This is to certify that the
thesis entitled

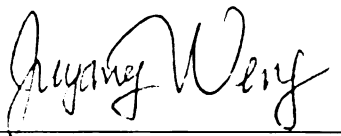
Autonomous Mental Development in High Dimensional and Continuous State and Action Spaces and its Application in Autonomous Learning of Speech

presented by

Ameet V. Joshi

has been accepted towards fulfillment
of the requirements for the

M.S. degree in Electrical and Computer Engineering



Major Professor's Signature

Wednesday, May 07, 2003

Date

Autonomous Mental Development in High Dimensional and Continuous State and Action Spaces and its Application in Autonomous Learning of Speech.

By

Ameet Joshi

A Thesis

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Master of Science

Department of Electrical and Computer Engineering

2003

Abstract

Autonomous Mental Development in High Dimensional and Continuous State and Action Spaces and its Application in Autonomous Learning of Speech.

By

Ameet Joshi

Autonomous Mental Development (AMD) of robots opened a new paradigm for developing machine intelligence, using neural network type of techniques and it fundamentally changed the way an intelligent machine is developed from manual to autonomous. The work presented in this thesis is a part of the SAIL (Self-Organizing Autonomous Incremental Learner) project which deals with autonomous development of humanoid robot with vision, audition, manipulation and locomotion. The major issue addressed here is the challenge of high dimensional action space (5 to 10) in addition to the high dimensional context space (hundreds to thousands and beyond), typically required by an AMD machine. This is the first work that studies a high dimensional (numeric) action space in conjunction with a high dimensional perception (context state) space, under the AMD mode. Two new learning algorithms, Direct Update on Direction Cosines (DUDC) and High-Dimensional Conjugate Gradient Search (HCGS), are developed, implemented and tested. The convergence properties of both the algorithms and their targeted applications are discussed. Autonomous learning of speech production under reinforcement learning is studied as an example.

ACKNOWLEDGMENTS

I would like to express sincere gratitude towards my thesis advisor, Dr. Juyang Weng for his continuous encouragement in my research work as well as expert guidance. Right from the beginning he nourished the passion towards the research. Working with him and pursuing the goals and finding solutions was a joyful experience.

I would also like to thank my fellow lab mates: Yilu Zhang, Nan Zhang, Xiao Huang, Yi Chen, Micky Badgero, Shuking Zheng, David Cherba, Raja Ganjikunta and Gil Abramovich. Working in this enthusiastic and amicable group was fun. I also had many interesting conversations with them which led to lot of new ideas.

I would like to express deep regards towards my beloved father, Mr. Vijay Joshi, mother Mrs. Madhuri Joshi and brother Mandar, who, in spite of being far away from me, made a deep contribution in this thesis by continuously encouraging me and making me believe in my abilities. This work was not possible without their support.

I would like to express sincere regards towards my thesis defense comity members, Dr. Fernanda Ferreira and Dr. George Stockman. Dr. Ferreira's insightful suggestions and comments helped me correct the thesis report on the basis of psychological

development and Dr. Stockman's subtle remarks helped me understand the different perspectives of the topic.

Last but not the least, I would also like to thank many of my friends and roommates with whom I shared a valuable time in the form of long discussions regarding my research work and who always encouraged me towards the research.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 What is Intelligence?

The concept of intelligence has always been an issue of heated debate among the philosophers all over the world. There are hundreds of definitions of the word intelligence available, most of which are the result of playing around with words and the nuances of their meanings. The common thought that flows in all of them is that the intelligence has to be based on the skills acquired by experience. The type of skills vary to a great extent from humans to other animals of birds. With the emergence of machines and their revolution and culmination into the birth of a computer has given a new direction to the definition of intelligence. Now it is not just human intelligence that we deal with but also the intelligence of the machines.

Computers possess very high raw computing capabilities, which are far beyond human scope. This fact leads to the interesting comparison of human intelligence with machine intelligence. Although these machines possess some powerful comput-

ing capabilities, they fail miserably when dealing with trivial things. There lies an important distinction between the type of tasks computers find easy to do and extremely difficult to do. The tasks are called as "muddy" tasks when humans find them trivial, but computers find them extremely difficult. For more information about the muddy tasks readers are suggested to read [34]. However it is undeniable that these machines do possess "something" which is better than humans. If someone wants to make a firm statement about the comparative intelligence of a human and a machine a clear definition of intelligence is customary. The concept of "Artificial Intelligence" has evolved in this quest of finding the solution to this problem. The definition of Artificial Intelligence as given by 'Alan Turing', a pioneering scientist and mathematician, in 1950 is considered as the most appropriate even till today by many people. Turing's test is defined as, *"You are talking to "somebody" behind the curtain and if, after talking to that "somebody" you feel that the "somebody" is a human then that "somebody" is intelligent.* This definition assumes that when we feel something as 'human' it is intelligent and associates the intelligence absolutely with each and every human behavior.

Due to the wide popularity of the phrase "Artificial Intelligence", the word was misused to represent the family of softwares which deal with complex logical computations and complex decision making processes. These softwares were called intelligent softwares or expert systems. The performance of these machines was impressive and they superficially used to appear 'intelligent', as these machines could do something which only an expert person in that field only could do. These software were written by experts who already knew how to solve a class of problems, hence the intelligence

shown was really a human intelligence. The program just used to act as an imitator and do its job. There was no new task that an expert could not solve and the machine could.

In order to make a system Artificially Intelligent it should pass through the above mentioned Turing's test. The most remarkable ability of humans is to learn new things from the knowledge of old things. We, can give birth to children who can learn new things that parents do not know, but we cannot, till date, make a machine which can outperform the programmers in learning new things. All the intelligence of the machine is present at the beginning of the system. With acquisition of some extra information the performance might improve, but there is no significant change the behavior of the system. The developmental machines proposed in this thesis do not have any expertise in any specific task. All they possess is the the capability of autonomously generating the symbolic representation of new things.

The developmental learning algorithm proposed by Weng [34],[32] is based on these considerations. The difference between the evolutionary algorithms and developmental algorithm can be stated as: the evolutionary theories are based on mutations or natural selection, most of which occur by accident or by chance, while the developmental algorithm is based on human "development" from childhood to adulthood. Another most discriminating factor in the developmental learning is the task non-specific nature of learning. None of the behaviors is pre-programmed by the programmer [33]. The SAIL project is aimed at making a humanoid using this algorithm.

1.2 SAIL Robot Project

SAIL stands for Self-organizing Autonomous Incremental Learner. This project deals with the creation of robotic machines which are capable of learning autonomously. Any human being can learn various behaviors in his physical capabilities by getting appropriate training throughout life while conventional machines, after their manufacturing, are just reproducers. The goal of this project is to break this hardwired concept about machines and develop robotic machines that are capable of learning after their creation. The robot has various sensors that are similar to humans or other living organisms and is capable of continuously acquiring information from them; store it and process it. The work reported in this thesis is part of this project and, hence, is also based on the developmental learning framework.

Speech is one of the most difficult modality even for a human baby to learn, as can be seen from its development. An average human baby takes about a year to speak the first word while by this time the development of the other organs is so advanced that it can even dance to music, wave a hand, and even play meccano [18],[20]. The properties of speech that make its learning difficult include the complicated process by which sound is generated from our vocal tract. The extent of variation in the production and sound of the same utterance even by the same person. Another aspect that becomes prominent in speech learning is overlapping of sensory information with its own action. In the early childhood the baby does not have a sense of separate identity. The fact that the baby is a separate individual and it is not a part of it's parents is also a learned concept. Hence, to separate its own utterances from surrounding

sounds is also something that a baby needs to learn from experience. There are no well defined actions in speech the way they are in locomotory organs. All these aspects reflect in the current development of robots and other intelligent systems. Thus it was expected that the performance of this system will be slow compared to the other locomotory and vision related autonomous systems being developed.

1.3 Autonomous Mental Development (AMD) and Speech

The Autonomous Mental Development (AMD) [33] paradigm is the heart of the whole system. It is characterized by:

1. **The existence of the body.** The body will possess intelligence and hence intelligence is not purely conceptual. Most traditional work in the field of AI assumes the system as a pure computer program which can solve complicated problems. The task is well defined and the input to the system is given electronically. However, with the context of a developmental robot, the first difference that needs to be considered for the development of the algorithm is the existence of the body and the system should be designed to optimize the performance that can be obtained from that body.

2. **The developmental program.** This program knows the capabilities of the body and can generate symbols for the perceptions obtained by the sensors on the body. This part is also present in the traditional programs.

3. **Birth.** This is again an important difference, as the real development of the machine starts after the birth. With the start of the program the machine becomes alive or the robot is born. It starts interacting with the environment and starts learning.

4. **Development.** The acquisition of the rewards from the environment facilitate the development of the robot and it starts producing actions that give him more and more rewards. This marks the final objective if the AMD.

In the view of AMD the body that is considered in this thesis consists of mouth in the form of multimedia speakers and an ear which is in the form of microphone. The brain of the robot and the developmental learning program is in the form of software. With the birth of the robot, it starts blabbering random utterances and also responds to the sounds from the environment. The rewards given by the environment help improve the utterances based on the context in which the reward was acquired.

The research focus of this thesis is AMD learning in high dimensional context and action spaces using generalized reinforcement learning mechanisms. The autonomous learning of speech by robot is taken as a challenging application to test the methods developed.

Chapter 2

Background

The system developed in the thesis is based on three main topics, 'Speech synthesis', 'Reinforcement Learning' and 'High Dimensional Search'. This section describes the three topics in detail. The early development of the language production in humans from the perspective of cognitive psychology is also essential for understanding of the system. Before discussing these details the objective of the this work is presented. This work deals with learning of primitive vowels only and is not aimed at learning of consonants or words.

2.1 Early Development of Language in Humans

The 'how' of the development of a child in its early months up to first year is one of the most puzzling question that all the psychologists and cognitive science people face. The absolute understanding of this development, if it is possible, may solve most of the mysterious problems faced by the psychologists and cognitive scientists

and also can lead to the production of real life humanoids. As this work is aimed towards making a robot, which can learn to produce sounds like a human baby, the development is viewed with the perspective of language acquisition in infants.

The baby starts crying and is also capable of listening immediately after the birth. However, the linguistic development takes place much slower than the development of other behaviors. The primary reason behind this can be given on the basis of the structural development hypothesis [10], [6]. According to this hypothesis, there are distinct levels in the behavior shown by the babies. The operations can be labelled as first order and second order. The first order operations include the logico-mathematical operations and physical operations. The logico-mathematical operations deal with reasoning out of the perceptual information. The infants try to correlate the observed phenomena. The physical operations include the movements that relate with the causality and the cognition of the space, time and objects. The infant starts building its understanding of the world from these elementary operations. The structural development also states that the development is recursive. The abilities obtained from the first order operations are used to generate the second order operations through experience and repeated use of the acquired behaviors. The high level linguistic development comes in the second order development. The understanding of the words and associating them with some physical objects is carried out in this stage. The words are assigned some meaning at this stage.

The later development in the language acquisition deals with the understanding of the grammar. This is an extremely complex process and is not discussed in detail here due to limitation of the scope of this thesis.

The initial vocabulary of the infant is marked with some meaningless blabbering. These utterances are produced with certain specific movement of the lips and the inner sound producing parts. The motion is governed by the amount to which the baby can stretch its organs. These utterances are shaped in later stages to generate the simple words like "Mommy" or "doggy". The shaping of the primitive words is the main focus of this thesis. The scope limited to five vowels, '/a/' as in 'car', '/e/' as in 'bell', '/I/' as in 'bit', '/ae/' as in 'hat', '/u/' as in 'put'.

2.2 Speech Synthesis

The field of speech synthesis finds its origin at the beginning of the twentieth century. From early 70s the real efforts towards the implementation of a machine or IC capable of synthesizing some form of speech were visible [2]. The speech synthesis was studied with different approaches and depending on the approach different methods were devised. Although I have discussed them briefly here, it is beyond scope of this thesis to describe all the approaches and methods, but an interested reader can find them from, [26] and [16].

The approaches to speech synthesis can be broadly classified on the basis of (1)Target Vocabulary, (2)Target "Intelligibility-Quality" tradeoff and (3)Consideration of biological and psychological aspects of speech production by humans. The third criteria primarily differs from the former methods on the issue of purely synthetic methods versus biologically motivated methods.

2.2.1 Classification based on Target Vocabulary

The target vocabulary can be limited to few words or it can have a large set of words. In the former case the system can be customized for the required utterances and system design is simple at the cost of loss of generalization. However if the target vocabulary is large then the customization does not work and more sophisticated model needs to be used. The systems targeted towards small vocabulary generally use synthesis by rule technique and have a database of the utterances. The details of such algorithms are discussed in [7].

2.2.2 Classification based on Target "Intelligibility and Quality" tradeoff

Some systems are required to produce voices which are supposed to be understood as commands. For example, the voices in the automated telephone answering machine. In such cases the quality and the fidelity of the sound are not important but the only requirement is to have an utterance which can be appropriately 'recognized' or in other words, the utterance should have good intelligibility. Other applications might demand for a good quality sound. For example, a machine capable of singing songs like human singers. With small vocabulary good quality of speech can be produced at a reasonable cost, but with large vocabularies the cost of the system can be astronomic if it is implemented using similar techniques [4].

2.2.3 Classification based on Synthetic versus Biologically motivated methods

For simple systems, discussed in the previous two classifications, the purely synthetic methods can be effectively used. Synthetic methods refer to algorithms which consider speech data as data obtained from a general non-stationary statistical process. These methods do not take into consideration the processes involved in the production of sound by humans. The data is then modelled using different techniques like Linear Predictive Coding (LPC) [2], Cepstral or Homomorphic Analysis [23], Vocoders [19], Formant Synthesizers [19], Hidden Markov Models (HMM) [19], etc. Although theoretically speech is a non-stationary process, for all practical purposes it is assumed to be stationary for a small time factor of 'twenty milliseconds'. Exploiting this limitation, these systems use a sampling window of twenty milliseconds and encode the data in each window. To make the operation smoother, overlapping windows are also used, e.g. hamming windows. The templates corresponding to each window are used for representing the utterances. For high quality sound representation the amount of data needed for encoding is enormous and the systems cease to be practical for large time requirements. The biologically motivated methods are more effective in these situations. These methods try to model the human vocal tract with a "source and filter model". The sources are models of the air puffs generated inside and the filters are models of the various organs in the auditory channel. To model a human vocal tract it is essential to have a complete understanding of the acoustics involved in human sound production. The structure of our vocal tract is shown in Figure 2.1.

The air puffs originating in the lungs form the source for sound production. It is then modulated as it travels through the various organs in the vocal tract until it comes out of the mouth. The entire column serves as filter for sound modulation. The nature

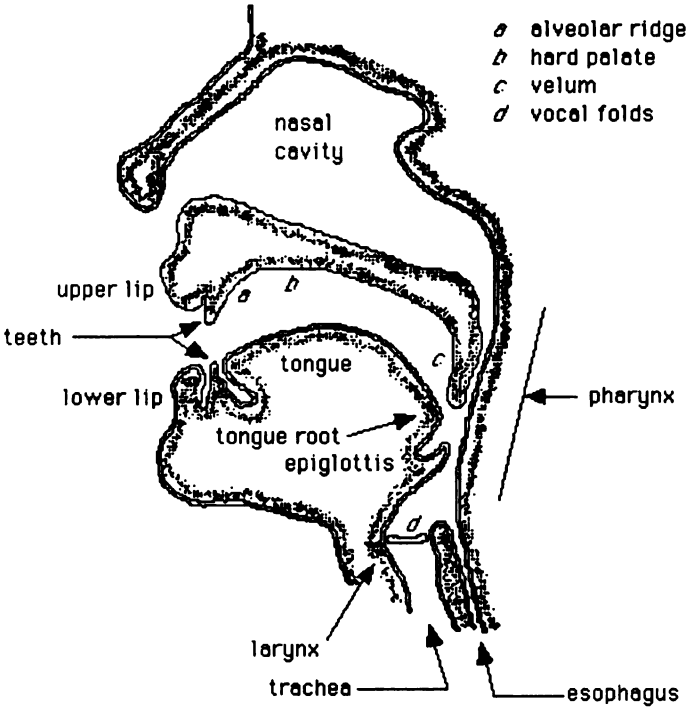


Figure 2.1: The vocal tract structure

of acoustics needed for modelling is complicated and requires an involved analytical computation and approximation to arrive at a set of parameters which represent the structure of the vocal tract. Obtaining the correspondence between the parameter values and the generated utterances remains after establishment of the model.

In view of the objective of this thesis, choice of biologically motivated method was mandatory. The most widely accepted model of the vocal tract, developed by Klatt

[1], is used in this project. The details of this model are discussed in the next chapter.

The Klatt model has been in use since 1988 and since that time many systems have been developed based on it. The current trends in speech synthesis revolve around such combinations [16] and also use psychological information in generating more human-like speech. Some of the modern systems use extremely sophisticated methods, targeted towards obtaining generalized results in multilingual speech synthesis [27].

In the background of existence of such systems, one might question about devising a new method to implement speech synthesis. However, this thesis should not be confused with a pure speech synthesis system. The objective of this thesis is to have a developmentally learning system like a human baby, which is capable of learning different behaviors based on the utterances of the five vowels. These behaviors can be unknown to the programmer and are designed by the teacher. The algorithm is task independent and speech synthesis is just an application of it. The following discussion, deals with reinforcement learning.

2.3 Reinforcement Learning

The reinforcement learning is the most intuitive way of learning based on experience. It is also the commonly observed learning mechanism in living organisms. In simple terms, the reinforcement learning means mapping the usefulness or effectiveness of actions to the situations, based on the rewards obtained. One interesting aspect of this learning is the delayed acquisition of rewards. This may not seem absurd in real life, as we are used to it, but in the background of computer algorithms, this makes

the problem quite complex. After an action is taken by machine a reward is obtained from the environment which is to be used to improve the action, which is already taken in previous context. Hence its effect can only be seen when the same context repeats. When the same context repeats the machines identifies its similarity with the previous context and also recollects the action taken before and the reward obtained in response. Using this information the machine then improves its performance. In a fundamental reinforcement learning experiment, we have three objects, the learner, the environment and the rewards, as shown in Figure 2.2. The steps in learning can be listed as:

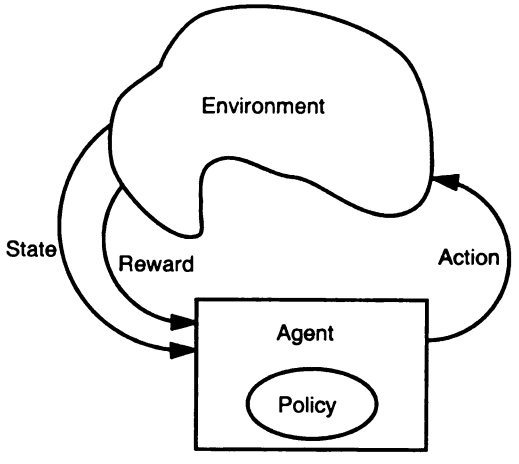


Figure 2.2: The fundamental model of reinforcement learning.

2.3.1 Reinforcement Learning Algorithm

1. The agent senses the current state of the environment and performs some action.

2. A reward is given to the agent from the environment in response to the action.

3. The action by the agent changes the state of the environment, also the environment is active itself and it can change its state on its own on top of the changes made by the agent. As a result a new state is created.

4. Go to step 1.

This cycle continues indefinitely. In most of the practical situations, there lies a fixed goal in the form of some desired action or a set of actions for given state or states. The rewards are generated in such a way that the actions taken by the agent keep improving and then ultimately they converge to the desired ones.

The mathematical modelling of the state and action spaces is the most crucial aspect of reinforcement learning. In a simple problem the number of states is small and the number of actions the agent can take can be modelled with few integer numbers. Maze solving problem is a typical case in such problems. Most of the research in this area assume these conditions and many strategies are devised and are tested in this area [13], [30]. However many times the situation is not as simple and the number of state spaces is large [14] or the space might also be continuous [17] and real valued. The continuous and real valued case is the most challenging as in this case we cannot have a discrete set of actions. Hence either a quantization has to be imposed on the state space or the whole search algorithm needs to be implemented in a different way. If in the case of continuous state and action spaces, the dimensionality is also high the problem becomes more challenging. Very few research examples are observed who have tried to tackle this problem [15]. The problem considered in this thesis has one of the most general framework of state and action spaces. The search

dimensions also suffer from inter-dependencies. This is discussed in more detail in chapter 4. Many times the same situation is described using Bayesian Network(BN) or Markov Decision Process (MDP). Few interesting examples of reinforcement learning with this framework can be found in [14],[3]. Although all these models model similar situations the mathematical framework is entirely different in them.

The reward can be obtained in variety of ways. It can be boolean in the form of 'Good' and 'Bad' or sometimes it can be tristate with possibility of no reward or zero reward. However sometimes the situations demand to have more information in the rewards and then we can have integral or even real numbered rewards. For most cases and especially for the situations where learning lasts for long time or for large number of iterations, the boolean reward is sufficient. When the process is long lasting the improvement in each iteration is also proportionally less, consequently the effect of the reward in the improvement is also less. In other words the reward carries lesser information.

The reinforcement learning is generally defined by the policy used by the agent, to improve the mapping of the actions to the states based on the rewards obtained. Although the nature of state and action spaces is variable the objective of the policy is unique and can be stated as "to reach the target as soon as possible". Although this goal is apparently simple it needs to be defined as a mathematical expression in order to implement it using computers. This conversion is not at all obvious. There were many attempts made to fix this, but every approach seems to lack something [30]. The most widely accepted goal is to maximize the long term rewards. The policy which achieves this is called as optimal policy. The definition of the long term

Table 2.1: The simulation of Reinforcement Learning

Iteration No.	State	Action	Reward
1	S1	A1	0
2	S1	A4	0
3	S4	A3	0
4	S3	A2	B
5	S2	A1	0
6	S3	A1	B
7	S1	A1	0
8	S3	A3	G
9	S3	A3	G

reward is also relative to the task at hand. Generalizing this statement, if we want to maximize the reward in next h iterations, then the optimal policy can be defined as the one which maximizes, $E(\sum_{t=0}^{h} r_t)$. A simple numerical example of reinforcement learning will make the whole discussion easier to understand.

2.3.2 Numerical example of reinforcement learning

Consider a system with six states denoted as 'S1' to 'S6' and four possible actions as 'A1' to 'A4'. The reward is tristate in nature, with 'G' for good, 'B' for bad and 'Zero (0)' for no reward. The objective is to train the agent to take action 'A3' in the state 'S3'. A sample learning is shown in table 2.1. The policy used here is choose actions randomly and avoid actions which received 'Bad' reward before.

The table illustrates the reinforcement learning as it occurs in discrete state and action spaces with specific target available. When we start generalization of the algorithm by making the state and action spaces continuous and hence infinite and then further making the availability of the goal obscure, the design becomes more and more complicated. Devising fast learning and also consistently convergent policy

in this situation becomes challenging.

2.3.3 Mathematical model of reinforcement learning

With reference to Figure 2.2 the mathematical model of reinforcement learning is now presented. Let "s" denote the current state of environment. Let "a" be the current action taken among the entire set of actions "A" available and "r" is the reward obtained by the agent in response to it. "s'" be the next state of the environment. The action chosen by the agent is the result of a certain policy that is being followed by it to decide a particular action among the set of available actions. Let us denote the policy as "Φ". Now as this policy is function of the set of possible actions and also the current state of the environment let us denote it as "$\Phi(s, a)$". In order to choose one of the possible actions and also to update their probabilities of occurrence based on the rewards obtained it is necessary to have a value system associated with the actions. These are commonly called as "Q" values. As there is a "Q" value associated with each action there exist a one-to-one mapping between them and we can we will use notation of $Q(s, a)$ for "Q" values. Thus the updated notation for the policy is "$\Phi(Q(a, s), a)$". With this understanding of the system setup, we can write the symbolic mathematical formulation of the learning algorithm as $Q'(s, a) = f(Q(s, a), r)$, where $Q'(s, a)$ is the updated set of Q values from the reward r.

Among the different variations of the above formula we will be considering only the algorithm called Q-learning first designed by Watkins in his PhD dissertation [5]. The information about the other variations and their scopes can be obtained from

[30].

2.4 Q-Learning

We have described the process of Q-learning in a symbolic way with the introduction of the policy of the action choosing and the function of Q-value update. The Q-learning is also called as an Off-Policy TD control. The value update rule is given in equation 2.1.

$$Q(s,a) = (1-\alpha)Q(s,a) + \alpha(r + \gamma\ max_A\ Q(s',a')). \tag{2.1}$$

The max_A in equation 2.1 means the Q-value of the action having maximum Q-value in the state s'. Let us call this value as maxQ. Here the usage of the future Q-values might seem to be absurd, but as this whole process is iterative and the same states are going to be visited again and again the future values for current states might already have occurred before. This makes the use of future Q-values justified. In the beginning of the process all the Q values are initialized to zero. 'α' is called as learning rate. There are no theoretical ways to decide the value of this learning rate, but most of the time a positive value which is very close to zero is used, e.g. '0.1'. This equation can be interpreted as, the current Q-value is updated by adding the difference between weighted maxQ and the current Q-value of the chosen action and the reward, both weighed by the learning rate to the original value. In this algorithm only one future reward r and one future Q-value is used to update the current Q-value $Q(a,s)$. This concept can be extended and multiple future discounted rewards and Q-values can be used to update current Q-value $Q(a,s)$. For proof of the convergence

of this method and its variations can be found in [30]. The former methods are called TD(1) while the generalized methods are called as TD(λ) methods.

2.5 Exploration and Exploitation

The choice of maximizing the immediate reward always points towards the action with maximum Q-value. If we just keep on taking that action, the system will soon be stagnated at a local optima and there will be no scope of improvement. In order to take other than optimal action we need to choose some action which has a Q-value less than the maximum. This raises a question as to which one to choose among a set of actions all of which are not optimal. The second best or third or the worst? This question is vague due to lack of information and most of the time the answer is to choose any action arbitrarily or randomly. The method of choosing the optimal action is called as exploitation and the method of choosing the action non-optimally is called as exploration. In order to have the system working well and reach the target in desired time we need to have suitable tradeoff between exploration and exploitation. The exploration always aims at reaching the absolute optima while exploitation uses the learned behavior and tries to give optimal performance in given situation, which might not be the globally optimal performance.

The considerations about the choice of exploration and exploitation are based on the situation where the learning is to be performed. Broadly the strategies of exploration can be classified into two types [31]: (1)Directed and (2)Undirected.

2.5.1 Directed Exploration

These methods exploit the process or task specific information and thus are more efficient. Mostly these methods give superior performance in small time duration tasks with fast convergence requirements [24]. However they do not have generalization capabilities.

2.5.2 Undirected Exploration

There are basically two types of undirected methods, (1)Random walk and (2)Boltzmann Exploration.

Random walk

The random walk exploration is purely random process as the name suggests and hence not useful in situations where the system needs to settle after some time even if the absolute optimal solution is not found.

Boltzmann Exploration

Before dealing with the mathematical formula of the Boltzmann exploration, it is essential to know the underlying concept. This method starts with almost random exploration and all the possible actions have equal probability of occurrence. However as time progresses the system starts giving more and more preference to the actions with higher Q-values. As the system learns it starts choosing the actions leading to optimal learned solution more often. Thus this method makes a smooth transition

from the learning phase to the testing phase. The details of the method are discussed in the following subsection.

Let us denote the probabilities of all the "n" actions as '$p_i, i = 1\ to\ n$' and Q-values as '$q_i, i = 1\ to\ n$'. Then the probabilities are generated based on following equation.

$$p_i = \frac{exp(q_i\theta^{-1})}{\sum_{i=1}^{n} exp(q_i\theta^{-1})} \tag{2.2}$$

θ is called as Boltzmann temperature. Essentially this factor controls how much importance the Q-value will get in deciding the individual probability of each action. The more the value of the temperature the lesser the importance of the Q-values. The lesser importance of Q-values means that almost all the actions will get equal probability irrespective of their Q-values. This is illustrated in Figure 2.3.

When Boltzmann temperature is infinity, all the actions are equally probable. This is ideal random case. The importance of this method is that the parameter θ can be controlled programmatically and thus the randomness in exploration can be monitored. As the machine learns, the Q-values start getting closer to their optimal values. The tendency towards using the previously learned actions should be increased along with. Maturity of a machine can be a controlling parameter in the randomness of the exploration. This modification in the randomness that is introduced through the Boltzmann temperature has not originated from any task specific information and hence it is general.

After the creation of cumulative density functions (cdfs) for all the actions using Boltzmann exploration formula, a random number between '0' and '1' is generated and the corresponding action is taken. This method essentially gives one action

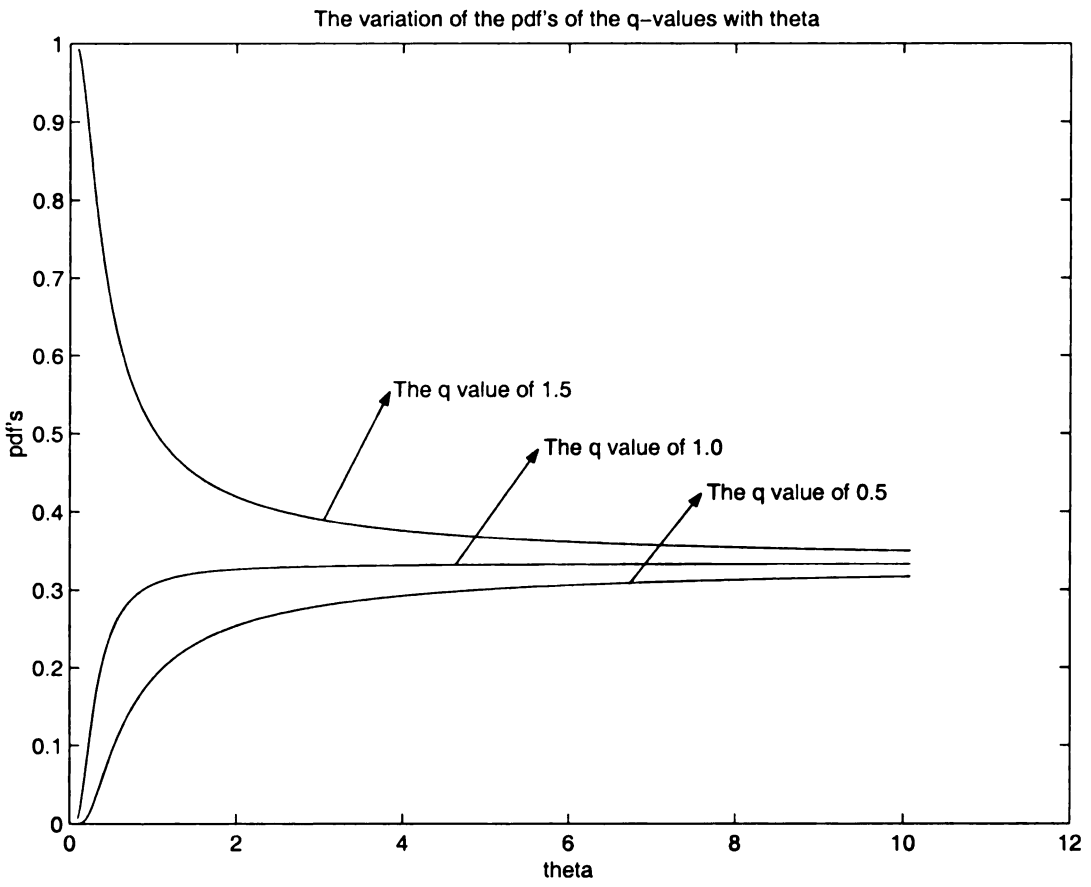


Figure 2.3: The variation in the probability distributions (pdf's) of the actions with change in Boltzmann temperature "θ".

among a set of actions from their Q-values. This limits the scope of this method to only discrete action space problems. When the action space is continuous and the number of actions is practically infinite, this method cannot be directly used. One way is to discretize the space and create Q-value system and then use the method, but in some cases and especially the one used in this project this was not feasible and hence a new system of exploration was developed. Though the mechanism is different it is based on the principle of slow maturation in time and based on that reduction in exploration probabilities. The method is explained in the chapter 4.

2.5.3 The Search Methods

In a generalized case of a learning environment, the state and action spaces are continuous and high dimensional. To have an efficient search algorithm, the problem of the curse of dimensionality that arises has to be tackled. The solution becomes more difficult when the data is unstable or sparse. The parameters used in this thesis for speech synthesis have sparsely distributed data and inter-dependency among the dimensions. The sparse distribution of data in reduced dimensionality is shown in Fig. 6.1. The inter-dependency in the dimensions arises due to the fact that the five formant frequencies need to have monotonically increasing values. I did not find any reference to this type of problem in the literature that I read during the two years of the development. Few had dealt with high dimensional cases, but the work that comes closest to mine is [15]. However, in this referred paper, the main objective was to solve the maze-like problems with well-defined goals. They did not have the problem of the sparseness of the data and dimensional inter-dependency.

The problem is tackled with two approaches, (1)Direct Update On Direction Cosines (DUDC) and (2)High-Dimensional Conjugate Gradient Search (HCGS). Both algorithms are discussed in detail in chapter 5

2.6 Developmental Learning

The most important aspect of developmental learning is that it tackles the problem of computational representation of the cognitive learning observed in human infants. Autonomous Mental Development (AMD) framework is capable of producing typical

classical conditioning described in the cognitive psychology [34].

The AMD mode of operation discussed in chapter 1 is the building block of the system. The most distinguishing property of the developmental program is that the programmer does not know the tasks the the robot ends up learning after the birth. Therefore a developmental program must be able to generate internal representation *on the fly* for virtually any task. The capability of the machine is developed through real time interactions with the physical world. It depends on five constraints: (1) sensor, (2) effector, (3) computational resource, (4) developmental program and (5) the way robot is taught. Before going into the details of the system architecture the main challenges that AMD tackles are discussed.

2.6.1 The eight challenges of AMD

1. Environmental openness: Due to the task non-specific nature, AMD must deal with unknown and uncontrolled environments, including various human environments.

2. High-dimensional sensors: AMD should be capable of dealing with continuous digitized signals coming from the different sensors. The problem of curse of dimensionality arises here which needs to be tackled.

3. Completeness in using sensory information: There is no way the robot can determine which information is more useful and which is not. Hence it must try to utilize all the information that is available to it. All it can take advantage of is the statistical similarity in it.

4. Online processing: The robot itself is affecting the state which is sensed by it. Off-line processing is unable to accomplish AMD.

5. Real-time speed: The sensory/memory refreshing rate must be high enough so that each physical event (e.g., motion and speech) can be temporally sampled and processed in real time (e.g., about 15Hz for vision). The notion of pseudo parallelism explains the duration of one second that is used in this thesis.

6. Incremental processing: Acquired skills must be used to assist in the acquisition of new skills, as a form of "scaffolding." This requires incremental processing. Thus, batch processing is not practical for AMD. Each new observation must be used to update the current complex representation and the raw sensory data must be discarded after it is used for updating.

7. Perform while learning: Conventional machines perform after they are built. An AMD machine must perform while it "builds" itself "mentally."

8. Muddy tasks: For large perceptual and cognitive tasks, an AMD machine must handle multi-modal contexts, large long-term memory and generalization, and capabilities for increasing maturity, all without catastrophic memory loss.

Chapter 3

Working of the Organs

The system comprises of three main organ simulations. 'The brain and associated memory' and 'the ear' and 'the mouth'. Before going into the details of the working of the system as a whole the individual organs (ear, mouth and memory) and their functionality is discussed in this chapter.

3.1 Working of Ear

The ear is a purely sensory organ and does not have any action associated with it. The function of ear is to collect the speech data as obtained from the multimedia sound card and encode it into some parameters, which can then be processed. The human ear perceives the sound information through the vibrations on the diaphragm and eventually it is converted into some form of electrical signal which is then carried to the brain through neurons. The choice of encoding is based on few criteria like the capability of representation and compression power. There need not be any hard and

fast rule about choice of encoding. Also it is not quite essential from the conceptual point of view that the parameters should be similar to what humans use. However, it is still unknown to us how exactly the encoding of speech signal works in human ear.

Out of the other possible alternatives like Linear Predictive Coding (LPC) parameters or the formant frequencies, the cepstral parameters were chosen. The main advantage of these parameters is their ease of computation.

The processing of the raw speech to cepstral parameter extraction is now explained in detail.

3.1.1 Cepstral parameter encoding

The speech is sampled at 11.025 Khz and with eight bit resolution for each sample. Thus one second duration of speech contains 11025 samples. A hamming window of size 256 is used to extract the cepstral coefficients. Instead of using separated hamming windows the neighboring windows are overlapped. An overlap of 56 samples is maintained in order to have smoother performance. Hence the effective window size is reduced to 200 samples as can be seen from Figure 3.1. The 200 samples correspond to precisely 18.14 milliseconds. The fastest response time for human ear is experimentally found to be near 17 milliseconds [19]. One set of cepstral coefficients contains 13 parameters. Thus, one second sample of the raw speech data is converted to total of 55 sets of cepstral coefficients, or total of 55 x 13, i.e. 715 cepstral parameters.

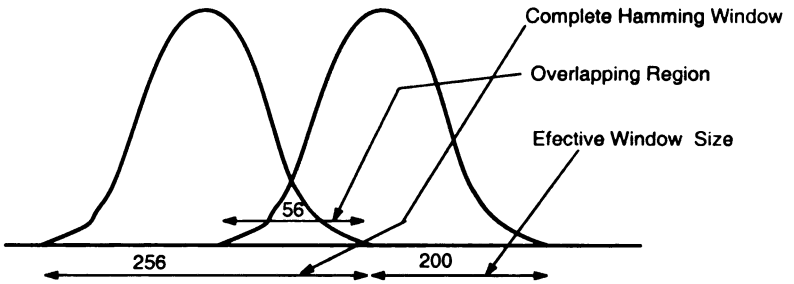


Figure 3.1: The hamming window schema.

These parameters are kept in a queue and as new parameters arrive, the queue is updated. The old items are discarded from one end and new items enter from another. The brain keeps track of this queue and acquires the information after the end of every one second. This forms the dimensionality of the external sensory information as 715.

3.2 Working of Mouth

The speech producing organ is the only active action generation organ or effector in the system. The speech synthesizer developed by Klatt, known as KLSYN88 [21], [1] is used. There are total of 40 parameters in the original model developed by Klatt. This model was later enhanced by Stevens and Bickley [28] who came up with total of ten constraints which were able to control the actual forty parameters. These ten parameters were called as 'High Level' parameters of HL parameters. This model is not an artificial speech generation model, but is entirely based on human vocal tract. These ten HL parameters can be broadly divided into two categories. Five of them represent the various physical dimensions of the vocal tract and its components and the remaining five represent the pitch and the formant frequencies. The structure of

vocal tract along with the HL parameters is shown in Figure 3.2

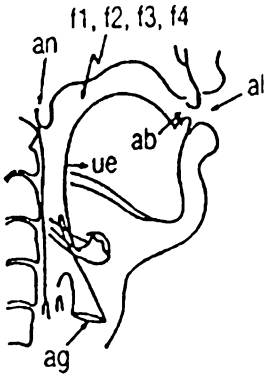


Figure 3.2: The vocal tract with HL parameters.

The block diagram of the KLSYN88 synthesizer is shown in Figure 3.3. The "Sensimetrics^{TM}Inc." used this model to develop a software called as 'HLsyn'. This software is used for this thesis. The program of HLsyn directly maps the HL parameters to raw speech data. This data can then be converted into wave file or played directly through program. The 'dll' file of the HLsyn is called directly from the main program to generate the sound from the HL parameters. The block diagram of HLsyn is shown in Figure 3.4.

3.3 Working of IHDR Tree

The associative memory is implemented using the HDR tree. HDR stands for Hierarchical Discriminant Regression. The incremental version of the HDR was then modified for incremental update as the data is acquired sequentially. This reduces the memory requirements drastically. The new version is called as IHDR. Here IHDR tree is described in brief as required for this thesis. Basically IHDR is a statistical

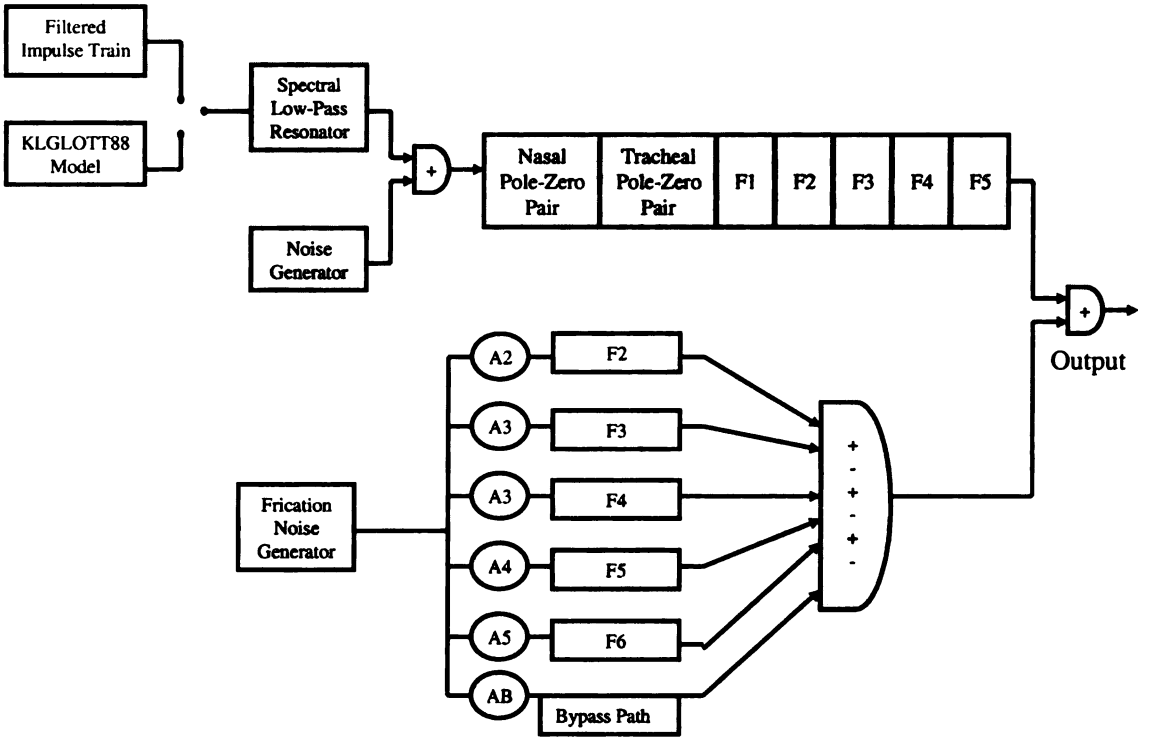


Figure 3.3: The block diagram of KLSYN99 synthesizer.

decision tree and its details can be found in [11] and [12]. The IHDR tree is a bimodal tree. One lobe of the tree (lobe A) is developed using the statistical or mahalanobis distance metric, while the other lobe (lobe B) is developed using euclidian distance. Two main operations can be performed on the tree, (i) adding a sample and (ii) retrieving a sample. The sample is always in the form of a set of two vectors, called $\mathbf{X}$ and $\mathbf{Y}$. Vector $\mathbf{X}$ goes to lobe A while vector $\mathbf{Y}$ goes to lobe B. During the process of adding a sample to the tree the sample is first searched for a nearest neighbor in lobe A. When a suitable match is found, it is added into that node. Then the corresponding node in lobe B is extracted from the mapping that is maintained with the development of the tree. Then the second vector is added to that position. As the samples are added to the tree only the statistics of the node is updated and only

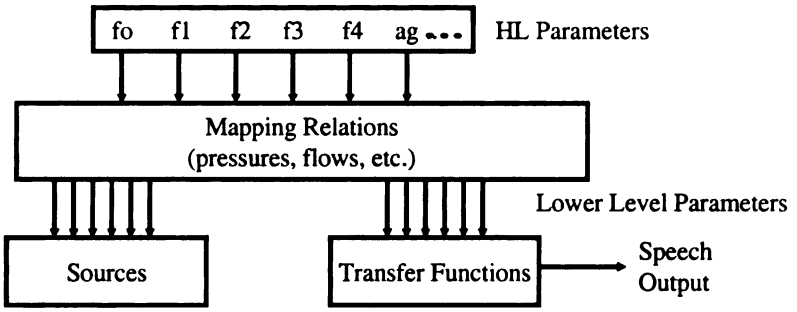


Figure 3.4: The block diagram of HLsyn.

limited number of samples are preserved. After a node is accumulated with sufficient number of additions the node is frozen and it spawns into child nodes. The number of maximum nodes that can be created in a tree is predetermined. In the initial phase the new nodes are created when the incoming sample has large distance from the pre-existing nodes. In the later cases when all the nodes are populated, the incoming sample is classified into one of the existing nodes.

The IHDR tree structure is capable of handling the seven stringent requirements that any regression tree should. The requirements are listed below.

1. It must take high-dimensional inputs, with unknown correlation between the components. Some input components might not be related to output at all.

2. It must perform one-instance learning. An event represented by only a single input sensory frame must be learned and recalled. Thus, iterative learning methods such as back-propagation learning are not applicable.

3. It must dynamically adapt to increasing complexity. It cannot have fixed number of parameters, like a traditional neural network, since the complexity of the desired regression function is unpredictable.

4. It must deal with the problem of local minima. Due to the online real-time learning requirement, the tree being built must be successful. Simultaneously, keeping multiple networks, each starting with a different random initial guess, and then selecting the best performing network, is not applicable to real-time online learning

5. It must be incremental. The input must be discarded as soon as it is used for updating the memory. It is impossible to save all the training samples since the space required is too large.

6. It must be able to retain most of the information of the long-term memory without catastrophic memory loss. However, it must also forget and neglect unrelated details for memory efficiency and generalization. With an artificial network with back-propagation learning, the effect of old samples will be lost if these samples do not appear later.

7. It must have low time complexity in computing and update so that the response time is within fraction of a second for real-time learning, even if the memory size has grown very large.

This outlines the development of the IHDR tree. When the sample is to be retrieved it is extracted using the vector $\mathbf{Y}$ from lobe B, using the euclidian distance. Then from the mapping, its counterpart, vector $\mathbf{X}$ from lobe A is extracted. Though the tree has two different lobes, different distance metric is used to classify them. Hence to exploit this feature both the vectors $\mathbf{X}$ and $\mathbf{Y}$ used in the sample are identical, and represent the context. The context consists of external sensory information

as discussed in 'Working of Ear' and internal sensory information in the form of HL parameters as discussed in 'Working of Mouth'.

Chapter 4

System Architecture and Objective

4.1 System Architecture

After dealing with the working of the individual organs, in this chapter the functioning of the 'system as a whole' is discussed. Although all the organs function as separate threads, a certain synchronization needs to be maintained. The issue of the true parallelism versus pseudo parallelism becomes important in this scenario. The next subsection discusses the need of having the pseudo-parallelism.

4.1.1 The Notion of Pseudo-Parallelism

As we are aware of our surroundings and at the same time we can listen to some music and also think about something and tapping hands on the table and also shaking legs we did not event notice the kind of parallelism that our body is performing. The brain is getting flooded with data from all the sensory organs which needs to be processed and based on that responses are to be generated for all the effectors

which will implement it. These processes are going on continuously, however we are unaware of it. However when it comes to design a system which is aiming at imitating human, it is customary to have clear understanding of these processes. Although we say our body is performing all the actions simultaneously, the brain is unique. It can be argued that there are multiple parts of brain and all of them are processing simultaneously (which is partly true also), but there can always be instances where the same part of the brain has to take multiple decisions at the same time. This is ideally impossible. However if the same processing part carries out a time slicing and take the different decisions one after the other, and if the time it takes for each one is sufficiently small the discontinuity would be unnoticed. In this case the actual processing is sequential but the observed type of processing is parallel. Also the notion of 'same time' is practically limited to the smallest portion of time as constrained by the flexibility of the organs. For example, gap of a microsecond in processing the information received by the ear is not going to make any difference in the resulting observed response. In other words we can say that all the observed parallelism in humans is limited by a certain small fraction of time. This emphasizes the fact that all the observed parallel human activities are quantized. In reality, these actions might be carried out sequentially one after the other. This 'observed' parallelism is termed as pseudo-parallelism. Most of the times we do not really come across activities which occur at a speed near our quantization limits, however sometimes it becomes impossible to explain things without its notion. A classic example is of television. It makes us believe that there is a motion picture going on, when its actually a series of colored dots running serially left to right and top to bottom on the screen.

If we could sense the picture in infinitesimally small time the television would not have come to existence. For, a simple sequence of few seconds would require almost infinite amount of data. The objective behind stressing the importance of pseudo-parallelism is the fact that it is impossible to have a machine which will process the data infinitely fast. Also the machine developed in the thesis is to be executed on a standard PC, which has only one processing unit and also it has to execute the operating system processes and some other programs along with this system. Hence the processing mechanism involved is inherently pseudo-parallel. The primary goal of this thesis is to generate an algorithm which is capable of learning to produce speech like human babies in an interactive context based environment using reinforcement learning. After considering all the practical aspects of the speech production using multimedia computer a time quantization of one second is fixed for the system. This time quantization is apparently large and unacceptable for humans, but for the scope of this thesis this is sufficient. The working of the system in this constraint will prove the validity of the algorithm and then this time quantization can be arbitrarily reduced with use of more powerful systems of using dedicated processors.

The pseudo-parallelism in this system is implemented by parallel running VC++ threads. There are exactly three threads, one for each organ. As the sensory and effector organs are working the brain keeps collecting the data and at the end of one second it performs its action by making the decision. The decision is based on the external data obtained from the ear and also on the internal data obtained from the mouth. The context is formed based on both the data streams. The rewards from the environment can come at anytime and as and when they are available they are

recorded by the brain thread. The rewards are used by the brain along with the context for decision making.

The timing diagram shown in Figure 4.1 gives precise idea of the system behavior.

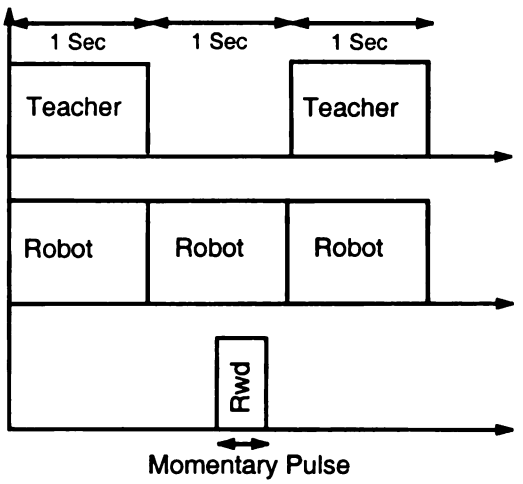


Figure 4.1: The temporal behavior of the system.

4.2 Context Based Learning

The learning is context based. The rewards are the only inputs given to the system to shape its behavior. The learning is non-supervised (the internal parameters are never controlled directly by the teacher) as an ideal human learning should be. With the reinforcement learning algorithm the system learns based on the rewards. The details of the context parameters and their acquisition is now explained.

4.2.1 Obtaining Context

The parameters taken from the ear are cepstral parameters and one set of cepstral parameters correspond to a duration of about 18 milliseconds. It is found experimentally that, although human ear can recognize sounds of a duration as short as 17 ms, most of the time it accumulates the sound of duration 200 ms [19] and then processes it. The detailed description of the functioning of an ear is given in the chapter 3.

The parameters used to model the mouth are HL parameters. Useful HL parameters are ten. There are actually 13 parameters in one set of HL parameters, as specified in the original model, but the parameters after ten are basically used to classify different speakers. We can use a constant set for implementing all of the sounds for a single person. Also, our robot is supposed to be a single person with some specific voice.

4.2.2 Normalization of the Parameters

There are total 715 parameters obtained from the ear and ten parameters obtained from the mouth. The context is based on the combination of both of them. Although there are lot more parameters from the ear the importance of them in deciding the context is equal to the ten parameters obtained from the mouth. The overall importance of the parameters should be independent of the number of the parameters and also on their values. For example, one set of parameters can have larger values and small quantity and vice versa. Hence, in order to remove the effects of these properties, the variance of each set is considered as the fundamental quantity and based on

that the parameters are normalized. The variance of both the sets is estimated by simulation data and the values are used in the program. In later course this can be done on the fly and the variance constants can be updated in each iteration.

4.2.3 Information Storage and IHDR

The most important aspect of the working of brain is the storage of the information. The storage and retrieval of the data from the memory is not sequential like traditional computer memory, which plays a key role in system architecture. It is not known till date the precise information about the information storage in brain. However, the associative learning observed in brain strongly points towards existence of a tree structure where the data is stored on the basis of its statistical similarity. The structure that is used in this thesis is called as 'IHDR' which stands for Incremental Hierarchical Discriminant Regression as discussed in chapter 3. As soon as the context is given to the brain, it searches the IHDR tree and tries to retrieve the nearest neighbor for that context. If a good neighbor is found, it is extracted. The choice of good neighbor is based on a threshold of the distance. The actions taken in the past in the similar context are retrieved along with their Q-values. Due to the continuous and real valued nature of actions, they are represented in the form of fixed number of sets of micro-clusters of similar actions or a single set of direction cosines along with a step size. The details about the creation of these micro-clusters and the direction cosines is described in chapter 5. The brain then uses this information and applies the policy of choosing action and generates a new conceptual action. The new

conceptual action is a ten dimensional vector in HL parameter space. The real action is then taken by the mouth. The policy of the brain to choose the next action is now discussed. This involves reinforcement learning and gradient search algorithms and also the tradeoff between exploration and exploitation.

4.3 Reinforcement Learning

The fundamental model of reinforcement learning and its algorithm are discussed in the chapter 2. In the traditional reinforcement learning model there is a distinct boundary between the environment to be served and the agent. However with the complex architecture of a humanoid robot, which is considered in this thesis, this model becomes inadequate. The humanoid robot has a physical body which consists of locomotory organs like hands, feet and mouth, sensory organs like ears and eyes and also has a brain that controls the organs. The brain is more of an abstract organ. In order for the robot to function properly its brain should have precise knowledge of the relative positions of the organs along with their limitations. During the response of the robot to the environment the conceptual action is generated by the brain and real action is taken by its organs. Hence, it is essential that the brain keeps track of the state of its own organs. Thus with regards to the brain the state of the organs is external, however this state is internal for the humanoid robot. This state information is called as internal sensory information and brain obtains it by communicating with respective organs. Figure 4.2 shows the picture of the modified reinforcement learning system that is used in this thesis.

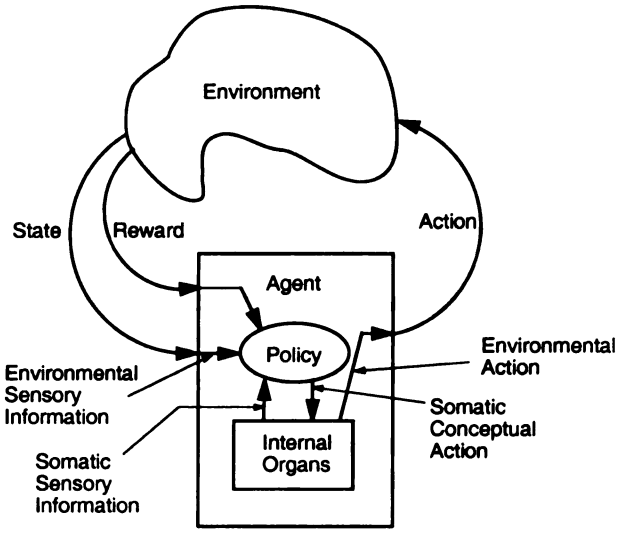


Figure 4.2: The modified reinforcement learning model used in the thesis.

The complete state information is generated by the combining the set of external state information and the set of internal state information. The external sensory state information is in the form of cepstral coefficients obtained from the ear and the internal sensory state information is in the form of the HL parameters obtained from the mouth. The total cepstral parameters are 715 as explained in chapter 3 and there are ten HL parameters, hence the total state space is of dimensionality 725.

Any reinforcement learning system has to deal with certain search space. The learning of the system is characterized by multitude of variations depending on the specifications of the search space and the different constraints imposed by it on the learning. The possible combinations of them are discussed below.

The search space can be 'n' dimensional, the 'n' can be small (in the range from one to three), or it can be large (more than three). The space can be discrete or continuous. In discrete case the number of actions in any state are fixed while con-

tinuous case the number of actions are always unlimited. With this discrimination we can have different possible combinations of the search spaces ranging from small dimensional discrete to high dimensional continuous.

One more aspect that often affects the learning is the various constraints imposed by the search space due to the practical limitations. The basic constraints arise due to boundaries of the search space in all the dimensions, as we cannot go from negative infinity to positive infinity in all the dimensions. The other limitation arises due to dependency among the various dimensions. For example, certain combinations of values are not possible to obtain practically, even all the values in each of the dimensions are within their individual specified range. This leads to a new concept of variable range in each dimension dependent on the other dimensions.

The other parameter that is crucial in the performance of the reinforcement learning method is the generation of rewards. The reward is the only guiding parameter that is used in reinforcement learning. The reward can be boolean ('1' and '0') or it can be tristate ('-1', '0' and '$+1$') or it can be quantized with more than three levels or it can be real valued. In the traditional reinforcement learning paradigm as given by the equation 4.1 any of these values can be used.

$$Q(s,a) = (1-\alpha)Q(s,a) + \alpha(r + \gamma\ max_A\ Q(s',a')). \tag{4.1}$$

The mathematical modelling of the Q-learning or any other variations of it is based on the ultimate goal of maximizing the reward. The time duration in which to maximize the reward is a crucial aspect in modelling the learning equation. In the cases with less time duration and also small dimensionality and discrete nature of search space

the reward need to be utilized heavily in order to improve while in the other extreme case when the search space is large and high dimensional and continuous and also having inter-dependency among the different dimensions, the time duration has to be large and the use of rewards is minimal. Most of the cases studied in this field focuss on the simpler cases and optimizing the learning performance in specific learning environments. Even the more general cases considered by many deal with optimizing the performance of the system to apply for some specific environment.

The current work started with the intention of developing a learning system to tackle the most general case in reinforcement learning with high (ten) dimensional and continuous state space. Each dimension having different ranges and also there exists an interdependency among the values in all the dimensions. The specific goal to be achieved is use this system to generate a robotic system capable of learning to speak like a human baby. There is absolutely no supervised learning and also the learning has to be general. The baby should learn to respond in the way the teacher desires. With the consideration of the most generalized case and also the imposition of not exploiting the task specific information which is not known to the baby, development of the algorithm is a very difficult task.

The system presented here has the objective to keep working for unlimited time. Hence, it should be capable of handling unlimited number of interactions. This goal is drastically different from the commonly used objective of achieving some desired behavior in limited number of steps. Hence it is important for the system to be capable of unlearning certain things learned in the past and also keep learning with the new data. The speed of learning is not very important criterion as time is not a

critical parameter, however system should continuously improve in certain direction, based on the interaction with the environment and in long run the program should converge to the target with probability of unity.

During the development of the system two different approaches and their variations are experimented and evaluated and the summary of each approach and its performance in specific cases is discussed in the next chapters.

Chapter 5

The Two Approaches

This chapter deals with the two approaches that are developed during the research. As described in the earlier chapters, the challenging issue in the research is to use the reinforcement learning model effectively in the areas which are more complicated that the assumptions of the traditional reinforcement algorithm assumes. In practical situations it is difficult to have a discrete set of actions which will be constant irrespective of the context state. However, the quantification used in reinforcement learning required it. The same applies to the existence of finite state while in reality there are infinite variations of states. Hence some modifications in the algorithm are mandatory. To tackle this problem two different approaches are tried. The details of the algorithms are discussed in the following sections.

5.1 Learning with Direct Update on Direction Cosines (DUDC)

The most important change in this method is removal of a set of actions and replacing it with a single set of direction cosines, "$\Psi(i),\ i = 1, ..., 10$". This set represents the attempt of the optimal direction in the given search space and in a given hierarchy of step size. The objective of the reinforcement learning is to find the optimal set of direction cosines. The set of direction cosines when coupled with the step size represents the action. Hence now we have just one set of action and the objective is to shape it using reinforcement learning. As this framework is different from the traditional learning a new mathematical model of learning is mandatory. Before explaining the new model of reinforcement learning it is essential to explain the exploration-exploitation tradeoff mechanism for this framework.

5.1.1 EMMP

The EMMP stands for "Exploration with Maturation with Multidimensional Perturbation". With the given set of direction cosines, '$\Psi(i)$', the concept of exploration is to change the given direction randomly, so that all the possible directions are equally likely. The concept of maturation comes with the tradeoff between exploration and exploitation. As the '$\Psi(i)$' are shaped with the reinforcement learning, the direct use of them is the exploitation. With the well known fact - 'exploration leads to finding the global optima and exploitation leads to using the learned local optima', the exploration is slowly reduced as the system matures. 'κ' is used as the quantitative

measure of the maturation. The mathematical model that is developed with this concept is described below.

$$\kappa = log(1 + \frac{\iota}{40}) \tag{5.1}$$

$$\Omega(i) = \epsilon + \Psi(i)\kappa \tag{5.2}$$

$$\xi = (\sum_{i=1}^{i=10} \Omega^2(i))^{1/2} \tag{5.3}$$

$$\Psi'(i) = \frac{\Omega(i)}{\xi} \tag{5.4}$$

'ϵ' is a real valued random number between -1 and $+1$ such that it can take all the values in the range with equal probability. 'ι' is the iteration number, which is incremented with the arrival of a context. The number '40' is chosen from the empirical evaluation of various other values. The first step calculates 'κ', the quantitative measure for the maturity of the system from the number of iterations. The next three steps describe how the new set of $\Psi(i)$ are obtained from the old ones using the EMMP method. The weights, denoted as '$\Omega(i)$' are the intermediate variables used before the normalization. 'ξ' is the normalizing variable. This method uses a variation of the exploration and exploitation tradeoff, which is based on the principles of the Boltzmann Exploration. The '$\Psi(i)$' represent the original direction cosines and the '$\Psi'(i)$' represent the new direction cosines obtained after doing the exploration.

The reinforcement learning model is based on the EMMP method and the equation is given below.

$$\Psi(i) \leftarrow \Psi(i) + \alpha r(\Psi'(i) - \Psi(i)) \tag{5.5}$$

The value of '0.1' is normally used for the 'α', which is also called as learning rate.

The EMMP followed by the reinforcement learning constitute first step in the entire hierarchical search mechanism. The complete action is obtained by multiplying the direction cosines with the step size δ. The choice of the step size δ is crucial in the working of the method. With any given step size δ there is limitation to how close we can reach to the target. In order to get desired proximity towards any given target and starting from any arbitrary point coarse-to-fine search is necessary. With change of step size the previously learned direction becomes useless. As can be seen from Figure 5.1. After reaching certain point with certain step size the step size is reduced. The direction required with new step size is totally different from the direction that was required with the previous step size. The method described here tries to improve the direction by trying different possible directions and does not try to model the possible distribution of the rewards and thereby this method does not seek the information form the neighborhood. The advantage of this method is speed of convergence as is discussed in chapter 6.

5.1.2 Robust Learning

The EMMP method finds the near optimal direction towards the target under the constraint of the given step size. With each reduction in the step size δ, the search becomes finer and scope of the search also reduces. The reduction in δ is solely controlled by the rewards, the details of which are discussed in the following subsection. In order to take into account the possible errors in the rewards, and make the algorithm converge in spite of erroneous rewards, the following rule is applied to change

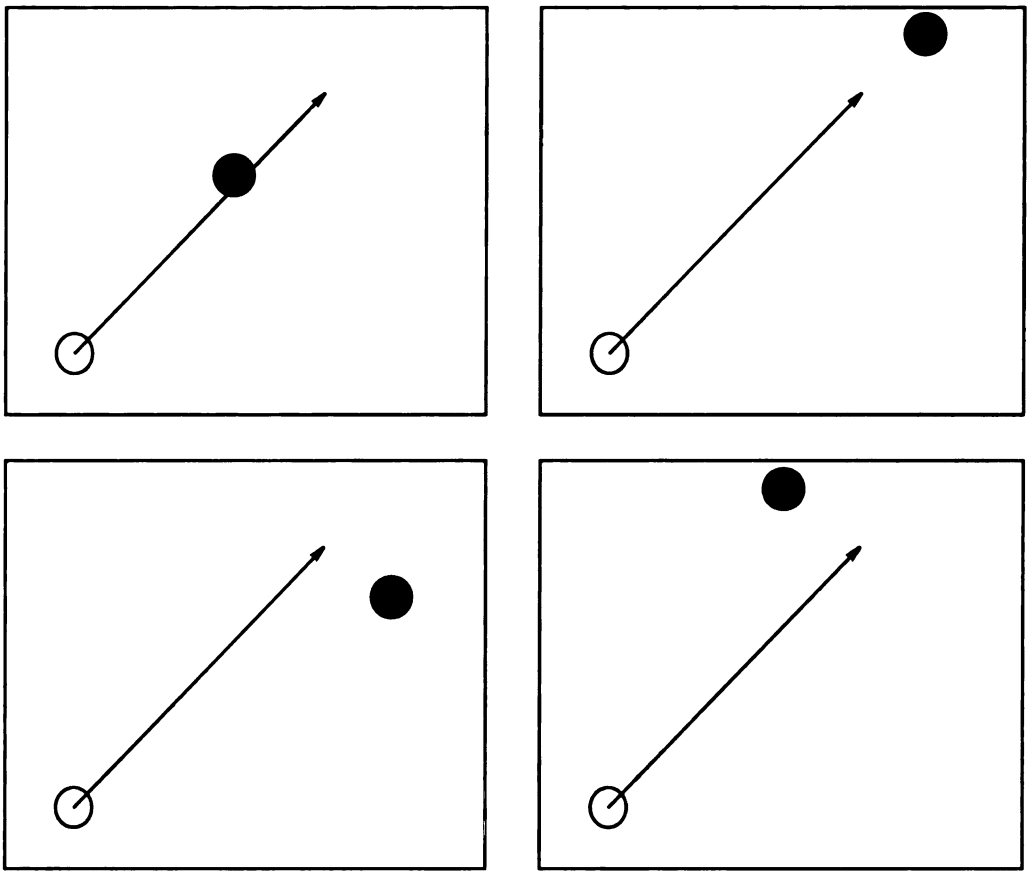


Figure 5.1: The solid circle represents the target and the hollow circle is the starting point. The arrow denotes the direction of the previous action and its end denotes the new start position. The different variations of the positions of the new starting point and the target are shown. Although in each case the distance from the target is same, the direction towards the target is entirely different from the starting direction.

the step size.

Reversible step size change algorithm:

1. Initialize δ.

2. Initialize counter for bad rewards.

3. Use EMMP for updating the $\Psi(i)$ also keep updating the bad reward counter.

4. If the number of bad rewards exceed the threshold then increase δ and restart

the EMMP.

The above algorithm makes sure that the EMMP is not stagnated at certain point in the search space due to the acquisition of a erroneous reward leading to unwanted reduction in δ. This makes the algorithm robust.

5.1.3 Consideration about Rewards

As is discussed before, the nature of rewards also play substantial role in overall performance of the system, the assumed behavior of the rewards in this mechanism is now discussed. In general, the rewards can be relative or absolute. In former case the 'Good' reward means that the performance of the system is improved compared to the previous attempt, while in latter case the 'Good' reward means that the performance of the system is within certain predetermined bounds irrespective of the previous performance. In this particular case, both the types of rewards are expected by the system in certain specific way. The rewards are of three types, 'Good', 'No reward' and 'Bad'. The 'Good' reward is considered as absolute, and the latter two rewards are considered as relative. The 'Good' reward means that the search has reached sufficiently close to the target and the step size can be reduced. The 'Bad' reward means that the performance is getting worse compared to previous performance. 'No reward' can mean the performance is unchanged or that it has improved compared to the previous attempt, but the improvement is still not within the predetermined bounds to go to the next hierarchical level. This structure of rewards is constructed based on the real time evaluation, where the rewards will be obtained from the human

teacher. It is observed that the 'Good' reward definitely means sufficient proximity to the target. When the system is exploring in areas away from the target, it is difficult for a human teacher to determine whether the system is going in right direction. Hence, in such cases no reward is offered. However, when the performance deteriorates giving 'Bad' reward is possible.

Algorithm

1. Initialize the start point.

2. Initialize the direction cosines in all the dimensions. The values are normalized with the norm as unity.

3. Initialize number of iteration, $n = 1$.

4. Start iterations.

5. Get the reward from the iteration.

6. Use the reinforcement learning equation to update the direction cosines.

7. Take the action.

8. Increment the iteration number $n = n + 1$.

9. Go to step five. (No Stop.)

The architecture is shown in Figure 5.2

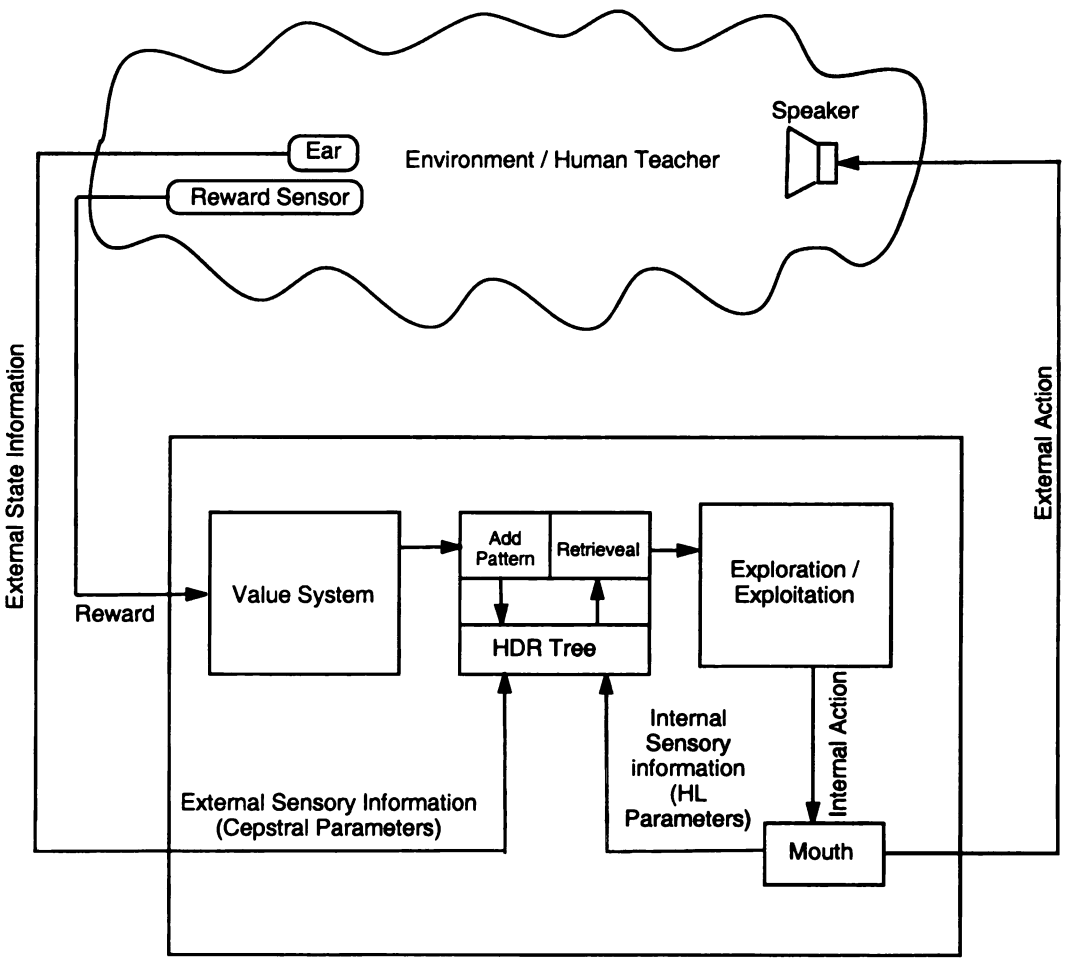


Figure 5.2: The system architecture with DUDC.

5.2 Learning with High-dimensional Conjugate Gradient Search (HCGS)

In the traditional Q-learning as is described in chapter 2, there are number of discrete actions and a Q-value is associated with each action. Depending on the policy an action is chosen and it is implemented. This method proves inadequate with the continuous and real valued state and action spaces. However, with the proven convergence properties of Q-learning and its robustness to occasional incorrect rewards [29], it is suitable for this problem. In order to fit current problem in Q-learning architecture, 'Vector Quantization' (VQ) of the action space is required. The VQ used in this thesis is called as amnesic VQ. The method is discussed in the following section.

5.2.1 Amnesic VQ in Action Space

The quantized actions in the action space are called as micro-clusters. Using a suitable threshold the actions are quantized. Each micro-cluster represents a single action and possesses a Q-value. Each context is limited to have a specific number of micro-clusters of actions. After trying values from '10' to '100' for this number, it was finally fixed to '30'. The mean value that represents a micro-cluster is updated using the method of 'Amnesic Average'. In traditional update with accumulation of large number of actions the contribution of the new action is reduced to infinitesimal. In practice this is not acceptable as all the system properties need to change with time. Amnesic update eliminates this drawback by using a different amnesic factor based

on number of updates on the action [34]. The amnesic update is based on amnesic parameter $l(n)$, which is defined as,

$$l(n) = \begin{cases} 0 & \text{if } n \leq n_1 \\ 2(n - n_1)/(n_2 - n_1) & \text{if } n_1 < n \leq n_2 \\ 2 + (n - n_2)/m & \text{if } n > n_2 \end{cases} \tag{5.6}$$

The typical values of 'n_1' and 'n_2' are '20' and '2000'. Thus when the number of updates is less than '20', the update is same as traditional average, however after that the amnesic parameter '$l(n)$' is designed in such a way that more priority is given to the incoming action vector. After n_2 updates the weight of the incoming sample is changed again and it increases with a rate $1/m$. The value of 'm' is chosen to be around '1/1000', so that after '2000' updates the weight of the incoming sample is almost constant at about '0.1%'. The update equation using this amnesic factor is stated as,

$$R^{n+1} = \frac{n - 1 - l(n)}{n} R^n + \frac{1 + l(n)}{n} V \tag{5.7}$$

Where 'R^n' and 'R^{n+1}' denote the 'n^{th}' and '$(n+1)^{th}$' updated vector respectively and 'V' denotes the new action taken in '$(n+1)^{th}$' iteration.

With the learning process the micro-clusters are developed and their Q values are updated. Action micro-clusters and actions are used interchangeably for representing the same list. These micro-clusters are used to generate the interpolation function for Q value in composite context and action space, as described in the next section.

5.2.2 Interpolation Using Q-values

In order to use the Q values of the various actions effectively to find optimal direction, it is essential to have an interpolation function representing the distribution of the Q-values in context-action space. The CG method can then effectively find the optimal action using this distribution. In order for CG method to work effectively it is important that the interpolation function is smooth and has a single well defined local minima. In order to get these properties in the function, a new interpolation method called "Density Sensitive Kernel Interpolation" (DSKI) is used. The details of the method are listed below,

$$G_n(x; x_1, x_2, ..., x_n) = \sum_{i=1}^{n} w_i(x) y(i) \tag{5.8}$$

$x_i \in R^d$, $y_i \in R^k$. Value of 'd' is '10' in the current context. $x_1, x_2, ..., x_n$ are n nearest neighbors of $x \in R^d$ from S. S is a set of micro-clusters in R^d representing the approximation of reciprocal of density in R^d. S can also be called as a set of finite number of neurons for each state. The definition of the weights w_i is based on the squared local sparseness σ^2, which is defined as,

$$\sigma^2 = \frac{k}{n} \sum_{i=1}^{n} \|x_i - x\|^2 \tag{5.9}$$

where, 'k' is called as 'kernel variance factor'. The more the value of 'k', the more is kernel variance and more flat the interpolated function is. The effect of 'k' on the distribution is shown in the figure. The expression for weights is,

$$w_i = Cexp(-\frac{\|x - x_i\|^2}{2\sigma^2}), \quad i = 1, 2, ..., n \tag{5.10}$$

The constant 'C' is computed such that $\sum_{i=0}^{n} w_i = 1$. $y_i = f(x_i)$ is the function to be approximated. In this case, y_i denotes the Q-value of the action. The action list contains about 30 samples. Hence the top-15 neighbors out of the 30 are chosen to interpolate the function. The sample plots generated from the DSKI method from the sample simulation one dimensional data are shown in Figure 5.3

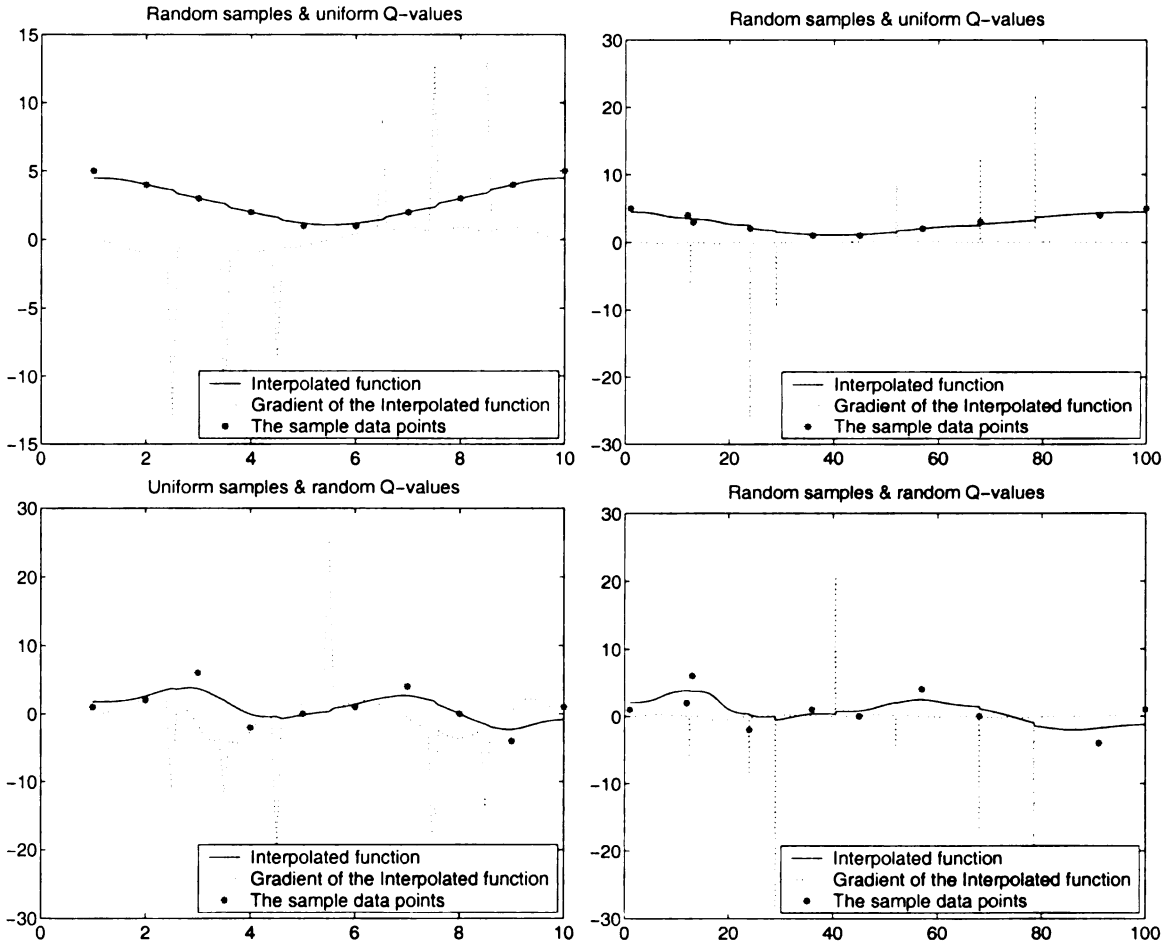


Figure 5.3: The plots of the Interpolated functions and its gradients for various combinations of the distribution of sample points and their Q-values.

5.2.3 Conjugate Gradient Method

The Conjugate Gradient method tackles the problem of searching in high dimensions in an elegant and sophisticated way. There are many iterative methods which deal with finding the minima of a function in multi-dimensions, for example Steepest Descent or Conjugate Directions [25]. Conjugate Gradient method is the best of all in terms of efficiency and memory usage. Originally the method is developed for finding minima of a symmetric quadratic function in the form

$$f(\mathbf{x}) = \frac{\mathbf{1}}{\mathbf{2}}\mathbf{x^T A x} - \mathbf{b x} + \mathbf{c} \tag{5.11}$$

where 'A' is a positive definite and symmetric matrix. This method guarantees convergence in 'n' steps, where 'n' is the number of dimensions of $\mathbf{x}$. The entire theory behind the development of the method is beyond the scope of this thesis and can be found in [25]. The same method can be extended to find the minima of a non-linear function. In this case the function along with its first and second order derivatives should be known either analytically or they should be easily computable numerically with less numerical errors. Nonlinear CG does not assure convergence in 'n' steps, it can also converge to a local minimum or it might just diverge depending on the given function characteristics. The detailed analysis of the convergence of the method in various cases can be found in [8]. However, most of the work in this area only concentrates towards finding global minimum in a bunch of local minima [9] [25], while the case of a function having a maxima in a neighborhood is not studied in detail. There are two variations of the method that are used in non-linear case, the 'Fletcher-Reeves' method and 'Polak-Ribiere' method. The 'Polak-Ribiere' method is better

among the two in most cases [25]. Both the algorithms share most of the steps, and vary only in the way an intermediate variable 'β' is designed. The algorithm is given below,

5.2.4 The Algorithm

1. $d_{(0)} = r_{(0)} = -f'(x_{(0)}$

2. Find $\alpha_{(i)}$ that minimizes $f(x_{(i)} + \alpha_{(i)} d_{(0)})$,

3. $x_{(i+1)} = x_{(i)} + \alpha_{(i)} d_{(0)}$,

4. $r_{i+1} = -f'(x_{(i+1)})$,

5. Fletcher-Reeves: $\beta_{(i+1)} = \frac{r_{(i+1)}^T r_{(i+1)}}{r_{(i)}^T r_{(i)}}$, $Polak - Ribiere : \frac{r_{(i+1)}^T (r_{(i+1)} - r_{(i)})}{r_{(i)}^T r_{(i)}}$

6. $d_{(i+1)} = r_{(i+1)} + \beta_{(i+1)} d_{(i)}$

In this nonlinear version of CG there is no direct way to estimate the step size 'α' as it is in linear version. Hence line minimization techniques are employed to find it. This step size denotes the optimal distance to be travelled in the direction 'd' as found by the CG algorithm. Iterative methods like 'Newton-Raphson' or 'Secant' are generally used for this purpose. Both the methods are based on the Taylor series expansion of the a function. The analysis of secant method is now presented.

$$f(x + \alpha d) \approx f(x) + \alpha [\frac{d}{d\alpha} f(x + \alpha d)]_{\alpha=0} + \frac{\alpha^2}{2} [\frac{d^2}{d\alpha^2} f(x + \alpha d)]_{\alpha=0} \quad (5.12)$$

$$\approx f(x) + \alpha [f'(x)]^T d + \frac{\alpha^2}{2} d^T f''(x) d \quad (5.13)$$

$$\frac{d}{d\alpha} f(x + \alpha d) \approx [f'(x)]^T d + \alpha d^T f''(x) d \quad (5.14)$$

The secant method tries to express the second order derivative of the function with the first order derivative at different positions. The expression is given below,

$$\frac{d^2}{d\alpha^2}f(x+\alpha d) \approx \frac{[\frac{d}{d\alpha}f(x+\alpha d)]_{\alpha=\sigma} - [\frac{d}{d\alpha}f(x+\alpha d)]_{\alpha=0}}{\sigma} \quad (5.15)$$

$$\approx \frac{[\frac{d}{d\alpha}f'(x+\sigma d)] - [\frac{d}{d\alpha}f'(x)]}{\sigma} \quad (5.16)$$

Using the equations 5.13 and 5.16 we can write the derivative of '$f(x+\alpha d)$' as,

$$\frac{d}{d\alpha}f(x+\alpha d) \approx [f'(x)]^T d + \frac{\alpha}{\sigma}\Big[[f'(x+\sigma d)]^T d - [f'(x)]^T d\Big] \quad (5.17)$$

Now in order to minimize the function the first order derivative '$f(x+\alpha d)$' is equated to zero. This gives us an expression for 'α'.

$$\alpha = -\sigma \frac{[f'(x)]^T d}{[f'(x+\sigma d)]^T d - [f'(x)]^T d} \quad (5.18)$$

Secant method suffers from subtle drawback. When it tries to find the expression for 'α', it can reach maxima or minima depending on which is closer. There is no way the method can guarantee that the optimum is a maxima or a minima. The immediate solution to this problem can be thought of as looking at the second order derivative to decide about the maxima or minima. This method has two main practical problems: (i) finding second order derivative is highly computation intensive and it also becomes considerably error prone when dealing with small neighborhood and slow changing functions; and (ii) most of the times the second order derivatives return arrays filled with zeros. In order to make remove this drawback and make the system always seek the minima a the algorithm is slightly modified. When a suitable step size 'α' is computed using the secant method, (the Newton-Raphson method requires computation of second order derivative, hence secant method is preferred over it)

the value of function in the direction opposite to that suggested by secant method is checked with the value in the suggested direction. If the value in the opposite direction is less than the value in opposite direction, it is assumed that the secant method is trying to get maxima instead of a minima and the opposite direction is chosen instead. It is also observed that sometimes due to the rapid changes in the value of gradients the step size is incremented drastically and the algorithm is diverging from the target. In order to restrict the performance of the secant method the value of the function at the consecutive values of x produced with the secant method are compared and if it is observed that the method is diverging it is immediately terminated to the previously found value of 'x' and the CG method then takes over. The effects of these changes are shown in the figures 5.4, 5.5, 5.6, 5.7.

However both the methods bear a considerable drawback when the function also has a maxima in the neighborhood. Both the methods rely on finding the point in the neighborhood of the given point where the first order derivative of the given function reduces to zero. This inherently finds either maxima or minima.

The function generated using DSKI is feeded to the CG search to find the minima. The Q-values are sign reversed before generating the interpolated function so that the minima of the function corresponds to maximum Q-value and hence optimal action. The plots of the interpolated function and performance of the CG method are shown in Figure 5.8 The performance of the CG in two dimensional case is shown in Figure 5.9. The starting point along with the trajectory followed by the CG to reach the optimal point is also shown.

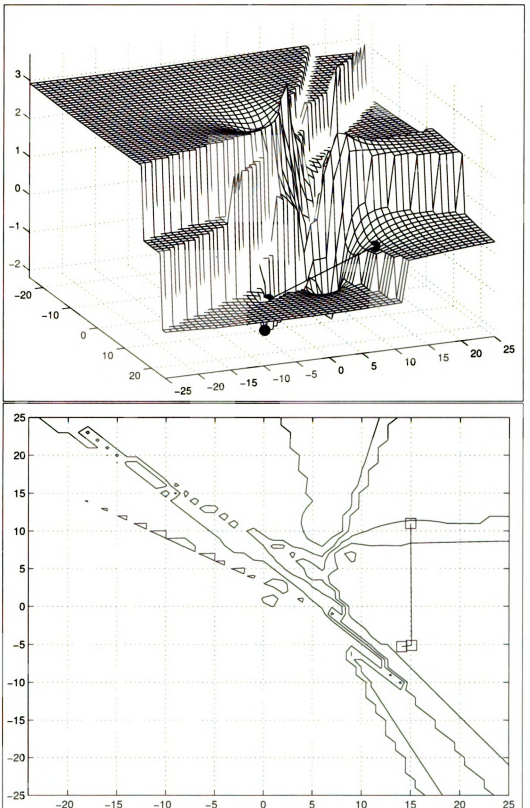


Figure 5.4: Modified Conjugate Gradient method applied to simulation data similar to practical data. The starting point is chosen arbitrarily.

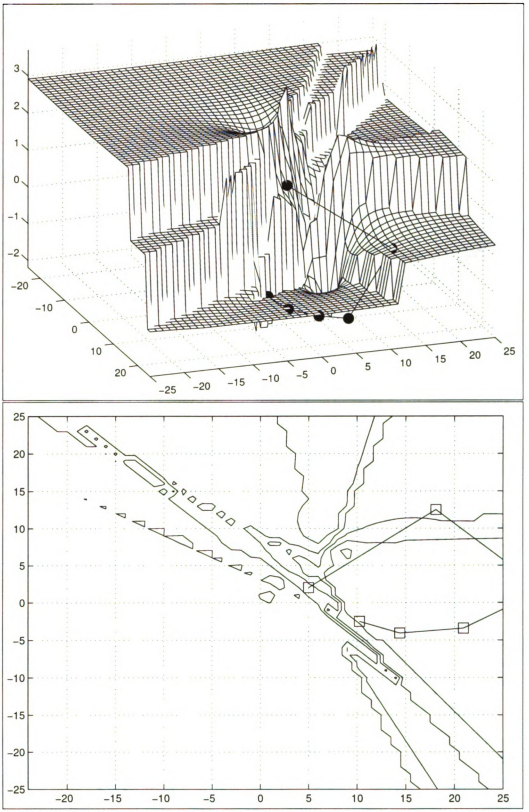


Figure 5.5: Modified Conjugate Gradient method applied to simulation data similar to practical data. In spite of starting point being near maxima, new search method still converges to minima.

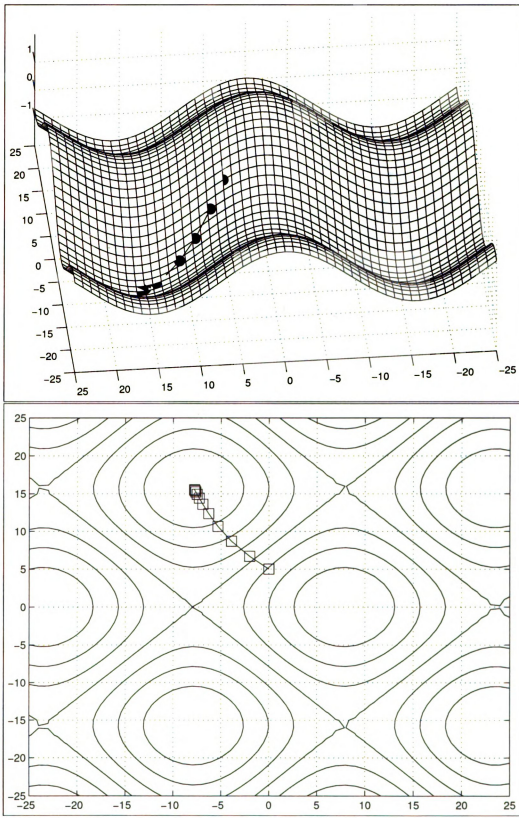


Figure 5.6: Modified Conjugate Gradient method applied to general sin-cosine function. The starting point is chosen arbitrarily.

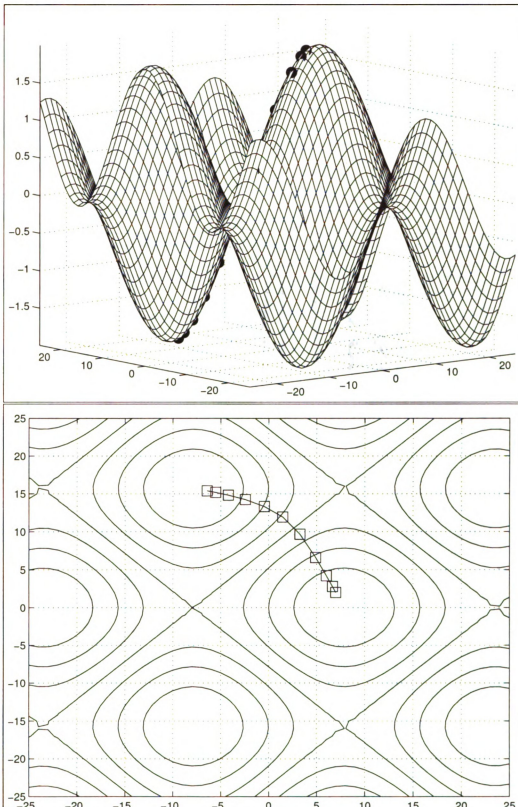
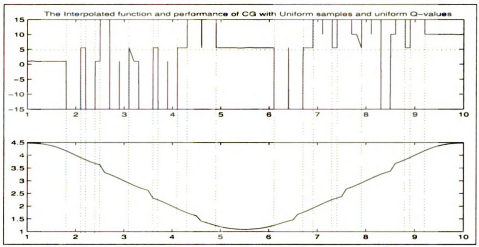
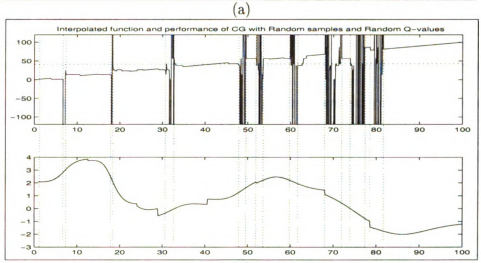


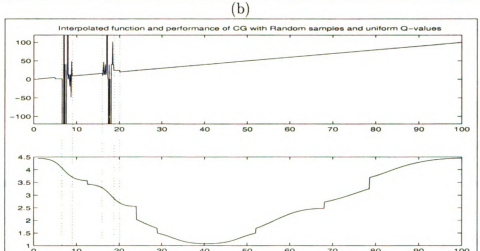
Figure 5.7: Modified Conjugate Gradient method applied to general sin-cosine function. In spite of starting point being near maxima, new search method still converges to minima.



(a)



(b)



(c)

Figure 5.8: The plots of the Interpolated functions and performance of CG on it. In each figure first plot shows convergence of CG v/s starting point of the search and second plot shows the interpolated function. The dotted lines enclose the region where the CG search is deviating from the target.

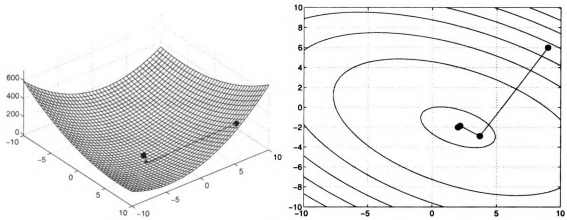


Figure 5.9: The Search trajectory of CG in two dimensions. The real trajectory in 3D and its contour path are shown.

Due to inherent limitations of linear line minimization methods they cannot be used to give optimal performance in all the situations. Hence to preserve the generality of the algorithm the line minimization part of the CG algorithm is removed. The initial guess for the step size is calculated based on the density of the samples at the starting point.

5.2.5 Q-Learning

The mathematical description of the Q-learning is explained in chapter 2. A queue of the 'n' recent states is maintained called a 'Prototype Updating Queue' (PUQ) and with each incoming reward the Q-values of the states in PUQ are updated. The Q-value of the most recent state is updated using the direct Q-learning rule as given in equation 2.1. The Q-values of the succeeding states are updated using,

$$Q(s,a) \;=\; (1-\alpha)Q(s,a) + \alpha(\gamma Q). \tag{5.19}$$

where 'Q' represents the updated Q value of the preceding state in PUQ.

5.2.6 Boltzmann Exploration

The details of Boltzmann exploration are given in chapter 2 and equation 2.2. The Boltzmann exploration also requires the actions in discrete nature, as is the requirement of the Q-learning, in order to make a choice. The generation of micro-clusters solved the problem with the Q-learning, but with careful investigation it can be understood that we still cannot use the Boltzmann exploration directly on these actions. The subtle difference in this list of action micro-clusters and traditional set is the traditional set encompasses all the possible actions, while current set does not. The actions outside the list of action micro-clusters can also be taken and they can even be better than the ones in the list. Hence a separate random exploration is used along with the choice of exploitation using CG search. The random exploration uses the upper and lower bounds on the action along with the inter-dependencies in the action values to generate a random action, while CG search uses all the action micro-clusters to generate new action. These two choices are feeded to Boltzmann exploration with probabilities which are based on the status of the action list. If the action list contains sufficient actions with good Q-values, then more probability is assigned to the exploitation and vice versa. However, ultimately the decision is taken by the Boltzmann Exploration whether to use the exploitation or random exploration.

5.2.7 The Detailed Behavior of the Action Micro-Clusters

During the initial phase when the action micro-clusters are being created for the first time a default distance threshold is used. In the later stages the micro-clusters

are updated with new samples. With each update the micro-clusters move towards the new sample. The amnesic averaging mechanism makes sure that even after large number of updates the new samples still makes noticeable impact on the micro-cluster.

It is observed that with the exploitation superseding the exploration some of the micro-clusters being closer to the target are updated more often than others. Also due to this most of the micro-clusters are under utilized. As an implication of this phenomenon, the required uniform convergence is not observed. After the search reaches a certain proximity, the algorithm stagnates. In order to address this issue the re-organization of the micro-clusters is performed after the context is repeated certain number of times. The mechanism of the re-organization is now discussed.

The weighted mean of the micro-cluster distribution is evaluated and then all the micro-clusters are pulled towards it based on the number of times each one is visited. This is followed by reduction in the search space. The reduction in search space ensures that the random exploration is also restricted to explore the important area. After the repositioning of the micro-clusters the count of the number of updates on each micro-cluster is initialized along with their Q-values. This starts the new search in the reduced space. This mechanism basically represents the hierarchical coarse to fine search.

In another approach the top 'n' clusters with highest Q-values are selected and remaining clusters are removed. They are then filled with the new incoming data. The search boundaries are also reduced along with this in order to restrict the random exploration. However, this method suffered from the drawback that many times the main target remains outside the reduced search space after few iterations of the cluster

re-organization. This causes the search to stop after reaching certain accuracy level.

Considering more detailed picture of learning. The IHDR tree is initialized with maximum of five clusters in each node and maximum number of samples that can be held in a node is restricted to 200. The sample list is maintained as a common list for the entire node and each sample is marked with the membership number which corresponds to the cluster in the node. With each context retrieval the nearest sample is found and if the distance to the nearest sample is above a threshold (a threshold of 7.0 is used) then it is assumed that there is a new context and it is added to the tree and a default action which is "keeping silent" is taken. If the distance to nearest neighbor is less than the threshold then it is assumed that the context is similar to the one that is observed in the past and the action list corresponding to that context is extracted and then using the policy as described above an action is chosen and is taken.

The block diagram of this method is shown in Figure 5.10.

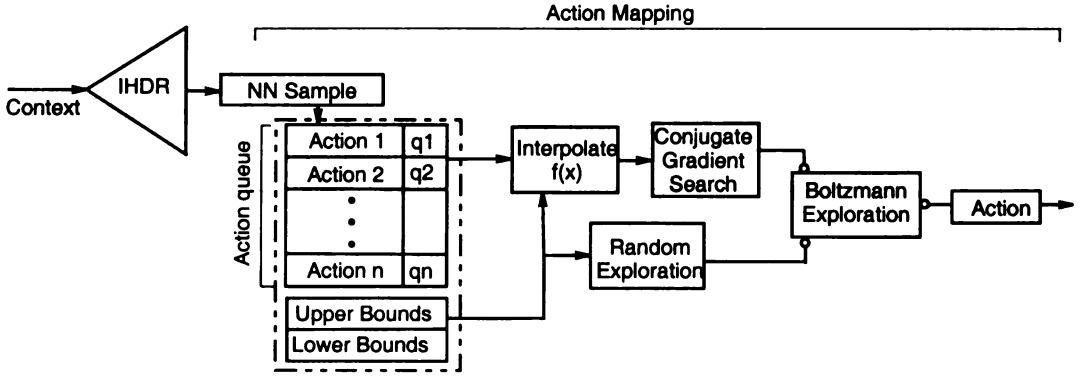


Figure 5.10: The system architecture with HCGS.

5.3 Developmental Learning

Finally all these components are grouped into a complete system in the Developmental Learning paradigm [36]. The software platform of SAIL project is used for the development of the system. The sound acquisition part using cepstral components is already developed and is directly used. The other components for the mouth and the learning system for the speech are independently coded and integrated into the system. The multi-threaded architecture of the system made it easy to add components having separate threads to execute. The IHDR tree is at the heart of the system providing the associative memory similar to human memory. The sensory information, internal as well as external is combined, normalized and added to the tree continuously as the system starts learning. The block diagram of the system architecture and the flow chart of all the parallel running processes in the system are shown in figures 5.11 and 5.2 respectively.

This thesis is part of the SAIL project, where an entire humanoid robotic system is being developed capable of having vision, and locomotion. The basic block diagram of the system as part of SAIL is shown in Figure 5.12. In the entire block diagram of SAIL the block of sensory mapping consists of all the sensory organs including the vision and locomotion. The details of the system can be found in [35].

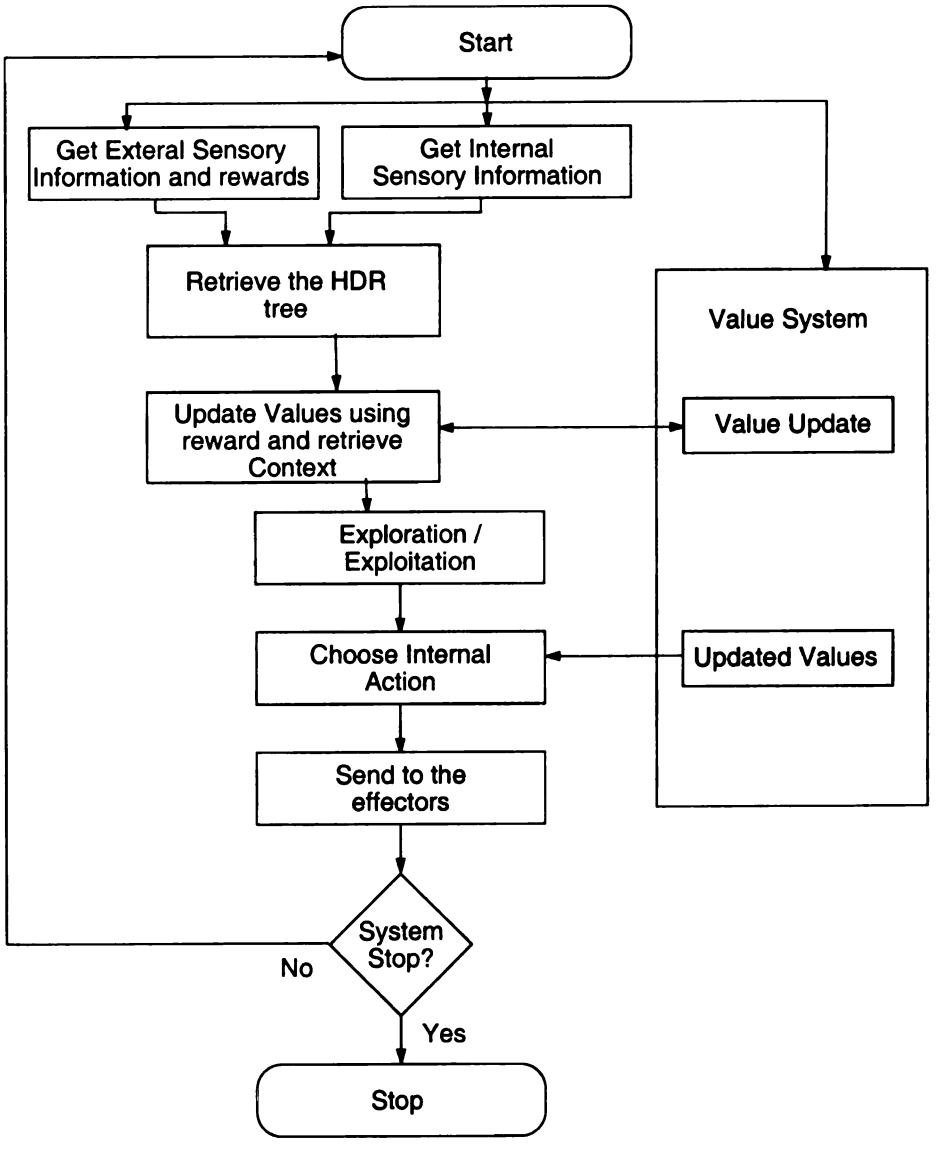


Figure 5.11: The Flow Chart of the Program.

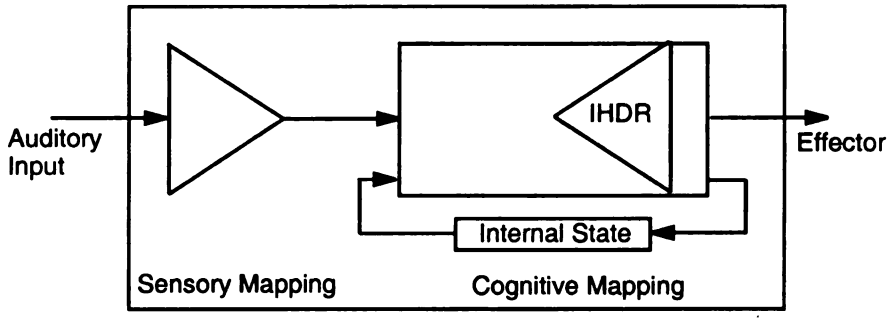


Figure 5.12: The block diagram of the system as part of the entire SAIL architecture.

Chapter 6

Results and Discussion

The testing of the system is performed in multiple stages. The testing of the two different approaches is performed independently according to their requirements and objectives and the results are discussed. The distribution of the four vowels '/a/', '/e/', '/i/', '/u/' to be learned in their formant space is shown in Figure 6.1. The data related to the fifth vowel '/ae/' was not present and hence is not shown in the figure. The original data is obtained from [22]. As can be seen there lies a considerable overlap among them which is a challenging issue in the learning.

6.1 Testing of DUDC

The objective of the system is to learn in a context based environment. The method of DUDC is designed to learn fast in high dimensional space by utilizing the absolute rewards. The distinction between absolute rewards and relative rewards is discussed in chapter 5. Although the system can handle erroneous rewards, it demands more

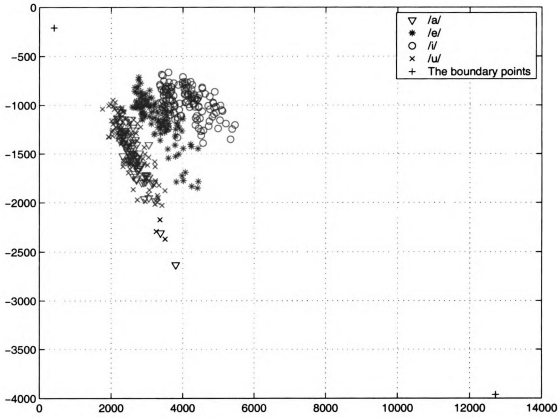


Figure 6.1: This figure shows the distribution of the four vowels in the reduced dimensions (using PCA). The point in the left top corner represents one end of the search space and the point in the right bottom corner represents the other end. The distribution as can be seen is sparse and also overlapping.

from rewards than the HCGS method. As the learning occurring at two different contexts is independent of each other, it is essential that the validity of the learning in individual context is tested first. With the mechanism of obtaining the context in place the rest of the functioning is extension of single context learning. Before tackling the actual system, the algorithm is tested using a simulation program. A data with variable dimensions from two to ten is used for the testing. The different locations of the target relative to the starting point are chosen and the results are tabulated in the table 6.1. The start point is chosen arbitrarily as its position is not

Dim.	Error Rate	Target Point Location	Avg. no. of steps
2	0%	Center	29.91
		Boundary	3.44
		Random	54.48
	25%	Center	35.24
		Boundary	3.84
		Random	44.32
5	0%	Center	257.37
		Boundary	55.66
		Random	319.18
	25%	Center	303.49
		Boundary	50.12
		Random	856.04
10	0%	Center	1248.22
		Boundary	445.27
		Random	1089.82
	25%	Center	1844.08
		Boundary	717.64
		Random	2279.45

Table 6.1: The table displaying the average learning rate with variable dimensions and variable error rates in rewards.

important, but the target point location is varied in three stages as 'near the center of the search space', 'near the boundary of the search space' and 'arbitrarily anywhere in the search space'. The testing is carried out for two, five and ten dimensions. Each experiment is conducted for 100 iterations of search to get near perfect statistics of the data. The table 6.1 shows the average values of iterations for each dimensional data and also in each case the rewards without error and with 25% error are used. The measure of convergence is defined as the reduction in the target distance by the factor of 20 with respect to the starting distance. This corresponds to 5% of the initial distance. The percentage rise of the average steps for convergence is plotted in Figure 6.2

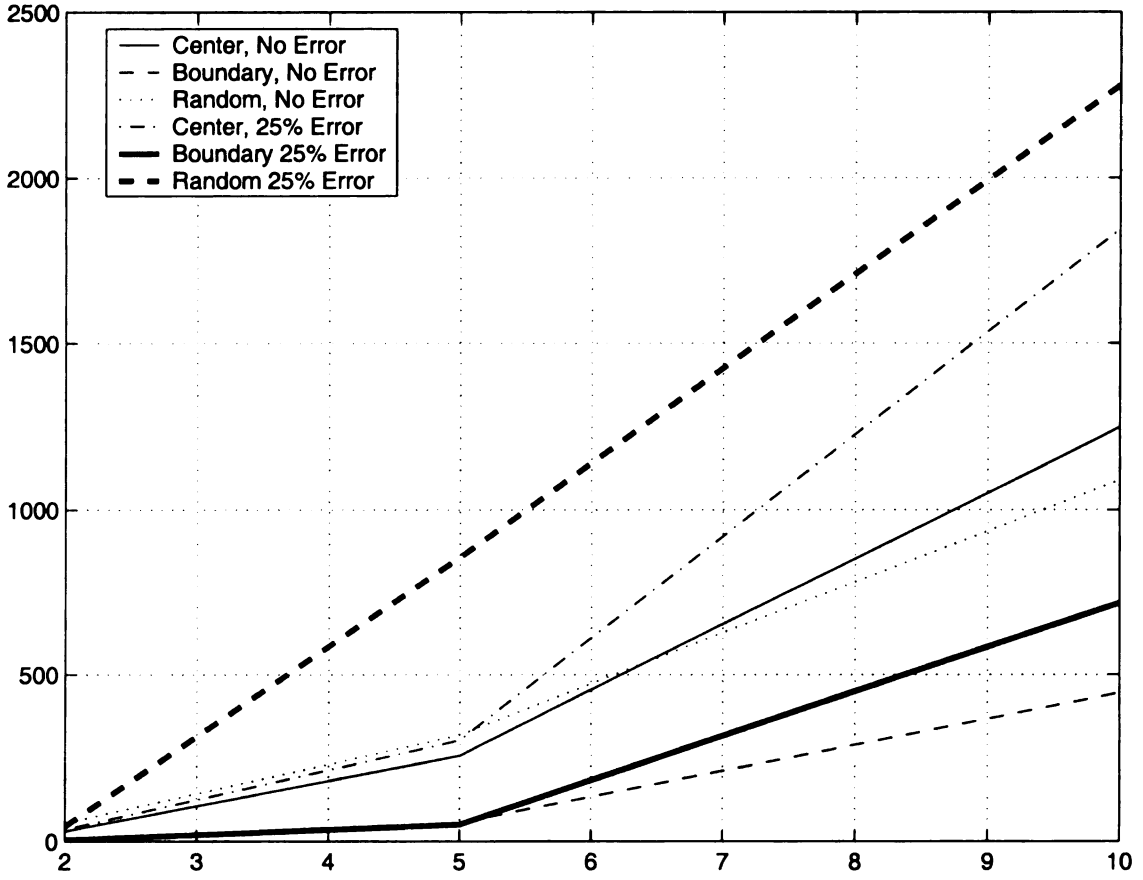


Figure 6.2: The variation in convergence rate with variable dimensions and error rate in rewards.

After establishing the convergence of the algorithm, it is incorporated in the main system having all the three organs working parallel, the brain, the mouth and the ear. The test results with different methods are now discussed.

6.1.1 Testing with Synthetic Teacher

The concept of a Synthetic Teacher (ST) is introduced. This ST is implemented as a standalone program running parallel with the system and doing the job of a human teacher. The testing with the ST is carried out in two stages. In the first stage, the ear and mouth are disabled and the learning context is not varied. This approach is

similar to the testing of the simulation data. The HL parameters, generated in the form of action, are directly fed to the ST to receive a reward. Due to the absence of delay due to utterance, the learning is fast. The plots of convergence are shown in the Figure 6.3

In the second stage, the ear and mouth are enabled. The system produces sounds and also listens to the environment. The ST produces sounds as pre-recorded wave files every alternate second to provide the context. However, the rewards are generated from the distance to the target using the HL parameters. Because this learning requires the one second wait in each state, the learning is slow. Also as the context is obtained from the external and internal sensory parameters, there are multiple contexts the system has to deal with. The plots of convergence are shown in Figure 6.4.

6.1.2 Testing with Human Teacher

Although the final goal of the system is to have it interact with humans and learn from the rewards obtained, some practical problems with this testing are pre-identified:

1. The synchronization of the states of the machine and the utterance of the teacher is difficult to obtain, which is needed for optimal performance.

2. The inherent variation in the human voice and change in the volume can make the similar utterances sensed as different contexts. This is theoretically acceptable but it will delay the learning process.

3. The teacher is likely to get confused in giving rewards when robot is speaking

Table 6.2: Difference in the two methods of testing

Testing with ST	Testing with Human Teacher
1. Accurate. 2. Have a single and well defined target.	1. Prone to errors. 2. The target is a cluster of different utterances which are recognized by the human teacher as similar.

simultaneously.

4. The similarity in the utterances is sometimes difficult to identify and this makes the learning process slow.

These problems are only going to make the learning slow while the capability of the system to converge remains unhindered. The results are plotted in Figure 6.6. Two plots are shown for each convergence one of them shows the distance from the final utterance and the second one shows the actual positions of the parameters in reduced dimensions. The dimension reduction is performed using PCA. As can be seen the convergence is faster, but the final utterance is not of very good quality as is obtained with the ST. The reason being all the rewards given by ST are perfect (excluding the case of erroneous rewards). The testing has been conducted for more than two hours. The temporal description of the mechanism is shown in the Figure 6.5.

6.2 Testing with HCGS

The method of learning with HCGS is designed to tackle the problem of developing the system to handle very large amount of interactions. The demand from the rewards is relaxed in this method and it can take absolute as well as relative rewards. The

objective is to have a convergence rate is not as fast as in the previous method, however the system should not deviate from the target in the long run.

The testing framework is similar to the previous method, except that due to slow rate of convergence the last two real time testing steps are not carried out. The simulation results on the generalized sine-cosine function and also on a sample data obtained from the real testing are discussed in chapter 5. The performance of the system in real time simulation with ST is now discussed.

The number of micro-clusters in the action list is the most crucial parameter in the overall performance. With the curse of dimensionality constructing an interpolation function in ten dimensions and then using the CG search on it to get the minima depends a lot on the number of samples available. The number of micro-clusters is varied from ten to 100 and in each case the convergence plots are generated. The convergence plots as shown in Figure 6.7 need some further explanation. The case with number of micro-clusters as 30 the convergence is better compared with the other case when the number of micro-clusters is 100. However in either cases the search does stagnates after certain level of convergence. The reason behind this can be explained as below. 'Initially as the micro-clusters are formed by random exploration some of the clusters are close to the target. However, along with learning the micro-clusters start moving with amnesic average, only the ones that are close to the target are chosen according to the HCGS method and the same clusters keep getting updated. As a result the other clusters remain unchanged and the clusters which are already closer to target keep moving.

Research has been carried out in order to organize these clusters after certain

stage and various approaches tried are discussed in chapter 5. However the results are not satisfactory in this direction as of now. It is kept as a future work to find the optimal method for the reorganization of the action micro-clusters.

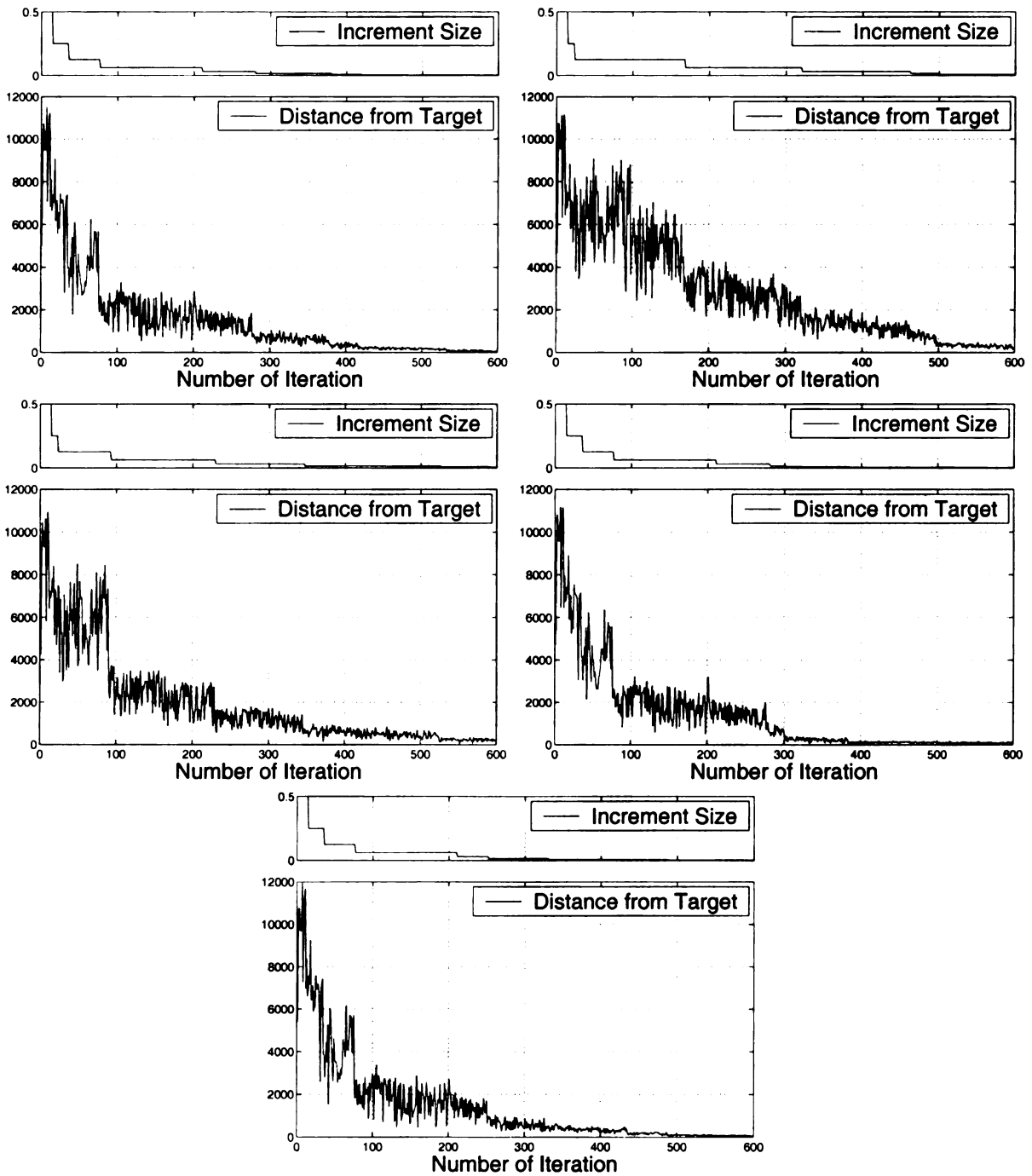


Figure 6.3: The Convergence plots generated during the learning of different vowels during 600 iterations using ST. The plots in the order of left to right and top to bottom are for vowels '/a/', '/e/', '/i/', '/ae/' and '/u/'. The context used in this testing was the direct internal sensory information in the form of HL parameter.

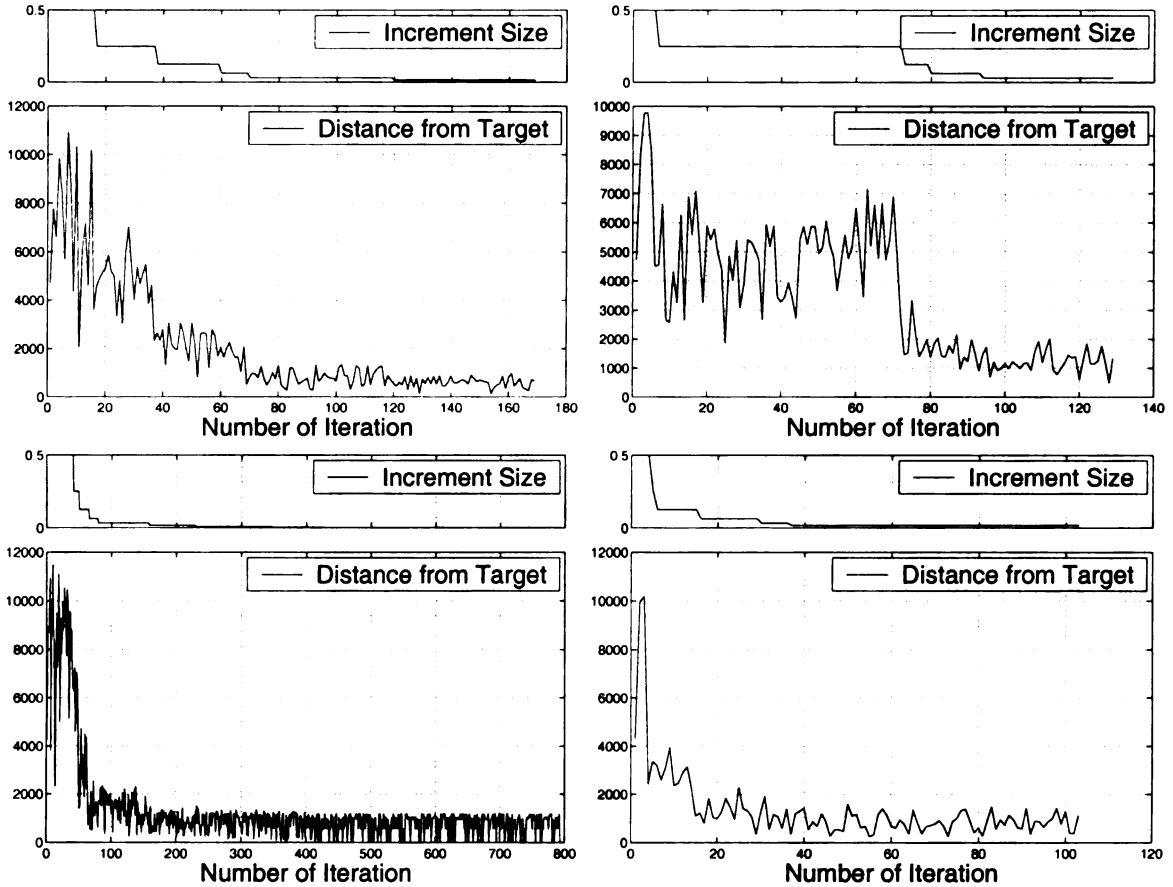


Figure 6.4: The figure shows the convergence plots obtained from the test carried with synthetic teacher. The plots show the results for contexts from one to four in the order left to right and top to bottom. The synthetic teacher played the wave files recorded in my voice to create context and gave rewards based on distance to the target.

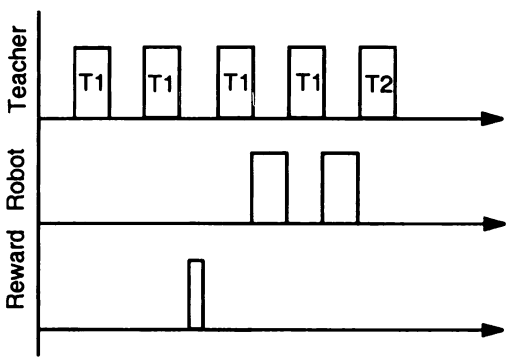


Figure 6.5: The modified timing diagram for learning with Human Teacher.

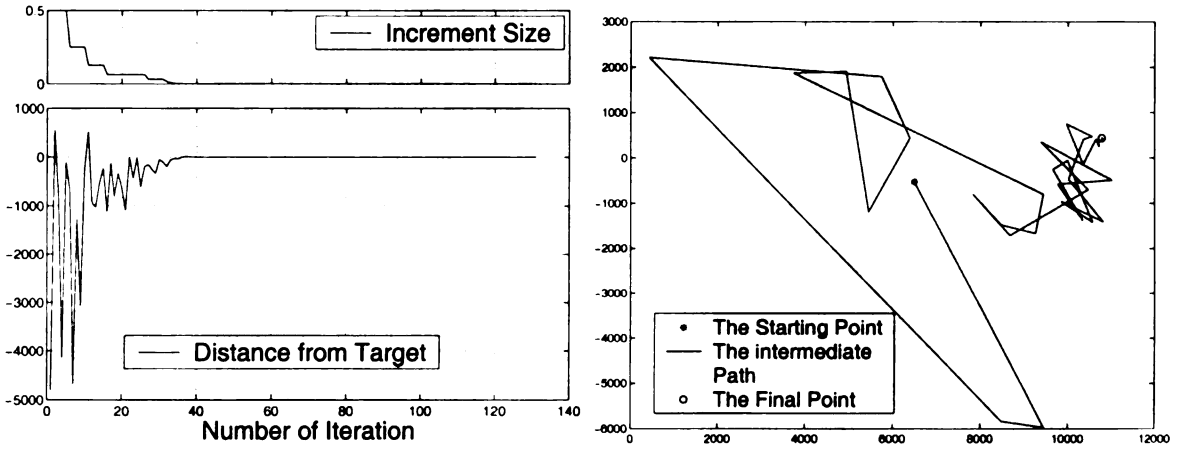


Figure 6.6: The Convergence plots from the real time test with human teacher. The left figure shows the convergence at a context based on the distance from the final point. Due to unknown target parameters the normalized distances are found from the final convergence point. Hence the distance of zero is the final convergence. The right figure shows the trajectory of the search is plotted in reduced dimensional space. The dimension reduction is done using PCA.

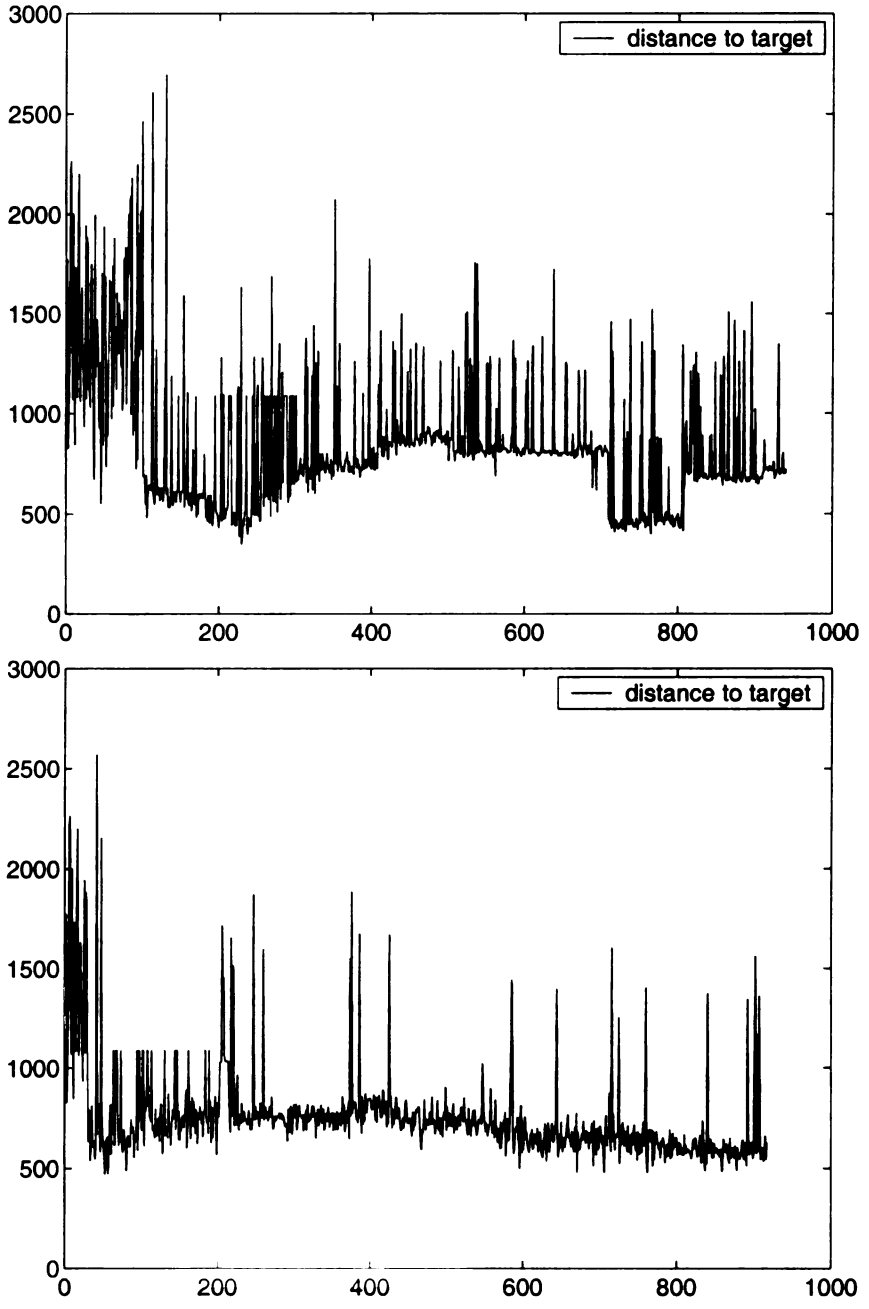


Figure 6.7: The figure on left shows the convergence with number of micro-clusters as 100 and figure on right shows the convergence with number of micro-clusters as 30.

Chapter 7

Contributions and Conclusions

7.1 Contributions

During the development of the thesis and working towards the objective of developing the system in AMD architecture, some new areas in the field of reinforcement were explored. The previously existing methods were found insufficient or in some cases incompatible with the current architecture. The pursuit of the problems resulted in some new contributions in the related areas. The main contributions are listed below.

(1) The first work on learning action in high dimensional action space (5 to 10 dimensions) using the new AMD mode.

(2) A new technique of DUDC is developed for reinforcement learning in continuous and high dimensional action space along with the development of the alternative for Boltzmann exploration as EMMP.

(3) The method of Conjugate Gradient is modified for use in high dimensions. The secant method, which is used for line minimization suffers from the drawback of

not being able to identify the neighboring optima as maxima or minima. The method is modified to tackle this issue successfully.

(4) The first work towards learning of speech production interactively using the reinforcement learning framework without supervised learning mode.

7.2 Conclusions

The methods developed in this thesis appear promising towards tackling the problem of autonomously developing a robotic system capable of learning to produce high dimensional (e.g. 10D), action interactively and autonomously. The techniques designed and implemented in this work seem capable of realizing the initial development of basic, early behaviors in a high dimensional space through the AMD mode.

In the later stages of the speech learning development the system shows goal-directed behaviors, which facilitates faster learning. It is marked by a representation of the goal from the early learning experience, using the goals to activate actions, changing direction explicitly (e.g., understanding the goal by biting down on one's tongue and giving it a few tries).

The current work is, however, new and very important in bootstrapping higher level goal directed learning in the later development stage.

Chapter 8

Future Scope

8.1 Future Scope

In this thesis the most generalized case in reinforcement learning is considered with the objective of making robotic system capable of learning vowels. After the literature survey conducted throughout the research it is found that even the proven sophisticated methods seem insufficient to handle the situation. Practical problems in learning are also identified and it gives a direction towards the future research that needs to be carried out. Two new methods, DUDC and HCGS are developed and tested. The former method worked well in the given circumstances, however the latter method also seems promising with some more research in the ways to organize the action micro-clusters.

The current thesis limits its scope at the production of simple vowels where the position of human mouth and vocal tract system is unchanged during the entire utterance. This also means that the HL parameters that are used for modelling the

vocal tract also do not change. There are numerous possibilities for enhancing the project. The ultimate goal is generation of a machine, which is capable of speaking like an adult human. Although the approaches discussed here are intended towards speech production, they are also useful in the general motor mapping in the developmental SAIL robot or in any situation where one has to deal with high dimensions and continuous real values search spaces.

This work marks the first step towards modelling high dimensional sensory inputs and effector outputs with reinforcement learning. All the new developments in the field of Artificial Intelligence cannot escape from these problems. Hence this work will provide some useful feedback in this direction.

The learning algorithm is motivated from the learning behavior observed in the human infants, however the current work does not claim to be the exact replica of a human baby. However, this work does provide an engineering solution to the type of problems in this area.

Bibliography

[1] Jonathan Allen, M. Sharon Hunnicut, and Dennis Klatt. *From Text to Speech: The MITalk System.* Cambridge University Press, Cambridge, 1987.

[2] B. S. Atal and Suzanne L. Hanauer. Speech analysis and synthesis by linear prediction of the speech wave. *Journal of the Acoustic Society of America*, 1971.

[3] Jonathan Baxter and Peter Bartlett. Direct gradient-based reinforcement learning, (parts i and ii). Technical report, The Australian National University, Canberra, Australia, 1999.

[4] D. Bursky. New algorithms, chips bestow human qualities on synthesized speech. *Electronic Design*, pages 113–129, May 1985.

[5] Watkins C.J.C.H. Learning from delayed rewards. Technical report, Cambridge University, 1989.

[6] Micheal Cole and Sheila Cole. *The development of children.* W. H. Freeman and Co., 1989.

[7] E.J.Yannakoudakis and P.J.Hutton. *Speech Synthess and Recognition Systems.* Ellis Horwood Limited, New York, 1987.

[8] R. Fletcher and C.M. Reeves. Function minimization by conjugate gradients. *Comput. Journal*, 7:149–154, 1964.

[9] Peter Géczy. Superlinear conjugate gradient method with adaptable step length and constant momentum term. *ICICE Trans. Fundamentals*, E83-A:2320–2328, November 2000.

[10] Claude-Alain Hauert. *Developmental Psychology: Cognitive, Perceptuo-Motor and Neuropsychological Perspectives.* North Holland, 1990.

[11] W. Hwang and J.Weng. Hierachical discriminant regression. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22(11):1277 1293, 2000.

[12] J.Weng. and W. Hwang. An incremental learning algorithm with automatically derived discriminating features. *Proc. Fourth Asian Conference on Computer Vision*, 22(11), 2000.

[13] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[14] Sam Maes, Karl Tuyls, and Bernard Manderick. Reinforcement learning in large state spaces - simulated robotic soccer as testbed. In *Proceedings of Robocup2002*, Fukuoka Japan, 2002.

[15] Andrew Moore and Christopher Atkeson. The party-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. *Advances in Neural Information Processing Systems*, 1994.

[16] Renato De Mori and Ching Y. Suen. *New Systems and Architectures for Automatic Speech Recognition and Synthesis.* Springer-Verlag Berlin Heidelberg, NY, 1984.

[17] Remi Munos. A convergent reinforcement learning algorithm in the continuous case: the finite-element reinforcement learning. *International Joint Conference on Artificial Intelligence*, 1997.

[18] L. Oesterreich, B. Holt, and S. Karas. *Ages and stages-newborn to 1 year, Iowa family child care handbook.* Extension Distribution Center, Ames, IA, 1995.

[19] Douglas O'Shaughnessy. *Speech Communication.* Addison-Wesley Publishing Company, New York, 1987.

[20] M. Scot Peck. *The Road Less Travelled.* Rider Books, London, 1978.

[21] Joseph S. Perkell and Dennis H. Klatt. *Invariance and Variability in Speech Processes.* Lawrence Erlbaum Associates, Publishers, Hillsade, New Jersey, 1986.

[22] Peterson and Barney. *Formant frequency databse of Vowels.* Bell Communications Research, NY, 1993.

[23] L. R. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals.* Prentice Hall Inc., New Jersey, NY, 1978.

[24] Stefan Schaal, Christopher G. Atkeson, and Sethu Vijaykumar. Rear-time robot learning with locally weighted statistical learning. *International Conference on Robotics and Automation*, April 2000.

[25] Jonathan Richard Shewchuk. An introduction to the conjugate gradient method without agonizing pain. Technical report, Carnegie Mellon University, 1994.

[26] J. C. Simon. Spoken language generation and understanding. In *Proceedings of the NATO Advanced Study Institute.* D. Reidel Publishing Company, Boston, U.S.A, 1980.

[27] Richard Sproat. *Multilingual Text-to-Speech Synthesis: The Bell Labs Approach.* Kluwar Academic Publishers, Boston, U.S.A, 1998.

[28] K. Stevens and C. Bickley. Constraints among parameter simplify control of klatt formant synthesizer. *Phonetics*, 1, 1990.

[29] Richard Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 1988.

[30] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning, An Introduction.* MIT Press, Cambridge, Massachusetts, 1998.

[31] Sebastian B. Thrun. Efficient exploration in reinforcement learning. Technical Report CMU-CS-92-102, Carnegie Mellon University, Pittsburgh, PA, Jan. 1992.

[32] J. Weng. The living machine initiative. Technical Report MSU-CPS-96-60, Michigan State University, East Lansing, MI, Dec. 1996.

[33] J. Weng, J. McClelland, A. Pentland, O. Sporns, I. Stockman, M. Sur, and E. Thelen. Autonomous mental development by robots and animals. *Science*, 291:599 - 600, 2000.

[34] Juyang Weng. *Mentally Developing Robots.* To appear in MIT Press, 2003.

[35] Yilu Zhang and Juyang Weng. Grounded auditory development by a developmental robot. In *Proceedings of International Conference on Machine Learning*, pages 1059–1064, 2001.

[36] Yilu Zhang and Juyang Weng. Chained action learning through real-time interactions. In *Proceedings of International Conference on Machine Learning*, 2002.