INTELLIGENT STUDENT ASSESSMENT AND COACHING INTERFACE TO WEB-BASED EDUCATION-ORIENTED INTELLIGENT EXPERIMENTATION ON ROBOT SUPPORTED LABORATORY SET-UPS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

HALİL ERDEM MOTUK

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2003

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan ÖZGEN

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Mübeccel DEMİREKLER

Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. İsmet ERKMEN

Co-Supervisor

Assoc. Prof. Dr. Aydan ERKMEN

Supervisor

Examining Committee Members

Prof. Dr. İsmet ERKMEN _____

Assoc. Prof. Dr. Aydan ERKMEN _____

Prof. Dr. Aydın ERSAK _____

Asst. Prof. Dr. Aydın ALATAN _____

Asst. Prof. Dr. İlhan KONUKSEVEN _____

ABSTRACT

INTELLIGENT STUDENT ASSESSMENT AND COACHING INTERFACE TO WEB-BASED EDUCATION-ORIENTED INTELLIGENT EXPERIMENTATION ON ROBOT SUPPORTED LABORATORY SET-UPS

Motuk, Halil Erdem
M. S., Department of Electrical and Electronics Engineering
Supervisor: Doc. Dr. Aydan Erkmen
Co-Supervisor: Doc. Dr. Ismet Erkmen

December 2003,

This thesis presents a framework for an intelligent interface for the access of robot-supported remote laboratories through the Internet. The framework is composed of the student assessment and coaching system, the experimentation scenario, and the associated graphical user interface. Student assessment and coaching system is the main feature of a successful intelligent interface for use during remote experimentation with a robot-supported laboratory setup. The system has a modular structure employing artificial neural networks and a fuzzy-rule based decision process to model the student behaviour, to evaluate the performance and to coach him or her towards a better achievement of the tasks to be done during the experimentation. With an experimentation scenario designed and a graphical user interface, the system is applied to a robotic system that is connected to the Internet for the evaluation of the proposed framework. Illustrative examples for the operation of the each module in the system in the context of the application are given and sensitivity analysis of the system to the change in parameters is also done. The framework is then applied to a mobile robot control laboratory. The user interface and the experimentation scenario is developed for the application, and necessary modifications are made to the student assessment and coaching system in order to support the experiment.

Keywords: Student Modelling, Remote Laboratories, Intelligent User Interfaces, Artificial Neural Networks, Fuzzy Systems

ÖZ

İNTERNET TABANLI ÖĞRENİM AMACI GÜDEN ROBOT DESTEKLİ LABORATUVARLARA ERİŞİM VE AKILLI DENEY YAPMA DENETİMİ İÇİN ÖĞRENCİ DEĞERLENDİRME VE YÖNLENDİRME AKILLI ARAYÜZÜ

Motuk, Halil Erdem
Yüksek Lisans., Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. Aydan Erkmen
Ortak Tez Yöneticisi: Doç. Dr. İsmet Erkmen

Aralık 2003,

Bu tezde, Internet üzerinden robot destekli laboratuvarlara uzaktan erişim için kullanılacak bir akıllı arayüz çerçevesi sunulmuştur. Söz konusu çerçeve, bir öğrenci değerlendirme ve yönlendirme sistemi, deney senaryosu ve bu sistemi destekleyen bir grafik kullanıcı arayüzünden oluşur. Öğrenci değerlendirme ve yönlendirme sistemi, robot destekli bir laboratuvar düzeneğiyle uzaktan deney yapmada kullanılacak başarılı bir arayüzün temel unsurudur. Sistem, öğrenciyi örnekleme, öğrencinin performansını değerlendirme ve öğrenciyi deney sırasında yapılması gereken görevlerin daha iyi gerçekleştirilmesi için yönlendirme işlevleri için yapay sinir ağları ve bulanık kural tabanlı karar verme işlemini kullanan birimsel bir yapıya sahiptir. Tasarlanmış bir deney senaryosu ve bir grafik kullanıcı arabirimiyle sistem, sunulan çerçevenin değerlendirilmesi için Internet'e bağlı bir robotik sisteme uygulanmıştır. Sistemdeki her birimin çalışmasını gösteren tanımlayıcı örnekler verilmiş ve sistemin değiştirgelerdeki değişime göre bir duyarlık çözümlemesi yapılmıştır. Bu çerçeve daha sonra bir mobil robot kontrol laboratuvarına uygulanmıştır. Kullanıcı arayüzü ve deney senaryosu bu uygulama için geliştirilmiş ve öğrenci değerlendirme ve yönlendirme sistemine deneyin desteklenmesi için gerekli degişiklikler yapılmıştır.

Anahtar Kelimeler: Öğrenci Örnekleme, Uzaktan Erişimli Laboratuvarlar, Akıllı Kullanıcı Arayüzleri, Yapay Sinir Ağları, Bulanık Sistemler

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURES

CHAPTER 1
INTRODUCTION

1.1. Background

Today, the Internet and especially the World Wide Web provides an ubiquitous medium for connecting devices and robots for teleoperation. These include distributed computer systems, surveillance cameras, telescopes, manipulators and mobile robots. From the educational perspective, the Internet can be used by the universities to cope with a common problem faced by institutions concerning the limited availability of expensive robotics and laboratory equipment that the students in the educational program can work with, in order to acquire a "hands-on" experience which is essential for most of the science and engineering programmes. This can be enabled by putting a robot-supported lab facility online and making it available to other students in different institutes for remote access and operation.

In this respect, distant access of robot-supported laboratories through the Internet is an important and new issue in distance education. It also defines new pathways for Internet telerobotics and teleoperation. The classical laboratory setups, which are normally open to local students, can be shared by other students, online. This implies that the setups should be assisted by robotic devices that can be manipulated through the Internet in real time. By this way, sharing of the robot

assisted experimental resources between different institutes worldwide can also be enabled and students outside campus can be provided with real lab experimentation online. In addition to that, with the added evaluation and guiding capabilities, students who are doing the experimentation online will not need a teacher or a lab assistant that may not be available in their institutions.

Providing a successful framework that can assess the student laboratory performance and guide the students for the better achievement of experiments, targeting robot-supported laboratories remotely accessible through the Internet is a research area that involves multiple disciplines of research such as teleoperation and telerobotics, intelligent control, human-computer interaction and web-based intelligent tutoring.

The field of telerobotics itself embodies mechanics, electronics, computer science, intelligent architectures, and network communication technology. Telerobotics research aims at solving the problems associated with remote access and control of the robots through the Internet or more specifically WWW. The main problem with Internet teleoperation is that, delay and throughput of the Internet are highly unpredictable, unlike traditional teleoperation where interfaces have fixed and guaranteed delays. The research in this field currently involves:

- Finding new methodologies to overcome the unpredictable delay through the Internet.
- Developing new and sophisticated interfaces that employ the aid of current technologies of web software development platforms such as Java, and augmented reality and for providing a better vision of the robot and its environment.

- Exploring new applications areas or contexts to employ Internet telerobotics.

Human computer interaction and web-based intelligent tutoring concepts come into play while implementing an online educational tool whose target is mostly unskilled or novice users. The users (the students in this context) have to be provided with tools that will be helpful in improving their skills in the targeted area. A successful web based education system should have intelligence to tackle the variation in student skills and backgrounds and it should also be able to adapt its contents according to that variation. These mentioned issues are the main concerns for web-based intelligent tutoring research area. For a robot supported laboratory the skill building is both to learn and to gain experience about the control of the robot involved in the experiment setup and to be successful in carrying out the experimentation that is required for the student in order to gain practical knowledge in the targeted area. In order to adapt the context of the experimentation to the variation in student behaviours, students should be modelled according to their skills and knowledge backgrounds. User modelling is an important aspect of both human computer interaction and web-based intelligent tutoring research areas. AI techniques can be applied to the user modelling for implementation of online experimentation framework to get useful information about the student skill and knowledge level for providing help when necessary and assessing his/her performance.

Human computer interaction field deals with enhancing the ways in which users interact with one or more computational machines through design, evaluation and implementation of interactive computing systems. From the perspective of

telerobotics or more specifically online robotic experimentation, human computer interaction field deals with providing interfaces for remote users which enable them to do the necessary manipulation successfully. There is a strong need for an intelligent interface for a framework for remote access of robot supported laboratories through the Internet. The two main reasons for that are:

1.) The need for intelligently coaching the student to achieve the goals of the experimentation successfully.

2.) The need for evaluating student's performance while carrying out the experiment.

Student evaluation, the first main issue mentioned above, is one of the key issues for a remote experimentation framework. Students who are carrying out the experimentation, online without a human assistant or a teacher, should all be evaluated according to their varying success levels. The interface should possess suitable intelligence to categorize the student according to his or her performance during the course of the experiment and possibly to evaluate whether an increase or decrease in performance is present according to the past performance of the users. Necessary grades can then be given to those students according to the performance category in which they tend to fall.

Students, while doing the experiments online by themselves should be coached just as in the case for a traditional laboratory work where the coach is a human assistant or a teacher. They can be given useful directions and recommendations in the form of messages on the interface. Another aspect of coaching is to adapt the level of the complexity of the experiment to the level of the student. Skilled students can be excluded from some parts of the experiment, where

unskilled students or students showing a poor performance can be directed to finish the fundamental parts or repeat the unsuccessful parts of the experiment. This idea coincides with the aim of using adaptive hypermedia for intelligent web-based tutoring tools, where the content of the tutor is changed adaptively to suit the student's individual needs and interests.

There are also other key aspects for a successful interface, which are:

- Having a layout that provides the student with all the necessary information about the objectives and the states of the experiment, and visual displays for aiding the users to see the state of the robot and the experimental setup.
- Providing a security mechanism that prevents unwanted and unauthorised access to protect the system from possible malicious use.

Another issue for the robot-supported online experimentation is providing a scenario for the experiment. The experiment should involve a useful scenario that is relevant to the educational context that it is applied to and which must have tasks that have different levels of complexity to be accomplished. By this way, using an intelligent interface for an online robot-supported experimentation will be justified. The educational contexts to benefit from remote experimentation can be range from mechatronics laboratories to chemistry laboratories. According to the scenario, the students can be directed to complete the levels of the experiment according to their skill level and be coached without the actual presence of a human assistant or a teacher.

1.2. Objectives and Novelties of the Thesis Work

In accordance with the issues and the needs stated in the previous section, the aim of the work given in this thesis is to build a user assessment and coaching framework for an intelligent interface in use during remote access of labs through the Internet involving telerobotics or teleoperation. The lab setup can be assisted by either a robot or any device that is connected to the Internet.

The specific goals of the approach are that:

1.) The interface should provide the student with "hands on" experimentation by using visual feedback and give the user as much freedom as possible to control the experiment;

2.) The system should evaluate the user performance, adapt the context to the level of acquired knowledge and skill of the user, and thus intelligently coach him/her to successfully do the experiment and get the most out of the experimentation.

The concepts and tools borrowed from fields such as web-based intelligent tutoring, human-computer interaction, user-adapted interaction and Internet telerobotics are necessary for the successful accomplishment of our goals in the education oriented lab access through the Internet.

The main objective of this study is, thus, to develop an intelligent interface that can be used for the Internet access of robot supported laboratory. The main differences from the previously surveyed works that are already present in the literature are that the proposed system learns how to assess based on the user behavior while providing online robotics-enhanced experimentation, and coaches him/her towards the successful achievement of the tasks while evaluating user

performances. Thus, the proposed approach is behavior-based task planning of online users by being a combination of concepts borrowed from intelligent tutoring, student modeling and Internet robotics. Some important properties of the system can be stated briefly as follows:

- From the nature of the Internet, the system serves to a diverse number of students each having different knowledge and skill levels. The system is adaptive to these different levels and provides each student with enough assistance for accomplishing the desired experiment and getting the necessary knowledge and experience.
- Assistance provided to the student is in the form of generated messages or mandatory commands such as the repetition of a previously failed step of the experiment.
- Students are assigned experiments having different complexity levels according to their past and present performances.
- The system grades students according to their performances, and stores grades and student profiles in a database.
- The system has an authentication module to ensure security and to recall a previous user from the database.

The other major aim of this thesis work is to propose new directions and methodologies in order to employ this assessment framework for the different fields of educational context where remote experimentation through the Internet is beneficial and can be supported by tele-robotics or tele-operation. From this point of view the necessary requirements for the framework are:

- The proposed experimentation scheme must correlate with the educational requirements of the respective field in order to enhance student practical knowledge. In other words, the remote experimentation framework should provide whatever the conventional experimentation in laboratories provides to the students.
- The framework should be extensible to support a variety of educational context ranging from mechatronics to chemistry and whichever field that requires a practical experimentation that follows a path of tasks that should be done in the correct order.

1.3. Methodology

The work done in this research is to find ways to fulfil the objectives stated in the previous section to provide a general framework for an interface that can be used for access to robot supported laboratories through the Internet or the WWW. The emphasis is on modelling and profiling the student behaviour and coaching the students according to this information. In addition, a useful and a relevant scenario for an online robot-supported experimentation that will be a test platform for the interface is proposed together with the robotic hardware preparation in the thesis work of Ali Osman Boyacı [46]. The framework is composed of two major parts; the student assessment and coaching system and the user interface and the experimentation setup and context.

The framework is applied to an existing remotely controlled robot system through the Internet, where it is integrated in the general Internet software for the robot. The connectivity to the Internet and the robot control parts of the software are

treated in another thesis work. [46] The student assessment and coaching system, which works as an intelligent program to model and evaluate the student performance, is called from the interface while the general experimentation program is running. In this respect, the experimentation framework and the underlying tele-robotic hardware is tightly integrated to form the general remote experimentation setup.

The student assessment and coaching system proposed in this thesis is composed of different modules which take errors done during the whole task as inputs and output informative messages sent to the experiment interface for the student such as new task assignments or repeats, as well as the user grade values subsequently added to the user record. An artificial neural network and a fuzzy-rule based decision process are used for modeling and classification of student behaviors, evaluating the student performance during the online experimentation and giving necessary grades and performance-improving directions to students.

The user interface of the framework is designed to be simple yet informative to the student and having the ability to support the needs of the tele-robotic setup that the framework is applied to.

1.4. Organization of the Thesis

Chapter 1 of this thesis presents a general background in the field of access of robot-supported laboratories through the Internet. In the background section, the multi-disciplinary nature of the problem is stated and the research fields that are associated are mentioned. The needs for a successful framework for online robotic experimentation are also mentioned in this section. The objectives and novelties of

the thesis work are stated in the next section. The chapter concludes with the statement of the methodology that is followed in this thesis work as a separate section.

In chapter 2, a literature survey of works and papers concerning the different fields in connection with providing remote access of robotic laboratory setups through the Internet is stated initially. The references are categorized according to their fields, and the influences and the differences from this proposed work are stated at the end of each respective section. A review of remote experiments in various educational fields ranging from mechatronics to chemistry is presented as a separate section. Background information on artificial neural networks and the use of them for user modelling is given next. The information starts with the fundamentals of the artificial neural networks, then elaborating on different architectures and training methods. A brief introduction to fuzzy systems is stated along with practical relevance of fuzzy modelling, rule-based fuzzy models and how to build fuzzy models.

Chapter 3 includes the student coaching and assessment system, which is the fundamental part of the intelligent experimentation interface, explained in detail. Before the treatment of each component in the system in detail, relevant topics for the implementation of the modules are explained. The use of neural networks for user modelling, rationale for using a neural network for the system and integration of fuzzy logic and neural networks, all of which form the basis for the student assessment and coaching system, are mentioned respectively with their relevance to the current application. The system is then explained in module by module basis starting with the input module of error quantization, continuing with the error

classification by artificial neural networks and lastly the decision process modules, each of which has their dedicated sections.

In chapter 4 the other aspects of the intelligent experimentation interface through the Internet framework proposed in this thesis work are dealt with. Apart from the student assessment and coaching system, the user interface constitutes an important part of the framework. The chapter begins with the treatment of the issues related to intelligent user interfaces along with the brief background information on user interfaces and relevance of this area to the proposed framework in this thesis. The choice of the software language platform to build the user interface and the application is treated in a separate section. The chapter then continues with the proposed application of the proposed system on an existing Internet robot control setup presented along with information on coding of the intelligent interface on Microsoft Visual Basic software development platform. In addition, the hardware involved in the experiment setup, which is the focus of a parallel thesis work in METU Mechanical Engineering Department, is mentioned and the proposed experimentation scenario explained in detail. The chapter ends with illustrative examples of the experimentation showing each module of the system in operation.

Chapter 5 includes a discussion on the proposed student assessment and coaching system performance and works done to analyze the system sensitivity to different user behaviours and the changing of the parameters. The sensitivity of the system to each parameter is analyzed by keeping the others constant and graphical results are presented along with the discussion on the performance.

Chapter 6 includes the extension of the remote experimentation framework, which consists of the experimentation scenario, the student assessment and coaching

system, and the graphical user interface, to another application for a remote laboratory. The application in this context is a mobile-robot programming laboratory. In each section of the chapter the development of the experimentation scenario, the application of the student assessment and coaching system, and the implementation of the user interface are separately treated in detail.

Finally, in chapter 7, conclusions and a discussion on possible future work in this area are given.

CHAPTER 2
THE BACKGROUND

2.1. Introduction

In this chapter a literature survey for the proposed framework and background information on artificial neural networks and fuzzy systems concepts, which are used to build the student assessment and coaching part of the proposed framework, are stated.

The chapter starts with the literature survey that is done on relevant areas considering the remote experimentation involving tele-robotics through the Internet. The section includes a brief survey of works done about the concepts of tele-robotics and tele-operation, stating some examples of Internet robots. Although these works are not directly related to the work proposed in this thesis, they deserve to be mentioned because they constitute the basis that the remote experimentation using devices or robots connected to the Internet is built upon, which is providing stable, reliable devices or robots that are connected and can be controlled through the Internet. Needless to mention, the proposed framework in this thesis is implemented for devices or robots that are connected and can be controlled through Internet, so it is crucial to take into account the shortcomings and requirements for those kind of system while developing architecture to fully exploit their capabilities. The survey

then continues with works that have been done on tele-robotics and tele-operation for educational purposes. The works address different aspects of remote experimentation through Internet using either devices or robots connected to the World Wide Web. Intelligent web based tutoring; the use of machine learning for user modelling, and the application of neural networks for user modelling constitutes the other areas that are included in the literature survey.

The next section of the chapter includes background information on artificial neural networks. The fundamentals, some application areas and the different architectures are mentioned briefly with the emphasis on perceptron, ADALINE and multi-layer feed-forward neural network which is used as the neural network for the work proposed in this thesis. The back-propagation training method is presented in detail as a part of the section on supervised training. The basic principles when applying a neural network based solution to a specific problem are stated in the following section. The principles stated in that section form the basis for the application of the artificial neural network for the proposed system in this thesis.

The chapter ends with the section dedicated to background information on fuzzy systems. In this section the concept of a fuzzy system is identified first section and the practical reasons for the use of fuzzy modelling are treated next. The name, fuzzy model, is used interchangeably for a fuzzy rule based system throughout this section. Among the two models for a rule based fuzzy system, especially the linguistic rule based system (model) is explained in detail in the following subsection. This is because this model provides a basis for the fuzzy rule based system used in the proposed work for this thesis. The structure of the rules, the inference and defuzzification methods are stated for this model. The next subsection

states the practical approaches for building a fuzzy rule based system. The knowledge-based design, where the system is modelled on the expert knowledge in the form of if-then rules is mentioned in detail. The steps to be taken while building a knowledge-based design are stated as well.

2.2. Literature Survey

This section states references to recent or important applications or research for robot supported online experimentation. It also presents works and research in the fields of telerobotics, human-computer interaction, web-based intelligent tutoring and user modeling. In addition, references on neural networks, fuzzy systems and their use and applications on the proposed work are included in this section as well.

Before giving references to online robot-supported experimentation applications, some important research and applications of telerobotics for enabling remote control of robots through the Internet, which is essential for online laboratories, can be stated.

The first generation of Internet robotic systems came into existence in 1994. The Mercury project was first carried out at that time to build a 6 DOF tele-manipulator, allowing users to pickup and manipulate various objects within its reach [1]. The telerobotic garden extended this idea to tending a garden situated around an Adept 6 DOF arm to allow users to dig and water the plants [2]. Various other devices have become available over time, such as the Bradford robotic telescope [3], a Web-based tele-manipulator [5], the "FortyTwo" telerobot at Manchester [6], an interactive 3D art viewing system [7], the VISIT telerobot system via computer network [8], Internet-based remote teleoperation [9], and the "MAX"

wireless teleoperation [10]. This generation of Internet robots is mainly based on robotic arms or simple mobile robots that are directly controlled by human operators. In other words, a human is in the control loop. These telerobots operate within a well-structured environment with little uncertainty and have no local intelligence. Other successful World Wide Web (WWW) based robotic projects include the Mercury project [1]. This later evolved in the Austria's Telegarden project [13], which used a similar system of a SCARA manipulator to uncover objects buried within a defined workspace. Users were able to control the position of the robot arm and view the scene as a series of periodically updated static images. The university of Western Australia's Telerobot experiment [14] provides Internet control of an industrial ASEA IRB-6 robot arm through the WWW. Users are required to manipulate and stack wooden blocks and, like the Mercury and Telegarden projects, the view of the work-cell is limited to a sequence of static images captured by cameras located around the workspace. On-line access to mobile robotics and active vision hardware has also been made available in the form of the Netrolab project [4] at the University of Reading.

The more recent Internet robots, including the Xavier- an office exploring robot at CMU [15], the Khep-on-the-Web project for open access to a mobile robot on the Internet [16], and the Museum tour-guide robot [17]. These researches focuses on autonomous mobile robots that navigate in a dynamic and uncertain environment. Supervisory control is the main focus in building this generation of the networked robots. However, how to handle multi-robot co-operation and remote robot programming remain the major challenges.

The issues and problems with the real-time monitoring and control over the Internet are discussed in [11] and [12]. The technical issues and requirements that affect the design, acceptability and performance of Internet based teleoperated systems are addressed in [11]. Such issues include, communication and the effect of delay, controllability, reliability, interfacing requirements, command structure along with software and hardware compatibility. In [12], a framework that supports Internet based monitoring and control systems was developed additionally.

From the educational perspective, the Internet has been used by the universities to cope with a common problem faced by the institutions which concerns the limited availability of expensive robotics and control equipment that the students in the educational program can work in order to acquire a "hands-on" experience. Some early works in this area includes the virtual engineering laboratory, developed by Carnegie Mellon University, which makes electronic test equipment such as oscilloscopes, function generators, etc available to users across the WWW and introduces students to the concept of remote experimentation [19], and the robot telescope project of the University of Bradford [3].(also mentioned in the above paragraph). Such early works were for mainly connecting the laboratory devices to the Internet for exploring online experimentation concept.

Recent applications and research for online robot-supported experimentation are stated in the paragraphs below.

Hu et al [18] describes the steps toward building an Internet-based robotic system for research and education in the University of Essex. The system has a standard network protocol and an interactive human-machine interface. Using a Web browser, a remote operator can control and/or program a mobile robot to navigate in

the laboratory with visual feedback and a local perceptual map via the Internet. The employment of an intuitive user interface enables Internet users to control and or program the mobile robot and to do some experiments remotely. The robot has a high degree of local intelligence to deal with the problems caused by low bandwidth and transmission delay of the Internet.

Design issues involved in providing remote users with Internet access to laboratory based hardware are stated in [20] for a virtual control and robotics laboratory that was launched on the World Wide Web (WWW). Students using networked computers can access the on-line laboratory to perform a series of interactive experiments with real-world hardware including a DC-servo motor control system and a six degree-of-freedom MA2000 robot. Simulation tools for the robotic hardware were developed using JAVA and VRML 97 to create a desktop virtual reality environment which improves the visualisation of the manipulator hardware and associated workspace. Communication between the remote user and project server via the Internet, interface electronics and control software is also discussed

In [21], an account of the design of a low cost Internet-based teleoperation system on China's Internet is presented. Using a multimedia-rich human-computer interface, combining predictive displays and graphical overlays, a series of simple tasks were performed within a simulated space environment scenario. A foundation for teleoperated science and engineering research is established, and some issues involving the time-delay associated with the Internet are addressed. The users can operate a simulated robotic teach-panel to control the movement of the robot,

observe the 3-D robotic simulation platform, and also program to control the remote robot trajectory.

Germany's DERIVE project develops a mechatronics learning environment where on-site and remote components merge into a cooperative learning process.[22] The learning environment includes a supportive web database with multimedia learning sequences providing theoretical background information, exercises and help to handle training tasks. Another German project VVL [23] focuses on research, development, testing and evaluation of virtual laboratories in engineering and computer sciences. The laboratories concentrate their multimedia and experiments both telecontrolled and teleobserved in real laboratories within the fields of automation technologies, robotics, machine tool technologies, image processing, computer sciences and communication technology. PEARL project [24] at Open University UK aims at delivering practical experimentation, where students work together over the Internet, much as they would do in a teaching laboratory.

A general framework for implementing and deploying remote experimentation solutions is presented in [25], where the motivation for remote experimentation is discussed and best practices for the selection of the physical systems in automatic control education, the client-server architecture and the user interface are suggested for an online access to an inverted pendulum at EPFL. User interface is designed as a so-called "cockpit-like interface" consisting of different sections to provide all the necessary information for the experiment. The system does not evaluate or judge the user performance. Problems occurring in conjunction with platform-free remote laboratory experiments such as scheduling time, security and authentication, analysis and visualization of the experiment data are addressed in

[26] within the context of a remote laboratory for control theory at the University of Hagen.

The main difference between the works and research stated in the above paragraphs and the proposed framework in thesis is, the framework proposed here not only enables the students to do the experimentation remotely but also evaluates their performances and guides them to better achievement and understanding of the aims of the experiment in a given context. For that reason, a methodology that is involving an intelligent web based tutoring perspective must also be considered for providing the mentioned improvements over the other remote experimentation systems.

From the intelligent web based tutoring point, the works that are referenced here are mainly on adaptive and intelligent systems that can change the contents of tutoring according to the variation in student behavior or skill level. In this respect, user modeling research takes an important role in intelligent web based tutoring, for it provides information on the students that the tutoring system needs in order adapt to the context.

[27] describes major trends in learner-adapted teaching systems towards greater learner control over the learning process. In the early teaching systems, the goal was to build a clever teacher able to communicate knowledge to the individual student. Recent and emerging work focuses on the learner exploring, designing, constructing, making sense and using adaptive systems as tools. Correspondingly, systems are being built to give the learner greater responsibility and control over all aspects of the learning, and especially over the learner model which is at the core of user-adaptation. The paper discusses several elements of the shift to greater learner

control, with a focus on the implications for learner modelling. Brusilovsky [28] reviews adaptive and intelligent technologies for web-based distance education. Available technologies and their implementations in a large-scale web based education are discussed. Another paper by Brusilovsky [29], presents the state of adaptive hypermedia, which is necessary for tackling the diversity of user profiles in a possible web based intelligent distance education system. We emphasize in our work that user modeling or user profiling is the key issue for user adaptivity in intelligent coaching.

A number of challenges for machine learning that have hindered its application in user modeling, including: the need for large data sets; the need for labeled data; concept drift; and computational complexity issues are examined in [30] and approaches to resolving them is reviewed. Reviews of generic user modeling systems over the past twenty years [31] cover user-adaptive systems, different design requirements for research prototypes and commercially deployed servers together with the analyses of different architectures. Capuano et al [32] present an “Adaptive Web Based Tutoring” used to automatically generate curricula for different students and to monitor student knowledge by testing them and assigning recovery material if necessary.

An overview of neural networks approaches to user modeling and intelligent interface [33] gathers works where artificial neural network techniques are used for the analysis, recognition, and learning of patterns generated by user interactions in different domains. The paper also emphasizes the importance of using learning capabilities of neural networks for user modeling. An empirical approach that makes use of neuro-fuzzy synergism to evaluate the students in the context of an intelligent

tutoring system is presented in [34]. A qualitative model of the student is generated, which is able to evaluate information regarding student's knowledge and cognitive abilities in a domain area. The model had been tested on a prototype tutoring system in the physics domain of vertical trajectory motions. An automated lab instructor (ALI) is developed in [35] to provide guidance to students working with the virtual labs. The system has a representation of the key relationships in the simulation model that the student should learn, and it uses this knowledge to interleave its teaching opportunistically with the student's own discovery learning. Specifically, it can recognize learning opportunities in a student's experiments, test the student's resulting understanding, and gently guide the student towards these learning opportunities when necessary.

2.3. Remote Laboratories in Educational Contexts

Our developed student assessment and coaching framework can be applied to many remote experimentations in the university undergraduate or postgraduate education ranging from mechatronics to physics and chemistry. In the following sub-sections, important remote experimentation schemes that have been implemented in some major fields of university education will be overviewed in detail.

2.3.1. Mechatronics

Robotic and mechatronic systems are complicated control systems that combine mechanical, electronic, and computer components such as motors, cameras, sensors, actuators, software, and data acquisition hardware. The experiments in the mechatronics field involve the control of the robotic hardware either a mobile robot or a fixed industrial robot.

The remote experiments that can be done with the aid of a mobile robot which will aid in teaching mobile robotics are: [48]

- Environment perception with ultrasonic sensors and a laser range finder.
- World modelling with uncertainty grids
- Localization
- Geometrical world modelling
- Trajectory planning
- Trajectory modeling
- Reactive control-simple behaviors
- Reactive control-behavior fusion
- Hybrid architectures

These experiments, while not fully implemented, involve the use of sensors, robot control and navigation programming.

In another remote experimentation scheme involving mechatronics [49], the concepts related to brushless dc motors are aimed to be taught by experimentation. In the experimentation scheme, 4 brushless dc motor experiments were developed. These are:

* 120 and 180-degree operation
* Delta and star connection
* CW and CCW rotation
* Correct Hall sensor position
* Measurements of line-to-line voltage

In the experiments, the student chooses between 120 and 180-degree operation in either CW or CCW rotation from the user interface. Moreover, the

position of the Hall sensors can be changed by the student by the appropriate buttons on the interface, and their position can be seen on the video feed. The line-to-line voltage and the neutral position voltage can be observed on the oscilloscope window that is also integrated into the user interface.

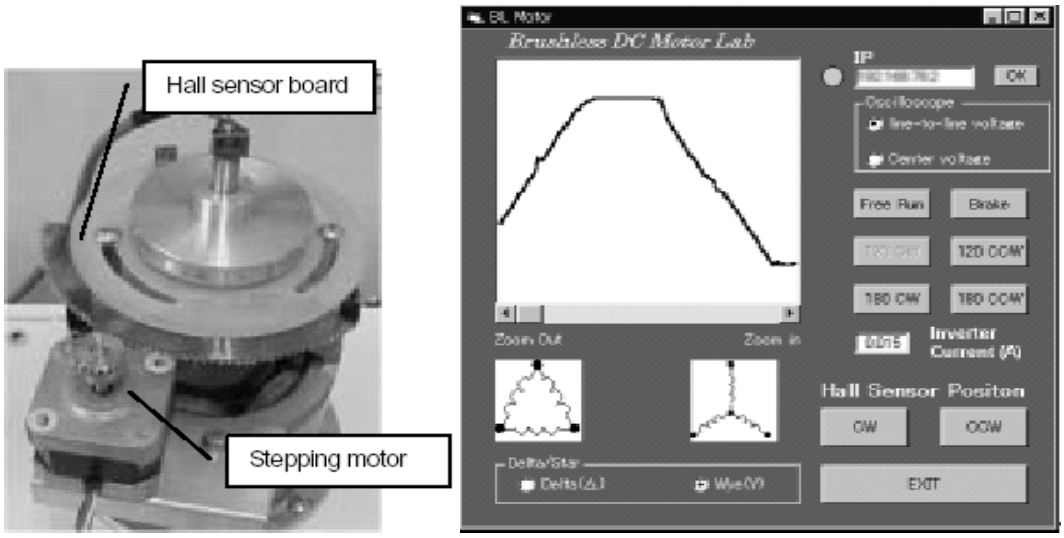


Figure 2.1: The brushless dc motor setup and the user interface for the remote experimentation [49].

Figure 2.1 shows the brushless dc motor setup and the user interface for this remote experimentation scheme. The experiments involve measuring the voltages by correctly positioning the hall sensors. In the 120-degree mode with the correct sensor positions, the input current will be minimized, the neutral potential will be triangularly-shaped, and the line-to-line voltage trapezoidal-shaped. Neutral point here means the common terminal of the three coils in the star connection. If the sensor is positioned correctly, the terminal voltage's waveform and frequency and the input current do not change even when the rotating direction is reversed. If the Hall sensor position is shifted to either direction, the line-to-line voltage will change,

and there will be a noticeable speed difference when the revolving direction is reversed.

2.3.2. Chemistry

Being an experimental subject, distance education for chemistry can not necessarily be without actual physical experiments. In the case of this field, simulations can not substitute for the practical experience because they are based on theoretical constructs and major differences between the simulated data and the actual data may exist.

The remote experimentation schemes in chemistry mainly depend on tele-presence, which is remotely observing the experiment and taking measurements and getting sensor data with the aid of data acquisition cards and the related software, such as LabView, installed on the server computer. The students can connect to the server by visiting the web page of the experiment and executing the web applet that controls the experiment server.

A particular remote experimentation scheme involves a remote-control reaction kinetics experiment suitable for chemistry students at the undergraduate or advanced secondary level. [50] The apparatus monitors the course of a fast chemical reaction spectrophotometrically using a simple continuous flow method. Students can use the data they collect to determine the rate laws for the reaction under a variety of conditions. By collaboratively analyzing the results of a large group of experiments performed by peers, students can propose a mechanism for the reaction. The experiment involves specialized apparatus that is unavailable in most undergraduate laboratories. Rapid data acquisition can allow many users to perform

experiments in real-time. The schematic diagram of the experimentation apparatus is depicted in figure 2.2.

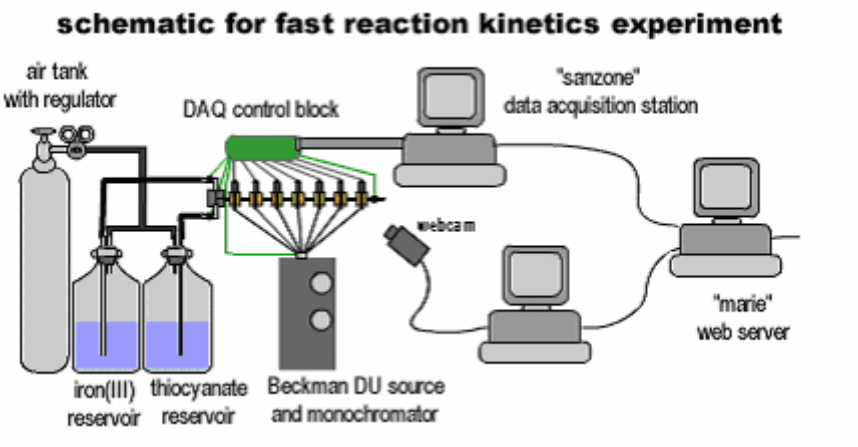


Figure 2.2: The schematic diagram of the experiment setup. Directly taken from [50].

The experiment described above is started by mixing two reactant solutions at the will of the student that is carrying out the experiment. From this point of view, the student actions are starting the experiment, watching the state of the experiment via webcam and getting the results and the sensor data in the end from the LabView program.

2.3.3. Physics

There are several remote experimentation schemes in physics, where generally the actual experimentation setups are costly and thus remote experimentation is necessary. The experimentation schemes involves driving a chaotic pendulum [51], supersonic jet thrust demonstration [52], and resonance [53]. There are also experiments in the field of optics, which are diffraction [54], and

liquid crystal optics [55], where the experimentation involves starting up the devices remotely and getting the results after the experiment finishes. Experiments in the astrophysics field are also possible which involves the use of a remotely controlled robotic telescope.

2.3.4. Electronics

The remote experiments in the electronics are aimed at providing access to the expensive measurement and data acquisition instruments such as multi-meters, signal generators, oscilloscopes and logic analyzers where a simulation of the experiments on the computer will not constitute for the real physical experiments in the circuit theory field. [56] In the remote experimentation schemes, these instruments are connected to the Internet, and test circuits can be formed and the necessary test points can be connected to the instruments using remotely controllable switch units or robots. In the experiments, either conventional instruments that are connected to the server computer via a shared bus or computer based instruments, which can only be controlled through a user interface program on the computer that they are installed in, having only test probes in their front panel, can be used. In the latter case development for the remote experimentation is easier because the instruments are already controlled by their graphical user interfaces, so there is no need to implement a separate control interface and since the instruments are already installed on the server computer, the connection between the instruments and the server computer is much faster.

As previously mentioned, the experiments related to circuit theory can be remotely done in this context. Robots can be used to set the circuits and connect the

test probes to correct points in the circuits. These actions can also be done by remotely controllable switches within a specially designed experiment setup, such that activating a switch means connecting two parts together, those parts can either be test probe leads or circuit elements.

2.3.5. Control Theory

In control theory field, a process can be controlled remotely by sending control commands remotely by a web user interface and the results can be received by the data acquisition hardware/software and the visual feedback can be provided by web cams connected to the Internet sending visual data to the web based user interface or any other Internet program, such as NetMeeting.

The remote experiments in the control theory field involve testing and applying different control schemes such as proportional (P), proportional-integral (PI), and proportional-integral-differential (PID) control to a physical process, such as a water tank [57] or a dc motor.

For a water tank system, the water level in the tank is to be controlled. Experimentation with the water tank system also involves the system identification, which is determining the tank capacitance, flow resistance of valve and piping, and pump characteristics. [57] Figure 2.3 shows the water tank system setup.

The control parameters can be applied through the user interface either by using sliders on the user interface that changes the control parameters or by typing the control the parameters in the provided boxes in the user interface and sending those parameters to the experimentation server by clicking on the send button. The results can be collected by the data acquisition hardware and sent to the

experimentation server where they are collected, tabulated, can be put into graphical form, and then sent to the user interface on the client computer for the students to see the results.

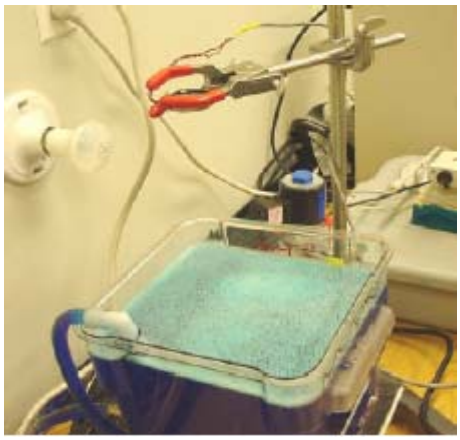


Figure 2.3: The water tank experimentation setup

2.4. Artificial Neural Networks

Artificial neural networks are nonlinear information (signal) processing devices that are built from interconnected elementary processing devices called neurons. An artificial neuron is a p-input single-output signal processing element which can be thought of as a simple model of a non-branching biological neuron.

An artificial neural network is comprised of architecture and a learning algorithm. The architecture is the arrangement of the neurons within the network, i.e. how they are linked together. The learning algorithm is the program that each neurone runs every time the network is executed.

Neural networks are practical and popular devices for solving problems that are intractable for traditional computer science approaches [36]. Their applications can be broken down into the following different approaches [37]:

- The task of pattern classification that is assigning an input pattern represented by a feature vector to one of pre-specified classes. Discriminant functions or decision boundaries are constructed from a set of training patterns with known class labels to separate patterns from different classes. Two important issues in a pattern classification task are feature representation/ extraction and decision making.
- The task of clustering, which is also known as unsupervised pattern classification, where there are no training data with known class labels. A clustering algorithm explores the similarity between the patterns and places similar patterns in a cluster.
- The task of function approximation is to find an estimate of an unknown function, given a set of n labelled training patterns (input-output pairs) which are generated from the unknown function subject to noise. The estimated function can be made to fit the training data with an arbitrary accuracy by adjusting its complexity.
- The task of prediction, where given a set of n samples in a time sequence, predicting the next sample at some future time.
- Forming associative memories or content-addressable memories that can be accessible by their content. The content in the memory can be recalled even by a partial input or distorted content.

The neural network in this thesis attempts to model students using their behaviour during online experimentation, which is a classification problem. A review of relevant literature, most notably in [38] and [39], it has been conclusively stated that the multi-layer feed forward architecture (MLFF) is the best architecture for the vast majority of neural network applications. Theoretically, a MLFF is capable of learning anything that can be learned by a neural network, although this does not mean that it is necessarily the most efficient model to employ.

2.5. The Basic Structure of an Artificial Neural Network

As already mentioned before, an artificial neural network is made up of interconnected elementary processing elements that are called neurons. A neuron in an artificial neural network is an analogy to a neuron in the human brain. Graphically, an artificial neuron is presented in several forms which are shown in figure 2.4. [41]

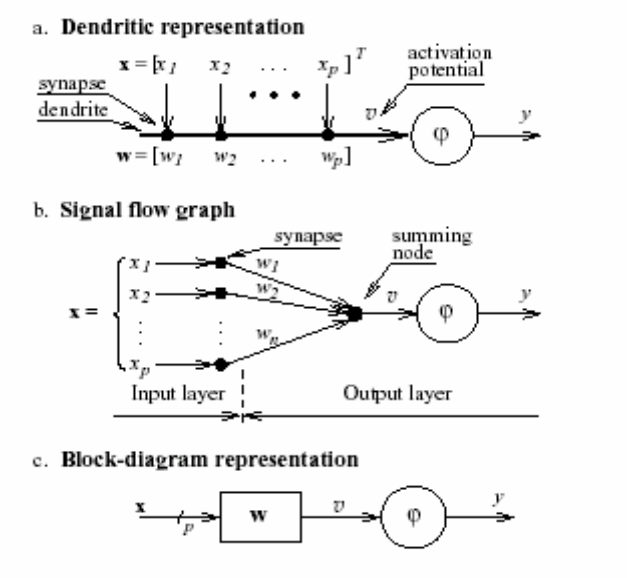


Figure 2.4: Three different forms an artificial neuron can be represented as.

From a dendritic representation of a single neuron, p synapses arranged along a linear dendrite which aggregates the synaptic activities and a neuron body generating an output signal can be identified.

The pre-synaptic activities, which correspond to the inputs of the neuron, can be represented as a p-element column vector.

$$\mathbf{x} = [x_1 \ \ldots \ x_p]^T$$

In other words the space of input patterns is p-dimensional. Synapses are characterized by weights, which are arranged in a p-element row vector.

$$\mathbf{w} = [w_1 \ \ldots \ w_p]$$

In a signal flow representation of a neuron p synapses are arranged in a layer of input nodes. A dendrite is replaced by a summing node. Weights are attributed to connections between the input nodes and the summing node.

Passing through the synapses and the summing node, the input signals are combined into the activation potential, which is a linear combination of input signals and the weights. Thus, the resulting activation potential is an inner product of weight and input vectors.

$$v = \sum_{i=1}^{p} w_i x_i = \mathbf{W} \cdot \mathbf{X} = \begin{bmatrix} w_1 & w_2 & \cdots & w_p \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix}$$

Subsequently, the activation potential is passed through an activation function, $\varphi(.)$ that generates the output signal.

$$y = \varphi(v)$$

The activation function is a typically saturating function which normalizes the activation potential to the standard values of output. The major types of activation functions can be stated as:

- A linear activation function, where y = w. Such linear processing elements are studied in the theory of linear systems.
- A step function, which can be either unipolar or bipolar, shown in figure 2.5.
- A step function with bias, where a bias can be added to both unipolar and biplolar cases. Then the neuron is said to be fired when the activation potential exceeds the bias.

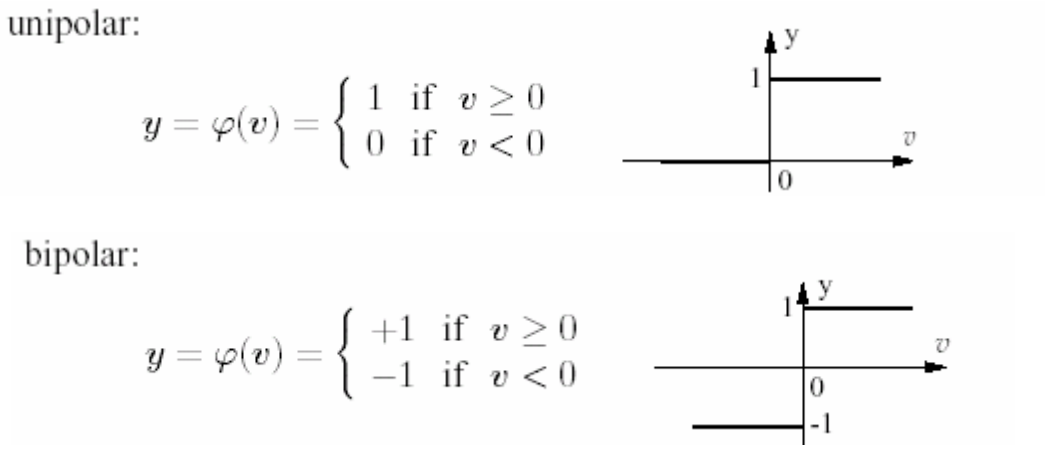


Figure 2.5: Unipolar and bipolar step functions

- A piecewise linear function, where for a small activation potential the neuron acts a linear combiner with a gain, for large activation potential the neuron saturates and generates the output 1. An example is shown in figure 2.6.

$$y = \varphi(v) = \begin{cases} 0 & \text{if } v \leq -\frac{1}{2\alpha} \\ \alpha v + \frac{1}{2} & \text{if } |v| < \frac{1}{2\alpha} \\ 1 & \text{if } v \geq \frac{1}{2\alpha} \end{cases}$$

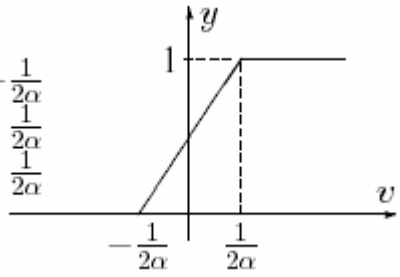
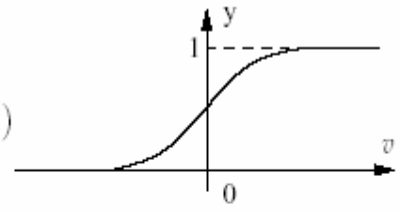


Figure 2.6: A piecewise linear activation function

- Sigmoidal functions, which have unipolar and bipolar types, shown in figure 2.7. The bipolar sigmoidal function is the most popular choice for an activation function specifically in problems related to function mapping and approximation.

unipolar:

$$\varphi(v) = \frac{1}{1+e^{-v}} = \frac{1}{2}(\tanh(v/2) - 1)$$



bipolar:

$$\varphi(v) = \tanh(\beta v)$$

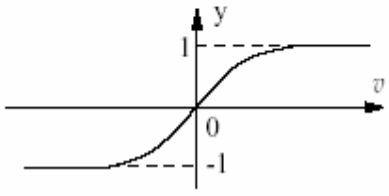


Figure 2.7: Unipolar and bipolar sigmoidal activation functions.

Neurons can be arranged into a layer of neurons. This can be thought of as a p-input single layer neural network consisting of m neurons. Similarly to a single

neuron, the network can be represented in all three basic forms: dendritic, signal-flow and block diagram form as shown in figure 2.8. [41]

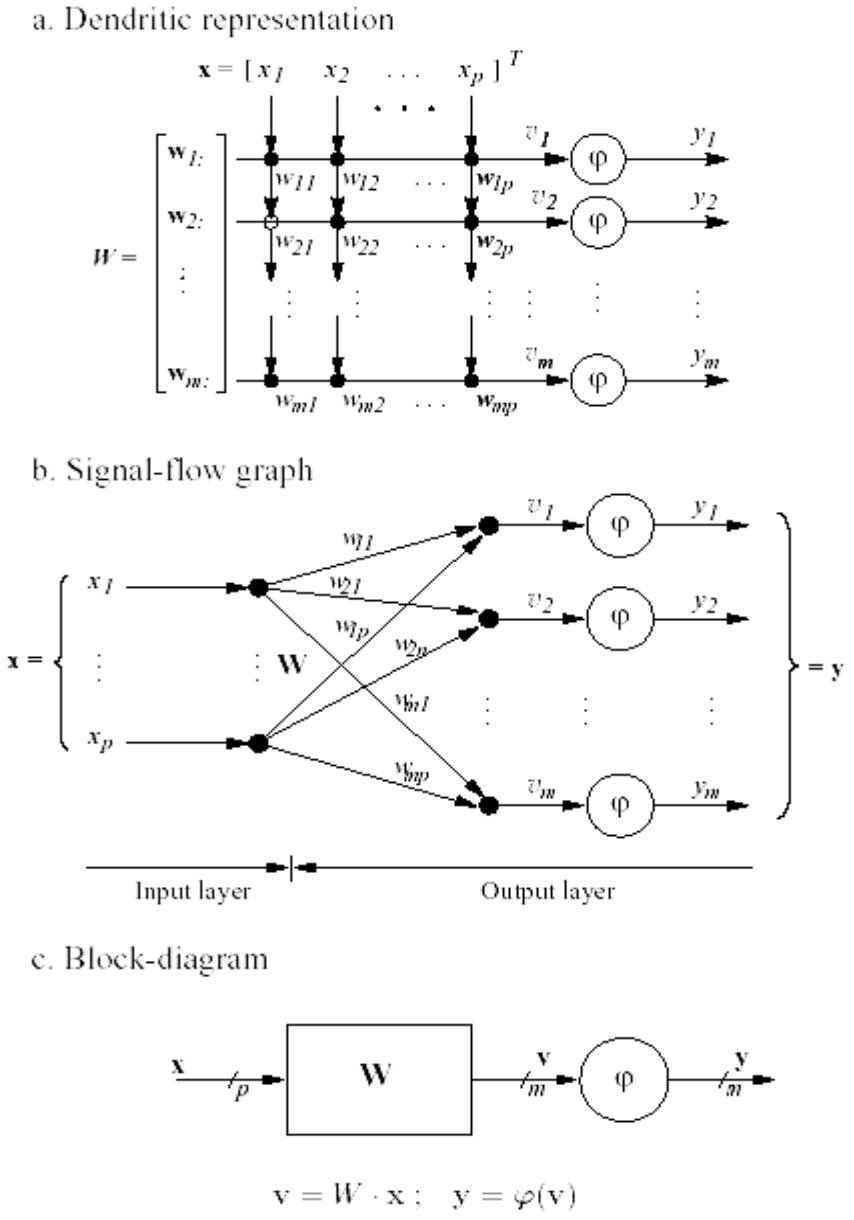


Figure 2.8: Three basic forms a neural network can be represented as.

From the figure, it seen that a layer of neurons is described by a m **x** p matrix, W, of synaptic weights. Each row of the weight matrix is associated with one neuron. Operations performed by the network can be described as follows:

$$\begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_m \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{1p} \\ w_{21} & \cdots & w_{2p} \\ \vdots & \cdots & \vdots \\ w_{m1} & \cdots & w_{mp} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_p \end{bmatrix} ; \quad \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix} = \begin{bmatrix} \varphi(v_1) \\ \varphi(v_2) \\ \vdots \\ \varphi(v_m) \end{bmatrix}$$

or equivalently in matrix form as:

$v = W \cdot x ; \quad y = \varphi (W \cdot x) = \varphi (v),$

where v is a vector of activation potentials. Each weight parameter w_{ij} is related to a connection between the nodes of the input layer and the output layer. Therefore, the name, connection strengths, for the weights is also justifiable.

2.6. Neural Network Architectures

This section makes a brief review of different neural network structures starting with the basic units of perceptron and ADALINE which formed the basis for the later structures such as multi-layer feed-forward neural network that is used in the proposed work in this thesis. The other structures of neural networks are briefly mentioned in order to keep integrity. The emphasis is on the multi-layer feed-forward neural network.

2.6.1. The Perceptron, ADALINE

The perceptron was introduced by McCulloch and Pitts in 1943 as an artificial neuron with a hard-limiting activation function, σ. The structure of a perceptron is depicted in figure 2.9.

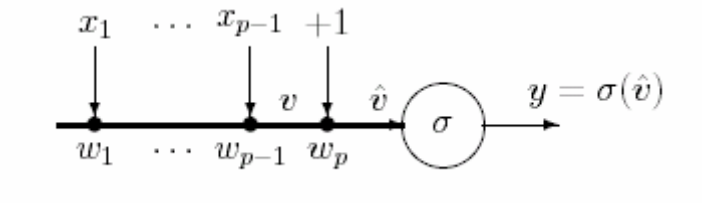


Figure 2.9: The perceptron

Input signals x_i, are assumed to have real values. The activation function, σ, is a unipolar step function (sometimes called the Heaviside function), therefore, the output signal is binary, y 2 {0, 1}. One input signal is constant (x_p = 1), and the related weight is interpreted as the bias, or threshold, θ. The input signals and weights are arranged in respective column and row vectors. Aggregation of the proper input signals results in the activation potential, v, which can be expressed as the inner product of proper input signals and related weights.

$$v = \sum_{i=1}^{p-1} w_i x_i = \mathbf{W}_{1:p-1} \cdot \mathbf{X}_{1:p-1}$$

For each input signal, the output is determined as:

$$y = \sigma(\hat{v}) = \begin{cases} 0 & \text{if} \quad v < \theta \quad (\hat{v} < 0) \\ 1 & \text{if} \quad v \geq \theta \quad (\hat{v} \geq 0) \end{cases}$$

Hence, a perceptron works as a threshold element, the output being active if the activation potential exceeds the threshold. A perceptron can be thought of as a classifier, which classifies the input patterns, x, into two classes.

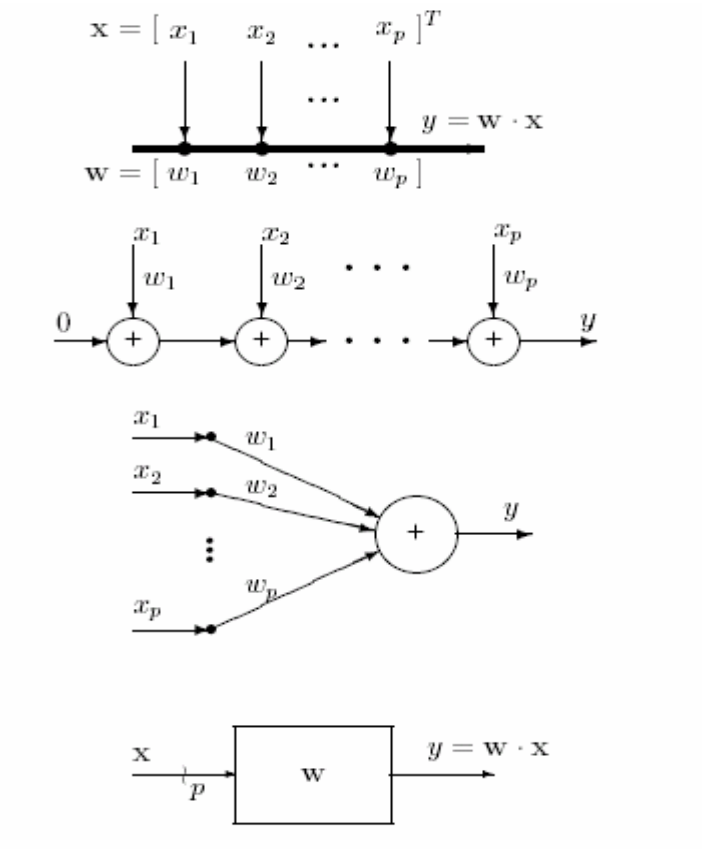


Figure 2.10: Various representations of ADALINE

ADALINE, the adaptive linear element, can be thought of as the smallest, linear building block of the artificial neural networks. The following conventions are used to illustrate the structure of ADALINE, which are shown in figure 2.10. [41]

In general ADALINE is used to perform:

- Linear approximation of a small segment of a non-linear hyper-surface, which is generated by a p-variable function, y = f(x). In this case bias is usually needed, hence wp =1.
- Linear filtering and prediction of data.
- Pattern association, that is generation of m-element output vectors associated with respective p-element input vectors.

2.6.2. The Hopfield Network

Hopfield network can be used to store information that may be retrieved by supplying only a portion of the information at the network's input. The network is robust in that some of the neurons may be partially damaged without, necessarily, impairing the stored memory. The operation of the Hopfield network is quite different from other networks in that the learning process is a highly iterative. Each neuron has connections that are excitatory or inhibitory and the state being either active or inactive. Its state is calculated based on the states of other neurons around its connected peripheral and its own state. Neurons are computed at random in order to finalise the state of the network.

2.6.3. The Boltzman Machine

The Boltzman Machine is a Hopfield Network with the addition of simulated annealing. The Hopfield network is not suitable for problem solving because it has a tendency to settle in local minima. Simulated annealing, which is an analogy to a metallurgic process of heating the metal to a very high temperature and then cooling it very slowly, is added to prevent the network from being trapped in a local minima. The simulated annealing approach is analogous to shaking the network to dislodge it

from a local minimum towards global minima with the learning rate gradually reduced to prevent the jumps when global minimum is reached.

2.6.4. The Multi-Layer Feed Forward Network

The Multi-Layer Feed Forward (MLFF) architecture is the backbone of modern neural networks. There are many variations, however in essence the architecture and learning algorithm are simple and fast. [40]

A MLFF network is comprised of several layers, an input layer, an output layer and one or more hidden layers. The input neurons are not actual neurons, as they perform no processing on the incoming data while hidden and output neurons are computational nodes, which yield a shaping of data by the output function. Figure 2.11 depicts the block diagram of a single hidden layer neural network having p inputs, L hidden layer neurons and m output neurons. [41]

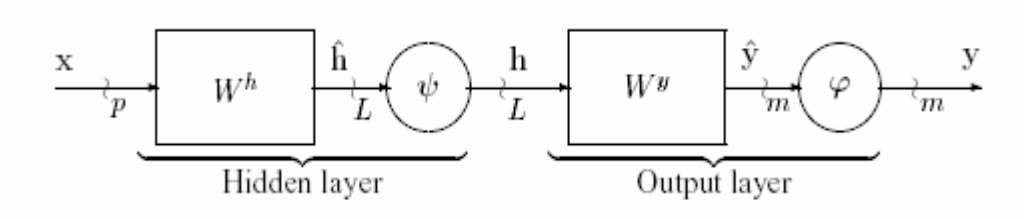


Figure 2.11: Block diagram of a feed-forward neural network having a single hidden layer.

The term feed forward relates to the fact that all neurons can only be connected uni-directionally to one or more neurons in the next layer (with the obvious exception of the output neurons). Each connection has a weight associated

with it; this is added to the signal which passes along the connection. Weights can be positive, excitatory, and negative, inhibitory. The weights in the hidden layer can be represented as the hidden weight matrix, W^h. Each neuron in the hidden layer calculates its activation by summing up its weighted inputs, which forms the hidden activation signal that is applied to neurons activation function. Once the output of every neuron in hidden layer has been calculated then the next layer can be calculated since the outputs of the current layer form the inputs of the next. Once all the neurons have been processed then the output neurons finally present their results, which may or may not be as desired. In the case where a discrepancy between the actual and the desired results occurs then learning is required.

The neural network gains its remarkable abilities because of the complex mathematical system that the combined neurons generate. One could argue that the biggest asset of connectionist approaches is also its biggest drawback, since their complexity makes them almost impossible to analyze or predict mathematically.

2.7. Supervised Learning

2.7.1. The Back Propagation Learning Algorithm

Learning in neural networks can be either supervised, unsupervised or hybrid. All MLFF networks are trained by the supervised method where a benchmarking exists to which the actual results are compared- the benchmark is the supervisor.

Back propagation method is in fact a mathematical technique for calculating errors in a complex mathematical system, such as a neural network. Such algorithms map the function onto a three-dimensional surface, with low land valleys and up land

hills. Depending on the problem, the lower is the point on the landscape the better is the output of the function.

Supervised training involves sampling a representative sample of examples from the problem domain, this is then split into the training set and the validation set. The network is initialised by setting its weights to random values. The network then alters these weights in response only to the training set. The validation set is used to determine that the network is successful with examples from the same population but with which it has not been trained. The network is presented with a subset of the training set, one example at a time; after it has "seen" the whole subset then it updates its weights in response to the measure of error incurred by the network. This process, termed an epoch, is repeated until the network is sufficiently trained. The size of the subset is termed the epoch size and may be the same as the training set size, if it is not then it is important to choose each member of the subset randomly from the training set.

The measure of error is the mean square error of output activations; it is calculated by squaring the difference between the actual and target activations and then finding the average across all output neurones. It is possible to determine which way the weights must move in order to reduce the error by finding the partial derivative of the error. The size of the step taken in that direction is termed the learning rate, if too big a step is used the optimum weights may be stepped over, too small and the optimum weights may take an eternity to reach.

The back propagation algorithm can mathematically be explained using the most common two layer feed-forward neural network in figure 2.12, which can also be called a two-layer perceptron structure. [41]

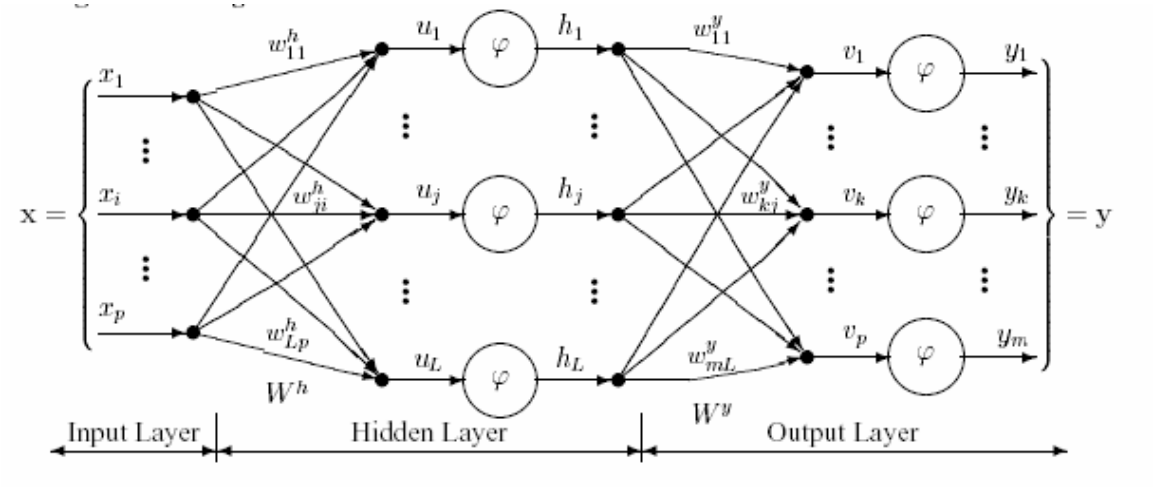


Figure 2.12: A two-layer feed-forward neural network

From the figure it can be seen that:

$$u_j = W^h_{j:} \cdot \mathbf{x}\,; \quad h_j = \varphi(u_j)\,; \quad v_k = W^y_{k:} \cdot \mathbf{h}\,; \quad y_k = \varphi(v_k)$$

Training data is organized as for the linear neurons, that is , there are N input vectors, x(n), and N desired output vectors, d(n), in two matrices:

$$\begin{aligned} X &= [\,\mathbf{x}(1) \;\ldots\; \mathbf{x}(n) \;\ldots\; \mathbf{x}(N)\,] \quad \text{is} \quad p \times N \quad \text{matrix,} \\ D &= [\,\mathbf{d}(1) \;\ldots\; \mathbf{d}(n) \;\ldots\; \mathbf{d}(N)\,] \quad \text{is} \quad m \times N \quad \text{matrix} \end{aligned}$$

For each input vector, x (n), the network calculates the actual output vector, y (n). For the whole training epoch the output vectors can be collected in a m x N matrix.

$$Y = [\,\mathbf{y}(1) \;\ldots\; \mathbf{y}(n) \;\ldots\; \mathbf{y}(N)\,] \quad \text{is} \quad m \times N \quad \text{matrix}$$

The complete set of the output and the hidden vectors can also be calculated as:

$$Y = \varphi(W^y \cdot H)\,; \quad H = \psi(W^h \cdot X)$$

The output error at the nth step

$$\varepsilon(n) = [\varepsilon_1(n) \dots \varepsilon_m(n)]^T = \mathbf{d}(n) - \mathbf{y}(n) \quad \text{is an } m \times 1 \text{ vector,}$$

each component being equal to:

$$\varepsilon_k(n) = d_k(n) - y_k(n) \quad \text{for } k = 1, \dots, m$$

As for the linear network, the objective is to find a set of weight matrices, W^h, W^y, such that the errors are as small as possible. The aim is trying to minimize the mean square error, which is specified as an averaged sum of instantaneous squared errors at the network output.

$$J(W^h, W^y) = J(\mathbf{w}) = \frac{1}{m\,N} \sum_{n=1}^{N} E(\mathbf{w}, n)$$

where the total instantaneous squared error, E (w, n) is defined as:

$$E(\mathbf{w}, n) = \frac{1}{2} \sum_{k=1}^{m} \varepsilon_k^2(n) = \frac{1}{2}\, \varepsilon^T(n) \cdot \varepsilon(n)$$

In the steepest descent learning algorithms the weight update should be made proportional to the gradient of the mean-squared error with respect to the total weight vector, w formed from all the elements of the weight matrices. This gradient can be calculated as:

$$\nabla J(\mathbf{w}) = \frac{1}{m\,N} \sum_{n=1}^{N} \nabla E(\mathbf{w}, n)$$

where $\nabla E(\mathbf{w}, n)$ is the gradient vector of the total instantaneous error, which is:

$$\nabla E(\mathbf{w}, n) = \left[\frac{\partial E}{\partial w_{11}^h} \cdots \frac{\partial E}{\partial w_{ji}^h} \cdots \frac{\partial E}{\partial w_{Lp}^h} \; \frac{\partial E}{\partial w_{11}^y} \cdots \frac{\partial E}{\partial w_{kj}^y} \cdots \frac{\partial E}{\partial w_{mL}^y} \right]$$

In the training methodology that is also followed for the proposed work in this thesis, the weight update is made proportional to the gradient of the total instantaneous error, and the weights are modified after the presentation of every training pattern as:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \eta \nabla E(\mathbf{w}, n)$$

For the whole back propagation algorithm, the relevant equations are:

$$\begin{aligned} W^y(n+1) &= W^y(n) + \eta_y \cdot \delta(n) \cdot \mathbf{h}^T(n) \\ W^h(n+1) &= W^h(n) + \eta_h \cdot \delta^h(n) \cdot \mathbf{x}^T(n) \end{aligned}$$

where n is the pattern index, y denotes the output layer, h denotes the hidden layer, **h** is the vector of hidden layer outputs, δ is the output error vector modified with the derivative of the activation function.

The biggest drawback of the traditional back propagation algorithm is that it is heavily influenced by the local gradient; it does not necessarily take a direct route. For example, if the global minimum was located at the end of a valley and the current position is at the side, above the valley, and then the back propagation algorithm would take a step in the direction of greatest gradient, over the valley side. Once it had discovered the other edge of the valley it would zig-zag back and forth making small steps towards the goal. The final route taken can be thousands of time longer than the shortest route, and hence learning time is thousands of times longer.

The standard back propagation algorithm can be improved by adding a momentum term into the equation. This effectively filters out local gradients and allows the more global gradient to take influence.

The step size, or learning rate, of the standard back propagation algorithm is fixed, this leads to the algorithm "thrashing about" around the minima as the algorithm can not precisely find the bottom of two gradients. I.e. it steps down one, over the minimum and lands half way up the gradient at the other side. The Conjugate Gradient method allows the learning algorithm to make progressively smaller steps as it approaches a minimum, thus it can reach a point close to the actual minimum very quickly.

2.8. Neural Network Design Principles

This section states general design principles while applying a neural network approach to a specific problem. The issues stated in this section relate especially to multi layered feed forward neural networks and they were considered when building the architecture for the artificial neural network, which is a part of the system proposed in this thesis.

The multi-layer feed forward network is comprised of an input layer, one or more hidden layers of neurones and an output layer of neurones. There are no theoretical guidelines that enable us to determine what architecture a particular problem requires. The only course of action therefore is to experiment with several variations. Consulting the practical neural network literature [42], [43] gives the following advice: In practical terms it is never necessary to have more than two hidden layers and uncommon to require more than one. Mathematical theory also unequivocally states that a two hidden layer neural network is capable of learning any problem capable of being learned by a neural network [40]. This does not necessarily mean that a two hidden layer is the best architecture for the problem

however; adding additional layers may make the network much slower in operation and learning, as well as perhaps introducing the problem of over fitting.

The number of neurones in the hidden layer is crucial to the performance of the neural network and is finely poised between ability and speed of learning. The more neurones the more capable the network but the longer it will take to train, more neurones also increases the likelihood that the neural network will memorise its training set (over fitting), rather than learning the parts of the training set that represent the population as a whole. [40] proposes a guideline for designing neural networks, given that most networks have more inputs than outputs, then the number of hidden neurones should be midway between the two, thus giving a triangular shape.

Selecting the number of hidden neurones is often a trial and error process; however it is better to underestimate than overestimate. The often reported problem of over fitting is really a symptom of having too many hidden neurones. The extra neurones give the network the ability to memorise the idiosyncratic details of the training set as well as the general details, thus they come to focus on the insignificant details and over fit the training set. Reducing the number of hidden neurones cuts down the network's memorising power and so, if trained properly, it will recognise the significant details only. Obviously too few hidden neurones will result in poor performance. However it is easier to determine that a network has too few neurones than too many, a network with too many may perform well until it is placed in its working environment. [40]

The overtraining of a neural network is the result of a neural network over fitting a training set because it has too many neurones or the training set is too small.

The problem of over fitting may also occur if the training is not representative of the population as a whole. The number of hidden neurones and the size of the training set in inexorably related, incorrectly defining either may result in over fitting or inability to learn.

From the above it can be seen that there are a significant number of variables, each of which can not be simply mathematically selected. Instead the network designer must experiment with many combinations using the heuristics described above in an attempt to attain the perfect network. Note that it is usually only an attempt as usually a near perfect network will suffice. However all this experimentation does present a very real problem to the designers of feed forward neural networks, simply because for each permutation of the network architecture involves completely retraining the network, which is the most time consuming aspect of utilising the feed forward architecture.

The designer must carefully select the inputs for the neural network, too many and the network becomes more complex and thus slower to train and execute. However a rich set of inputs allows the network to make connections, which may not be readily apparent to the human observer. There is no easy solution to this problem; again it is simply brute force in experimenting with different combinations.

2.9. Fuzzy Systems

In this section a brief review of fuzzy systems is given with the connections to the fuzzy rule based decision system of the proposed system in this thesis. A treatment of fuzzy theory, which explains concepts such as membership functions,

operations on fuzzy sets – union, intersection and complement, fuzzy relations, max-min composition, extension principle), is left out of scope for this thesis.

A static or dynamic system which makes use of fuzzy sets or fuzzy logic and of the corresponding mathematical framework is called a fuzzy system [44]. There are a number of ways fuzzy sets can be involved in a system, such as:

- In the description of the system. A system can be defined, for instance, as a collection of if-then rules with fuzzy predicates, or as a fuzzy relation. An example of a fuzzy rule describing the relationship between a heating power and the temperature trend in a room may be: "**If** the heating power is high **then** the temperature will increase fast. "
- In the specification of the system's parameters. The system can be defined by an algebraic or differential equation, in which the parameters are fuzzy numbers instead of real numbers. As an example consider an equation: y = ~3x1 + ~5x2; where ~3 and ~5 are fuzzy number "about three" and "about five", respectively, defined by membership functions. Fuzzy numbers express the uncertainty in the parameter values.
- The input, output and state variables of a system may be fuzzy sets. Fuzzy inputs can be readings from unreliable sensors ("noisy" data), or quantities related to human perception, such as comfort, beauty, etc. Fuzzy systems can process such information, which is not the case with conventional (crisp) systems.

A fuzzy system can simultaneously have several of the above attributes. Table 2.1 gives an overview of the relationships between fuzzy and crisp system

descriptions and variables. In this section, the focus will be on the last type of systems, i.e., fuzzily described systems with crisp or fuzzy inputs.

Table 2.1: Crisp and fuzzy information in systems.

System description	Input data	Resulting output data	Mathematical framework
Crisp	Crisp	Crisp	Functional analysis, linear algebra, etc
Crisp	Fuzzy	Fuzzy	Extension principle
Fuzzy	Crisp/ fuzzy	Fuzzy	Fuzzy relational calculus, Fuzzy inference

Fuzzy systems can be regarded as a generalization of interval-valued systems, which are in turn a generalization of crisp systems. This is depicted in figure 2.11, which gives an example of a function and its interval and fuzzy forms. The evaluation of the function for crisp, interval and fuzzy data is schematically depicted as well. A function **f: X → Y** can be regarded as a subset of the Cartesian product X x Y, i.e., as a *relation*. The evaluation of the function for a given input proceeds in three steps:

1) Extend the given input into the product space X x Y (vertical dashed lines in Fig. 2.11).

2) Find the intersection of this extension with the relation.

3) Project this intersection onto Y (horizontal dashed lines in Fig. 2.13).

This view is independent of the nature of both the function and the data (crisp, interval, fuzzy).

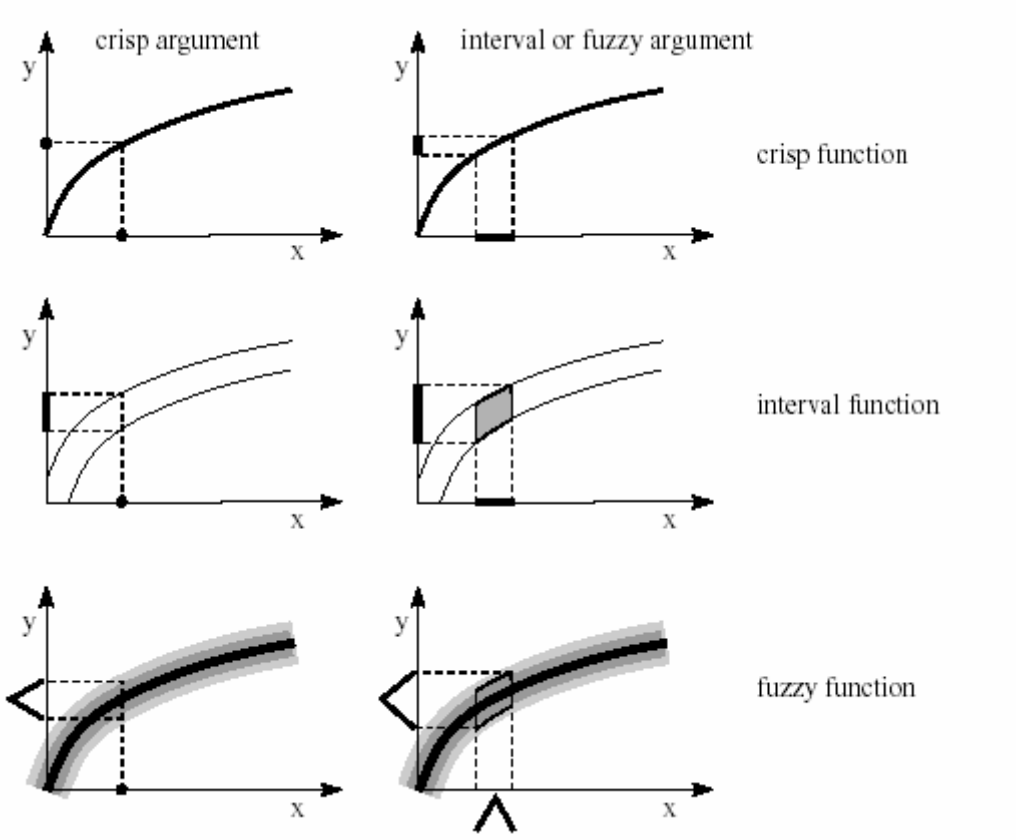


Figure 2.13: Evaluation of a crisp, interval and fuzzy function for crisp, interval and fuzzy arguments.

Most common are fuzzy systems defined by means of if-then rules: rule-based fuzzy systems. Fuzzy systems can serve different purposes, such as modeling, data analysis, prediction or control. In this text a fuzzy rule-based system is for simplicity called a *fuzzy model*, regardless of its eventual purpose.

2.10. Practical relevance of fuzzy modeling

***Incomplete or vague knowledge about systems**:* Conventional system theory relies on crisp mathematical models of systems, such as algebraic and differential or difference equations. For some systems, such as electro-mechanical systems,

mathematical models can be obtained. This is because the physical laws governing the systems are well understood. For a large number of practical problems, however, the gathering of an acceptable degree of knowledge needed for physical modelling is a difficult, time-consuming and expensive or even impossible task. In the majority of systems, the underlying phenomena are understood only partially and crisp mathematical models cannot be derived or are too complex to be useful. Examples of such systems can be found in the chemical or food industries, biotechnology, ecology, finance, sociology, etc. A significant portion of information about these systems is available as the knowledge of human experts, process operators and designers. This knowledge may be too vague and uncertain to be expressed by mathematical functions. It is, however, often possible to describe the functioning of systems by means of natural language, in the form of if-then rules. Fuzzy rule-based systems can be used as knowledge-based models constructed by using knowledge of experts in the given field of interest. From this point of view, fuzzy systems are similar to expert systems studied extensively in the "symbolic" artificial intelligence.

Adequate processing of imprecise information*:* Precise numerical computation with conventional mathematical models only makes sense when the parameters and input data are accurately known. As this is often not the case, a modeling framework is needed which can adequately process not only the given data, but also the associated uncertainty. The stochastic approach is a traditional way of dealing with uncertainty. However, it has been recognized that not all types of uncertainty can be dealt with within the stochastic framework. Various alternative approaches have been proposed, fuzzy logic and set theory being one of them.

***Transparent (gray-box) modeling and identification**:* Identification of dynamic systems from input-output measurements is an important topic of scientific research with a wide range of practical applications. Many real-world systems are inherently nonlinear and cannot be represented by linear models used in conventional system identification. Recently, there is a strong focus on the development of methods for the identification of nonlinear systems from measured data. Artificial neural networks and fuzzy models belong to the most popular model structures used. From the input-output view, fuzzy systems are flexible mathematical functions which can approximate other functions or just data (measurements) with a desired accuracy. This property is called general function approximation. Compared to other well-known approximation techniques such as artificial neural networks, fuzzy systems provide a more transparent representation of the system under study, which is mainly due to the possible linguistic interpretation in the form of rules. The logical structure of the rules facilitates the understanding and analysis of the model in a semi-qualitative manner, close to the way human reasoning about the real world.

2.11. Rule-Based Fuzzy Models

In rule-based fuzzy systems, the relationships between variables are represented by means of fuzzy if–then rules of the following general form:

If antecedent proposition **then** consequent proposition:

The antecedent proposition is always a fuzzy proposition of the type "$\mathcal{X}$ is $\mathcal{A}$ where $\mathcal{X}$ is a linguistic variable and $\mathcal{A}$ is a linguistic constant (term). The proposition's truth value (a real number between zero and one) depends on the

degree of match (similarity) between x and $\mathcal{A}$. Depending on the form of the consequent two main types of rule-based fuzzy models are distinguished:

- *Linguistic fuzzy model*: both the antecedent and the consequent are fuzzy propositions.
- *Takagi–Sugeno (TS) fuzzy model*: the antecedent is a fuzzy proposition; the consequent is a crisp function.

In the next section "Linguistic Fuzzy Model", which is used a base for the fuzzy rule based decision process of the system proposed in this thesis will be explained in detail. The letter fuzzy model will not be explained because it is not very relevant to the work done in this thesis.

2.11.1. Linguistic fuzzy model

The linguistic fuzzy model has been introduced as a way to capture available (semi-) qualitative knowledge in the form of if–then rules:

$$\mathcal{R}_i: \text{If } \tilde{x} \text{ is } \mathcal{A}_i \text{ then } \tilde{y} \text{ is } \mathcal{B}_i \qquad (1)$$

Here $\tilde{x}$ is the input (antecedent) linguistic variable, and, $\mathcal{A}_i$ are the antecedent linguistic terms (constants). Similarly, $\tilde{y}$ is the output (consequent) linguistic variable and $\mathcal{B}_i$ are the consequent linguistic terms. The values of $\tilde{x}$ ($\tilde{y}$), and the linguistic terms $\mathcal{A}_i$($\mathcal{B}_i$) are fuzzy sets defined in the domains of their respective base variables: $x \in X \subset \mathbb{R}^p$ and $y \in Y \subset \mathbb{R}^q$.The membership functions of the antecedent (consequent) fuzzy sets are then the mappings: $\mu(x)$: $X \rightarrow [0,1]$, $\mu(y)$: $Y \rightarrow [0,1]$. Fuzzy sets $\mathcal{A}_i$ define fuzzy regions in the antecedent space, for which the respective consequent propositions hold. The linguistic terms $\mathcal{A}_i$ and $\mathcal{B}_i$ are usually

selected from sets of predefined terms, such as Small, Medium, etc. By denoting these sets by $\mathcal{A}$ and $\mathcal{B}$ respectively, we have $\mathcal{A}_i \in \mathcal{A}$ and $\mathcal{B}_i \in \mathcal{B}$. The rule base $\mathcal{R} = \{\mathcal{R}_i \mid i = 1, 2, \ldots, K\}$ and the sets $\mathcal{A}$ and $\mathcal{B}$ constitute the knowledge base of the linguistic model.

In order to be able to use the linguistic model an algorithm, which allows computing the output value, given some input value, is needed. This algorithm is called the *fuzzy inference* algorithm (or mechanism). For the linguistic model, the inference mechanism can be derived by using fuzzy relational calculus, as shown in the following subsections.

2.11.2. Relational representation of a linguistic model

Each rule in (1) can be regarded as a fuzzy relation (fuzzy restriction on the simultaneous occurrences of values x and y): $\mathcal{R}i$: $(\mathcal{X} \times \mathcal{Y}) \rightarrow [0; 1]$. This relation can be computed in two basic ways: by using fuzzy conjunctions (Mamdani method) and by using fuzzy implications (fuzzy logic method). Fuzzy implications are used when the if-then rule (1) is strictly regarded as an implication $\mathcal{A}i \rightarrow \mathcal{B}i$, i.e., "$\mathcal{A}$ implies $\mathcal{B}$". In classical logic this means that if $\mathcal{A}$ holds, $\mathcal{B}$ must hold as well for the implication to be true. Nothing can, however, be said about $\mathcal{B}$ when $\mathcal{A}$ does not hold, and the relationship also cannot be inverted.

When using a conjunction, $\mathcal{A} \wedge \mathcal{B}$, the interpretation of the if-then rules is "it is true that $\mathcal{A}$ and $\mathcal{B}$ simultaneously hold". This relationship is symmetric and can be inverted. The Mamdani (conjunction) method is preferred for its simplicity in this thesis. The relation $\mathcal{R}$ is computed by the minimum ($\wedge$) operator:

$$\mathcal{R}_i = \mathcal{A}_i \times \mathcal{B}_i, \text{ that is } \mu_{\mathcal{R}i}(x,y) = \mu_{\mathcal{A}i}(x) \wedge \mu_{\mathcal{B}i}(y). \quad (2)$$

The minimum is computed on the Cartesian product space of X and Y, i.e., for all possible pairs of x and y. The fuzzy relation $\mathcal{R}$ representing the entire model (1) is given by the disjunction (union) of the K individual rule's relations $\mathcal{R}_i$:

$$\mathcal{R} = \bigcup_{i=1}^{K} \mathcal{R}_i \quad \text{that is,} \quad \mu_{\mathcal{R}i}(x,y) = \max_{1 \le i \le K} [\mu_{\mathcal{A}i}(x) \wedge \mu_{\mathcal{B}i}(y)]. \quad (3)$$

Now the entire rule base is encoded in the fuzzy relation $\mathcal{R}$ and the output of the linguistic model can be computed by the relational max-min composition ($\circ$):

$$\bar{y} = \bar{x} \circ R. \quad (4)$$

2.11.3. Max-min (Mamdani) inference

In the previous subsection, it can be seen that a rule base can be represented as a fuzzy relation. The output of a rule-based fuzzy model is then computed by the max-min relational composition. In this section, it will be shown that the relational calculus can be by-passed. This is advantageous, as the discretization of domains and storing of the relation R can be avoided. To show this, suppose an input fuzzy value $\bar{x} = A'$, for which the output value B' is given by the relational composition:

$$\mu_{\mathcal{B}'}(y) = \max_{X} [\mu_{\mathcal{A}'}(x) \wedge \mu_{\mathcal{R}}(x,y)]. \quad (5)$$

After substituting for $\mu_{\mathcal{R}}(x,y)$ from (3), the following expression is obtained:

$$\mu_{\mathcal{B}'}(y) = \max_{X} \{\mu_{\mathcal{A}'}(x) \wedge \max_{1 \le i \le K} [\mu_{\mathcal{A}i}(x) \wedge \mu_{\mathcal{B}i}(y)]\} \quad (6)$$

Since the max and min operation are taken over different domains, their order can be changed as follows:

$$\mu_{\mathcal{B}'}(y) = \max_{1 \le i \le K} \{ \max_{X} [\mu_{\mathcal{A}'}(x) \wedge \mu_{\mathcal{A}i}(x)] \wedge \mu_{\mathcal{B}i}(y) \} \tag{7}$$

Denote $\beta_i = \max_X [\mu_{\mathcal{A}'}(x) \wedge \mu_{\mathcal{A}i}(x)]$ the "degree of fulfilment" of the ith rule's antecedent. The output fuzzy set of the linguistic model is thus:

$$\mu_{\mathcal{B}'}(y) = \max_{1 \le i \le K} [\beta_i \wedge \mu_{\mathcal{B}i}(y)], \qquad y \in \mathcal{Y} \tag{8}$$

The entire algorithm, called the max-min or Mamdani inference, is summarized in table 2.2 and visualized in Fig. 2.14.

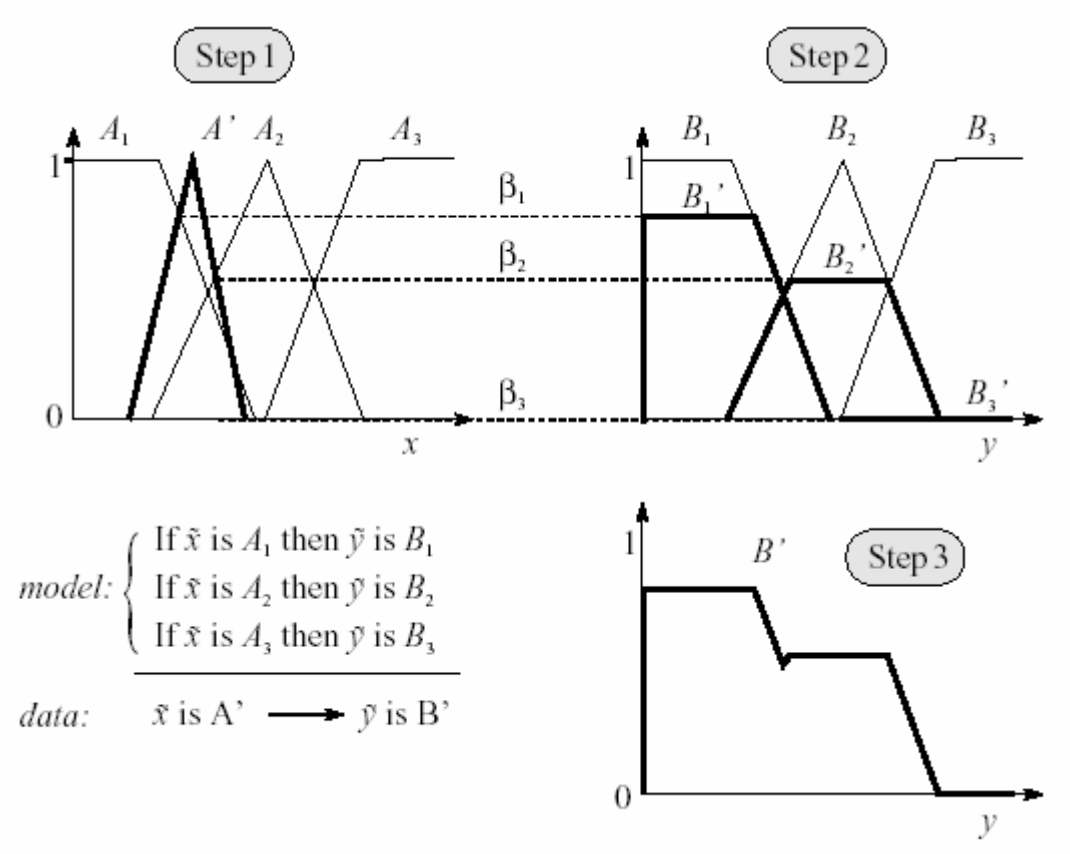


Figure 2.14: A schematic representation of the Mamdani inference algorithm.

Table 2.2: The algorithm for max-min inference

1. Compute the degree of fulfilment by: $\beta_i = \max_X [\mu_{A'}(x) \wedge \mu_{Ai}(x)]$, $1 \leq i \leq K$.
2. Derive the output fuzzy sets B_i': $\mu_{Bi}'(y) = \beta_i \wedge \mu_{Bi}(y)$, $y \in Y$, $1 \leq i \leq K$
3. Aggregate the output fuzzy sets B_i': $\mu_{B'}(y) = \max_{1 \leq i \leq K} \mu_{Bi}'(y)$, $y \in Y$

2.11.4. Multivariable systems

So far, the linguistic model was presented in a general manner covering both the SISO and MIMO cases. In the MIMO case, all fuzzy sets in the model are defined on vector domains by multivariate membership functions. It is, however, usually, more convenient to write the antecedent and consequent propositions as logical combinations of fuzzy propositions with univariate membership functions. Fuzzy logic operators, such as the conjunction, disjunction and negation (complement), can be used to combine the propositions. Furthermore, a MIMO model can be written as a set of MISO models. Therefore, for the ease of notation, the rules for MISO systems will be written here. Most common is the *conjunctive form* of the antecedent, which is given by:

R_i: **If** x_1 **is** A_{i1} **and** x_2 **is** A_{i2} **and … and** x_p **is** A_{ip} **then** y **is** B_i; i = 1; 2, … ,K. (9)

The above model is a special case of (1), as the fuzzy set $\mathcal{A}_i$ in (1) is obtained as the Cartesian product of fuzzy sets $\mathcal{A}_{ij}$: $\mathcal{A}i = \mathcal{A}_{i1}$ x $\mathcal{A}_{i2}$ x … x $\mathcal{A}_{ip}$. Hence, the degree of fulfilment (step 1 of the algorithm) is given by:

$$\beta_i = \mu_{\mathcal{A}i1}(x_1) \wedge \mu_{\mathcal{A}i2}(x_2) \wedge \ldots \wedge \mu_{\mathcal{A}ip}(x_p), \qquad 1 \leq \mathrm{i} \leq \mathrm{K} \tag{10}$$

Other conjunction operators, such as the product, can be used. A set of rules in the conjunctive antecedent form divides the input domain into a lattice of fuzzy hyperboxes, parallel with the axes. Each of the hyperboxes is a Cartesian product-space intersection of the corresponding univariate fuzzy sets. This is shown in Fig. 2.15. The number of rules in the conjunctive form, needed to cover the entire domain, is given by:

$$\boldsymbol{K} = \prod_{\mathrm{i}=1}^{\boldsymbol{p}} \boldsymbol{N}_{\mathrm{i}}, \tag{11}$$

where $\boldsymbol{p}$ is the dimension of the input space and $\boldsymbol{N}_{\mathrm{i}}$ is the number of linguistic terms of the $\boldsymbol{i}$th antecedent variable.

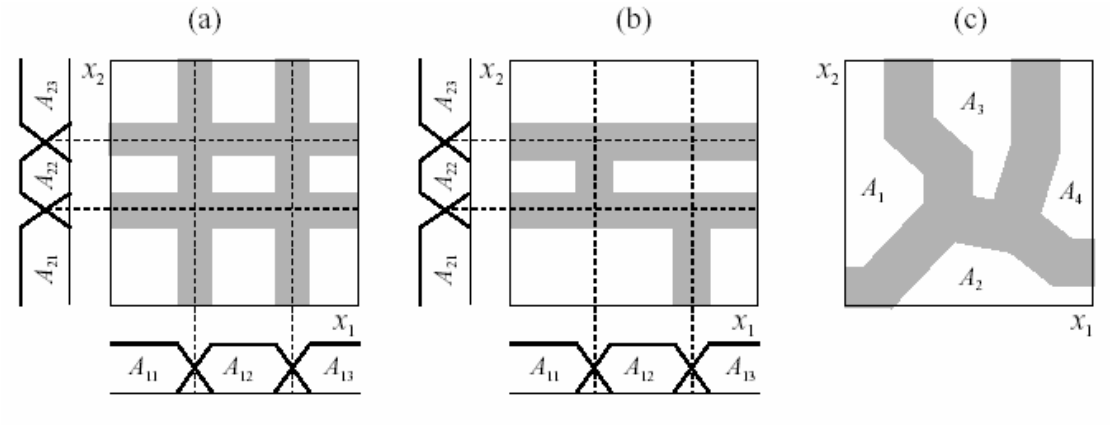


Figure 2.15: Different partitions of the antecedent space. Gray areas denote the overlapping regions of the fuzzy sets.

By combining conjunctions, disjunctions and negations, various partitions of the antecedent space can be obtained, the boundaries are; however, restricted to the rectangular grid defined by the fuzzy sets of the individual variables, see Fig. 2.15. As an example consider the rule antecedent covering the lower left corner of the antecedent space in this figure:

If x_1 is not $\mathcal{A}_{13}$ and x_2 is $\mathcal{A}_{21}$ then …

The degree of fulfilment of this rule is computed using the complement and intersection operators:

$$\beta = [1 - \mu_{\mathcal{A}13}(x_1)] \wedge \mu_{\mathcal{A}21}(x_2) \tag{12}$$

The antecedent form with multivariate membership functions (1) is the most general one, as there is no restriction on the shape of the fuzzy regions. The boundaries between these regions can be arbitrarily curved and opaque to the axes, as depicted in Fig. 2.15. Also the number of fuzzy sets needed to cover the antecedent space may be much smaller than in the previous cases. Hence, for complex multivariable systems, this partition may provide the most effective representation. Note that the fuzzy sets $\mathcal{A}_1$ to $\mathcal{A}_4$ in Fig. 6c still can be projected onto x_1 and x_2 to obtain an approximate linguistic interpretation of the regions described.

Another way to reducing the complexity of multivariable fuzzy systems is the decomposition into subsystems with fewer inputs per rule base. The subsystems can be inter-connected in a flat or hierarchical (multi-layer) structure. In such a case, an output of one rule base becomes an input to another rule base, as depicted in Fig. 2.16. This cascade connection will lead to the reduction of the total number of rules. As an example, suppose five linguistic terms for each input. Using the conjunctive

form, each of the two sub-rule bases will have $5^2 = 25$ rules. This is a significant saving compared to a single rule base with three inputs which would have $5^3 = 125$ rules.

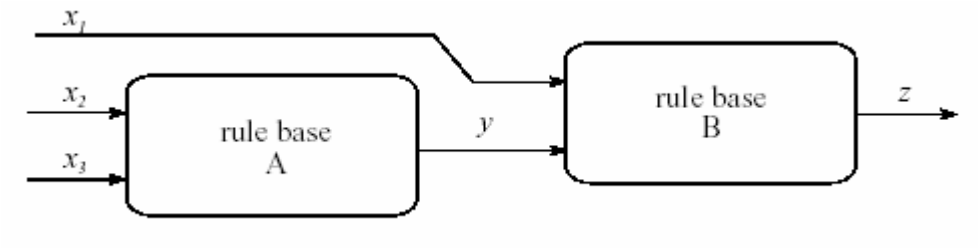


Figure 2.16: Cascade connection of two rule bases

2.11.5. Defuzzification

In many applications, a crisp output **y** is desired. To obtain a crisp value, the output fuzzy set must be defuzzified. With the Mamdani inference scheme, the center of gravity (COG) defuzzification method is used. This method computes the y coordinate of the center of gravity of the area under the fuzzy set $\mathcal{B}'$:

$$y' = \mathrm{cog}(\mathcal{B}') \quad = \quad \frac{\sum_{j=1}^{\mathcal{F}} \mu_{\mathcal{B}'}(y_j)\, y_j}{\sum_{j=1}^{\mathcal{F}} \mu_{\mathcal{B}'}(y_j)} \tag{13}$$

where $\mathcal{F}$ is the number of elements y_j in $\mathcal{Y}$. Continuous domain $\mathcal{Y}$ thus must be discretized to be able to compute the center of gravity.

2.11.6. Singleton model

A special case of the linguistic fuzzy model is obtained when the consequent fuzzy sets $\mathcal{B}_i$ are singleton fuzzy sets. These sets can be represented simply as real numbers b_i, yielding the following rules:

$$\mathcal{R}_i\text{: If } \bar{\mathbf{x}} \text{ is } \mathcal{A}_i \text{ then } y = b_i,\ i = 1; 2, \ldots, \mathcal{K}, \tag{14}$$

This model is called the singleton model. A simplified inference/defuzzification method is usually used with this model:

$$y = \frac{\sum_{i=1}^{K} \beta_i b_i}{\sum_{i=1}^{K} \beta_i} \tag{15}$$

This defuzzification method is called the fuzzy mean. The singleton fuzzy model belongs to a general class of general function approximators, called the basis functions expansion

2.12. Building Fuzzy Models

Two common sources of information for building fuzzy models are the prior knowledge and data (process measurements). The prior knowledge can be of a rather approximate nature (qualitative knowledge, heuristics), which usually originates from “experts”, i.e., process designers, operators, etc. In this sense, fuzzy models can be regarded as simple fuzzy expert systems.

For many processes, data are available as records of the process operation or special identification experiments can be designed to obtain the relevant data. Building fuzzy models from data involves methods based on fuzzy logic and approximate reasoning, but also ideas originating from the field of neural networks, data analysis and conventional systems identification. The acquisition or tuning of fuzzy models by means of data is usually termed fuzzy identification.

Two main approaches to the integration of knowledge and data in a fuzzy model can be distinguished:

1) The expert knowledge expressed in a verbal form is translated into a collection of if– then rules. In this way, a certain model structure is created. Parameters in this structure (membership functions, consequent singletons or parameters) can be fine-tuned using input output data. The particular tuning algorithms exploit the fact that at the computational level, a fuzzy model can be seen as a layered structure (network), similar to artificial neural networks, to which standard learning algorithms can be applied. This approach is usually termed neuro-fuzzy modelling

2) No prior knowledge about the system under study is initially used to formulate the rules, and a fuzzy model is constructed from data. It is expected that the extracted rules and membership functions can provide an a posteriori interpretation of the system's behavior. An expert can confront this information with his own knowledge, can modify the rules, or supply new ones, and can design additional experiments in order to obtain more informative data.

These techniques can be combined, depending on the particular application.

2.12.1. Knowledge-based design

To design a (linguistic) fuzzy model based on available expert knowledge, the following steps can be followed:

1). Select the input and output variables, the structure of the rules and the inference and defuzzification methods.

2). Decide on the number of linguistic terms for each variable and define the corresponding membership functions.

3). Formulate the available knowledge in terms of fuzzy if-then rules.

4). Validate the model (for instance by using data). If the model does not meet the expected performance, iterate on the above design steps.

It should be noted that the success of this method heavily depends on the problem at hand, and the extent and quality of the available knowledge. For some problems, the knowledge-based design may lead fast to useful models, while for others it may be a very time-consuming and inefficient procedure (especially manual fine-tuning of the model parameters). Therefore, it is useful to combine the knowledge-based design with a data-driven tuning of the model parameters.

CHAPTER 3

THE STUDENT ASSESSMENT AND COACHING SYSTEM

3.1. Introduction

In this chapter, the student assessment and coaching system, which is the main part of the remote experimentation framework that is proposed in this thesis work is explained in detail.

In the beginning sections of this chapter the use of neural networks for user modelling and rationale for using a neural network approach are mentioned in detail. This is because the ideas that are presented in those sections form the basis for using a neural network approach for the student modelling part of the student assessment and coaching system and the rationale for using a neural network section justifies the methodology followed in this work. The following section states the integration of fuzzy logic and neural networks, which lays the foundation for using a fuzzy rule base to make decisions about student performance levels to evaluate and coach the students. The integration of a fuzzy rule based decision process with an artificial neural network constitutes a form of a so-called fuzzy-neuro system based approach to real-world problems such as student modelling

The treatment of the student assessment and coaching system is done by explaining the details of the system in separate sections dedicated to each module

that forms the whole system. These modules begin with the error quantization module, where the input to the system is taken from the underlying robotic architecture and quantized to form an input to the next module in the system which is the artificial neural network classifying the error patterns. Error classification module is treated in the next section, after which the fuzzy rule-based decision process, from which the student grades and coaching messages, the outputs of the whole system, are output, is also treated in a separate section.

3.2. Rationale for Using Neural Networks in User Modelling

3.2.1. User Modelling

User modelling is a process of recognising a user's pattern (e.g. a user's behaviour pattern, knowledge pattern, cognitive pattern, etc.), based on the context of interaction. For artificial neural networks, the native function is pattern recognition (pattern matching). The pattern recognition task requires an abstraction of features, classification, fault tolerance, graceful degradation and signal enhancement. Neural networks have all these features and therefore are suited for implementing the learning of user models. Neural networks can be trained in such a way that the net can generalise over many similar examples. This generalised knowledge can then be used to recognise unknown sequences.

There are various concerns when trying to apply a neural network to the user modelling task. These range from the applicability of the problem, collecting data, selecting a learning algorithm and the architecture of the network. However, one of the most difficult problems to solve at the beginning is coming up with the correct knowledge representation. Whenever human behaviour is modelled, the first

decision to be made must concern the representation of the behaviour. Because of the black box characteristics of neural networks, the knowledge represented by the network is not much help. The weights learned by neural networks are often difficult for humans to interpret. Many knowledge representation schemes have evolved over the years to meet this need. This includes decision trees, semantic nets, frames, predicate logic, etc. Among these, the representation in fuzzy *if* . . *then* rules has been proved to be a very useful technique for various applications but has not developed yet for user modelling. The proposed work in this thesis involves combining two approaches, which are artificial neural network, and fuzzy rule based decision to represent the user (student) behaviour models. The resultant architecture is some form of a neuro-fuzzy architecture, which will be explained in detail later in this chapter.

Another important requirement for the analysis of data by neural nets is the requirement for training data. Network training algorithms typically require longer training times than, say, decision tree learning algorithms. Training times can range from a few seconds to many hours, depending on factors such as the number of weights in the network, the number of training examples considered, and the setting of various learning algorithm parameters.

Another data collection consideration is the amount and nature of the task and interaction collected. One advantage of the neural network approach is the net's ability to generalise over a data set. If data is presented in suitable amounts, the system can be trained to generalise over a set of similar but inexact examples of a phenomenon. Various techniques provide the ability to validate this generalisation, and can be used to provide a measure of confidence in the network performance.

Generalisation is important if the phenomenon to be identified has variability or is corrupted by noise. For instance, if it is possible to identify user interactions that are typical of an erroneous response, then the neural net could be trained on this example. It should be able to form a distributed representation of the correlation between the stereotypical examples and the desired system response. This correlation may in fact be highly nonlinear–another advantage of the neural net approach is that this does not pose a problem for the system. Neural nets are able to learn and model nonlinear phenomena.

The descriptive power of data is also an important aspect of using neural networks, because the results rely heavily on how the data is prepared for the network. The data must be described and categorised before the network can even be used. Here the subjectivity of the observer comes into play. Some networks do not use subjectivity in their analysis; for instance, the Kohonen network is free of the subjectivity of the developer. Here, the subjectivity may enter into the interpretation of the output.

All of these applications will require much experimentation with neural net parameters to fine tune the net's performance, but the basic ability of the net to handle a user model relies mostly on the availability of appropriate examples for training, and an appropriate network topology for handling the temporal nature of the information. One drawback of neural nets is its inflexibility in terms of adapting to new concepts. Introducing a new node into the network causes structural change to the network, thus the network needs to be retrained. User patterns may change during the course of using a computer application quite often. The continuous

adjustment and recording of changing user patterns must be considered in the design and implementation of the user model.

User modelling has a variety of practical applications. The successful modelling of human task performance can lead to improvements in computer aided instruction, and also in adaptive interfaces, as it would be possible to determine the interaction of the user and effectiveness of the problem solving strategies executed. In this manner, the integration of neural networks into the system architecture could establish an efficient interface for solving user interaction problems, as well as adapting to the user's individual preferences.

3.2.2. Rationale for Using a Neural Network

As stated in the previous section, the neural network approach is most suitable for modelling user behaviours from a pattern matching point of view because of its abilities of generalization over the training data set to deal with the fuzzy nature of the user behaviour data. A rule-based system only on its own would require every combination of possible user behaviour data should be explicitly encoded within. Therefore employing a neural network is a feasible solution to the problem of modelling students while doing an online experimentation by using previously defined behaviour stereotypes.

Among the other reasons for employing a neural network based approach, a neural network approach is desirable if the application data is intensive and depends on multiple interacting criteria, the problem area is rich in historical data or examples and the data set is incomplete and/or noisy and/or contains errors. For the case of

student action patterns during an online experimentation involving telerobotic manipulation, the above claims can be justified.

The application data is rich in examples because there are some action pattern examples that represent stereotype student behaviours, such as making erroneous manipulation during the last part or in the middle of a particular experiment task, which applies to the case of robot-supported experimentation. The data set involved with the student behaviour modelling for a robot-supported experimentation can also be said to be noisy because the student action data in most cases don't show regular patterns. Instead, as in the case of most application data sets involving human interaction, they can be irregular and unpredictable.

A neural network system is fast to execute and is the only practical choice for situations where the input data may be imprecise or not fully understood.

A final reason for employing a neural network is the relative speed of development, compared to a rule-based system. Research has indicated that the development of a neural network system is several times faster than a rule-based system [45]. This is because although the neural network acts as a rule-base, its rules are not explicitly encoded within it, but instead learned from a set of examples from the chosen domain. It is argued that as long as sufficient examples exist a neural network can learn its rules much more quickly than a human designer can formulate them.

3.3. Integration of fuzzy logic and neural networks

Hybrid systems combining fuzzy logic, neural networks, genetic algorithms, and expert systems are proving their effectiveness in a wide variety of real-world problems.

Every intelligent technique has particular computational properties (e.g. ability to learn, explanation of decisions) that make them suited for particular problems and not for others. For example, while neural networks are good at recognizing patterns, they are not good at explaining how they reach their decisions. Fuzzy logic systems, which can reason with imprecise information, are good at explaining their decisions but they cannot automatically acquire the rules they use to make those decisions. These limitations have been a central driving force behind the creation of intelligent hybrid systems where two or more techniques are combined in a manner that overcomes the limitations of individual techniques. Hybrid systems are also important when considering the varied nature of application domains. Many complex domains have many different component problems, each of which may require different types of processing. If there is a complex application that has two distinct sub-problems, say a signal processing task and a serial reasoning task, then a neural network and an expert system respectively can be used for solving these separate tasks. This kind of approach is also useful for the case of student assessment system, where the neural network is the best tool to model the student behaviours and the fuzzy rule approach is suited for providing meaningful decisions based on the networks output. The use of intelligent hybrid systems is growing rapidly with successful applications in many areas including process control, engineering design, financial trading, credit evaluation, medical diagnosis, and cognitive simulation.

While fuzzy logic provides an inference mechanism under cognitive uncertainty, computational neural networks other exciting advantages, such as learning, adaptation, fault-tolerance, parallelism and generalization.

To enable a system to deal with cognitive uncertainties in a manner more like humans, one may incorporate the concept of fuzzy logic into the neural networks. The computational process envisioned for fuzzy neural systems is as follows. It starts with the development of a "fuzzy neuron" based on the understanding of biological neuronal morphologies, followed by learning mechanisms. This leads to the following three steps in a fuzzy neural computational process.

- Development of fuzzy neural models motivated by biological neurons.
- Models of synaptic connections, which incorporates fuzziness into neural network.
- Development of learning algorithms (that is the method of adjusting the synaptic weights).

Two possible models of fuzzy neural systems are

- In response to linguistic statements, the fuzzy interface block provides an input vector to a multi-layer neural network. The neural network can be adapted (trained) to yield desired command outputs or decisions.

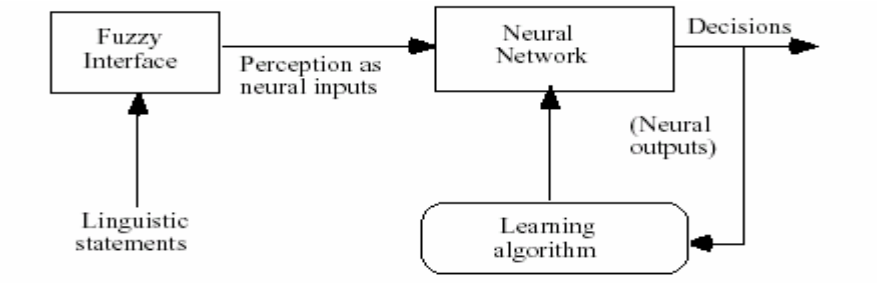


Figure 3.1: The first model of fuzzy neural system

o A multi-layered neural network drives the fuzzy inference mechanism.

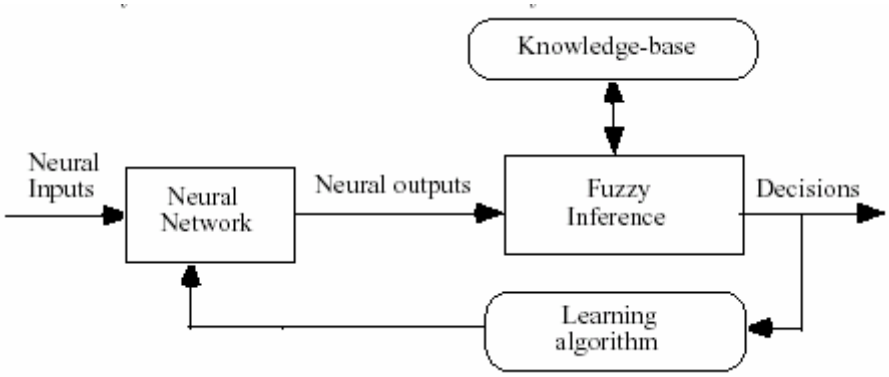


Figure 3.2: The second model of fuzzy neural system

The approach taken in the proposed work of this thesis is a variation of the second model for the fuzzy neural systems. The output of the neural network serves as an input to the fuzzy inference. The difference is that the decision output from the fuzzy inference does not affect the neural networks training. The neural network is trained a priory with previously determined input-output vector pairs.

Generally, neural networks are used to tune membership functions of fuzzy systems that are employed as decision-making systems for controlling equipment. Although fuzzy logic can encode expert knowledge directly using rules with linguistic labels, it usually takes a lot of time to design and tune the membership functions, which quantitatively define these linguistic labels. Neural network learning techniques can automate this process and substantially reduce development time and cost while improving performance.

In theory, neural networks, and fuzzy systems are equivalent in that they are convertible, yet in practice each has its own advantages and disadvantages. For neural networks, the knowledge is automatically acquired by the back propagation

algorithm, but the learning process is relatively slow and analysis of the trained network is difficult (black box). Neither is it possible to extract structural knowledge (rules) from the trained neural network, nor can we integrate special information about the problem into the neural network in order to simplify the learning procedure.

Fuzzy systems are more favourable in that their behaviour can be explained based on fuzzy rules and thus their performance can be adjusted by tuning the rules. But since, in general, knowledge acquisition is difficult and also the universe of discourse of each input variable needs to be divided into several intervals, applications of fuzzy systems are restricted to the fields where expert knowledge is available and the number of input variables is small.

To overcome the problem of knowledge acquisition, neural networks are extended to automatically extract fuzzy rules from numerical data.

Cooperative approaches use neural networks to optimize certain parameters of an ordinary fuzzy system, or to pre-process data and extract fuzzy (control) rules from data.

Based upon the computational process involved in a fuzzy-neuro system, one may broadly classify the fuzzy neural structure as feedforward (static) and feedback (dynamic). The structure used in the proposed work for this thesis can be said to be feed-forward

The concept of neuro-fuzzy computing will not be explained further here because the approach taken in the proposed work for this thesis is not a strictly neuro-fuzzy approach, where fuzzy neurons having crisp inputs but fuzzy outputs calculated by t-norm or t-conorm, or some other continuous operation, resulting in a

form of hybrid neural network. Back-propagation algorithm can not directly be used for hybrid neural networks though they can be trained with steepest descent methods. As stated before in this section, the approach used in this work is employing a trained neural network and a static fuzzy rule base separately in a way that is close to the second model for a fuzzy-neural system mentioned earlier in this section.

3.4. The Student Assessment and Coaching System

We introduce the general architecture of the student assessment and coaching system in this section. The modules of the system are briefly considered here to form a preview to the following sections, where they will be analyzed in details. These modules are the error quantization, error classification by artificial neural network, and the fuzzy rule-based decision process.

The student assessment and coaching system is the major component of the remote experimentation framework proposed in this thesis. The other components of the framework are the user interface and the underlying robotic hardware that can be controlled through the Internet that has been developed in another thesis work [46], which will be explained in chapter 4. The system is designed to be integrated to the tele-robot control software and the user interface software. It is basically a software component that is to be called from the general program which encapsulates the robot control software and the Internet connection module. [46] The user assessment and coaching system that is explained in this chapter is specially tailored to the needs of the underlying hardware, the general software framework and the proposed experimentation scheme for the tele-robotic hardware. According to the needs of the different applications, experimentation schemes, and underlying hardware structures,

the necessary modifications can be made easy, that is why a modular approach for developing the system is followed. Necessary modules can be modified without breaking down the general structure of the system.

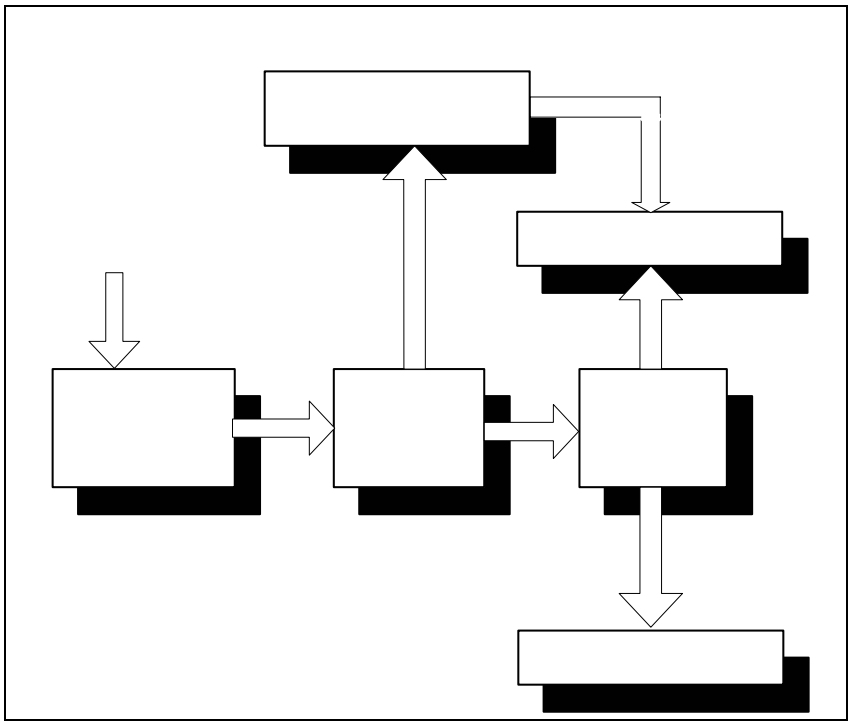


Figure 3.3: The block diagram of the assessment and coaching system

The block diagram of the proposed system is given in figure 3.3. The system is composed of an error quantization module, a classifier of error types based on an artificial neural network and a decision process. There is also an authentication subsystem for the security of the application, which denies access to unidentified user with bizarre error patterns. This module, which will be explained in detail in chapter 4, coupled with the user record part also creates personalized user records with personalized evolution of error patterns mainly for off-line storage and for the next possible experiments that can be done by a particular user. The whole system

takes errors done during the task and outputs informative messages sent to the experiment interface for the student such as new task assignments or repeats, as well as the user grade values to be added to the user record. The inputs to the system come from the underlying tele-robotic hardware as the x and y coordinate position values of the pen that the robot holds. The outputs are sent to the general program that is responsible for the user interface and the module that keeps the database connection.

3.4.1. Error Quantization Module

The purpose of the error quantization module is to represent the user behavior in a way that is suitable for input to the neural network. The student behavior is represented as vectors having “+1” or “-1” values as elements. In fact, this module is like an observation module, which monitors user actions and records them in vector form for enabling the system model the user behavior using this information.

For each experiment and its levels, the correct sequence of actions and success limits are determined a priori by the laboratory experimentation manager. The laboratory experiment manager is a part of the error quantization module and it is tailored according to the experimentation scheme that is in use. In the current application involving letter drawing experiments with the robot that is connected to the Internet, for each experiment levels the correct sequence of drawing the shapes and their correct positions on the paper is predetermined. The robot control software sends the position of the drawing pen as x and y coordinates to the error quantization module. Depending on the experiment step the coordinate values and the action sequence is compared to the correct ones coded in the experiment manager.

The error quantization module on the other hand, outputs automatically either of the two values “–1” or “1” as a measure of performance for each user action. The “1” value means the coordinate values are in between the allowed error limits and the sequence of the action is correct and therefore the action is right. The correct sequence of the action is set to be the path that the students are required to follow. In this context, it means a pre-specified direction that the students must draw the shapes. As an example form the letter drawing experiment, the student must first draw the left line, then the right line and lastly the middle line of the letter “A”. The “-1” value means that either the coordinate values are outside of the allowed error limits or the sequence of the action is wrong. The sequence of the action can also be wrong if the student repeats the step that has already been done. Looking at the same example, while drawing any line for the letter “A”, the student is not allowed to go back and forth with the pen; the direction that the pen follows should be the same during drawing the whole line. Generally speaking, “1” value means that the user action is inside the success limits in the correct context of the experiment and therefore the action is right. “-1” value means that the action is outside of the limits of success in the related context, and therefore the action is wrong. This point of view proves to be useful while extending the user assessment and coaching system for other experimentation schemes.

“-1” and “1” values are collected in the form of an input vector for the classifier until a specified number of actions are completed. This specified number of actions determines an evaluation interval, which sets the dimension of the input vector to the classifier. An experiment consists of many evaluation intervals. For the implementation given in this proposed work, the evaluation interval takes a value

among three preset values; 8, 12, and 16. This means that the student will be evaluated by the system after each 8, 12 or 16 actions. Dividing the experiment into evaluation intervals is a step towards better guidance, by this way the errors that the student makes during the experiment are more closely watched and necessary measures can be taken for the student to correct his/her errors. As previously mentioned, these measures are in the form of messages that informs the student of the performance during the interval and interventions to force the student t repeat the failed interval. This methodology is also useful for easing the computational load for the computer that runs the program, because with the experiment divided into smaller intervals, the neural network size will be smaller, so there will be no interruptions arising from the computation of large data sizes during the real time performance.

The evaluation interval value varies for a specific user according to his/her performance while carrying out a certain experiment. It can either increase or decrease in the course of the experiment. By default, the value is taken initially as 8. During the experiment, this value increases one step (to 12 for example) in the case of three successive successful intervals with zero experiment repeat value in the user record or two successive fully successful intervals with a repeat value smaller than two. The evaluation interval value is decreased one step in the case of successive failures or full failures. (The notions of full-success, success, full-failure and failure are the outputs of the decision process, which will be presented in the following section on the decision process).

Using different values of evaluation interval comes from the need of adaptation of the evaluation system to the variations in student skill levels. The

evaluation interval value defines somewhat a confidence given to the user. In the case of a large interval, the performance is evaluated more rarely and evaluation results are more in the favor of the user: the user may make minor errors but will still be considered as successful. The system builds this confidence based on the student repeated successes in his/her several tasks (actions). Changing of the evaluation interval value during the course of the experiment is done by the system without informing the user of this change. This further supports the adaptive nature of the student assessment and coaching. By this way, the evaluation of the experiment and coaching a particular student is adjusted to the level of his or her capability and knowledge.

3.4.2. Error Classification Using ANN

After each completed evaluation period, the resulting bipolar vector is input to an ANN based classifier that typifies the error made. The error classification module includes three multi-layer neural networks corresponding to each evaluation value. The correct neural network is chosen according to the evaluation interval value that is currently being used. In the software implementation, the error quantization module sends the evaluation interval value along with the error vector to the error classification module as parameters, so the correct network is chosen for the interval.

Each neural network is a feed-forward network having one hidden layer. The number of input nodes of the network is determined by the evaluation interval value and has the value of 8, 12 or 16. The numbers of hidden nodes are 3, 5, and 7 respectively for the three neural networks.

These values are determined experimentally also taking into account the sensitivity analysis of the outputs of the system to changes in number of hidden nodes. The number of the output nodes is 3 in all cases typifying the student performance. Figure 3.4 shows the structure of the feed forward neural network where the number of input nodes is 8. The activation functions of the neurons are linear with saturating at -1 and 1.

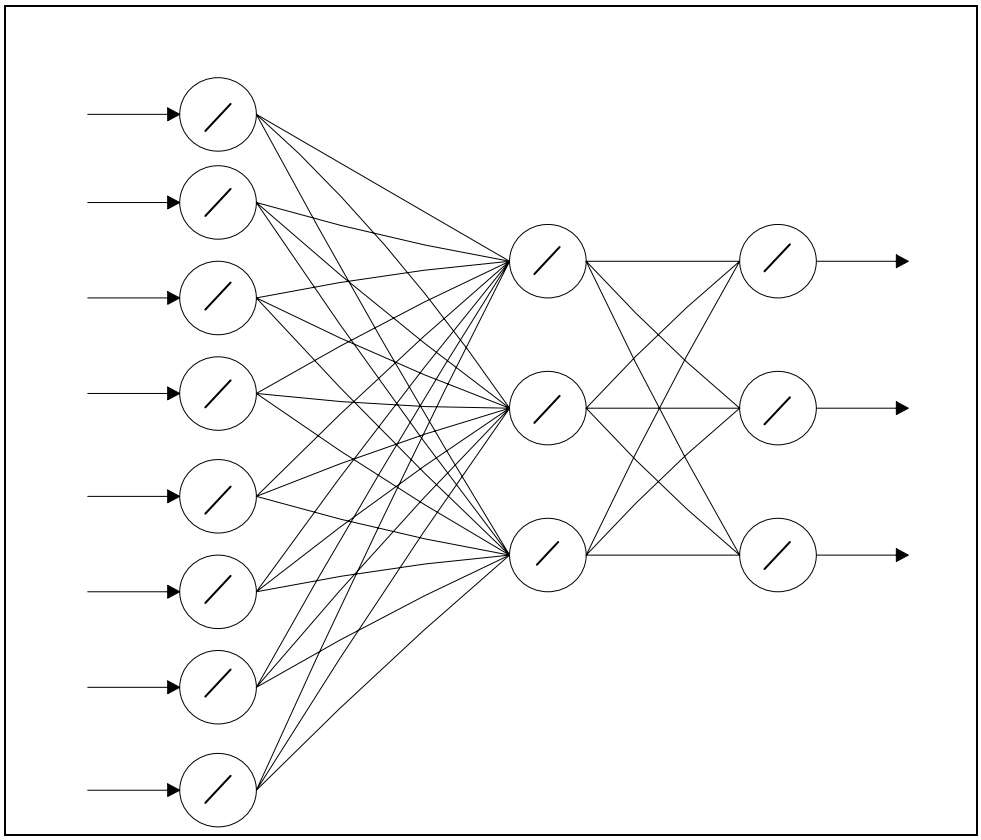


Figure 3.4: The structure of the feed forward neural network for the 12 input neurons case

Each neuron in the output layer outputs a real value in [-1 1] interval. The number of output nodes is chosen to be 3 for the reason to provide enough output space size to represent 8 main different user characteristic classes that the system classifies the student behaviors in. These classes are treated in more detail in the following paragraphs.

A particular output vector of the neural network does not itself give precise information about the student behavior type. The fuzzy rule based decision process outputs the category that the student belongs to, taking the output vector of size 3 from the neural network as an input. Therefore, the neural network can be thought of as reducing the size of the student actions space, which is determined by the evaluation interval value. By this way, it is much more convenient to build a reasonably sized rule base to correctly classify the student behaviors after the neural network.

The neural networks are trained with the back propagation algorithm. Linear activation function, which the output activity is proportional to the total weighted input, is used for the output of each neuron in the respective neural network. Each neural network is trained by default by 8 input vectors representing different typical user behaviors for the evaluation intervals, 8, 12 and 16. The training vectors are chosen to represent behavior stereotypes on which the evaluation system can generalize its inputs with the aid of neural network.

Table 3.1: Input-output training pairs for evaluation interval value is 12

Input vector	Output vector
[1 1 1 1 1 1 1 1 1 1 1 1]	[1 1 1]
[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]	[-1 -1 -1]
[1 1 1 1 -1 -1 -1 -1 1 1 1 1]	[1 -1 1]
[-1 -1 -1 -1 1 1 1 1 -1 -1 -1 -1]	[-1 1 -1]
[1 1 1 1 1 1 1 1 -1 -1 -1 -1]	[1 1 -1]
[-1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1]	[-1 -1 1]
[1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1]	[1 -1 -1]
[-1 -1 -1 -1 1 1 1 1 1 1 1 1]	[-1 1 1]

Training a particular neural network involves presenting the input vectors to the network repeatedly while updating the connection weights until the output mean square error gets below a certain value that is determined experimentally while evaluating the system performance, which is explained in more detail in chapter 5. The details on back propagation algorithm and the training of the neural networks can be found in chapter 2 of this thesis. In table 3.1, the input and output training vectors are shown in the case where evaluation interval value is 12.

Each element of the output vector of a particular neural network during actual runs is valued between -1 and 1. The output vector of this error classifier module more specifically represents a model of the user performance within the respective evaluation interval in terms of the error evolution as: small error, thus good performance; medium error, thus mediocre performance with need to exercise more; large error, thus critically bad performance. As stated in the above paragraph, the

training vectors represent 8 classes of user behaviors. These 8 main user characteristics are:

- No errors in all steps, which represented by the input vector in the first row of table 3.1, where the vector is all “1”.
- Small number of errors in the first steps, which represented by the input vector in the last row of table 3.1, where the first four elements of the vector are valued as “-1”.
- Small number of errors in the middle steps or distributed unevenly, which is represented by the input vector in the third row of table 3.1, where four “-1” valued elements are placed in the middle.
- Small number of errors in the last steps, which is represented by the input vector in the fifth row of table 3.1, where the last four elements of the vector are valued as “-1”.
- Errors in all steps, which is represented by the vector in the second row of table 3.1, where all elements of the vector are valued as “-1”.
- Large number of errors in the first steps, which is represented by the vector in the sixth row of table 3.1, where only four “1” valued elements are at the end of the vector.
- Large number of errors during the course of experiment, distributed unevenly, which is represented by the vector in the fourth row of table 3.1, where only four “1” valued elements are in the middle of the vector.
- Large number of errors in the last steps, which is represented by the vector in the seventh row of table 3.1, where four “1” valued elements are placed in the beginning of the vector.

Table 3.2 shows the input-output training vector pairs for the case where evaluation interval value is 16. The same 8 main user characteristics mentioned above apply here as well.

Table 3.2: Input-output training pairs for evaluation interval value is 16

Input vector	Output vector
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1]	[1 1 1]
[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]	[-1 -1 -1]
[1 1 1 1 1 -1 -1 -1 -1 -1 -1 1 1 1 1 1]	[1 -1 1]
[-1 -1 -1 -1 -1 1 1 1 1 1 1 -1 -1 -1 -1 -1]	[-1 1 -1]
[1 1 1 1 1 1 1 1 1 1 1 1 -1 -1 -1 -1]	[1 1 -1]
[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 1 1 1 1]	[-1 -1 1]
[1 1 1 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1]	[1 -1 -1]
[-1 -1 -1 -1 1 1 1 1 1 1 1 1 1 1 1 1]	[-1 1 1]

3.4.3. Decision Process

The decision process is a rule-based system, which uses fuzzy control principles to make decisions about the performance of the user according to the respective interval. In accordance with the principles stated in the previous sections of this chapter, a fuzzy rule based system is constructed to work with the neural network for student assessment and coaching. Decision process takes its input from the error classification module as a real valued vector of size 3, namely the output vector of the artificial neural network.

Upon receiving, each of the three outputs of the neural network is fuzzified into 4 sets. These sets are labeled as LOW, MED1, MED2 and HIGH and triangular membership functions are used as in fig. 3.5. They cover up the interval between –1 and 1 which are the outmost margins of each network output. This interval also constitutes the universe of discourse for the four fuzzy sets that are stated here.

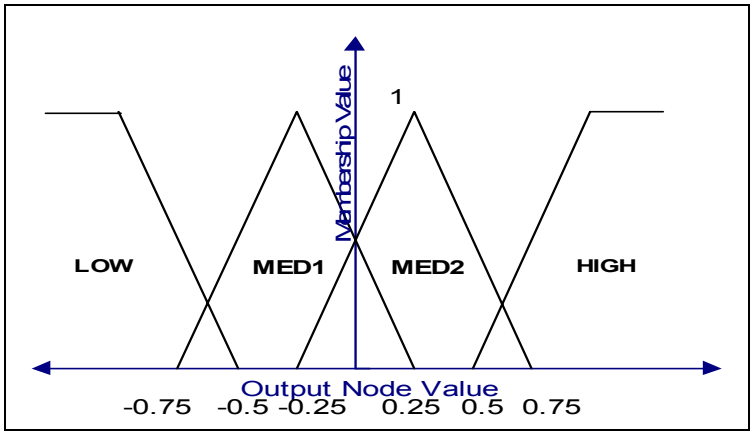


Figure 3.5: Fuzzification functions

In this respect, the set of antecedent linguistic terms is: $\mathcal{A}$ = {***low***, ***med1***, ***med2***, ***high***}. The antecedent linguistic terms are named according to their positions in the interval [-1, 1]. Denoting the neural networks outputs, which are the fuzzy model's inputs (and also the linguistic variables), as x1, x2, and x3 respectively, table 3.3 shows the membership functions, $\mu(x)$ for each antecedent linguistic variable for a particular output of the neural network as an example.

Table 3.3: An example for the calculation of membership values of inputs

Linguistic term	Input variables		
	X1 = 0.83	X2 = 0.2	X3 = -0.5
Low	0	0	0.75
Med1	0	0.775	0.975
Med2	0	0.975	0
High	0.91	0	0

The neural network, in this context, can also be seen as a discretizer for the input domain of the fuzzy rule-based system. The input domain is discretized in the form of the neural network output values, as can be seen from the table 3.3, where X = {0.83, - 0.1, 0.65}.

The behavior of the decision process is characterized by linguistic description rules based on expert knowledge. This expert knowledge depends mainly on experimentation with different student behavior patterns to the system. The fuzzy system considered here can be said to be multiple-input single-output (MISO) system because the rules have 3 linguistic variables in their antecedents and one linguistic variable in their conclusions. Some examples of rules from the rule base are:

$\mathcal{R}_1$: If ***x1*** is LOW and ***x2*** is LOW and ***x3*** is LOW then ***y*** is F.

$\mathcal{R}_2$: If ***x1*** is HIGH and ***x2*** is HIGH and ***x3*** is HIGH then ***y*** is S.

$\mathcal{R}_3$: If ***x1*** is LOW and ***x2*** is MED1 and ***x3*** is HIGH then ***y*** is F1.

Fuzzy rules are constructed to select among 8 levels of success, therefore there are 8 linguistic terms for the output linguistic variable, y. The 8 levels are f, f1, f2, f3, s3, s2, s1, and s respectively, and are defined as:

- F – Full failure
- F1 – Failure with errors mostly in first part of the interval.
- F2 – Failure with large number of errors distributed unevenly on the interval.
- F3 – Failure with errors mostly in the last part of the interval.
- S3 – Success with small number of errors in the first part of the interval.
- S2 – Success with small number of errors in the middle part or distributed unevenly.
- S1 – Success with small number of the errors in the last part of the interval.
- S – Full success in the interval.

The total number of rules is 64. The number of all possible rules with three input variables out of 4 fuzzy sets are however 81. This is because the probability of firing of the remaining rules is very low from the nature of the neural network output and student behavior characteristics. As can be seen from the total number of rules, more than one rule is associated with the same output.

For the implementation of rules, the logical connector "and" is modeled by "min" operator, therefore the result of each rule is found by taking the minimum of membership values of each input variable. This can be illustrated by using the previously given rules as an example:

$$\mu_{\mathcal{R}1}(x1, x2, x3, y) = \min(\mu_{low}(x1), \mu_{low}(x2), \mu_{low}(x3))$$

$$\mu_{\mathcal{R}2}(x1, x2, x3, y) = \min(\mu_{high}(x1), \mu_{high}(x2), \mu_{high}(x3))$$

$$\mu_{\mathcal{R}3}(x1, x2, x3, y) = \min(\mu_{low}(x1), \mu_{med1}(x2), \mu_{high}(x3))$$

For each output variable, the membership value is found by employing the "max" operator among the output of the rules corresponding to the same output variable.

The evaluation decision is finalized by selecting the output set which has the maximum membership value. According to this selection, a corresponding message informing the user of his/her performance is generated, a new task or repetition of the previous one is assigned and the overall grade is computed for the user at the end of the experiment by taking the arithmetic mean of the grades of each interval.

For the full success case during an interval, the highest grade, which is 100 out of 100, is assigned. In other success cases the assigned grade is lower, being the least for S2 among successes, which is 70 out 100. The assigned grade is 80 out of 100 for S1, and 90 out of 100 for S3. This kind of grade assignment comes form the methodology that, a student who is able to correct his/her mistakes during the course of the experiment should be awarded more, where a student who makes arbitrary mistakes during the course of the experiment should be graded less. The grades assigned decreases towards F, which is assigned the lowest grade of 0. The system asks the user to repeat the experiment from the beginning in case of failure that is for the cases F1, F2, F3 and F. While the student is directed to repeat the experiment, he/she is informed by the messages of where most errors have been made.

CHAPTER 4
THE ILLUSTRATIVE APPLICATION

4.1. Introduction

In this chapter, the tele-robotic application used to demonstrate our intelligent student assessment and coaching system within a remote experimentation framework is introduced. The development of the application involves integrating the remote experimentation framework to the existing tele-robotic hardware and user interface that was developed in [46], more specifically, designing and developing the user interface for the experiments, coding the student assessment and coaching system that is previously described and the development of the experimentation scenario for the robot-assisted setup.

The chapter starts with the emphasis on the issues related with the implementation of the user interface for a remote access to a robot-supported laboratory framework through the Internet, previously stated in the first chapter of this thesis. The needs for providing an intelligent interface are stated in detail in this section with giving information on forms of intelligence that a user interface can have. In addition to this, the necessary elements for an informative interface layout are discussed in this section as well.

The next section of the chapter deals with the proposed remote experimentation system with a robot-assisted setup through the Internet. In the following subsections, the hardware layout of the robot-assisted setup, the user interface for the experimentation framework and, the choice for the software development platform to code and integrate the subsystems are treated in detail.

Development of the experimentation scenario constitutes another subsection. The experimentation scenario developed with the idea of making use of the currently existing tele-robotic hardware setup to demonstrate the abilities and the performance of the remote experimentation framework. Each step of the experimentation is explained from the easiest tasks to the hardest ones.

The chapter ends with a subsection devoted to illustrative examples showing each module of the student assessment and coaching system in action.

4.2. User Interface Issues

As previously stated in chapter 1, providing an intelligent interface with a simple but informative layout is a key aspect in the implementation of remote robot-supported laboratory access through the Internet. The definition and the important properties of intelligent user interfaces will be briefly stated in the following paragraphs along with their importance in order to implement a successful user interaction. The relations to the intelligent remote experimentation framework that is proposed in this thesis are also mentioned. The second main user interface issue, namely the graphical layout of the interface is also treated at the end of this section.

4.2.1. Intelligent User Interfaces

Intelligent user interfaces, (IUIs) is a subfield of Human-Computer Interaction. The goal of intelligent user interfaces is to improve human-computer interaction by using smart and new technology. This interaction is not limited to a computer, but can also be applied to improve the interface of other computerized machines, for example the television, refrigerator, or mobile phone.

A formal definition of intelligent user interfaces can be given as:

"Intelligent user interfaces specifically aim to enhance the flexibility, usability, and power of human-computer interaction for all users. In doing so, they exploit knowledge of users, tasks, tools, and content, as well as devices for supporting interaction within differing contexts of use."[47]

A normal user interface is defined as a method of communication between a human user and a machine. If we extend this definition, it can be said that an *intelligent* user interface uses some kind of intelligent technology to achieve this human-machine communication. In other words, IUIs are interfaces with the ability to adapt to the user, communicate with the user, and solve problems for the user.

Using techniques from artificial intelligence, IUIs deal with different forms of input and output and try to help the user in an intelligent fashion. They try to solve some of the problems that the current direct-manipulation interfaces cannot, such as:

Creating personalized systems:

No two persons are the same and people have different habits, preferences, and working methods. An intelligent interface that takes these differences into account can provide a personalized method of interaction. The interface knows the user and can use that knowledge in the way it communicates with the user.

Information overflow or filtering problems:

Intelligent interfaces can reduce the information overflow associated with finding information in large databases or complex systems. By filtering out irrelevant information, the interface can reduce the cognitive load on the user. In addition, the IUI can propose new and useful information sources not known to the user.

Providing help on using new and complex programs:

Computer systems can be very complicated to work with when you first start to use them. Many computer users fail to keep up with these new functions. Intelligent help systems can detect and correct user misconceptions, explain new concepts, and provide information to simplify tasks.

Taking over tasks from the user:

An IUI can also look at what you are doing, understand and recognize your intent, and take over some of your tasks completely, allowing you to focus on other things.

Other forms of interaction:

Currently, the most common interaction devices are the keyboard and the mouse. IUI-research looks at other forms of interaction (e.g. speech or gestures). By providing multiple forms of interaction, people with a disability will be able to use computers more easily.

To summarize, instead of the user adapting to the interface, and IUI can adapt to the user. The IUI tries to determine the needs of an individual user and attempts to maximize the efficiency of the communication with the user.

The most important property of IUIs is that they are designed to improve communication between the user and machine. It does not matter much what kind of technique is used to achieve this improvement, as long as it can be regarded as intelligent. Below a list of several types of techniques that are being used today in intelligent user interfaces is given.

1) In*telligent input technology* uses innovative techniques to get input from a user. These techniques include natural language (speech recognition and dialogue systems), gesture tracking and recognition, facial expression recognition, gaze tracking and lip reading;
2) *User modeling* covers techniques that allow a system to maintain or infer knowledge about a user based on the received input;
3) *User adaptivity* includes all techniques that allow the human-machine interaction to be adapted to different users and different usage situations;
4) *Explanation generation* covers all techniques that allow a system to explain its results to a user (e.g. information visualization, or tactile feedback in a virtual reality environment).

Other important properties of IUIs are personalization and flexibility of use. To achieve personalization, IUIs often include a representation of a user. These user models log data about the user's behaviour, knowledge, and abilities. New knowledge about the user can be inferred based on the input and interaction history of the user with the system. In order to be flexible many IUIs use adaptation or learning. Adaptation can occur based on the stored knowledge in a user model or by make new inferences using current input. Learning occurs when stored knowledge is changed to reflect new encountered situations or reflect new data. Because of the

difficulties involved in creating IUIs and the amount of knowledge engineering that is needed, most IUIs focus on a specific method of interaction (e.g. speech) or on a particular narrow application domain.

An often-made mistake is to confuse an IUI with an intelligent system. A system exhibiting some form of intelligence is not necessarily an intelligent interface. There are many intelligent systems with very simple non-intelligent interfaces and the fact that a system has an intelligent interface does not say anything about the intelligence of the underlying system (see also Figure 4.1).

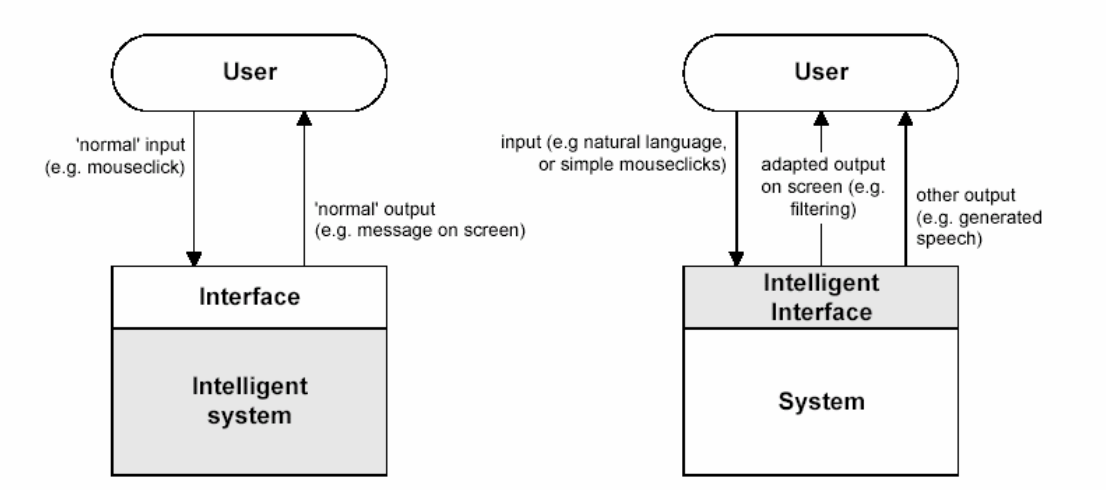


Figure 4.1: Two types of interface-system interaction

Unfortunately, the boundary between a system and its interface is not always very clear. Often the technology used in an IUI is also part of the system, or the IUI forms the entire system itself. For example, a speech recognition system can be part of an intelligent interface to a system, but it can also be the complete system. If an IUI can be regarded as a system on its own, then it is by definition an intelligent

system. However, an intelligent system does not necessarily have an intelligent interface.

In accordance with the concerns and properties above, the user interface for a remote access to a robot-supported lab should be intelligent to tackle with the diversity of student behaviours and skill levels. In addition to that, the interface should make correct decisions about the performance and provide necessary guidance to the students who will be using it to interact with the robot supported experiment hardware. The interface for the remote experimentation framework is developed to possess the intelligence needed. The user modelling and evaluation is done by employing neural network techniques and fuzzy system methodologies as stated in the previous chapters. The interface adapts the context of the experimentation and assessment according to the skill level and performance of the user, which is learnt through modelling and past performance records. User modelling and adapting to the user behavior, which are the two important properties of intelligent user interfaces, are provided by this way.

4.2.2. Graphical Considerations for User Interfaces

The graphical properties of the user interfaces have been the most important aspect for a successful interaction with the devices since the beginning of the research for user interfaces. Since, the graphical user interface is the main tool for successful human-computer interaction; it should allow the user to concentrate on the task at hand by making the interface between computer and the human seamless.

For a successful graphical user interface design, some important major considerations must be taken into account. These considerations are related to the

human psychology, further in an educational context, they are closely related to the student pedagogy. The main considerations can be summarized as:

- Keeping the interface simple enough not to cause additional user distraction. Avoiding unnecessary detail makes the interface easy to navigate and the abilities of the underlying program easy to learn.
- Grouping of information well enough for display to improve readability and to highlight relationships between the information. In this context the consistent use of colour without overusing to present different groups with different colour and the similar elements with same colour scheme, providing graphical boundaries around elements for grouping them, and highlighting the elements that are in use are the major methods for the efficient grouping of information.
- Sequencing the information clearly to help the users to find the information they need easily. The layout of the user interface becomes important, because the user wants to receive the information to be presented to him/her in an order that is psychologically accurate. This can be done in several ways, some of which are placing the important information at a prominent location, placement of frequently used elements in an easy-to-reach places, and placement of the elements in a left-to-right and top-to-left order for these are the directions which majority of the users take while scanning the interface.

In the context of the remote experimentation through the Internet, one of the most important requirements for the user interface is to depict the experiment setup as clear as possible, for the students can easily see the manipulation they are doing

and the remote experimentation will not be far from the reality. By this way, the expected “hands-on” experience and the feeling of actually carrying out the experiments instead of running simulations, which are the main objectives of remote experimentation, can be provided to the students. The other requirements, which are in consistency with the considerations that are presented in the above paragraphs, are to place the different elements in the layout correctly to aid the student to clearly understand the information given. The elements considered can be message boxes, robot control area, the area for viewing the experimentation state and buttons to start and stop the experiment. The interface for this proposed application is developed with these considerations in mind.

4.3. The Proposed Application

4.3.1. Hardware and experiment layout

The student evaluation and coaching system is applied to a Tele-robotics system in the BILTIR CAD/CAM & Robotics Center of the Middle East Technical University. An ABB IRB 2000 6-DOF industrial robot is controlled by the intelligent interface through the Internet with a remote experimentation scenario. The manipulator is a specially designed pen realized in [46], attached to the robot by a pneumatic mechanical gripper. In the experiment setup, there also exist a plain paper with colored regions and figures that is mounted on a table and a webcam that is displaying the state of the pen and the paper. The robot is placed such that the tip of the pen that the robot arm holds by the pneumatic gripper is just above the plain paper in a touching distance before the start of the experiment. When the experiment starts, the gripper (and therefore the pen) is moved to the correct place on the

sketching paper for the experiment step to be done with the movement of the robot arm. This movement is done automatically by the robot control software at the start of each experiment.

The robot control software is written in Microsoft Visual Basic 6.0 [46]. The proposed framework is integrated into the robot control software to form the general experimentation program, where the student assessment and coaching system acts as module that is called upon as a function after completion of each user action. The experiments consist of drawing various shapes and letters on a plain paper by the pen held by the robot gripper, ranging from the easier tasks of line and curve drawings to more elaborate ones as shape drawings and then to much harder ones such as letter drawings. Figure 4.2 shows from two different angles the robot gripper equipped with the pen.

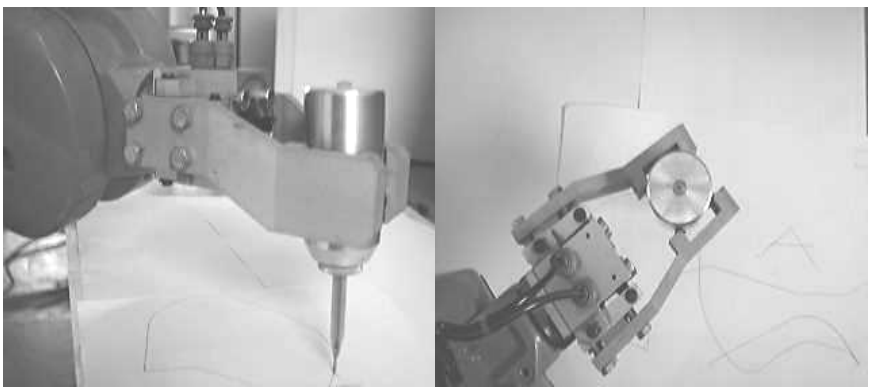


Figure 4.2: The robot gripper shown from two angles [46]

4.3.2. The User Interface

With the student assessment and coaching system integrated to the robot-control user interface, the interface acquires the properties of an intelligent interface, which are mentioned in section 4.2.1.

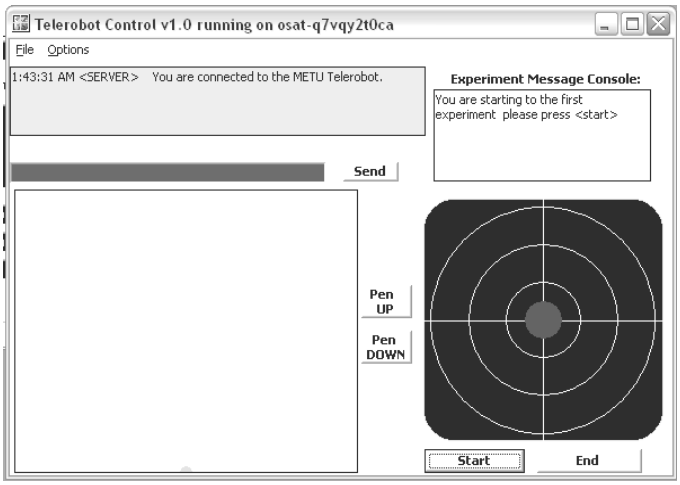


Figure 4.3: The user interface

The graphical user interface for the robot control and experimentation can be seen in figure 4.3. The interface consists of several major areas for manipulating the robot, seeing the movement of the pen and receiving the messages. The bottom right part of the interface is dedicated to manipulation of the robot. The robot is manipulated by clicking and moving the red spot on the black screen by the mouse, the main idea behind which is to simulate a joystick that may not be present in the setup that the remote experiments are carried out from. The big white area on its left represents a simulated version of the actual drawing paper that is used during the experiment, and the position of the tip of the pen is shown in this area by a small dot that is green when the pen is up and red when the pen touches the paper. The

resulting actual drawing done by manipulating the robot is also simulated in real time in this area for visual feedback of the task to the user.

The part above the robot manipulation area is dedicated to the messages that are shown to the users. The system evaluates the user performance, gives grade for each drawing experiment, assists the user by informing of his/her performance and directs the user to repeat an experiment in the case of failure or directs him/her to a new experiment of higher level of complexity. All of the information regarding the student performance and whether or not the student will repeat the experiment are typed in the message area. The messages that are shown to the user are as following:

- "You have to repeat the experiment. All of your actions were wrong during the last interval."
- "You have to repeat the experiment. Most of your actions were wrong during the last interval with errors mostly in the beginning."
- "You have to repeat the experiment. Most of your actions were wrong during the last interval with errors mostly in the last parts."
- "You have to repeat the experiment. Most of your actions were wrong during the last interval."
- "You can carry on the experiment, but you had some errors in the beginning of the last interval."
- "You can carry on the experiment, but you had some errors towards the end of the last interval."
- "You can carry on the experiment, but you had some errors in the last interval."

- "You can carry on the experiment; you didn't make any errors during the last interval."

In the context of this experimentation, for the evaluation interval number 8, at most 2 errors in an interval is considered as success, where at most 4 and 5 errors are considered as success in the cases of evaluation intervals 12 and 16 respectively.

The webcam video stream of the robot in action and the drawing sheet are not included in the general user interface for not making the user interface too big in size. Instead it is provided as a separate window that can be turned on and off according to the user preference. This was a necessary measure, because it required further development for providing the video stream through the interface by encoding the video stream for the transmission. The "Net Meeting" software by Microsoft was used for displaying the video stream, since this software comes bundled with Windows operating system; this brought no extra software installation by the student on his/her computer. The transmission of video streams is another major subject in itself, so it will not be dealt with further detail here.

The system also has an access to a database containing user information where updates to the user record fields are made during user profiling. Stored data helps the system to recall a user that has previously done experiments according to the user model based on the error patterns previously made and their type of evolution. Moreover, the experiment level to start from and the initial evaluation interval value are determined using this information on user model.

4.3.3. Choice for the Software Development Platform

The software development platform that is chosen for coding and integrating the components of the remote experimentation framework is Microsoft Visual Basic 6.0. The student assessment and coaching system is initially developed on JAVA, and then ported to Visual Basic. The reason for choosing the JAVA platform at the first instance and its advantages will be mentioned in the following paragraphs.

Several reasons can be counted for the porting of the student assessment and coaching system to Visual Basic, and further developing the interface on it:

- The need for a full integration with the control software of underlying tele-robotic hardware that was written Visual Basic. During the actual runs, the information exchange in the form of student assessment and coaching system receiving parameters and the interface receiving the state of the robot should be made between the robot control software and the user interface. For this reason, one integrated program is implemented instead of two independent programs that should communicate in real-time.
- Ease of use for Visual Basic. Being a visual development language, Visual Basic brings pre-defined useful objects and components for the development of user interfaces in a very short time. Its syntax tries to resemble English language and it has no pointer arithmetic to complicate program development. It has pre-coded Internet and database connection objects and functions, making a development of an Internet connected application very easy. In addition, data base creation and query functions are very useful for taking the burden off the programmer.

- Being an event-driven language, the Visual Basic is a good choice for the user interface that is to be developed for the tele-robotic application that is mentioned in this chapter. The application is event-driven, which can be shown for several examples:
    - The tele-robot control software sends the parameters to the student assessment and coaching system after an action is completed, which clearly makes up for an event.
    - The server side of the tele-robot control software receives the commands for the manipulation of the robot from the client side in an event-based fashion.
    - The end of an evaluation interval is also an event, which triggers the error classification and decision process.

The porting of student assessment and coaching system's code from JAVA to Visual Basic was relatively easy, because most of the code was composed of arithmetic operations for the calculation of the main modules' outputs. As previously stated, the student assessment and coaching system along with the basic user interface for demonstration was first coded with JAVA language. There are several reasons for using JAVA for a development of remote experimentation system. These can be summarized as:

- JAVA is a platform independent language that can run in different operating systems on different hardware, displaying the same output and functions. The applets written by JAVA language can be integrated into the web pages, so that no software implementation is necessary, whereas in the case of Visual Basic the client program has to be installed. By this

way a truly web based remote experimentation system can be implemented in JAVA.

- JAVA also provides capabilities for the ease of graphical user interface development with the pre-defined and coded objects and functions.
- Being an event-based language, it is suitable for the development of remote experimentation systems for the similar reasons that are mentioned for Visual Basic.
- JAVA provides capabilities for displaying video streams and 3-D graphics on the web applets, where displaying these kinds of streams on web browser without necessary plug-in software is impossible.
- The persistent and fast socket connection for the client to remotely control the tele-robotic software is easily implemented in JAVA on the web applets. Further, a JAVA client sends requests to the server without waiting for a reply; at the same time, an independent processing thread in the client program constantly listens for messages from the server. As they are received, this listener thread forwards those server messages to the central applet component for display in the appropriate portion of the user interface. An example of this is the display of messages to the student without any request from him/her.

The use of JAVA and its capabilities will be further elaborated in chapter 5, where propositions and recommendations for a generic remote experimentation system will be made.

4.3.4. Development of Experimentation Scenario

In order to implement a remote experimentation through Internet, a suitable experiment scenario must be designed. The design should satisfy the requirements below:

- The scenario must be composed of different experiment parts. Each experiment must be divided into sub tasks. This provides an easier evaluation of the user's performance by the intelligent evaluation module, which takes the role of a lab assistant, or a teacher.
- The experimentation should be progressive. Successful accomplishment of an experiment should result in new experiments. This progress must go from the easiest experiment to the hardest one. If the user finishes an experiment then the system should direct him/her to a new and a harder one.
- The scenario should satisfy the user's learning needs. The appropriate experiments must be presented to the user in an appropriate order. Unnecessary workload for the user should be avoided. The users should be able to receive maximum benefit from the experimentation for his/her intended use.

With the requirements mentioned above in mind, an experimentation scenario is determined. The whole scenario is divided into three main experiment parts. In addition, these main experiment parts are divided into sub sections. The order of the experiments are running from easier to harder, and appropriate tasks are expected from the user in each part and sub divisions.

The experimentation scenario involves drawing lines, shapes, and then finally letters with manipulating the robot on a sheet of paper that is provided in the tele-robotic setup. This experimentation context is chosen in order to demonstrate the ideas that are mentioned in the above paragraphs. Although the experimentation involving drawing letters with manipulating the robot may be trivial and unnecessary above a certain educational level, this example has all the properties of an experimentation scenario that is required to be suitable for student performance evaluation and guidance. For this reason, this proposed scenario can demonstrate the properties and capabilities of the remote experimentation system proposed in this thesis. Further, in chapter 6, the ways to tackle with the requirements and properties of the experimentation schemes that have more educational value will be discussed.

In the experiments, the student is either expected to draw in a coloured region or he/she is required to trace the contour of a figure. The coloured regions are provided for line drawings and the figure contours are provided for shapes and the letters. These coloured regions and figures provide the user throughout the experiments with the guidance for what to achieve with the manipulation he/she is doing with the tele-robot. This was a necessary measure, because in this context, the student is not additionally provided with the information regarding the steps of the experiment. These visuals present the simple aim of the experiment, which is to draw correctly the lines, the shapes and the letters on the sketching paper with the remotely controlled robot.

The figures and the coloured regions are displayed both in the user interface and the actual sketching paper. The reason for this is that the student will also be able to see the actual sketching paper via the webcam, and thus will perceive what

the manipulations he/she is doing on the user interface correspond to in the real world. While the user is performing the experiments, only the current step of the experiment is shown in the virtual paper in the user interface.

4.3.5. Experiment Steps

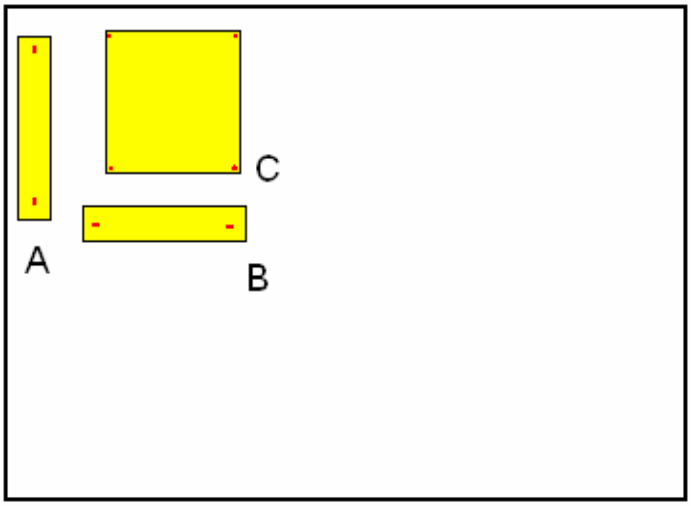


Figure 4.4: The first experiment

As previously stated, the experimentation scheme is composed of three main experiment steps and three sub-steps in each experiment step.

In the first experiment, the user is expected to draw straight lines on the sketching paper. The yellow regions on the sketching paper are provided for this experiment.

In the first part, the task is to draw a vertical line in the limits of the rectangle A provided and starting from and ending at the red points placed inside.

The second part involves drawing a horizontal line between the starting and end points in the rectangle B provided.

Finally, the first experiment finishes with the third part where the user is expected to draw diagonal lines between the red points inside the rectangle C. The experiment steps can be seen altogether in figure 4.4.

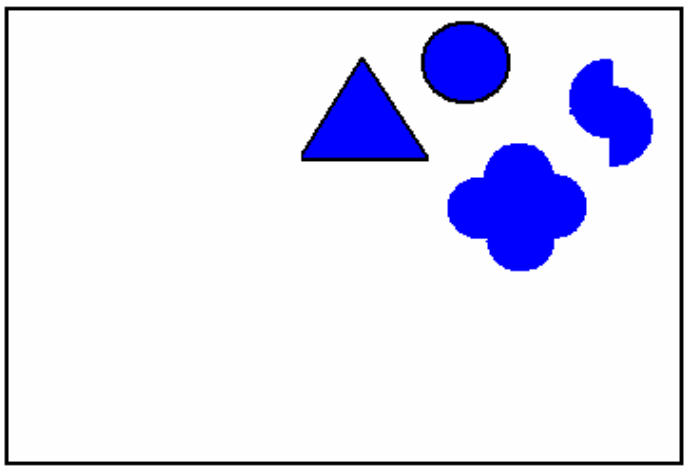


Figure 4.5: The second experiment

For the second experiment, the user is expected to draw closed surfaces and curvatures tracing the contours of the regions provided. This experiment involves four parts, which represent different closed surfaces. Firstly the contour of a triangle and then a circle must be drawn. After these, the contours of the provided additional two closed surfaces should be drawn as well. Each closed surface represents a different experiment part. The easiest one is the triangle, and the hardest one is the closed surface presented at the end. The experiment steps can be seen altogether in figure 4.5.

Figure 4.6: The third experiment

Finally, in the third experiment, the user is required to draw letters, which are predetermined and present on the sketching paper; the aim is making the user successfully trace the given letters. The main difference from the second part is, to correctly trace the given letters the user must elevate the pen and touch it back to a suitable place on the paper. There are three letters "E ", "A", "R" to be drawn and they are laid down on the sketching paper respectively. The experiment steps can be seen altogether in figure 4.6.

Figure 4.7 shows all of the experiments in the virtual sketching paper.

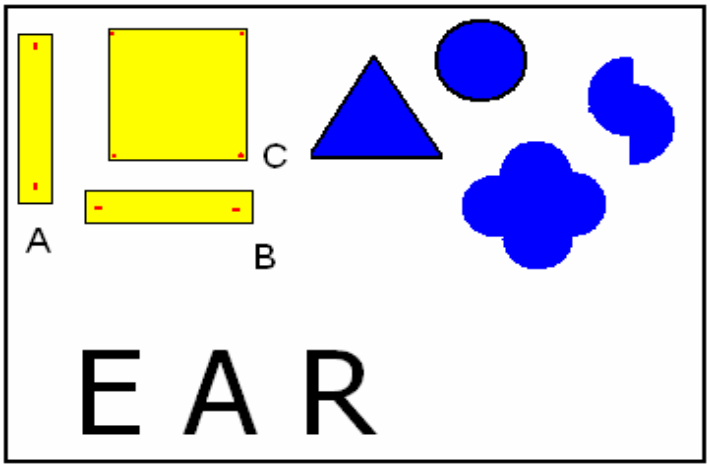


Figure 4.7: The experiment sheet layout

4.3.6. Illustrative Examples from the Application

To illustrate each system module in operation, the results from a circle drawing sub-step of the second experiment will be considered as an example in this section.

As previously mentioned in the chapter dedicated to the operation of student assessment and coaching system's modules, for the circle drawing experiment sub-step, the error quantization module takes the actual position coordinate values of the pen and compares them to the desired values. If the position error is outside the performance limits then "-1" value is assigned automatically to the action performed, otherwise "1" value is assigned which indicates an acceptable move.

Figure 4.8 shows the magnified version of the fist part of the actual circle drawing experiment and the error made on a portion of the arc drawn over the desired circle. Table 4.1 shows input error values and the corresponding quantized error outputs for the case of an evaluation interval value of 12. The desired position values that the error quantization module uses to compare the actual values with are obtained from the circle equation.

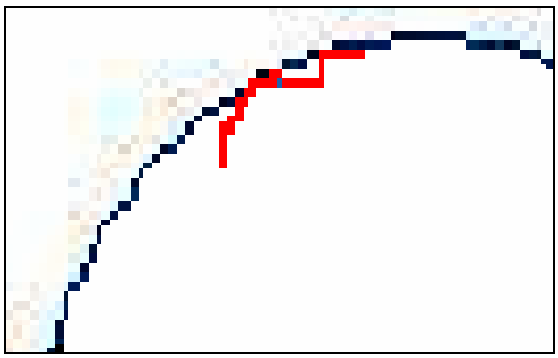


Figure 4.8: The red line shows the actually drawn arc in the circle drawing experiment with the evaluation interval 12.

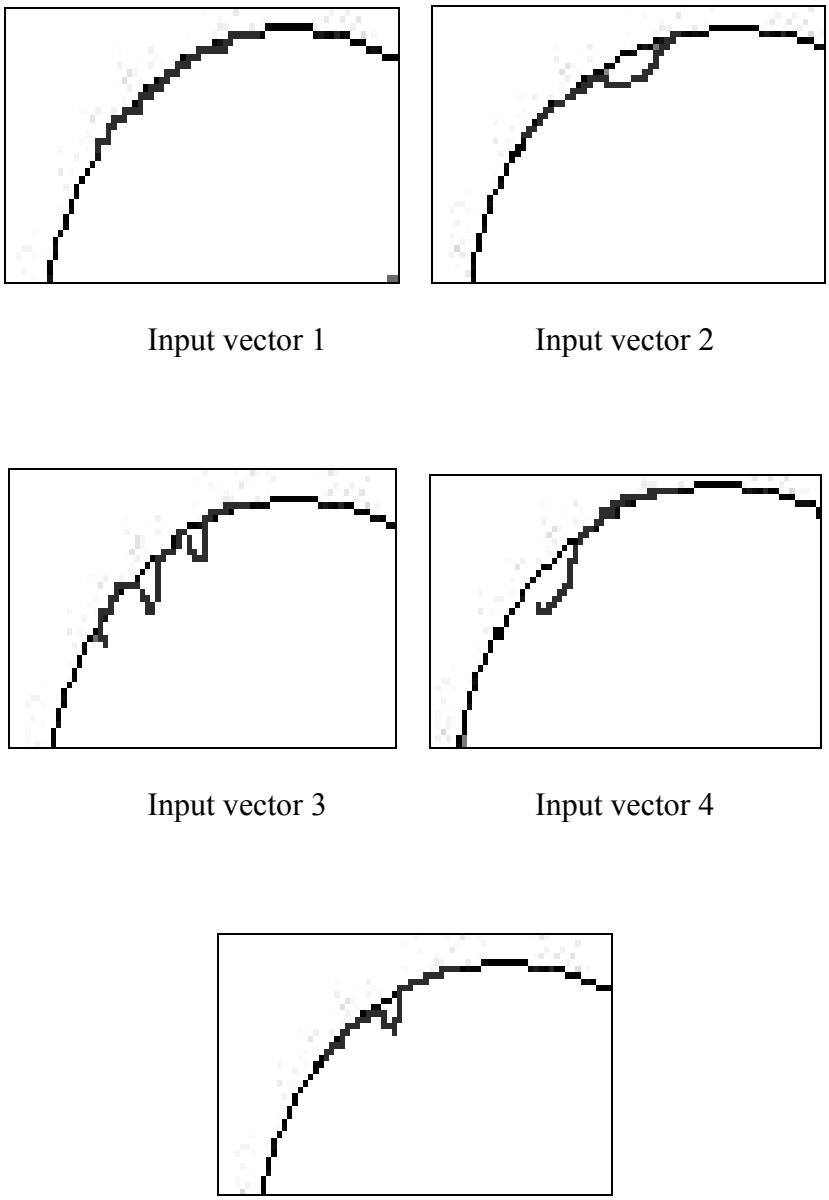
Table 4.1: Results from error quantization part (threshold value =10)

Action No.	Error	Output
1	5	1
2	10	1
3	12	-1
4	25	-1
5	5	1
6	4	1
7	0	1
8	2	1
9	9	1
10	14	-1
11	35	-1
12	22	-1

The sensitivity of this module is directly dependent on the quantization threshold, which is the error limit between the actual and the desired position coordinate values in this context. Quantization is used in the student assessment and coaching system to unbias initially errors from local vibration of the pen due to user fatigue, instant loss of attention, etc.

The resulting unbiased behavior vector, given in the last column of table 4.1, is the input to the artificial neural network of the error classification module. For the illustration of the inputs and outputs of the error classification module, the artificial neural network outputs that correspond to different input vectors are shown in table 4.2. The actual arc drawings that generate the input vectors are given in figure 4.9. At this stage, the error classification module's output just indicates an error type or class that defines roughly the user performance as explained in chapter 3, while the

fuzzy decision making module assigns meaning to the error classification module's output through evaluation.



Input vector 1 Input vector 2

Input vector 3 Input vector 4

Input vector 5

Figure 4.9: Actual drawings of arcs (in red) generating the input vectors of table 4.2

Table 4.2: Results from the error classification module

Input Vector	Output Vector
[1 1 1 1 1 1 1 1 1 1 1 1]	[1 1 1]
[1 -1 -1 -1 1 1 1 1 1 1 1 1]	[-0.5 1 1]
[1 1 -1 1 1 -1 1 -1 -1 1 1 -1]	[0.5 -2.7e-4 -6.9e-4]
[1 1 1 1 1 1 1 -1 -1 -1 -1 -1]	[1 0.5 -1]
[1 1 1 1 -1 -1 1 1 1 1 1 1]	[1 9.1e-4 1]

The fuzzy decision process generates the meaning to the error classification module's outputs by determining the class that a particular user behavior falls in and calculates the user grade for that evaluation interval. As a final evaluation output, a message is issued according to the determined performance class and the grade for the evaluation interval. This grade is also temporarily stored in order to be used for the calculation of the overall grade of the whole experiment. Table 4.3 shows the success evaluation outputs of the fuzzy decision process for each of the error classification module's outputs that are also shown in table 4.2.

Table 4.3: Results from the decision process

Input Vector	Resulting Class	Resulting Grade
[1 1 1]	S	100
[-0.5 1 1]	S3	70
[0.5 -2.7e-4 -6.9e-4]	F2	- (Repeat)
[1 0.5 -1]	F3	- (Repeat)
[1 9.1e-4 1]	S2	80

CHAPTER 5

SENSITIVITY ANALYSIS OF STUDENT ASSESSMENT AND COACHING SYSTEM

5.1. Introduction

In this chapter the performance of the student assessment and coaching system applied to the particular tele-robotic application for drawing experiments, which is explained in detail in chapter 4, will be discussed. In addition to that the results of the sensitivity analysis of the student assessment and coaching system according to the parameters that are used for each module in the system are presented along with the discussion on their effects to the sensitivity.

In the first section of this chapter, the shortcomings of the student assessment and coaching system are explained when the system is presented with some particular error vectors which represent an unwanted user behavior. In this context, unwanted user behavior means that the student shows no effort in learning or successfully carrying out the experiment goals.

The second section includes the sensitivity analysis of the system towards the system's parameters. Among these parameters, the evaluation interval value is the general system parameter, whereas the number of neurons in the hidden layer, the training sets, and the error tolerance value are related to the artificial neural networks, which are employed in the error classification module of the system. The

number of the fuzzy sets and the areas under them for the fuzzification of error classification module outputs constitute the parameters for the decision process module of the system. Analysis of the performance of the student assessment and coaching system is done for each parameter by changing that parameters value while keeping the other parameters constant.

5.2. Discussion on system performance

During test runs of the experimentation framework, which is explained in chapter 4 for the drawing example, the main confusion of the decision module is found to be between the “F2” and the”S2” cases that is due to the input vector of the ANN exceeding the number of errors that is considered as success. For the example mentioned in chapter 4, in an interval, a maximum number of 4 errors can be evaluated as success when the evaluation interval value is 12. In such cases where the number of errors exceeds the limit that is dictated by the evaluation interval value, the system may decide “success” in the interval instead of “failure”. This happens when the excess errors are distributed unevenly throughout the evaluation interval, that is the input vector to the ANN has "-1" values distributed unevenly among "1" values. As an example, the input vector [-1 1 1 -1 1 1 1 -1 1 1 -1 -1] is classified as "S2", although the correct class should be "F2". Consequently, the student assessment and coaching system exhibits a poor evaluation performance when the user does not show an effort of learning when executing the task, which results in random-like distribution of errors. Random-like errors or no effort of learning during experimentation is naturally not within the aim of the study. The student assessment and coaching system focuses on evaluating students that execute

experimentation while learning from it. However, for completeness of performance analysis, the system is still tested with randomly generated input user task error vectors. This meant that the user makes no learning efforts during experiments. The evaluation system then, generated wrong decisions and grades but, still with low enough error percentages:

- evaluation interval value=8 error rate=4%
- evaluation interval value=12 error rate=6%
- evaluation interval value=16 error rate=10%

The other lack of sensitivity of the system to randomly created inputs that became apparent after testing is that, the "F2" decision dominates much of the user behaviors. For example in the case the evaluation interval value is 16, the inputs, with number of errors (“-1" values in the input vector to the ANN) ranging from 6 to 12, result in the "F2" decision category. Even if there are less consecutive errors say 5 in number, in the first 5 actions, the remaining errors distributed unevenly over the other actions render the performance as "F2". Therefore not only excess number of errors but also their uneven distribution adversely affects the performance of the system. This means that any user (student) that does not exhibit an effort for improving his/her experimentation will be doomed by the system to continuously repeat the same task with failure grades.

Needless to say that, under actual user operation for distance education with lab access, input error vectors will not at all be random since the user will make learning efforts and, the confusion of the system by consistently pointing to failure will dramatically decrease. The user behaviors will then fall into correct categorizations with much better success.

5.3. Sensitivity Analysis of the Student Assessment and Coaching System

The statements in the paragraphs above necessitate a rigorous sensitivity analysis of the student assessment and coaching system. For that reason, the sensitivity of the system towards the change of each parameter in the design is analyzed by changing one parameter while keeping the others untouched.

5.3.1. Evaluation Interval Value

In order to analyze system sensitivity to user behaviors during the course of an experiment, with different evaluation intervals, an error series is given to the system as a fixed input. Evaluation interval values are then changed systematically for the same error vector.

The error series in quantized form is applied to three different cases where the evaluation interval value is 8, 12 and 16. The error series is a bipolar vector of size 48: [1 1 1 1 -1 1 1 -1 1 -1 -1 1 1 -1 -1 1 -1 -1 1 1 -1 1 1 -1 -1 -1 -1 -1 1 1 1 -1 1 -1 1 1 1 1 1 1 -1 -1 1 -1 -1 1 1 1 1].

Figure 5.1 shows the classification result changes for changes in the evaluation intervals at the output of the decision process (thus, at the output of the whole system).

As can be seen from the figure, as the evaluation interval value increases, the system sensitivity to the variations in a single user behavior decreases: this means less classification changes at the output of the decision process for increasing evaluation intervals. This feature is in accordance with the trend of relying more to capable and experienced users while novice users are more closely evaluated in an

adaptive way: as explained in chapter 3 of this thesis which is the evaluation interval value increases by consecutive successful evaluation intervals.

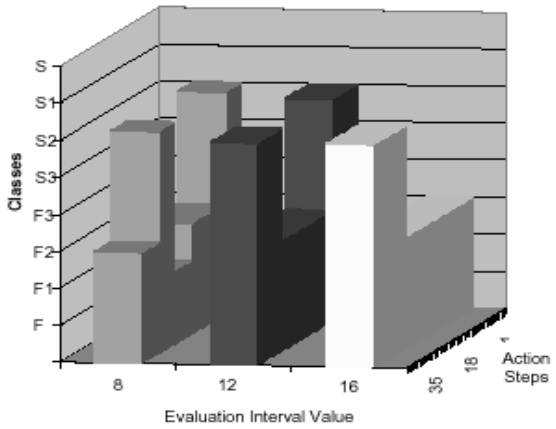


Figure 5.1: Classifications of actions for the changes in evaluation interval value

When the evaluation interval is kept fixed, the system's sensitivity to different input vectors, or in other words different user behaviors, depends on several parameters, the most important of which are those of the neural network and, the number and topologies of fuzzy sets.

5.3.2. The Number of Neurons in the Hidden Layer

Neural network properties that affect the performance of the system are mainly the number of nodes in the hidden layer, the residual error in the convergence of the algorithm and the training vectors.

The number of neurons in the hidden layer strongly affects the performance of the neural network, and must create a balance between the ability and the speed of learning. Too few hidden layer neurons will result in a poor performance and a

network with too many hidden neurons will take longer time to train in addition to the possible problem of over-fitting.

In addition, a poorly trained network will be incapable of classifying different user behaviors correctly whereas an over-trained network will exactly fit the training data, but be unable to draw generalizations regarding the test data. Thus, a trade-off is necessary while choosing a suitable number of hidden layer neurons for the neural network, in order not to decrease the correct classification capability.

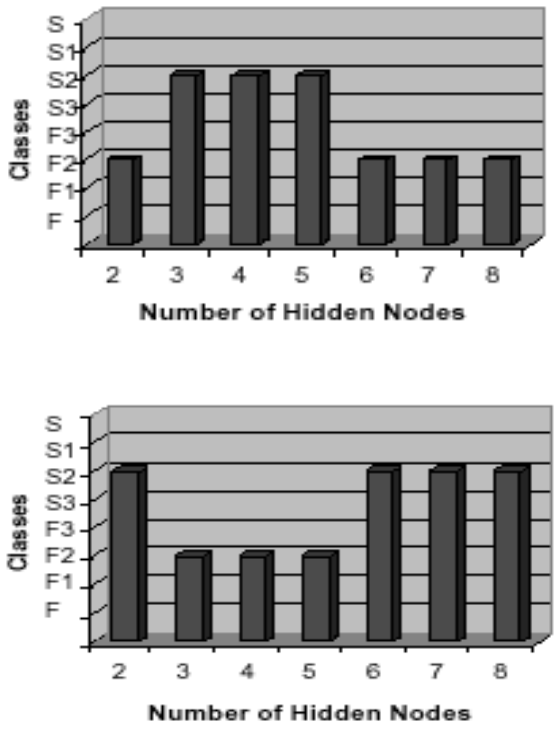


Figure 5.2.a, 5.2.b: The classification classes versus the number of hidden neurons for the first (above) and the second (below) fixed error vector

Figure 5.2.a and 5.2.b show the evaluation system output versus the number of hidden neurons in the neural network for fixed input error vectors of [1 1 1 -1 -1 1 1 -1 1 -1 1 1] and [-1 -1 1 1 1 1 1 -1 1 1 -1 -1] respectively, where the evaluation interval value, and hence the number of neurons in the input layer is 12. For the first

input vector the correct class is “S2”, where for the second one the correct class is “F2”. As can be seen from the figures, the system makes correct categorizations when the number of hidden layer nodes is 3, 4, and 5. Values other than these create confusion in the system between S2 and F2 and lead to wrong decisions about the performance. This confusion was also discussed in section 5.2 in this chapter in the context of the proposed application in chapter 4.

5.3.3. The Training Set

Another parameter that affects the neural network performance and hence the system sensitivity is the training set of the neural network. If the training set is small or composed of dependent vectors, then this set can not represent the whole input space of the artificial neural network of the error classification module, and thus the system classification performance will be degraded. For the implementation of the student assessment and coaching system, the training vector set size is kept at a value of 8 where the 8 vectors are independent and characterize the whole error space for the evaluation interval of a task. This set of 8 vectors has led to quick training without much processing load. These training vectors are found to represent fundamental user characteristics for the given implementation with the three evaluation interval values used. However, less training vectors for the same number of success categories can be given to the system in order to decrease the sensitivity to different user behaviors, especially for the case where the size of the input vector is 16, which implies a neural network with simpler connectivity.

Figure 5.3 shows the effect of the training vector set size on the classification results, given a fixed evaluation interval value (12) and two fixed input error vectors

to the neural network: [1 -1 1 1 -1 1 -1 1 1 -1 1 -1], and [-1 1 1 -1 1 1 1 1 -1 1 1 1]. For the first input vector, the output of the system should be “F2”, and for the second one the output should be “S2”. As can be seen from the figure 5.3, when the neural network is trained by 6 vector pairs instead of 8 the outputs are wrong, due to insufficient training.

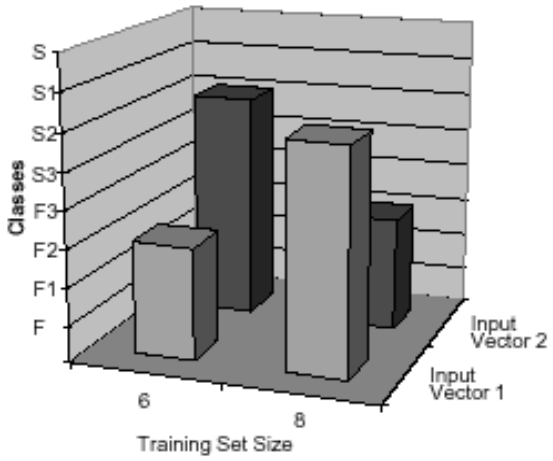


Figure 5.3: Changing of classifications due to the changing of training set size for two fixed input error vectors and evaluation interval value 12.

5.3.4. The Error Tolerance Value

The last parameter affecting the neural network performance is the tolerance value in the convergence during training, which is the residual error of the training output.

Along with the number of hidden layer neurons and the training vector size, the tolerance value determines whether the neural network performance is satisfactory. If the desired tolerance is too small, the training algorithm will execute much iteration, resulting in overfitting of the weights to the training data. With a trial

and error process, a reasonable tolerance value has to be set in order to provide quick convergence yielding the desired classification capabilities. Figure 5.4 shows an example of class decision changes generated by changes in tolerance. The input error vector applied to the system is [1 -1 1 -1 1 -1 -1 1 -1 -1 -1 -1], which would normally give the classification result as "F2", when the evaluation interval value is 12 and the number of hidden layer nodes is 5. As can be seen in the figure, the classification is correct when the tolerance value is either 2e-6 or 4e-6. When these values are decreased or increased by 10 times, wrong classification occurs due to overfitting or underfitting.

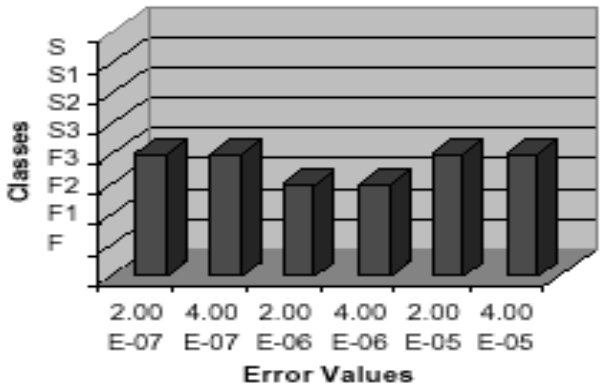


Figure 5.4: Changing of classifications due to the changing of error (tolerance) values for a fixed input vector and evaluation interval value 12.

5.3.5. Fuzzification

The number and the shapes of the fuzzy sets used in the decision process have effects on the sensitivity of the system as well. The effect of changing the number of fuzzy sets can be seen in figure 5.5. The frequency of wrong decisions increases when the evaluation interval values are higher (12 and 16) and the outputs

of the NN are fuzzified into lower number of sets (3 sets, which are LOW, MED and HIGH instead of 4). Less partitioning in error types and higher evaluation interval values naturally makes a loss of sensitivity that can be observed in the graphs in figure 5.5.

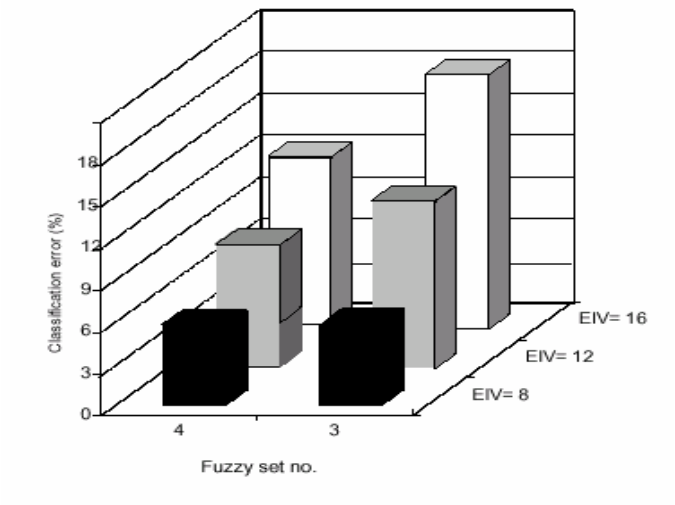


Figure 5.5: Results from different number of fuzzy sets

The last parameter analysis to be considered is changing the support of the fuzzy sets, thus the areas under them. All the other parameters of the system are kept fixed at their original values, the decision rules are not changed, and the effect of changing fuzzy membership functions on the sensitivity of the system are observed comparatively for fuzzy sets that are actually used in the system, which are shown in figure 5.6, and the sets depicted in figure 5.7. The classification outputs for the two cases are shown in figure 5.8 when the input error vectors are [1 -1 1 1 1 -1 1 -1 1 1 1 -1], [-1 1 -1 1 1 -1 1 1 1 -1 -1 1], [1 1 -1 1 1 1 -1 -1 -1 -1 -1 -1], and [1 -1 1 1 -1 1 -1 1 -1 -1 -1 -1], whose correct output classifications are “S2”, “F2”, “F3”, and “F2” respectively. “System1” denotes the original system with fuzzification functions that

are shown in figure 5.6, and "System2" denotes the system with fuzzification functions that are shown in figure 5.7. As can be seen from figure 5.8, the change in fuzzy membership functions and their overlaps leads to system confusion in the classifications for input vectors 1, 2 and 4. Total confusion occurs by the overlapping of more than two fuzzy sets, generating failure when success is expected or vice versa.

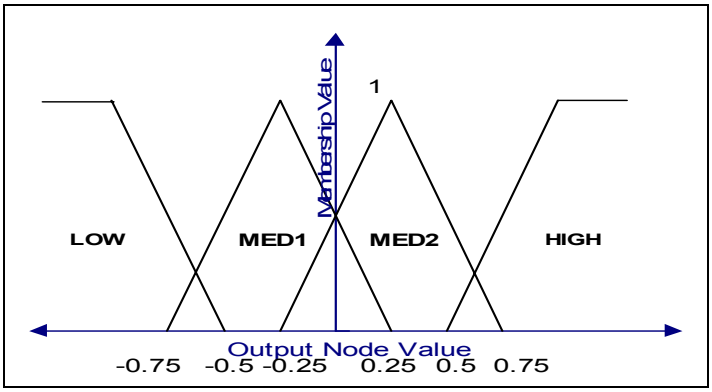


Figure 5.6: Actual fuzzification functions that are used in the system

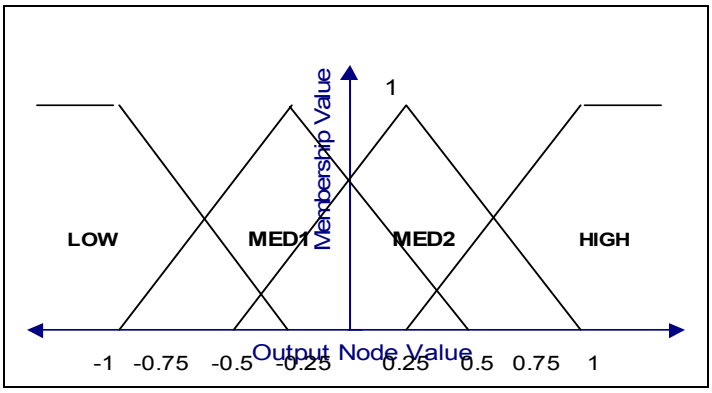


Figure 5.7: Fuzzification with the areas of the sets changed

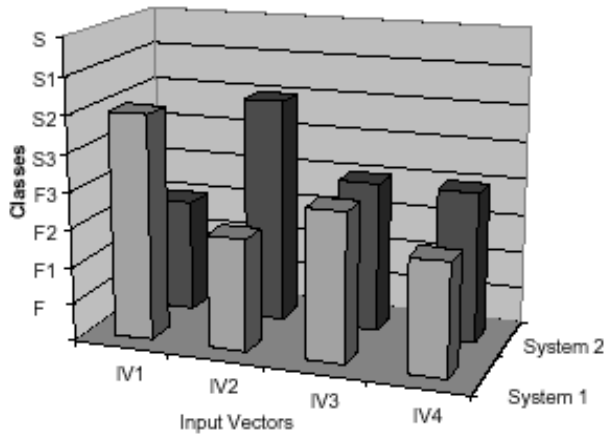


Figure 5.8: Changing of classifications due to changing of fuzzification for 4 fixed input vectors where the evaluation interval value is 12.

CHAPTER 6
EXTENSION TO ANOTHER EDUCATIONAL CONTEXT

6.1. Introduction

In this chapter, in accordance with the aim of providing a generic remote experimentation framework for different educational contexts, the student assessment and coaching system and the associated interface, which is developed for applications to robot-supported remote experimentation scheme, will be applied for demonstration on another robot-supported remote experimentation scheme. The laboratory considered is a mobile robot control laboratory, where students developed their mobile robot control programming skills taking the course associated with the lab and attending weekly lab sessions [59]. This laboratory is not a remote laboratory, however it can be improved to support remote access of the experiment hardware. Providing this remote access of the laboratory through the Internet is not within the context of this thesis work so we will not treat it in this chapter. This chapter only includes the application of the student assessment and coaching system to the experimentation scheme and development of the graphical user interface and the experimentation scenario to support a possible underlying system for remotely accessing this mentioned laboratory of Carnegie Mellon University [59] through the Internet.

The main idea behind the student assessment and the coaching system is the assumption that the particular experiment scheme in context is composed of a number of tasks and sub-tasks that has to be accomplished within a sequential ordering of actions. This assumption best manifests itself when a robot-supported system is in context. In order to do the experiment, the robot should be manipulated correctly in a sequential order that is; each control command sent to the robot should follow a pattern either to make a correct trajectory for motion or do an ordered set of actions to accomplish a task. Mobile robot control laboratory is thus chosen for the application of the intelligent remote experimentation framework.

In the following sections the development of the experimentation scenario, which consists of two experiments, is treated in detail first. Then the working procedure of the student assessment and coaching system with the underlying robotic hardware within the established context of the experimentation scenario is stated. The last section consists of the implementation of the user interface for the two experiments that are developed.

6.2. Application to a Remote Mobile Robot Laboratory

In accordance with the propositions stated in the above paragraphs, the remote experimentation framework is applied to a mobile robot experimentation scheme. The mobile robot control laboratory in context is a conventional laboratory that is taught in Carnegie Mellon University [59] as stated previously. For the experimentation, the students are required to program mobile robots to follow a path, using the sonar sensors to avoid obstacles and apply complex motion planning skills. The mobile robots are programmed by the supplied laptop computers that are

connected to the mobile robots. The students are required to go to the lab and spend time there to work on the experiments.

The proposed framework is applied to this experimentation scheme without working on the connection of the robotic hardware to the Internet. The application involves the user interface, the student assessment and coaching system and an experimentation scenario to complete the proposed framework.

6.2.1. The Experimentation Scenario

The experimentation scenario is composed of two experiments, which involve reaching a target in the lab environment through the Internet control of the robot such as to follow a certain path up to the predefined goal.

In the first experiment, the students can control the mobile robot with the sliders that correspond to the translational motion and the rotation of the robot. All the low-level controls generated to make sure that the robot follows the commands that are sent through the interface are handled by the underlying tele-robot control software. Therefore, the first experiment aims to provide the students with basic mobile robot manipulation skills. In addition to that, the other major aim of the experiment is make the students plan the most optimized path to the goal that is the shortest distance with fewer rotations and turns possible. Lastly, accomplishing the experiment with the smallest number of actions is another major aim of the first experiment.

The environment that the mobile roams in has obstacles in forms of walls. The student has to find the shortest path to reach the target without turning towards dead-ends and without going towards the walls. Right amount of translational motion

and necessary rotation are needed to steer the robot in that environment. The robot position coordinates and the map of the environments are provided for the students.

In the second experiment, the goal is again to reach a target in the environment. This time the students have to control the robot movement by entering commands for the motion. There are two commands for the motion to be accomplished in this experiment, which are:

1) TurnTo (int <tenths-of-degrees>): When called, the robot will turn <tenths-of-degrees> relative to its current orientation. Furthermore, the robot should turn the shortest way, for example, if given TurnTo (2700) the robot should turn clockwise 90 degrees.

2) GoTo (int <tenths-of-inches>): When called, the robot moves <tenths-of-inches> forward if <tenths-of-inches> is positive, otherwise backward.

These two commands should have to be developed from more lower level robot commands like "SetVelocities (rightwheel,leftwheel) and SetTimeout (period)". The difference between these higher level commands and that of low level is that these higher level commands specify movement to achieve relative positional change instead of specifying motion velocities. The students need to attain at least 90% accuracy using these functions in order to move the robot exactly as commanded to correctly follow the path to the experiment goal. For these reasons, the students need to take into account the encoder readings, which give a feedback of robot's position as x and y coordinates and orientation as an angle. For the accuracy

and the increased speed of the commands, the students should consider both PID control and delay compensation control.

In the second experiment, there are no obstacles in the environment and the path that the robot should follow is presented to the student on the virtual map on the user interface. The main aim of the second experiment is making the students write position relative control programs in order to introduce them to low level mobile robot motion control programming.

6.2.2. The Student Assessment and Coaching System

The student assessment and coaching system takes the feedback of robot's position coordinates. The error quantization module builds up the input vector to the classifier according to the robot's position after each command. If the robot's position stays within the limits of the path that has to be followed then the corresponding action is considered as being correct. The system presents an output after the end of each evaluation interval According to the decision process module's output; the student is either allowed to carry on the experiment or made to repeat the last interval. Upon failure, the student has to press a button on the user interface to carry on the experiment to be repeated. When this button is pressed, the robot goes back to the end of the last successful interval and then sending commands to the robot is enabled. As the case for the previous application, the number of success levels is 8. However, unlike the previous application only 3 of the levels mean success in the respective interval. That is because of the nature of the experimentation, which requires a strictly correct following of the path that leads to

the target in the environment. The success levels and the corresponding messages are:

- F: "Full failure on this interval, you should repeat this interval again. Please click the button at the bottom of the window."
- F1: "You have found the way but you have made many mistakes in the beginning. You have to repeat the last interval. Please click the button at the bottom of the window."
- F2: "You stayed in the wrong track mostly though you had some correct moves. You have to repeat the last interval. Please click the button at the bottom of the window."
- F3: "After a few initial correct actions you have lost your way. You have to repeat the last interval again. Please click the button at the bottom of the window."
- S: "Full success on this interval, you can carry on the experiment."
- S1: "You have lost the way but you were successful in the beginning. You can carry on the experiment, there is a chance that you can steer to the right path "
- S2: "You stayed on the right track mostly, but you have to repeat this interval again. Please click the button at the bottom of the window."
- S3: "You are on the right track after few initial mistakes at the beginning. You can carry on the experiment."

6.2.3. The Graphical User Interface

The graphical user interface for the experimentation is designed with the requirements that are stated in chapter 4 in mind. The user interface provides the students with the necessary information and manipulation tools in order to carry out the experiments successfully. The user interfaces for the first and the second experiments are shown in figures 6.1 and 6.2, respectively.

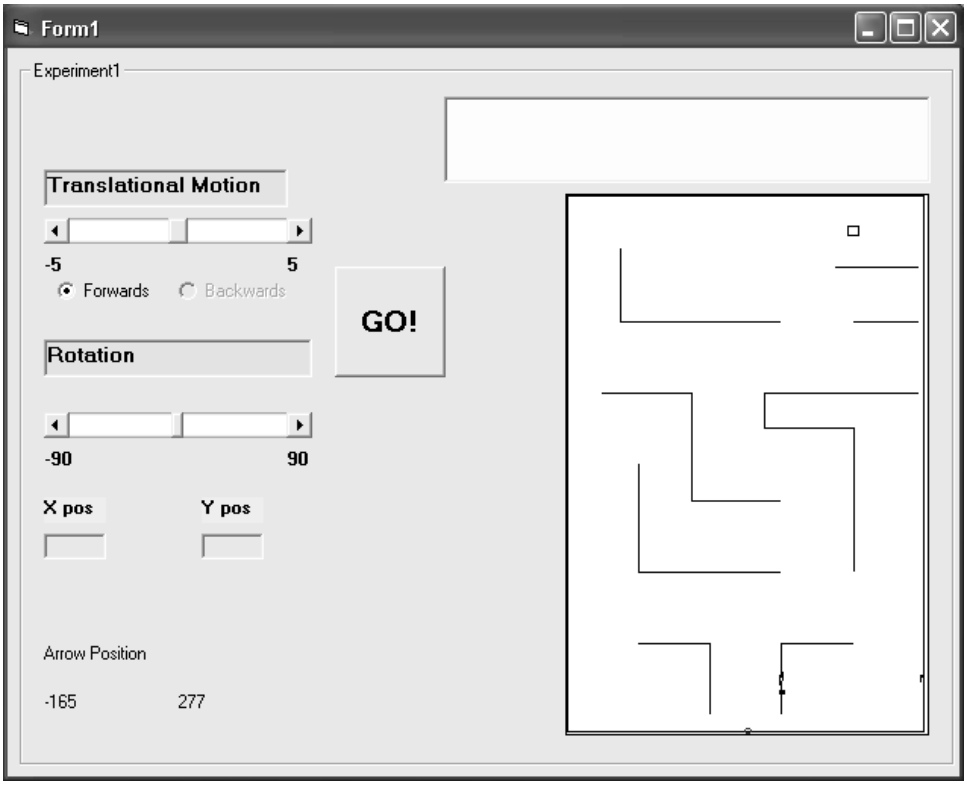


Figure 6.1: The user interface for the first experiment

The user interface for the first experiment includes the controls for controlling the robot's motion, the area for the messages that are presented to the student and the virtual map of the environment. The robot movement is controlled by the sliders that correspond to translational motion and robot's rotation. After

adjusting the required translational motion and rotation by the sliders, the command is sent to the mobile robot by clicking on the button labeled as “Go”. The robot’s current position coordinates is displayed in the areas under the sliders, which are labeled as “Xpos” and “Ypos”. In the virtual map, the robot is represented by a small, red circle, and the path that is traversed by the robot is displayed as a dotted red line. After each evaluation interval period, a message that informs the user of his/her performance during that interval is displayed on the yellow area above the virtual map. When the student has to repeat the last evaluation interval, the “Go” command is disabled and a separate button labeled as “Click” appears in the bottom of the window. Upon clicking this button, the small, red circle, which represents the robot on the virtual map, goes back to the end of the last successful interval, the traversed path is erased and the “Go” button is enabled again.

The user interface for the second experiment includes the code window, the boxes for commands, the message area, the virtual map of the environment and the area for the encoder readings. Upon loading of the user interface window, depending on the student, either the code window or the boxes for the robot control commands are enabled. The boxes for the robot control commands correspond to the two major commands that the students will be using for the movement of the robot. These commands are “SetVel” and “TimeOut”. For the students that are doing this experiment for the first time, these boxes are enabled, and the values for the left wheel velocity, the right wheel velocity and the timeout period can be entered. The text box for entering robot control commands is enabled for more expert students, who has to write their own program code as mentioned in section 6.6.1 to make the robot move or turn relative to the current position. The commands are sent to the

robot by clicking on the button labeled as “Send Command”. The encoder readings are displayed in the boxes below the command area, which are labeled as “Xpos”, “Ypos”, and “Theta” respectively. As in the case for the first experiment, the messages to the student are displayed in the yellow box above the virtual map, and after each failed evaluation interval the button labeled as “Click” appears in the bottom of the window, while “Send Command” button is disabled, to start from the end of the last successful evaluation interval. In this experiment, the correct path to be followed is shown on the virtual map as a red line and the robot is represented by a small, black circle, where the traversed path is shown by black dotted line.

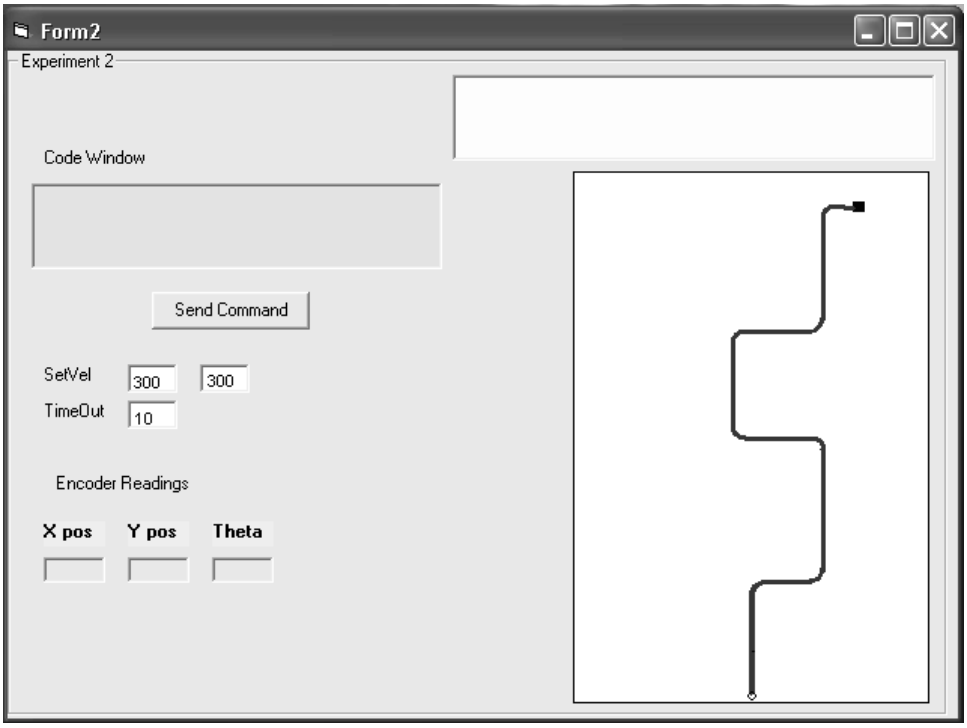


Figure 6.2: The user interface for the second experiment

CHAPTER 7
CONCLUSION

7.1. General

In this thesis, a general framework for an intelligent remote experimentation interface on robot-supported laboratory setups has been proposed. The connection medium for this kind of experimentation is the ubiquitous Internet and especially the World Wide Web. The framework has then been proposed to be extended to other remote experimentation platforms where experimental setups consisting of test and measurement devices connected to the Internet.

Student evaluation and guidance form the fundamental parts of the framework that has been proposed. In addition, the graphical user interface constitutes the other major issue in remote experimentation architectures. The work in this thesis firstly involves developing and implementing a student assessment and coaching system, and then an associated user interface is developed and applied to remote experimentation setups that are supported by robotic equipments controlled through the Internet. The system and the associated interface are extensible to other remote experimentation schemes through the Internet in various educational contexts. The student assessment and coaching system is composed of several modules to model the student behaviour, to classify the student behaviour into major

categories, to evaluate the student performance and to guide the student for better achievement of the experimentation goals. Student guidance is done by providing the students with information regarding the success degree of their actions and the places where they have been weak. The whole course of an experiment is divided into intervals, after which the evaluation is done and the student is informed. This paves the way to the adaptivity, because the continuation of the experiment depends on the student performance, the unskilled students will be directed to more basic experiments.

The other aspect of this research is to propose an experimentation scenario that can be suitable for the education oriented use of remote experimentation. From this implementation, a generic methodology for developing experimentation scenarios for remote experimentation through the Internet, which can be supported by the student assessment and coaching system has been reached.

The student assessment interface and the graphical user interface have been applied to an existing tele-robotic hardware that can be controlled through the Internet. While doing that, an experimentation scenario that can be used with the proposed framework and the hardware involved has been developed. The experimentation scenario has been designed to be in consistency with the properties of the student assessment and coaching system and the tele-robotic hardware. The details of overall development of the experimentation framework have been presented along with the illustrative results that show the whole system in operation and the parameter sensitivity analysis for the student assessment and coaching system.

With the advance and the widespread use of Internet and the WWW, the e-learning concept, that is distance tutoring through the Internet and the WWW, has already become a reality. Further, to enhance e-learning and to extend the concept to other educational contexts, there has been a great deal of research done in the past couple of years and this research area is still one of the most active areas involving multiple disciplines such as educational science, computer science and mechatronics. The main difference between the work proposed in this thesis and the other remote experimentation research is the emphasis on the educational aspect, which is the evaluation of student performance and guidance towards successful completion of the experiment, and therefore better understanding of the subject in context. This has been possible with the addition of an intelligent system, which can model the user behaviour, change the contents according to the user and provide the users with useful information relevant to the context. In other words, the main significance of this work is trying to create a synergy between the conventional remote experimentation research that deal with the tele-operation, graphical user interfaces and the experiment scenarios, and the research on intelligent distant tutoring and human-computer interaction which involve student modelling, adaptive interfaces and student evaluation and guidance.

7.2. Further Work

The future work mainly involves modifications and further developments for the student assessment and coaching system and the associated graphical user interface for remote experimentation to support different educational contexts.

After implementing the student assessment and coaching system and the user interface associated with it, one of the most important further works is to extend it to other educational contexts in order to provide a generic remote experimentation framework. As previously stated, extensibility is the major reason behind the implementation of the system as a collection of modules. The modifications that are necessary to satisfy the requirements of a particular experimentation schemes thus can be made on each module without compromising the general philosophy behind the student assessment and coaching system. For the mechatronics experiments with a mobile robot mentioned in section 2.3.1, an experimentation scheme involves environment modeling with ultrasonic sensors and laser range finder. The student assessment and coaching system can be applied to this scheme such that the robots movements can be tracked and the positions that the student commands to take measurements can be input to the error quantization module which outputs the corresponding error vector after comparing the actual positions to the correct position values. In this case, the error classification module can be trained by a training vector set that corresponds to a number of vectors denoting different trajectories of measurement points. For the chemistry experiment mentioned in section 2.3.2, the student can change the parameters of reagent concentrations, pH, temperature, ionic strength and flow rate before starting up the experimentation. In order not to cause damage to the experiment setup and to evaluate the students' knowledge on changing the parameters, the student assessment and control system can track these parameter changes. The error quantization module can form the error vector as -1 and 1 values corresponding to each parameter change. Needless to say that there might be success situations even some parameters are not commanded to

be set in the established limits. For the electronics experiments, which are mentioned in section 2.3.4, the students can build up the electronic circuits using the user interface. The circuits can be built by either remotely controllable switches or robotic devices. Setting up the correct circuit for the experiment is one of the major practical aims of the electronics laboratory. The other practical aim is using the test probes and the measurement equipment correctly in order to get the desired current and voltage measurements. The student assessment and coaching system will be modified to track student commands and evaluate the student performance during the experiment, and provide necessary coaching information. In the control theory experiments that are stated in section 2.3.5, the student assessment and coaching system can be integrated within the remote experimentation framework on the assumption that the student makes a number of experiments to apply different control methods to the system. The student assessment and coaching system can track which control method is applied and how the parameters are changed in what order.

Another important further work is porting the tele-robotic experimentation framework that is presented in chapter 4 to the World Wide Web for providing better portability between various platforms and accessibility with the commonly used web browsing programs.

Designing new remote experimentation scenarios for different educational contexts that can aid better understanding of the subject matter while doing experiments over the Internet is also a further step in improving the remote experimentation framework.

The additional further work to improve the remote experimentation framework is making changes in the student assessment and coaching system methodology to further underline the concepts of student modelling and adaptive media, and further research on the graphical user interface issues, taking into account the psychological and ergonomical requirements for providing better human-computer interaction.

Lastly, testing the resulting framework in a real distance learning scheme, where the students will be the subjects that can provide necessary and useful feedback on the system's performance, usability and relevance to the educational context they are learning, is another major further work.

REFERENCES

[1] K. Goldberg, M. Mascha, S. Gentner, N. Rothenberg, C. Sutter, and J. Wiegley, Desktop tele-operation via the World Wide Web, Proc. IEEE Int. Conference on Robotics and Automation, pages 654-659, 1995

[2] K. Goldberg, J. Santarromana, G. Bekey, S. Gentner, R. Morris, J. Wiegley, and E. Berger, The Telegarden, Proceedings of ACM SIGGRAPH, pages 135-1140, 1995

[3] M. J. Cox and J.E.F. Baruch, Robotic Telescopes: An Interactive Exhibit on the World-Wide Web, Proc. 2nd International Conference of the World-Wide Web, Chicago, October 17-20th 1994

[4] G.T. McKee, and Barson, R. NETROLAB: providing access to robotics technology using the Internet. Robotics and Machine Perception Special issue: Networked Robotics. USA: SPIE, 5, 1, 1996

[5] E. Paulo and J. Canny, Delivering real reality to the world wide web via telerobotics, Proceedings of IEEE International Conference on Robotics and Automation, pages 1250-1256, 1996

[6] U. Nehmzow, A. Buhlmeier, H. Durer and M. Nolte, Remote control of mobile robot via Internet, Dept. Of Computer Science, University of Manchester, Technical Report Series, UMCS-96-2-3, 1996

[7] K. Goldberg, et al., DIGIMUSE: An interactive telerobotic system for remote viewing of 3D art objects, IROS'98: Workshop on Web Robots, pages 55-60, Victoria, Canada, 12-17 October 1998

[8] K. Kosuge, J. Kikuchi and K. Takeo, VISIT: A Teleoperation system via computer Network, Proc. IROS'98: Workshop on Web Robots, pages 61-66, Victoria, Canada, 12-17 October 1998

[9] K. Brady and T.J. Tarn, Internet-based Remote Teleoperation, Proc. IEEE International Conference on Robotics and Automation, pages 65-70, Leuven, Belgium, 1998

[10] A. Ferworn, R. Roque and I. Vecchia, MAX: Wireless teleoperation via the World Wide Web, Proc. IASTED Conf. on Robotics & Applications, pages 158-162, Santa Barbara, 28-30 Oct. 1999

[11] M. Habib, "Real Time Control and Monitoring over the Internet", IMEKO/IFAC/IFIP workshop on Advanced Robot Systems and Virtual Reality (ISMCR 2000), Vienna, Austria, 23-30 Sept. 2000

[12] Maki K. Habib, "Tele-Monitoring and Control Through the World Wide Web: Issues and Development", GMD-Japan Research Laboratory.

[13] K. Goldberg, J. Santarromana, "A Tele-Robotic Garden on the World Wide Web", SPIE Newsletter, spring 1996

[14] *http://telerobot.mech.uwa.edu.au*

[15] R. Simmons, "XAVIER: An autonomous mobile robots on the Web", Proceedings of IROS'98 Workshop on Web Robots, pages 43-48, Victoria, Canada, 12-17 October 1998

[16] P. Saucy and F. Mondana, "KhepOnTheWeb: Open access to a mobile robot on the Internet", IEEE Robotics and Automation Magazine, pages 41-47, March 2000

[17] D. Schulz, W. Burgard, D. Fox, S. Thrun, and A.B. Cremers, "Web interface for mobile robots in public places", IEEE Robotics and Automation Magazine, pages 48-56, March 2000

[18] Lixiang Yu; Pui Wo Tsui; Quan Zhou; Huosheng Hu, "A Web-based Telerobotic System for Research and Education at Essex", IEEE/ASME International Conference on Advanced Intelligent Mechatronics, Como, Italy (2001)

[19] *http://www.ece.cmu.edu/~stancil/virtual-lab/concept.html*

[20] R. Safaric, M. Debevc, R. Parkin, and S. Uran, “Telerobotics Experiments via Internet”, IEEE Transactions on Industrial Electronics, 48(2):424–31, April 2001

[21] S. You, T. Wang, E. Roy, M. Cai, and Q. Zhang, “A Low-Cost Internet-Based Telerobotic System for Access to Remote Laboratories”, Journal of Advanced Engineering Informatics, vol. 15, issue 3, pp. 265−274, March 2001

[22] Ulrich Karras, Hauke Ernst, "DERIVE, Distributed Real and Virtual Learning Environment for Mechatronics and Tele-service", WS 2001, International Workshop on Tele-Education in Mechatronics Based on Virtual Laboratories, July 2001, Weingarten, Germany.

[23] http://www.vvl.de

[24] J. M. Martins Ferreira, Ricardo J. Costa, Gustavo R. Alves, Martyn Cooper, “The PEARL Digital Electronics Lab: Full Access to the Workbench via the Web”, 13th EAEEIE Annual Conference, York, England, April 2002

[25] Xavier Vialta, Denis Gillet and Christophe Salzman, "Contribution to the Definition of Best Practices for the Implementation of Remote Experimentation Solutions" IFAC Workshop on Internet Based Control, IFAC Workshop on Internet Based Control Education, IBCE’01, Madrid, Spain, December 12-14, 2001

[26] Christoph Rohrig, Andreas Jochheim, "Java Based Framework for Remote Access to Laboratory Experiments" In Proc. IFAC/IEEE Symposium on Advances in Control Education, Gold Coast, Australia, 2000

[27] Judy Kay,”Learner Control”, User Modeling and User-Adapted Interaction 11: 111-127, 2001, Kluwer Academic Publishers

[28] Peter Brusilovsky, "Adaptive Educational Systems on the World-Wide-Web: A Review of Available Technologies" Workshop "WWW-Based Tutoring" In: Proceedings of at 4th International Conference on Intelligent Tutoring Systems (ITS'98), San Antonio, TX, August 16-19, 1998.

[29] Peter Brusilovsky, "Adaptive Hypermedia", User Modeling and User-Adapted Interaction 11: 87-110, 2001, Kluwer Academic Publishers

[30] G. I. Webb, M, J. Pazzani and D. Billsus, "Machine Learning for User Modeling", User Modeling and User-Adapted Interaction 11: 19-29, 2001, Kluwer Academic Publishers

[31] Alfred Kobsa "Generic User Modeling Systems"; User Modeling and User-Adapted Interaction. 11(1-2), 49-63. (2001)

[32] Nicola Capuano, Marco Marsella, Saverio Salerno, "ABITS: An Agent Based Intelligent Tutoring System for Distance Learning", Proceedings of the International Workshop on Adaptive and Intelligent Web-based Educational Systems held in Conjunction with ITS 2000 Montreal, Canada.

[33] R. Yasdi, "A Literature Survey on Applications of Neural Networks for Human-Computer Interaction", Neural Comput & Applic (2000)9:245–258 2000 Springer-Verlag London Limited

[34] Stathacopoulou R. , Magoulas G.D. and Grigoriadou M., "Neural network-based fuzzy modeling of the student in intelligent tutoring systems", in Proceedings of the INNS-IEEE International Joint Conference on Neural Networks, Washington, U.S.A., July 1999

[35] A, D'Souza, J. Rickel, B. Herreros, and W. L. Johnson, "An Automated Lab Instructor for Simulated Science Experiments", In Proc. of Tenth International Conference on AI in Education, pp. 65-76, IOS Press, May 2001

[36] Alexander I., "Why neural computing? A personal view", In Journal of Information Technology, Vol 4, No 2, June 1989, pp 108-111

[37] Anil K. Jain and Jianchang Mao, "Artificial Neural Networks: A Tutorial," IEEE Computer, March 1996, pages 31-44

[38] Maren A. J., Harston C. T., Pap R. M., “The Handbook of Neural Computing Applications”, Academic Press, 1990.

[39] Hagan M. T., Bemurth H., Beale M., “Neural Network Design”, Pws Publishing Co., 1996

[40] Masters T., “Practical Neural Network Recipes in C++”, Academic Press, 1993

[41] Andrew P. Paplinski, “Lectur Notes on Artificial Neural Networks”, Monash University, Australia, 2002

[42] Diamantaras K. I., Kung S. Y., “Principal Component Neural Networks”, 1996

[43] Cichocki A., Unbehauen R., “Neural Networks for Optimisation and Signal Processing”, 1993

[44] R. Babuska, “Fuzzy Systems, Modeling and Identification”, Delft University of Technology, Department of Electrical Engineering Control Laboratory, 1998

[45] Alan Dix, Janet Finlay, Gregory Abowd, Russell Beale, “Human-Computer Interaction Second Edition”, PRENTICE HALL © 1998

[46] Ali O. Boyaci, Control of an Industrial Robot via Internet" Msc. Thesis Middle East Technical University, Mech. Eng. Dept., Ankara, Turkey September 2002.

[47] Patrick Ehlert, "Intelligent user interfaces: introduction and survey", Research Report DKS03-01 / ICE 01, Mediamatics / Data and Knowledge Systems group Department of Information Technology and Systems Delft University of Technology, The Netherlands, 2003

[48] F. Rodríguez, A. Khamis, M. Salichs, “A Remote Laboratory for Teaching Mobile Robotics”, IFAC-Conference on Telematics Applications and Robotics, Weungarten, Germany, 24-26 July (2001)

[49] S. Fukuda, T. Kikuchi, A. Fukuzaki, M. Ishii, K. Nagaoka, K. Tanaka, D. A Harris, "TMIT Remote Laboratory Experiment between Stanford University and NIME", Proc. of the Memoirs of Metropolitan Institute of Technology, No15, pp.165-170, Nov 2001

[50] Senese, F. A., Bender, C., “The Internet Chemistry Set: Web-based Remote Laboratories for Distance Education in Chemistry”, Proc. Ed-Media 2000, Montreal Canada

[51] M. Marone, "The Mercer Online Interactive Chaotic Pendulum", Computing in Science & Engineering, 4 (4), 94-c3 (2002)--on the web at http://physics.mercer.edu/PENDULUM/

[52] http://dms.rutgers.edu/irle

[53] http://olbers.kent.edu/alcomed/remote

[54] www.opticsforkids.org

[55] M.I. Godfrey, T. Kosa, L. Holmberg, P. Palffy-Muhoray, "Real Physics on line: A remote experiment with liquid crystals", APR98 Meeting of the American Physical Society, April 1998

[56] I Gustavsson, "Laboratory Experiments in Distance Learning", Proceedings of the ICEE 2001 Conference in Oslo, Norway, August 6 - 10, 2001

[57] S.K. Esche, "Remote Experimentation – One building Block in Online Engineering Education", in Proceedings of the 2002 ASEE/SEFI/TUB International Colloquium on Global Changes in Engineering Education, October 1-4, 2002, Berlin, Germany

[58] Panu Harmo, Aarne Halme, Hannu Pitkänen, Petri Virekoski, Matias Halinen, Jussi Suomela, “Moving Eye – Interactive Telepresence over Internet with a Ball Shaped Mobile Robot”, WS 2001, International Workshop on Tele-Education in Mechatronics Based on Virtual Laboratories, July 2001, Weingarten, Germany

[59] http://www-2.cs.cmu.edu/~illah/rix62.html

[60] H. E. Motuk, A. M. Erkmen, I. Erkmen, “Student Performance Evaluation in Web Based Access to Robot Supported Laboratories”, ICRA 2003, IEEE International Conference on Robotics and Automation, May 2003, Taiwan