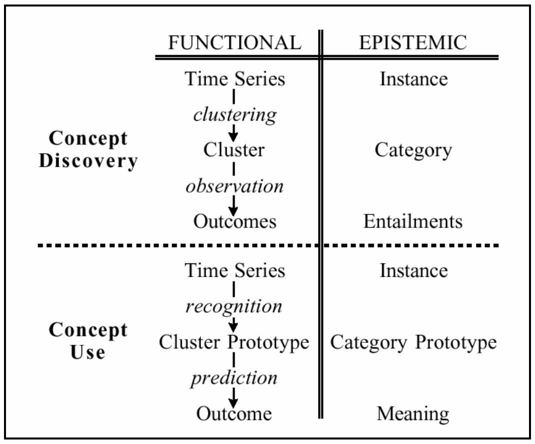


Figure 35: Function view of concepts and the corresponding epistemic terms [80]

The next two sections will discuss the methods Cohen's group has used for concept discovery. As shown in the figure and discussed above, this is done in two steps: from activities to prototypes and from prototypes to concepts.

From Activities to Prototypes

Dr. Cohen's group has developed several methods to form prototypes, called, collectively, clustering by dynamics [77]. Two methods deserve some discussion here: the method based on dynamic time warping, and the method based on delay coordinate embeddings. The method based on dynamic time warping was developed by Tim Oates [77, 81, 82]:

1. A long multivariate time series is divided into segments, each of which represents an *episode* such as moving towards an object, avoiding an object, crashing into an object, and so on.

2. A dynamic time warping algorithm compares every pair of episodes and returns a number that represents the degree of similarity between the time series in the pair. Dynamic time warping "morphs" one multivariate time series into another by stretching and compressing the horizontal (temporal) axis of one series relative to the other [83]. If two multivariate series are very similar, relatively little stretching and compressing is required to warp one series into the other. A number that indicates the remaining mismatch after the optimal warping is thus a proxy for the similarity of two series.
3. Having found this similarity number for every pair of episodes, it is straightforward to cluster episodes by similarity. An agglomerative algorithm merges clusters until a simple stopping criterion is met.
4. Another algorithm finds the "central member" of each cluster, which is called the cluster prototype.

Another way to cluster episodes is with the delay coordinate embedding method developed by Michael T. Rosenstein [80, 84]. This method converts a time series into a point in a high dimensional space, and this transformation is called ***delay coordinate embedding***. In this transformed space, the Euclidian distance between two points is a measure of similarity of episodes. In one experiment [80], they implemented two simulated agents, each of which adopts one of nine behaviors, including crash, avoid, kiss, and so on. Delay coordinate embeddings were constructed for each episode. With a little training, the learning procedure came up with six clusters (Figure 36).

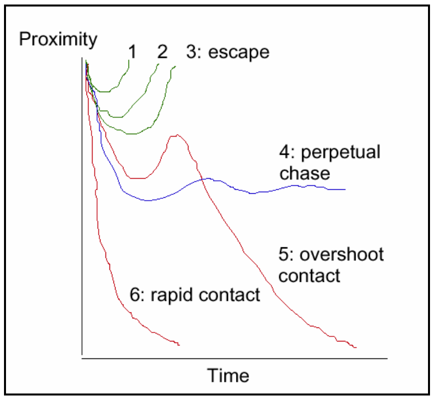


Figure 36: Six prototypes for agent interactions [77]

In all their work, the concept of "episode" is the central idea. Episodes are either partitioned by humans, or automatically found by algorithms that look for simultaneous changes in multiple state variables. In [85], two methods are discussed to find the episode boundaries automatically, simple compression and tree building.

From Prototypes to Concepts

In Cohen's work, "roles" are the key to having the robot learn ontology through interaction with its environment [77]. Each of the robot's activities has roles. Roles are predicates that describe the relationships between the robot's actions and objects in its environment. For example, the robot is an actor, it approaches an object, it pushes the object to a destination, and passes another object on its left. A scene is described and presumably represented in terms of the roles of its participants – the causal relations that hold among the participants' actions.

Prototypes are monolithic representations of activities. For prototypes to become concepts, they have to denote things and relationships explicitly. Cohen et al [86] have developed another approach to "clustering by dynamics" that works with the output of a primitive perceptual system. The idea is to first collect objects that play roles in activities and then inductively extract the features of the objects that cause it to play the roles it has in given activities.

Example: Continuous Categories for a Mobile Robot

Michael Rosenstein and Paul Cohen gave an example of how this framework works in [7]. They used a RWI Pioneer 1 mobile robot, which were equipped with a camera, a gripper, and sonar sensors. As discussed before, the concept discovery has two steps, first, from activities to prototypes, and then, from prototypes to concepts. In the first step, time series (i.e., activities) are partitioned into episodes, and prototypes are formed by averaging all corresponding episodes. In the second step, prototypes are clustered to form categories (i.e., concepts).

Figure 37 is a schematic of the entire learning algorithm that runs both incrementally and in real time as the robot interacts with its environment. On the left, raw sensor readings are made available to the robot at a rate of 10Hz, with the event detectors continuously monitoring the time series for abrupt changes. When an event occurs, the algorithm collects a five-second segment of sensor readings and constructs a bit vector that indicates the active sensors. Figure 38(a) shows the robot bumping into a wall. By averaging seven instances, the prototype is obtained. Component time series are: (b) translational velocity in mm/s, (c) forward sonar in mm, and (d) bump sensor (on or off). Gray regions indicate the level of prototype variability (one standard deviation from the mean).

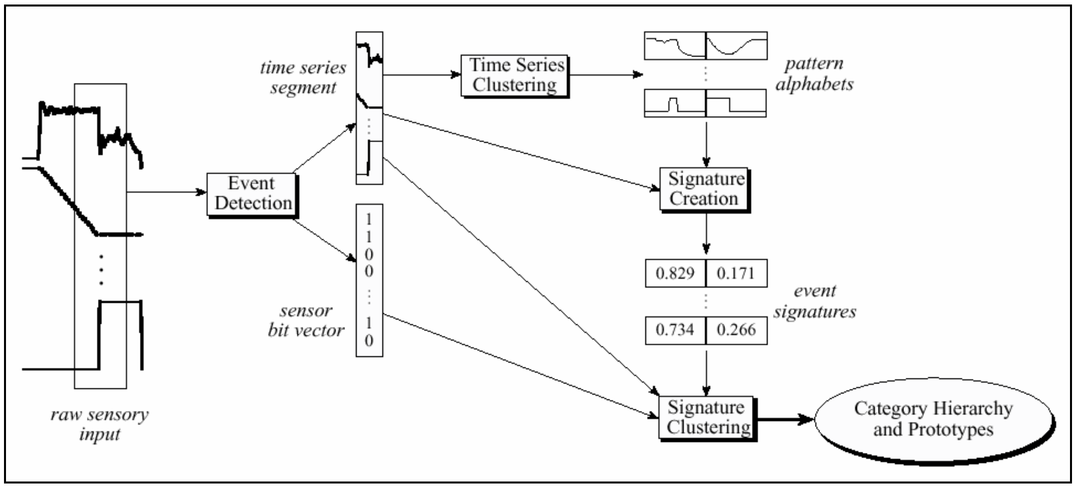


Figure 37: Schematic of the category learning algorithm [7].

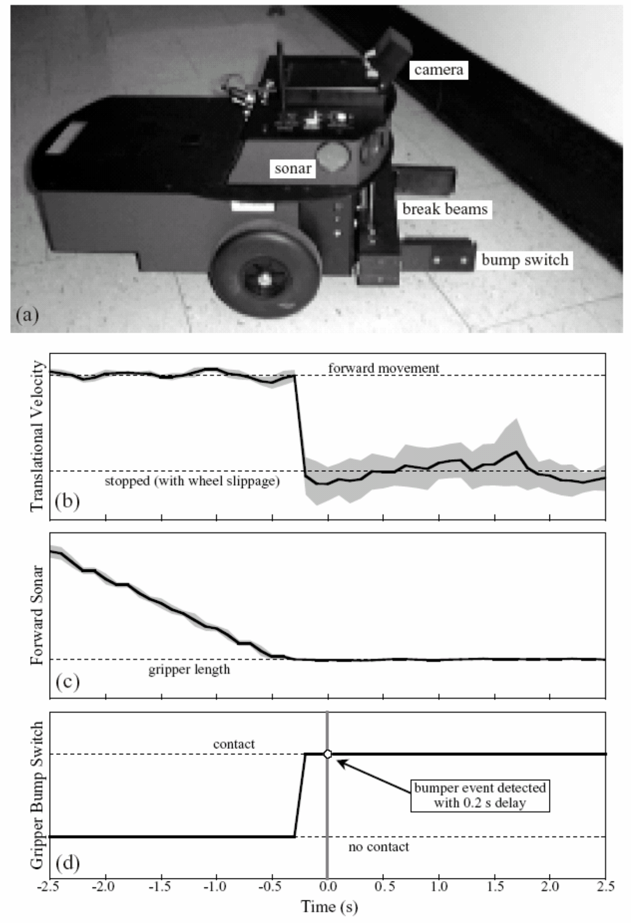


Figure 38: Prototype for seven instances of the Pioneer 1 mobile robot [7]

These sensory-motor data are then passed to the clustering algorithms for additional analysis. First, the new time series are used to update the pattern alphabet for each sensor. Next, these same time series are converted to a set of signatures for subsequent clustering. On the right of Figure 37, the final output is a hierarchy of sensory categories with each category represented by a prototype like the one in Figure 38. Figure 39 shows the resulting category hierarchy for five interactions with each of six objects

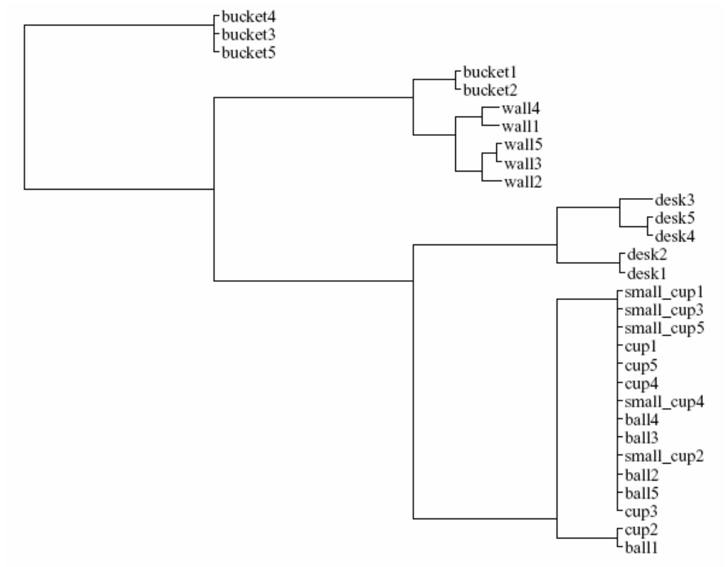


Figure 39: Category hierarchy for five interactions with each of six objects [7]

Critiques of Episode-Based Learning

Traditional AI uses symbols as the fundamental building blocks. In robotics, this traditional approach usually leads to the notorious "symbol grounding problem". In his behavior-based robotics, Rodney Brooks introduced the radically different "intelligence without representation" philosophy [10]. He totally abandoned the representation issue, and regarded "the world is its own best model" [12]. However, learning is a big benchmark for true "intelligence", and learning usually requires representations and concepts. Even in Brooks' subsumption architecture, a behavior (e.g., obstacle avoidance) requires the designer to have the concepts of "obstacle" and "avoidance". Episode-based learning tries to let the robot learn these concepts by interaction with the world instead of "inherit" these concepts from the designer.

Currently Dr. Cohen is the main person working on this episode-based learning approach in robotics, though the idea has been around in philosophy and psychology. This approach is a good method to solve the symbol grounding problem at the bottom level. Currently it is still at the beginning stage and needs more time to mature.

Summary

Learning is regarded as the hall-of-fame of human intelligence. It is what we want the robot to be able to imitate from us. There are many ways to classify learning in general. In Chapter 18 of [87], learning is categorized into six classes: speed-up learning, learning by taking advice, learning from examples, clustering, learning by analogy, and discovery. This chapter has presented three current learning approaches in robotics: sensory-motor coordination, imitation-based learning, and episode-based learning. Loosely speaking, sensory-motor coordination is similar to reinforcement learning. The agent learns a specific sensory-motor coupling based on reward functions. This is evident in both Pfeifer's and Grupen's experiments. Imitation-based learning is a type of supervised learning. The agent observes a teacher's behavior, understands it either in symbols, basis behaviors (primitives), or other means, then tries to reproduce that behavior. Episode-based learning is unsupervised learning. The agent tries to learn concepts by forming prototypes and clusters.

All three methods are related. Basis behaviors or primitives are widely used in imitation-based learning as the canonical base for representation. A basis behavior itself is usually a tight sensory-motor coupling. In episode-based learning, an episode is also usually a tight sensory-motor coupling. The prototype of the episode cluster can be regarded as the basis behavior or its superposition.

They are also related to competency modules in Pattie Mae's Spreading Activation Network (SAN) [88, 89]. In SAN, each competency module is connected to both pre-conditions and post-conditions. Pre-conditions can be regarded as the sensory conditions for a behavior, the competency module is the behavior itself, and post-conditions the result of the behavior. A path in SAN can be thought of as a sequence of sensory-motor coupling, as a sequence of basis behaviors, or as a sequence of episodes. In this sense, sensory-motor coordination/basis behavior/episode can be the building blocks for planning/sequencing to implement complex, autonomous behaviors.

From the vector space point of view, all these methods try to find the stable and/or optimal clusters in the high dimensional sensory-motor space. These clusters can be explicit or implicit. An example of explicit clusters is Cohen's clustering in episode-based learning. Implicit clusters have many forms. In Pfeifer's SMC agent, clusters are converged regions in the neural networks. In Grupen's grasping experiment, clusters are attractors of the controllers.

All these learning methods try to solve the mid-level learning problem. It is assumed that the agent has already acquired a rudimentary set of basic behaviors, and the goal is to learn how to sequence these rudimentary behaviors and how to acquire new basic behaviors. Behavior sequencing requires detection and learning of salient sensory features, the focus of the next chapter.

CHAPTER VI

SALIENT FEATURE DETECTION FROM SENSORY-MOTOR SEQUENCES

The research presented in this dissertation is on the extraction of salient visual and non-visual features based on the SMC principle. This chapter is organized as follows. Section 1 presents the hypothesis, and its underlying assumptions. Section 2 discusses the procedure for salient feature detection. Section 3 gives the mathematical algorithm for salient feature discovery, and Section 4 discusses system performance evaluation.

Hypothesis and Assumptions

The discovery of environmental features salient to the action of an independent agent by signal processing/computer vision (SP/CV) algorithms alone has enjoyed some success in structured environment; however, in semi-structure and unstructured environment, these methods have had limited success [90]. Real time CV applications in mobile robots are still rudimentary. One reason for this is that, as discussed in Chapter III and IV, internal information processing approaches are insufficient to solve many perception/categorization problems. This is due to the fact that the set of all feature descriptors used by these SP/CV algorithms is too large to search [25]. However, salient feature detection coupled with motor control reduces the search space significantly. This has been shown by Pfeifer *et al* [6, 38, 91, 92] and others. It is referred to as the sensory-motor coordination (SMC) principle.

One objective of this research is to enable a mobile robot to discover features salient to tasks, such as navigation, by incorporating the SMC principle. This is based on the hypothesis that SMC forms clusters in the sensory-motor space, i.e., features that are consistent with respect a specific behavior in a specific environment are the salient features, while those features that are NOT consistent with respect to a specific behavior are not salient and can be ignored. After such sensory-motor couplings are found, they can be sequenced later to carry out similar tasks.

An SMC ***event*** is defined as a motor action that occurs in response to a sensory stimulus, or a sensory stimulus that occurs in response to a motor action. An SMC ***episode*** is defined as the state of the robot's sensors and actuators between consecutive SMC events [9, 85]. If the sensor values and the motor control parameters are written together as a vector, then an episode is a quasi-stationary vector time-series (cf. delay coordinate embedding transformation discussed in last chapter). Some of these vectors form clusters in the sensory-motor vector space. Each cluster corresponds to a consistent sensory-motor couplings. An exemplar (i.e., prototype) of such a cluster is a category that either specifies the action to be taken under specific sensory conditions, or the sensory conditions to expect during a specific action in a specific environment. In effect, the exemplar defines a basic behavior (cf. Chapter V, Mataric's work). By learning SMC categories a robot could acquire basic behaviors. Tasks can then be described by a sequence of exemplar vectors.

The learning of task-level behavior sequences has been based on a number of hypotheses. A robot is assumed to have necessary sensing modalities and motor abilities for a given task. When an operator performs the given task using the robot via teleoperation, it is his/her SMC that is controlling the robot. So controlled, the robot's sensors detect its own internal states and those of the environment as it moves within it. Thus the robot can make its own associations between coincident motor actions and sensory features as it is teleoperated. In repeating a task several times, a teleoperator will perform similar (albeit nonidentical) sequences of motor actions whose dynamics will depend on her/his perceptions of similar sensory events that occur in similar sequence. As a result, the robot will detect a similar set of SMC events during each teleoperation trial. Therefore, each trial can be partitioned into *p* episodes, demarcated by the common SMC events. Sensory events that are salient to the task will occur in every trial. Sensory signals that differ across trials are presumed to be insignificant for the task and can be ignored. By simply averaging the time-series for each episode point-wise over the trials, a canonical representation of the motor control sequence can be constructed. Also, as the result of the averaging, true events in the sensory signals will be enhanced while those that are random will be suppressed. These canonical SMC sequences can later be executed to carry out the task under similar circumstances. In these executions, the episode transitions are triggered by the salient sensory events that the robot has learned.

This approach to behavior and task learning has been applied on two humanoid robots. Cambron [8] used this approach to extracted salient features for a humanoid robot to grasp poles of different shapes. Peters et al [9] applied this approach to facilitate Robonaut, NASA's space capable humanoid robot, to acquire task-level skills. My research is the first to use this approach on a mobile robot.

Experimental Procedures

To enable a mobile robot to learn salient features for navigation, it is assumed that the mobile robot has already implemented a rudimentary set of basic behaviors such as, move forward, turn, obstacle avoidance, etc. The robot is to learn how to navigate in a specific semi-structured or unstructured environment using the salient features it detects.

The learning procedure described here is independent of the environment; by using this procedure a robot should be able to learn any specific environment.

Salient feature detection is implemented in three steps: teleoperation and data recording, off-line data association, and execution and evaluation. First, the mobile robot is teleoperated in an environment along a path. This is done several times (5 to 7 times in my experiments). Each time, the robot runs along essentially the same path with random small deviations. All sensory data, both visual and non-visual, along with the motor commands, are recorded throughout the whole trial and saved. During the off-line association phase, signal analysis and image processing techniques constrained by motor commands (cf. next section) are used to identify salient features that are relevant to the task. These techniques also identify the events that consistently co-occur with specific motor states or state transitions. Each SMC event comprises a motor state together with sensory pre-conditions and post-conditions. An observed sequence of such events define a chain whose nodes are basic behaviors and whose edges are the transitions between them. If the robot is put back into the same environment somewhere along the path through which it has been previously teleoperated and asked to carry out the task it has learned, its performance can be measured.

The key phase is the off-line data association, which involves a number of steps. First, the original SMC time sequences are cut into episodes. There exist a number of techniques for the automatic partitioning of a vector time series into episodes [61, 62, 93]. In our experiments, episodes were partitioned according to the motor command variations, i.e., a significant change in motor activity demarcates an episode boundary.

All sensory data, including visual and non-visual, are partitioned according to the motor episodes. The episodes from various trials are then split and merged (cf. next section) so that each trial has the same number of episodes and so that the same episode from different trials corresponds to the same phase of the task. The corresponding episodes in all trials are time-normalized (i.e., resampled to have the same signal length) then averaged across the trials. The result is an exemplar signal of motor commands and sensory events. The sensory features that correspond to motor events in the exemplar sequence are chosen as the salient features, while those that don't correspond well are discarded. The results are used to define an event sequence.

Once the salient features have been extracted, the robot is put back into the environment and is asked to perform the task it has been taught. The robot compares the current sensory-motor patterns with the learned ones. When a learned salient event is detected, the corresponding behavior is performed. Thus the robot performs the task by following the salient SMC event sequence.

Note that this learning procedure is independent of the environment, that is, through this procedure a robot could learn any specific environment. It combines some elements from three of the approaches discussed in the last chapter, such as using human's operation to bootstrap the learning process (imitation-based learning), cutting sequence into episodes (episode-based learning), and incorporating motor commands to extract salient visual features (sensory-motor coordination). However, overall, this approach differs from all the methods discussed in the previous chapter:

- Among the three approaches, it resembles the imitation-based learning method the most. The main difference here is that instead of letting the robot observe the teacher, the teacher teleoperates the robot directly. Recognizing teacher's movement is not a trivial task, and teleoperation simplifies this step.
- The purpose of Cohen's episode-based learning is to let the robot learn concepts by interaction with the real world. The purpose of interaction here is only to learn

concepts. He has not shown how these concepts can be used to perform a certain task. Instead, the aforementioned learning procedure discovers features salient to a certain task in a specific environment.

- Pfeifer's and Grupen's approaches to categorization are "on-line" learning methods, which form SMC clusters (Pfeifer) or attractors in control strategy space (Grupen). This procedure is an off-line learning method that uses all the learning history.

Offline Data Analysis

This section explains the offline data analysis in formal mathematics. Assume there are M trials $T_1,\cdots,T_M$ of task performed during teleoperation. For each trial, T_i, assume N separate signals (from different sensors) $\mathbf{s}_{i,j}(t)$, are recorded. Then

$$\mathbf{v}_i(t) = \begin{bmatrix} \mathbf{s}_{i,1} & \cdots & \mathbf{s}_{i,N} \end{bmatrix}^T (t) \tag{6.1}$$

is the vector time-series recorded during trial T_i. In general, $\mathbf{s}_{i,j}(t)$ is itself a vector time-series, such as odometry $[x(t), y(t), \theta(t)]^T$. But $\mathbf{s}_{i,j}(t)$ could also be a scalar signal, such as digital compass output. We assume that t is discrete, defined only at integer multiples of the sampling interval, τ, so that

$$t \in \{n\tau\}_{n=1}^{\infty} \tag{6.2}$$

Hence, without loss of generality we define $t \in \mathbb{Z}^+$, the positive integers.

By assumption, each trial contains the same number, P, of episodes, $E_{i,k}$, which follow the same sequence, $E_{i,1},\cdots,E_{i,P}$, within each trial, $i=1,\cdots,M$. Thus $\mathbf{s}_{i,j}(t)$ is the *j*-th signal from the *i*-th trial, and $E_{i,k}$ is the *k*-th episode from the *i*-th trial. Moreover,

$$E_{i,k} = \{\mathbf{v}_i(t)\}_{t=(t_{i,k-1})+1}^{t_{i,k}} \tag{6.3}$$

for $k = \{1, \cdots, P\}$, where $t_{i,k-1} + 1$ is the time at which the *k*-th episode, $E_{i,k}$, starts in trial T_i, and $t_{i,k}$ is the time at which the *k*-th episode, $E_{i,k}$, ends. We define $t_{i,0} + 1$ as the starting time of trial T_i. Note that in general

$$t_{\eta,k} - t_{\eta,k-1} \neq t_{\nu,k} - t_{\nu,k-1} \tag{6.4}$$

That is, the *k*-th episode from trial η will not have the same duration as the k-th episode from trial ν.

If the tasks are all recorded with the same τ, then the number of samples in corresponding episodes will differ. Let $\#\{\bullet\}$ represent the cardinality operator so that $\#\{E_{i,k}\}$ is the number of samples in episode *k* of trial *i*. Then usually

$$\#\{E_{\eta,k}\} \neq \#\{E_{\nu,k}\} \tag{6.5}$$

if $\eta \neq \nu$. To compute a characteristic representation of the task, the corresponding episodes in each task must have the same duration – the same number of samples. Therefore, each episode, $E_{i,k}$, was resampled to form a new one, $E'_{\eta,k}$ such that

$$\#\{E'_{1,k}\} = \cdots = \#\{E'_{M,k}\} = \frac{1}{M}\sum_{i=1}^{M} \#\{E_{i,k}\} \tag{6.6}$$

The length of episode $E'_{i,k}$ is the average over all trials of the number of samples in *k*-th episode. Thus,

$$E'_{i,k} = \{\mathbf{v}'_i(t)\}_{t=t_{i,k-1}+1}^{t_{i,k}} \tag{6.7}$$

where $\mathbf{v}'_i(t)$ is the resampled vector time-series,

$$\mathbf{v}'_i(t) = \left\lceil \mathbf{s}'_{i,1} \; \cdots \; \mathbf{s}'_{i,N} \right\rceil^T (t) \tag{6.8}$$

and indices $\{t_{i,k}\}_{k=1}^{P}$ have been reassigned to the new time-series. The resampling was done in **MATLAB** using its **resample** function in experiments [94].

Episode Extraction

There are a number of technique to automatically partition a vector time-series into episodes. Cohen used dynamic time warping [82, 93]. Mataric used the mean-squared magnitude of the joint velocities of the arm and hand for a humanoid robot [62]. Peters used both elbow joint angle and Mataric's method on Robotnaut [9]. In the experiments for this work, episode boundaries were detected by thresholding the linear and angular velocity, i.e., whenever there was a sudden change of linear and/or angular velocity, a boundary was marked. The partition results were then verified by human's visual inspection due to high-frequency noise. This will be further discussed in Chapter VIII.

Computation of Characteristic Signals

Once the episodes were found in all trials the characteristic signals were computed by averaging. In summary, from the M trials, a characteristic vector time-series was computed as follows:

1. The vector time-series, $\mathbf{v}_i(t)$, were compared across all trials to determine the motor events characteristic of the task.
2. The motor events were used to partition each $\mathbf{v}_i(t)$ into P episodes, $E_{i,1},\cdots,E_{i,P}$.
3. Each episode $E_{i,k}$ was time-normalized through resampling to create a new representation, $E'_{i,k}$. This is based on the hypothesis that during a specific episode, the salient sensory signals take on discernible characteristic shapes. During separate trials

the time elapsed in a given episode would vary, but the shape should remain more or less consistent well within the limits of detectability.

4. For each episode, each time-normalized signal was averaged across trials. Mathematically, a mean vector time-series is computed via

$$\bar{\mathbf{v}}_k(t) = \frac{1}{M}\sum_{i=1}^{M}\begin{bmatrix}\mathbf{s}'_{i,1} \\ \vdots \\ \mathbf{s}'_{i,N}\end{bmatrix}(t) \tag{6.9}$$

   for each $k \in \{1, \cdots, P\}$.

5. The resultant characteristic time series is the concatenation of the episode sample-wise means,

$$\bar{\mathbf{v}}(t) = \{\bar{\mathbf{v}}_1(t) \mid \bar{\mathbf{v}}_2(t) \mid \cdots \mid \bar{\mathbf{v}}_P(t)\} \tag{6.10}$$

   The final characteristic time series defines an exemplar sensory-motor sequence. It is then used to extract salient sensory-motor features.

Sensory Signal Association

The characteristic vector time series, $\bar{\mathbf{v}}(t)$, contains not only motor control data but also sensory signals. A simple replay of the average of the motor control signals is not very useful since the task could only be successful if the environment were in exactly the same configuration as it was when the data were collected. Instead, the sensor data were analyzed trying to find features that occurred within a short time window of episode boundaries. Any sensory event that was a random occurrence in the M trials would be diminished, cancelled, or masked by noise through the episode averaging procedure. On the other hand, any event that occurred consistently with respect to an episode boundary would be preserved by the averaging. These features were selected as the salient features for that specific episode.

There is no universal definition of "salient". From the above discussion it can be concluded that, for a feature to be regarded as "salient" for a behavior, the feature needs to satisfy at least the following two conditions:

1. It should be consistent between different trials;
2. It should correlate well with the behavior change within a narrow time window.

These two criteria are applied in the following salient feature selection procedure:

1. Consistency Criterion

A square window, W_k, is used on every episode boundary B_k, and a consistency index, $CI_{j,k}$, is calculated for each signal by summing and averaging in the window across all the trials $i = 1, \cdots, M$:

$$CI_{j,k} = \sum_{W_k} \left(\frac{1}{M} \sum_i \left| \frac{\mathbf{s}'_{i,j}(t) - \overline{\mathbf{s}}'_{i,j}(t)}{\max(\overline{\mathbf{s}}'_{i,j}(t))} \right| \right) \tag{6.11}$$

where $\mathbf{s}'_{i,j}(t)$ is the episode-normalized signal of trial i and feature j; $\overline{\mathbf{s}}'_{i,j}(t)$ is the average signal across all the trials obtained by the procedure given above. The smaller the value of $CI_{j,k}$ is, the more consistent the $\mathbf{s}'_{i,j}(t)$ (and $\mathbf{s}_{i,j}(t)$, the original sensory signal) is across all the trials. If $CI_{j,k}$ is less than a preset threshold, feature j of signal $\mathbf{s}_{i,j}(t)$ is selected as a candidate salient feature for episode k.

Alternatively, a variance index, $VI_{j,k}$, can be similarly calculated for window W_k:

$$VI_{j,k} = \sum_{W_k} \left(\sqrt{\frac{1}{M} \sum_i \left(\frac{\mathbf{s}'_{i,j}(t) - \overline{\mathbf{s}}'_{i,j}(t)}{\max(\overline{\mathbf{s}}'_{i,j}(t))} \right)^2} \right) \tag{6.12}$$

Similarly, the smaller the value of $VI_{j,k}$ is, the more consistent the $\mathbf{s}'_{i,j}(t)$ (and $\mathbf{s}_{i,j}(t)$, the original sensory signal) is across all the trials.

The difference between these two indexes is that $CI_{j,k}$ uses H^1 distance while $VI_{j,k}$ uses H^2 distance. One interesting question here is: what is the difference between these two measurements in real application? This will be answered empirically in Chapter VIII using experimental data.

2. Correlation Criterion

Every sensory feature is correlated with the motor drive command sequences within every episode boundary W_k:

$$\rho_{s_j,m,W_k}(z) = E\{(s_i(t+z) - \mu_{s_i,W_k})(m(t) - \mu_{m,W_k})\} \tag{6.13}$$

where $\rho_{s_j,m,W_k}(z)$ is the correlation coefficient between feature sequence $s_i(t)$ and motor sequence $m(t)$ within k-th boundary window W_k with a lag of z steps; $E\{\bullet\}$ is the expectation; μ_{s_i,W_k} and μ_{m,W_k} are the mean values of $s_i(t)$ and $m(t)$ in the W_k window. The higher the correlation, the more salient the features is for the behavior. All the features with a correlation higher than a preset threshold and a consistency index (or variance index) less than a preset threshold are chosen as the salient features for that behavior. If a salient feature happens before the motor command change, this feature is a sensory precursor to the behavior; otherwise, it is a sensory response to the behavior.

After salient feature selection, each episode usually had a salient sensory precursor and a salient sensory response with a behavior in between. This can be expressed as a vector

$$\mathbf{e}_i = \begin{bmatrix} \mathbf{s}_{i0} \\ \mathbf{m}_i \\ \mathbf{s}_{i1} \end{bmatrix} \tag{6.14}$$

where $\mathbf{e}_i$ was the *i*-th episode, $\mathbf{s}_{i0}$ was the sensory precursor before the behavior $\mathbf{m}_i$ happened, and $\mathbf{s}_{i1}$ was the sensory response after behavior $\mathbf{m}_i$ had carried out, which was also usually the sensory precursor for the next episode $\mathbf{e}_{i+1}$. Thus, a sensory-motor sequence can then be expressed as

$$\mathbf{v}(t) = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_P \end{bmatrix} = \begin{bmatrix} \mathbf{s}_{10} & \mathbf{s}_{20} & \cdots & \mathbf{s}_{P0} \\ \mathbf{m}_1 & \mathbf{m}_2 & \cdots & \mathbf{m}_P \\ \mathbf{s}_{11} & \mathbf{s}_{21} & \cdots & \mathbf{s}_{P1} \end{bmatrix} \tag{6.15}$$

where the sequence $\mathbf{v}(t)$ is the sequential play of each episode $\mathbf{e}_i$.

This can also be graphically represented as a event-behavior chain (Figure 40). In this chain, each SMC event is a directional edge and each node is a behavior (episode). In the execution step, the robot performs the task by comparing the current sensory reading with the salient features and decides the next behavior. If the robot starts in the very beginning state, then it carries out the task by following the sequence if all the salient features are detected correctly.

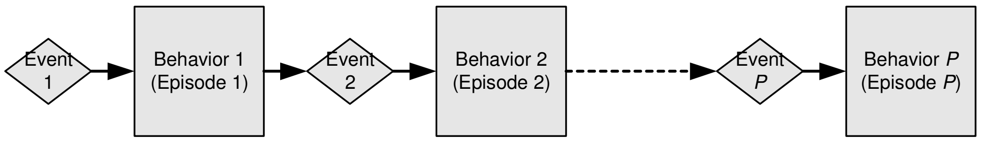


Figure 40: SMC sequence represented as a chain

Performance Evaluation

Though research on navigation (vision-based navigation in particular) has had a history of more than forty years[90], thorough evaluation of feature saliency has been rare. DeSouza and Kak classified indoor navigation into three categories [90]:

- *Map-based navigation*. These are systems that depend on user-created geometric models (usually CAD models) or topological maps of the environment [95, 96].
- *Map-building-based navigation*. These are systems that use sensors to construct their own geometric or topological models of the environment and then use these models for navigation. A recent contribution using this approach was proposed by Thrun [97, 98]. Another recent progress is simultaneous localization and mapping (SLAM) [99].
- *Mapless navigation*. These are systems that use no explicit representation at all about the space in which navigation is to take place, but rather resort to recognizing objects found in the environment or to tracking those objects by generating motions based on visual observation [100, 101].

Both map-based navigation and map-building-based navigation approaches evaluate the system's performance based on the difference between observation and the CAD model, which is the ground truth [97, 98]. The method presented here falls under the mapless navigation category. In these systems, no ground truth (such as a CAD model) exists. Most of these systems used two criteria for evaluation [100-105]:

- Distance between the robot's final position and the target position
- Deviation between the real and learned (or desired) trajectories.

These two criteria were also adopted to evaluate our system's performance. The robot was put back in the same environment and was commanded to navigate to the target position using the learned salient features. While the robot carried out the task automatically, all the sensory data and motor control commands were recorded for performance evaluation purpose. The first criterion measurement was calculated by computing the difference between the final odometry reading and desired target measurement. If the robot's final position was more than 1.5 meters away from the target position, the task was deemed "failed"[1]. The second criterion was calculated by computing the mean and deviation between the odometry sequence and the exemplar odometry sequence[2].

In summary, this chapter presented the hypothesis and the underlying assumptions of our experiments. The hypothesis is that by incorporating SMC principle a mobile robot can discover those features that are salient to the tasks, such as navigation. The salient feature discovery procedure involves three steps: teleoperation, data association, and execution and evaluation.

[1] The robot's main body is about 0.4x0.8 meters. If tires and bumpers included, the size is about 0.7x1.0 meters. Comparing to this size, 1.5 meters is a reasonable measurement to determine success/failure.

[2] As will be discussed n the next Chapter, odometry is accurate for a short distance, which was the case in this experiments. For longer navigation sequences, other measurement sensors (e.g. GPS) should be adopted.

CHAPTER VII

SCOOTER: A MOBILE ROBOT

This chapter will discuss the experimental platform, Scooter, a mobile robot. Section I gives the overview. Section II discusses all the sensors onboard Scooter. Section III introduces the software architecture, and Section IV presents some basic behaviors.

Scooter: Overview

Scooter is the name given to a mobile robot in the Intelligent Robotics Lab, Vanderbilt University. It is an ATRV-Jr made by RWI (Real World Interface, Inc.), see Figure 41.

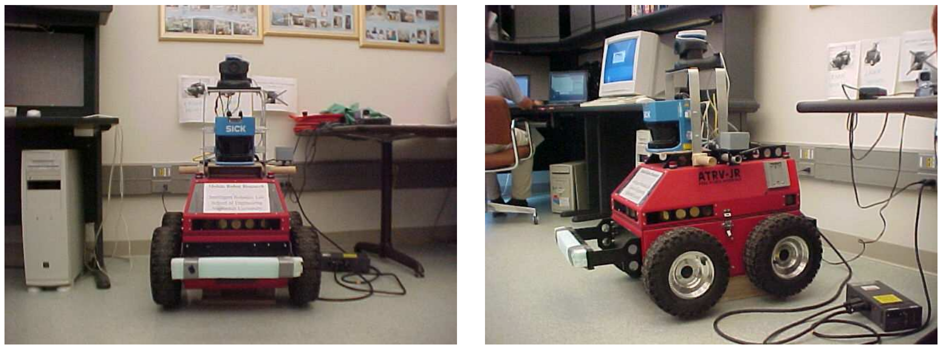


Figure 41: Mobile robot Scooter

This robot is powered by two 12 Volts gel-cell lead batteries. It has a differential drive system (Figure 42a), which uses two electrical motors. Each motor drives two wheels on one side coupled with a timing belt. The benefit of differential drive is that the turning radius is 0, i.e., the robot can spin along its motion center. A mobile robot that has a turning radius of 0 is sometimes called a holonomic robot, while a robot with a turning radius greater than 0 is called a non-holonomic robot [106, 107]. Holonomicity makes this robot suitable to work in confined space (because it does not require extra space for turning.), and it also simplifies the path planning greatly (because the robot can be mathematically treated as a dot with two-degree-of-freedom on a plane, while a non-holonomic must be treated as an object with kinetic- and dynamic-constrains. Please refer to [106, 107].)[3]

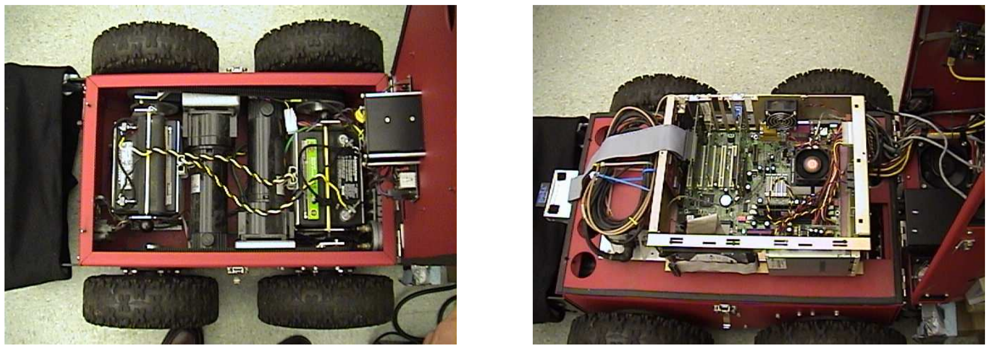
Let $[x(t), y(t)]$ be the robot's Cartesian position; $\theta(t)$ be the heading (orientation); and $[x(t), y(t), \theta(t)]$ is called the robot's pose. The dynamics model of the this kind of robot can be described by the following differential equations [108]:

$$\begin{cases} \dot{x}(t) = v(t)\cos\theta(t) \\ \dot{y}(t) = v(t)\sin\theta(t) \\ \dot{\theta}(t) = \omega(t) \end{cases} \tag{7.1}$$

where $v(t)$ is the ***linear velocity*** and $\omega(t)$ is the ***angular velocity*** of the mobile robot, and they are the control variables. Robot's motion is controlled by giving the right linear and angular velocity profile. Also, the robot can reach a specified pose $[x, y, \theta]$ by controlling these two parameters. This field is usually called trajectory planning or path planning [14].

A standard PC on-board running Linux operating system controls the whole robot (Figure 42b). This PC has an AMD Athlon XP 1.4 G CPU, 256 Megabyte PC133 RAM, and an 18 Gigabyte hard drive. The robot is equipped with a suite of sensors, including pan-tilt-zoom NTSC camera, sonar, LIDAR, odometry, gyroscope, and compass. These sensors will be discussed further in the following section.

[3] The automobiles we drive are nonholonomic, and thus require path planning. A good example of path planning for automobiles is parallel parking.



(a) (b)

Figure 42: Differential drive system (a) and on-board computer (b)

The on-board computer communicates with other computers via wireless Ethernet IEEE802.11b. Programs running on the on-board computer are programmed in a multi-agent software architecture, MOBILITY, which is supplied by RWI. On the user end, we use laptop and workstation computers that run Microsoft Windows 2000 operation systems. On these computers, all programs are written in another multi-agent software architecture, IMA2, which is developed by the Intelligent Robotics Lab, Vanderbilt University [109].

Sensors On Scooter

Scooter is equipped with a suite of sensors, both visual and non-visual, including pan-tilt-zoom NTSC camera, sonar, laser range finder, odometry, DMU, and compass. Each sensor is explained in the following sections.

Vision System

Vision is a useful passive sensing modality. On Scooter, the vision system consists of two parts, a video camera and a frame grabber. The camera, a Sony EVI-D30 pan-tilt-zoom (PTZ) NTSC video camera, is mounted on the top of the robot (Figure 41), and the frame grabber is installed inside the on-board computer (Figure 42b). The specifications of this camera are :

- Image Sensor: 1/3" Color CCD
- Effective Pixels: 768 H x 492 V
- Horizontal Resolution: 460 TV Lines
- Vertical Resolution: 350 TV Lines
- Lens: X12 Power Zoom, ~f=5.4 to 64.8mm, F1.8 to F2.7
- Horizontal Angle of View: 4.3° (tele end) to 48.8° (wide end)
- Vertical Angle of View: 3.2° (telephoto end) to 37.6° (wide angle end)
- Pan/Tilt: Horizontal ± 100° (Maximum speed 80°/sec), Vertical ± 25° (Maximum speed 50°/sec)
- Video Output: RCA pin jack, 1Vp-p, 75 ohm unbalanced
- Control Terminal: RS232C, 8 pin mini DIN, 9600bps, Data 8 bit, Stop 1 bit
- Operating Temperature: 0°C to 40°C
- Dimensions(W/H/D): 142mm x 109mm x 164 mm
- Weight: 1200g

The operation of the camera can be control by a host computer through a RS232 serial port using the VISCA commands, a Sony proprietary communication protocol. The camera parameters---including exposure, shutter speed, gain, and white balance---can either set to automatic or controlled by VISCA commands. In addition, the camera's lens zoom (focal length and zoom speed), pan (pan angle and pan speed) and tilt (tilt angle and tilt speed) are all controlled by VISCA commands. In the experiments, pan and tilt angles were both initialized to 0 and did not change. Also, the initial experiments revealed that the widest horizontal angle of view, 48.8°, was not wide enough, so a RAYNOX 0.5X wide angle screw-on lens was used, and the final horizontal angle of view is about 85°[4].

The frame grabber is made by Hauppauge, and is based on Brook Tree BT848 chip. The v4l (video-for(4)-Linux) protocol is used to fetch the digitized image from the frame grabber to the host computer memory. Due to the computational complexity of image processing and real-time navigation requirement, the digitized image size was set to 160×120 in all the experiments. The processing rate is 3-4 frames/second, depending on the complexity of the image processing.

Sonar Sensors

Sonar is a form of active sensing. It operates on the time-of-flight principle. A high frequency click of sound is emitted that reflects off a nearby surface and returns back. The distance to the surface that reflected the sound is half of the speed of the sound multiplies the delay time:

$$d = \frac{1}{2} v \Delta t \tag{7.2}$$

Scooter is equipped with 17 Polaroid sonar sensors (see Figure 43): two in the back, two on the left and right side, and 13 in the front. The range of these sensors is from about 0.5 meters to 10 meters. The main advantage of sonar sensors is its low cost. However, due to the nature of sonar sensors (cross talk, specular reflection, etc) [110], the readings from these sensors are neither reliable nor accurate.

[4] Due to equipment limitation, we cannot measure the horizontal angle of view accurately. This measurement is however unimportant to our experiments.

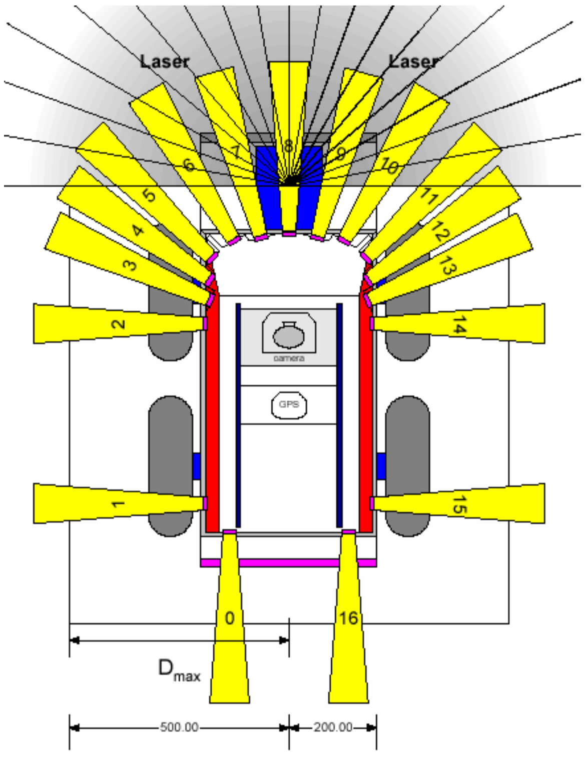


Figure 43: Distribution of sonar and laser sensors on Scooter

Laser Range Finder

A laser range finder is also known as laser radar or lidar. It is another form of active sensor and, like sonar, is also based on the time-of-flight principle. An extremely short pulse of light (infrared laser beam) is transmitted towards an object. Part of the light is reflected back later. The distance to the object is half of the speed of the light multiplied by the delay time. To scan a 2- or 3-dimensional space, a rotating mirror is used to deflect the pulsed light beam to many points in a semi-circle. The precise direction is given by an angular sensor on the mirror. A large number of coordinates measured in this way are put together to form a model of the surrounding area's distance contours.

Scooter is equipped with a SICK LMS-200 laser range finder. The specifications of this sensor are:

- Range: maximum 8 meters
- Angular resolution: 1.00°
- Response time: 13ms
- Measurement resolution: 10 mm
- System error: typical ± 15 mm
- Statistical error standard deviation (1 sigma): typical ± 5 mm
- Power consumption: Approximately 20 W
- Laser protection class: 1 (eye-safe)
- Weight: 4.5 kg (9.9 lb)

The measurement data from the laser range finder are transferred to the onboard computer via a RS-232 serial port. The advantages of laser range finder are its accuracy, reliability, and richness of the information. The foremost drawback is its cost, usually many orders of magnitude over sonar sensors.

Odometry

The drive shafts of Scooter are equipped with optical encoders. Given initial position and heading, the robot's current position and heading can be calculated by integrating the optical encoders' readings. This technique is usually called dead reckoning. For small distances, the odometry reading is reasonably accurate. However, due to the slippage between the wheel and the ground, the odometer's error accumulates over long distance and this error is unbounded.

DMU

Scooter is equipped with an inertial measurement unit, DMU-6X, made by Crossbow Technology, Inc. Crossbow calls it "dynamic measurement unit", or DMU [111]. The DMU-6X is an intelligent six-axis (X, Y, Z, acceleration and Roll, Pitch, Yaw, angular rates) measurement system designed for accurate acceleration and angular rate measurement in dynamic environments. The DMU-6X employs a high performance digital signal processor to provide outputs that are compensated for deterministic error sources within the unit. All six of the DMU-6X sensor elements are micro-machined devices. The three angular rate sensors consist of vibrating plates that utilize the Coriolis force to output angular rate independently of acceleration. The three MEMS accelerometers are surface micro machined silicon devices that use differential capacitance to sense acceleration. The measurements are transferred to the onboard computer via a RS-232 serial port. The specifications of this sensor are:

Performance	**Gyros**	**Accelerometers**
Full scale Ranges	± 50, 100, 150°/sec	± 1, 2, 4, 8, 10, 25G
Bandwidth	DC – 10 Hz	DC – 100 Hz
Bias Stability (25ºC)	$\pm 1° / \text{sec}$	± 10 mG
Bias Stability (-40 to 85)	$\pm 9° / \text{sec}$	± 70 mG
Resolution	0.05º/sec	5 mG
Random walk	0.35º/rtHr	0.15 mG/rtHr

By dead reckoning in software, the robots current X, Y, Z velocity and position and roll, pitch, yaw angle can be computed from the DMU readings. DMU is more accurate and reliable than the odometry, and it does not suffer from the slippage between the wheel and ground. However, it suffers the same drawback as odometry; the error accumulates over long distance and is unbounded due to drifting.

Compass

Scooter is equipped with a KVH-C100 digital compass made by KVH industries. All compasses work on the principle of the earth's magnetic field. However, ordinary compasses are prone to the hard and soft iron magnetic interference of the host platform. KVH-C100 can automatically compensate for the interference and thus provide accurate heading data. The output is transferred to the onboard computer via a RS-232 serial port. The specifications are:

- Accuracy: ±0.5°
- Repeatability: ±0.2°
- Resolution: 0.1°
- Response time: 0.1 to 24 seconds (user selectable)

The earth's magnetic field is often disturbed by other magnetic fields, such as power lines and electric motors. In the outdoor environment, these disturbances are usually scarce and weak, and the compass reading is an accurate and reliable measurement of the robot's heading. However, this disturbance is usually more often in the indoor environment, and the compass reading is not a reliable measurement of the robot's heading when the disturbance is strong. On the other hand, most of the time these disturbance are usually static, and could be used as landmarks for robot navigation.

Software Architecture

A distributed computer hardware system is used to control Scooter. Figure 44 shows the hardware configuration. The system consists of three or more computers, including Scooter onboard machine, a laptop, and one or more desktop computers. Scooter's onboard computer runs Redhat Linux operating system, while all the laptop and desktop computers run Window NT 4.0 or Windows 2000. All computers communicate via IEEE 802.11b wireless Ethernet.

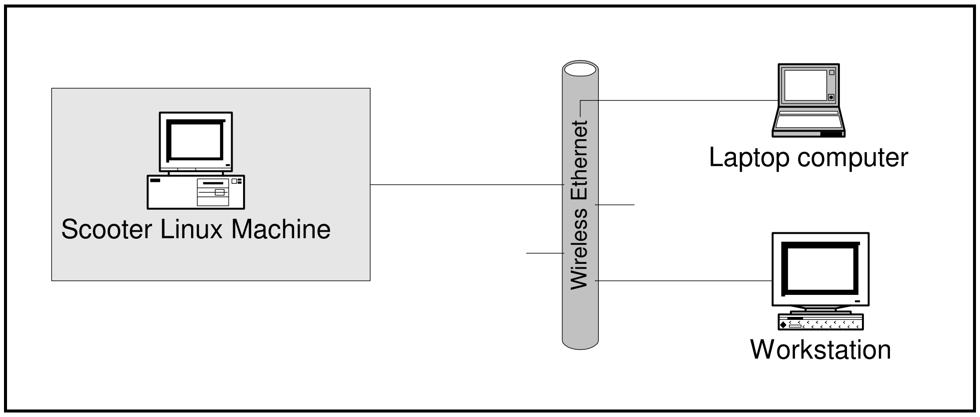


Figure 44: Scooter computer hardware configuration

Due to this distributed hardware system, a distributed software architecture is required to control the robot, and multi-agent system is adopted. Multi-agent systems are becoming an increasingly prevalent paradigm in robotics research, especially in mobile robotics researches. This is because such systems offer numerous advantages over single threaded systems, including increased flexibility, fault-tolerance, and reusability. On the other hand, they also introduce new problems due to the complexity and sophistication of agent interactions.

On Scooter, RWI supplies a distributed software system called MOBILITY, which runs on the onboard Linux computer. On the user end, another distributed software system, IMA, runs on all the Windows computers. Both software frameworks uses software agent-based mechanisms for sharing data and knowledge [112]. Both software frameworks were developed by the same person, Todd Pack, thus they share many similar structures and features.

MOBILITY

MOBILITY is a distributed multi-agent system software framework developed by RWI. It uses CORBA (Common Object Request Broker Architecture) as the underlining communication mechanism, thus it supports many languages across many computing platforms. It consists of an object model for robot software (Robot Object Model) and an object-oriented class framework (Mobility Class Framework) to simplify code development.

The Robot Object Model defines a robot system as a distributed, hierarchically organized set of objects. Each object is a separate unit of software with an identity, interfaces, and state. Objects represent abstractions of whole robots, sensors, actuators, behaviors, perceptual processes, and data storage. Objects provide a flexible model of a robot system that can be reconfigured as new hardware, new algorithms, and new applications are developed. The Mobility Class Framework mirrors the Robot Object Model and handles much of the routine work involved in programming robot software. Be deriving a new class from the Mobility Class Framework, new sensors, actuators, behaviors, perceptual processes, and data classes can be easily added to the system.

IMA/IMA2

Todd Pack at Intelligent Robotics Lab, Vanderbilt University developed a software architecture called the Intelligent Machine Architecture, or IMA, to be used as the framework for developing intelligent robot control systems [109][5]. This architecture was further enhanced by Roberto Olivares [113] and called IMA2. IMA uses Microsoft DCOM (Distributed Component Object Model) as the underlining communication mechanism, and each component can be written in any COM-compatible language, e.g., Microsoft Visual

[5] Todd Pack later went to work for RWI after graduation and developed MOBILITY.

C++, Visual Basic, and Java. IMA provides a two-level software framework for the development of intelligent machines:

- The ***robot-environment level*** describes the system structure in terms of a group of atomic software agents connected by a set of agent relationships. At this level, IMA defines several classes of atomic agents and describes their primary functions in terms of environmental models, behaviors, or tasks.
- The ***agent-object level*** describes each of the atomic agents and agent relationships as a network of software modules called component objects.

IMA can be used to implement virtually any robot control architecture, from SMPA to behavior-based and hybrid architecture. Moreover, different architectures can be implemented simultaneously within separate agents so that a robot can have reactive agents for fast interaction with the environment and deliberative agents for planning or other supervisory control. Interaction between agents with different architecture is possible because the internal structures are completely independent.

IMA uses a system-level model that is based on primitive agents, that is, each resource, task, or domain element is modeled in software as a primitive agent. These agents are classified into four categories, resource/hardware, behavior/skill, environment, and sequencer agents:

- ***Resource or hardware agents*** interface to the sensors or actuators. These agents provide abstractions of sensor and actuator hardware, as well as low-level algorithm such as servo-control loop (for actuators) or filtering (for sensors). These agents use standard interfaces to exploit the resource agents' resources and to communicate with each other.
- ***Behavior and skill agents*** encapsulate reactive, close-loop processes. These agents process sensory data and control actuators to achieve a certain sensory-motor goal. The

distinction between skills and behaviors is that skills are parameterized at run time (e.g., visual tracking or navigation), while behaviors have their parameters fixed at design time (e.g., avoid-obstacle or collision avoidance).

- *Environment agents* contain mechanisms that process sensory data to update an abstraction of the environment such as wall, blue box, etc. These primitive agents may also support operations such as "move to" or "look at" which invoke and supply parameters to a skill agents.
- *Sequence agents* perform a sequence of operations and often interact with the above three types of agents. These agents encapsulate decision-making capacity for invoking skills, behavior, and environment agents.

Overall Architecture

Overall, all the low-level, real-time critical agents run on the onboard computer and are written in MOBILITY framework using C++ language. All the supervisory agents and user-interface agents are written in IMA2 framework using Microsoft Visual Basic 6.0. Sensory data server agents in MOBILITY communicate with corresponding IMA proxies using TCP/IP sockets. Each sensory data channel sets up a refreshing rate according to its own requirements, that is, each agent runs asynchronously.

A hybrid control architecture was adopted (Figure 45). A behavior-based, subsumption-like architecture is adopted for all low-level functions on Scooter. All these modules, or agents, were written in MOBILITY. Besides sensor data servers and the base drive server, several basic behavior agents are implemented on Scooter:

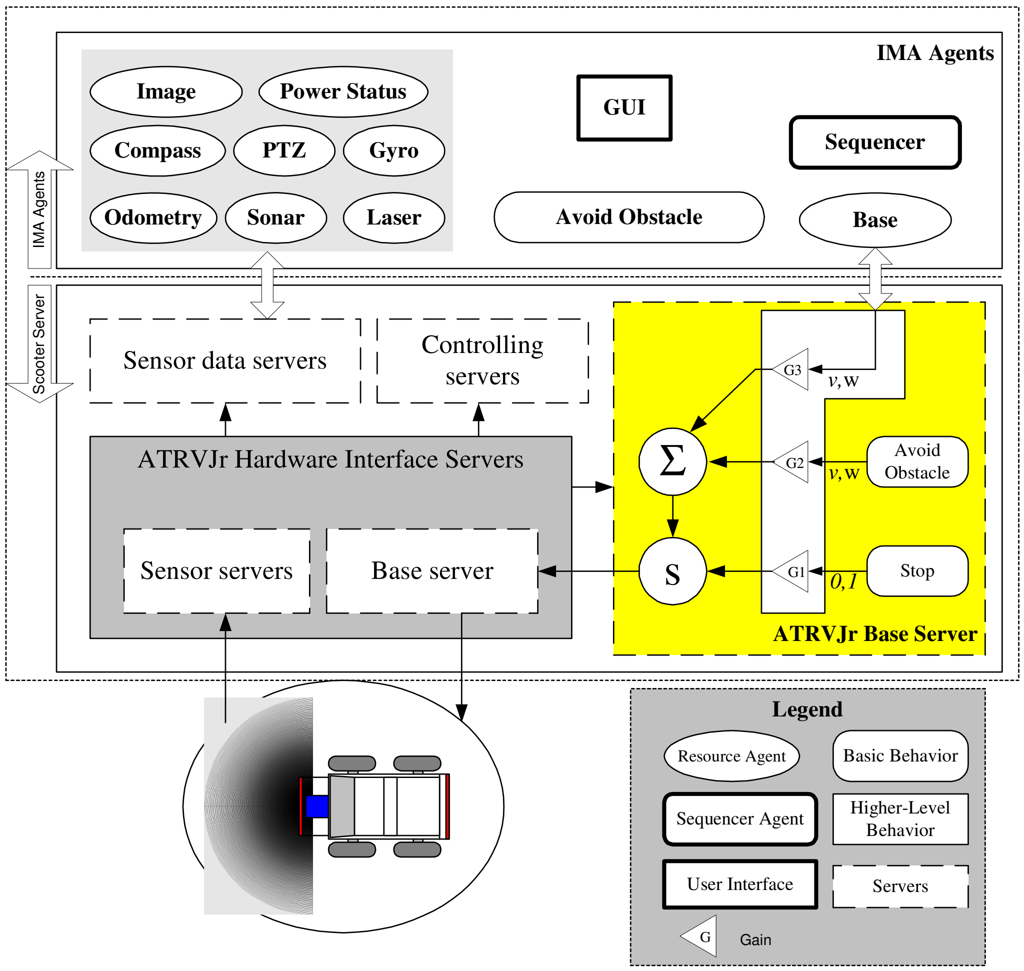


Figure 45: Scooter software architecture

- **Drive**. This primitive agent accepts driving commands for Scooter from other higher level agents and user interface.
- **Emergency stop**. When the minimum distance measurement from range sensors (sonar and laser range finder) is less than 0.5 meters, the robot stops immediately. This is to prevent collision.
- **Obstacle avoidance**. This obstacle avoidance behavior is based on potential field, similar to Arkin's motor schema [14]. Final targets act as attractors and have attraction

force on the robot; while obstacles act as repulsor and have repulsion force on the robot. The vector sum of all these forces is the command the robot should follow.

Also, all the image capturing, processing, and control functions were also implemented on Scooter in the MOBILITY framework due to the bandwidth requirement of the video stream. This will be discussed further in the next chapter. All the user interface control and high-level sequencing agents were implemented on laptop computers. These agents were written in IMA2 framework using Microsoft Visual Basic 6.0. They communicate with corresponding low-level agent (written in MOBILITY) via TCP/IP sockets.

In summary, this chapter presented the experiment platform, Scooter, a mobile robot. Its hardware, sensors, computer hardware, and software were discussed.

CHAPTER VIII

EXPERIMENTAL RESULTS

This chapter will discuss the experimental results. Two sets of experiments, indoor and outdoor, were performed using the mobile robot Scooter. This chapter is organized as follows. Section I gives the overview, Section II discusses the indoor experiments, Section III presents the outdoor experiments, and Section IV discusses a failed approach.

Overview

As presented in Chapter VI, the learning method is a three-step process, teleoperation and data recording, off-line data association, and execution and evaluation. First, the mobile robot is teleoperated in an environment along a path. This is done several times. Each time, the robot runs along essentially the same path with some deviations. All sensory data, both visual and non-visual, along with the motor commands, are saved for the whole trial. In the off-line association phase, signal analysis and image processing techniques constrained by motor commands are used to identify salient features that are relevant to the task. These techniques also identify the SMC events that consistently co-occur with specific motor states or state transitions. Combining these SMC events and behaviors forms a event-behavior chain. Finally, the robot is put back into the same environment somewhere along the path it has been teleoperated and asked to carry out the task it has learned. The performance is then measured.

A mobile robot, Scooter, was used as the experiment platform (cf. Chapter VII). During teleoperation, all sensory reading and drive commands were saved to the hard drive at 3 Hz, i.e., the sampling period is about 333ms. The dimensions of these data were:

- Motor drive commands: Commands have two components, linear velocity and angular velocity.
- Video frame: Video frames are captured in 24-bit color (RGB) with the image size 160×120, totally 160×120×3=57,600 bytes/frame.
- Sonar: Scooter is equipped with a total of 17 sensors, each providing one distance measurement.
- Laser: One laser scan gives 181 distance measurements, corresponding to 0^o to 180^o with 1^o increment.
- Odometry: Odometry has three component, $[x, y, \theta]^T$, where $[x, y]^T$ is the position and θ is the heading.
- DMU: DMU readings have 15 components, corresponding to
    - The displacement, velocity, and acceleration along the three axes $[x, y, z]$, and
    - The angular displacement and velocity of the roll, pitch, and yaw angles.
- Compass: Compass outputs one reading, the current absolute heading.

Indoor Experiments

Indoor experiments were performed on the third floor of the Featheringill Hall, Vanderbilt University (Figure 46). There are totally four corners at this floor. Two corners were selected and two subsets of experiments were executed. One subset was performed at the corner outside Room 300. The other subset was performed at the corner outside Room 341. Two routes were carried out at each corner.

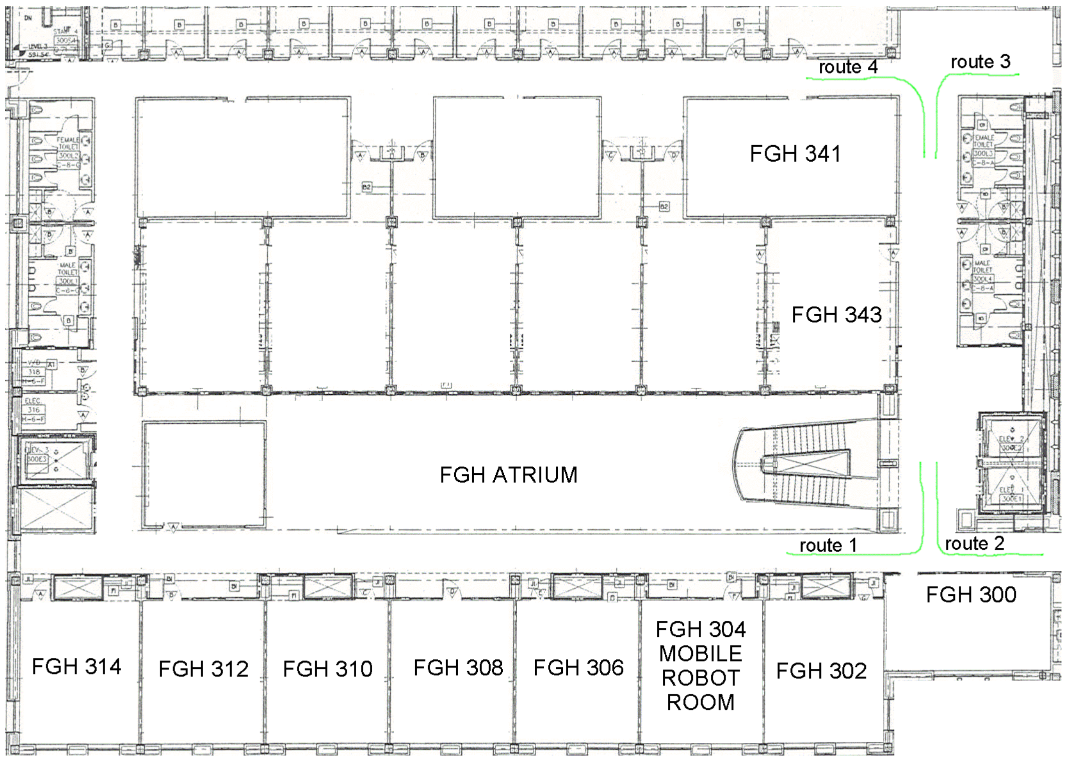


Figure 46: Indoor environment: third floor of Featheringill Hall

First Indoor Experiment Subset

The first indoor experiment subset was to let the robot learn how to navigate the corridor at the intersection outside of Room 300. The goal was to learn two routes, one route starting from outside of Room 302, turning left and reaching the elevator area, the other starting from the elevator area, turning left and reaching to the outside of Room 300. The experiment on the first route will be discussed first.

- **Route 1**

The robot is teleoperated to navigate from outside of FGH302 to the elevator area. This was performed five times. All sensory data and drive commands were saved. Figure 47 shows some sample frames from one of such sequences captured by the vision system. Figure 48 shows the sample laser contour plot for the same sequence. The laser range readings were thresholded at 3 meters (the maximum reading from the laser range finder is 8 meters) to reduce the processing time. Thresholding laser range readings did not have measurable impact, because the size of the robot is 400mm$\times$800mm and the robot moved relatively slow in the experiments, 0.5meters/sec maximum. Figure 49 shows the odometry position $[x, y]$ plot for all the teleoperation sequences. It is obvious from this plot that all five sequences followed essentially the same, yet non-identical, paths.

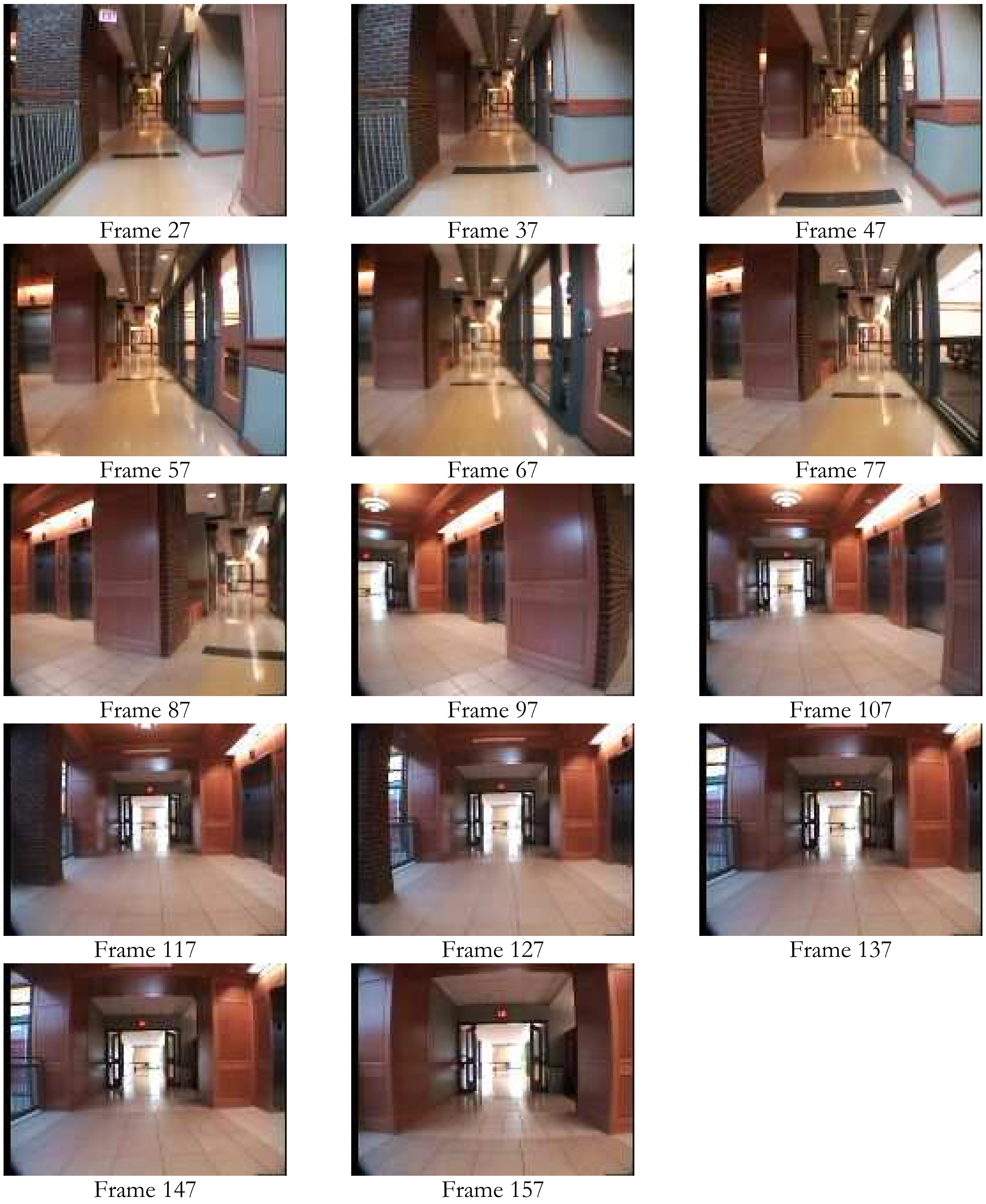


Figure 47: Sample video frames from route 1 teleoperation sequence

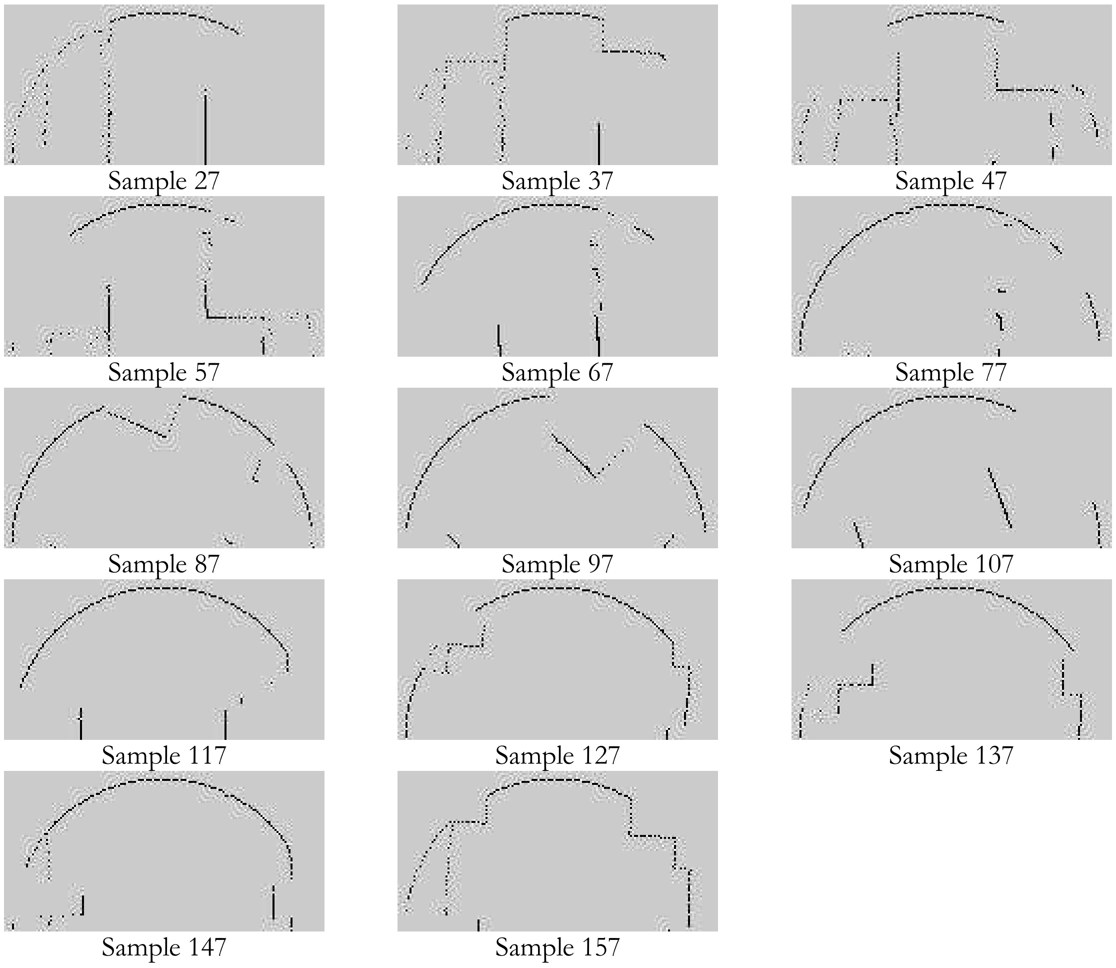


Figure 48: Sample laser plot from the same sequence as in Figure 47

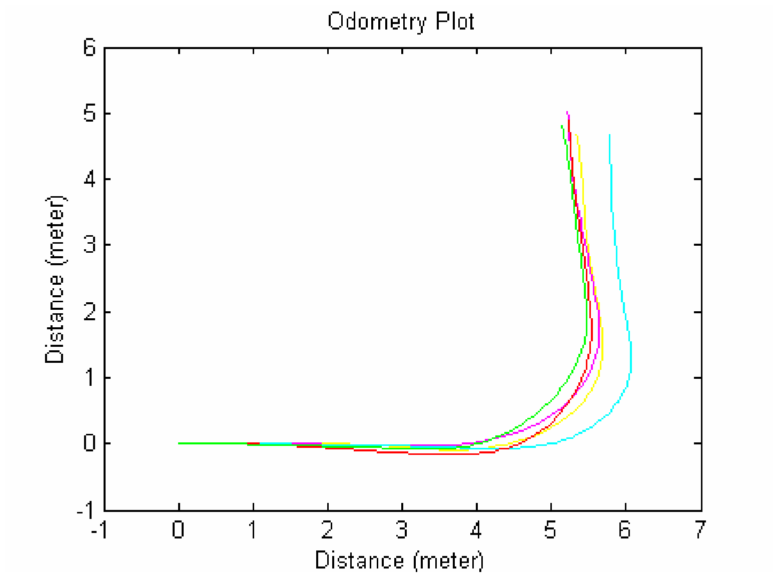


Figure 49: Odometry plot of the route 1 experiment

Offline data association has several steps (cf. Chapter VI). The first step is to cut the sequence into episodes according to the drive commands, then normalize all the sequences through re-sampling. After visual inspection of the drive command sequences, it was determined that there were two significant angular-velocity changes that cut the sequence into three episodes. By this visual inspection, two different thresholds for angular velocity were determined and were used to find the episode boundaries in each sequences. Then each sequence was episode-normalized and the average of these sequences was computed.

Figure 50 shows the episode-normalized angular velocity sequences, and Figure 51 shows the episode-normalized linear velocity sequences. In these plots, the five dashed lines represent the five tele-operation sequences after episode-normalization, while the black solid line represents the average of these normalized five sequences. Two abrupt changes in angular speed, happened at sample 40 and 90, cut the whole sequence into three episodes: sample 0 to 40 (Episode 1), 40 to 90 (Episode 2), and 90 to 120 (Episode 3). These two changes at sample 40 and 90 mark the episode boundaries. From these velocity profiles, it can be concluded that the robot first moved forward (Episode 1), then turned left (Episode 2), and finally moved forward (Episode 3) until it reached the target area.

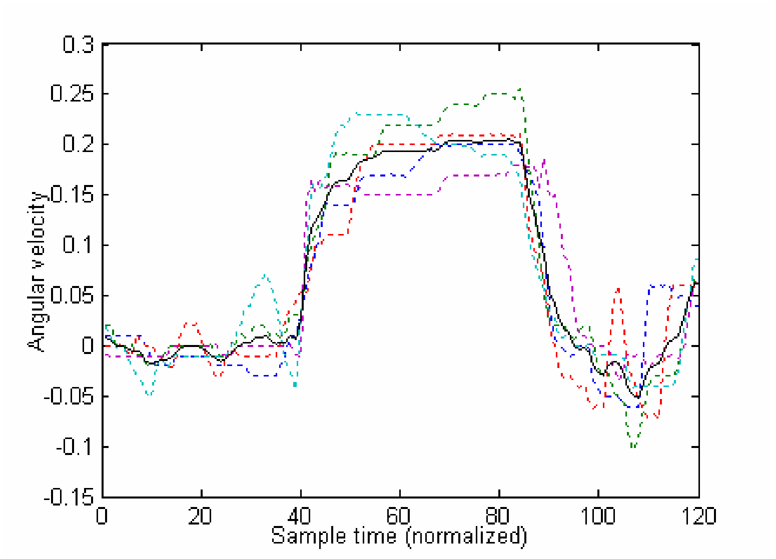


Figure 50: Episode-normalized angular velocity for route 1 sequences

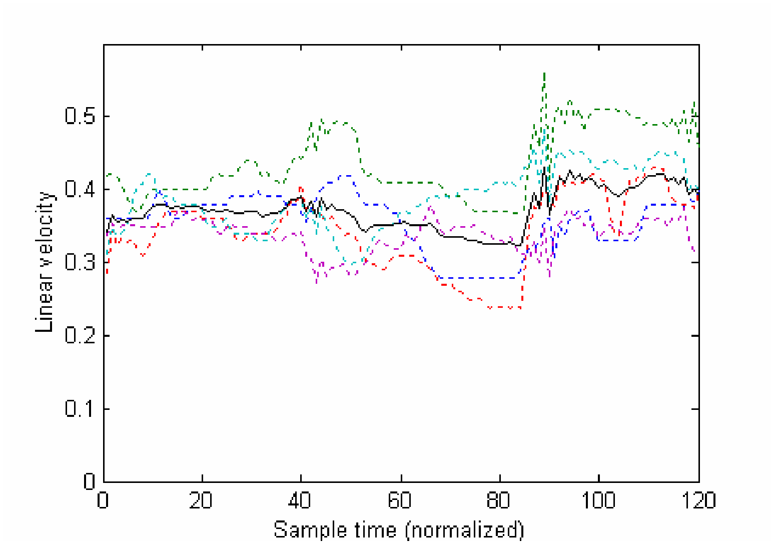
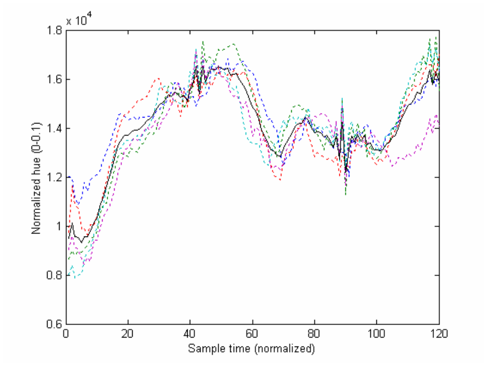
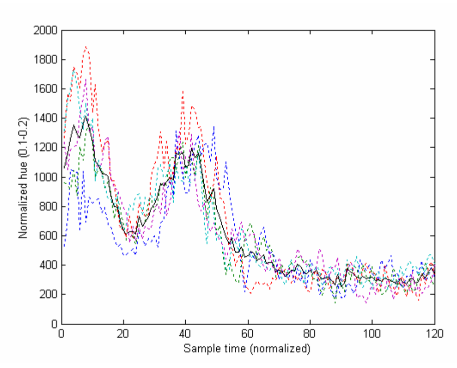


Figure 51: Episode-normalized linear velocity for route 1 sequences

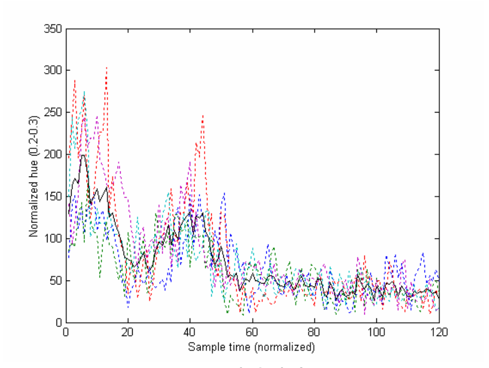
Next procedure is to discover salient features that co-occur with these episode boundaries. This involves several phases. First, many candidate features were extracted from the raw sensory data. For vision, these candidate features were used: hue count, saturation count, brightness count, vertical edge count, and horizontal edge count. For laser, the candidate feature was the length of the longest continuous laser segment. For other sensory modalities the candidate features were their individual components. Using the episode boundary markers, these sensory features were also normalized (cf. Chapter VI). Figure 52 shows the episode-normalized hue count for 10 hue bands, ranging from 0 - 0.1 to 0.9 - 1.0; Figure 53 shows the episode-normalized length of the longest laser segment; Figure 54 shows the episode-normalized compass readings; and Figure 55 shows the episode-normalized relative heading readings from the DMU sensor.



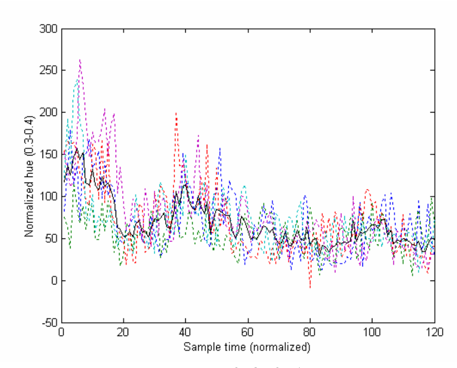
Hue 0-0.1



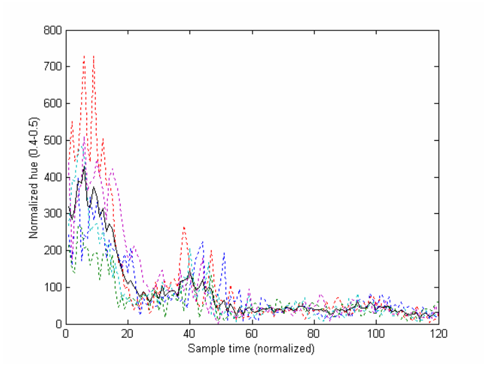
Hue 0.1-0.2



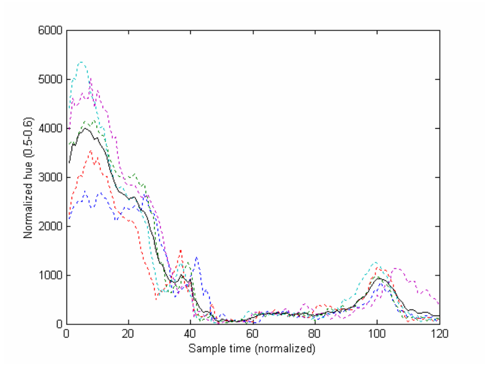
Hue 0.2-0.3



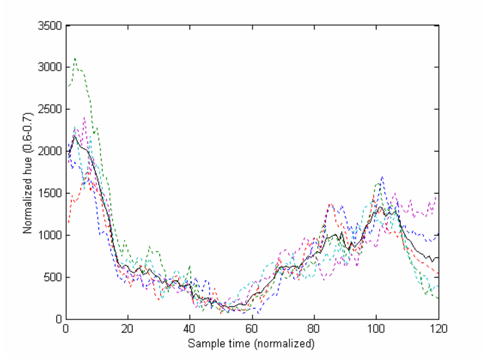
Hue 0.3-0.4



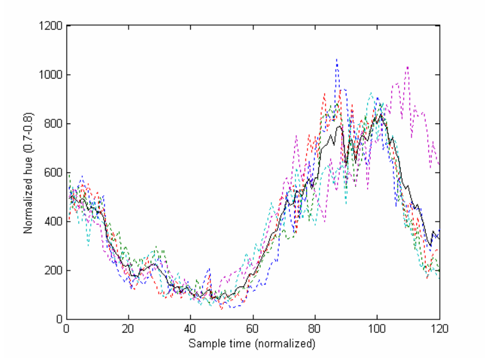
Hue 0.4-0.5



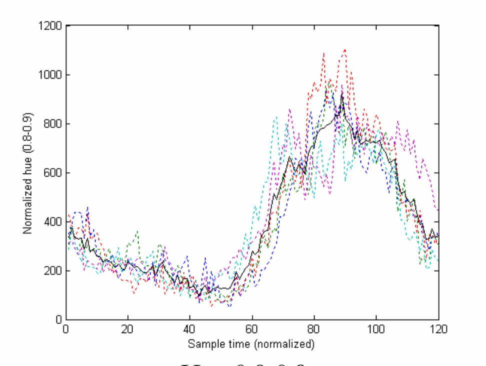
Hue 0.5-0.6



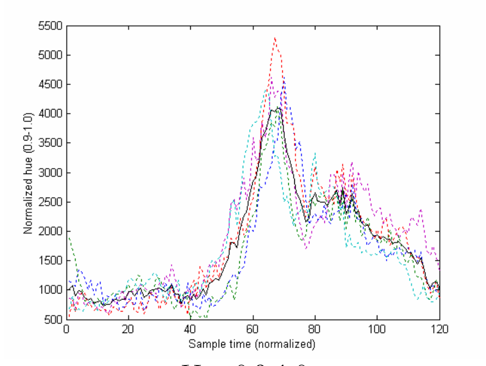
Hue 0.6-0.7



Hue 0.7-0.8



Hue 0.8-0.9



Hue 0.9-1.0

Figure 52: Episode-normalized hue sequences for route 1

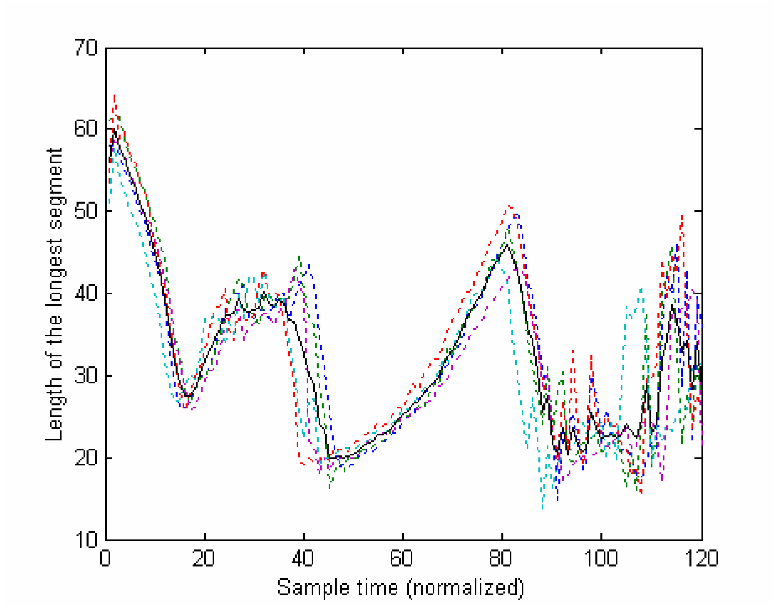


Figure 53: Episode-normalized length of the longest laser segment for route 1

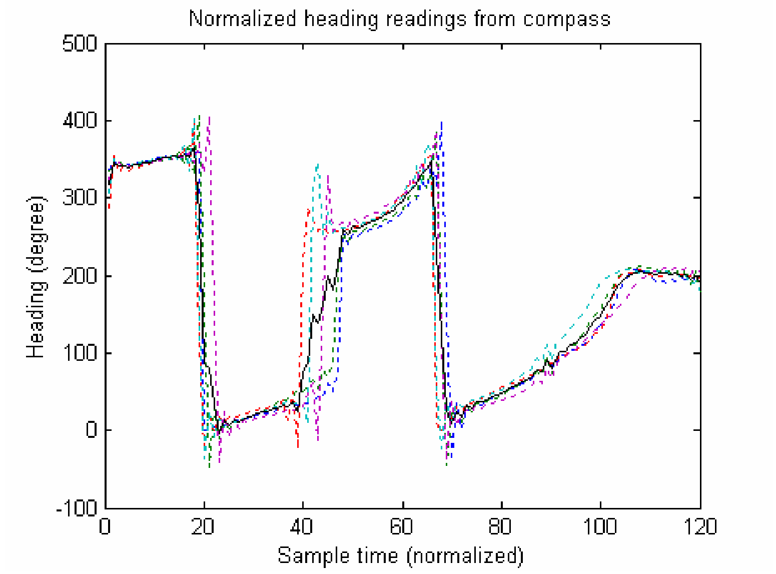


Figure 54: Episode-normalized compass readings for route 1

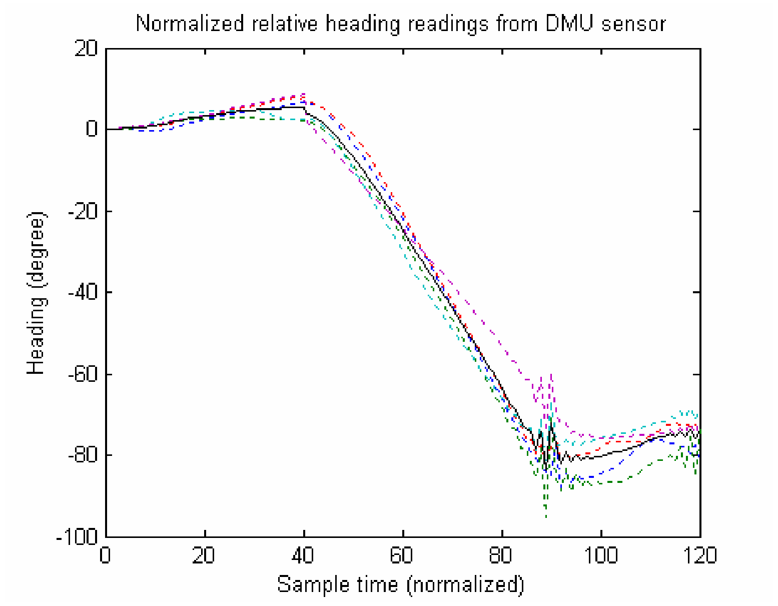


Figure 55: Episode-normalized relative heading from DMU sensor for route 1

Both compass and DMU measure the robot's heading. However, compass is an absolute heading sensor, i.e., the reading is always relative to the North, while the DMU is a relative heading sensor, i.e., the reading is relative to the beginning of the trial. Absolute heading is circular, i.e., $360° = 0°$. This is evident in Figure 54. Around sample time 20 and sample time 70, the compass heading readings changed from 360° to 0°. DMU is a relative and accumulative heading sensor, so the reading from DMU is continuous, no sudden changes from 360° to 0°. Also, as mentioned in the last chapter, compass is sensitive to the magnetic interferences from the environment. This is also obvious in Figure 54. The robot turned totally about 90° during the trial (evident from the trajectory plot (Figure 49)), while the compass reading changed about 150°. DMU apparently gave the correct heading changes (Figure 55). It is also evident that the magnetic interferences in this case were static since the variances between different trials were small. These magnetic interferences can be used as environmental landmarks. One such landmark existed between sample time 40 and 50, when the heading angle increased rapidly.

By using the procedure outlined in Chapter VI, five salient features were selected: hue 0-0.1 count, hue 0.9-1.0 count, length of the longest laser segment, compass, and DMU heading. Other hue counts (hue value between 0.2 to 0.9) and other features, such as brightness count, vertical edge count, and horizontal edge count (all these are visual sensory features), other components of non-visual sensory outputs, were discarded. Figure 56 to Figure 58 show the correlation between three salient features (length of the longest laser segment, compass, and the DMU heading) and the angular velocity command with comments on the graph.

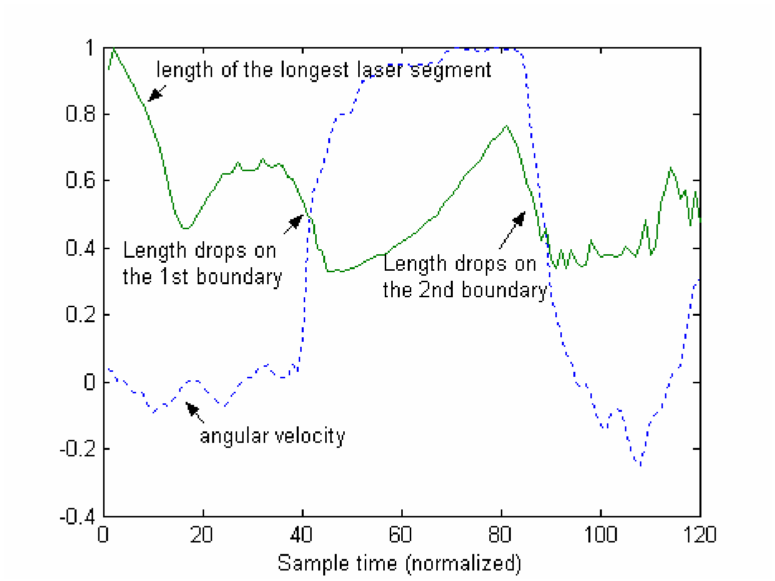


Figure 56: Length of the longest laser segment vs. angular velocity

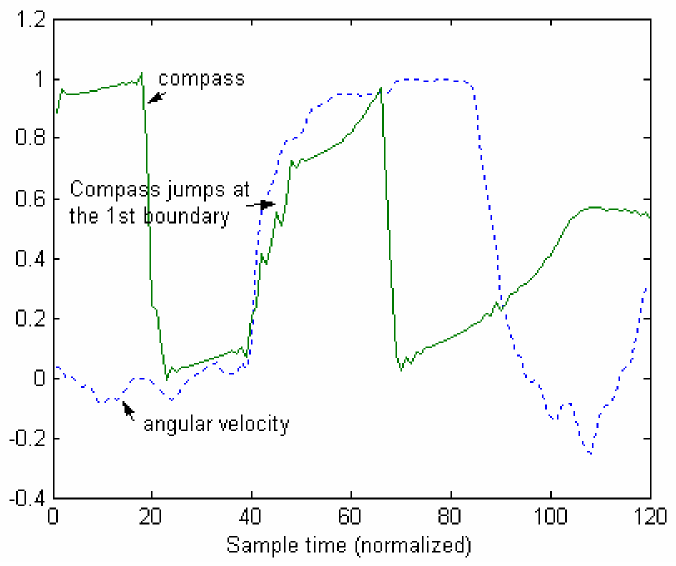


Figure 57: Compass readings vs. angular velocity

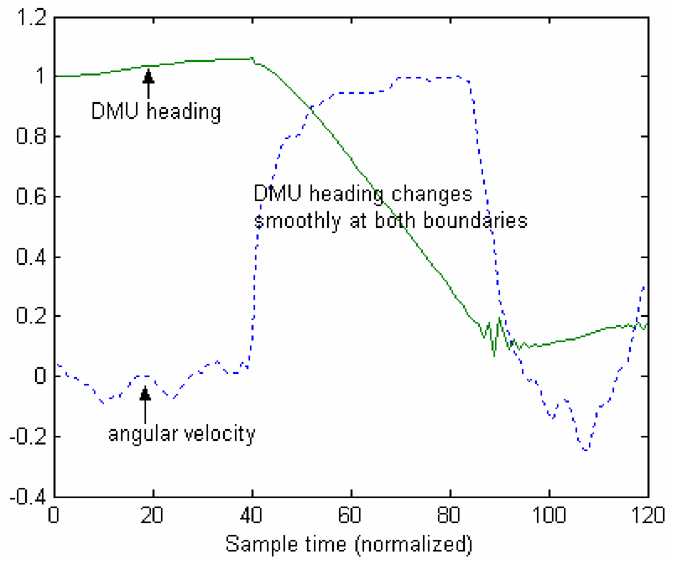


Figure 58: DMU heading readings vs. angular velocity

For performance evaluation, Scooter was put back in the same environment and was commanded to navigate to the target position using the learned salient features. Ten evaluation trials were performed, and the robot succeeded to reach the target in all the trials. Figure 59 shows the trajectory of all the evaluation trials.

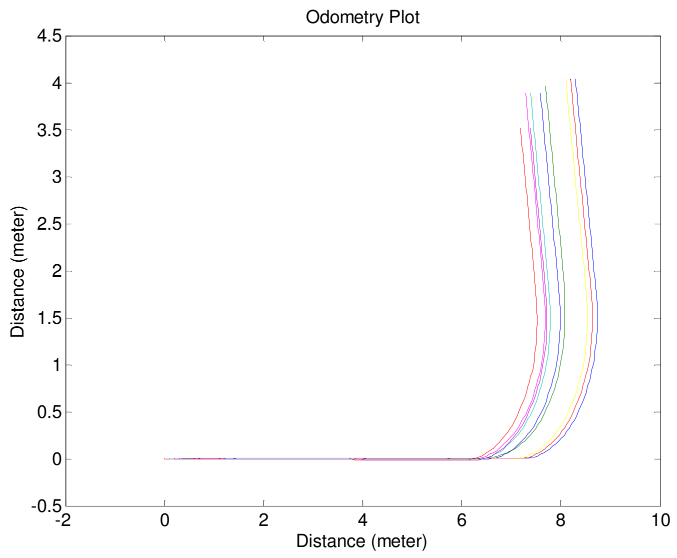


Figure 59: Trajectories of the evaluation trials for route 1

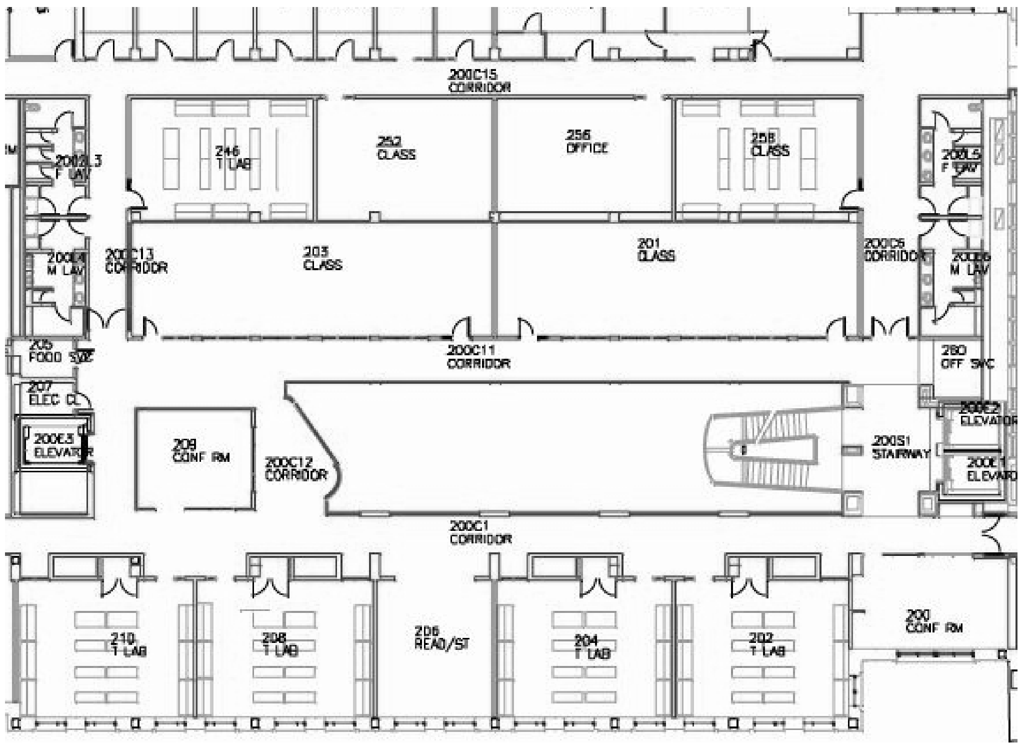


Figure 60: FGH second floor map

The robot was then put in a similar environment, second floor of Featheringill Hall (FGH) (Figure 60). It was commanded to navigate from corridor outside of Room 202 to the elevator area. Again, ten evaluation trials were performed, and the robot succeeded to reach the target in all these trials. This is not surprising, since the environment of this cover at both levels were very similar.

- **Route 2**

On the second route, the robot started from the elevator area, turned left, and reached to the outside of Room 300. Again, the robot was first teleoperated along this route, and all sensory-motor data were recorded. Figure 61 shows some sample frames from one tele-operation trial. Figure 62 shows the odometry position $[x, y]$ plot for all the teleoperation sequences for route 2.

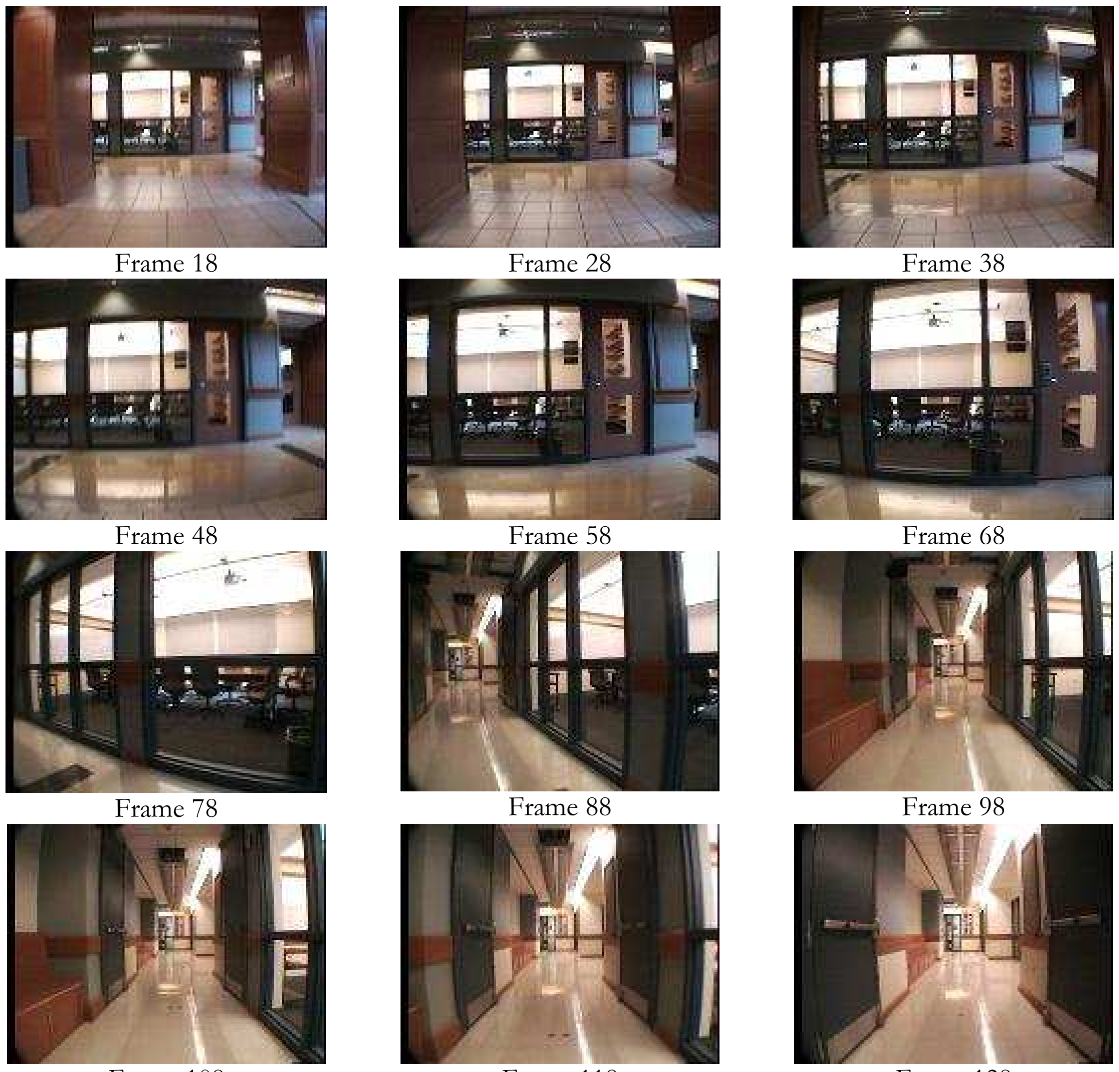


Figure 61: Sample video frames from route 2 sequence

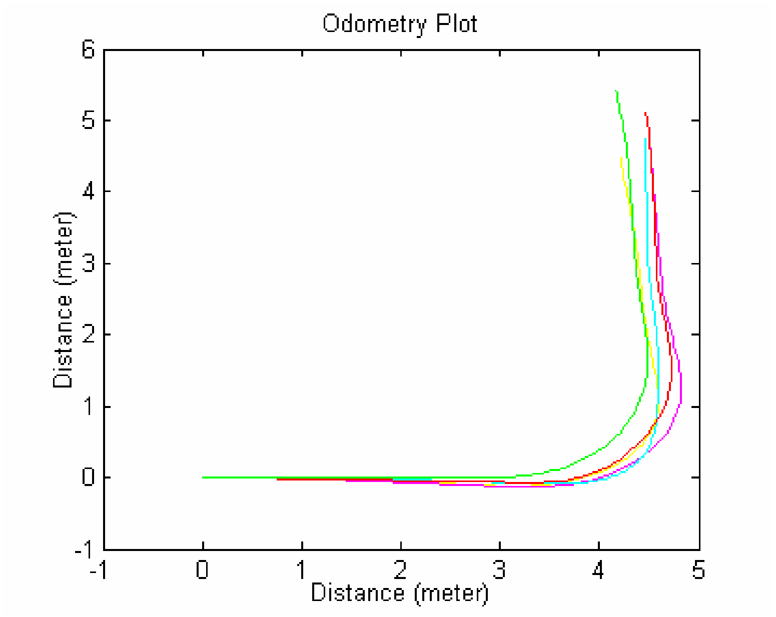


Figure 62: Odometry plot of the route 2 experiment

In the offline data association procedure, first, the motor control sequences were partitioned into episodes. Figure 62 and Figure 63 show the episode-normalized angular and linear velocities, respectively. Two quick changes in the angular speed, happened at sample 40 and 80, cut the sequence into three episodes: sample 0 to 40 (Episode 1), 40 to 80 (Episode 2), and 80 to 90 (Episode 3). From these velocity profiles, it can be concluded that the robot first moved forward, then turned left, and moved forward again until it reached the target area. All candidate feature sequences were then also normalized according to the episode boundaries. Figure 65 shows the episode-normalized hue count for 10 hue bands, ranging from 0 – 0.1 to 0.9 – 1.0. Figure 66, Figure 67, and Figure 68 show the episode-normalized length of the longest laser segment, compass reading, and relative heading from DMU sensor, respectively.

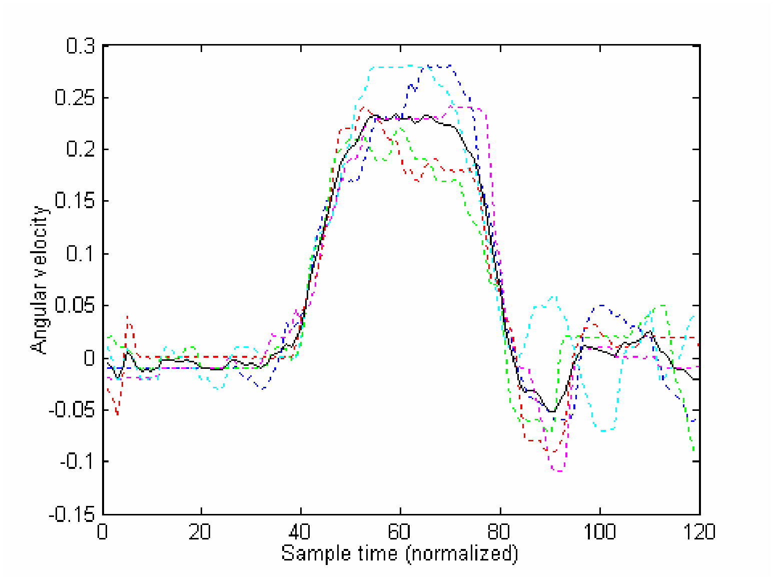


Figure 63: Episode-normalized angular velocity sequence for route 2

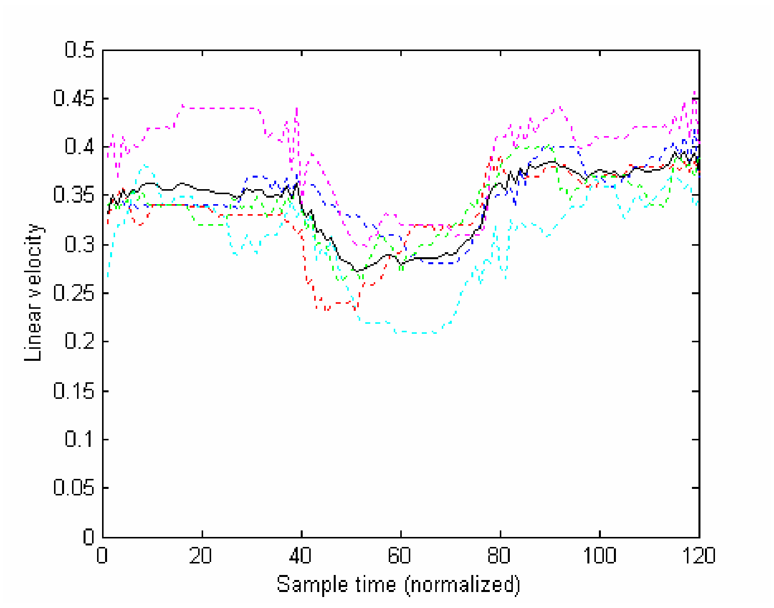
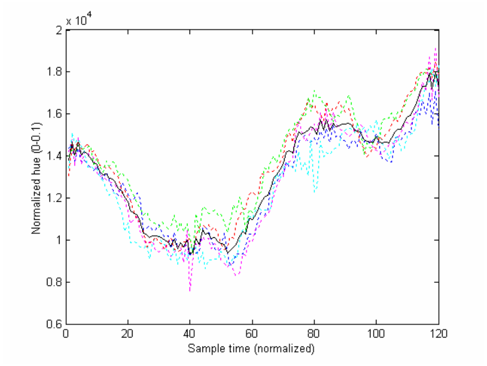
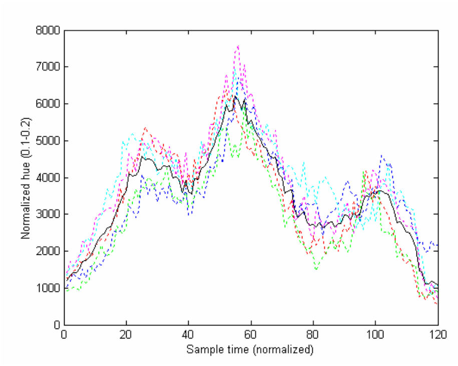


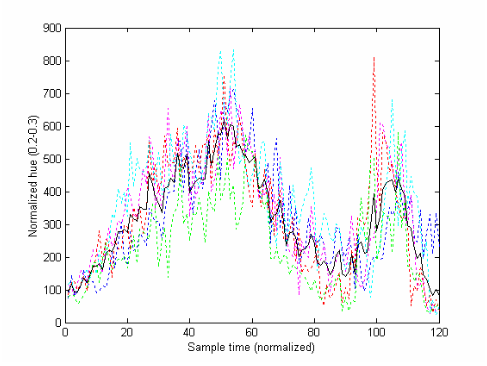
Figure 64: Episode-normalized linear velocity for route 2 experiment



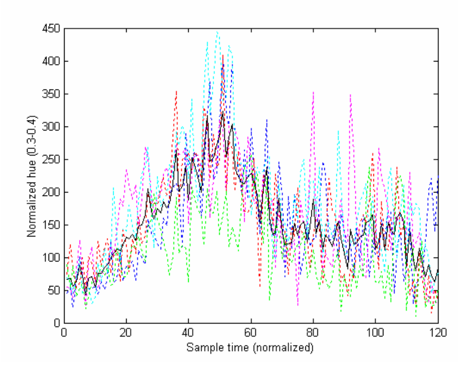
Hue 0-0.1



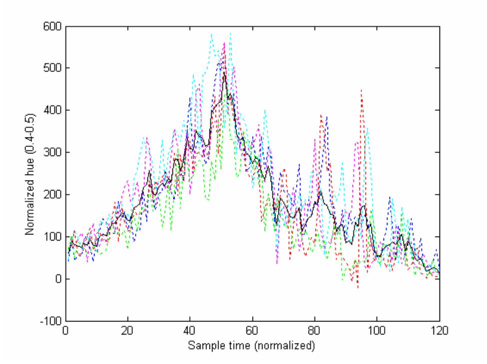
Hue 0.1-0.2



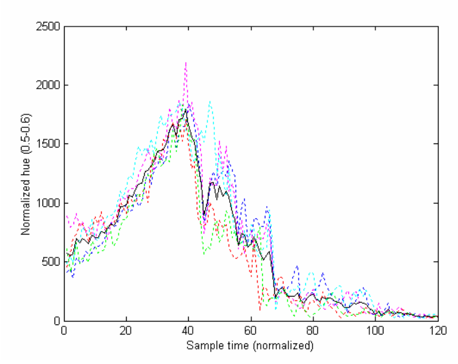
Hue 0.2-0.3



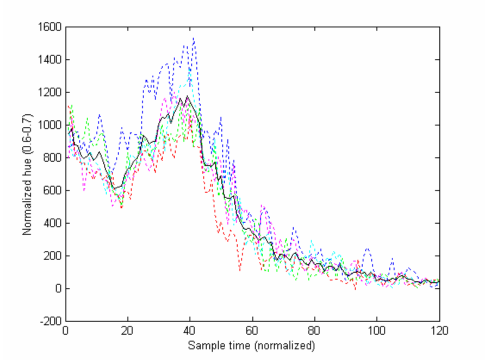
Hue 0.3-0.4



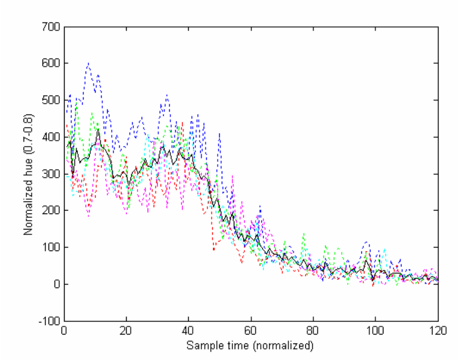
Hue 0.4-0.5



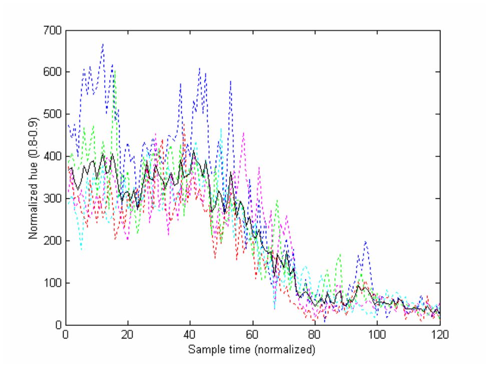
Hue 0.5-0.6



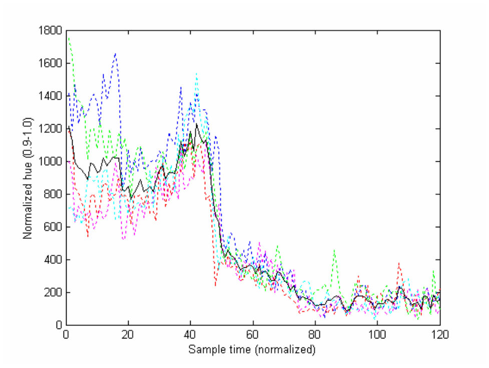
Hue 0.6-0.7



Hue 0.7-0.8



Hue 0.8-0.9



Hue 0.9-1.0

Figure 65: Episode-normalized hue sequences for route 2

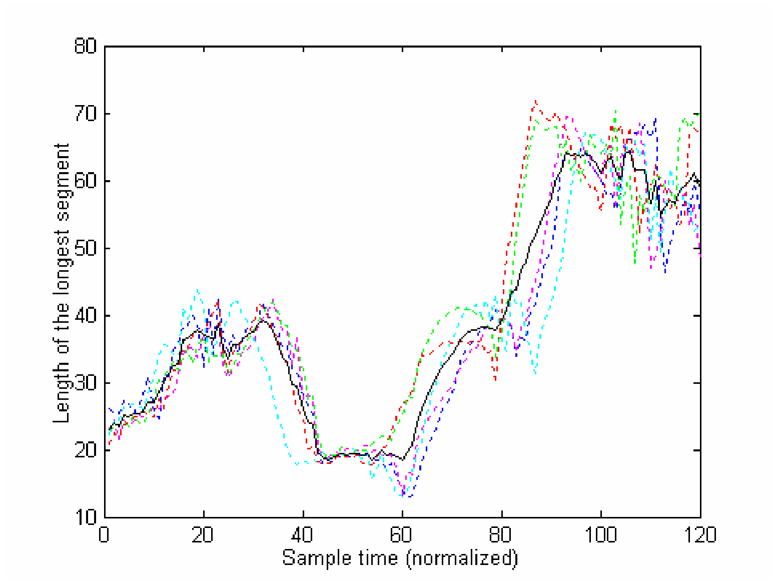


Figure 66: Episode-normalized length of the longest laser segment for route 2

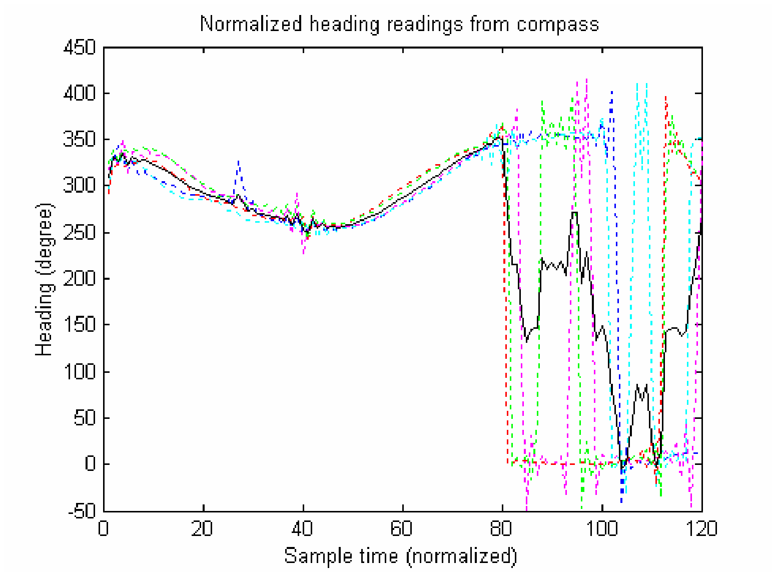


Figure 67: Episode-normalized compass reading for route 2

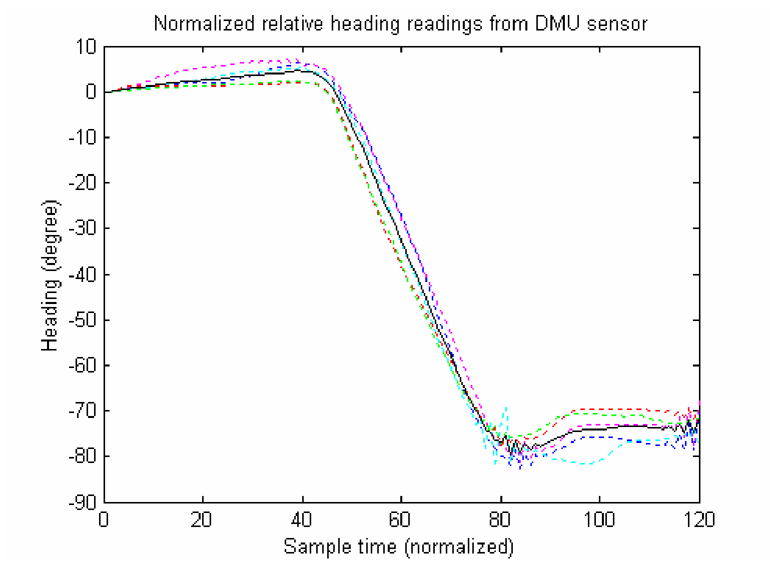


Figure 68: Episode-normalized relative heading from DMU sensor for route 2

Hue 0-0.1 count, length of the longest laser segment, and DMU heading were selected as the salient features, while other features were discarded. Two salient features of route 1 were missing here: hue 0.9-1.0 count and compass. The reason is clear when looking at the normalized hue 0.9-1.0 count sequence (Figure 65) and compass sequence (Figure 67);

both sequences were not consistent between different trials. This also shows the importance of random variance between trials in the teleoperation: this variances is necessary to discover consistent features.

For performance evaluation, Scooter was put back in the same environment and was commanded to navigate to the target position using the learned salient features. Ten evaluation trials were performed, and the robot succeeded to reach the target in all the trials. Figure 69 shows the trajectory of all the evaluation trials.

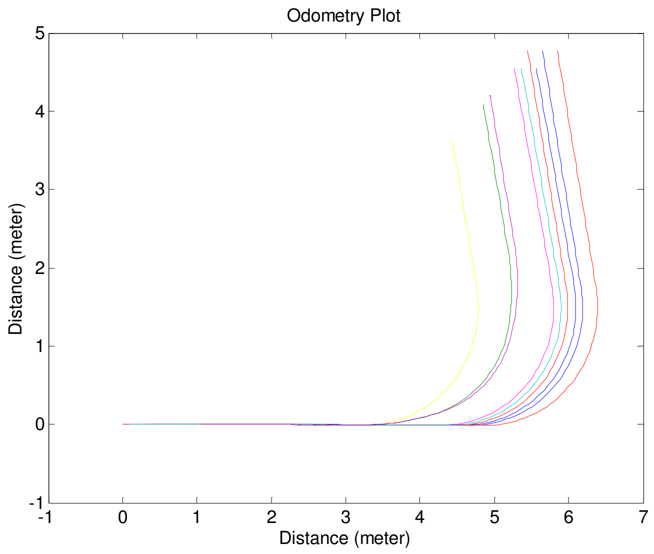


Figure 69: Trajectories of the evaluation trials for route 2

Second Indoor Experiment Subset

The second indoor experiment subset was to let the robot learn how to navigate the corridor at the intersection outside of Room 341 (Figure 46). The goal was to learn two routes, one route (route 3) starting from the outside of Room 343, turning right and reaching the passage, the other (route 4) starting from the outside of Room 341, turning right and reaching the outside of Room 343 area.

- **Route 3**

The robot was first teleoperated along this route five times. Figure 70 shows the odometry plot of these five trials. In the second procedure (offline data association), first the sequences were partitioned into episodes. Figure 71 and Figure 72 shows the episode-normalized angular velocity sequences and linear velocity sequences, respectively. Two quick changes in the angular speed, happened at sample 70 and 110, partitioned the sequence into three episodes: sample 0 to 70 (Episode 1, robot moving forward), 70 to 110 (Episode 2, robot turning right), and 110 to 150 (Episode 3, robot moving forward and reaching the target). Figure 74 shows the episode-normalized length of the longest laser segment, Figure 75 shows the episode-normalized compass reading, and Figure 76 shows the episode-normalized relative heading from DMU sensor.

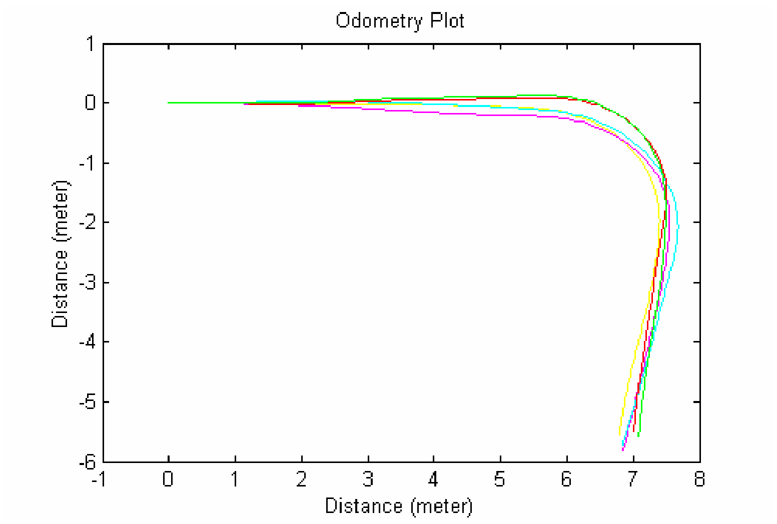


Figure 70: Odometry plot of the route 3 experiments

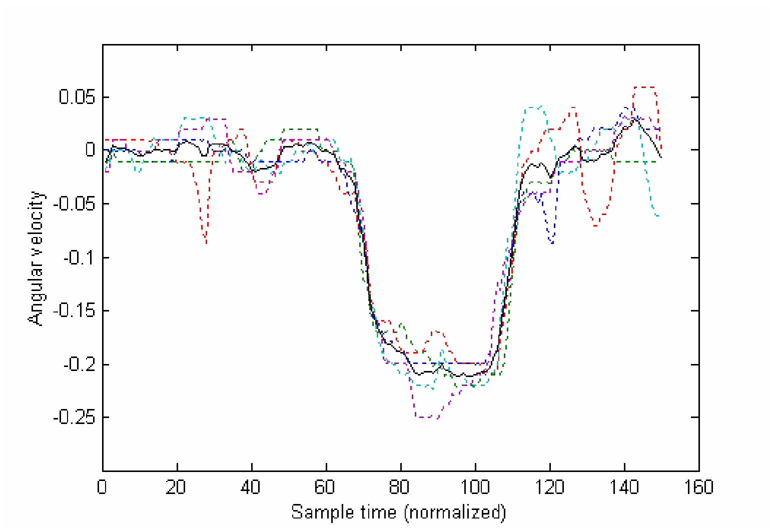


Figure 71: Episode-normalized angular velocity sequence for route 3

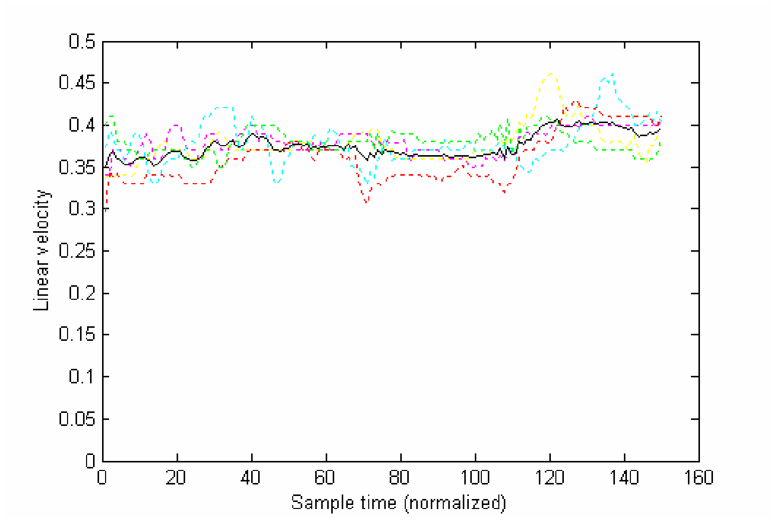
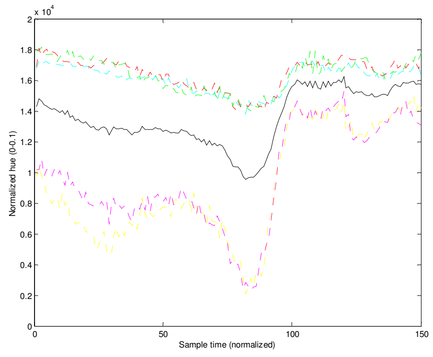
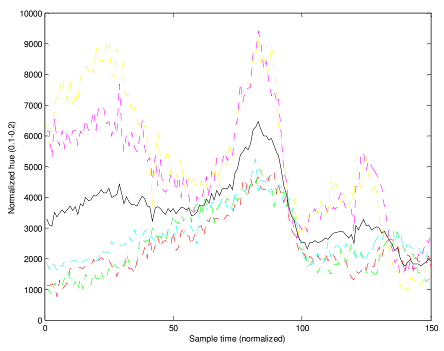


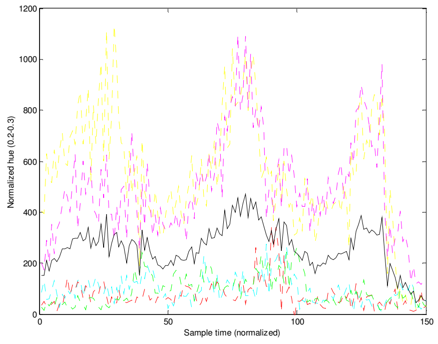
Figure 72: Episode-normalized linear velocity sequence for route 3

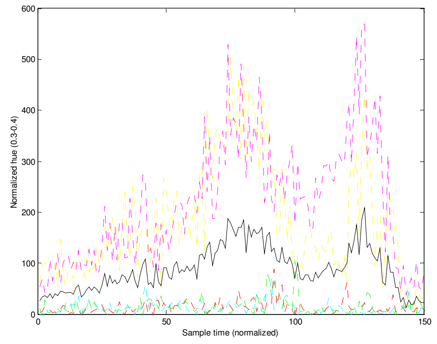




Hue 0-0.1

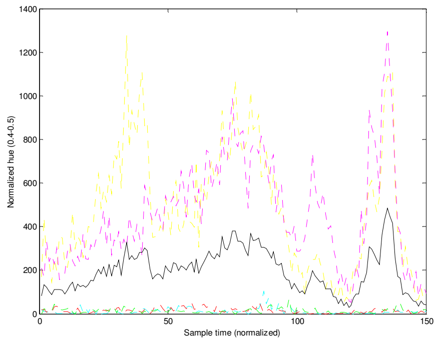
Hue 0.1-0.2

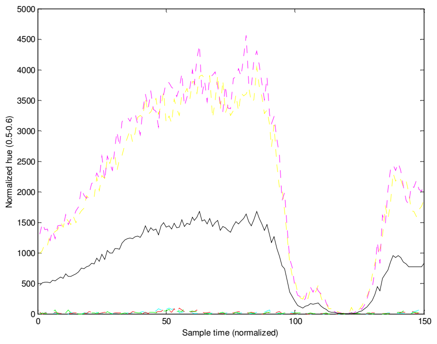




Hue 0.2-0.3

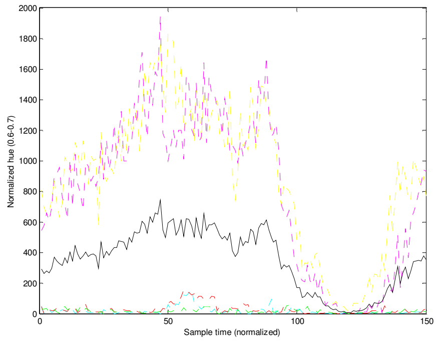
Hue 0.3-0.4

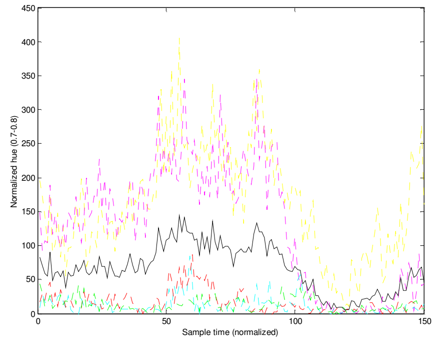




Hue 0.4-0.5

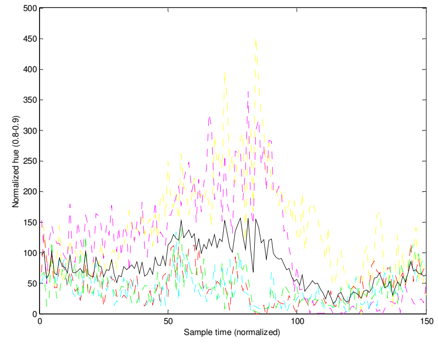
Hue 0.5-0.6

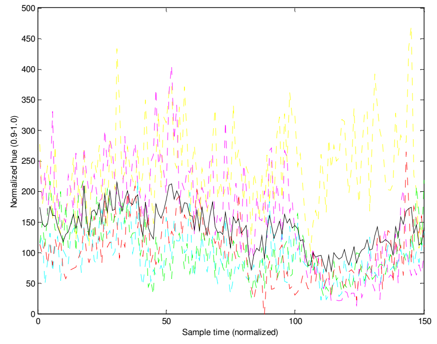




Hue 0.6-0.7

Hue 0.7-0.8





Hue 0.8-0.9

Hue 0.9-1.0

Figure 73: Episode-normalized hue sequences for route 3

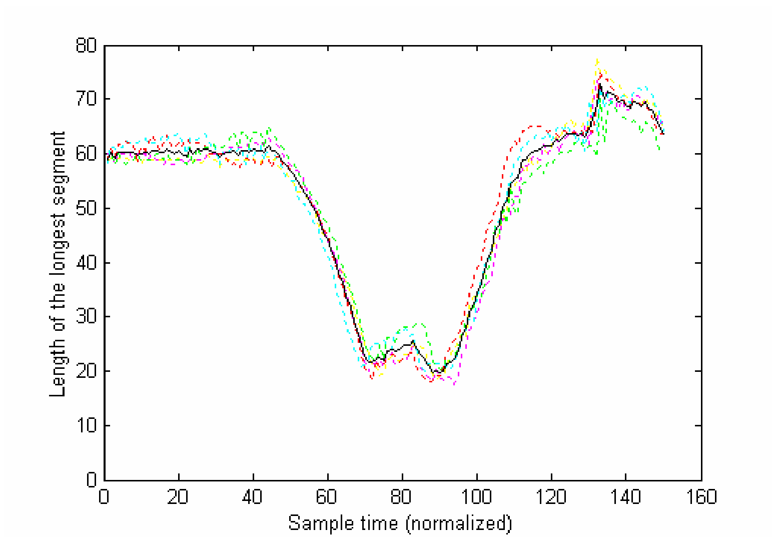


Figure 74: Episode-normalized length of the longest laser segment for route 3

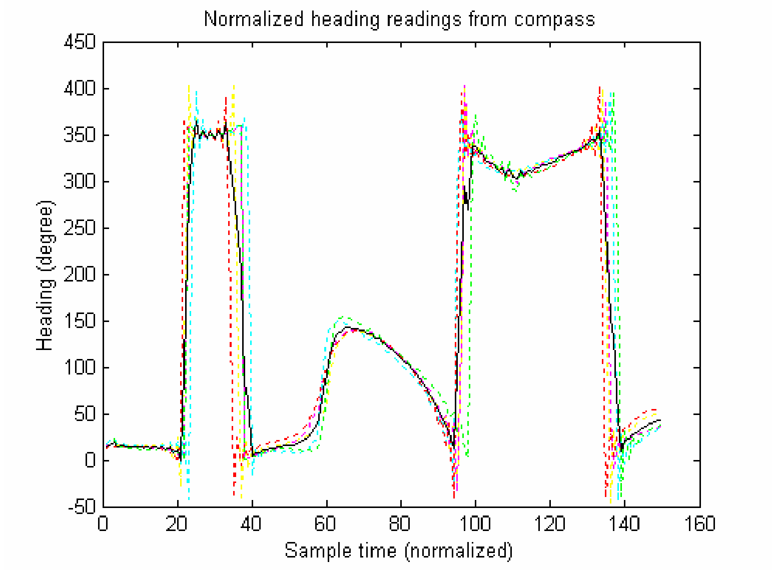


Figure 75: Episode-normalized compass reading for route 3

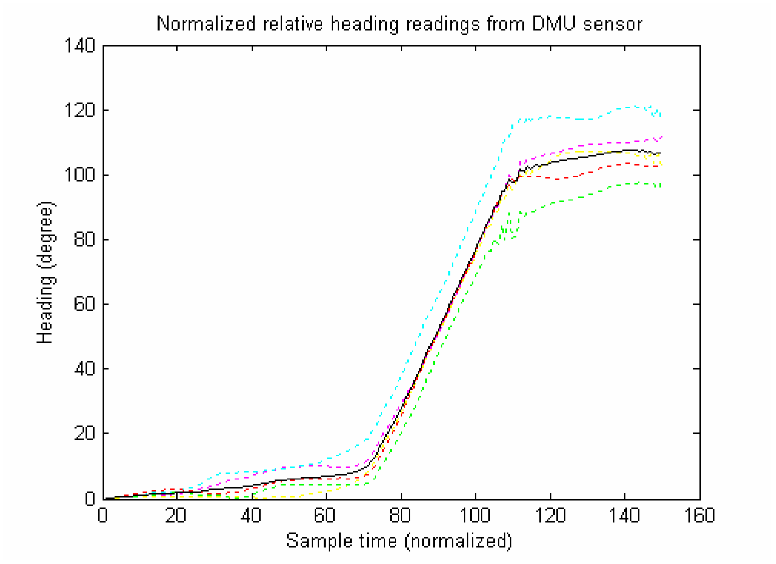


Figure 76: Episode-normalized relative heading from DMU sensor for route 3

One interesting phenomenon here is that all the hue sequences are not consistent (Figure 73). This is due to the fact that there are several big windows at that corner. The lighting condition changes drastically (both intensity and specular reflection from the tile floor) due to the sunlight through the windows. Figure 77 shows two images from different teleoperation trials. They were taken roughly at the same place in the daytime and in the evening. Finding consistent features across such drastic pictures is still a great challenge for computer vision research.

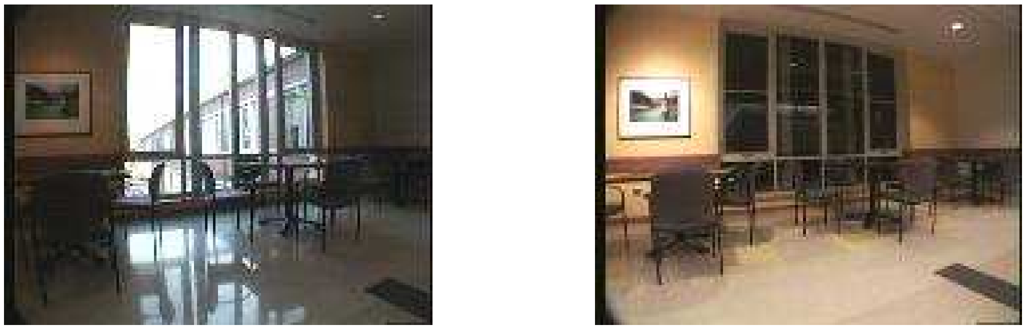

(a) daytime, sunny (b) evening

Figure 77: Sample images from route 3 showing the challenges of computer vision

Only non-visual features were selected for this route: laser, compass, and the DMU heading. For performance evaluation, Scooter was put back in the same environment and was commanded to navigate to the target position using these learned salient features. Ten evaluation trials were performed, and the robot succeeded to reach the target in all the trials. This was due to the fact that this indoor route was static (i.e., it does not vary much), thus the sensor readings were very consistent (cf. the non-visual sensor features, from Figure 74 to Figure 76). Figure 78 shows the trajectory plot of all the evaluation trials.

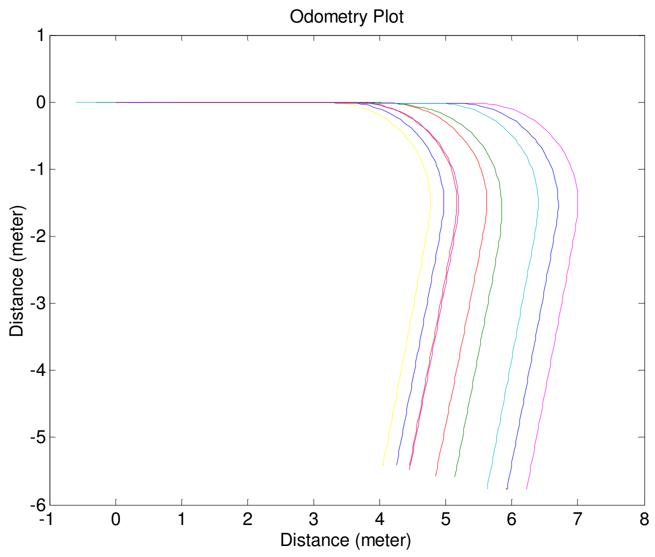


Figure 78: Trajectories of the evaluation trials for route 3

- **Route 4**

Again, the robot was first teleoperated along this route five times. Figure 79 shows the odometry plot of these five trials. In the offline data association phase, first the sequences were partitioned into episodes. Figure 80 and Figure 81 shows the episode-normalized angular velocity sequence and linear velocity sequences, respectively. Two quick changes in the angular speed sequence, happened at sample time 50 and 90, cut the sequence into three episodes: from sample 0 to 50 (Episode 1, robot moving forward), from 50 to 90

(Episode 2, robot turning right), and from 90 to 140 (Episode 3, robot moving forward and reaching the target). Figure 82, Figure 83, and Figure 84 show the episode-normalized length of the longest laser segment, the compass reading, and the DMU heading, respectively.

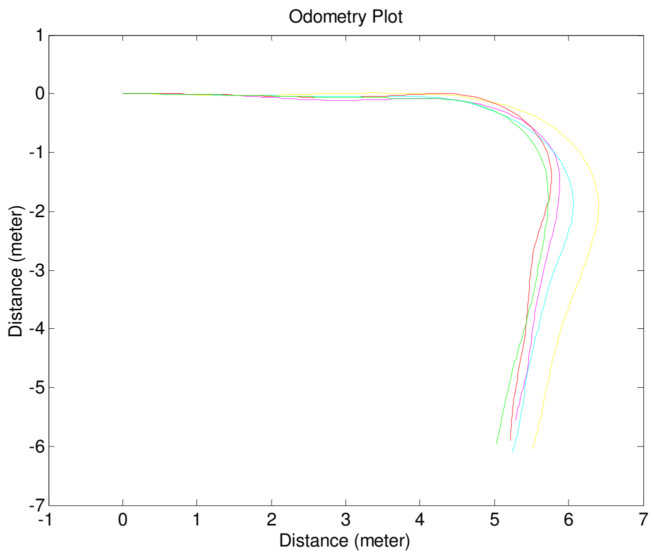


Figure 79: Odometry plot of the route4 experiments

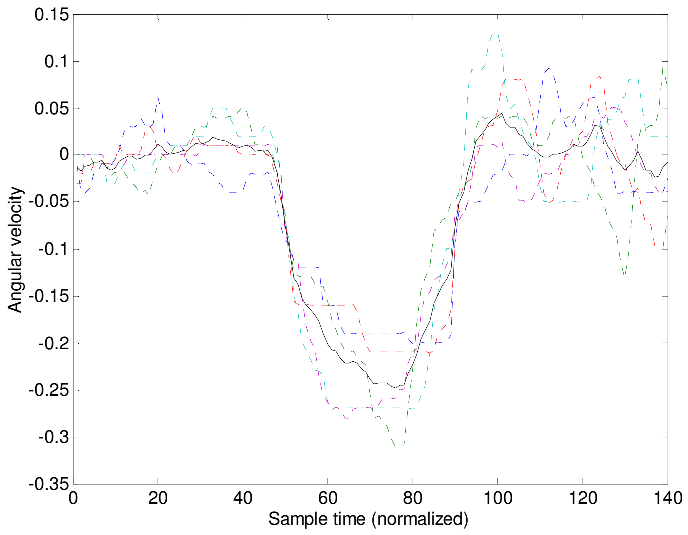


Figure 80: Episode-normalized angular velocity sequence for route 4

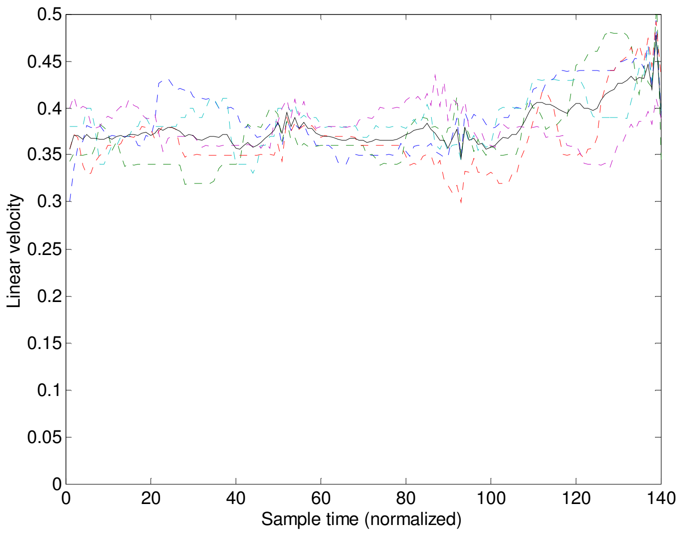


Figure 81: Episode-normalized linear velocity sequence for route 4

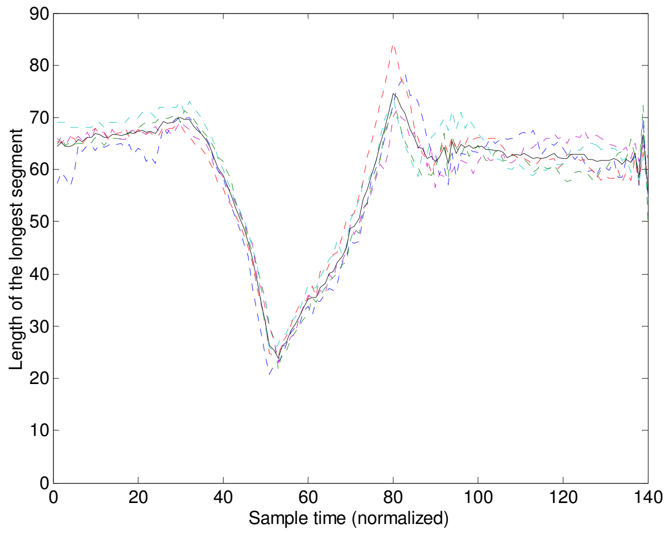


Figure 82: Episode-normalized length of the longest laser segment for route 4

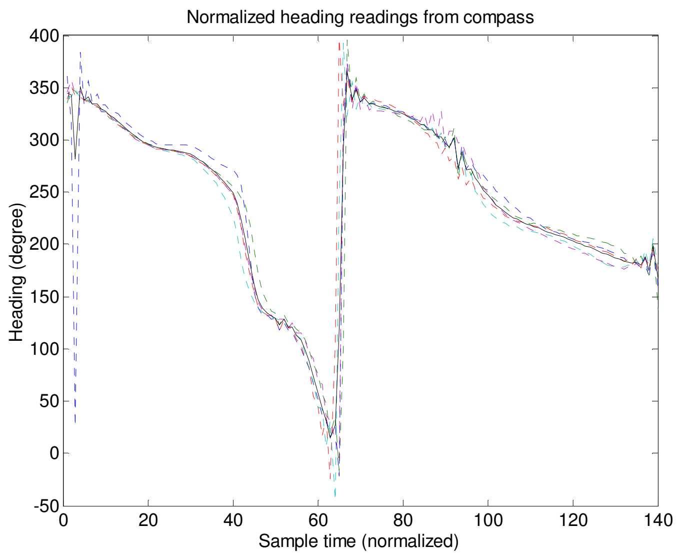


Figure 83: Episode-normalized compass reading for route 4

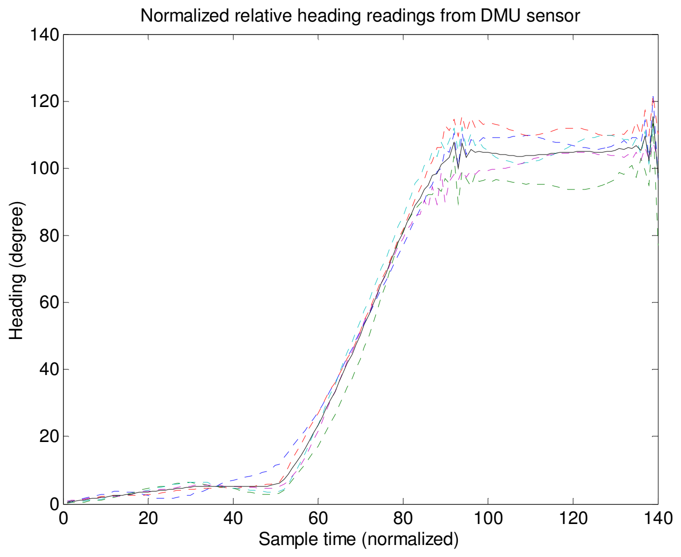


Figure 84: Episode-normalized DMU heading for route 4

Again, due to the big windows on the wall at this corner, no consistent visual features were extracted. On the other hand, all the non-visual features were very consistent, and these were selected as the salient features. For performance evaluation, Scooter was put back in the same environment and was commanded to navigate to the target position using these learned salient features. Ten evaluation trials were performed, and the robot succeeded to reach the target in all the trials.

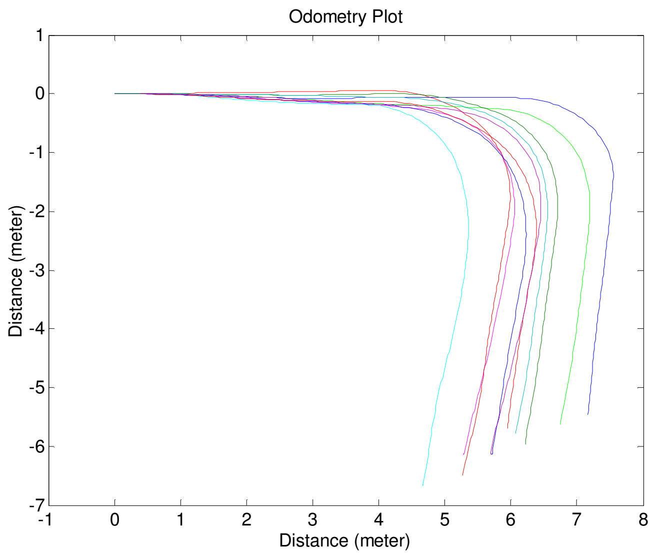
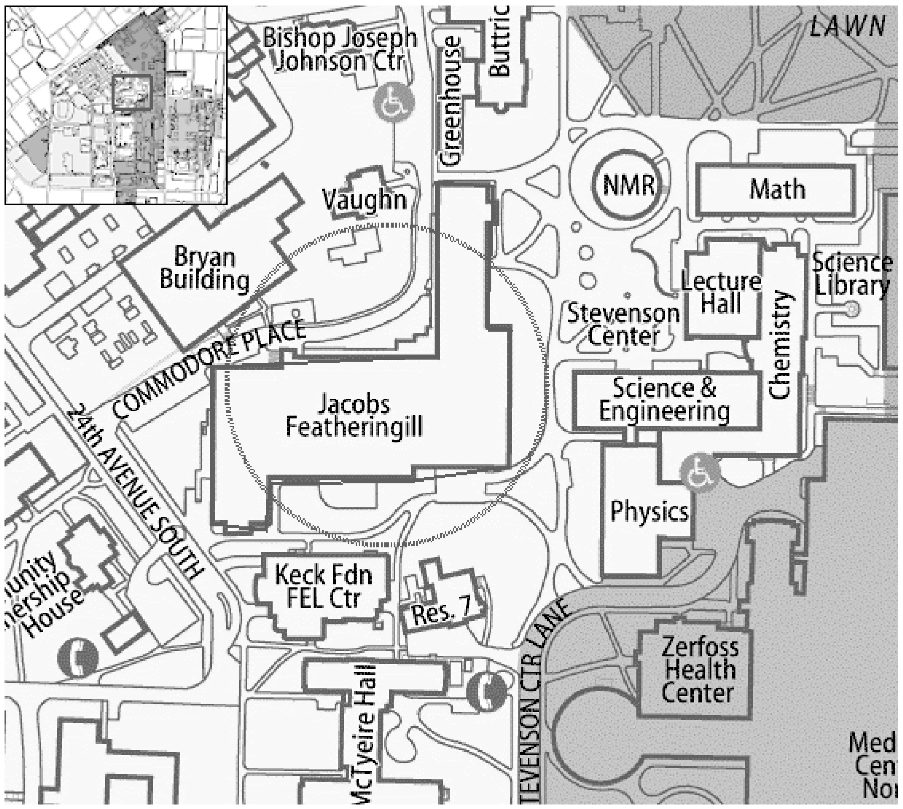


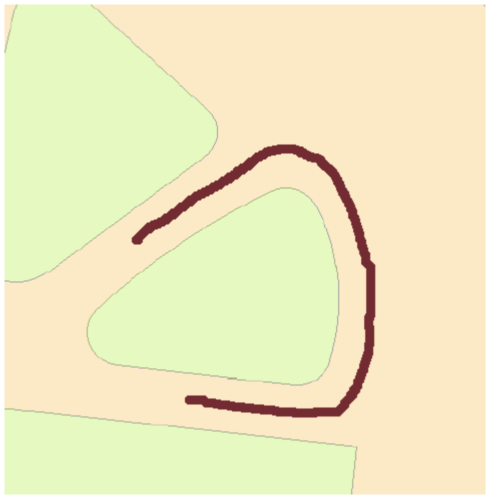
Figure 85: Trajectories of the evaluation trials for route 4

Outdoor Experiments

Outdoor experiments were performed on Vanderbilt campus, in the area between Jacobs/Featheringill Hall and Stevenson Center (Figure 86). The goal was to let the robot learn how to navigate a sidewalk in this area. First, the robot was teleoperated along this sidewalk five times. Figure 87 shows some sample video frames from one trial. Figure 88 shows the trajectory plot of the five trials according to the odometry recordings.



(a)



(b)

Figure 86: Outdoor experiments environment
(a) overall map (b) detailed map with trajectory illustrated

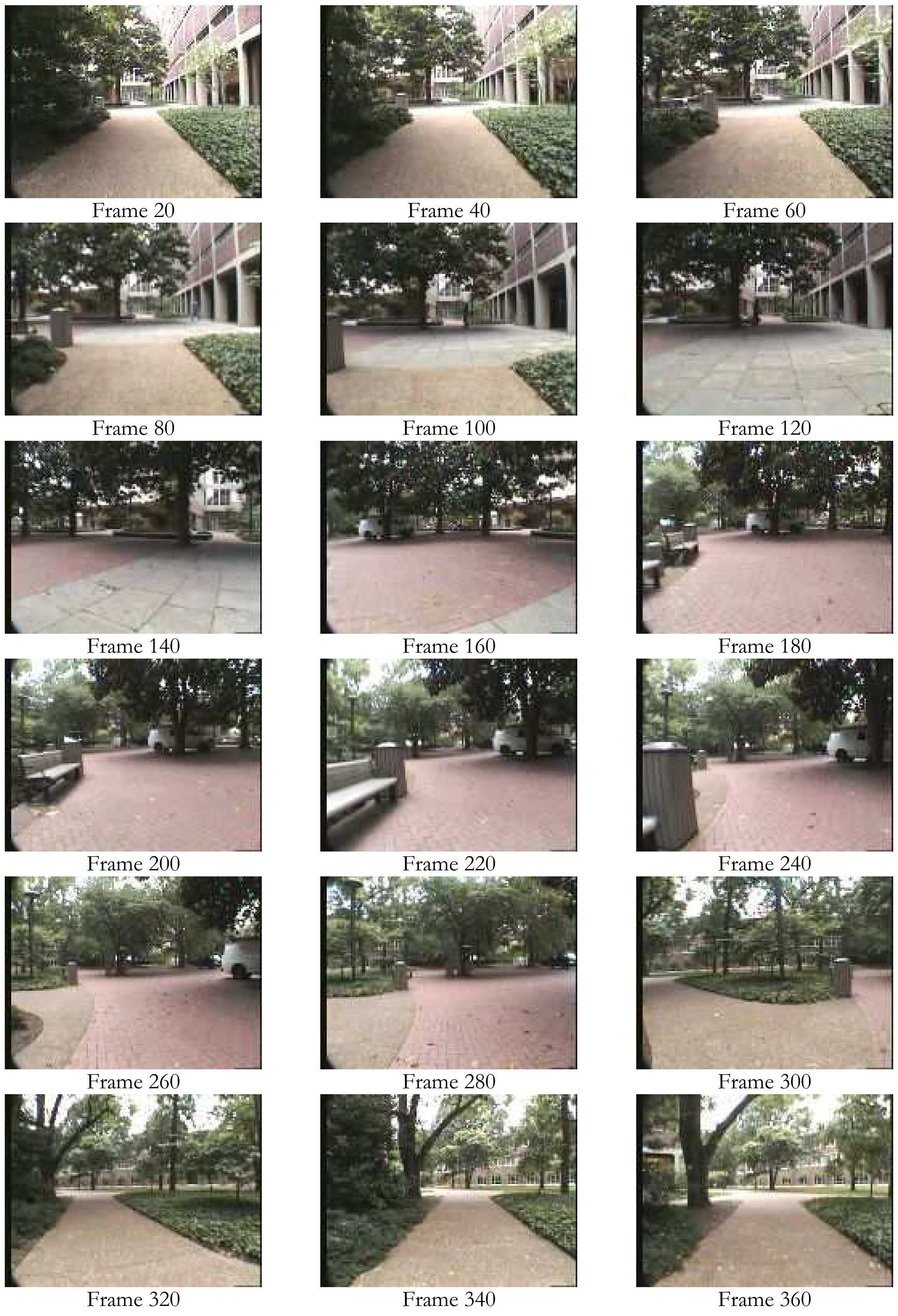


Figure 87: Sample video frames from outdoor teleoperation sequence

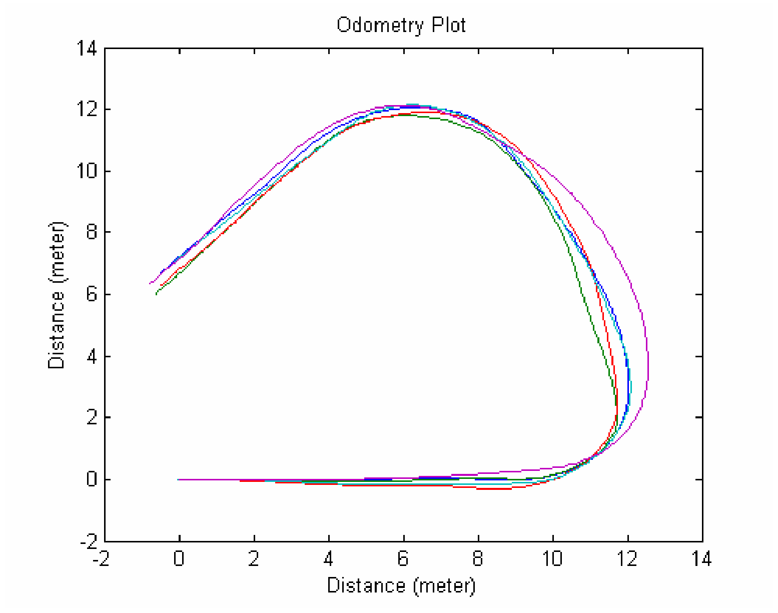


Figure 88: Odometry plot of the outdoor experiment

In the offline data association procedure, first, the sequences were partitioned into sequences using abrupt changes in angular speed as the markers. Figure 89 and Figure 90 shows the episode-normalized angular velocity sequence and linear velocity sequence, respectively. Four quick changes in the angular speed cut the sequences into five episodes. In Episode 1 (sample 0 to 100) the robot moved forward; in Episode 2 (sample 100 to 150) the robot turned left; in Episode 3 (sample 150 to 250) the robot moved forward and slowly turned left, in Episode 4 (sample 250 to 300) the robot turned left again, and finally in Episode 5 (sample 300 to 350) the robot moved forward until it reached the target position.

Figure 91 shows the episode-normalized hue count for 10 hue bands. Figure 92 shows the episode-normalized length of the longest laser segment. Figure 93 shows the episode-normalized compass reading, and Figure 94 shows the relative heading from DMU sensor.

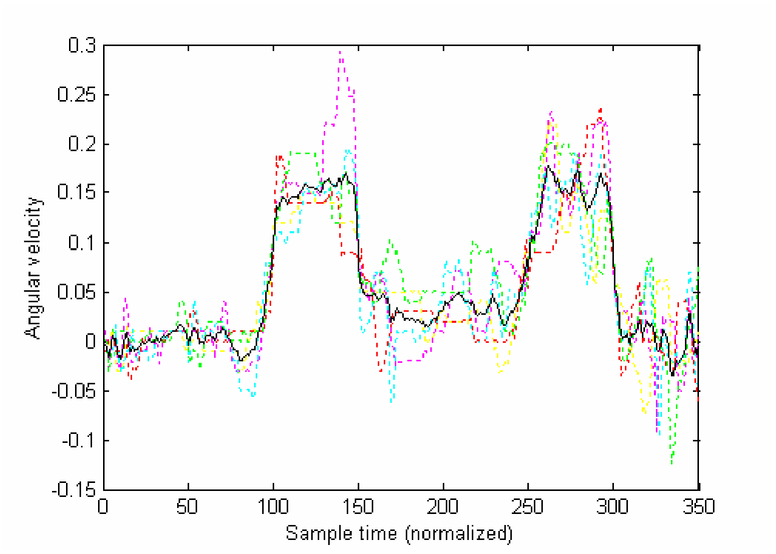


Figure 89: Episode-normalized angular velocity for outdoor sequences

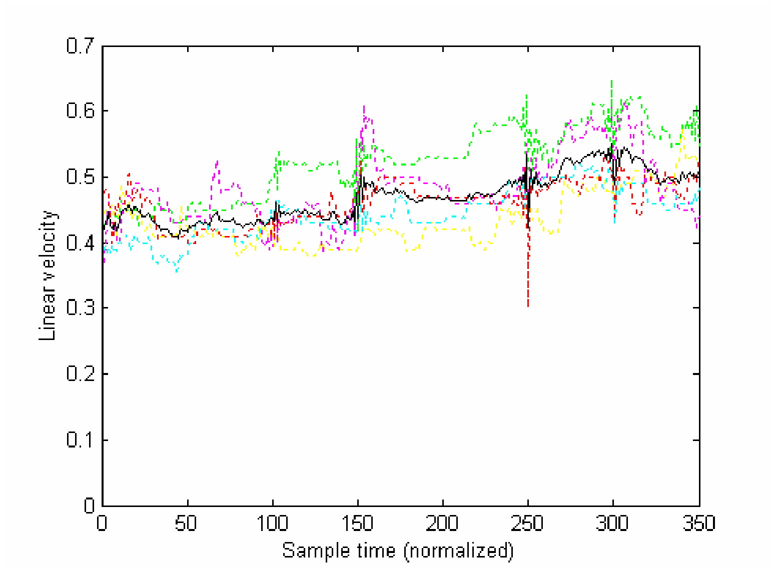
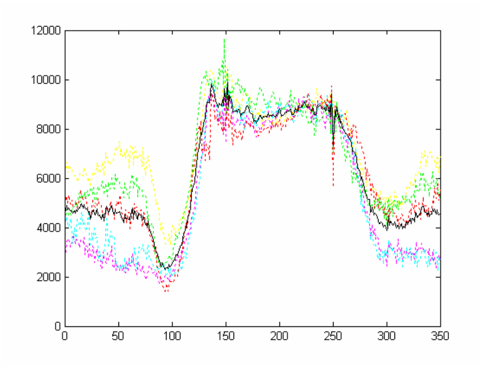
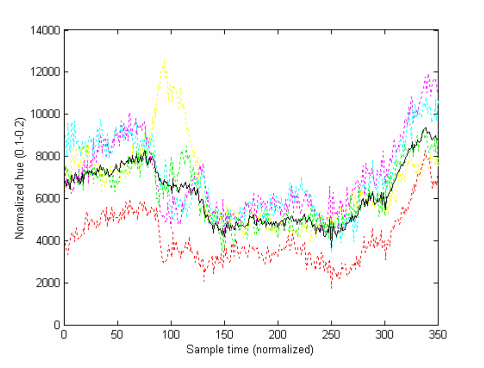


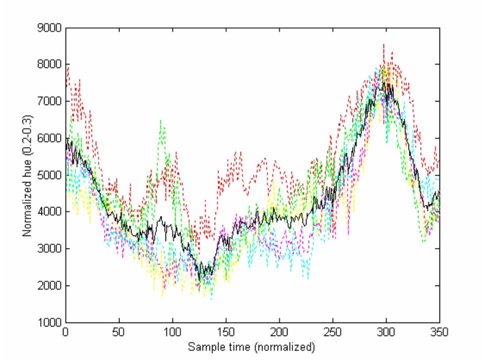
Figure 90: Episode-normalized linear velocity for outdoor sequences



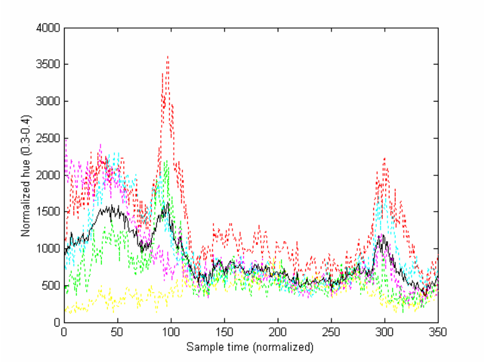
Hue 0-0.1



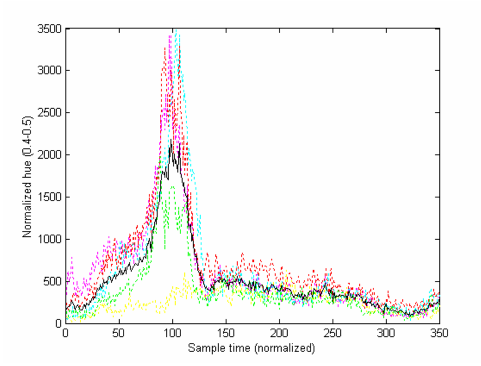
Hue 0.1-0.2



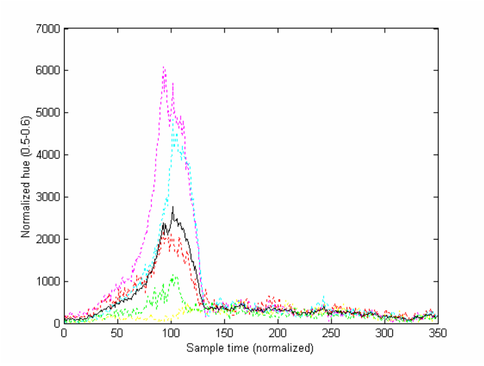
Hue 0.2-0.3



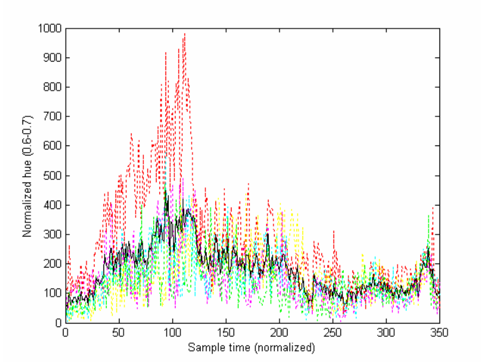
Hue 0.3-0.4



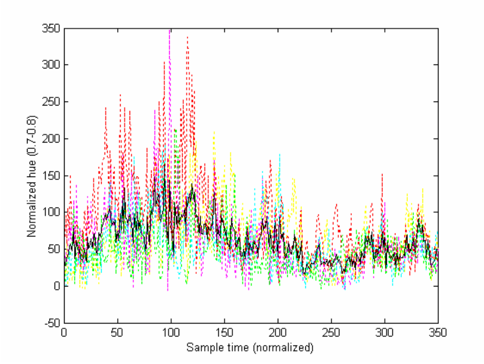
Hue 0.4-0.5



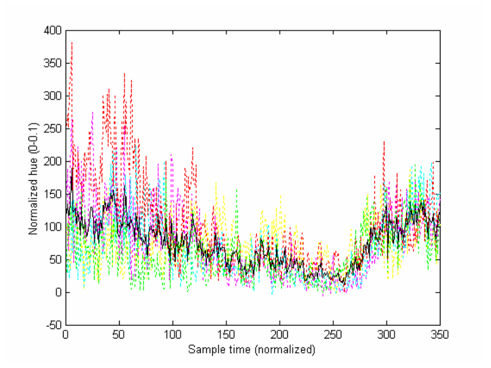
Hue 0.5-0.6



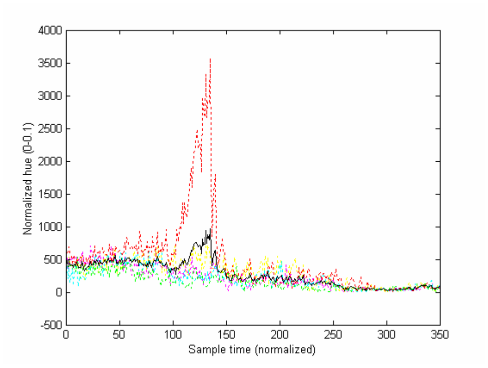
Hue 0.6-0.7



Hue 0.7-0.8



Hue 0.8-0.9



Hue 0.9-1.0

Figure 91: Episode-normalized hue sequences for outdoor experiments

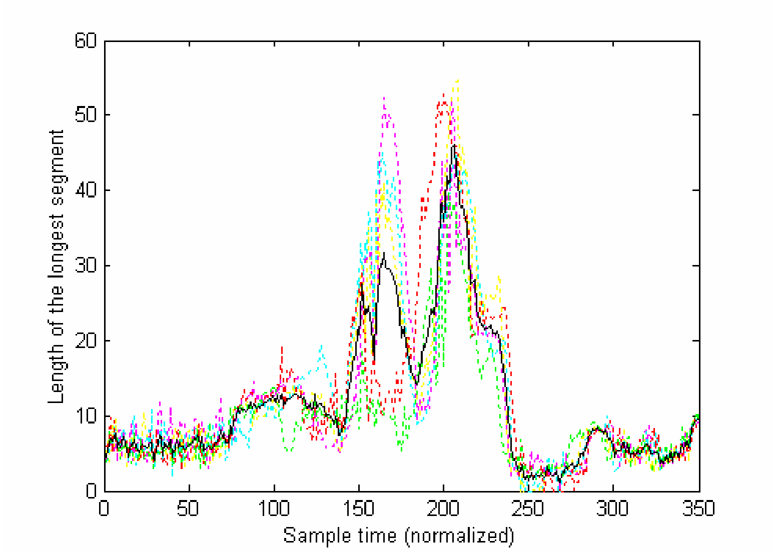


Figure 92: Episode-normalized length of the longest laser segment for outdoor experiments

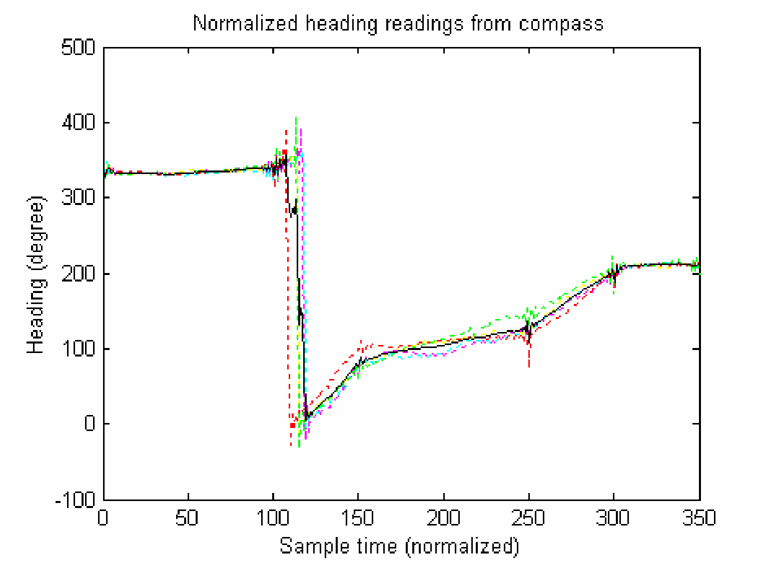


Figure 93: Episode-normalized compass readings for outdoor sequences

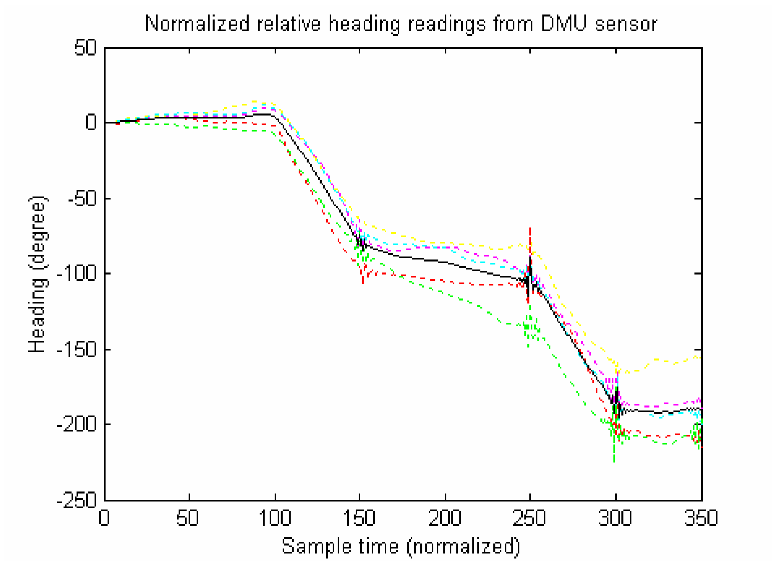


Figure 94: Episode-normalized relative heading from DMU sensor for outdoor sequences

Using the consistency and correlation criteria (cf. Chapter VI and next section of this chapter), following features were selected as the salient features for the four episode boundaries:

- Boundary 1:

Laser, DMU, Hue1

- Boundary 2:

  Laser, Compass, DMU, Hue3

- Boundary 3:

  Compass, DMU, Hue1, Hue10

- Boundary 4:

  Laser, Compass, DMU

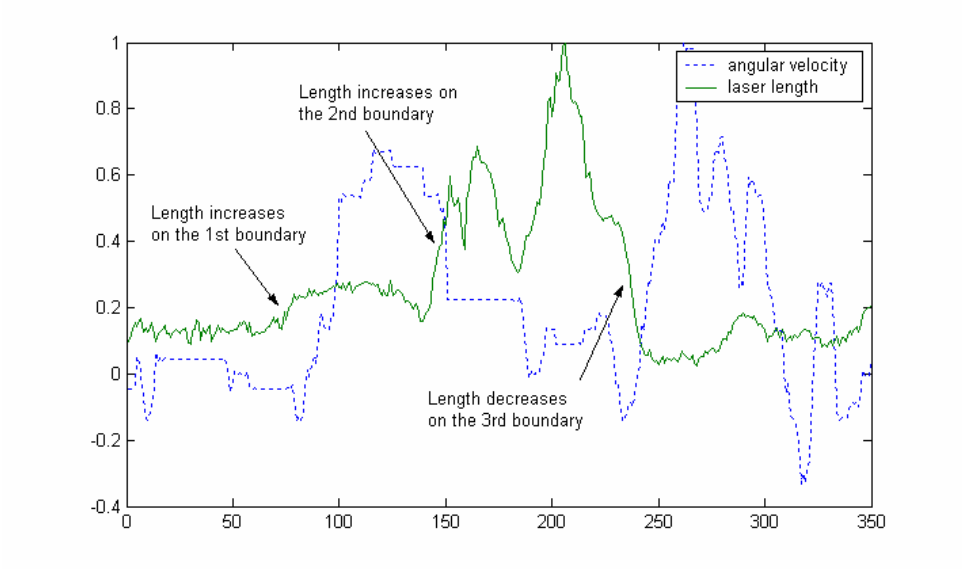


Figure 95: Correlation between laser and angular velocity

As an example, the above figure shows the correlation between the length of the longest laser segment and the robot's angular velocity. To evaluate the validity of the feature saliency, the robot was put back in the same environment and commanded to move to the same target position starting randomly on the first path. This was done 15 times in three batches on three days under different lighting conditions.

- First batch: 6 trials, sunny, morning

- Second batch: 4 trials, sunny, afternoon
- Third batch: 5 trials, cloudy, morning

Using a distance criterion of 1.5 meters, 12 times the robot successfully navigated to the target position, thus the success rate is 80%. Figure 96 shows the trajectories of the 12 successful evaluation trials, while Figure 97 shows the trajectories of the 3 failed evaluation trials. Note that all the failure happened in Episode 3 (cf. Figure 97).

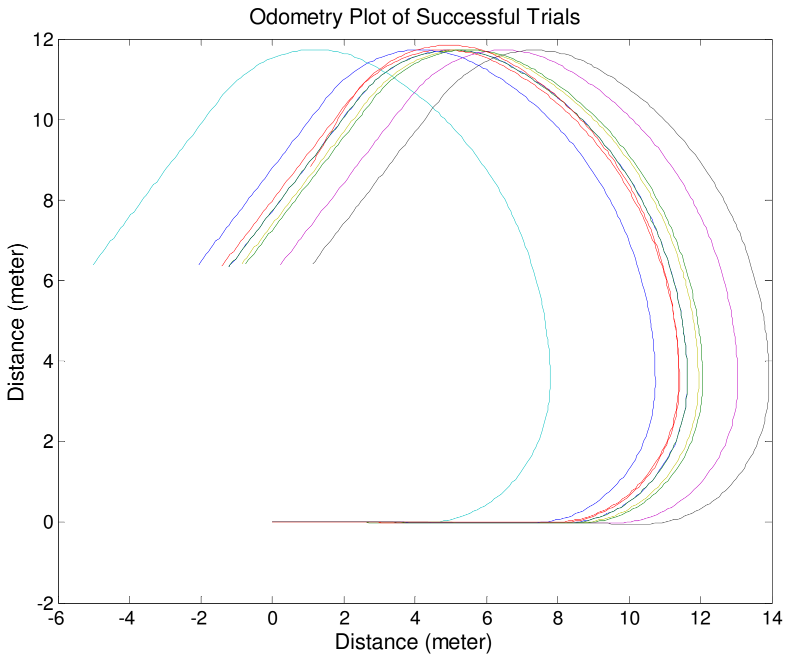


Figure 96: Trajectories of the successful evaluation trials for outdoor route

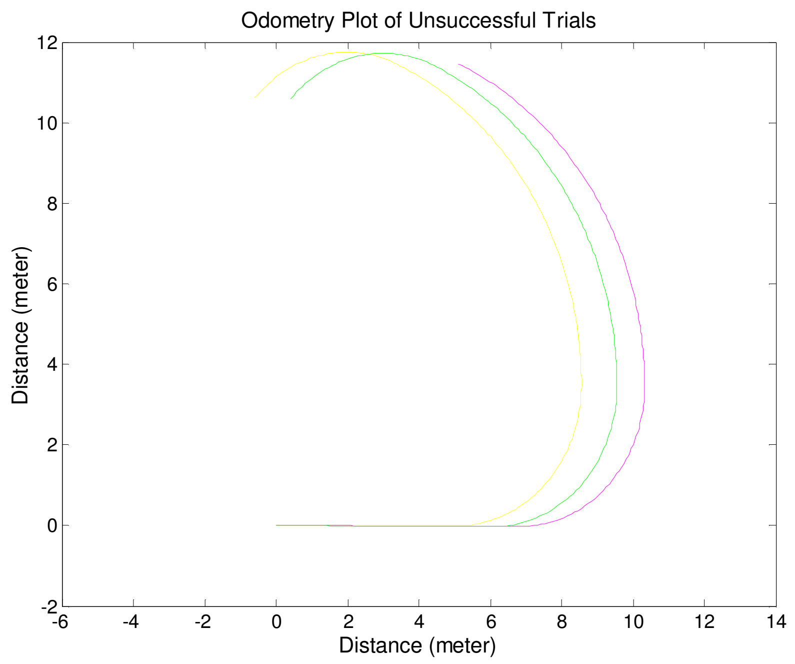


Figure 97: trajectories of the unsuccessful evaluation trials for outdoor route

Some Issues on Salient Feature Extraction

Two questions arise naturally from the discussion in Chapter VI and the above experiments:

- Two consistency indexes were proposed in Chapter VI, is there any significant difference between them in real world applications?
- In all the experiments, five teleoperation trials were used to extract salient features, and obtained satisfactory result. The question is: why five trials? How about 2, 3, or 4 trial?

This section will address these two questions empirically using the outdoor experiment data.

Consistency Indexes

Two consistency indexes were proposed in Chapter VI based on H^1 ($CI_{j,k}$) and H^2 ($VI_{j,k}$) distances:

$$CI_{j,k} = \sum_{W_k} \left(\frac{1}{M} \sum_i \left| \frac{\mathbf{s}'_{i,j}(t) - \overline{\mathbf{s}}'_{i,j}(t)}{\max(\overline{\mathbf{s}}'_{i,j}(t))} \right| \right) \tag{8.1}$$

$$VI_{j,k} = \sum_{W_k} \left(\sqrt{\frac{1}{M} \sum_i \left(\frac{\mathbf{s}'_{i,j}(t) - \overline{\mathbf{s}}'_{i,j}(t)}{\max(\overline{\mathbf{s}}'_{i,j}(t))} \right)^2} \right) \tag{8.2}$$

Figure 98 and Figure 99 show the result of applying these two consistency indexes to all the sensory feature signals at the four boundaries of the outdoor sequences. It can be observed that these two indexes give compatible results. This is not surprising, since H^1 and H^2 are both distance measurements.

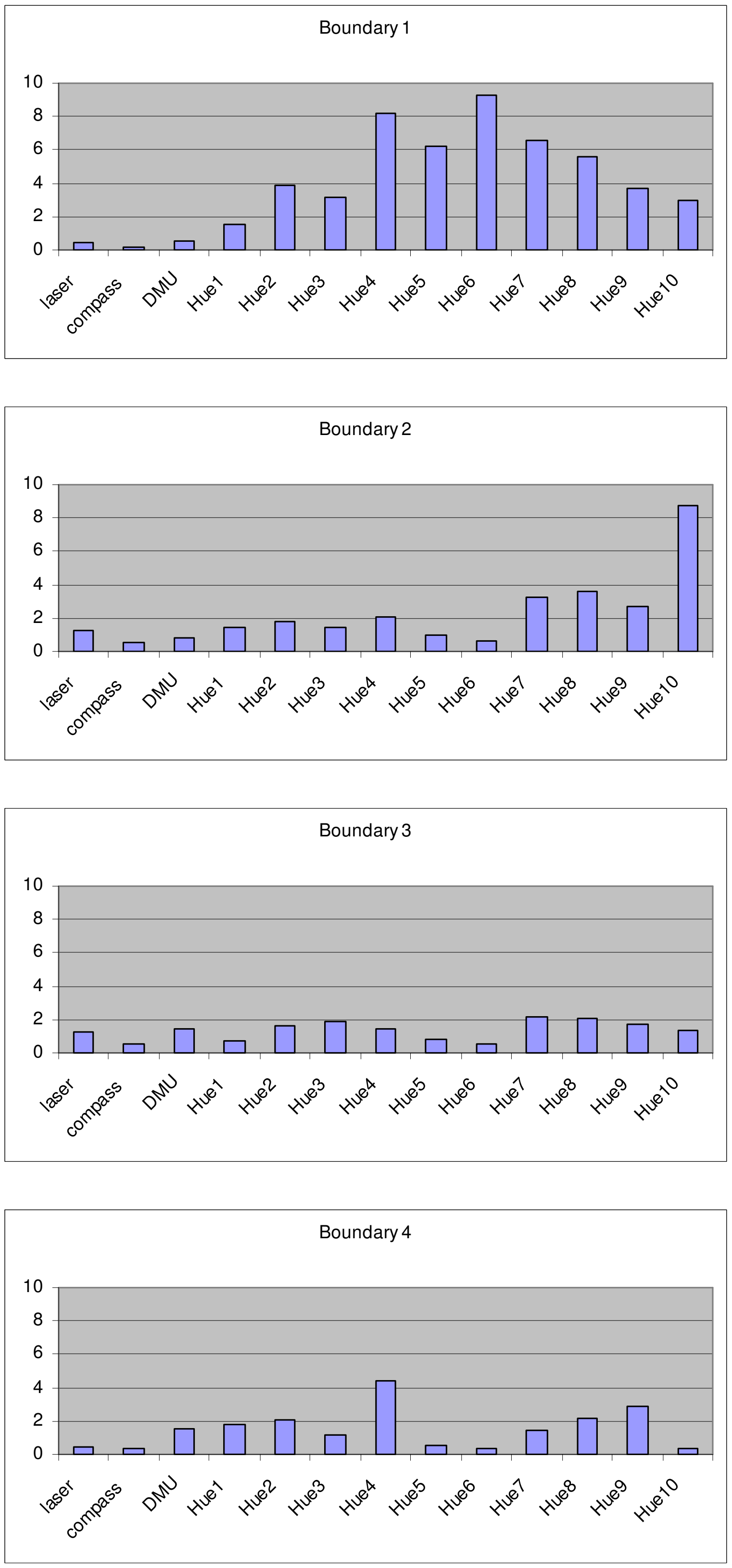


Figure 98: Consistency Index $CI_{j,k}$ for the four boundaries

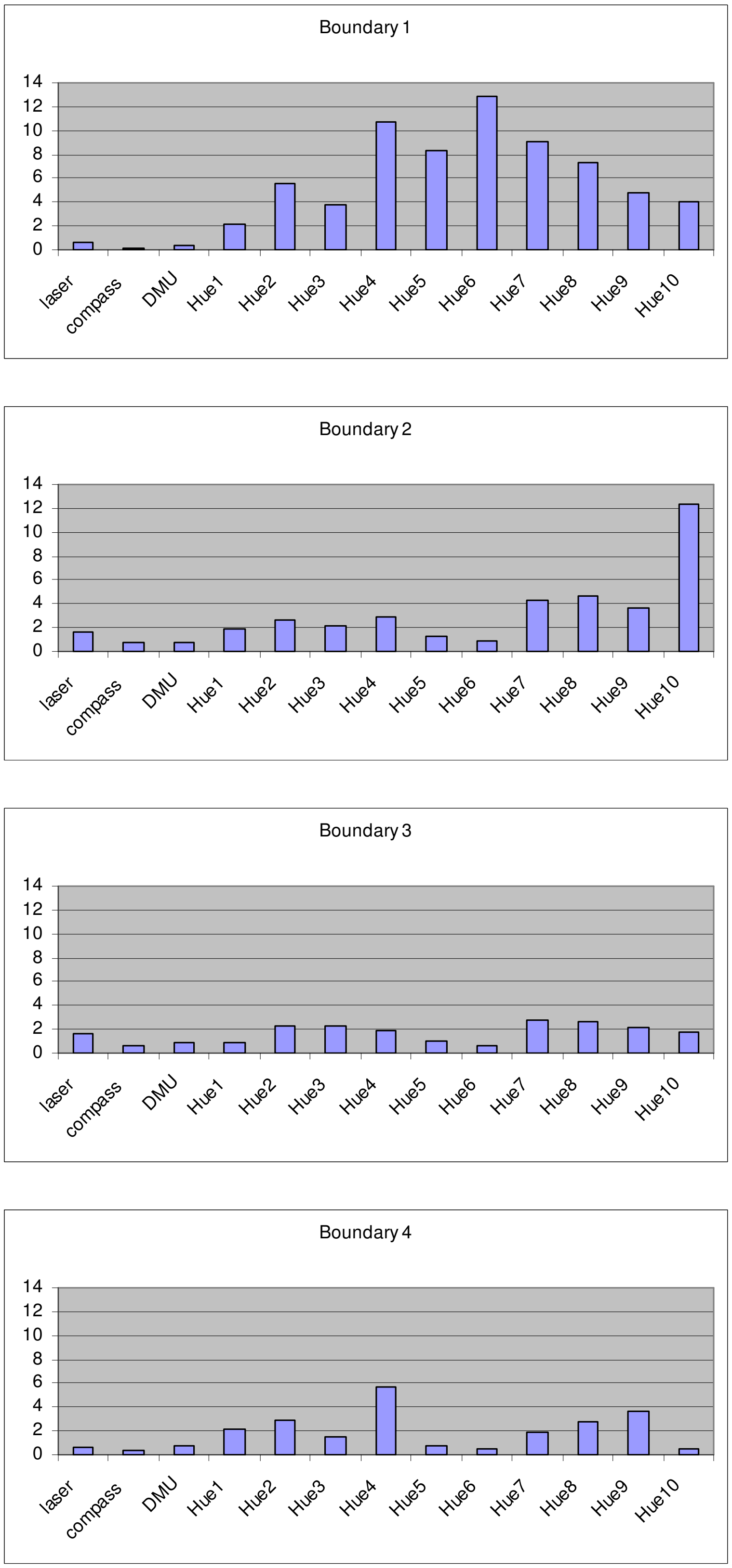


Figure 99: Consistency Index $VI_{j,k}$ for the four boundaries

Number of Teleoperation Trials

To check the effect of number of teleoperation trials on salient feature extraction, the original five teleoperation trials were combined into groups of two trials, three trials, and four trials. For group of two, three, and four trials, there were totally ten ($C_5^2 = 10$), ten ($C_5^2 = 10$), and five ($C_5^2 = 10$) combinations, respectively. Only consistency correlation was considered here, because the other criterion, correlation, would remain similar.

Both type-I error (regard a random signal as a salient features) and type-II error (regard a salient feature as a random signal) were obvious in these combinations. Using a threshold of 2 (this value of threshold was used for five teleoperation trials.), the error rate were as follows:

	Type-I error rate	**Type-II error rate**
2 trials	21%	11%
3 trials	18%	9.2%
4 trials	5.4%	4.3%

In my experiment, 5 trials gave satisfactory result. Another question arises: How about more trial (e.g., 6, 7, 8)? How many trials are necessary to extract salient features? These questions need to be explored in the future research.

Attention Operator: Saliency Map

I originally proposed in my research proposal to use Saliency Map (an attention operator, cf. Appendix A) to find consistent patterns in the image sequences and to use these consistent patterns as salient features for navigation. By using this saliency discovery procedure, I found that this approach cannot extract useful salient visual features. This is another example of an infamous phenomenon: the designer cannot anticipate all the possible outcomes when designing the control architecture for a robot; instead, learning and adaptation are crucial to the success of intelligent systems.

There is no universal definition for "saliency". In this research, a "salient feature" means that this feature is (a) consistent and (b) correlate well with a particular motor behavior change. In Itti's Saliency Map, "saliency" is a measurement of how much a particular image location differs from its neighborhood (cf. Appendix A), i.e., a "salient location", also called an "attention focus", is a "pop-up" point in the image. Figure 100 shows an outdoor image marked with eight (8) attention focuses.

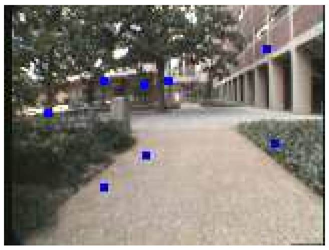


Figure 100: An image with attention focuses marked

My original research idea is as follows. If some of the attentional focuses from Saliency Map are consistent and correlate well with the motor behavior, then these attentional focuses could serve as "landmarks" and guide the robot's navigation. However, after some preliminary research, it turned out that the attentional focuses from the Saliency Map were not consistent. Figure 101 gives an example. It shows the images from the five teleoperation trials taken at similar location with attention focuses marked. It can be concluded from these images that there is no single attention focus that appears in all five images.

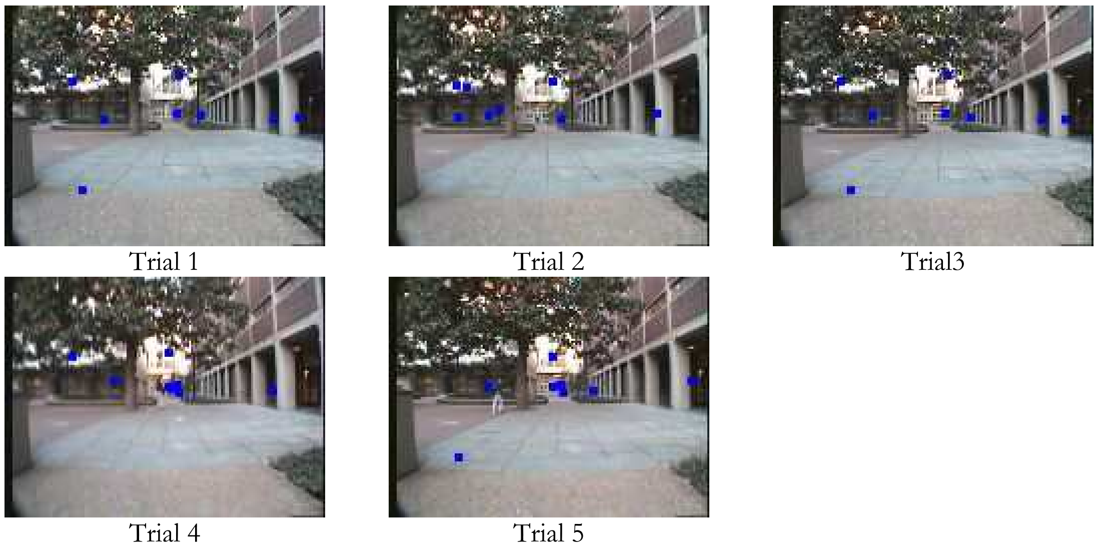


Figure 101: Images from five trials, taken at similar locations, with attention focuses marked

One explanation for this inconsistency of attention focuses is that Saliency Map is a fine-grained attentional operator. A slight change in the environment will produce different result. Figure 102 shows the change of the attention focuses when a person walked in font of the robot in consecutive images. In this example, the environment did not change much, yet the attention focuses changed, especially some focuses were following the moving person. This approach was finally abandoned.

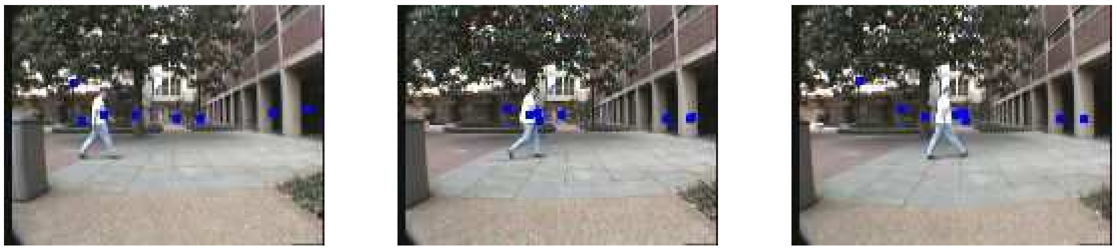


Figure 102: Three images showing a walking person with attention focuses marked

CHAPTER IV

CONCLUSIONS AND FUTURE RESEARCH

Conclusions

The objective of this dissertation research was to extract features that are salient to the behaviors for mobile robot navigation and to use these features for autonomous navigation. The work was implemented on a mobile robot, Scooter, and was evaluated in both indoor and outdoor environments.

This dissertation reports a test of the hypothesis that a robot can acquire SMC through repetitive control by a human and then use the SMC information for autonomous operation. If during repeated human-controlled trials of a task the robot records all its sensory signal and control commands, then through offline statistical analysis, those sensory-motor couplings which accompany purposeful motions can be identified. These sequential sensory-motor couplings form a chain of salient feature events and behaviors. The robot later uses this event-behavior chain to navigate in the learned environment autonomously.

It is assumed that the mobile robot has already implemented a rudimentary set of basic behaviors such as, move forward, turn, obstacle avoidance, etc. The goal is for the robot to learn how to navigate in a specific environment. More specifically, it is desired for the robot (1) to extract features[6] salient to the navigation task, and (2) to determine if these features can be used by the robot to navigate autonomously. The procedure described here is independent of the environment, that is, by using this procedure a robot should be able to learn any specific environment:

[6] Here 'signals' and 'features' mean the same and are used interchangeably.

1. The mobile robot is teleoperated in an environment along a path several times (five times in my experiments). Each time, the robot runs along essentially the same path with random small deviations [7].. All sensory data, both visual and non-visual, along with the motor commands, are recorded throughout the whole trial and saved.
2. The saved signal set for each trial is partitioned (in time) into episodes by motor state changes, which act as temporal markers. Due to the user control, the motor state changes will be similar for every trial.
3. Across all trials the corresponding episodes are normalized (through resampling) in duration. An average signal is also computed.
4. Within a given episode, across all trials, corresponding sensory signals are compared and correlated with the motor commands. The features that are consistent across all the trials and correlate well with the motor commands are considered salient to the episode and are retained; while those features that vary randomly across the trials or don't correlate well with the motor commands are considered uninformative to the task and are discarded.
5. The robot is put back into the same environment and commanded to navigate to the target position using the extracted salient features. Its performance is measured. This is also performed in an environment similar to the environment that the robot has learned.

In the last procedure, the evaluation is performed in both the same and similar environments. This not only evaluates the validity of the features' saliency, but also tests the adaptability of these features. Experiments were carried out in both the indoor environments

[7] It is necessarily the case that conditions are different during each trial, because it is impossible for the human controller to exactly repeat his own actions, and because the environment conditions constantly change.

and outdoor environments. The successful rate was 100% in the indoor environments, while it was 80% in the outdoor environment. This was not surprising, because indoor environment was more static than outdoor environment.

The main novel contributions of this research were:

1. Development of a method for extracting features salient features via teleoperation;
2. Demonstration of using this method to extract features salient to mobile robot navigation;
3. Evaluation of the performance of autonomous navigation using extracted salient features.

The limitation of this research were:

1. Motor sequences must have abrupt changes in order to partition the sequences into episodes, so this method cannot be applied when there is not such abrupt changes.
2. This method uses human operator's teleoperation to bootstrap the learning process, thus the performance (esp. the features extracted) will highly depend on the skills of the operator (this has not been verified). Different teleoperators might generate different results.
3. The feature extraction process was tedious. This is due to (a) multiple teleoperation trials are required, and (b) offline data association takes time.
4. Two criteria were used to extract salient features: consistency and correlation. These two criteria did not check that the extracted salient features were unique in the whole sequence. In certain situations features extracted by using these two criteria might not be salient. For example, imagine in an indoor environment

with oscillating lighting conditions (bright-dark-bright-dark, etc.). If the above two criteria were used, then this oscillating brightness would be regard "salient" (it is NOT.) However, this hardly happens in real situations, and current experiments gave satisfactory result.

Future Research

In order to improve the system's performance and robustness, it would probably be useful to focus on following issue.

1. As pointed above, a third criterion should be used to check the uniqueness of the extracted salient features.
2. Five teleoperation trials were used in the experiment, and it showed empirically that less trials (2, 3, or 4 trials) were not able to discover the consistent features. It would be interesting to see what is the effect if more teleoperation trials (6, 7, and more) were used.
3. It showed empirically that Saliency Map could not discover salient visual features. This might due to the fact that Saliency Map is a highly-localized and data-driven attention operator. Other types of attention operator that are global and model-driven might be able to find consistent attention focuses.

This research could be extended to incorporate other researches. One possible extension is to use finite state machine (FSM) to express the event-behavior relationship. An FSM M can be specified by a quadruple[14, 114]:

$$M = (Q, \delta, q_0, F) \tag{8.3}$$

with Q representing the set of allowable behavioral states; δ being a transition function mapping the input and current state to another state; q_0 denoting the starting behavioral configuration; and F representing a set of accepting states, a subset of Q, indicating completion of the sensory-motor task. An FSM can have multiple starting/ending states.

FSM can be visually expressed as a FSM diagram. In this FSM, each SMC event is a directional edge and each node is a behavior. In the execution step, the robot performs the task by (a) comparing the current sensory reading with the salient features and (b) comparing current state with the desired goal state, thus decides the next behavior.

Other models, such as spreading activation network (SAN) [88, 89] and Hidden Markov Model (HMM) [115] can also be used to express the event-behavior relationships.

APPENDIX A

SALIENCY-MAP: A VISUAL ATTENTION OPERATOR

In a general sense, visual attention can be defined as the ability to select a region of interest for extracting information useful for a given task. It has long been studied by researchers in psychology (for example [116] [117]) as well as those in computer vision (for example [118]). The attention operator used in this research was proposed by Laurent Itti [119]. It is a saliency-based visual attention method. 'Saliency-Map' is the name I call it for convenience; Itti himself does not give it a short name.

Saliency-Map builds on a biologically plausible architecture proposed by Koch and Ullman [120]. It is related to the so-called "feature integration theory", explaining human visual search strategies. Visual input is first decomposed into a set of topographic feature maps. Different spatial locations then compete for saliency within each map, such that only locations which locally stand out from their surroundings can persist. All feature maps feed, in a purely bottom-up manner, into a master "saliency map", which topographically codes for local conspicuity over the entire visual scene. The model's saliency map is endowed with internal dynamics which generate attentional shifts. This model consequently represents a complete account of bottom-up saliency and does not require any top-down guidance to shift attention.

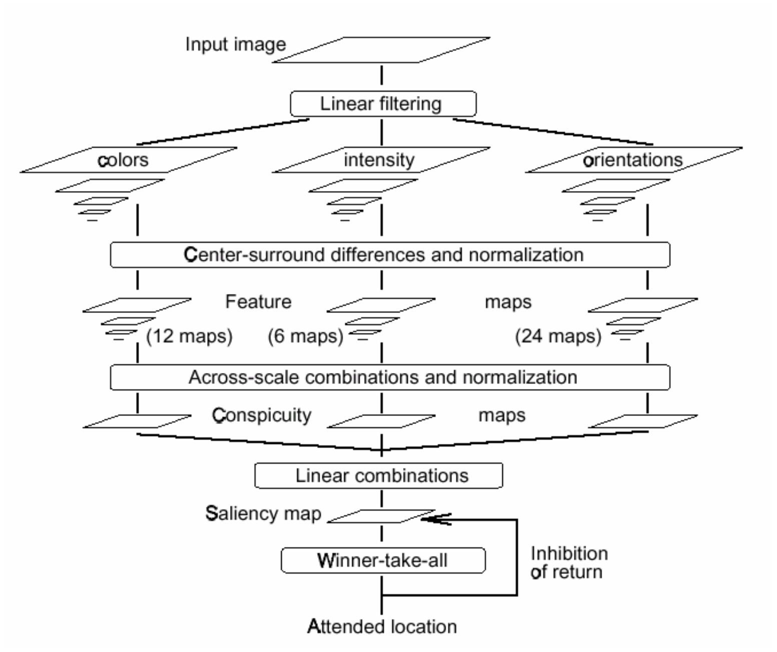


Figure 103: General architecture of the saliency map model [119]

Figure 103 shows the general structure of this model. Input is provided in the form of static color images. Nine spatial scales are created using dyadic Gaussian pyramids. Each feature is computed by a set of linear "center-surround" operations akin to visual receptive fields. Totally 42 feature maps are computed: six for intensity, 12 for color, and 24 for orientation. These feature maps are combined into three "conspicuity maps", intensity, color, and orientation at the last scale of the saliency map. These three conspicuity maps are normalized and summed into the final input to the saliency map. A winner-take-all neural network finds the maximum of the saliency map. This maximum defines the most salient image location, to which the focus of attention should be directed.

REFERENCES

[1] U. Nehmzow and T. Mitchell, "The prospective student's introduction to the robot learning problem," Technical report UMCS-95-12-6, University of Manchester, Manchester, UK, 1995.

[2] M. A. Hearst and H. Hirsh, "AI's greatest trends and controversies," *IEEE Intelligent Systems and Their Applications*, no.1, pp. 8-17, 2000.

[3] N. J. Nilsson, "Shakey the Robot," Technical Report 323, SRI International, Menlo Park, CA, 1984.

[4] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, vol. 2, no.1, pp. 14-23, 1986.

[5] M. J. Mataric, "Integration of representation into goal-driven behavior-based robots," *IEEE Transactions on Robotics and Automation*, vol. 8, no.3, pp. 304-312, 1992.

[6] R. Pfeifer and C. Scheier, "Sensory-motor coordination: The metaphor and beyond," *Robotics and Autonomous Systems*, vol. 20, no.2-4, pp. 157-178, 1997.

[7] M. T. Rosenstein and P. R. Cohen, "Continuous categories for a mobile robot," *16th National Conference on Artificial Intelligence*, pp. 634-640, Orlando, FL, USA, 1999.

[8] M. E. Cambron, "Sensory motor control for grasp pose estimation in humanoid robot," Ph.D. Dissertation, Vanderbilt University, Nashville, TN, 2001.

[9] R. A. Peters, C. Campbell, W. Bluethmann, and E. Huber, "Robonaut Task Learning through Teleoperation," *International Conference on Robotics and Automation*, 2003.

[10] R. A. Brooks, "Intelligence without representation," *Artificial Intelligence*, vol. 47, no.1-3, pp. 139-159, 1991.

[11] R. A. Brooks, "A robot that walks: emergent behaviors from a carefully evolved network," *IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, USA, 1989.

[12] R. A. Brooks, "Elephants don't play chess," *Robotics*, vol. 6, no.1-2, pp. 3-15, 1990.

[13] M. A. Arbib, P. Erdi, and J. Szentagothai, *Neural Organization: Structure, Function, and Dynamics*. Cambridge, MA: MIT Press, 1998.

[14] R. C. Arkin, *Behavior-Based Robotics*. Cambridge, MA, USA: MIT Press, 1998.

[15] R. Hartley and F. Pipitone, "Experiments with the Subsumption Architecture," *IEEE International Conference on Robotics and Automation*, pp. 1652-1658, Sacramento, CA, 1991.

[16] D. Kirsh, "Today the earwig, tomorrow man?," *Artificial Intelligence*, vol. 47, pp. 161-184, 1991.

[17] M. A. Arbib, "Perceptual structures and distributed motor control," in *Handbook of Physiology---The Nervous System II: Motor Control*, V. B. Brooks, Ed. Bethesda, MD: American Physiological Society, 1981, pp. 1449-80.

[18] R. C. Arkin, "Motor schema based navigation for a mobile robot: An approach to programming by behavior," *IEEE International Conference on Robotics and Automation*, pp. 264-271, Raleigh, NC, 1987.

[19] R. C. Arkin, "Integrating behavioral, perceptual and world knowledge in reactive navigation," *Robotics and Autonomous Systems*, vol. 6, pp. 105-122, 1990.

[20] R. A. Brooks, "From earwigs to humans," *Robotics and Autonomous Systems*, vol. 20, no.2-4, pp. 291-304, 1997.

[21] E. Gat, "On Three-Layer Architectures," in *Artificial Intelligence and Mobile Robots*, D. Kortenkamp, R. P. Bonnasso, and R. Murphy, Eds.: MIT/AAAI Press, 1997.

[22] R. C. Arkin and T. Balch, "AuRA: Principles and Practice in Review," *Journal of Experimental and Theoretical Artificial Intelligence (JETAI)*, vol. 9, no.2-3, pp. 175-189, 1997.

[23] B. Adams, C. Breazeal, R. A. Brooks, and B. Scassellati, "Humanoid robots: a new kind of tool," *IEEE Intelligent Systems and Their Applications*, vol. 15, no.4, pp. 25-31, 2000.

[24] R. Pfeifer, "Cognition - perspectives from autonomous agents," *Robotics and Autonomous Systems*, vol. 15, pp. 47-70, 1995.

[25] R. Pfeifer and C. Scheier, *Understanding Intelligence*. Cambridge, MA, USA: MIT Press, 1999.

[26] T. Mitchell, *Machine Learning*: McGraw Hill, 1997.

[27] U. Nehmzow, "Experiments in competence acquisition for autonomous mobile robots," Ph.D. thesis, University of Edinburgh, 1992.

[28] U. Nehmzow, "Autonomous acquisition of sensor-motor coupling in robots," Technical report UMCS-94-11-1, University of Manchester, Manchester, UK, 1994.

[29] L. W. Barsalou, *Cognitive Psychology: An Overview for Cognitive Scientists*. Hillsdale, NJ: Lawrence Erlbaum, 1992.

[30] S. Ullman, *High-Level Vision: Object recognition and Visual Cognition*. Cambridge, MA: MIT Press, 1996.

[31] M. K. Hu, "Visual Pattern Recognition by Moment Invariants," *IRE Transactions on Information Theory*, vol. 8, no.2, pp. 179-187, 1962.

[32] H. Wechsler, *Computational Vision*. San Diego, CA: Academic Press, 1990.

[33] K. Pahlavan, T. Uhlin, and J. Eklundh, "Active vision as a methodology," in *Active Perception*, Y. Aloimonos, Ed. Hillsdale, NJ: Lawrence Erlbaum Associates, 1993, pp. 19-46.

[34] R. Bajcsy, "Active perception," *Proceedings of IEEE*, vol. 76, no.8, pp. 996-1005, 1988.

[35] J. Dewey, "The reflex arc in psychology," in *The Philosophy of John Dewey*, J. J. McDermott, Ed. Chicago, IL: University of Chicago Press, 1896, pp. 136-148.

[36] J. J. Gibson, *The ecological approach to visual perception*. Boston, MA: Houghton Mifflin, 1979.

[37] M. M. Waldrop, *Complexity: the emerging science at the edge of order and chaos*. New York, NY: Simon & Schuster, 1992.

[38] C. Scheier and R. Pfeifer, "Classification as sensory-motor coordination: a case study on autonomous agents," *Advances in Artificial Life: Third European Conference on Artificial Life*, 1995.

[39] S. Nolfi, "Adaptation as a more powerful tool than decomposition and integration," *13th International Conference on Machine Learning*, 1996.

[40] D. Lambrinos and C. Scheier, "Extended Braitenberg Architectures," Technical Report 95.10, AI Lab, University of Zurich, Zurich, Switzerland, 1995.

[41] T. Kohonen, *Self-Organizing Maps*. Berlin: Springer, 1995.

[42] D. Lambrinos and C. Scheier, "Categorization in a real-world agent using haptic exploration and active perception," in *From animals to animats: Proceedings of Fourth International Conference on Simulation of Adaptive Behavior*: MIT Press, 1996.

[43] D. Lambrinos and C. Scheier, "Building complete autonomous agents: A case study on categorization," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Osaka, Japan, 1996.

[44] J. A. Coelho, Jr., J. H. Piater, and R. A. Grupen, "Developing Haptic and Visual Perceptual Categories for Reaching and Grasping with a Humanoid Robot," *IEEE-RAS International Conference on Humanoid Robots*, Cambridge, MA, 2000.

[45] J. A. Coelho, Jr., J. H. Piater, and R. A. Grupen, "Developing Haptic and Visual Perceptual Categories for Reaching and Grasping with a Humanoid Robot," *Robotics and Autonomous Systems*, vol. 37, no.2-3, pp. 195-219, 2001.

[46] J. H. Piater and R. A. Grupen, "Toward learning visual discrimination strategies," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 410-415, 1999.

[47] J. H. Piater, "Learning Visual Features to Recommend Grasp Configurations," *ICML-2000 Workshop on Machine Learning of Spatial Knowledge*, Stanford, CA, 2000.

[48] J. H. Piater, "Visual Feature Learning," Ph.D. Dissertation, University of Massachussetts Amherst, Amherst, MA, 2001.

[49] R. Pfeifer, "Building "fungus eater": design principles of autonomous agents," in *From animals to animats: Proceedings of Fourth International Conference on Simulation of Adaptive Behavior*: MIT Press, 1996, pp. 3-12.

[50] E. L. Thorndike, *Animal Intelligence*. New York: Macmillan, 1911.

[51] P. Bakker and Y. Kuniyoshi, "Robot see, robot do: an overview of robot imitation," *AISB96 Workshop: Learning in Robots and Animals*, 1996.

[52] R. Yando, V. Seitz, and E. Zigler, *Imitation: A Developmental Perspective*. Hillsdale, NJ: Lawrence Erlbaum, 1978.

[53] S. Schaal, "Is imitation learning the route to humanoid robots?," *Trends in Cognitive Sciences*, vol. 3, no.6, pp. 233-242, 1999.

[54] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: Extracting reusable task knowledge from visual observation of human performance," *IEEE Transactions on Robotics and Automation*, vol. 10, no.6, pp. 799-822, 1994.

[55] G. Hayes and J. Demiris, "A Robot Controller Using Learning by Imitation," *2nd International Symposium on Intelligent Robotic Systems*, Grenoble, France, 1994.

[56] M. J. Mataric, "Designing emergent behaviors: from local interaction to collective intelligence," in *From animals to animats: Proceedings of Second International*

Conference on Simulation of Adaptive Behavior. Cambridge, MA, USA: MIT Press, 1992, pp. 432-441.

[57] M. J. Mataric, "Interaction and Intelligent Behavior," Ph.D. Dissertation, MIT, Cambridge, MA, USA, 1994.

[58] M. J. Mataric, "Issues and approaches in the design of collective autonomous agents," *Robotics and Autonomous Systems*, vol. 16, no.2-4, pp. 321-331, 1995.

[59] M. J. Mataric, "Designing and understanding adaptive group behavior," *Adaptive Behavior*, vol. 4, no.1, pp. 51-80, 1995.

[60] M. J. Mataric, M. Nilsson, and K. T. Simsarian, "Cooperative multi-robot box-pushing," *IEEE International Conference on Intelligent Robots and Systems*, pp. 556-561, Piscataway, NJ ,95CB35836, 1995.

[61] A. Fod, M. J. Mataric, and O. C. Jenkins, "Automated Derivation of Primitives for Movement Classification," *First IEEE-RAS International Conference on Humanoid Robotics (Humanoids-2000)*, Cambridge, MA, 2000.

[62] A. Fod, M. J. Mataric, and O. C. Jenkins, "Automated derivation of primitives for movement classification," *Autonomous Robots*, vol. 12, no.1, pp. 39-54, 2002.

[63] M. J. Mataric, "Getting humanoids to move and imitate," *IEEE Intelligent Systems and Their Applications*, vol. 15, no.4, pp. 18-24, 2000.

[64] O. C. Jenkins, M. J. Mataric, and S. Weber, "Primitive-Based Movement Classification for Humanoid Imitation," *First IEEE-RAS International Conference on Humanoid Robotics (Humanoids-2000)*, Cambridge, MA, 2000.

[65] A. Billard and M. J. Mataric, "A biologically inspired robotic model for learning by imitation," *Proceedings of the Interantional Conference on Autonomous Agents*, pp. 373-380, New York, NY, 2000.

[66] E. Freund and J. Rossmann, "Projective virtual reality: Bridging the gap between virtual reality and robotics," *IEEE Transactions on Robotics and Automation*, vol. 15, no.3, pp. 411-422, 1999.

[67] Y. Kuniyoshi and L. Berthouze, "Neural Learning of Embodied Interaction Dynamics," *Neural Networks*, vol. 11, no.7-8, pp. 1259-1276, 1998.

[68] L. Berthouze and Y. Kuniyoshi, "Emergence and Categorization of Coordinated Visual Behavior Through Embodied Interaction," *Machine Learning*, vol. 31, no.1/2/3, pp. 187-200, 1998.

[69] A. J. Ijspeert, J. Nakanishi, T. Shibata, and S. Schaal, "Nonlinear Dynamical Systems for Imitation with Humanoid Robots," *IEEE/RAS International Conference on Humanoid Robots*, 2001.

[70] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Trajectory Formation for Imitation with Nonlinear Dynamical Systems," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 752-757, Maui, Hawii, 2001.

[71] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement Imitation with Nonlinear Dynamical Systems in Humanoid Robots," *IEEE International conference on Robotics and Automation*, pp. 1398-1403, Washington, DC, 2002.

[72] D. C. Bentivegna and C. G. Atkeson, "A Framework for Learning From Observation Using Primitives," *Symposium of Robocup 2002*, Fukuoka, Japan, 2002.

[73] D. C. Bentivegna and C. G. Atkeson, "Learning How to Behave from Observing Others," *Workshop on motor control in humans and robots (SAB 2002)*, Edinburgh University, 2002.

[74] P. Gaussier and S. Zrehen, "PerAc: A neural architecture to control artificial animals," *Robotics and Autonomous Systems*, vol. 16, pp. 291-320, 1995.

[75] P. Gaussier, S. Moga, J. P. Banquet, and M. Quoy, "From perceptionaction loops to imitation processes: A bottom-up approach of learning by imitation," *Applied Artificial Intelligence*, vol. 12, no.7-8, pp. 701-727, 1998.

[76] P. Gaussier, S. Moga, J. P. Banquet, and M. Quoy, "From perceptionaction loops to imitation processes: A bottom-up approach of learning by imitation," *Socially Intelligent Agents*, pp. 49--54, Boston, MA, 1997.

[77] P. R. Cohen, "Learning Concepts by Interaction," Technical report 00-52, University of Massachusetts, Amherst, Amherst, MA, 2000.

[78] G. Lakoff, *Women, Fire, and Dangerous Things*: University of Chicago Press, 1984.

[79] J. M. Mandler, "How to build a bady II: Conceptual Primitives," *Psychological Review*, vol. 99, no.4, pp. 587-604, 1992.

[80] M. T. Rosenstein and P. R. Cohen, "Concepts from time series," *15th National Conference on Artificial Intelligence*, pp. 739-745, Menlo Park, CA, 1998.

[81] T. Oates, M. D. Schmill, and P. R. Cohen, "Identifying qualitatively different experiences: experiments with a mobile robot," *The Sixteenth International Joint Conference on Artificial Intelligence*, 1999.

[82] T. Oates, M. D. Schmill, and P. R. Cohen, "Identifying qualitatively different outcomes of actions: Gaining autonomy through learning," *Proceedings of the Interantional Conference on Autonomous Agents*, pp. 110-111, New York, NY, 2000.

[83] D. Sankoff and J. B. Kruskal, *Time Warps, String Edits, and Macromolecules: Theory and Practice of Sequence Comparison*. Reading, MA: Addison-Wesley, 1983.

[84] M. T. Rosenstein and P. R. Cohen, "Symbol Grounding with Delay Coordinates," *Grounding of Word Meaning: Data & Models Workshop, AAAI-98*, pp. 20-21, 1998.

[85] P. R. Cohen, N. Adams, and D. Hand, "Finding Patterns that Correspond to Episodes," Techinical Report 01-11, University of Massachusetts, Amherst, MA, 2001.

[86] P. R. Cohen, M. Ramoni, P. Sebastiani, and J. Warwick, "Unsupervised clustering of robot activites: a bayesian approach," Technical Report, University of Massachusetts, Amherst, MA, 2000.

[87] S. Russell and P. Norvig, *Artificial Intelligence: Modern Approach*: Prentice Hall, 1995.

[88] P. Maes and R. Brooks, "Learning to Coordinate Behaviors," *National Conference on Artificial Intelligence*, pp. 796-802, 1990.

[89] P. Maes, "How to do the Right Thing," *Connection Science Journal, Special Issue on Hybrid Systems*, vol. 1, 1990.

[90] G. N. DeSouza and A. C. Kak, "Vision for Mobile Robot Navigation: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no.2, pp. 237-267, 2002.

[91] C. Scheier and D. Lambrinos, "Adaptive classification in autonomous agents," *Applied Artificial Intelligence*, vol. 11, no.2, pp. 119-130, 1997.

[92] C. Scheier and S. Egner, "Visual attention in a mobile robot," *IEEE International Symposium on Industrial Electronics*, pp. 48-52, Guimaraes, Portugal, 1997.

[93] P. R. Cohen and N. Adams, "An algorithm for segmenting categorical time series into meaningful episodes," *Fourth Symposium on Intelligent Data Analysis*, pp. 198-207, 2001.

[94] MathWorks, " Signal Processing Toolbox User's Guide," 2000.

[95] S. Atiya and G. Hager, "Real-Time Vision-Based Robot Localization," *IEEE Transactions on Robotics and Automation*, vol. 9, no.6, pp. 785-800, 1993.

[96] M. Meng and A. C. Kak, "Mobile Robot Navigation Using Neural Networks and Nonmetrical Environment Models," *IEEE Control Systems*, no.Oct, pp. 30-39, 1993.

[97] S. Thrun, "Learning Metric-Topological Maps for Indoor Mobile Robot Navigation," *Artificial Intelligence*, vol. 99, no.1, pp. 21-71, 1998.

[98] S. Thrun, "Probabilistic Algorithms in Robotics," Technical Report CMU-CS-00-126, Carnegie Mellon University, Pittsburgh, PA, 2000.

[99] A. J. Davison and D. W. Murray, "Simultaneous Localization and Map-Building Using Active Vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no.7, pp. 865-880, 2002.

[100] Y. Matsumoto, M. Inaba, and H. Inoue, "Visual Navigation using View-Sequenced Route Representation," *IEEE International Conference on Robotics and Automation*, pp. 83-88, Minneapolis, MN, 1996.

[101] P. Gaussier, C. Joulain, S. Zrehen, J. P. Banquet, and A. Revel, "Visual navigation in an open environment without map," *IEEE International Conference on Intelligent Robots and Systems*, pp. 545-550, 1997.

[102] T. Ohno, A. Ohya, and S. i. Yuta, "Autonomous navigation for mobile robots referring pre-recorded image sequence," *IEEE Int'l Conf. Intelligent Robots and Systems*, pp. 672-679, 1996.

[103] S. D. Jones, C. Andresen, and J. L. Crowley, "Appearance based processes for visual navigation," *IEEE Int'l Conf. Intelligent Robots and Systems*, pp. 551-557, 1997.

[104] C. Joulain, P. Gaussier, A. Revel, and B. Gas, "Learning to build visual categories from Perception-Action associations," *IEEE International Conference on Intelligent Robots and Systems*, pp. 857-864, 1997.

[105] N. Winters and J. Santos-Victor, "Information sampling for vision-based robot navigation," *Robotics and Autonomous Systems*, vol. 41, no.2, pp. 145-159, 2002.

[106] J.-C. Latombe, *Robot Motion Planning*. Boston, MA: Kluwer Academic Publishers, 1991.

[107] Z. Li and J. F. Canny, *Nonholonomic Motion Planning*: Kluwer, 1992.

[108] Y. Kanayama, Y. Kimura, F. Miyazaki, and T. Noguchi, "A stable tracking control method for an autonomous mobile robot," *IEEE International Conference on Robotics and Automation*, pp. 384-389, 1990.

[109] R. T. Pack, "IMA: The Intelligent Machine Architecture," Ph.D. dissertation, Vanderbilt University, Nashville, TN, 1998.

[110] H. R. Everett, *Sensors for Mobile Robots: Theory and Application*. Natick, MA: A K Peters, 1995.

[111] Crossbow, "Crossbow Product Catolog." San Jose, CA, 1999.

[112] J. Ferber, *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Singapore: Addison Wesley, 1999.

[113] R. Olivares, "The Intelligent Machine Architecture Version 2.5," MS thesis, Vanderbilt University, Nashville, TN, 2003.

[114] R. Murphy, *Introduction to AI Robotics*. Cambridge, MA: MIT Press, 2002.

[115] L. R. Rabiner, "A tutorial on Hidden Markov Models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no.2, pp. 257-286, 1989.

[116] K. R. Cave and J. Wolfe, "Modeling the role of parallel processing in visual search," *Cognitive Psychology*, vol. 22, pp. 225-271, 1990.

[117] M.-S. Kim and K. R. Cave, "Spatial attention in visual search for features and feature conjunctions," *Psychological Science*, vol. 6, pp. 376-380, 1995.

[118] M. Jagersand, "Saliency maps and attention selection in scale and spatial coordinates: an information theoretic approach," *5th International Conference on Computer Vision*, pp. 195-202, 1995.

[119] L. Itti, C. Koch, and E. Niebur, "Model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no.11, pp. 1254-1259, 1998.

[120] C. Koch and S. Ullman, "Shifts in selective visual attention: toward the underlying neural circuitry," *Human Neurobiology*, vol. 4, pp. 219-227, 1985.