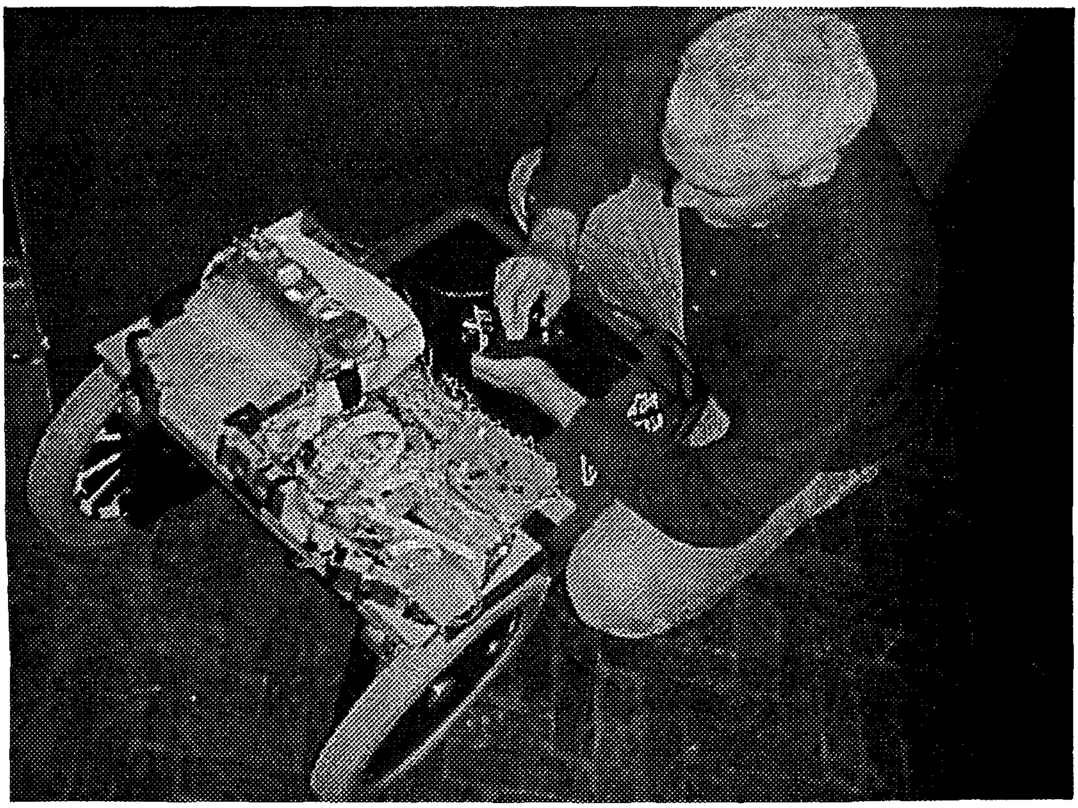
A self-contained, intelligent, micro-controller expert system to augment powered wheelchair users.

by

Ian James Stott



23 May 2002

A Dissertation submitted in partial fulfilment of the requirements of the award of the degree of Doctor of Philosophy of the University of Portsmouth following research conducted by the Author in the Faculty of Technology, University of Portsmouth.

The Author has not been a registered candidate for another award of a University during the research program.

UNIVERSITY OF PORTSMOUTH,
DEPARTMENT OF MECHANICAL AND
MANUFACTURING ENGINEERING.

A SELF-CONTAINED, INTELLIGENT, MICRO-CONTROLLER EXPERT SYSTEM TO AUGMENT POWERED WHEELCHAIR USERS.

by

Ian James Stott

23 May, 2002

A Dissertation submitted in partial fulfillment of the requirements for the award of the degree of Doctor of Philosophy of the University of Portsmouth following research conducted by the author in the Faculty of Technology, University of Portsmouth.

The author has not been a registered candidate for another award of a University during the research program.

ABSTRACT

An innovative prototype intelligent wheelchair was created to test novel algorithms to help disabled wheelchair users navigate through an unstructured environment. Local planning methods were selected to assist in steering. A simple, on-board expert system attempted to find a suitable trajectory that was close to the route requested but which moved away from close objects. The algorithms simulated an expert powered wheelchair driver. The algorithms were weighted by the wishes of the user. If the expert system detected that the user was unsure or inconsistent, the on-board expert system was given more importance in the selection of the wheelchair route. If the user had been consistent, then the wheelchair user was given more importance. The expert system could be over-ridden by the user if the user was consistent in the use of the joystick.

The prototype intelligent wheelchair could detect the environment, modify wheelchair control data and detect the wishes of the user. Decisions were made by the expert system and pre-planned responses could be activated.

Autonomous operation of the prototype intelligent wheelchair was demonstrated during tests. A micro-controller was embedded into the wheelchair control path. Information was read from the joystick and new sensor system and signals were sent to the wheelchair controller. The raw sensor data were processed to improve the reliability of the range data by mapping the sensor data onto a histogram certainty grid.

The intelligent wheelchair created during this work augmented the control that the disabled user of the wheelchair could provide. It was important that the expert system operated in real time in order to assist the user. There were two real time inputs; the user input and the sensors. The user indicated a speed and direction for the wheelchair. The sensors gathered information about the environment. A *sensor expert* system then analysed the sensor information and made a recommendation for a path that would prevent collisions. The data inputs sometimes conflicted. Another expert, called the *Fuzzy Mixer* considered both inputs and was responsible for the final outputs to the motor controller. The *joystick monitor* expert was responsible for interpreting the wishes of the user. Variables such as joystick position and consistency were examined by the *joystick monitor* to assess the desired wheelchair trajectory.

The prototype intelligent wheelchair was tested in the laboratory. It was self-contained which allowed realistic testing without the burden or restriction of trailing umbilical cables. Obstacles were placed in the path of the wheelchair and the response was encouraging.

ACKNOWLEDGEMENTS

The author wishes to express his gratitude towards the Dynamic Systems Engineering Research Group at the University of Portsmouth. Specifically to:

Dr David Sanders for his guidance and inspiration throughout the work.

Dr Giles Tewkesbury for his constructive advice and expert help on all matters relating to electronic engineering.

Mr Neal Hewer for expert assistance in 3D modeling and simulation using IGRIP.

Dr. Mike Goodwin for the endless arguments and dreams of bike rides.

Dr. Adam Hudson for the music, and the lessons on expert systems, even whilst suffering from self inflicted motor cycling injuries.

Mr Martin Langner for inspiration and understated excellence of his work.

Prof John Billingsley who started it all.

Also to	Dr Bing Luk	Mr. Ron Dadd
	Dr Althea de Souza.	Mr Alexander Lassauniere
	Mr YC Tan	Mrs Zainab Rasol
	Mr. Barnaby Perks	Mr. Richard Bullock

and
Staff and Students at the Chailey Heritage Special School in Sussex
and my family
Nicola, Thomas and Megan.

The work was supported by a grant from the Teaching Company Directorate in collaboration with QED Ltd.

The research was conducted at the University of Portsmouth, UK over the period January 1998 to August 2000.

A self-contained, intelligent, micro-controller expert system to augment powered wheelchair users.

By

Ian James Stott

Chapter One

Introduction.

The new work described in this Dissertation concerns the creation of a novel prototype wheelchair system. The prototype was designed and built to test new innovative algorithms to help a disabled wheelchair user to navigate a wheelchair through an environment such as a home. Local planning methods were selected to assist the operator in steering the vehicle. The algorithms attempted to find a suitable trajectory that were close to the vector requested by the operator but which moved away from close objects. The algorithms simulated an on-board expert powered-wheelchair driver and suggested a route that would prevent the wheelchair driving close to objects. The algorithms were weighted by the wishes of the user. If the system detected that the user was unsure or inconsistent, the on-board powered wheelchair driving expert system was given more importance in the selection of the wheelchair route. If the system detected that the user had been consistent in the joystick inputs, the wheelchair user was given more importance in the selection of the wheelchair path. The system could be manually over-ridden by the user if the user was consistent in the use of the joystick.

The new integrated wheelchair systems could detect the environment, modify wheelchair control data and detect the wishes of the user through the joystick. Decisions were made by the new systems and pre-planned responses could be activated.

The work completed during this research achieved fully autonomous operation of innovative wheelchair systems. New control circuits based on an ATMEL micro-controller were created as the final system and new prototype systems based around a Philips microcontroller were created during the development of the ideas. A serial interface, analogue to digital converter and new pulse width modulation interface controlled the wheelchair motors. A modified interface to the wheelchair controller allowed the new systems to communicate with the existing powered wheelchair controller. Software code was compiled, assembled and linked from ANSI C and assembly level functions to operate SPI, PWM interfaces, and create a novel expert system for wheelchair trajectory modifications based on a simplified Blackboard Architecture. Histogrammic maps were used to improve reliability of the sensor systems and access and record sensor confidence. VR simulations of the new algorithms were created within Superscape and IGRIP environments to test new models and algorithms. Models of joystick data, user intentions and local environment were created from sensor information. A novel expert system was implemented on a small low cost hardware platform with a minimised production rule set which avoided the need for elaborate, processor hungry algorithms.

A micro-controller based system was embedded into the wheelchair control system of a Bobcat II powered wheelchair. The new micro-controller system read information from the joystick and sensor system and sent signals to the wheelchair controller. The raw sensor data was processed to improve the reliability of the range data by mapping the sensor data onto a histogram certainty grid. The grid represented specific areas in front of the wheelchair-mounted sensor system.

Novel algorithms to provide control of the wheelchair were created to combine the signals from the joystick and sensor system. During normal operation the user had complete control of the wheelchair. When the sensors detected an obstacle in the path of the wheelchair, an expert system interpreted the joystick signals and the data from the sensor system. This new information was considered along with historical information from the joystick and sensors. The expert suggested a new trajectory for the wheelchair. The new trajectory could be over-ridden by the user as the joystick expert checked the joystick inputs for consistent contradiction of the suggested route set by the expert system.

The new assistive system created during this work augmented the user of the wheelchair. It was important that the system operated in real time in order to assist the user. There were two real time inputs to the system; the user input (a joystick for example) and the sensor system. The user indicated a speed and direction for the wheelchair. The sensor system gathered information about the environment. A sensor expert then analysed the sensor information and made a recommendation for a path that would prevent collisions. The data inputs were sometimes in conflict. Another expert, called the Fuzzy Mixer considered both inputs and was responsible for the final outputs to the motor controller. Joystick monitor was responsible for interpreting the wishes of the user. Variables such as joystick position and consistency were examined by the joystick monitor to assess the desired wheelchair trajectory.

The systems were tested in the laboratory. All components were mounted on the wheelchair chassis and the micro-controller was programmed through a serial cable. When this cable was removed, the complete system was a self-contained, independent vehicle capable of autonomous operation. The intelligent wheelchair system was free to roam during certain tests. This allowed realistic testing without the burden or restriction of trailing umbilical cables. Obstacles were placed in the path of the wheelchair and the response was satisfactory. The new work has been reported in several publications.

The Dissertation concludes with a description of future work aimed at improving the mixing and control algorithms and suggestions for new sensor arrangements.

1.1 Dissertation overview.

There have been many investigations into the application of technology to wheelchairs and Chapter 2 includes some general information on new intelligent systems to assist users of powered wheelchairs. Before looking for a solution to the question of how to help a powered wheelchair user, typical users were investigated. Some examples of medical case studies are described in Chapter 2 to explore the needs of some severely disabled powered wheelchair users. Where a user was not able to perform certain tasks associated with navigating a powered wheelchair, a degree of automation was possibly a solution to some of the problems encountered by the user. Robotic systems were investigated generally to discover the capabilities of automated and robotic systems. Later systems have used intelligence to improve effectiveness and have increased ability to react to unexpected inputs.

Powered wheelchairs are by definition, mobile devices. The process of getting from A to B via C has been widely investigated by Autonomous Guided Vehicle (AGV) researchers. AGVs were investigated for their ability to navigate on a specified route yet some have the delegated responsibility to avoid unforeseen obstacles in the path of the vehicle. Some of this technology was relevant to the novel and innovative work described in this Dissertation.

In order to change an existing or future powered wheelchair system from total reliance on the user or a carer for navigational input, a computer system of some sort was required. Some trends in control system technology have distributed the control system. If powered wheelchair control signals are to be detected, analysed and sometimes modified, similarities with new "Fly-By-Wire" (FBW) control systems were noted. Existing FBW systems were investigated and are described in Section 2.3.

In order for any mobile vehicle to navigate successfully in an unstructured and cluttered environment with a degree of independence, an ability to sense the environment was implied. Sensor systems were investigated and many different sensor systems compared in terms of suitability, accuracy, practicability and cost. Notably in Section 2.4, ultrasonic, vision and laser systems were considered and advantages and disadvantages to the powered wheelchair application documented.

Many sensor systems were considered, including the human sensor system in the form of the user. For efficient use of information from different sensors, the information gathered from one system needed to be combined with information from another. This led the general investigation to sensor fusion techniques which are discussed in Section 2.5.

A major difference between the research described in this Dissertation and AGVs is the fact that a human operator will be present within the control loop of the powered wheelchair. Section 2.6 discusses the effect on the design of systems which have a human operator, and an assistive control system working together.

Many systems use controllers with built-in intelligence. Artificial Intelligence and Fuzzy logic systems are discussed in Section 2.7 and ideas are presented on how these techniques may be useful to the work described in this Dissertation.

An Expert powered wheelchair driver can drive a powered wheelchair accurately and safely under normal circumstances. Where the user cannot apply these "expert" standards of driving to a powered wheelchair through disability, a possible solution may be to equip the powered wheelchair with an expert driving system of it's own. Expert systems and their user interfacing are discussed in Section 2.8. Some notes on implementation of an expert system are included and methods of representing expert knowledge are presented.

Some other advanced wheelchair systems are discussed in Section 2.9. Their advantages and suitability are examined. Practicality, cost and functionality are considered.

Section 2.10 examines some current methods of training potential powered wheelchair users. Methods range from the user simply exploring the wheelchair system with some supervision to structured approaches with innovative equipment. Innovative methods for potential powered wheelchair users are proposed and discussed later in the Section.

Chapter 2 concludes with a general discussion on current and the possible future development of powered wheelchair systems.

Powered wheelchairs have normally been presented to a first time user in a large uncluttered environment so that the user could explore the capabilities of the wheelchair without the danger of colliding with other objects. VR has been investigated to explore alternative and safer methods for training. Chapter 3 describes how Virtual Reality (VR) worlds were generated and viewed in real time in order to provide environments where a powered wheelchair could be simulated and new algorithms could be tested in safety.

Savings can be made if the new user does not travel to training centers and a more manageable training regime of regular but relatively short training sessions may be easier. VR systems are non-contact training systems and the use of VR in powered wheelchair training systems is discussed in Section 3.1

The immersion of the user in the virtual world depended on the type of user interface available. Section 3.2 presents some of the ways that a user can be interfaced to the virtual world and the method used in the work described in this Dissertation.

A simulation of a new powered wheelchair system and its sensor system is described in Section 3.3. Realistic visualisations of a powered wheelchair were created, initially in Superscape VRT for the first prototype system as described in Section 3.4 and then in IGRIP for the second prototype as described in Section 3.5. Simple algorithms were created and implemented on the VR simulations to test them before using them on advanced wheelchairs.

A modern powered wheelchair called a Bobcat II was selected as a test rig for this work and is described in Chapter 4. The wheelchair was modified so that new prototype equipment could be attached to test the new systems being created. The hardware consisted of a new sensor system and micro-controller interfaces. Strategies for detecting objects are discussed in Section 4.5 and experiments were conducted to test the effect on the accuracy and effectiveness of the sensor system with the sensor transducers mounted in different configurations. The experiments are described in Section 4.6.

The micro-controlled systems are described in Section 4.7. Two prototype systems were created, based on a Philips and an Atmel derived micro-controller. The new systems allowed autonomous operation; previous wheelchair systems created at Portsmouth were tied to a desk top computer system by an umbilical cable.

Sections 4.8, 4.9, and 4.10 describe the peripheral hardware devices and describe the serial interface between the analogue to digital converter and the micro-controller, the motor speed control interface, and the joystick interface respectively.

Chapter 4 finally describes the prototype wheelchair system hardware that was created as a test rig for the new work described in this Dissertation.

Chapter 5 describes the system software created during this work to test ideas and algorithms. Initial experimental systems were first constructed and used as a foundation for later work. The initial experimental systems and the prototypes that resulted from this initial work are described.

The second set of prototypes consisted of an ATMEL based microcontroller and peripheral devices to interface the micro-controller to the powered wheelchair. An integrated programming environment (IAR) was used to program the microcontroller. Programming was completed in a mixture of American National Standards Institute C (ANSI C), and assembler code. The IAR compiled, assembled and linked the code, creating a file which could then be loaded into the ATMEL micro-controller through in system programming port. This meant that the micro-controller could be reprogrammed repeatedly whilst it was still fitted into the prototype circuit board. This was more convenient than the first prototype which necessitated the micro-controller be removed from the circuit board before the code could be erased from the EPROM and "reblown".

Section 5.3 describes the sensor system hardware in detail and the micro-controller code required to operate them is described in Section 5.3.1. Some different strategies for sensor operation are described in Section 5.3.2.

Ultrasonic range finders are inherently noisy and suffer from random mis-reads, missed detections and inaccuracies due to specula reflections. Many techniques are documented to overcome these drawbacks. The method used in the work described in this Dissertation is described in Section 5.4. Many sensor operations were conducted, and the results of all of them combined in a histogram which tended to minimise the effect of the random errors and maximise the consistent data. As the range finders were usually correct, the correct readings reinforced themselves and the random errors tended to decay away as background noise.

The micro-controller used in this work was fitted with dual pulse width modulation (PWM) outputs. If smoothed, PWM can be useful to provide a dc voltage that can be easily varied by an algorithm built into the micro-controller code. The dual PWM outputs were used to control the wheelchair motors by mixing the joystick voltage with a voltage generated by the PWMs through the PWM interface.

The analogue to digital converter (ADC) was connected to the micro-controller through a serial interface called the Serial Peripheral interface (SPI) as described in Chapter 4. The SPI was controlled by the micro-controller.

The joystick fitted to the standard Bobcat II powered wheelchair was retained for use on the new prototype systems created during this work. The joystick was modeled so that it could be represented inside the micro-controller. The joystick representation and the code to convert the joystick data to the modeled data is described in Section 5.7. The conversion algorithm firstly converted the joystick data from Cartesian co-ordinates to a polar form, then allocated the joystick position to a simplified table to represent the intended direction that the wheelchair driver wished to move in.

The new wheelchair systems could detect the environment, modify joystick data and detect the wishes of the user. It was necessary for the system to interpret all this information in order to assist the user, the expert knowledge of the process or problem had to be acquired. An expert system was created to assist the user and this is described in Chapter 5.

Knowledge acquisition is the process of extracting knowledge from the expert sources. This knowledge could be organised and structured so that it could be used by the problem solving mechanisms of an expert system. This process is described in Section 5.8.1.

Once the Knowledge had been acquired and represented in the micro-controller, the system needed to act on the information. An example expert system is described where a simple rule based program was written in QuickBasic. A "Number puzzle game", (similar to the "sliding tile game familiar to many children) solving program was written to test AI program techniques. The program used a "hill climbing" algorithm to find a solution. A pre-programmed puzzle matrix was included, along with a target array. As the program was stepped towards a solution, the space was "moved" within the matrix under a set of rules until the example matrix matched the target matrix.

The first prototype intelligent wheelchair Expert System is described. This was the first step in creating a new system to augment the user of the new intelligent wheelchair. It was important that the system operated in real time in order to assist the user. There were two real time inputs to the system; the users input (a joystick for example) and the sensor system. The first prototype algorithms to assist a user of a powered wheelchair are described in Section 5.9. Algorithms were created to mix data from the joystick and the new sensor system. Algorithms were also created to interpret the information from the sensor system and the joystick. The interpretation of the data was important as it influenced the way in which the novel systems were able to react to the changing circumstances of the wheelchair system and needs of the user.

Section 5.9.1 describes the relationship between the sensor system and the joystick; Section 5.9.2 describes the mixer (known as the fuzzy mixer) that was constantly assessing the two inputs to the system. Control algorithms controlled the task of apportioning control between the two inputs.

Many conditions that would necessitate a user having to use a powered wheelchair can result in the user being unable to provide steady inputs to the joystick. Inaccurate or unsteady inputs could cause a control system to react unpredictably to the user input. It was important to build an algorithm into the system that would filter out and be tolerant to in-consistent and unsteady joystick inputs. An algorithm was used to assess the joystick consistency. The average joystick position was calculated in real time and a smoothed joystick voltage waveform was created. The average smoothed voltage was compared to the instantaneous value and an assessment of the actual joystick position estimated. The consistency was used as an indication of the confidence in the joystick position that the new systems had in the position of the joystick and the belief that the use wished to move in that direction.

The Sensor expert applied a set of algorithms to the information generated from the two sources of sensor data.

Section 5.10 describes the tests conducted on the first prototype expert system. The wheelchair system was tested by driving the wheelchair in an unstructured but uncluttered environment. The response of the wheelchair system was slow and inflexible. Parts of the system were redundant and were removed for the second prototype system which is described in Section 5.11.

The second prototype system was created to simplify some of the processes. Code was written in modules and they are described. A module was written for each situation that the new wheelchair system might encounter. Decisions were made by the system and a pre-planned response activated. The novel expert systems continued to re-assess and monitor the sensor inputs and changed the system responses appropriately.

The second prototype system was tested by driving the wheelchair in an unstructured but uncluttered environment. The response of the wheelchair system was fast enough for the wheelchair to navigate itself along a corridor and align itself with a doorway with the joystick held in the forward position. The path that the wheelchair took indicated that the sensor expert was recommending trajectory changes to the wheelchair controller. Section 5.12 describes the tests.

In the conclusion of Chapter 5 the adoption of a simplified Blackboard framework for the creation of the prototype is discussed. When operating a joystick controlled vehicle, users tend to use large deflections of the joystick to maneuver the vehicle. The dynamics of the wheelchair controller and the physical dynamics of the wheelchair make the application of large deflections of the joystick a suitable method for accurate control of the wheelchair. Small deflections caused a sluggish reaction from the system or the inputs were ignored completely by the system. Large changes in the controller input voltages caused smooth changes to be made to the wheelchair trajectory.

The results of the prototype tests are presented in Chapter 6. Chapter 6 begins with a general description of the new systems. Section 6.1 presents the results of the Hardware tests including the results of the joystick analysis and shows how the joystick could be read by the micro-controller system. Data for controlling the wheelchair motors using the micro-controller system is presented.

The sensor system was tested with the sensor transducers mounted as single pairs. These tests allowed the beam pattern of the sensor pairs to be plotted. The work was conducted inside an anechoic chamber to reduce the influence of external noise and reflections. Results of the single sensor pair tests are presented. The sensors were also tested with more than one sensor pair mounted in an array. The arrays and strategies used in this work are presented in along with the results of the four-sensor array and the results of the three sensor pair tests.

The Virtual reality systems were tested and the results of the Superscape and IGRIP based prototype systems are presented in Section 6.2.

The expert systems and controller code were tested as described in Section 6.3. The software was written in modules and applied to the hardware as the wheelchair was progressively modified. This procedure ensured the software modules could be tested in isolation before they were integrated with the rest of the system.

The joystick, controller data and sensor system provided data in the form of an analogue signal. The joystick had two analogue channels and these were read during each cycle of the main program. The joystick needed only one analogue to digital conversion per channel to provide an indication of the joystick position. The sensor data consisted of an analogue signal from the receiver. The receiver signal was sampled and analyzed in real time by the micro-controller code.

Depending on the sensor configuration used, three or four bits of an eight-bit input/output (IO) port of the micro-controller were used to control the operation of the transmitters. Section 6.3.2 discusses the response of the new system to the new code to operate the digital outputs.

Three sets of data were acquired from the wheelchair system. These were the data from the joystick, the voltages of the inputs to the wheelchair controller and the sensor data. These data were acquired by the novel expert system and successfully interpreted and analyzed. This resulted in successful wheelchair test runs where the wheelchair trajectory was modified by the wheelchair systems and the wheelchair was guided around obstacles in its path.

The intelligent wheelchair system was tested in the laboratory in a variety of environments and in a corridor. Examples of test runs and the reactions of the wheelchair systems are presented.

The intelligent wheelchair system made numerous test runs along a corridor and was proficient at hunting for the middle of the corridor. A wheelchair user who was proficient at driving a powered wheelchair along a corridor in a straight line would probably not benefit from a navigational assistance system of this type. This system was aimed at the severely disabled who have little fine motor control skills or spatial awareness.

A general discussion of the new system tests is presented in Section 6.5. The wheelchair system was tested in controlled conditions in and outside of the laboratory, but always indoors. Specific wheelchair maneuvers were used as complex examples for the tests.

Chapter 7 is a discussion of the whole project. The purpose of the work is introduced in Section 7.1. The achievements are summarised in Section 7.2 with conclusions and recommendations for future work made in Section 7.3.

1.2 Novel discoveries and innovations made during this work.

- A novel expert system was created to modify the wheelchair trajectory based on a simplified Blackboard Architecture.

- Novel expert systems can be implemented on simple low cost hardware platform using a minimal production rule set.
- Simple but innovative algorithms were effective for controlling an assistive wheelchair system.
- Innovative histogrammic maps were used to generate confidence in sensor and user data.
- Innovative training methods and simulations of novel algorithms were created using virtual reality in Superscape and IGRIP.

1.3 First named publications resulting from this work.

- STOTT IJ (1998). "**A review of advanced and intelligent wheelchair research**". Intelligent Mobility,8, published by PCR, pp 4-12, ISBN: 0-9527524-1-7.
- STOTT IJ & SANDERS DA (1996). "**Modification of a Wheelchair Trajectory by the Integration of Ultrasonic Sensor Data**". Proc. of 1st Symposium on Powered Wheelchairs for Disabled Persons, Uni of Portsmouth, published by PCR, pp 4-7. ISBN 0-9527524-0-9
- STOTT IJ & SANDERS DA (1998). "**Advanced intelligent wheelchairs**". Intelligent Mobility, 5, published by PCR, pp 23-31, ISBN: 0-9527524-1-7.
- STOTT IJ, SANDERS DA AND GOODWIN MJ, (1997). "**A software algorithm for the intelligent mixing of inputs to a tele-operated vehicle**". Journal of Systems Architecture, Vol 43, No 1, pp 67 - 72.
- STOTT IJ & SANDERS DA, (2000a), "**The use of virtual reality to train powered wheelchair users and test new wheelchair systems**" International Journal of Rehabilitation Research, Vol 23 (issue 4), p. 321-326
- STOTT I.J & SANDERS D.A. (2000b), "**New powered wheelchair system for the rehabilitation of some severely disabled users**". The International Journal Of Rehabilitation Research, Vol. 23(3) ISSN: 0342-5282..
- STOTT I.J., SANDERS D.A. & TEWKESBURYG.E. (2000). "**Low cost ultrasonic sensors for tele-operatd vehicles**". Sensor Review: the international journal of sensing. Vol.20, no.3, pp227-234, ISSN: 0260-2288.

Chapter Two

LITERATURE SURVEY AND BACKGROUND TO THE RESEARCH

There have been many investigations into the application of high technology to wheelchairs [Beattie(1993), Bell *et al.* (1993 & 1994), Bourhis *et al.*(1993), Nisbet(1995), Nisbet *et al.* (1988), Stott (1998), Sanders *et al.*(2000) and Sanders and Stott(1999).]. Reviews of research are included in Stott(1997 and 1998) and Stott and Sanders(1998) Some investigations have applied AGV technology directly to a wheelchair application [Rao and Kuc(1989)]; other investigations have used adapted AGV technology or attempted to create new algorithms and systems to assist a wheelchair user [Pruski and Bourhis(1992)].

The technology to drive automated platforms around in structured environments such as warehouses or factories has been available for many years. Automated Guided Vehicles (AGVs) transport goods and raw materials from place to place within predetermined and controlled areas. Some AGVs depend on hidden tracks, often inductive loops or ferrite markers buried in the floor of a factory [Wakaumi *et al*(1989)]. The inductive guidance carrier frequency can be varied to indicate different routes [The FMS Magazine(1987)]. It is possible to change the path of some vehicles by sending radio instructions [Hartley(1987)]. Visible tracks painted or built into the floor of a factory have been practical alternatives to sinking cables into the floor [Hartley(1987) & Goodwin(1992, 1997a & 1997b)]. Marking the floor has been suitable for 'clean' environments where the floor does not become too obscured. Bostel and Sagarm(1996) described how AGVs could use static beacons or "bar codes" placed at strategic positions along a route. Between these positional fixing points the vehicle could navigate using odometric sensors mounted on the wheels. It was possible for more advanced AGVs to navigate around temporary obstructions in the path of the AGV or to take different routes to avoid other AGVs working in the same areas. To achieve a local avoidance maneuver an AGV has been capable of detecting the environment. These AGV systems completed maneuvers without causing other problems, such as crashing or straying into dangerous or undesirable areas.

Sonar sensors using the time of flight principle to measure range have been widely used on sensor systems for autonomous vehicles. Jörg and Berg(1996) discussed such systems and how they could be prone to misreading caused by the properties of the sensor system. Hinkel *et al.*(1988) described how some vision-based systems used a stereoscopic view to detect obstacles but these systems could require time-consuming data processing. As an alternative a laser range finder could provide accurate and fast range information but these systems can be prohibitively expensive [Sanders(1993)].

The new research work described in this Dissertation used ultrasonic time of flight sensors. Walls, boxes and cabinets were easier to detect due to their large reflecting surfaces, although these could also become invisible to detectors if the energy was not reflected back to the receiver. Office furniture such as tables and chairs were difficult to detect if the sensors were fitted at leg height. The narrow legs of chairs and tables were generally good reflectors of energy but because of their small physical size and rounded section, little of that energy may be returned to the receiver.

Automatic wheelchair planning algorithms must co-ordinate the essential aspects of the problem:

a) Detecting obstacles.

b) Appropriate representation of the wheelchair and obstacles.

c) Derivation of a suitable path and trajectory for the wheelchair through the obstacles.

Pieper(1969) investigated automatic programming for robots in the United States. Later work to design a robot for use in planetary exploration was completed by Udupa(1977) at CalTec. Since then some major contributions in the field of robot path planning have come from Lozano-Pérez(1990) and Brooks(1983a-c, 1986 & 1991b), Brooks and Lozano-Pérez(1983), Gouzenes(1984) and Donald(1984). These both used polyhedral models to represent obstacles. Donald(1987), Gouzenes(1984) and Canny(1985, 1986 & 1987) have extended this work and, further work has been done by Tseng(1987), Hwang(1988) and Sanders(1993 & 1998). De Pennington *et al*(1983) reported an alternative approach where the mover was modeled by a series of interconnected spheres. Since then other authors have added to this research.

This research into wheelchair technology, AGVs and robotics formed the starting point for the research work in this Dissertation. The remainder of this Chapter describes the literature considered during the PhD research.

2.1 Medical case studies.

Studies of the behavior of children with motor disabilities showed them to be poorly motivated and dependent on their carers [Butler(1986)]. They did not possess a curiosity about their environment that a child would normally exhibit. This was the case even when their disability should not have interfered with their potential performance. Butler described how young children who did not develop the ability to move independently by walking or crawling within the first two years of life were deprived of an important stage of development referred to as the "contingency experience"

Butler(1984) used specially constructed miniature powered vehicles for her work with young disabled children. Some improvements in behavior and motivation were noticed during the work. Powered wheelchairs that are easy to use and safe could improve the chances of a severely disabled child achieving the contingency experience. The research described in this Dissertation created a novel and innovative system to help achieve this.

The contingency experience may not be experienced for two important reasons: motor disability and low expectations of parents and carers [Langner(2000)]. Deprivation of contingency experience may lead to 'learned helplessness' a condition in which a child has given up trying to control his world [Verburg *et al*(1984) and Stott(1997)].

The failure to experience the contingency experience could significantly increase the time required to acquire the skill necessary to drive a powered wheelchair. Some of the problems experienced when teaching a new user to drive a powered wheelchair are described in Chase and Bailey(1990). "Chris" a 38 year-old spastic quadriplegia patient had lived in a state school for persons with mental retardation for approximately 21 years. In 1983, she was selected as a possible candidate for a powered wheelchair and the 5-year process involved in training to use a powered wheelchair is described.

Douglas & Ryan(1987) described teaching a pre-school child to steer a powered wheelchair via a mouth-operated joystick. Some problems were encountered during learning, but the positive effect on emotional, intellectual and behavioral development were noted.

The process involved with teaching a person to drive a powered wheelchair can be long and difficult for the teacher and the person learning. The length of time required for some potential users to learn even the basic skill can cause the cost to be large. A project at the CALL centre in Edinburgh has used a SMART wheelchair [Nesbit(1995 & 2000)] to teach potential users some of the basic skills of wheelchair driving [Craig *et al*(1995)]. The response of the wheelchair to stimuli from the user was pre-programmed to suit the learning level of the user. As the user gained in skill, the responses of the wheelchair system could be changed to encourage development of the user. Langner(1995 and 2000) used a track system at a special school to encourage self-initiated mobility. Some children found the system beneficial as it allowed them the freedom to move unaided from classrooms to the toilets and dining areas. The system has been in use for the past five years and has been expanded to include the majority of the school including the living areas.

A discussion of the need for and the effect of smart wheelchair technology is presented in Nelson *et al.*(1990). Nelson calls for the sympathetic application of technology to wheelchairs. He points out that a wheelchair that does all the work for a disabled user will not force the user to improve and adapt a point also made by Goodwin and Sanders(2000). Also the safety aspect of the technology is discussed. It would be an interesting project to decide what would be appropriate to apply to each individual person. The range in abilities that would be encountered suggests that a different approach may be required for each one. This does not suggest that an intelligent wheelchair assistive system could not be generic and programmed to suit each individual.

Devito(1996) proposed a wheelchair system for integration into a nursing home. An interesting argument was that the elderly (whom this wheelchair system was aimed at) were probably motor car users. A smart wheelchair system would probably cost no more than a second hand car and could provide them with a higher standard of living. The proposed system used Intelligent Vehicle Highway System technology to navigate through a few known routes within the nursing home.

2.2 Robots.

Galt(1999) reported that ethical questions have arisen concerning the future development of robotic machines, in particular the relationship between robots and humans. Serious concerns regarding increased unemployment statistics have fuelled the debate whether automation is beneficial to mankind. Many people are simply scared by the thought of the inevitable loss of control that is a consequence of automation. Some experts have even forecasted that machines would rise above humans as the most dominant "species" on the earth [Warwick(1997)]. However, at the turn of the millennium intelligent life on earth will be able to relax in the knowledge that there will be two limiting factors impeding the further development of such "super-robots"; machine intelligence and robot mechanics. The state of robot technology can be put into perspective if you consider that scientists would be delighted if they were able to create a robot as intelligent or as mobile as the common garden ant.

A majority of 20th century robots are stationary or, have at most limited motion along fixed guideways in one or two directions [Galt(1999)]. A vast majority of commercially used robots have very little or no intelligence whatsoever: rather they perform a pre-programmed set of tasks. Thus, any significant advances in current robotics technology in the 21st century must stem from the development of both machine intelligence and mobility

The majority of robots in use throughout the world are not mobile at all in the sense that this research was interested. One of the most common types of robot is the fixed-base manipulator arm used in industrial production lines Sanders(1993). These robot "arms" are generally only mobile in the extension of the joints and as such have tightly controlled, well-defined workspaces. A mobile robot does not necessarily entertain this luxury.

The task of developing a mobile robot becomes evidently more difficult when it is considered that :

- The environments in which the vehicle may be required to operate will often be geometrically unstructured, cluttered with obstacles and difficult to model or predict.
- The complex nature of the operating environments may provide uncontrollable factors that cause malfunction or damage.
- The facilities that a mobile robot has for sensing the environment are often incapable of achieving high accuracy and can give only approximate indications of the environmental structure. Also, without the facility for absolute calibration, these sensing errors will accumulate over prolonged periods.

2.2.1 Intelligence in Robots.

Robotic applications developed to operate in industrial environments have tended to be specialised and require highly structured environments. Any change in the environment, or to the specific task to be undertaken has tended to require off-line re-programming of the robots. [Lozano-Perez,(1990), Goodwin,(1992), Sanders(1993) and Tewkesbury(1994)].

Brooks(1991a) described how work in Artificial Intelligence has had a strong influence on aspects of computer architectures. The Von Neumann model of computation has influenced the application and structure of Artificial Intelligence systems. However, intelligence in biological systems is different. Work in behavior-based Artificial Intelligence has produced new models of intelligence that are closer in spirit to biological systems. The non-Von Neumann computational models they use share characteristics with biological computation. The ability to apply past experience to new but similar situations are considered to be an indication of intelligence.

Kezhong *et al.*(1996) introduced the intelligent mobile robot key technique for an intelligent mobile robot. The robot exhibited path planning and simulation, transfer and multi-sensor fusion techniques. The design and implementation of information-reduced techniques showed how a system could be flexible in its interaction with the environment. Research into "perception-action" behavior and the application of fuzzy control to the robot controller are described.

Galt(1999) considered the most productive mobile robots to be predominately wheeled or tracked vehicles. That is they all have wheels and some have the addition of their own "rolling road". The majority of basic wheeled vehicles are confined to environments with the ideal walking surface, for example, flat, smooth and no stairs or other major obstacles. Yet, there are many of this type of mobile robot employed in places such as factories, offices, hospitals, airports and even theme parks.

Galt(1999) reported that the most versatile mobile robots employ sets of tracks for locomotion. Tracked vehicles have had much success on difficult terrain that may be a problem to conventional wheeled vehicles. A good example of the latest tracked wheel technology can be found on one of the most capable of terrain locomotion devices in operation today, the Merlin robot shown in figure 2.1. For any mobile robot system that was autonomous, it was important for the system to be sufficiently competent to perform self-location. This was particularly important if the robot became disturbed by outside stimuli or slippage in odometers. Moshe *et al.*(1997) described techniques for sensor fusion in robot navigation. A test robot (LIAS) was fitted with a suite of sensors. Algorithms for self-location were tested on the robot. The sensor suite consisted of ultrasonic sensors and a rotating infrared (I.R.) sensor. Low level fusion was used for direct integration of sensory data resulting in parameter and state estimates. High level fusion was used for indirect integration through command arbitration and integration of control signals suggested by different modules. Other work on sensor fusion can be found in Fu *et al*(1987) and Sanders(1993).

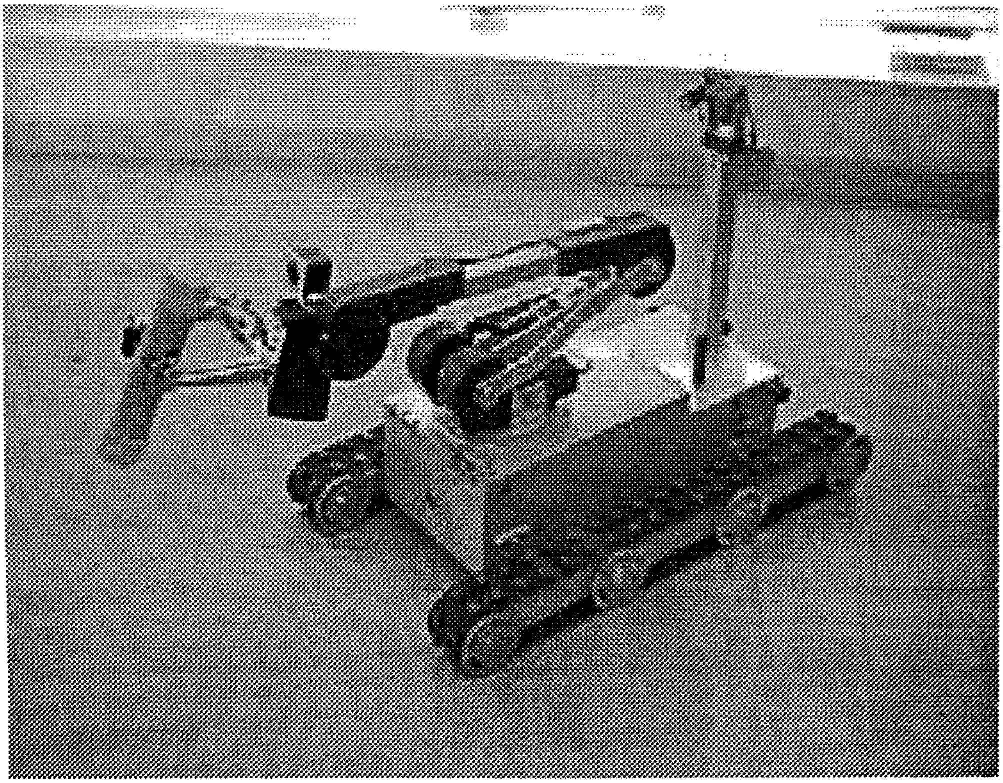


Figure 2.1 The Merlin Robot. Reproduced from Galt(1999).

Numerical systems that use recursive techniques can be suitable for implementation on computers and micro-controllers. Mutambara(1996) described a technique called “general recursive estimation” to systematically derive a solution to a sensor fusion problem. The Kalman filter was discussed as a suitable method for the reduction of sensor inaccuracies. Mutembara presented The “Information Filter” and the “Extended Information Filter” which could form the basis for decentralised data fusion techniques that could be applied to a modular wheeled mobile robot.

More information about Industrial Robots can be found in [Schilling(1990)]. Other definitions are included in McCloy and Harris(1986), Appleton and Williams(1987), Fu *et al*(1987), McKerrow(1991) and Sanders(1993).

2.2.2 Autonomous Vehicles.

The application of mobile robots to structured environments is well documented. Typically, Autonomous Guided Vehicles (AGVs) operating in factories or warehouses were designed to transport goods or raw materials around the production areas. Many are described in Stott(1997). Some more sophisticated vehicles, for example the system described by Hartley(1987) were able to leave a pre-set path and navigate a different route around an obstruction. It was common though for the systems to have limited autonomous capability on the grounds of cost and safety.

In 1998, a robot named Nomad was deployed into the Patriot Hills region of Antarctica. As one of the harshest environments on Earth, Antarctica was described as "a unique place to test planetary robotic technologies". Antarctica contains a substantial concentration of meteorites which have been preserved in the ice. Nomad was created to search for meteors (or rocks) in the ice. Nomad drove 10.3km autonomously under a variety of weather and terrain conditions. The robot was designed as an independent, isolated autonomous vehicle with little chance of operator intervention. The ability of a stereo vision system and laser scanner to perceive polar terrain and the autonomous navigation system are described in Moorhead *et al*(1999). Navigation was assisted using landmark-based navigation and millimetre wave radar.

Space exploration has been a testing environment for robot probes operating autonomously in unstructured environments. A recent NASA mission to Mars involved the operation of the Mars Pathfinder Rover or Microrover spacecraft. Stone(1996) described the Microrover as a small spacecraft in its own right even though it was part of the mission payload. It was capable of all the functions associated with a spacecraft; navigation, communication, power generation and instrument, sensor and actuator control. The Microrover was deployed from the mission vehicle into the surface of Mars and was then employed on experiments and tasks that involved moving across the surface of Mars.

The Microrover was a six wheeled, all wheel drive mobile robot [Matijevic(1996)]. It was equipped with many sensors to assist the vehicle to safely negotiate the Martian landscape. The sensors included accelerometers, bumpers, laser proximity sensors, inclinometers and potentiometers. An operator in a control centre on Earth controlled the Microrover. However because of the communication delay caused by the distance from Earth to Mars, the autonomous system was required to deal with any unexpected hazards independently of the operator [Stone(1996) & Morrison and Nguyen(1996)]. The rover uplink team generated sequences of commands for the rover to execute. The operator wore special battery powered goggles that allowed him to see the a 3D image of the Microrover on Mars. A unique joystick called a Spaceball was used to move a model of the rover on the screen so that the rover model looked just like it would on Mars. The Microrover control system continuously calculated the co-ordinates of the rover model and this is used to tell the rover where to go.

The rover control workstation provided a "virtual reality" type interface to view the surface of Mars from any location and angle around the lander and to create rover waypoints or goals. This allowed the driver to make decisions about the traversability of the terrain and to watch out for hazards that the rover should avoid.

Many mobile robots are wheeled vehicles, however mobile robots are not exclusively wheeled. Walking robots have been investigated as an alternative. A walking robot can have advantages over a wheeled robot especially in unstructured and rugged terrain.

Several articulated legged vehicles have been built and the potential of these mechanisms for negotiating rough terrain have been described by Kumar & Waldron(1988), however none of these vehicles could climb vertical structures.

Climbing vehicles have been developed for the nuclear industry, notably these included the NERO and MAVIS vehicles as described by Burrows *et al*(1991) and Luk *et al*(1994), but these vehicles were non-articulated and were designed to operate in a particular mode. They were not suitable for unstructured environments.

At the University of Portsmouth a team led by Professor Collie have developed several remotely operated articulated-legged climbing vehicles principally for nuclear environments [Beer(1994)]. The vehicles used for this research work were generic in concept; they adopted an entomological (insect like) structure. The hardware and software systems created were modular by design to allow for rapid development and / or modification. An essential feature of these vehicles was the use of pneumatics for limb actuation, providing a high power to weight ratio and allowing the vehicles to carry a payload equal to that of their own weight. A unique control technique provided a complaint force / position servo control at the limb level and a task orientated control system at the operator level [Luk *et al*(1991)]. With an articulated thorax [Collie *et al*(1993)] RobugIIs was capable of performing autonomous floor-wall transfers [Bevan *et al*(1994)] and climbing semi-structured vertical surfaces, for example the side of a ships hull.

Complex tasks contain unpredictable events, changing environments, and systems that are difficult to model. The environment in which a robot is required to work necessitates the need for intelligent control. Many of the assumptions made by control theory need to be questioned if robots are to handle future tasks. Systems that are subject to multiple purposes and abstract goals cannot be handled by current methodologies [Franklin & Selfridge(1990)].

Hewer (1995) has described mobile articulated limbed vehicles as redundant, actuated closed-kinematic-chains. They experience a range of indeterminate contact forces and joint torques which are task dependent. Related problems have been studied [Kumar & Waldron(1988)] [Orin & Oh(1981)]. Only a few authors have examined planning motions requiring large forces over substantial ranges of motion [Madhani & Dubowsky(1992), Cooke(1999) & Hewer *et al(*1994)].

Previous work at the University of Portsmouth has investigated systems that would provide reflex control for tele-robotic vehicles [Luk *et al*(1993), Collie *et al*(1993) Beer(1994), and Hewer(1995)].

2.3 Fly-By-Wire.

The new wheelchair systems created during this research involved placing a computer between the human operator and the wheelchair, This made the system Fly-By-Wire (FBW). This is a term which loosely describes a system where control data is passed around a system using electronic signals by wire. Machine Design(1992) gives a brief general introduction to FBW technology. The technology has been applied to aircraft, notably the Airbus aircraft. Learmount(1991) examined two Airbus crashes that cast doubt over the reliability of the FBW systems in aircraft. The reports are interesting because they may indicate the over reliance of the pilots in the technology.

Hammond *et al*(1989) describes the Electronic Flight Control System(EFCS) destined for use on Boeing aircraft. The use of multi-channel independent control systems are presented and the safety applications discussed. The entirely electronic FBW system will replace the mechanical cable/quadrant/pushrod system used on earlier aeroplanes. The FBW system, must meet high standards of integrity and reliability. The heart of the FBW concept is the use of redundant, dissimilar computing and communication channels.

As an alternative to FBW systems Fly-by-Light (FBL) technology is being developed for space and aerospace applications [NASA(1998)]. Optical components and subsystems will replace electronic data transmission lines, mechanical control linkages and electronic sensors. Utilising FBL and Power-by-Wire (PBW) technologies will provide lightweight, highly reliable, highly electro-magnetically immune fibre-optic control systems and all-electric secondary power systems for advanced subsonic civil transport aircraft.

There are many benefits of FBW systems [NASA Facts(1998)]. One of them is the technique of modifying the control data before the operation is carried out. In this way dangerous situations can be avoided. The Airbus for instance will not stall regardless of how far the stick is pulled back at low speed. The Airbus crashes of the early nineties were examples of how users of intelligent systems can form an over reliance on the technology. Both crashes seemed to have been caused by the pilot using the anti stall capability of the aircraft control system without paying attention to other aspects of flying the plane. A lesson is that technology should be introduced carefully to avoid over-reliance or testing of the whole system.

2.4 Sensors

To make decisions on how to move and where to move to, the wheelchair used in this research needed to be capable of detecting the environment. A sensor system for a powered wheelchair needed to be accurate, rugged, reliable, and suitable for attaching to a wheelchair and cheap. A human operator was the most accurate source of data about the environment but the accuracy of a human could be impaired by disability. In these cases a complementary sensor system was required to assist the operator.

Sonar sensors using the time of flight principal to measure range have been widely used on AGVs (Hinkel *et al*, 1988). Some vision-based systems have used a stereoscopic view to detect obstacles but these systems can require time-consuming data processing. As an alternative a laser range finder could provide accurate and fast range information [Gutmann & Schlegel,(1996)].

Sensor systems could be considered to be from the following categories;

- Contact sensors.
- Non-contact sensors.
- Human operator.

Contact sensors could be the simplest form of sensing. Contact bumpers were fitted to the SMART wheelchair [Nisbet *et al.*(1996) and Nisbet(2000)] and the chair described by Langner(2000). They were used to assist the new wheelchair user when the wheelchair came into contact with the environment. The contact bumpers could also be used to initiate automatic sequences for the wheelchair such as reverse and turn. Other contact sensing methods were used by Luk *et al*(1988) and Sanders *et al*(1987) who used the change in motor currents of a manipulator to detect contact with the environment.

Korzeniowski(1996) optimised the tracking of a surface by an end effector. As the environmental conditions changed the reliability of sensors could also change. The amount by which the reliance on sensor data must also change was discussed. Tracking a surface is an integral task for many autonomous systems. It could be used for navigation, surface preparation or object recognition. There are two types of control for surface following, continuous and discontinuous. A robot may maintain contact and continuously track the surface or touch the surface at discontinuous points.

In order that the whole process be optimised in terms of time to complete a task and the amount of data collected a balance was sought between the continuous and non-continuous methods. The tracking method was computed by the tracking algorithm using the partial data sets provided by sensors. It was common practice to outfit automated systems with the ability to gather data from many sensors. As the environmental conditions changed the system's reliance on sensor data also changed. This work focused on the addition of the supervisory learning module for choosing the method of surface tracking.

Non-contact sensors use a variety of techniques and mediums to sense the environment. Typically the non-contact sensors may be passive or active. Passive sensors are dependent upon the target providing an output that can be detected by a sensor; such as heat, light or sound. Passive sensors would not normally be useful to mobile vehicle applications as the sensor system could initiate a sensor sweep. Active sensor systems can initiate sensor sweeps of the environment on command of the control system. Data could therefore be provided to the control system when it was required, regardless of the target to be detected. Typically, a sensor system could use sound as in an ultrasonic sensor system or electromagnetic energy. Visible light was often used by sensor systems that used vision-based systems. Infrared could be used as a form of proximity sensing. Radio frequency (radar) could also be used along with scanning and rotating lasers. Some examples of suitable non-contact sensors are examined in this Section:

- Ultrasonic sensing

- Vision sensing
- Laser range finding

2.4.1 Ultrasonic sensing.

Ultrasonic range-finding and object-detection systems have used several principles, including: modulated waves [Teshigawara *et al*,(1989)], frequency sweeps [Lindstedt,(1996)] or echolocation [Stott & Sanders(1996) & Stott *et al*,(1997)]. Kay used a novel sensor array and constant transmission frequency modulation (CTFM) sweeps to assist blind people to detect their environment [Kay(1999a & b)]. A CTFM system examines the response of the environment to a frequency swept sine wave. The shape of the envelope returned to the receiver is determined by the properties of the environment. It is possible to recognise objects by the frequency spectrum of the receiver signal. McKerrow & Harper(1999) successfully differentiated between types of leafy plants with their system. More information on ultrasonic sensor systems can be found in [McCloy & Harris(1986), and Fu *et al*(1987)].

Single frequency pulse-echo systems can encode information on pulses in order to reduce interference between different sensor systems. Kleeman(1999) used a known separation between two pulses to identify different systems. The receiver signal was rapidly sampled and the separation read as each pulse train was received. Pulses from other sensors were rejected. Accurate angular resolution was also possible by using two receivers mounted close to the transmitter and measuring slight time delays between pulses.

Previous low cost solutions for navigational assistance for disabled wheelchair users have provided little more than simple obstacle and collision avoidance, or have followed a pre-defined fixed route defined by a white line or a buried wire such as early work by Langner(2000).

Other research has used complex high cost multi-sensor systems closely resembling industrial, military or space exploration applications. These advanced wheelchairs used natural features or artificial beacons to produce accurate maps of the operating environments. The progress of the vehicle is monitored and corrected using multi-sensor techniques such as vision cameras, odometry and triangulation from beacons located in the environment. Such systems have required modification of the operating environment and have resulted in a fully autonomous vehicle providing little or no overall control by the user. Whilst proving technical feasibility, their cost and complexity has not resulted in practical and affordable solutions for the wheelchair user.

The purpose of the present study was to bridge the gap between these two previous areas of research and to provide navigational assistance at an affordable cost, in the area described by Goodwin(1998) as maximum functionality at minimum cost. Low cost ultrasonic sensors enabled a wheelchair to operate in an unknown (previously unmapped) environment whilst leaving the user in overall control.

Hardware modifications to a commercial powered wheelchair enabled data from ultrasonic arrays and the user's joystick to be interrogated and mixed by a computer to provide appropriate signals for the wheelchair drive motors [Stott & Sanders (1996 & 2000) and Stott *et al.*(1997) & (2000)].

A simulation program was created to interpret the sensor signals that would be generated from the various conditions likely to be encountered by a wheelchair and to develop the various control strategies. The simulation was able to differentiate between the various environmental conditions and select the appropriate action using the newly created control algorithms.

The sensor data interpretation modules together with the control algorithms, from the simulation, were incorporated into a practical system for controlling the wheelchair [Stott & Sanders(2000)]. In tests, data from the sensors were used to detect and evaluate localised changes in the environment and used to determine appropriate signals for the drive wheel motors. It was found that the wheelchair controller and the geometry of the wheelchair resulted in a degradation of the expected wheelchair response. This was overcome in two ways: firstly by modifying the control algorithm and secondly by changing the wheelchair geometry.

Ultrasonic sensors were selected for the new work described in this Dissertation.

2.4.2 Vision Systems.

Vision systems usually involve a camera, a lighting system and a computer. The camera must be capable of providing images of sufficient quality for the detection algorithms to decode and recognise objects or features. There are at least two different ways to implement an attentive vision system, using either a bottom-up or a top-down approach [Garibotto(1999)]. In the case of a bottom up approach the low-level features extracted from the camera image are used to orient the system to a particular area of interest. In a top-down approach the visual recognition task is driven by a specific goal and is constrained within a set of processing algorithms. Garibotto referred to mail sorting machines, car-plate location and reading traffic control signals and signs and landmark identification in autonomous navigation systems. Garibotto suggests that a top-down scheme is more effective as more accurate control of the system is possible.

Another example of a top-down approach is in Chunrui-Zhang & Siyal(2000). Flexible image processing techniques were used for collecting and analysing road traffic data. A new technique, based on colour motion segmentation and split-merge segmentation approaches was used. The new colour mapping technique was compared to a more traditional greyscale technique whilst determining the rough position of moving vehicles in a sequence of images. The split-merge segmentation on the colour images saved the need to process the whole image, saving computation time. One of the drawbacks of vision based systems was addressed by this work in that the work required by the computer was reduced. The workload of the computer is still an important consideration for the designer of a sensor system which needs to work in real time.

Sebastian *et al*(1999) described a system for inspection of machine parts. The system consisted of a binocular stereo mount, a three degrees of freedom positioning device and a structured light system. The system was created to investigate automatic quality inspection. An interactive interface was proposed so that the user may define specific features for inspection. The system aimed to achieve the greatest possible accuracy and underlined the need for structured lighting and a controlled environment for accurate use of a vision system.

Vision systems are a suitable consideration when selecting a sensor system for a mobile vehicle. However the cost and complexity, and the need to control the environment can make them a less desirable choice. Other information on vision systems can be found in [Appleton and Williams(1987), McKerrow(1991) and Sanders(1993)].

2.4.3 Laser range finders.

Laser range finders can be effective and fast methods for measuring a distance to an object. Due to their fast response, they can be useful in real-time systems [Parnichkun & Samadi(1999)]. In their paper, a long-range laser range finder (LRF) was developed. They described a LRF as "a single point optical triangulation instrument, which detects distance information quickly and easily without touching the object being measured". Parnichkun & Samadi were mainly concerned with developing new methods to improve the range of an LRF. The range of the sensor system used on a mobile vehicle depended on the use for the vehicle and the speed at which it traveled. Not all LRF systems were designed for range finding. Jiminez *et al*(2000) described a laser-based computer vision system used on an automatic fruit recognition system. It was based on an infrared laser range-finder sensor that provided range and reflectance images and was designed to detect spherical objects in non-structured environments. Similar techniques could possibly be applied to navigational systems for the recognition of waypoints or landmarks.

A novel automated mine detector was described in Das *et al*(1999). A human operator was capable of sweeping a mine detector across rough ground without hitting it. The operator can follow the ground profile with the detector head close to the ground without hitting the ground or any objects on it. An early prototype of a mine detector was described which incorporated a metal detector for a mine sensor and ground following was assisted by an array consisting of a laser range finder and four ultrasonic sensors. Other information on laser range finders can be found in [Schilling(1990), Dodd & Rossol(1979) and Fu *et al.*(1987)].

2.5 Sensor Fusion

A single sensor system working in isolation provided limited information. Sensor arrays may be used to enhance data rate and accuracy of the system. Sensor fusion methods are described in Purcell & Huisoon(1993), Chang & Song(1996), Guey *et al*(1997), Kam *et al*(1997) and Sanders(1993).

Song and Chen(1996) described a navigation system that allowed a robot to travel in an environment about which it had no prior knowledge. Data from multiple ultrasonic range sensors were fused into a representation called Heuristic Asymmetric Mapping to deal with the problem of uncertainties in the raw sensory data caused mainly by the transducer's beam-opening angle and specular reflections. It featured a fast data-refresh rate to handle a dynamic environment. A potential field method was used for on-line path planning based on the constructed grid based sonar map. The mobile robot could therefore learn to find a safe path according to its self-built sonar map. To solve the problem of local minima in conventional potential field methods, a new type of potential function was formulated.

The problems faced by users of ultrasonic sensors include the high incidence of misreads. Histogrammic In Motion Mapping (HIMM) and Vector Field Histograms (VFH) have been used to reduce the effect of misreads. Murphy *et al.*(1996) Compared HIMM/VFH and Dempster-Shafer techniques for obstacle detection for a moving robot. The HIMM/VFH methodology worked well for a robot navigating at high speeds, but the algorithms showed poorer performance at lower speeds in cluttered areas as the beam width created a wide, slow moving area in which the object could be located. Dempster-Shafer was investigated instead. The number of grid elements updated varied as a function of the robot's velocity. In addition, varying the beam width with the velocity of the robot improved the updating of an occupancy grid using Dempster-Shafer theory versus that of HIMM. The Dempster-Shafer method tended to handle noise better and make smoother and more realistic maps.

Gilbert & Johnson(1985) and Khatib(1986) applied the method of representing obstacles by distance functions, in the case of Khatib, motivated by the electrostatic repulsion between like charges. For example, the mover and obstacles could be represented by positive charges. This artificial potential repulsion approach was aimed at the local, short-term avoidance of obstacles in real time. The function tended to infinity as the point approached the surface and was zero beyond a certain distance from the obstacle. This representation had the advantage that the task of calculating the distance between a robot and an obstacle was replaced by the task of evaluating the simpler function. Compared to solid geometry or polyhedral models, these calculations were relatively fast.

The repulsion force was generated by a fictitious potential field around each obstacle due to a potential assigned to it. When any link of a robot arm approached an obstacle, a repulsive force pushed the link away from the obstacle. If P was the potential function used, and D was a function of the minimum distance between the link and the obstacle, then P became larger as D became smaller, and became zero beyond a pre-set distance from the obstacle. A development from this technique was used in the work described in this Dissertation.

2.6 A Human operator within the control loop.

Automated systems tend to carry out a set of tasks or procedures in response to stimuli such as sensor input or a pre-set collection of rules. A tele-operated system receives commands from an operator remote from the system via a communication link. The operator remains in control. In a system where the operator and a separate control system are working together, conflicts can occur. The problem of placing an operator within the control loop has been investigated widely. Notably within the drive to introduce intelligent and assistive systems to motor vehicles.

Galer(1995) described new in-vehicle systems that provided the driver with information about the driving task, routes to take on a journey, the availability of parking facilities, congestion ahead and so on. Some systems went further than simply providing the driver with information, they also took appropriate action in place of the driver.

Examples included autonomous intelligent cruise control and lane keeping devices. These technological advances have the potential to improve comfort, convenience and safety and reduce accidents. When vehicular problems become demanding, system designers have often proposed replacing the driver with computer automation (and non-human sensors) [McIlvaine Parsons(1996)]. Little(1997) presents more information on the proposed systems for "smart vehicles".

Lee(1994) and Lee & Morey(1992) investigated the increasing use of automation to supplant human intervention in controlling complex systems. The use of automation could change the operators' role from active controllers (directly involved with the system) to supervisory controllers (managing the use of different degrees of automatic and manual control). Trust, combined with self-confidence were examined and showed how trust and self-confidence relate to the use of automation. In general, automation was used when trust exceeded self-confidence, and manual control when the opposite was true. Since trust and self-confidence were two factors that guided operators' interactions with automation, the design of supervisory control systems should include provisions to ensure that operators' trust reflects the capabilities of the automation and operators' self-confidence reflects their abilities to control the system manually. There is an interesting connection with the two Airbus disasters discussed in Learmount(1991) and the work by Lee(1994).

Carroll(1997) discussed the use of computers in improving the usability of computer systems and applications in a history of Human-Computer interaction over the last 20 years. Human-computer interaction (HCI) is described as the area of intersection between psychology and the social sciences, on the one hand, and computer science and technology, on the other. HCI researchers analyse and design specific user-interface technologies (e.g. three-dimensional pointing devices, interactive video). They study and improve the processes of technology development (e.g. usability evaluation, design rationale). Through the past two decades, HCI has progressively integrated its scientific concerns with the engineering goal of improving usability, and establishing a body of technical knowledge and methodology.

Parsons(1996) presented Generic Intelligent Driver Support (GIDS) project. The method proposed in the GIDS project was to create four simultaneous computer simulations (experts) of a car driver's response to a common road situation. One simulation represented what a driver would be doing, another what a reference or ideal driver would be doing and the third detected the difference between them. The fourth model advised the driver what he should be doing. GIDS would adapt to the driver's intentions, capabilities and abilities.

Tele-operation was described by Kwitowski *et al*(1995) as a technology that could provide significant health and safety improvements in mining operations. The US Bureau of Mines developed a tele-operation system for continuous mining and haulage equipment. This system featured near real-time closed-loop control.

A tele-operation system for a powered wheelchair was described by Gunderson *et al*(1996). A remote camera system was fitted to a powered wheelchair and a video link established when required to an operator at a control station. It the user of the wheelchair became ill or tired whilst outside the care centre, they could call the operator via a radio link, then the control station operator would guide the wheelchair back to safety using a tele-operation system and wheelchair mounted video cameras.

Wiker(1993) discussed Tele-robotics in general. Studies had shown that the effect of tele-robotics work on tele-operators could be having the reverse effect to that desired. It could take an operator 500-800 times longer to complete a task using tele-operation. Some operators become fatigued and frustrated and suffered from motion sickness during tasks. Whilst it was still desirable to use manually operated or automatic systems, there was still a place for tele-robotics in inaccessible or dangerous areas [Hewer(1995)].

2.7 Controllers & Systems.

Wheelchair controllers are produced by a number of manufacturers and are essentially servo controllers that allow the wheelchair operator to control the large currents of the drive motors using a joystick or other input device. Other features are built into the controllers that provide safety for the user. Acceleration, deceleration, speed and turning are affected by safety systems in the controller [Crane(1995)]. The safety systems can be adjusted by carers to suit the individual requirements of the user. Modular systems are available that have simplified the connection of controllers and their sub-systems using a bus system [Taylor(1995)]. The majority of wheelchair controllers rely on a joystick as the only input. In most cases this is sufficient. The work described in this Dissertation was aimed at those situations when the user required a more intelligent controller to drive a powered wheelchair safely.

In the past, the control system has been regarded as the most crucial aspect of a mobile robot [Galt(1999) and Song and Waldron(1989)]. The development of an effective control system is difficult. Gray(1996) stated that the development of system architectures for mobile robots and the integration of the required enabling techniques of control, actuators, sensors and artificial intelligence provide some of the most interesting intellectual challenges in current engineering development.

The control system is ultimately responsible for stable operation. Any foreseeable, and often unforeseeable event must be dealt with if the vehicle is to be used in serious applications where robustness, reliability and efficiency are essential. The control system must be planned and constructed so that the derived control behaviour, conforms to safety and efficiency specifications. Also, the control system should allow for future expansion and provide a means to be reconfigured. The problem is considerable when the device to be controlled has many input and output parameters.

It is difficult to arrive at the solution for such a complex problem in a single step and it is important to maintain an orderly method of problem deduction and break down [Galt(1999)]. This is true of most complex engineering projects and as such there are accepted methodologies for efficiently dealing with the problem. These are discussed in Galt(1999).

It is possible to pre-define the requirements for the control system based on general requirements and experience. These requirements provide an indication of the scheme required for the structuring of the control system.

The most important considerations are the safety and reliability of both the wheelchair and the human operator (and other humans). Such issues generally vary depending on the application. In application areas such as healthcare, transportation, construction, surveillance, agriculture and the service industries, advanced robotic devices will come into close proximity to humans without the usual safety barriers normally associated with manufacturing robots. An advanced powered wheelchair should have the ability to operate in unstructured environments where hazards may be encountered.

Safety issues become paramount for both operational and legal requisites. This raises a paradox within the systems. Complex, intelligent behavior can be unpredictable yet it must be made possible to predict that the control system will ensure safety at all times. In essence, the control behavior should be inherently robust and enable the device to cope with a span of difficult scenarios so that performance will degrade in a predictable and safe manner.

The architecture of the control system should be designed for minimum representation and complexity. This suggestion aims at creating an architecture which will be simple to follow and understand, and thus easy for new operators to use and modify. Also, reducing complexity will inevitably reduce development time by simplifying debugging operations.

It was clear that it would be difficult to devise an ideal control structure within the first iteration. It was therefore important to provide a generic framework that could be updated and improved with new principles and characteristics, without the requirement for major reconfiguration. This approach also allows for the latest technologies to be integrated into the existing system, for example the changes of micro-controller and software made during the new research described in this Dissertation.

Adaptive control is being used more and more for the development of mobile vehicles [Galt(1999)]. A suitably applied control system that was self-optimising could make a wheelchair faster, safer and more efficient.

2.7.1 Artificial Intelligence.

If the function of a user was impaired, an intelligent controller may have assisted the user. "Artificial Intelligence: a philosophical view" [Copeland(1993)], is a history of Artificial Intelligence (AI) and a discussion of whether a computer could be intelligent. The difference between a computer AI system and a human brain was described as being enormous in the way it operated and was constructed. AI systems could be good at specific tasks but were not adaptable due to their narrow view of the world. An introduction to how neural networks work was also included in the discussion.

Expert systems are a type of AI system. An expert system can be a collection of rules and reactions to stimuli gathered by knowledge engineers [Hudson(1997) and Rasol(2001)]. Knowledge is collected from experts, skilled operators and craftsmen. The knowledge engineer's job is to extract that expert knowledge from the expert source and to then represent that knowledge in a computer program. This is called knowledge mining and is essential to the effectiveness of the expert system.

Almost all environments of genuine interest to mobile robots are subject to dynamic change on a variety of time-scales [Galt(1999)]. If an intelligent wheelchair or mobile robot is to negotiate in an unknown environment then it should be capable of dealing with such dynamic variation [Galt(1999) and Cliff *et al*(1995)]. Most human beings take the ability to reason and make decisions for granted; this ability stems from the fact that humans are highly intelligent beings. Programming a machine to be able to think for itself is a problem that has plagued researchers.

Artificial intelligence (AI) is the label given to the study of intelligence using the ideas and methods of computation [Addis(1996)]. Also referred to as soft computing or computational intelligence [Galt(1999)]. AI techniques offer a new perspective and a new methodology in the generation of autonomous robots. The central goal is to make computers intelligent, both to make them more useful and to understand the principles that make intelligence possible.

Vijaykumar(1987) and Galt(1999) reported that although many autonomous mobile robots existed, there were few intelligent systems, they were either tele-operated or had to use a pre-programmed, inflexible strategy with little ability for making decisions. Artificial intelligence techniques are improving. Relatively new techniques such as fuzzy logic, genetic algorithms (GAs) and artificial neural networks (ANNs) have been extensively used as AI tools for developing autonomous systems [Cox(1987), Haupt and Haupt(1998), Zalzala and Morris(1996)]. These techniques have made it possible to create computer programs capable of complex behavior and reasoning. A debate continues on whether these programs are truly intelligent or are just an example of very advanced logic programming. What is evident is that these tools have provided a means by which a mobile robot can be programmed to exhibit types of behavior that were not possible before and these behaviors can be used in advanced powered wheelchairs.

AI techniques have been used very little in commercial applications [Galt(1999)]. When a mobile robot is programmed to exhibit complex behavior it becomes difficult to predict, making it difficult to get the robot verified under current commercial safety standards. As a result, a majority of existing autonomous robots are simply not considered reliable enough for most applications.

2.7.2 Fuzzy logic.

Artificial intelligence systems are typically used to solve problems which have no precise mathematical model. Holve & Protzel(1996) used fuzzy control for controlling a 1/10-scaled model car. The classical problem of how to reverse park the model car was demonstrated. The model car was fitted with an onboard microcontroller and sensors for the task of autonomous parking. This was similar to the new intelligent wheelchair controller created during the new research described in this Dissertation. There was no precise mathematical model of the car's kinematics and the controller had to cope with heavy sensor uncertainty, imprecise actuators, and a sketchy world model. Learning approaches were not used, but the parking knowledge was expressed in the form of fuzzy rules. The reverse parking problem is a typical problem facing drivers. The drivers being experts at solving the problem even when the external conditions could change slightly for each parking maneuver.

Fuzzy logic systems also found applications within the design of a Decision Support System (DSS) and to improve the efficiency of fuzzy supervision of Statistical Process Control (SPC) [Zalila *et al(*1998)]. SPC was widely used in industry to control manufacturing processes and product quality. Using control charts, two statistical parameters (mean and standard deviation) were regularly checked to maintain the process under control and to improve performance. A fuzzy-rule-based supervision system was proposed as an alternative to SPC. The system modeled the analysis, diagnosis and decision of an expert who controlled a process by means of control charts. It was divided into three modules and delivered two gradual visual alarms. The system was been tested in simulation with real data and induced better control in comparison to the classical chart method. The fuzzy supervisor made anticipation of the process behavior possible and advised the operator.

2.8 Expert systems & Artificial Intelligence.

To assist a powered wheelchair user to navigate safely within an unstructured environment it was proposed to augment the user's input with recommendations from an on-board wheelchair expert system.

The representation of uncertain data within an expert system was investigated and preliminary work was completed to investigate the use of fuzzy representations for the expert knowledge. The use of simple sensor systems made the data available to the expert system uncertain and gave minimal indication of the environment around the wheelchair. This made the data suitable to be considered in fuzzy terms, as an accurate definition of the problem facing the expert system was not possible.

An Expert System could solve problems by heuristic or approximate methods. A heuristic is essentially a rule of thumb. A strength of Expert systems was that they could make assessments and decisions even when data were incomplete.

Artificial Intelligence is a branch of computer science concerned with the design and implementation of programs which are capable of emulating cognitive skills such as problem solving, visual perception and language understanding.

Hudson (1997) described an expert system as consisting of a three level structure:

- Data level
- Controller level
- Knowledge level

The main elements were the User Interface, the Inference Engine and the Knowledge base and these related to the levels of the Expert System as shown below:

Data level	USER INTERFACE
Controller level	INFERENCE ENGINE
Knowledge level	KNOWLEDGE BASE

Figure 2.2 shows the basic structure of an expert system as reported by Hudson(1997).

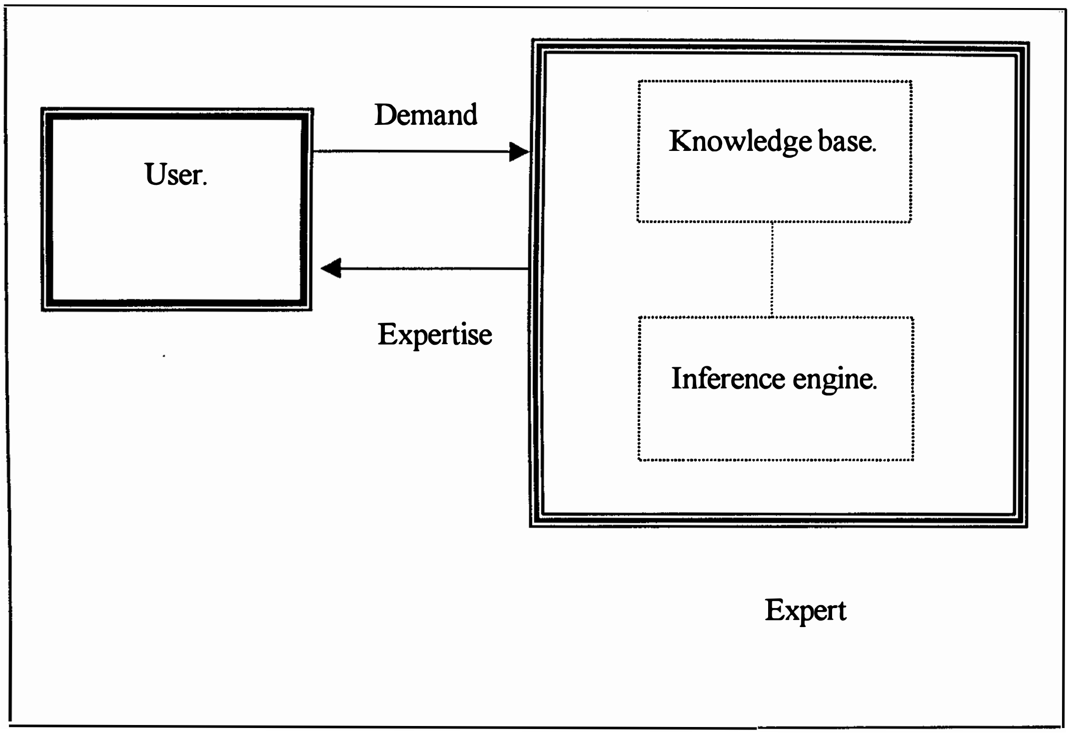


Figure 2.2. The basic structure of an expert system [Hudson(1997)].

2.8.1 The User Interface.

The User Interface dealt with all communication between the user and the system. In a conventional expert system, the user presented a problem to the expert system. The system prompted the user to provide information, responded to inquiries and returned a solution. Expert systems could explain why decisions were made and incorporate a vast array of domain specific knowledge. Users could possess a wide range of experience so the User Interface had to be suitable for use by any user.

A User Interface could allow bi-directional communication between the expert system and the user. The user must be able to pose a problem to the expert system and to stimulate a response.

The User Interface design could offer a number of practical benefits to a potential powered wheelchair user:

- Increased user acceptance of prototypes and the eventual system.
- Increased frequency of use.
- Decreased user-training time.
- Increased speed of performance.

In order to meet the needs of many users, expert system developers have been increasingly considering the use of intelligent or adaptable interfaces. Adaptable interfaces included mechanisms such as model-based modules to store user profiles. A profile might include information about the level of expertise or sets of preferences. This would effectively be another expert system adapting the main expert system to meet the requirements of a particular user.

Important issues of responsibility needed to be addressed, such as who was responsible for the consequences of users accepting advice from a powered wheelchair expert system. The issue of misinterpretation of advice or trajectory was also of importance. These issues were of particular importance where such systems were implemented in traditionally non-computing environments such as rehabilitation technology.

User Interface design was a major field of research in itself and beyond the scope of this research project although they are considered briefly in the next Chapter.

2.8.2 The Inference Engine.

The Inference Engine guides the manipulation of knowledge in a Knowledge Base. The inferential reasoning mechanism is distinct from a Knowledge Base. During execution it could complete a series of cycles. Each cycle would begin by testing to see which rule had its condition satisfied by trying the rules against the current information in memory. When a matching rule was found it was asserted, (that is, the 'action' part of the rule was performed). This changed the state of the current solution or information and so the cycle began again. This cycle was known as the 'recognise-act cycle'. An important distinction was between 'condition-driven processing' and' consequence-driven reasoning'.

condition	➔	action (behaviour)
antecedent	➔	consequent (inference)

A system driven by the conditions to the actions used a reasoning method called forward chaining. Forward chaining started with the data, examined the IF clauses and searched for a solution by working from the data towards a goal or solution. When all of the IF clauses in a rule were satisfied, the expert system gave the user the recommended solution (by altering the trajectory of the wheelchair). Forward chaining searches could expand rapidly, since a large number of rules could be examined. For this reason, controls were needed to limit the number of goals examined in order to reduce the time and computer capacity required for processing. Production rule based systems typically used this technique, a notable exception being MYCIN [Shortliffe(1976)].

Consequence driven reasoning was known as backward chaining, which involved starting with one or more possible goals. In a rule-based system, for example, the inference engine tested each goal to see whether all of the IF clauses in the rule containing each possible goal were true. Each rule was tested in turn until an answer was found (that is, until a rule was found in which all the IF clauses were true) or until all possible rules had been examined and no answer existed.

2.8.3 The Knowledge Base.

The rules and facts of the particular knowledge domain were stored in the Knowledge Base. Information structuring and the choice of representation for particular applications depended on the problem to be solved and on the form in which knowledge could be most easily described and used. The knowledge could be classified into two types; declarative and procedural. The facts, laws or terminology within a specialist domain constituted the declarative knowledge, emphasising concepts and their relations to other concepts. The following representation example is in the artificial intelligence (AI) language PROLOG:

isa (dog,mammal). isa (Harry,dog).

The first fact states that a dog is a mammal, the second states that Harry is a dog. From these facts the rules of inheritance deduce that Harry is a mammal. Declarative knowledge was explicitly specified rather than being implicit in a procedure. Procedural knowledge specified how to solve a given problem or task rather than specifying the actual problem. It was represented by heuristics and procedures.

2.8.4 Knowledge Representation.

The method of knowledge representation adopted was constrained by the type of knowledge in the expert system. There were many formalisms available. The most common are described. An analysis of these different approaches to knowledge representation can be found in Jackson(1990). The areas covered here are:

(a) Symbolic Calculation.
(b) Production Systems.
(c) Associative Nets and Frame Systems.

(a) Symbolic Calculation. A symbol was used to represent an item, the 'designate' of the symbol. In order to use symbolic calculation the following needed to be specified:

- Syntactic Rules -These dictated how to form symbol structures from symbols so that the structure had a meaning that was a function of the constituent symbols.
- Transformation Rules -These specified how to transform one symbol structure into another.

A symbolic program typically took one or more symbol structures representing the initial state of a problem as its input, performed transformations and output a symbol structure that represented the solution.

Symbolic systems were realised as physical symbol systems, which could be described as a machine with the following components located in *some* environment [Newell(1982)]:

- A memory containing symbol structures, which could vary in number and content over time.
- A set of operators for manipulating symbol structures, such as reading, writing and copying.
- A control for the continual interpretation of whatever symbol structure was currently active or being attended to.
- An input from its environment via receptors and an output to that environment via some motor component.

The most common symbol structure implementation was LISP, which used a list as the symbol structure. The operators that provided the functionality of the system were based on a logistic system called Lambda Calculus.

(b) Production Systems. Production rules were usually used to encode empirical associations between patterns of data presented to the system and the actions that the system should perform as a consequence. They were general rules of behaviour. They were sometimes called condition-action or situation-action rules and were based on the IF - THEN construct. Production rules were the most common method used to represent knowledge in expert systems. Examples of production rules are shown in figure 2.3. The premise could be a single condition, or a conjunction of conditions. A typical use of production rules is demonstrated in Rasol (2001).

IF the CLEARANCE LEFT is insufficient, THEN STEER RIGHT.

IF COLLISION, THEN STOP MOTORS.

IF the JOYSTICK conflicts with sensors, THEN increase the JOYSTICK effect.

Figure 2.3. Examples of production rules for steering a powered wheelchair.

(c) Associative Nets and Frame Systems. Associative nets and frame system representations were more suited to systems in which the properties and relationships between entities were the most important considerations. The representation of system semantics was the key aspect of these methods. Such an approach was not suitable for this application and so these methods are not considered further.

2.8.5 The Expert System Shell.

Expert system shells provided a framework of components from which a customised expert system could be created. Expert systems were originally developed using programming languages such as LISP and PROLOG. As the field expanded many other development tools and expert system shells became available, ranging from development versions of programming languages (such as OPSS and INTERLISP) to larger shells (such as ART, ESE, and KEE) to more limited shells (such as FLEX VP-EXPERT).

Some shells required the use of a mainframe computer or were designed for use on PROLOG or other special-purpose (dedicated) symbolic processing machines. Kantrowitz (1995) provided a detailed review of expert system shells available. Microcomputer expert system shells were good for developing small systems to test

The Advantages.

Expert systems offered numerous advantages [Riley (1990) and Hudson (1997)].

- Increased availability. With the expertise available in computer *form* it could be copied and broadcast to whomever found the information useful.
- Reduced costs.
- Reduced danger. Expert systems could be placed in environments that could be hazardous to humans.
- Permanence. Human expertise is not permanent. Humans could die, retire, or leave, taking their expertise with them. Information placed into an expert system was permanent.
- Multiple expertise. The level of expertise combined from several experts could exceed that of a single human expert.
- Increased reliability. Expert systems could provide a safety net by providing a second opinion to a human expert. This was particularly useful in situations where the human experts were placed in situations of stress and fatigue.

- Explanation. Human limitations may effect the decision given by the human expert and their logic might be confusing. An expert system could explain the reasoning that led to a conclusion.
- Fast response. Depending on the type of system built, the expert system could respond faster than a human expert. This could be important in real-time and emergency situations.
- Steady. Unemotional and complete response at all times. Due to the emotional and physical components of a human expert, they will not operate at 100% at all times. An expert system had none of these conditions.
- Intelligent tutor. The expert system could act as an intelligent tutor by letting a student run sample programs and explaining the system's reasoning.
- Intelligent database. Expert systems can be used to access a database in an intelligent manner.

Disadvantages.

- Human experts know the extent of their knowledge and qualify their advice as the problem reaches their limits of ignorance. A human expert also knows when to 'break the rules'. Unless expert systems were explicitly designed to deal with uncertainty (e.g. fuzzy logic), recommendations were made with the same confidence even if the data supplied was inaccurate or incomplete.
- Inability to represent causal knowledge. That is, expert systems did not have an understanding of the underlying causes and effects of a system. It was much easier to program an expert system with shallow knowledge based upon empirical and

heuristic knowledge than deep knowledge based upon the basic structure, function, and behaviour of objects.

- Expert systems were inflexible. They were limited to the knowledge domain for which they were designed.
- Typical expert systems could not generalise their knowledge by using analogy to reason about new situations in the way that humans could.
- The capture of expert knowledge was time consuming. This problem of transferring human knowledge into an expert system was called the knowledge acquisition bottleneck.
- When knowledge was placed into an expert system it needed to be validated in some way to check that the experts knowledge was exact.

2.8.6 Rule-based Systems

Some expert systems are based on techniques known as rule-based programming. The rules are condition-action rules like IF... THEN rules. Chained rules occur when the conclusions of one relate to the conditions of the next. It is not a sequential execution through the program rules; they are used when they are appropriate not just when they are encountered. Each rule is independent.

IF condition(s) THEN action(s)

What is meant by a condition being satisfied is that the `condition' can somehow match information in the memory or the input specification. That information may be the goal i.e. the collision free path of the wheelchair, or it may be an intermediate action such as the wheelchair stopping and waiting for an updated set of inputs (a response from the user).

Execution of the program may consist of a number cycles. Each cycle begins by testing to see which rule has its condition satisfied, by trying the rules against the current information in memory. When the rule is found it is asserted, that is the "action" part of the rule is carried out. That changes the result or the current information, and so the cycle begins again. This cycle is known as the `recognise-act cycle'.

2.8.7 Blackboard Architecture for expert systems.

The blackboard expert system construction was created as part of the work on the Hearsay speech understanding system. The ideas were developed into what has been described as the standard blackboard architecture in Hearsay II [Erman *et al*(1980), Lesser *et al*(1975) and Nii(1986a)]. As an example, the construction of Hearsay II is shown in figure 2.4, reproduced from Sanders and Hudson(2000). The blackboard monitor identified which knowledge sources should be triggered in response to changes on the blackboard and updated the focus-of-control database. The scheduler used this information when determining which knowledge source to execute.

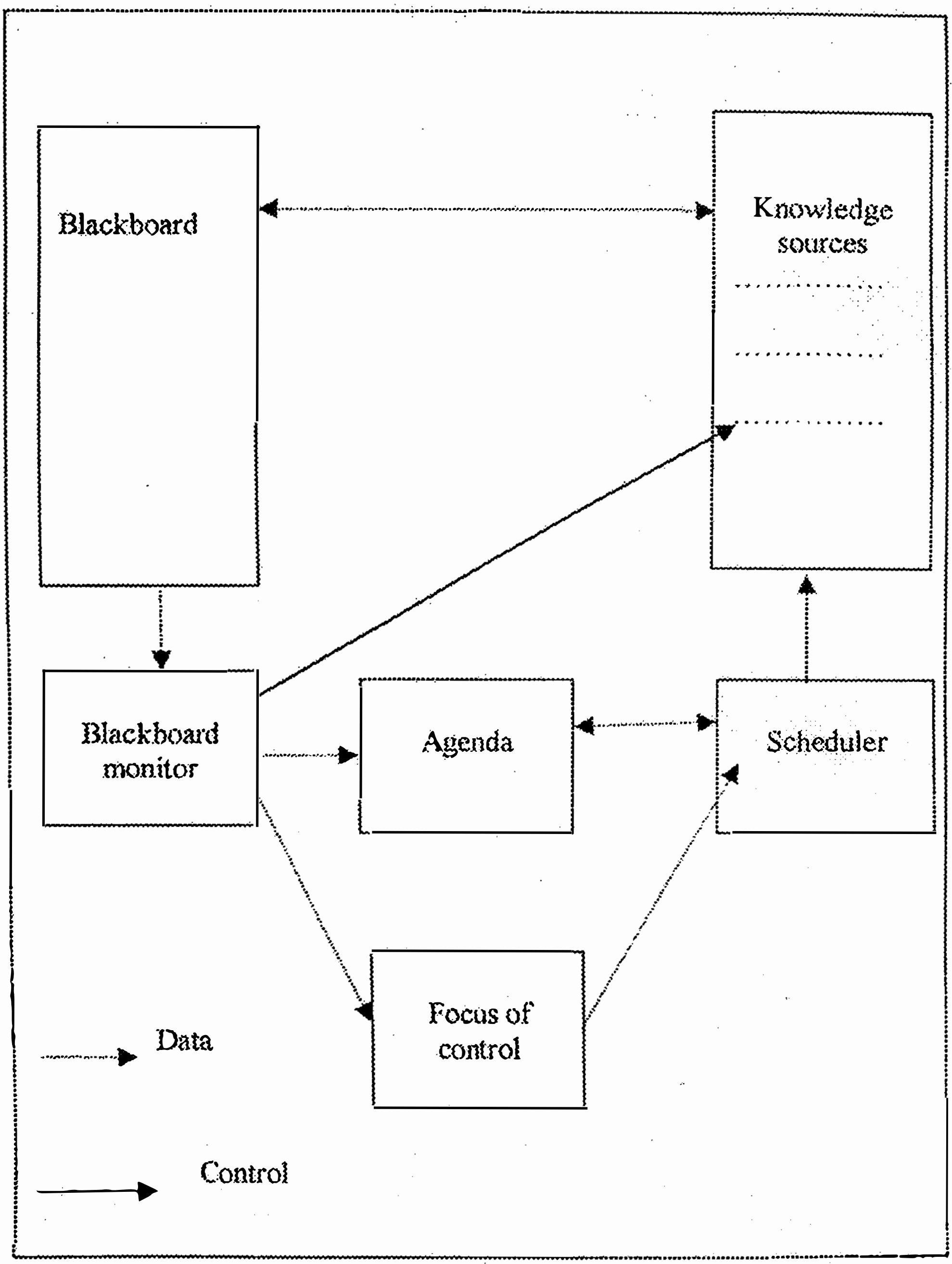


Figure 2.4 . The structure of the Hearsay II blackboard framework. ***Reproduced from Sanders and Hudson(2000).***

The HSAP/SIAP project [Nii *et al.*(1982), Nii(1986b)] for interpreting sonar signals used a control mechanism based on the occurrence of pre-defined events. Events were the primary basis of control in this construction. This was the case in the new system described in this Dissertation where the wheelchair moved from event to event and so this architecture was considered as a possible method for the expert system.

The CHRYSALIS system for protein crystallography [Engelmore and Terry(1979) and Terry(1988)] introduced the idea of hierarchical control of blackboard systems. The framework used a hierarchy of control or 'Task' knowledge sources to control domain knowledge sources, as shown in Figure 2.5 reproduced from Sanders and Hudson(2000). The domain knowledge sources caused events to occur on the blackboard. The state of the blackboard was described using a features list that was monitored by the strategy knowledge source. The strategy knowledge source examined the features list, selected an area of the blackboard on which to focus and specified a sequence of task knowledge sources to execute. The task knowledge sources selected events that affected the selected area of the blackboard and specified domain knowledge sources to execute. This construction gave the system two levels of control: strategy and task. This control structure provided a more explicit goal-directed control architecture than that of the Hearsay system, but reduced the level of opportunistic reasoning. Another early system to use a hierarchical control system was VISIONS [Williams *et al*(1977)].

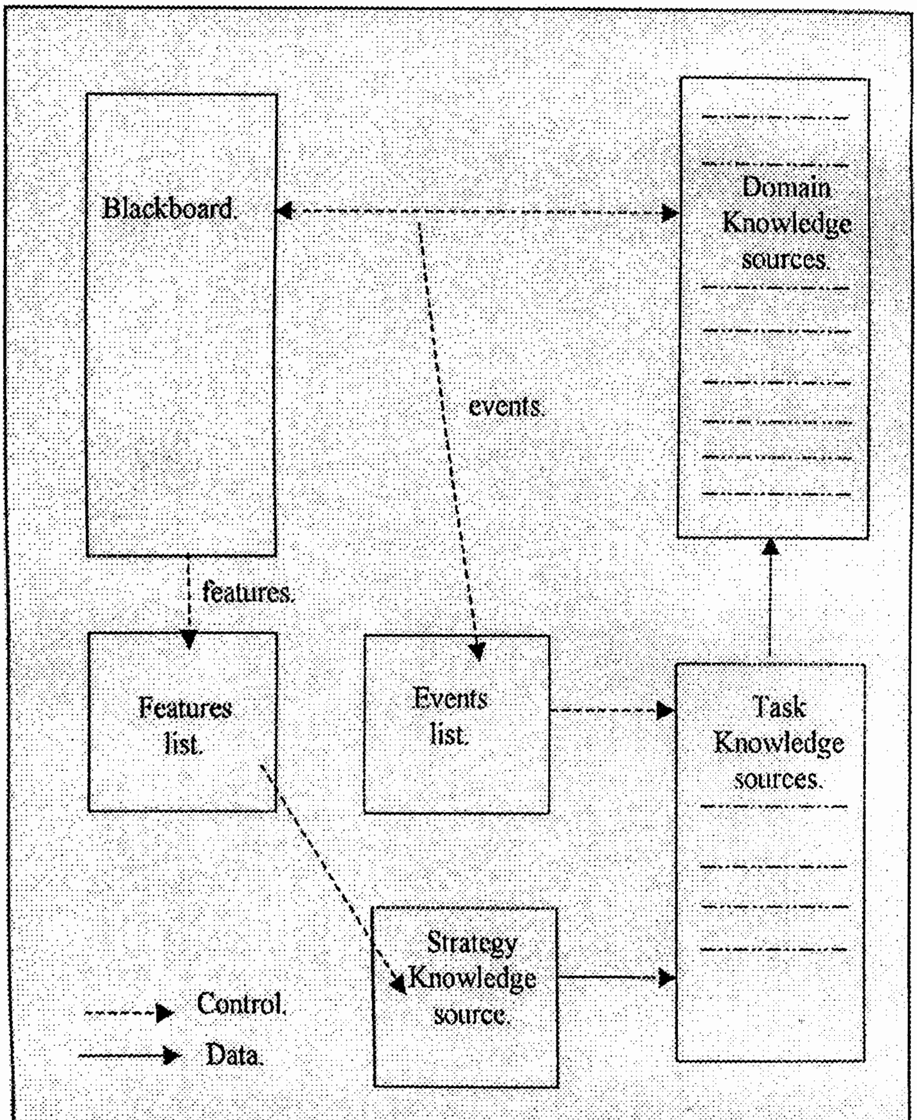


Figure 2.5. The structure of the Chrysilis blackboard architecture.
Reproduced from Sanders and Hudson(2000).

The Goal Directed Blackboard Architecture [Corkill *et al*(1982), Lesser *et al*(1989)] extended the Hearsay II architecture by adding a goal blackboard and a goal processor. The framework was first implemented in the Distributed Vehicle Monitoring Test-bed [Lesser and Corkill(1983)]. The structure of the goal blackboard was the same as the domain blackboard. The goal processor used three mapping functions; hypothesis to goal mapping; goal to sub-goal mapping; goal to knowledge source mapping. These mapping functions were used to post goals and sub goals and to trigger knowledge sources. The architecture is shown in Figure 2.6 reproduced from Sanders and Hudson(2000). The architecture integrated goal and data directed reasoning because goals and sub goals were generated from data changes on the domain blackboard and from changes on the goal blackboard. The assistive wheelchair system under construction in this research was predominantly data driven. The data input to the wheelchair system was modified in response to movements of the wheelchair caused by the previous responses of the system. The application was not of significant complexity at this prototype stage to require devolving into a series of goals and sub goals and so this framework was not considered suitable at this stage of the work. However it should be considered as a possible future framework for the system. Other blackboard systems are described in Sanders *et al.*(2000) and Sanders & Hudson(2000).

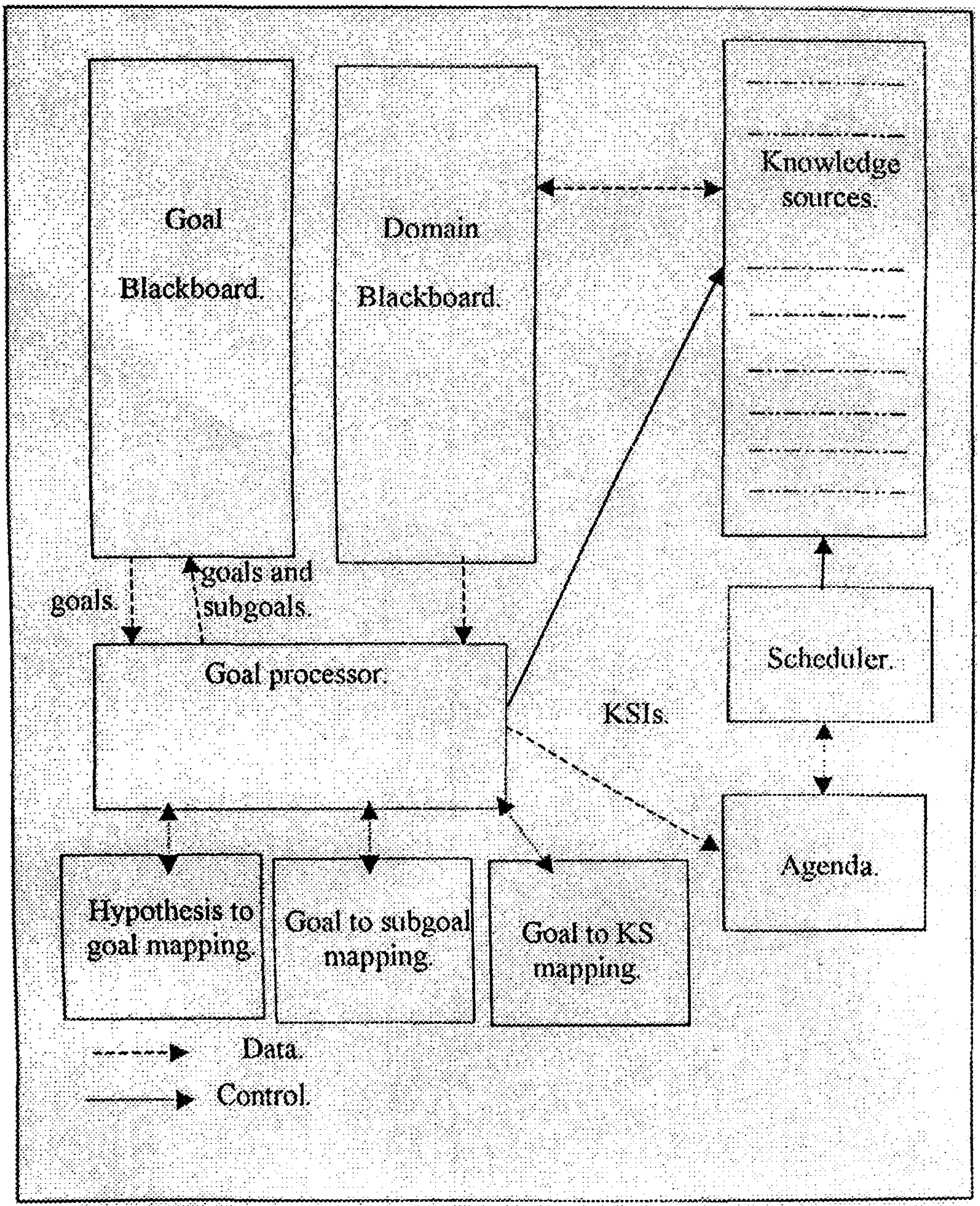


Figure 2.6. The Goal Directed Blackboard Architecture
(Reproduced from Hudson (1997)).

2.9 Advanced Wheelchair Systems.

Stott(1997) discussed advanced and intelligent wheelchair systems and some of these are described in this Section. Of particular relevance to the work described in this Dissertation was the SMART Wheelchair [Craig *et al*(1995) and Nesbit(2000)]. Which is discussed in Section 2.9.2.

The NavChair project discussed in Section 2.9.6, aimed to create a wheelchair system that co-operated with the user in a shared decision making process [Bell *et al*(1994)]. A more robotic approach was the Senario project [Beattie(1993)]. The wheelchair was equipped with many sensors including scanners to read wall mounted bar-code beacons, laser range finders and radio beacon detectors to provide an automated navigational system. At an estimated cost of 20,000 ECU per wheelchair the system may prove prohibitively expensive. The Senario project is discussed in Section 2.9.3.

Gomi & Ide (1996) described another wheelchair drawing heavily on robotic and autonomous technology. An autonomous wheelchair system was built into an unmodified Fortress wheelchair. This system was interesting as behavior based AI was used to control the wheelchair which reduced the dependence of the control algorithms on the hardware (by having a more flexible approach). The sensors were collision, IR rangers and a stereo vision system faced forward of the wheelchair. The vision system used landmark navigation to move autonomously. The joystick could be used at any time to override the control hardware. A behavior-based approach was used to establish on-board autonomy at minimal cost, maximum safety and transparency in appearance. Two power wheelchairs had been built and tried with the new system.

Some autonomous navigation outside was demonstrated. The system was able to navigate the wheelchair in an automatic mode where the user did not need to make any decisions about the route that was being driven. The system was complex which would tend to make it expensive. A system of this type runs the risk of taking control from the user and becoming an automated platform. High cost systems will also be outside the scope of many powered wheelchair users. The systems described in this Dissertation aimed to provide some of the functions of the system described by Gomi and Ide useful in order to provide assistance to a powered wheelchair user without becoming too expensive to be practical.

Bourhis & Pino(1996) presented the initial results of the VAHM project which aimed to improve the control of powered wheelchairs by adding autonomous mobility systems. The VAHM project is discussed in Section 2.9.5. Yoder *et al*(1996) discussed the development of an automatically guided wheelchair for individuals who were severely disabled. A combined vision and acoustic sensor system was used on the wheelchair. The techniques used were mainly concerned with autonomous navigation which differs from the approach adopted at Portsmouth described by Stott(1997) where the user retained overall responsibility for navigating the wheelchair.

Another technique for assisting a user to steer a powered wheelchair was described by Brienza & Angelo(1996) with a force feedback joystick and control algorithm for wheelchair obstacle avoidance. Common tasks such as traversing through doorways, turning around in halls or travelling on a straight path are described as being complicated by "an inability to accurately and reliably control the wheelchair with a joystick or other common input device. Or by a sensory impairment that prevented the user from receiving feedback from the environment". An active joystick with force feedback to indicate obstacles in the environment was developed. The development of the joystick and associated control algorithms are described. An active force feedback joystick was used to assist a user to navigate in an unknown environment. Sensors were fitted to the wheelchair to sense the environment. The wheelchair system had two modes of assistance; passive assist where the joystick became stiffer to discourage trajectories which would cause a collision, and active assist where the wheelchair system moved the joystick degrees of freedom (moved the central position). Active assist steered the wheelchair away from objects. Passive assist tried to make the user steer away from an object. The paper lead to the suggestion that this system could be connected to a system that could assist in unknown environments and to help in pre-mapped environments along predetermined paths.

2.9.1 INCH: An Intelligent Wheelchair Prototype.

Work by Rao & Kuc(1989) was an early look at the problems faced by visually impaired wheelchair users but could also be applied to wheelchair users with sensory-motor impairment. It was recognized that a danger faced by wheelchair users was a sudden change in floor level or a "drop off".

The "INCH" prototype wheelchair platform was powered by two stepper motors and a microprocessor based control system. Six Polaroid ultrasonic transducers were fitted to this platform in the following locations: directly in front of the front wheels for forward motion, directly behind the rear wheels for reverse operation and one each side of the front of the wheelchair for turning motions.

The sensors were directed towards the floor to constantly check the chair to floor distance. Input from the user was simulated and the controller was able to modify the trajectory of the wheelchair when a drop off was detected. This early work demonstrated the positive assistance that could be afforded to a wheelchair user by a computer system. The prototype was successful in avoiding drop-offs when programmed to move in random directions on a tabletop.

2.9.2 The CALL Center "Smart Wheelchair".

Work conducted since 1989 at the CALL center in Edinburgh resulted in the Smart Wheelchair described in CALL Center publications, (1993a) and (1993b), Nisbet (1998), (1995) and (2000) and shown in figure 2.7. The Smart Wheelchair used a distributed microprocessor based sensor and control system. This system enabled

Figure 2.7. The CALL Center Smart Wheelchair.
***Reproduced from* Nisbet(2000).**

severely impaired children and adolescents to learn how to control a powered wheelchair. The "bump" sensors detected collisions and the wheelchair could be programmed to stop, reverse or reverse and turn. Improvements in the motivation and quality of life were observed in the patients who underwent the Smart Wheelchair training program [Call Center publication(1993a)]. The Smart wheelchair could also navigate through the environment using line followers and other floor based navigation markers. Modification of the environment was required when floor based navigation markers were used which made them more suitable for use in schools where the marks were permanent. Ultrasonic range finders were used to slow the wheelchair down as it approached an obstacle. The ultrasonic device was called the "slow down tool" and was described in Call Center publication(1993a) as "not intended to stop the pilot getting to where he or she wishes to be, only to stop them hitting it quite as hard".

The Smart Wheelchair was designed around a modular design method to be integrated with commercially available wheelchair systems [Nisbet(1995) and (2000)]. The software and hardware architectures were presented in Craig *et al*(1995).

The Smart Wheelchair program was notable for its clinical based approach. Craig *et al*(1995) discussed evaluation methodologies that were used during the Smart program. The engineering and design work for the Smart Wheelchair was conducted at the University of Edinburgh. Clinical trials were conducted at three schools and the Bioengineering Unit at Princess Margaret Rose Hospital.

2.9.3 SENARIO.

A large "smart" wheelchair project was the European TIDE project "SENARIO" described by Beattie(1993). This project aimed to produce an autonomous wheelchair for the hospital environment. The project included fitting a laser range finder, tilt detectors, wall barcode readers, radio beacon detectors and an intelligent navigational capability to a powered wheelchair. The wheelchair system was designed to have an autonomous capability to enable a patient to be transported around a building without the need for a carer to be present. The projected cost of the SCENARIO wheelchair was in the region of 20,000 ECU per unit. This may be too expensive for even large institutions. The new work described in this Dissertation was aimed at low-cost solutions to some of the problems associated with mobility. An approach was taken during this new work described in this Dissertation to maximise the impact of the new systems whist remaining affordable and transparent to the wheelchair user.

2.9.4 M3S.

The M3S project ran under the IMMeDIAte program which was part of the European Union's Strategic Program for Innovation and Technology, (SPRINT). The project was concerned with providing a standard protocol for the connection of input and output devices to systems for use by the disabled [Dillon(1995)]. A standard wheelchair was limited to one input and one output device. The input device was normally a joystick and the output device or end effector was the wheelchair itself. As rehabilitation technology equipment increases in complexity and flexibility the need for standard plug-in connections may become more important. The M3S project allowed flexibility for the user of powered wheelchairs through the use of a standard bus system. All devices were to connect to this bus and communicate automatically. This opened the possibility of a user having access to more than one input device, possibly a joystick for normal use and a set of head switches for use while carrying things. Other output devices could be operated from the same input device. For example it may be more convenient to the user to have access to a keypad as well as a joystick to allow the seamless transition from driving the wheelchair to operating a manipulator.

The M3S was also to allow the addition of "smart" devices such as collision avoidance modules or navigational hardware. With an advanced system such as M3S the addition of high technology devices to a wheelchair could become more practical and allow wheelchair systems to be developed in line with an individual's changing requirements. The new work described in this Dissertation would be suitable for integration to a system such as M3S as with a suitable interface, the new devices could be attached directly to the M3S bus. New sensor modules and new joystick controllers could be plugged into the M3S system which would allow rapid modification from a powered wheelchair to a new intelligent wheelchair system with little extra cost for installation.

2.9.5 VAHM and Assisted navigation for powered wheelchairs.

Pruski and Bourhis(1992) and Bourhis *et al*(1993) described how a powered wheelchair was automated in order to assist a disabled user. Their work at The University Of Metz in France demonstrated how automating some of the low-level tasks involved in driving a powered wheelchair could assist a severely disabled user. The VHAM wheelchair is shown in figure 2.8. The user would indicate a goal to be achieved on a computer model of a bounded global environment. The automated wheelchair would then move to that location without any further input from the user. The wheelchair could be stopped or its trajectory modified at any time.

The powered wheelchair received information about its environment from ultrasonic sensors, infra red detectors, odometers and contact bumpers. Navigation through the environment was achieved using the world model that had previously been mapped. The wheelchair always started from a known coordinate. The turning points and the goal were derived from data supplied by the odometric devices attached to the wheels of the wheelchair. The ultrasonic and infra red sensors were used to avoid obstacles.

The wheelchair successfully navigated through a known environment and moved from room to room. Navigational errors occurred due to inaccurate odometric data and non-geographical features of the environment model caused problems. For example a door could be closed when the model indicated an opening.

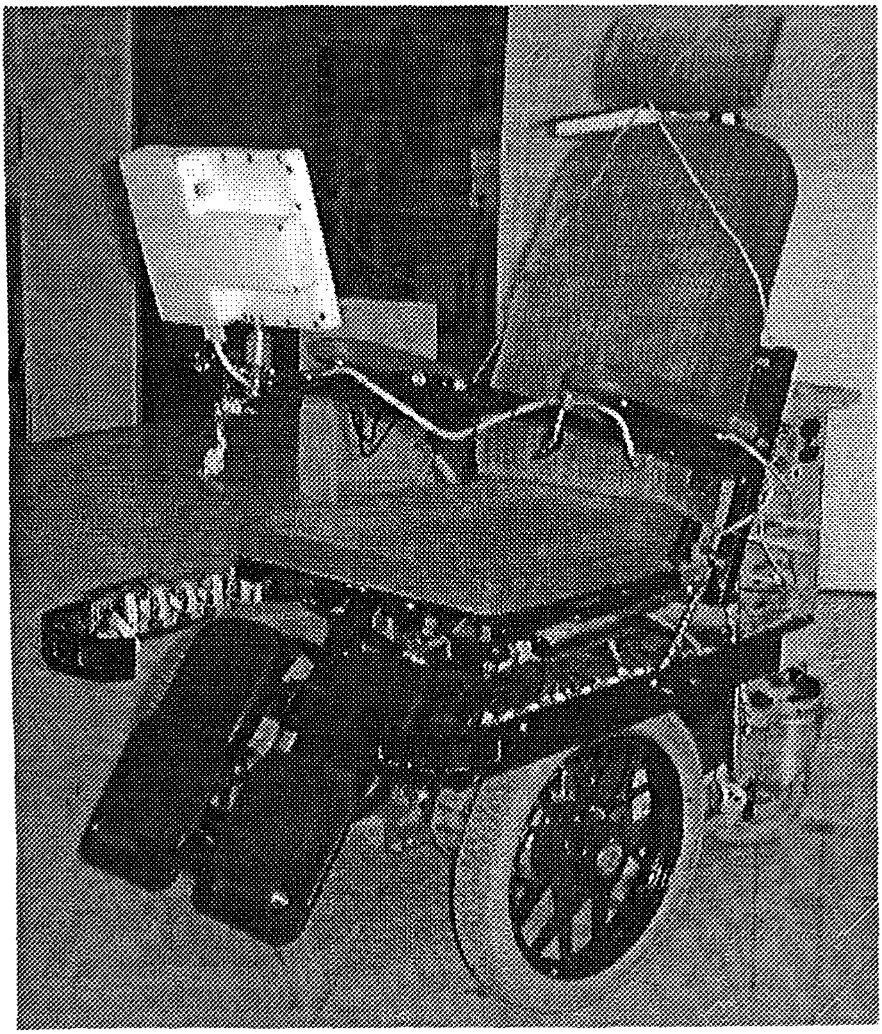


Figure 2.8. The prototype VAHM wheelchair.

At the time of printing, this image could be viewed at the University of Metz website at: http://www.fernuni-hagen.de/www2bonsai/FTB/aaate99/paper/99_12/99_12.htm

2.9.6 NavChair.

Work at the University of Michigan and The University of Michigan Hospital, USA developed a prototype assistive wheelchair called the NavChair shown in figure 2.9 . The work intended to meet the needs of multiply handicapped people who were unable to operate available powered wheelchairs. The NavChair aimed to share control with the user of the wheelchair to prevent unsafe maneuvers.

Adaptive shared control techniques were used on the NavChair in later work [Levine *et al*(1994) & Bell *et al*(1994)]. A wheelchair was fitted with ultrasonic sensors to detect the environment. The sensors could detect the environment and indicate the existence of obstacles in the path of the wheelchair. A problem occurred when trying to provide assistance to a user because only that person knew the real intentions of the user. The sensors sometimes detected several possible safe paths. For the wheelchair to act in an assistive manner it needed to detect the intentions of the user and take the user more easily to that goal. An identification technique was used to attempt to interpret the wishes of the user and polar histograms were used to identify safe paths for the wheelchair. This information was combined and a modified path executed by the wheelchair. Conflicts occurred which interfered with the path selected by the wheelchair. When approaching a doorway for example, the wheelchair could not distinguish between the user's wish to perform a "door passage" routine or an "obstacle avoidance" routine, which would take it past the door, and alongside the wall. The system required more information about the intention of the user in order to automatically perform mode selection.

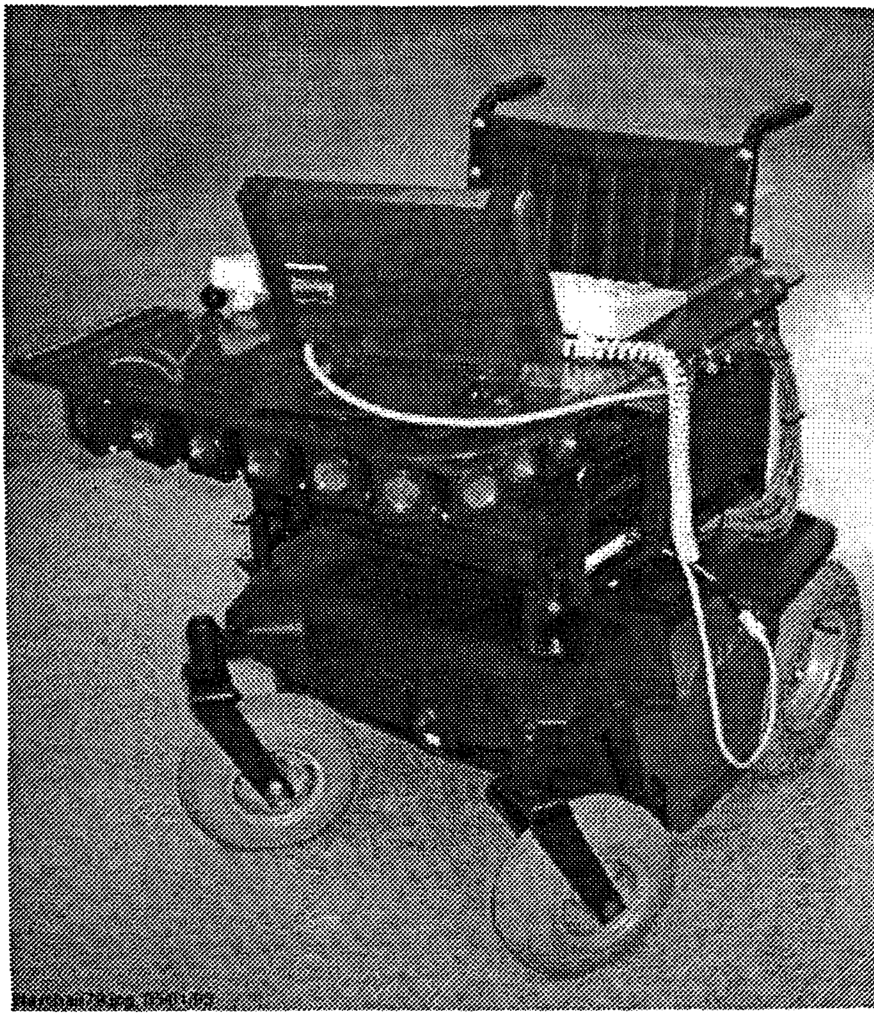


Figure 2.9 The NavChair.

At the time of printing, this image could be viewed on the University of Michigan website at:

http://www-personal.engin.umich.edu/~johannb/navchair.htm

A technique called "Stimulus Response modeling" (SRM) was developed that automatically determined the most appropriate mode of operation for the wheelchair. This improved the Navchair's ability to interpret the wishes of the user. The SRM technique learnt the user response to stimuli by initiating a disturbance to the wheelchair movements and measuring the user's response to the disturbance. Also the Vector Field Histogram technique was developed into the Minimum Vector Field Histogram technique. The performance of the NavChair improved to the point where the chair would follow walls, avoid obstacles and navigate through doorways.

An aspect of this work was the requirement to successfully interpret the wishes of the user and the incorporation of this information in the execution of a collision free path. Sonar sensors were used to detect the environment and a local map was built in the computer. Tests showed that the NavChair could be driven through a corridor by a blindfolded operator at about half the speed of an experienced sighted user without collisions.

2.10 Training potential wheelchair users.

Approaches to training new wheelchair operators are described in Douglas & Ryan(1987), Chase and Bailey(1990) and Stott & Sanders(2000a & b). Often a potential wheelchair user was selected after an assessment procedure which indicated the likelihood of successfully training the potential user to be a competent wheelchair driver. It is possible that some suitable powered wheelchair users are not selected as the procedures can be time consuming for the carers. Other methods for selecting potential powered wheelchair users could include new methods of simulating a powered wheelchair without the costs and supervision requirement of putting the potential user into a real powered wheelchair. One such method could be to use

Virtual Reality to simulate a powered wheelchair in a suitable environment [Stott and Sanders(2000a)].

Johnson and Aylor(1985) developed a dynamic model for an electric wheelchair. A computer simulation of the non-linear differential equations of motion for the powered wheelchair was developed. The simulation program was used to study appropriate control algorithms for implementation in a new digital controller design for the electric wheelchair. Such computer simulations could be used to create a wheelchair to operate in a virtual reality (VR) environment.

Stanton *et al*(1996) applied VR technology to study the training and enhancement of spatial skills possible using the medium of VR. Three-dimensional and two dimensional exploration tasks were completed by both disabled and able-bodied people. Three-dimensional exploration was found to be more effective at conveying complex information than two-dimensional exploration. The technology was used as a training tool with physically disabled children and it was found that interactivity and three dimensionality seemed to be crucial to spatial learning.

VR technology was originally developed by the military in the 1960s to help train pilots. This was useful because in VR, dangerous situations could be simulated without risking personal injury or loss of expensive equipment [Inman *et al.*(1997)]. Inman used VR technology to simulate a powered wheelchair to help teach mentally and physically disabled children to drive. Also a science teaching platform had been produced to help physically disabled students to become involved with science experiments.

Inman wrote "*Our interest in VR technology is not driven by it's availability or affordability – of course, we could not have embarked upon our recent work in the absence of these adventitious conditions. Our work stems from the needs of disabled children to acquire skills to function independently in the world. Our initial attempt to use this technology focussed on teaching children how to operate motorized wheelchairs*".

VR was used in this work to test the new algorithms as they were created. The work is discussed in Chapter 3.

2.11 Discussion

Many previous low cost systems of navigational assistance have been aimed at providing simple obstacle and collision avoidance functions. Other systems, based on industrial automated guided vehicle technology, have enabled a wheelchair to follow a line or a buried wire along a pre-defined fixed route within a school or institutional environment. The routes were defined by the positioning of lines or buried wires, which have involved modification of the environment. Whilst reducing the wheelchair users dependence on helpers, these systems have provided limited personal choice in the selection of routes and increased a users level of independence to an extent where some have progressed to "normal" powered wheelchairs.

Other research has been conducted using complex multi-sensor mode systems closely resembling industrial, military or space exploration applications. Natural features, artificial beacons and accurate maps of the operating environment have been created from which a trajectory has been determined. The movement of the vehicle has sometimes been monitored and corrected using multi-sensor techniques such as vision cameras, odometry and self-location from natural landmarks or artificial beacons located in the environment. Such advanced sensing devices have enabled a vehicle to operate in a previously known and modified environment.

More complex vehicles, that did not require modification of the environment, have enabled operation in a previously unknown environment. Both approaches however have resulted in fully autonomous vehicles that could provide little or no personal control to be exercised by the user. Although proving the technical feasibility of such systems within the laboratory, their cost and complexity have not provided the wheelchair user with a practical and affordable solution.

Powered wheelchairs and many other aids for disabled people tend to be personally adapted for the individual person. Some requirements result in a unique piece of equipment being manufactured in a rehabilitation workshop. Even successfully deployed powered wheelchair systems such as the SMART wheelchair are not considered suitable for all powered wheelchair users. A generic solution, although theoretically and financially desirable, is probably not feasible.

The provision of navigational assistance rather than fully autonomous transportation to a powered wheelchair user which assisted the user to operate in a previously unknown and unmodified environment was a promising area for research.

Fully autonomous vehicles have tended to require accurate and fast sensing systems. Slow or inaccurate sensor systems tend to limit the speed of the vehicle and the accuracy that the vehicle could navigate. Any form of mapping an unknown environment by a free ranging vehicle can be a long process and demand a specific pattern for the vehicle to follow. It is possible that a free ranging wheelchair could gather enough sensory data to map an environment. It is likely that the map would contain gaps. It may be possible to use simple low cost sensors to provide navigational assistance for a free-ranging powered wheelchair. The extent to which the assistance could be intelligent, given the low quality of information that a low-cost sensor system could produce was investigated during the novel and innovative work described in this Dissertation.

Variations in the operating conditions, such as floor surfaces, gradients and tyres make the maintenance of a straight line course difficult even for an expert powered wheelchair user [Collins and Kauzlarich(1998)]. The interface between the wheelchair system and the user could sometimes be confused. An obstacle avoidance system can prevent a user from driving a wheelchair to an exact position. An intelligent wheelchair system should aim to allow the user access to any safe area without damaging the environment, injury to the user or damaging the wheelchair. The novel intelligent wheelchair created during this work would allow a severely disabled person to use a powered wheelchair more safely than with an unmodified powered wheelchair.

Chapter Three.

VIRTUAL REALITY.

The control and handling of the micro-rover tele-operated robot [Matijevic (1996)], using Virtual Reality (VR) inspired similar research for wheelchairs during the work described in this Dissertation. Virtual Reality worlds were generated and viewed in real time in order to provide environments where a powered wheelchair could be simulated and novel algorithms could be tested in safety.

During normal development, in the first few years of life, a child develops the expectation that movements around the environment could be controlled and executed at will. The process of normal development introduces mobility to a child in a way that allows cognitive and navigational skills to be acquired at a rate equivalent to the competence of the child. A child who had not had the benefit of the natural training process may find the concepts of self initiated mobility difficult to grasp if they were suddenly to become available. A child who had relied totally on others for mobility during his life and was then presented with a powered wheelchair for the first time may find the experience difficult.

Powered wheelchairs have normally been presented to a first time user in a large uncluttered environment so that the user could explore the capabilities of the wheelchair without the danger of colliding with other objects. Other methods of introducing powered wheelchairs to new users have included the introduction of one wheelchair function at a time [Nisbet (1995)]. The wheelchair response could be modified to introduce one control function at a time. The user would then be gradually introduced to the idea that the joystick caused movement. Next the wheelchair could move forward and backwards for example, and so on until the full function of the wheelchair had been introduced.

Children develop the expectation that movements can be controlled and executed at will [Butler (1986)]. Normal development introduces mobility in a way that allows cognitive and navigational skills to be acquired at a rate equivalent to maturity. A child who had not benefited from natural training processes may find the concepts of self initiated mobility difficult if they were suddenly available. Powered wheelchair manufacturers have reported that a child presented with a powered wheelchair for the first time finds the experience exciting and exhilarating but difficult [Stott (1997) and Stott & Sanders (2000b)].

A common way to introduce a new user to a powered wheelchair is to let them drive under supervision and under controlled conditions in a large uncluttered area. More demanding environments can be presented as skill levels increase. This could be achieved in virtual reality (VR) [Stott & Sanders (2000a) and Tewkesbury and Sanders(1995)].

A different problem was how to test novel algorithms and hardware as they were created during the work described in this Dissertation. If tests are simulated then a computer simulation could complete some initial testing. This could also be useful for user training and could be adjusted for individuals.

Wheelchair driving lessons have been vital to some new wheelchair users and have allowed a new user to gradually develop navigational and driving skills. The lessons could be time consuming and require highly trained staff to run them. Training sessions have sometimes been included in the cost of a new wheelchair and costs could be high. The use of VR could reduce these costs.

3.1 Teaching a new powered wheelchair user to drive.

Powered wheelchairs have often been presented to a first time user in an open space so that the user could safely explore the capabilities of the wheelchair. In the work of Nisbet (1995 & 2000) the wheelchair response could be modified to move forward a short distance when the wheelchair joystick was pressed. The user should eventually realize that the joystick caused movement. Next the wheelchair could be programmed to move forward, backwards and so on until full functionality of the wheelchair had been introduced.

VR has been investigated to explore alternative and safer methods for training [Inman *et al* (1997) and Woolhouse (1998)]. Suitable VR systems could relieve some of the burden on clinical or educational staff and users could gain extra "wheelchair hours" with less supervision. A typical VR system is shown in figure 3.1. The output could be a computer screen, a total immersion helmet, tactile devices with force feedback or a combination of outputs. Inputs could be any device that can generate an input signal to a computer and should emulate the input devices used by the wheelchair driver. Savings can be made if the new user does not travel to training centers and a more manageable training regime of regular but relatively short training sessions may be easier. VR can offer non-contact training programmes. A new user can try a new skill and experiment. Crashes are not expensive but the freedom to make mistakes must be balanced with incentives to avoid them. Methods for encouragement include; music, animated effects and progression to a new challenge on completion. Foreman (1993) used VR to simulate a wheelchair moving in the real world. A VR simulator was used to

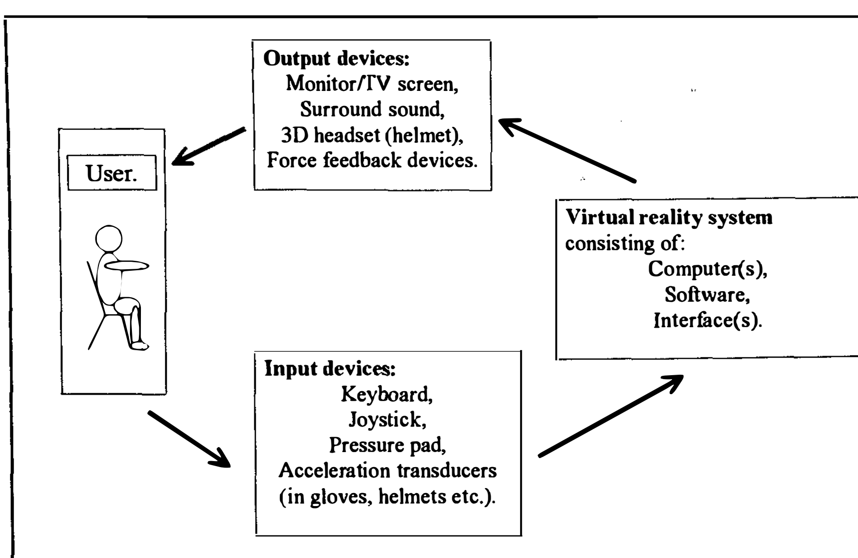


Figure 3.1. A typical virtual reality system.

investigate how disabled children could be taught the layout of a building. In later work, a building was recreated in VR [Stanton *et al* (1996)]. Disabled children used the two representations to familiarize themselves with a building.

The VR environment created during the work described in this Dissertation was partly designed to explore an alternative and safer method for wheelchair user training. VR has tended to be transportable, which could allow a user to train at home or school to reduce transportation costs and time. VR could provide valuable training without the danger associated with a new wheelchair user driving around a home. VR wheelchair trainers could also stimulate interest in severely disabled users by promoting interaction with the reward of music or pictures and by game playing.

VR environments have been shown to be suitable and effective for the conveyance of spatial information to disabled children [Stanton *et al*(1996)]. The VR world created in this work explored the possibility of using VR as a training tool for a wheelchair-driving simulator as well as for testing the novel algorithms [Stott & Sanders (2000a)].

The immersion of the user in the virtual world depended on the type of user interface available. A computer monitor was used as a common user interface. The monitor provided a limited immersion capability as the user could still see the real world around the edges of the monitor. A more effective user interface could have been a total immersion headset, which could fit over the head and cover the eyes. A total immersion set would have allowed the user to perceive three-dimensional shapes, images and sounds without distraction from the outside world [Zwern & Goodrich (1996)], but this was too expensive for the project. Foreman (1993) used virtual reality to simulate a wheelchair moving in the real world as a training system to investigate how disabled children could be taught the layout of a building. Systems of this nature could also be suitable to train and familiarise a potential user with an intelligent wheelchair.

A VR simulation of a wheelchair could be used in the following ways:

> Design Tool: Designs could be simulated and developed quickly and cheaply within the virtual world. Control algorithms could also be created and tested on different wheelchair designs.
>
> Educational and Training Tool: Environments could be constructed and modeled so that techniques of wheelchair control could be practiced without placing the user in any dangerous situations and reducing training costs.
>
> Communication Tool: The virtual world could be used to explain ideas and to convey information using three dimensional visual and audible media.
>
> Creative Tool: Purely imaginary objects could be constructed and animated and used to encourage or reward the disabled user.

For this work "Superscape VRT version 3.50" running on a DELL Pentium P60 personal computer (PC) was used initially to model the environment and wheelchair. Superscape VRT was capable of simulating suitable environments and a moving wheelchair within the environments. The software required no specialist equipment to run on a desktop PC. Superscape VRT was capable of basic simulation of the virtual wheelchair. It's capability as a dynamic simulation tool was limited. A different and more up-to-date virtual reality package, IGRIP, was investigated later in the research and some of the techniques developed using Superscape were transferred to IGRIP. Compared to Superscape, IGRIP was more suitable for producing graphical representations of the virtual environment and the control of the wheelchair movements was enhanced. The motion of the wheelchair was also smoothly simulated which improved the realism of the virtual environment. The wheelchair was able to perform simple control algorithms such as avoidance of an obstacle and moving safely through a doorway more accurately using IGRIP.

3.2 Immersing a user in VR.

Realism can be key to the success of VR. Achieving some realism can generate interest and assist in relating the virtual to the real world.

Immersion in VR depends on the user interface. A monitor can provide limited immersion, as the user could see the real world around the edges. This was demonstrated by the author on "Good For You" (Meridian TV 1995). A more effective user interface can be a total immersion headset to allow the user to perceive three dimensional shapes, images and sounds without distraction [Zwern & Goodrich (1996)].

Visual immersion was important but if more realism was required, sound and motion could be considered. To perceive the real world, a person has tended to use a combination of senses. Sight, sound and touch were typically used whilst moving from place to place. VR could provide these. An advanced headset could provide appropriate audio-visual stimuli. Force feedback devices appropriately connected could provide tactile sensing. Inman *et al* (1997) used heavy rollers driven by the wheelchair wheels to add mechanical inertia to the system. An advantage of this was that the input was through the wheels of the user's wheelchair, sounds were real and the wheelchair rocked and vibrated realistically.

Johnson & Aylor (1985) modeled a powered wheelchair to simulate dynamic response during the development of a controller. The center of mass of a powered wheelchair tends to shift as the user changes position, wheel casters can be unpredictable and external factors such as floor surface created random inputs. The model of the wheelchair could be added to the VR parameters. When coupled to force feedback joysticks and appropriate audio and visual stimuli applied, VR. could provide a realistic training environment.

3.3 Powered wheelchairs with intelligence.

An innovative assistive wheelchair has been created to improve the response and use of powered wheelchairs for the severely disabled [Stott (1997)]. Advanced and intelligent sensor and controllers have been created to assist a wheelchair user to steer [Bell *et al* (1994), Craig *et al*, (1995), Langner (1995 & 2000), Stott & Sanders (1996) & Stott *et al*, (1997)].

Simple artificial intelligence (AI) has been used by the author to interpret data from ultrasonic sensors and user input devices [Stott (1997) & Stott and Sanders (2000b]. Decisions were based on the wheelchair trajectory, the input device and data from sensors [Stott *et al* (2000)]. Testing regimes that required the real wheelchair were expensive and consequently VR was investigated.

Information from the user and ultrasonic sensors was simulated in VR. Algorithms mixed the two data sets so that the wheelchair could be driven safely without collision in the virtual world. The algorithms applied the following general rules:

- User to remain in overall control.
- Modification of the wheelchair trajectory when necessary.
- Wheelchair movements to be smooth and controlled.

Some general notes on modeling are in Sanders (1998).

An extension of the methods by Gilbert & Johnson (1985) and Khatib (1986) was tested in VR. Obstacles were represented by distance functions. For example the mover and obstacles could be represented by positive charges. This artificial potential repulsion approach was aimed at real time local avoidance. The function tended to infinity as the moving point approached the surface and was zero beyond a certain distance. Appropriate torques were simulated to follow trajectories generated by a human operator and the force from the artificial potential field was incorporated to generate torques at the wheels. This allowed the wheelchair to follow the required trajectory closely while avoiding collision. The role of the artificial potential field was not to plan a trajectory, but to bend it away from obstacles such as doorposts. The avoidance problem was realised at the lower control levels for real-time execution and were mixed with the real time information from a disabled wheelchair user in order to assist the user.

Path planning has been divided into two categories, local and global methods [Stott (1997), Balding (1987), Sanders (1993) and Sanders & Stott (1999)]. Local methods use algorithms that find a path by repetitively considering configurations that are closer to the goal. When obstacles are encountered then alternative strategies are attempted, such as "MOVE LEFT". The advantage of these methods was that planning could take place when it was not possible to have a global world model. This was the case in this work.

The home is a complex environment subject to randomly scattered moving and movable objects. The example maneuver of driving a wheelchair through a doorway was selected as an example as it was a complex and common exercise for a wheelchair user. A doorway in this work was considered to be any gap through which a wheelchair may pass and could be of variable width. For this example the sensors had four possible modes of detection:

i) No detection by either sensor.

ii) Left sensor detected and the right sensor did not.

iii) Right sensor detected and the left sensor did not.

iv) Both sensors detected.

These detection modes could be interpreted by a microcomputer in the following ways:

i) No detection by either sensor; the wheelchair was moving in free space and no action was necessary.

ii) or iii) One sensor detected and the other did not; the wheelchair was moving towards an isolated object or was near to one side of a doorway.

iv) Both sensors detected; a single object was detected by both sensors, for example a single isolated object or a linear object (wall) or each sensor detected a different object such as two isolated objects or the opposite sides of a doorway.

In the case of iv) where both sensors detected an object, the case could also be specified as:

iv.a) Both sensors detected an object at equal distance from the array. This indicated a "square on" approach to a wall, an isolated object on the axis of the wheelchair or that the wheelchair was moving centrally through or towards a doorway.

iv.b) The sensors detected an object at different distances from each array. This indicated an angled approach to a wall, an isolated object in the path of the wheelchair but not on the wheelchair axis or the wheelchair moving at an angle through a doorway or not on the centre line of the doorway.

The objective was to help the wheelchair to avoid contact, steering gently away from the object. In the case of both sensors detecting an object, reducing the speed of the wheelchair was an advantage.

Two software elements (denoted as cells) were invented in order to model the environment. Each cell (Left),(Right) contained a certainty value C(Left) and C(Right) that indicated the measure of confidence that an obstacle existed within the cell area. The greater C(Left) or C(Right) the greater the level of confidence that the cell was occupied. If an obstacle produced a reading from the sensor sweeping that cell then the corresponding cell contents C(Left/Right) were incremented in proportion to the distance measured to the obstacle. Random misreading did not cause an excessive reaction by the wheelchair. This method provided a reliable representation in spite of sensor inaccuracies.

3.4 The initial virtual reality prototype.

To test some of the ideas presented in this Dissertation, a VR prototype was interfaced to a powered wheelchair. A virtual world and wheelchair was created using Superscape VRT 3.50.

3.4.1 Superscape VRT 3.50.

Figure 3.2(a) shows a simplified block diagram of a wheelchair before connection to the VR. interface. Figure 3.2(b) shows the powered wheelchair

connected to the VR. computer. The joystick was connected directly to the computer and the wheelchair controller was bypassed. A simple virtual world was created using "Superscape VRT version 3.50". The virtual world was a flat plain and a building for a virtual wheelchair to move into. The virtual world was displayed on a computer monitor. The virtual world is shown in figure 3.3.

A virtual wheelchair was created and imported into the virtual world. The virtual wheelchair was modeled on a "Bobcat II" powered wheelchair (shown in Figure 3.4) as this was later used for testing the real world intelligent wheelchair in Chapter 5 and Chapter 6. The virtual wheelchair is shown in Figure 3.5.

A virtual sensor system was required to indicate the proximity of an object. In the

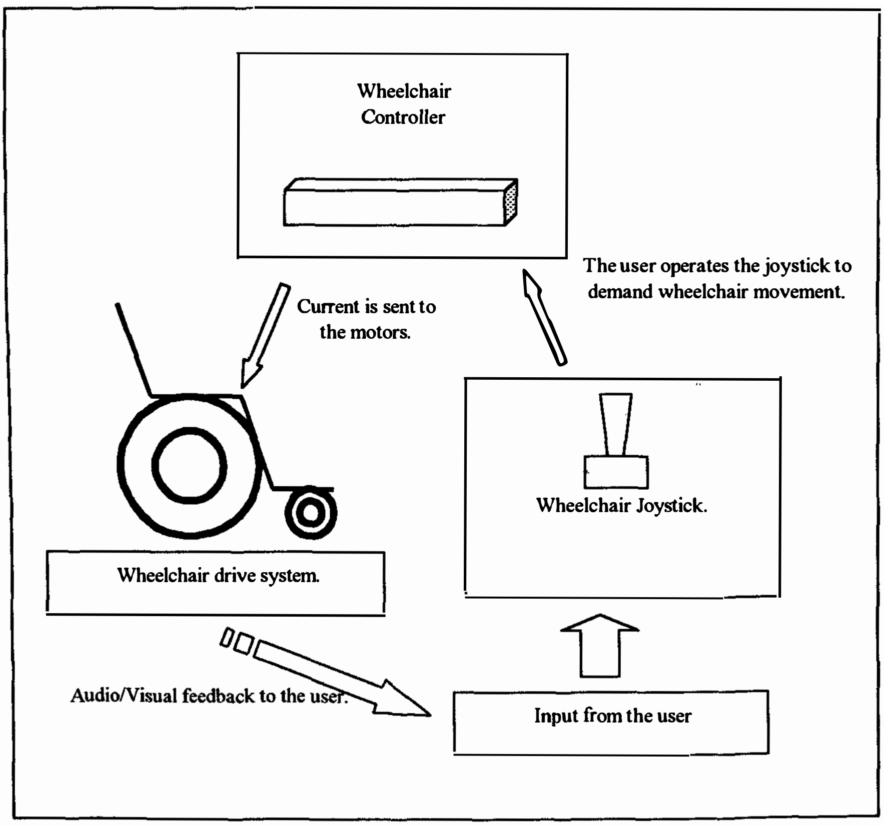


Figure 3.2(a). The unmodified wheelchair system.

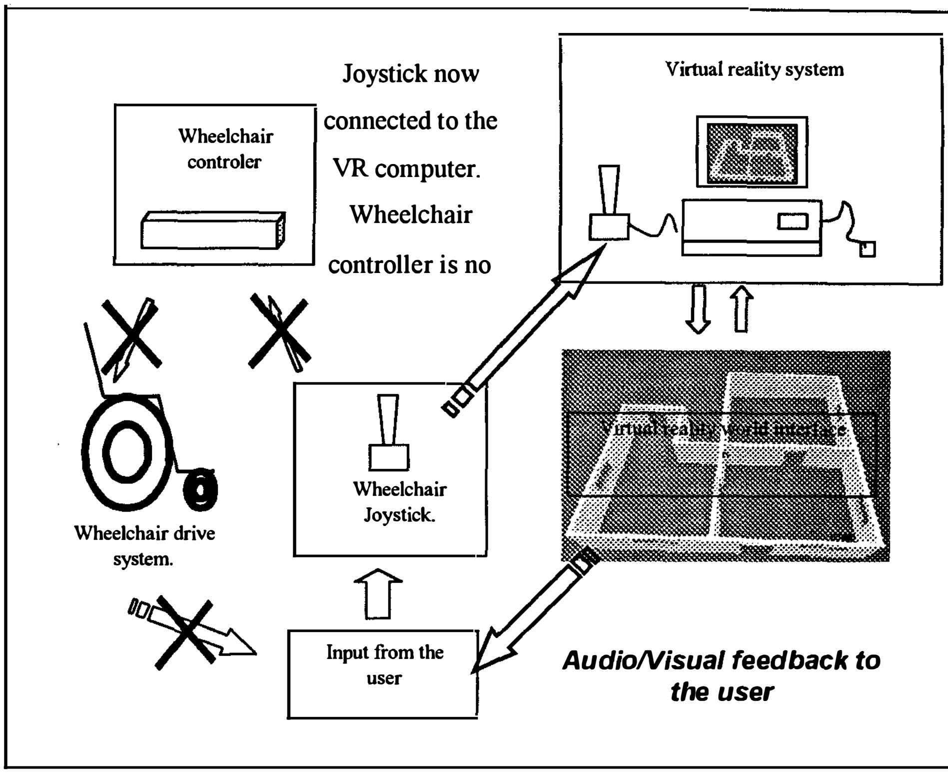


Figure 3.2(b). The wheelchair integrated into the VR system.

virtual world all objects existed at a known set of global co-ordinates, these co-ordinates were used to test the distance of the virtual wheelchair to the object. Using a simple distance function [Stott & Sanders (1996) and Stott (1997)], the wheelchair trajectory was modified to avoid an object. The movements of the wheelchair simulated the real wheelchair although it was difficult to re-create the dynamic model of the wheelchair within the virtual world.

Algorithms for the virtual wheelchair were based on:

$$\underline{Output}(left) = \underline{Input(left)} - \underline{F(right)}$$

Output(right) = Input(right) - F(left)

where Output, Input, and F were all vector quantities, each having two values, one for each wheel (left and right). This output was presented to the wheelchair controller driving the wheels for the real wheelchair and was adapted for the virtual wheelchair . The effect of this algorithm was to turn the wheelchair gently away from an object or, if the trajectory was "square on" to the object the wheelchair would gradually slow down.

3.4.2 Creating a Virtual World.

A virtual world was created for a virtual wheelchair to navigate through [Stott & Sanders (2000a)]. The virtual world consisted of a flat plain with a roofless building for the virtual wheelchair to navigate. This is shown in Figure 3.3. The creation and manipulation of virtual objects used two of the editors available in Superscape: the Shape Editor and the World Editor. The Shape Editor defined the shape of objects; in this case the walls. The World Editor was used to place objects within the virtual world. Both editors were combined to form a visualiser in which the virtual world could be displayed on the user interface; in this case a computer monitor.There were other editors which could be used to add texture, sound, dialogue and to customise the screen layouts of the virtual world. These other editors were not used in this work.

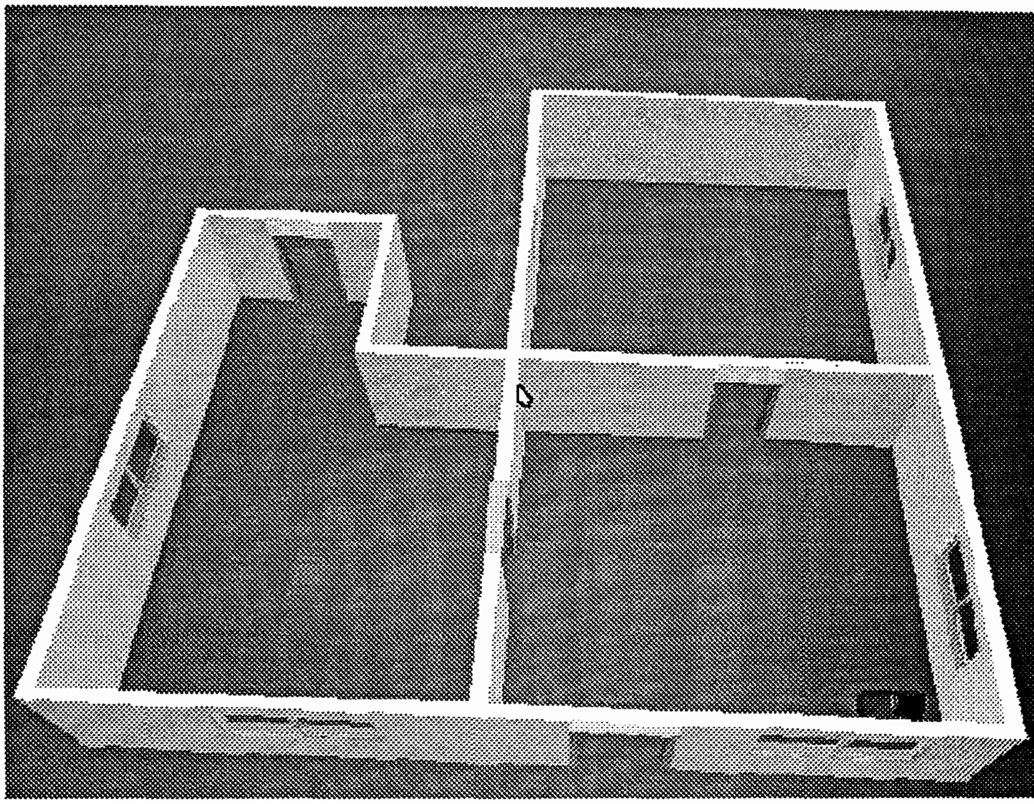


Figure 3.3. A virtual world created for the initial prototype system.

3.4.3 Shape Editor.

All shapes in the virtual world were created using the Shape Editor. Points were fixed in Cartesian space and then linked with other points to define a surface or two-dimensional facet. Additional facets could be connected together to form a shape. The Shape Editor could copy, duplicate, extrude and transform features of a shape. Colour and animation could also be added to the shape using the Shape Editor.

It was not possible to produce curves within the Shape Editor, only straight lines. Where curves were required on a shape, such as the rear compartment on the virtual wheelchair, many small flat surfaces were created and joined. If the facets were not ordered correctly then the shape did not look convincing. Facets at the back of a shape were ordered first. It was important to accurately represent the real world or the impact of the VR environment on the user could be lost due to a lack of realism.

3.4.4 World Editor.

Shapes were exported from the Shape Editor into the World Editor. The shapes were then given attributes to produce "objects". Objects were manipulated to produce a virtual world. It was then possible to create, position, size, colour, bend, rotate and animate objects. The attributes of objects could be controlled by attaching Superscape Control Language (SCL) code to the object.

Groups. Objects had invisible boundaries around them known as 'Groups'. It was possible to toggle the groups on and off so that they could be seen or be hidden. Groups could not pass through each other unless they were placed within each other or were defined as "enterable". A moving group tended to stop upon meeting another group. A group could be given an attribute such as movement, rotation or position. This was useful when an object was to be controlled by the user; for example control of the virtual wheelchair by joystick.

Relations. Each object within the World Editor was connected to a "family" tree structure. The objects were defined as: parents, children and siblings. Each object was related to its parent, for example a virtual person created within the virtual world had a parent - the environment, surrounding it. It had children which could be the left and right arms. Each arm was considered to be a sibling of one another. Similarly, the virtual person could have had more children, for example legs, ears and eyes. Children could have more children.

Objects could be 'adopted' within the virtual world and the position of a child was relative to the parent position.

3.4.5 Visualiser.

The Visualiser allowed the user to see the virtual world and to interact with it. The Visualiser was used to simulate the user entering the virtual world. The user could move around the virtual world and inspect objects and shapes from any angle. The Visualiser gave the virtual world some realism. The Visualiser could be set so that the environment was viewed from the seat of the virtual wheelchair.

3.4.6 Key Editor.

The user could define quick keys in any of the editor modes including the Visualiser. The user was then able to quickly perform repetitive tasks such as reset and switching between modes with the use of one key.

3.4.7 Superscape Control Language (SCL).

Superscape Control Language (SCL) was similar in structure to the "C" programming language. SCL could be used to control objects within the virtual world. Tasks such as rotation and movement were controlled by SCL code attached to objects. SCL could be created and attached to any object within the World Editor. SCL could be attached to an object and then used to control a different object. SCL code was executed once per frame and could be programmed to change the properties of the object. For example, code could be attached to make objects bounce off each other if they collided or to alter velocity depending on other factors; for example to slow the wheelchair when approaching an obstacle.

3.4.8 The virtual wheelchair.

A virtual wheelchair shown in figure 3.4 was created, modeled on the Bobcat II shown in figure 3.5. It was constructed by producing suitable shapes in the Shape Editor and then exporting and grouping them together within the World Editor. The wheel shapes were produced using linear segments. An eighteen-sided polygon

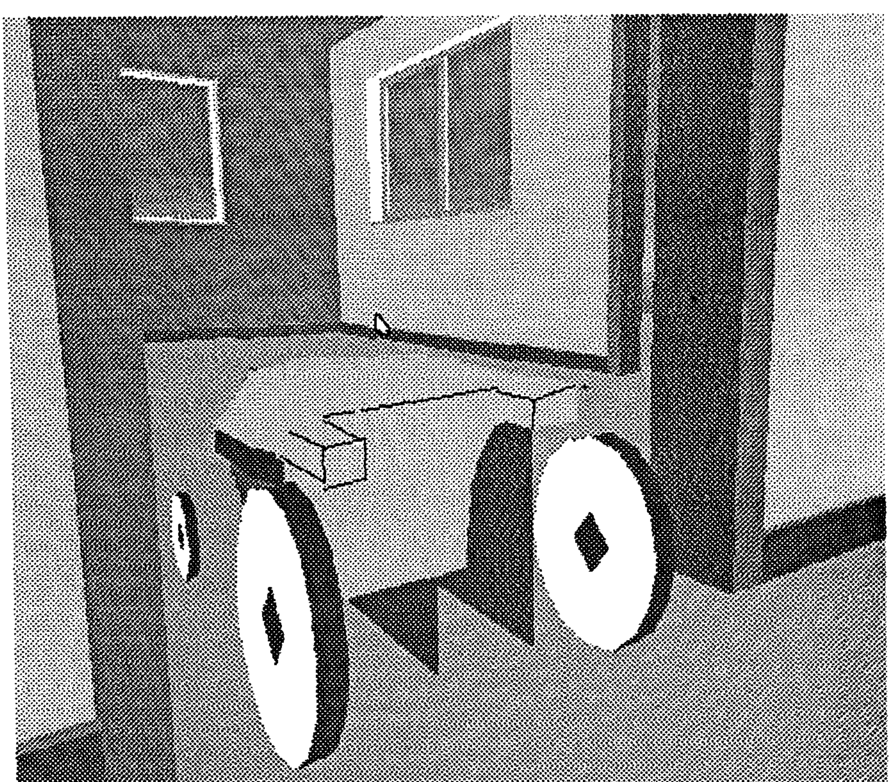


Figure 3.4. A Bobcat II powered wheelchair.

Figure 3.5. A Bobcat II powered wheelchair.

produced an acceptable representation of a circle. A single point was placed in space. Eighteen points were placed in a radial Y-plane around the initial point. The points were then connected to produce a circular-like facet. The facet was then duplicated and placed alongside the original. By connecting the copied points to the originals in the Z-direction, a shape of a wheel was formed. A square shape was placed at the centre of the wheel to provide a visual reference so that the wheel could be seen to rotate.

The shape was given colour and then exported to the World Editor. The wheel shape was copied three times and two wheels were increased in size by 150% to simulate the larger driving wheels. The four wheels were attached to a wheel group.

3.4.9 Control of the Wheels.

Two groups were created around the wheels: an outer group - the grand parent and an inner group - the parent. The wheels were the children of the inner group.

A **rotation** attribute was added to the inner group so that the wheels would appear to rotate and a **dynamics** attribute was added to the outer group so that the wheels would appear to move along. The inner group was 'coupled to' the outer group using the Dynamics Menu, and a 'movable flag' was placed on the outer group. This allowed the whole group to be moved.

The SCL code required to rotate the wheels was:

yangv(me)=10

Where:

yangv defined a rotation function.

*(me)*specified that the code should be applied to the object to which the code is attached.

=10 defined the angular velocity (frames per revolution).

A viewpoint was attached to the wheels so that when the wheel group was moving the user could move with the group using the Visualiser. This could give the user the impression of sitting inside the wheel group. The wheel group could then be moved in a direction determined by the cursor keys or the joystick.

3.4.10 Virtual bodywork for the wheelchair.

The Shape Editor was used to create an acceptable representation of the virtual wheelchair bodywork. The shapes were then exported to the virtual world in the World Editor and attached to the wheel group. The virtual wheelchair was then complete and could be navigated through the virtual world using the viewpoint to drag the virtual wheelchair or could be programmed to move using SCL code

3.4.11 Control of the velocity of the Superscape virtual wheelchair.

The virtual wheelchair could be dragged along by attaching the viewpoint to the wheelchair group and operating the cursor keys. The viewpoint moved at a constant speed and was not controllable whilst the virtual wheelchair was in motion. This was because the virtual wheelchair was controlled by the cursor keys which had no facility to modify velocity, only to indicate a direction and to start or to stop the virtual wheelchair.

Using SCL code it was possible to control velocity and routines were written to demonstrate control over the virtual wheelchair. A virtual sensor system was required to simulate a wheelchair fitted with a sensor system. The virtual sensor system needed to react dynamically with the environment. In the real world objects were detected physically. In the virtual world, objects were defined by software code and labeled. All objects within the virtual world were defined by SCL code and named. The SCL code defined the shape, orientation and position of the object by global coordinates. In a virtual world cluttered with objects it was difficult to identify objects in the path of the virtual wheelchair.

Early programs contained only one object and this simplified detection. When the single object had been detected and the position and shape read then it was possible to program the virtual wheelchair to react to the object. Code to slow the virtual wheelchair down and stop in front of the object was:

```
Fixed a;

        a=(zpos('CUBE')-zpos('buggy-hold'));

        zdrive('buggy hold')=(a-10000)*0.005;
```

Where:

Fixed a;	Variable "a" used for distance from the virtual wheelchair to the object.
*(a-10000)*0.005;*	Reduces the value of zdrive - the velocity of the virtual wheelchair.

A virtual sensor system would be required to detect objects in the path of the virtual wheelchair. In this example the object was known and was called CUBE. In a more complex and realistic world the name of the object in front of the virtual wheelchair may not be known and so the distance to it may not be calculable.

A simple collision avoidance algorithm was created to enable the wheelchair to avoid colliding with an object in its path. The program monitored the distance between the wheelchair and the object and applied a force to the wheelchair perpendicular to its trajectory. The effect was to move the wheelchair to the side of the object so that it could pass without a collision. The following code was used:

```
If((Zpos('object1')-Zpos(wheelchair))<10000):

Xdrive = Zdrive*0.5;
```

Where:

If((Zpos('object1')-Zpos(wheelchair))<10000):	Tested the distance between the object1 and the wheelchair.
*Xdrive = Zdrive*0.5;*	Moved the wheelchair sideways.

The program caused the wheelchair to move diagonally away from the object. The wheelchair did not turn when the algorithm was invoked so the image on the monitor did not look convincing as it seemed to slide to the side rather than turn and drive to the side. The program also only worked for previously defined objects.

A simple prototype virtual sensor system was created during this work. The code required for a fully functional virtual system would have been complex and time consuming to produce and beyond the scope of this research. The work did indicate some methods by which such a system could be produced and these are discussed in the following sections.

3.4.12 Results.

Figure 3.6 shows the view from the driving seat of the virtual wheelchair whilst negotiating a traffic cone in VR. Detection of other objects was difficult. The most effective method used in this work was to know the position of objects by their global co-ordinates and to test the wheelchair position. The positions of all the objects in the virtual world needed to be recorded and the possibility of the virtual wheelchair colliding with any of them calculated. If an object were identified as being in the path of the virtual wheelchair, avoiding action could be calculated and executed. This allowed the novel algorithms and innovative wheelchair hardware to be simulated in VR.

When more than one object existed within the virtual world, the code became complex. For ease of description a single object was used as an example (Figure 3.6). As the wheelchair approached the object, code was activated to guide the

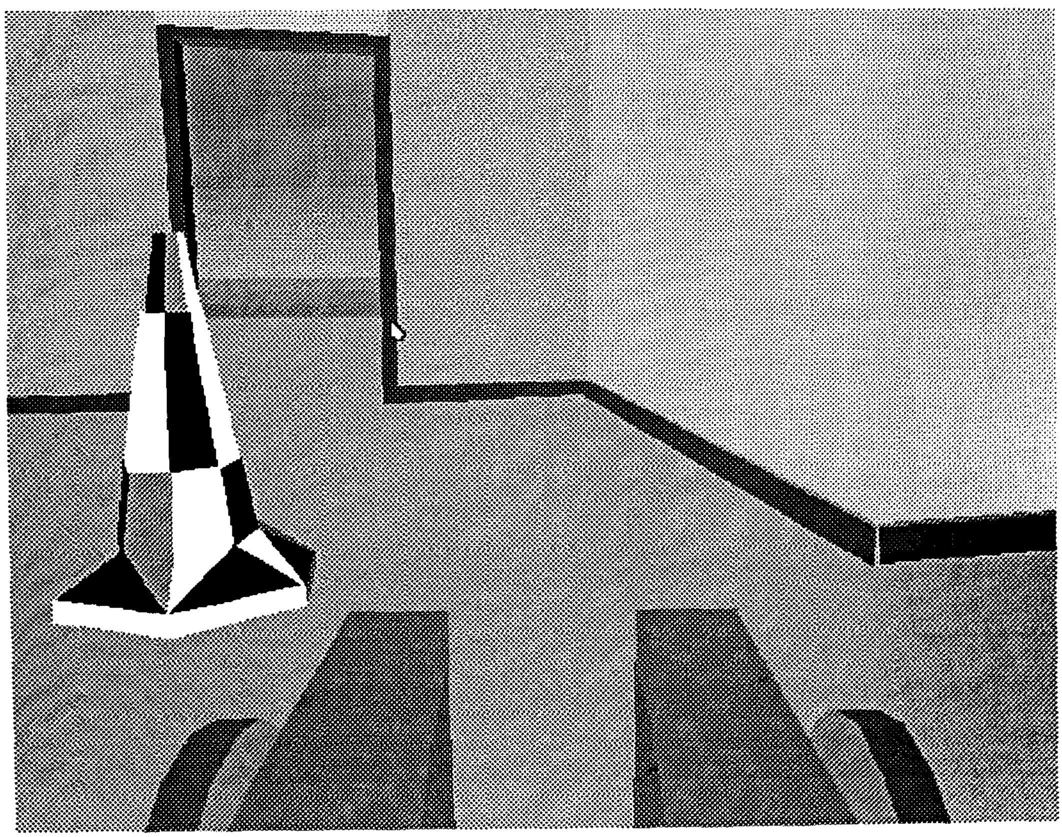


Figure 3.6. The view from the virtual wheelchair seat.

wheelchair onto a trajectory to avoid the colliding with the object. An example of Superscape Control Language (SCL) is shown below.

```
Fixed a;

        a=(zpos('CONE')-zpos('CHAIR-hold'));

        zdrive('CHAIR-hold')=(a-10000)*0.005;
```

Where:

Fixed a;	Defines “a” as the distance from the chair to the object.
*(a-10000)*0.005;*	Reduced the velocity (zdrive) of the virtual wheelchair.

This SCL code invoked a simple collision avoidance action by the virtual wheelchair. As the virtual wheelchair approached a “cone” in the virtual building, the velocity of the virtual wheelchair reduced to zero in proportion to the distance from the virtual wheelchair to the cone. This example was a model of a similar function that the real wheelchair had performed in the real world. It was designed originally as a function to stop a real wheelchair from involuntarily running into a wall or other object.

Other more complex algorithms were considered such as; object avoidance and automatic doorway navigations. The limited dynamic function of the Superscape virtual wheelchair made realistic simulations of more complex algorithms difficult. It was possible to pre-program the virtual wheelchair to follow pre-determined paths. This approach was not considered within the scope of the work described in this Dissertation and was discontinued in order to investigate the IGRIP platform described in the next Section.

3.5 The second prototype virtual reality system.

The second prototype virtual reality system was created using IGRIP version 5.0 simulation environment [Deneb (1999)]. A Process similar to that described for the Superscape model was used to generate a wheelchair simulation and virtual environment.

IGRIP provided an interactive, 3D graphic simulation tool for design, evaluation and analysis. Any manufacturing process could be constructed, programmed and analyzed for cycle time, collisions and motion constraints. IGRIP consisted of three primary systems:

(a) Menu (User) System.

(b) Graphic Simulation Language (GSL).

(c) Command Line Interpreter (CLI).

(a) Menu (user) system.

IGRIP provided a mouse driven, point and click graphical user interface (GUI). There were eleven major components (contexts) of the *IGRIP* menu system shown in Fig.3.7 and these included:

- **CAD** contained features for creating a 3D visual representation parts.
- **DEVICE** was the context for creating devices by attaching parts together. It provided features for assigning kinematics and attributes such as degrees of freedom, speeds, acceleration rates, dynamic properties and travel limit to the device.
- **LAYOUT** was used to assemble the devices together and features that allowed the user to connect the necessary I/O signals.
- **MOTION** containd functions used to run simulations of a device.
- **PROG** was the program context used to generate program using on-screen menus to script the syntax automatically into the device's program.
- **DRAW** was a two dimensional world context that allowed the user to create simple geometry that could be extruded into three dimensional parts.
- **ANALYSIS** assisted in identifying various items in the world, as well as determining the distances and angles between them.
- **SYS** was the system context that provided the ability to define system attributes such as world view (lights, grid, background and colour) and files (creating

directories, printing and configuration file management).

(b) Graphic simulation language (GSL).

GSL was a programming language used to control the behaviour of devices in the workcell. GSL incorporated conventions commonly used in high-level computer languages. Specific enhancements for device motion and simulation environments were included within GSL.

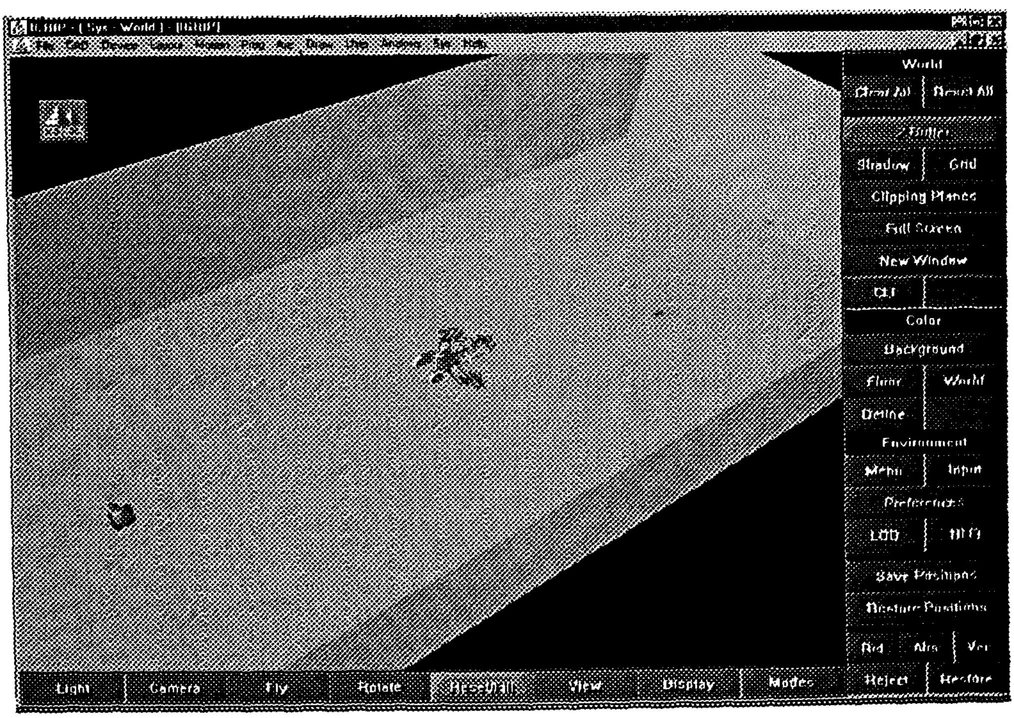


Figure 3.7. A general view of the user menu system available in IGRIP.

(c) Command line interpreter (CLI).

CLI was a powerful communication, command and control system that interacted with the IGRIP simulation environment. It was a "control" language in the sense that it behaved as a "user" outside of a running simulation. It could load a workcell, manipulate devices, paths and parts, and run simulations. It differed from GSL, which focused on the motion of a particular device within a simulation.

High quality graphical representation of the wheelchair was important to reflect the realism of the computer simulation. The powered wheelchair consisted of two large driven wheels mounted to the front of the wheelchair chassis and two smaller caster wheels mounted to the rear of the main chassis. A traffic cone representing an obstacle was created and placed with the wheelchair within a roofless, single floor building.

Figure 3.8 shows the relationship between each object within the workcell. A workcell (virtual environment) consisted of devices (wheelchair, obstacle and rooms for example) and each device such as "wheelchair" was constructed by attaching parts together such as the front wheels, rear wheels and chassis.

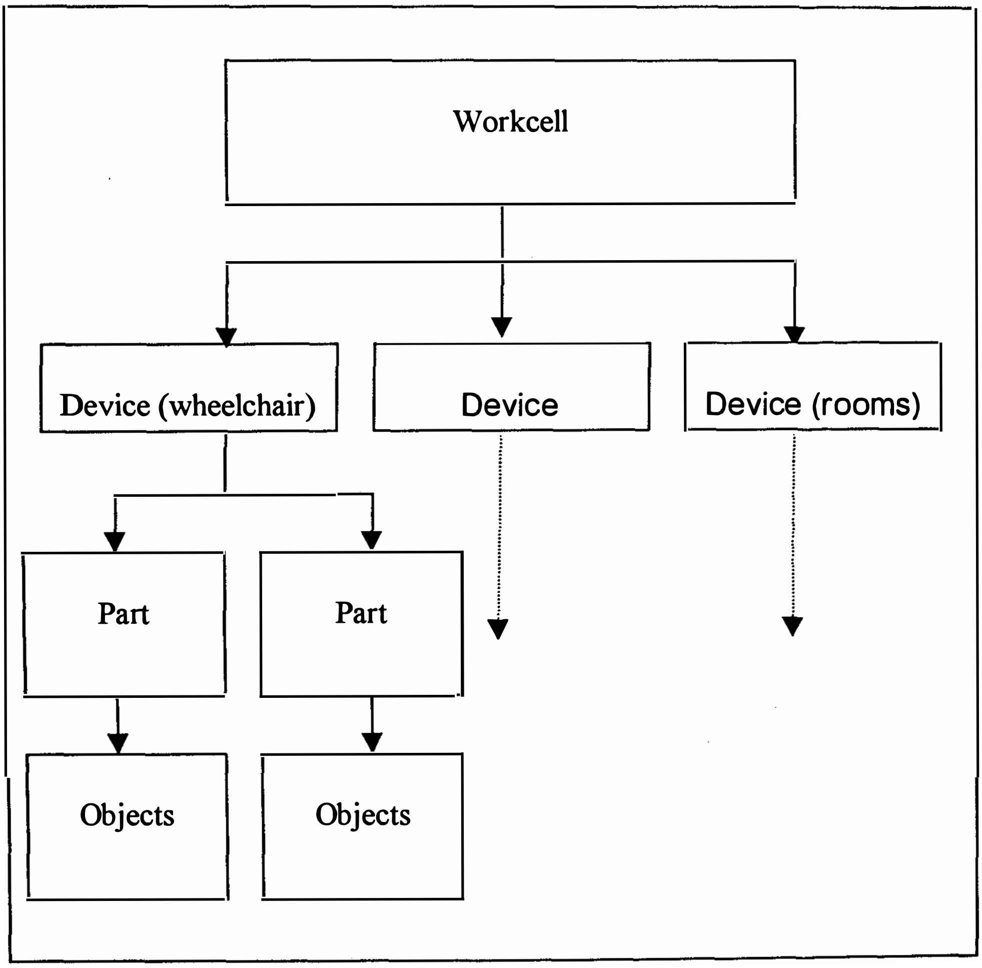


Figure 3.8. The relationships between parts in the workcell.

The part consisted of one or more objects. Simple objects were created using CAD primitives provided by the software such as block, cylinder, cone, wedge and pipe.

When parts were created, files were saved individually in case of mistakes because an "undo" function was not available within IGRIP and should a corrupted file have been saved then work may have had to start again. When the parts for each device had been constructed, they were integrated to produce a device (wheelchair). The wheelchair device was then assigned six degrees of freedom using a simple kinematics function. The wheelchair was then visually complete and ready to be programmed. The completed wheelchair device is shown in figure 3.9. The obstacle and rooms were built in a similar way to the wheelchair and are shown in figure 3.10.

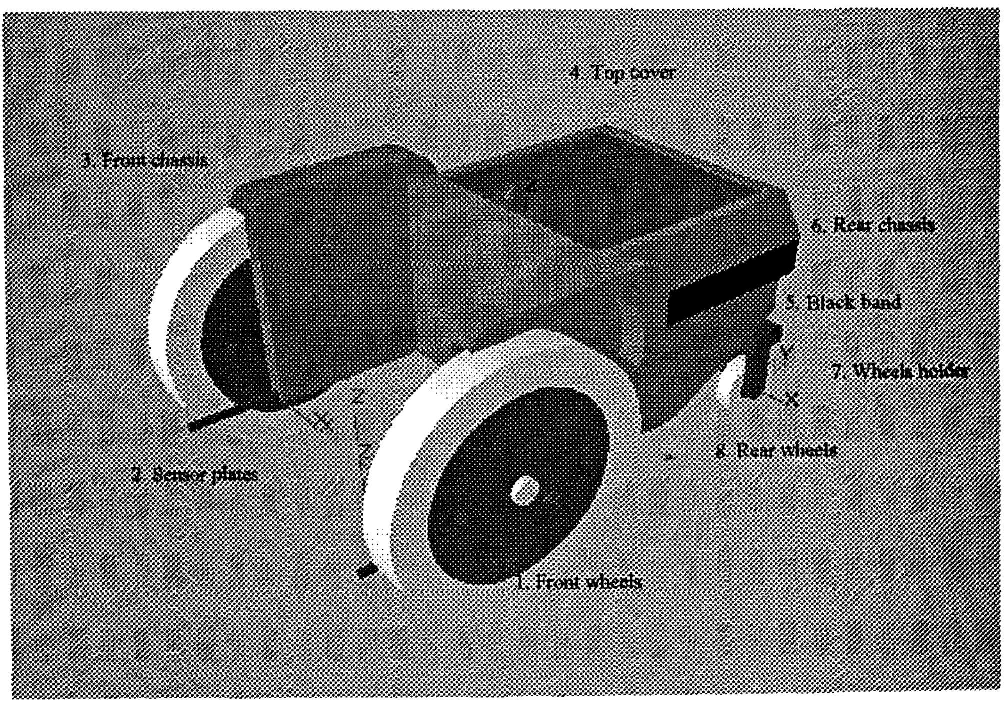


Figure 3.9. The completed IGRIP virtual wheelchair.

3.5.1 The virtual sensor system

A virtual sensor system was created to simulate the range finding ultrasonic sensor system fitted to the real wheelchair. A method was required to detect and calculate the distance to an object from the virtual wheelchair. IGRIP contained a function called raycast. Raycast was a function that simulated a "ray" (which could be visualised as a laser). The ray was emitted from a user-defined co-ordinate within the virtual world. The co-ordinates could be defined in terms of the global co-ordinates or local co-ordinates. The virtual wheelchair "owned" its own set of local co-ordinates and the raycast could be defined from them. This meant that the raycast could travel with the wheelchair. This simulated a sensor system attached to a real wheelchair. The raycast beam continued through the virtual world to infinity but the maximum effective range could be limited by the programmer. The raycast function returned a virtual object's location and distance from the raycast base. The raycast function was used to simulate the real wheelchair's sensor system. The results from the virtual sensor system are discussed in Chapter 6.

3.5.2 Results from the second prototype VR. system.

The wheelchair model was created to the dimensions of a "Bobcat II" paediatric wheelchair to increase the realism of the virtual reality simulation (figure 3.4). The virtual wheelchair device was constructed from 8 main parts; front wheels, sensor plates, front chassis, top cover, black band, rear chassis, wheel holder and rear wheels.

A simple environment consisting of three roofless rooms, a wheelchair and an obstacle were created for the simulation. The environments appearance was not

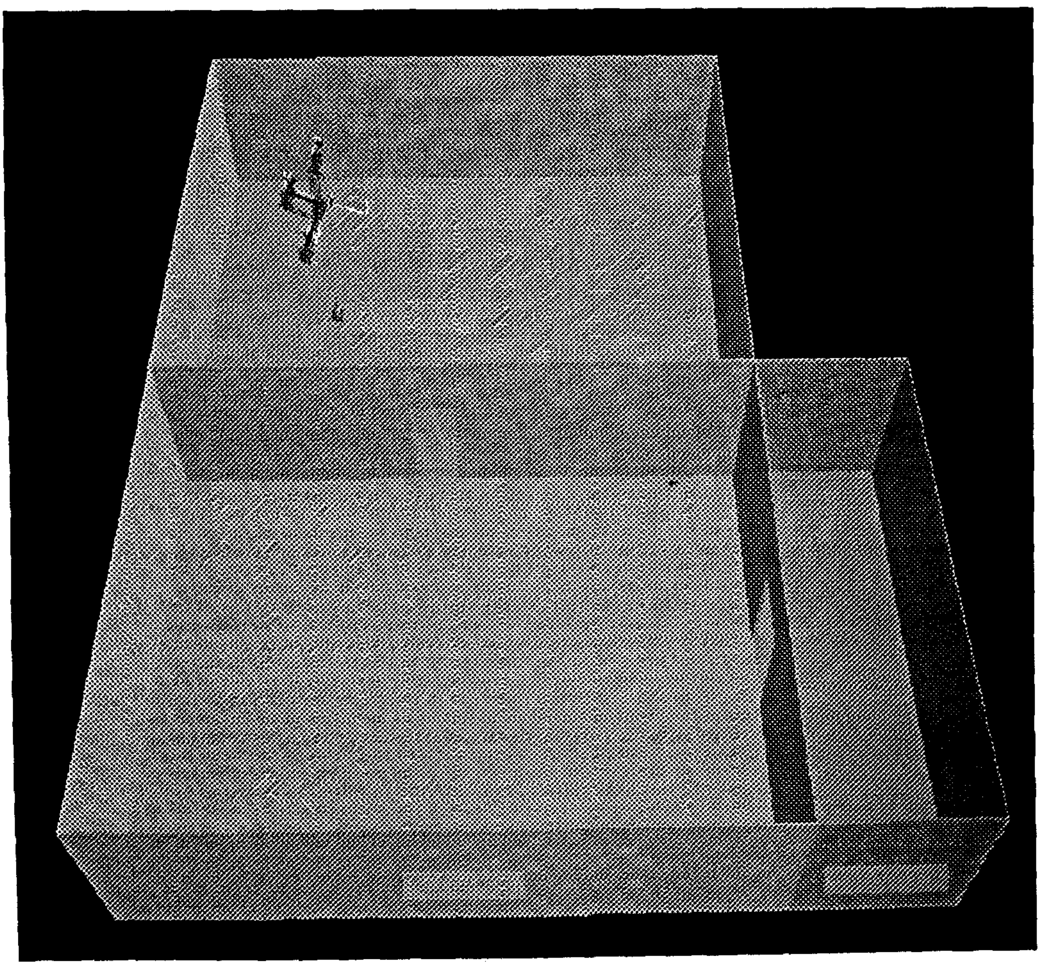


Figure 3.10. The IGRIP virtual environment.

made too complex but kept to a minimal design as it was more important to use IGRIP to test algorithms. The more complex the environment, the slower IGRIP would update the simulation. This could lead to the virtual wheelchair and environment moving in a series of jerks on the screen rather than in a smooth manner which was desirable for a realistic simulation.

A virtual sensor system was created which could accurately simulate the ultrasonic sensor system of the real wheelchair. Features of the real sensors such as beam width and maximum range could be simulated on the virtual sensor system. As the virtual environment could be detected by the virtual wheelchair system and it was possible to model the dynamics of the real wheelchair, an accurate simulation of the real wheelchair was created.

Algorithms used during the virtual reality tests were modeled on algorithms created for real wheelchairs. An algorithm was created in GSL which was based on a "Smart wheelchair" control algorithm as part of user training. The wheelchair was allowed to be driven by the user under instruction until the wheelchair sensed an obstuction in its path. The wheelchair would stop, move back, turn to the left and wait for the user to drive the wheelchair forward again. The GSL code to complete this manouver is shown below.

```
if (dist_r <= 200) then
        MOVE JOINT 2 BY 500 NOSIMUL
        MOVE JOINT 4 BY 10 NOSIMUL
endif
```

If the distance from the front of the wheelchair to the obstruction was less than 200 units, the wheelchair stopped and moved 500 units in the Y-axis direction (backwards) and then rotated by 10 units in Z-axis (Yaw). A schematic diagram of the effect of this algorithm is show in figure 3.11.

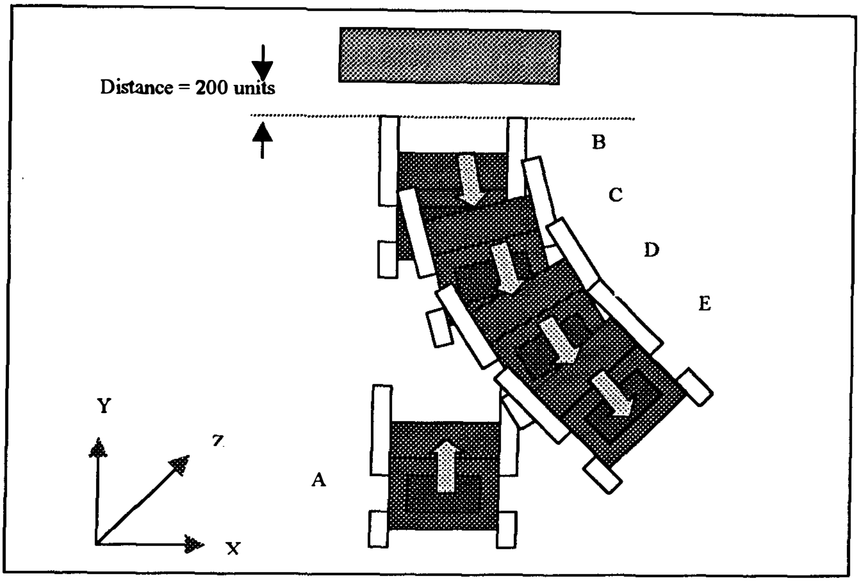


Figure 3.11. A stop, reverse and turn algorithm.

Another algorithm was implemented on the virtual reality system so that instead of reversing automatically, then turning, the wheelchair was programmed to decrease its speed and turn away from the object to the left or right. A GSL program based on this algorithm was tested on the virtual reality wheelchair and is shown below.

```
if (dist_r <= 2000 ) then
        s = s – ((dist_r – 200) / 360)
        l_c = l_c + ((2000 – dist_r) / 10000)
        MOVE JOINT 2 BY s SIMUL
        MOVE JOINT 4 BY l_c NOSIMUL
endif
```

During this algorithm, the distance to the nearest object was tested. If the distance was less than or equal to 2,000 units, the forward speed of the wheelchair was gradually decreased and the trajectory altered to turn the wheelchair away from the object. The algorithm used distance functions (dist_l and dist_r) to move the wheelchair away from the object in the correct direction. The schematic of the wheelchair trajectory is shown in figure 3.12.

Figure 3.12 shows the wheelchair gradually turning to the left as the sensors detected an obstacle within the predefined range. The user continued to push the joystick in a forward direction and did not change the joystick position.

Chapter 6 discusses the performance of the virtual wheelchair system.

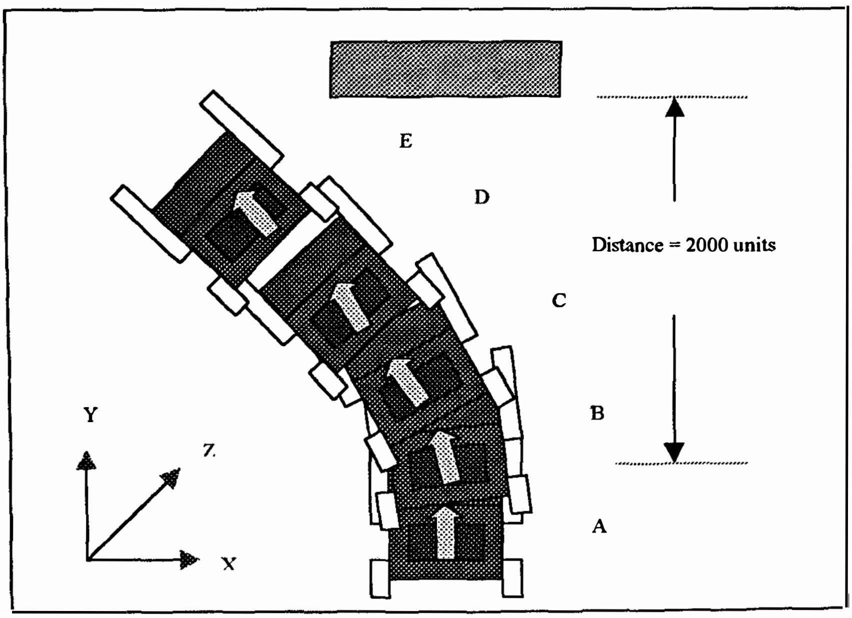


Figure 3.12. An algorithm to slow and turn the wheelchair.

3.5 Discussion.

The use of VR on Superscape demonstrated a method for simulating an intelligent wheelchair system and of training intelligent wheelchair users. As a training system, potential wheelchair users may be introduced to the concepts of mobility without danger or injury. The wheelchair training system could be tried with the minimum of supervision. A VR training system can be based on a personal computer and could be transported and used in non-specialist locations. This could save a potential user from having to travel to special locations to learn to drive a powered wheelchair. Savings in cost and manpower could be achieved with the use of VR.

VR has also been shown to be a useful tool for developing rehabilitation systems and algorithms. Novel algorithms could be tested in VR without the need for real-world trials so algorithms could be created and tested quickly and cheaply. Initial bugs and problems could be identified early in the development process and eliminated before any hardware had been programmed with the algorithms.

Several children tested the first prototype VR system, one of whom suffered from cerebral palsy [Meridian TV 1995]. The results suggested that VR could be used for training a powered wheelchair user. The quality of the user interface used in this work was limited, as it was a computer screen, but it did provide a representation of the virtual world.

The virtual model of the virtual environment created by the *IGRIP* software showed a high quality of graphical simulation. This was important to achieve a realistic simulation. More importantly to this work, the dynamic modeling of a wheelchair made IGRIP a practical choice for a VR wheelchair simulator. Functions of the software made the representation of real-world systems possible. The simulation of the ultrasonic sensor system was an example of the flexibility of IGRIP. The second prototype VR system had many advantages over the first prototype. It must be noted that the second prototype was created using a recent version of the IGRIP software. In comparison the first prototype was created on an older version of Superscape VRT.

Chapter Four

HARDWARE

A modern powered wheelchair called a Bobcat II was selected as a test rig for this work and was made available for the research work by QED Ltd, Gosport [QED (1995)]. A photograph of a Bobcat II powered wheelchair fitted with a child's seat is shown in figure 4.1. The wheelchair base was modified so that new experimental equipment could be attached to test the novel intelligent wheelchair control algorithms. The new hardware consisted of a micro-controller, interfaces, the wheelchair power base and a

Figure 4.1. An unmodified Bobcat II powered wheelchair.

Penny and Giles PG8 series digital powerchair controller, associated analogue interfacing and a wheelchair joystick. The final prototype configuration is shown in figure 4.2. The hardware for the new intelligent wheelchair prototype was required to:

- Interface a micro-controller to the wheelchair controller to allow rapid prototyping of algorithms and testing of responses.
- Sense the environment around the wheelchair.
- Interface a joystick monitor to the micro-controller so that the wheelchair could be driven by "fly by wire".
- Preserve the “normal” operation of the wheelchair.

To achieve these objectives the wheelchair was modified to include a sensor system and a micro-controller interface. The joystick connection to the wheelchair controller was preserved and the interface connected so that the joystick signals could be monitored by the micro-controller. A new sensor system was built and connected to the micro-controller to allow the sensors to be controlled and operated. The wheelchair motor control lines were connected to an interface so that the micro-controller could modify the wheelchair motor speeds if necessary. This prototype allowed the wheelchair to be driven under micro-controller control, or for the wheelchair to be driven as an unmodified wheelchair.

The micro-controller, interface and sensor boards were designed and constructed as part

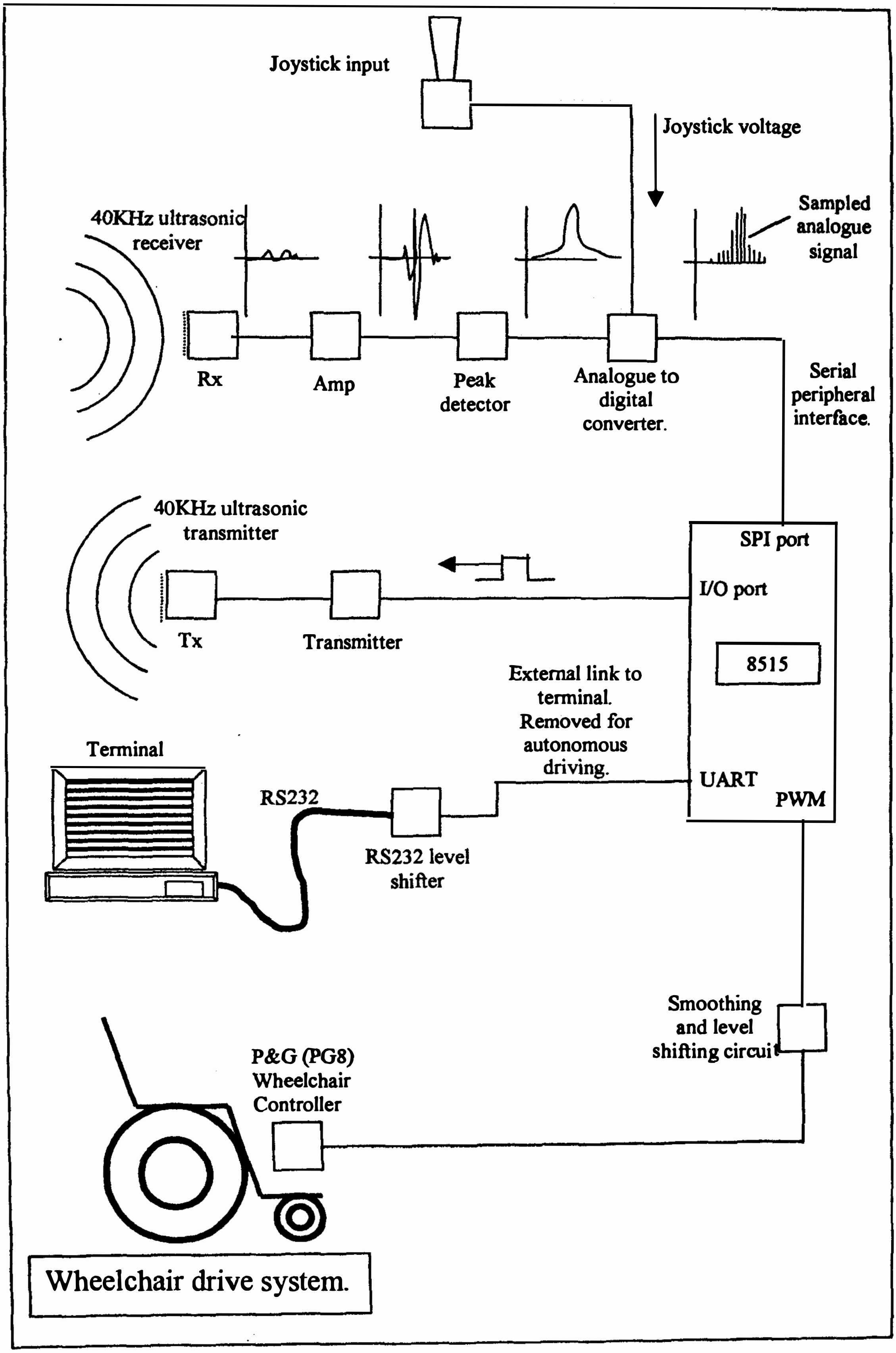


Figure 4.2. The final prototype hardware system.

of the research work. Previous work [Stott (1997)] had involved modifications to the wheelchair base and simulation of the input device, in this case a joystick. As the controller was sensitive to the input and reference voltages from the joystick a part of the previous work involved learning how to create a simulation of the joystick.

Autonomous operation of the wheelchair was important during the new work, so all the processing of data and calculation of new trajectories was performed by an onboard micro-controller. The new intelligent wheelchair had advantages over the assistive wheelchair prototype created for the previous work described in Stott (1997). These included:

- Fully autonomous operation, the previous assistive wheelchair prototype was tied to a desktop computer by an umbilical cable.
- Faster response as the data rate between the sensors and the micro-controller was higher.
- Greater flexibility as new circuit boards could be added as required without reducing performance.
- An integrated development environment was available for the ATMEL micro-controllers, which allowed rapid interfacing between high and low level languages during programming and rapid debugging during testing.

The addition of a sensor system was required in order to assist the wheelchair user in navigational tasks. A human operator has a highly sophisticated and capable navigational capability even when illness or disability has impaired some of those attributes. Although the intelligent wheelchair described in this Dissertation was aimed at severely impaired users of wheelchairs, the intention was not to replace the user but to assist them. There are social, developmental and physiological reasons for this approach, which were discussed in Chapter 2 and Chapter 3. The expert system and controllers needed to work in real time and to be robust. The intelligent wheelchair had to be affordable to be acceptable. The sensors needed to be low cost and robust whilst providing object detection and acceptable ranging accuracy. Vision and laser ranging systems, infra red detectors and ultrasonic sensors were considered. Ultrasonic sensors were selected that operated on the principle of echolocation. Similar sensors to those used in Stott (1997) were constructed and fitted to the wheelchair base. Innovative software was written so that the micro-controller could control the wheelchair and read data from the sensors and joystick.

A simplified representation of the final prototype hardware used to test the novel algorithms is shown in figure 4.3. The completed intelligent wheelchair prototype is shown in figure 4.4.

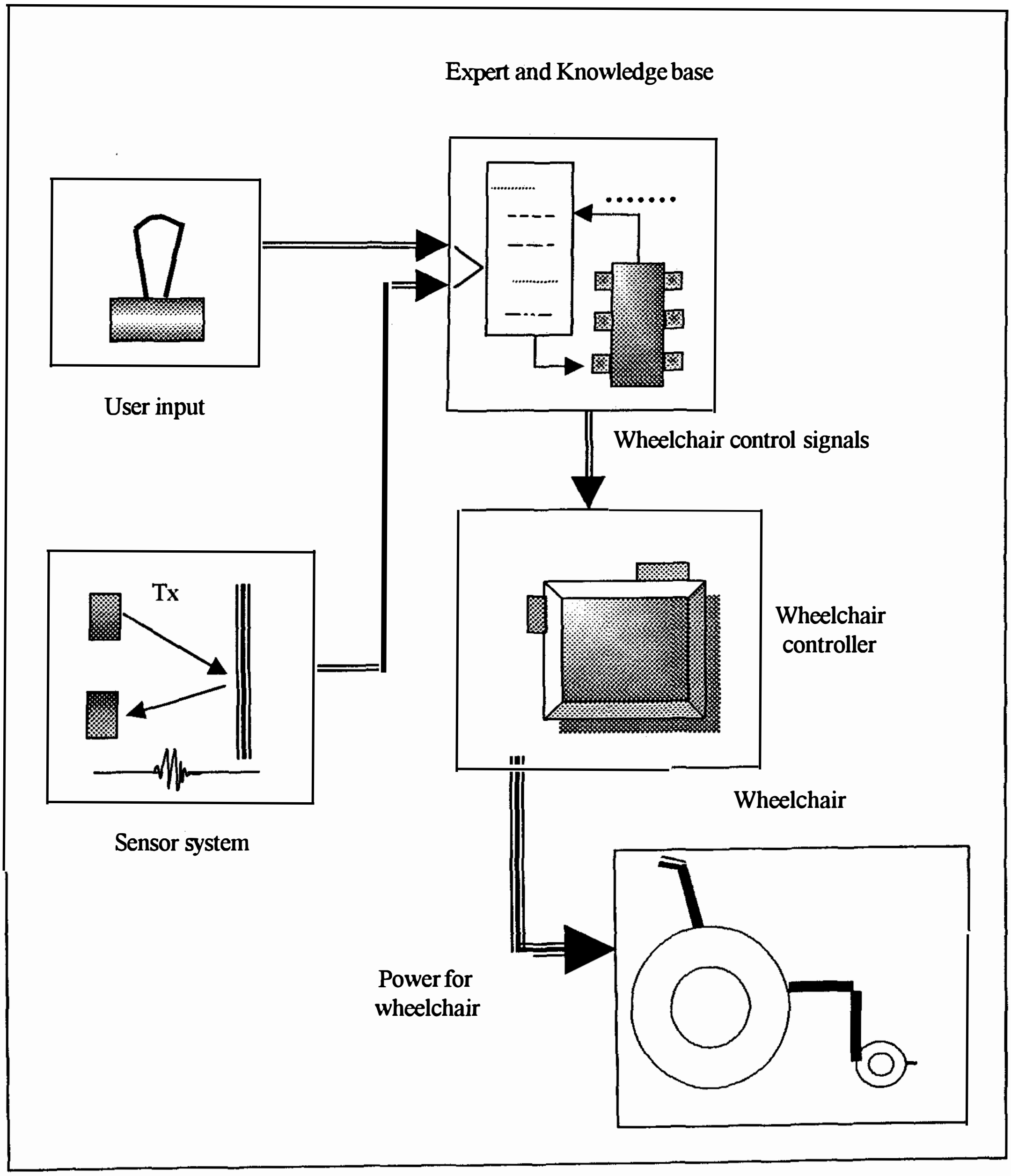


Figure 4.3. A simplified block diagram of the new system.

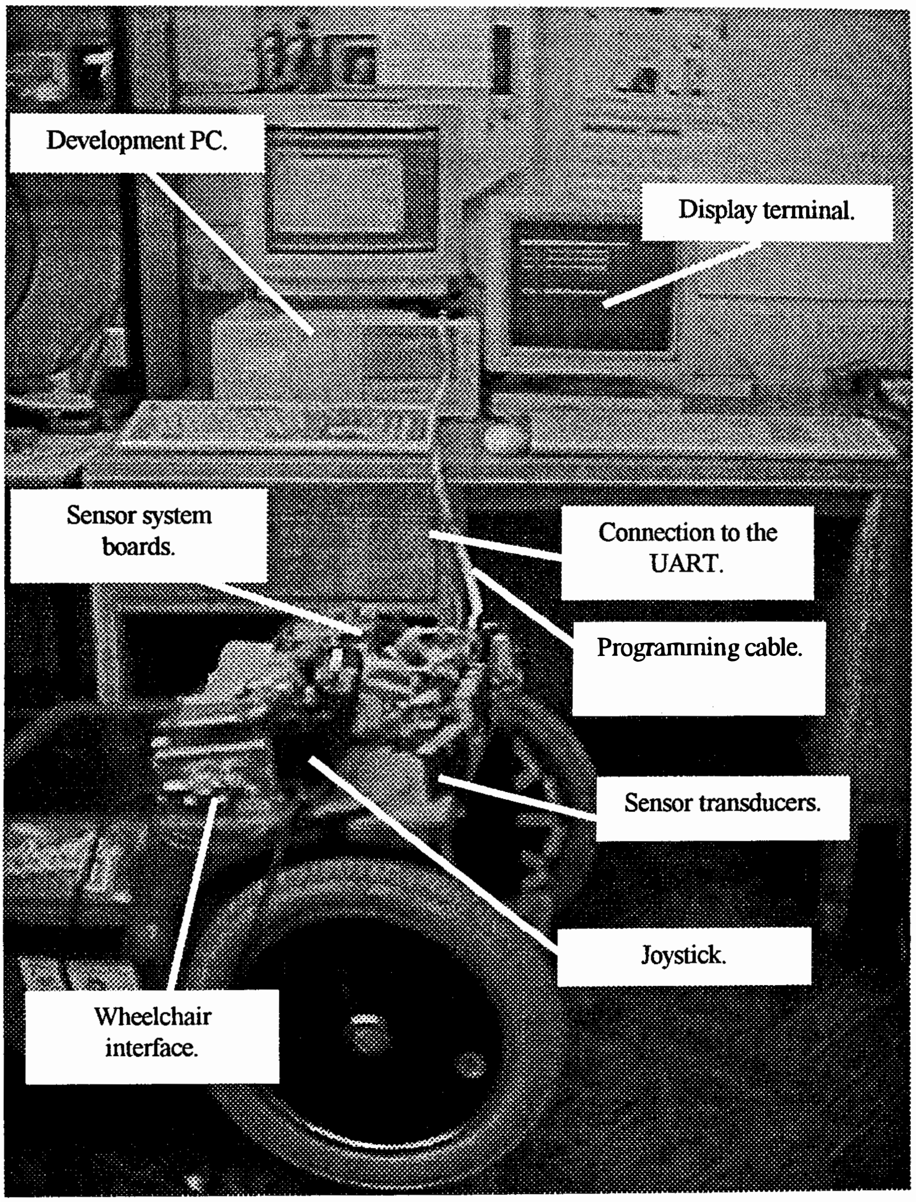


Figure 4.4: Photograph of the completed system and explanatory key.

4.1 The prototype intelligent wheelchair.

The standard powered wheelchair is shown as a block diagram in figure 4.5. The user had control over the wheelchair by the use of an input device, in this case a joystick. The controller interpreted the control signals from the joystick and provided the power for the wheelchair motors. The wheelchair controller was required to cope with different types of user with different capabilities and skill levels. Also the way the joystick was used by one user may be different from another.

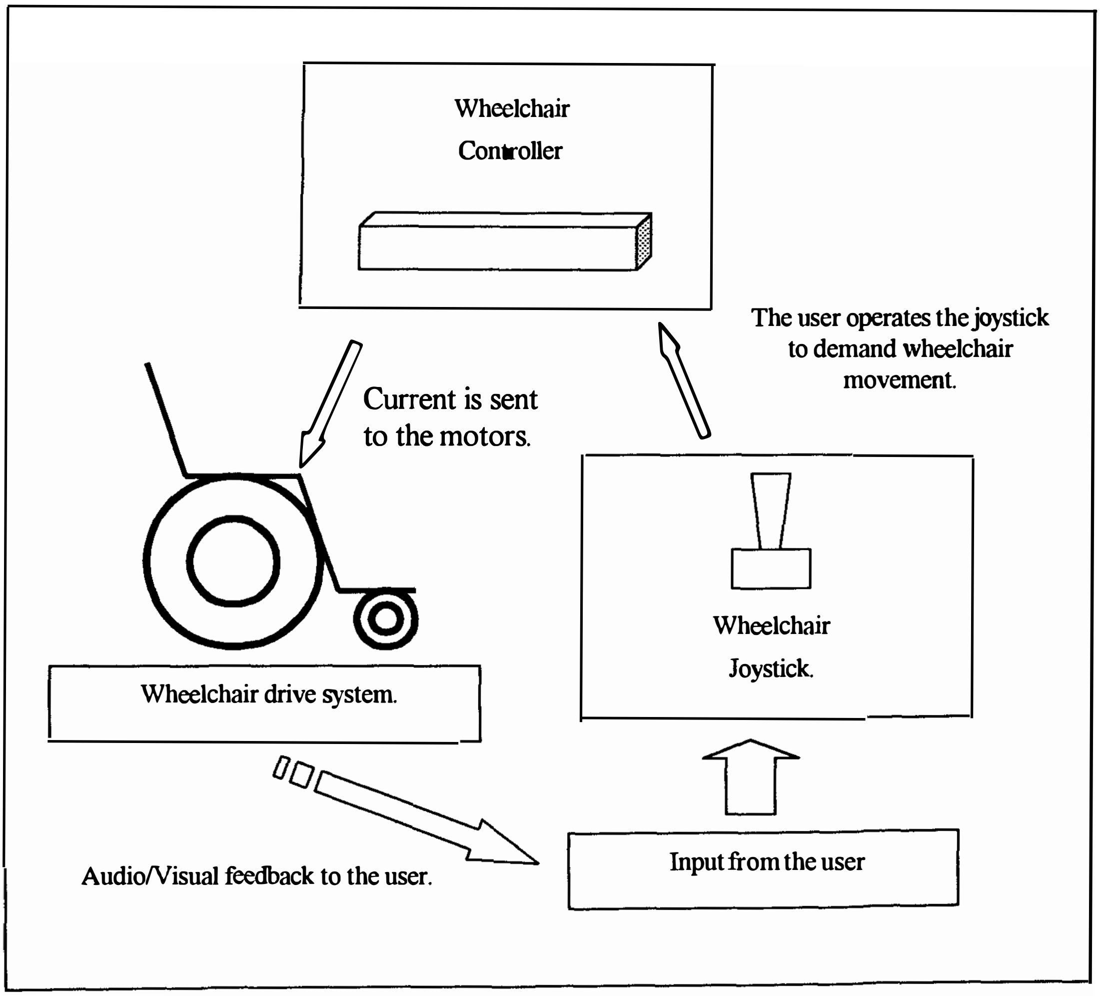


Figure 4.5. The unmodified wheelchair system.

Wheelchair control was open loop if only the hardware was considered. A feedback route for the wheelchair controller did exist through the user of the wheelchair. If the user was incapable or unwilling to provide accurate control inputs to the wheelchair controller then the wheelchair could crash and cause damage to itself or the surroundings, or cause injury. The new intelligent wheelchair controller described in this Dissertation was created to provide another source of feedback to augment a user who did appear to be operating the powered wheelchair safely.

4.2 The Wheelchair Base.

The wheelchair base used for the research work was a Bobcat II base which was originally manufactured by QED Ltd. It is now manufactured by Smile. The bobcat II wheelchair was a highly manoeuvrable, electrically powered paediatric wheelchair. The unit consisted of a front wheel drive chassis and fibreglass body. The base consisted of a heavy steel plate chassis to provide stability, rigidity and a framework for the components of the wheelchair. For this work the base was not fitted with a seat or elevator during development.

The wheelchair was a four-wheeled model with two driven wheels and two trailing casters. The driven wheels were situated at the front of the base and the seat when fitted was situated between these two wheels. The trailing casters were fitted to the rear to support the rear of the base.

The driving wheels were powered by two 12V DC parvalux motors through a worm drive right angle reduction gearbox. Altering the differential of rotational speed of the driving wheels effected steering. If the wheels were set to rotate in opposite directions then the wheelchair base would spin. The motors could draw up to a peak current of 40 Amps each for a short period. The wheelchair was fitted with two 35 Amp-hour sealed and leak proof (Gel) lead-acid batteries. There was space within the battery compartment to fit additional two batteries, effectively doubling the range of the wheelchair.

A ceramic parking brake was automatically applied to the motor output shafts when the wheelchair base had been stationary for a short while. The brakes would be automatically released when the joystick was next operated.

4.3 The Wheelchair Controller.

The powered wheelchair consisted of a power source, drive motors, an input device and a controller. The controller could be connected to any input device with a suitable interface. The controller microprocessor could be programmed via a socket on the controller housing. The dynamic response to command inputs could be programmed by using programmer DP1b available from Penny and Giles Drives Technology Ltd. The function of the programmer was to assign the controller parameters that set the normal drive characteristics of the wheelchair. The parameters included acceleration and deceleration. In normal operation the parameters could be set up by carers or therapists to suit the requirements of the user.

The wheelchair was fitted with a Penny and Giles Drives Technology "PG8 Remote Interface" controller. Technical information was available in PG8 Series Digital Powerchair Controller, [Crane (1995)]. The controller performed a number of safety checks when the wheelchair was switched on in order to ensure that the controller was functioning within tolerances set at the factory. These also included checking that the input device was connected properly, was not damaged and was in the centre or "STOP" position. The controller also calibrated and checked the joystick by comparing reference voltage levels. The controller continually monitored the joystick and wiring during the operation of the wheelchair to ensure the safe operation of the wheelchair. If a fault condition was detected then the wheelchair was automatically stopped and the fault was rectified before the wheelchair could be used again. It was essential that the wheelchair control system was failsafe in order to avoid injury to the user or damage to property.

It was necessary to have a suitable wheelchair controller fitted [Crane (1995), Taylor (1995)]. The first powered wheelchairs to be produced were simple and required skill to drive them smoothly and accurately. More modern wheelchairs had a controller fitted. The controller allowed the high current motors to be controlled using a low current input device. The power was fed to the motors in a controlled and "soft" manner. The potentiometric controls could be replaced by other circuitry, for example the output of a digital to analogue converter. The substitute-input device would need to be so constructed to present similar voltage and impedance levels to the controller.

The current drawn by each of the motors could be as much as 40 amps for short periods. This current was applied by slowly building up to the full value over a period of time. This allowed the motor torque to be gradually built up onto the wheels of the wheelchair, giving a smooth and comfortable ride to the user and reducing the stress on the wheelchair components. The controller could also limit the turning speed of the wheelchair. Should the wheelchair be travelling at top speed and the user command the wheelchair to turn sharply, the controller would only allow the wheelchair to turn at a safe rate. These safety features reduced the potential for the wheelchair to wheel spin, skid or turn over.

There were disadvantages to the application of control strategies by the controller. The main disadvantage was the way the controller appeared to ignore the instantaneous input from the user. There was sometimes a noticeable delay in the response of the wheelchair to the commands of the user. Users quickly learnt how the wheelchair responded and compensated by applying control signals earlier. They also quickly learnt to estimate the stopping distance of the wheelchair. A user for whom the wheelchair controller was designed would learn to use the wheelchair with practice. The new controllers fitted into the existing wheelchair during this research also required the ability to adapt and anticipate the action of the controller

4.4 The ultrasonic sensors.

To assist the wheelchair user to drive the wheelchair safely, ultrasonic sensors were mounted onto the wheelchair to detect the environment. Ultrasonic sensors can be capable of accurate operation and could be implemented in a number of ways. A fast response and high resolution was achieved by Teshigwara *et al.* (1989) who used a Frequency Modulated Continuous Wave (FM-CW) to achieve a resolution of 0.2mm at an acquisition time of 2ms. The centre acoustic frequency was 850KHz and was continuously swept until a point of maximum constructive interference was achieved. The distance to the target could then be accurately determined. Although accurate and fast, the range was limited to 30-70mm, suitable for product detection and counting type tasks.

Kay used a novel sensor array and constant transmission frequency modulation (CTFM) sweeps to assist blind people in detecting their environment [Kay, (1999b]. A CTFM system examines the response of the environment to a frequency swept sine wave. The shape of the envelope returned to the receiver is determined by the properties of the environment. It was possible to recognise objects by the frequency spectrum of the received signal. McKerrow & Harper, (1999) successfully differentiated between types of leafy plants with their system.

Single frequency pulse-echo ultrasonic sensor systems can encode information on the pulses in order to reduce interference between different sets of sensors. Kleeman (1999) used a known separation between two pulses to identify different arrays. The receiver signal was rapidly sampled and the separation read as each pulse train was received. Pulses from other sensors were rejected. Accurate angular resolution was also possible by using two receivers mounted close to the transmitter and measuring slight time delays between pulses.

Advances in ultrasonics have improved reliability and accuracy. The new work described in this Dissertation aimed to use a cheap range finder that could be afforded by disabled users and to allow for the shortcomings of ultrasonic sensors with in-built intelligence. Although many techniques existed, the pulse-echo or time of flight technique was the simplest method and was used in this work.

Single frequency echolocation range finders were a popular sensing system for mobile vehicles. A drawback with a single frequency was the possibility that multiple return paths to the receiver existed. It was possible that the return paths would cause destructive interference at the receiver resulting in deterioration of the signal. Lindstedt

(1996) used frequency sweeps during detection operations to reduce the possibility of some objects becoming invisible to the sensors. Another advantage of sweeping the frequency was that some reflecting surfaces were more easily detected at one frequency than another.

If the sensors moved within an environment such as if they were attached to a moving vehicle, some of the interference problems could be avoided.

Several configurations of sensors were tested. Stott (1997) employed two transmitter and receiver pairs mounted at the front of the wheelchair in positions which sensed the environment forward of the wheelchair. An ultrasonic image was converted to a simple representation of the environment. The image identified the presence of objects in the path of the wheelchair which were within the range of the sensors. This method was limited as it only had two sensed areas that did not overlap. Triangulation was not possible as the hardware did not support more complex operations and was slow. However, the sensors for wheelchair driving gathered useful information.

In the new work described into this Dissertation, configurations consisting of three and four sensor transducer pairs were used and sensing strategies including time of flight (TOF) ranging and a combination of TOF and triangulation were tested.

4.4.1 Time of flight range finders.

The time of flight for the purpose of calculating a range to an object was the time taken for a pulse of ultrasonic sound to travel from a transmitter, be reflected off an object, and to then return to a receiver figure 4.6. The recorded time may be used to calculate

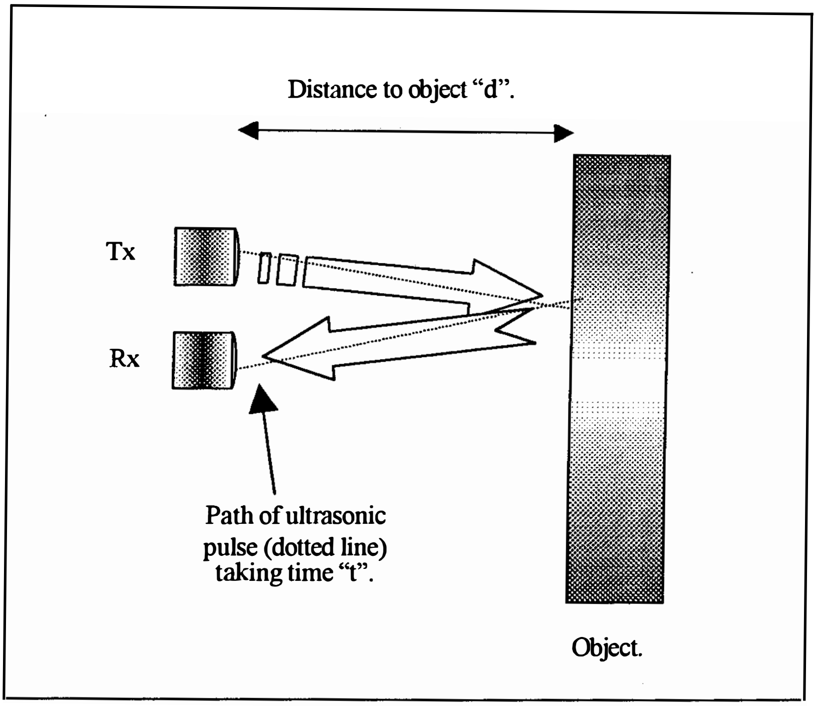


Figure 4.6. The time of flight principal.

the distance to the object from the sensors using equation 4.1.

Equation 4.1.

$$d = \frac{ct}{2}$$

Where: d = Distance to object,

c = The speed of sound in air (~330m/s),

t = The time taken for the pulse to return to the receiver.

TOF sensors were generally good at detecting the presence of an object. Due to the divergence of the transmitted beam of ultrasonic energy, the sensors tended to have limited directional capabilities [Stott (1997)]. The beam width can be limited in sensor systems that emit a pulse of ultrasonic energy rather than a continuous beam. In Kleeman (1999), the beam width was narrowed due to the large number of harmonic frequencies in the transmitted beam. The harmonics were generated during the fast switch-on and switch-off phases of the pulse. The harmonics tended to cause destructive interference patterns around the edges of the beam, which had the effect of reducing beam width. Results of tests on the ultrasonic range finders created for the work described in this Dissertation are presented in Chapter 6. The location of an object may be estimated to be laying on a locus described by an arc with radius equal to the range and falling within the beam width as shown in figure 4.7.

4.4.2 Object location by triangulation.

Triangulation techniques are similar to simple TOF ranging methods as the distance to an object can be calculated from the time taken for a pulse of energy to travel from a transmitter to an object and return to the receiver. However to employ triangulation, a known distance must separate the transmitter and receiver. The distance from the transmitter to the receiver forms one side of a triangle and the calculated distance from the transmitter to the object and then to the receiver forms the other two sides of the triangle. With information from a single triangulation calculation, the position of an object will still not be known. This is because the estimated position of the object describes a locus somewhere in front of the sensors as shown in figure 4.7. In order for the actual position of the object to be known a second set of sensor data is required to

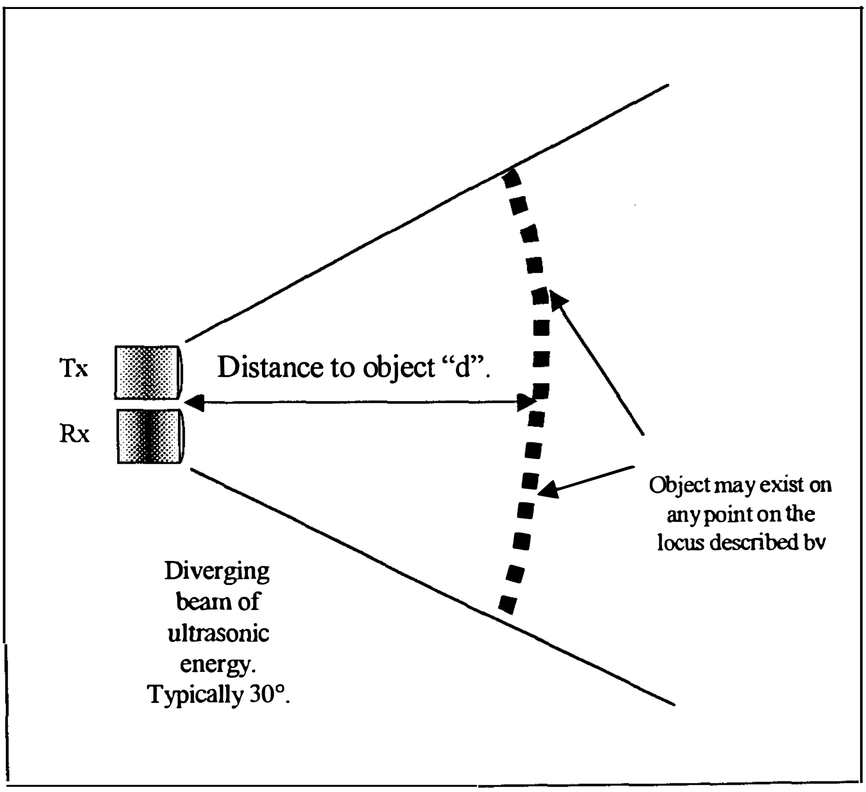


Figure 4.7. The locus of the object's position.

"fix" the position of the object. The second data set may be derived by triangulation from a separate sensor pair or from a simple TOF range acquired by an additional receiver positioned near to the transmitter as in figure 4.8. Equation 4.2 could be used to calculate the two-dimensional position of the object.

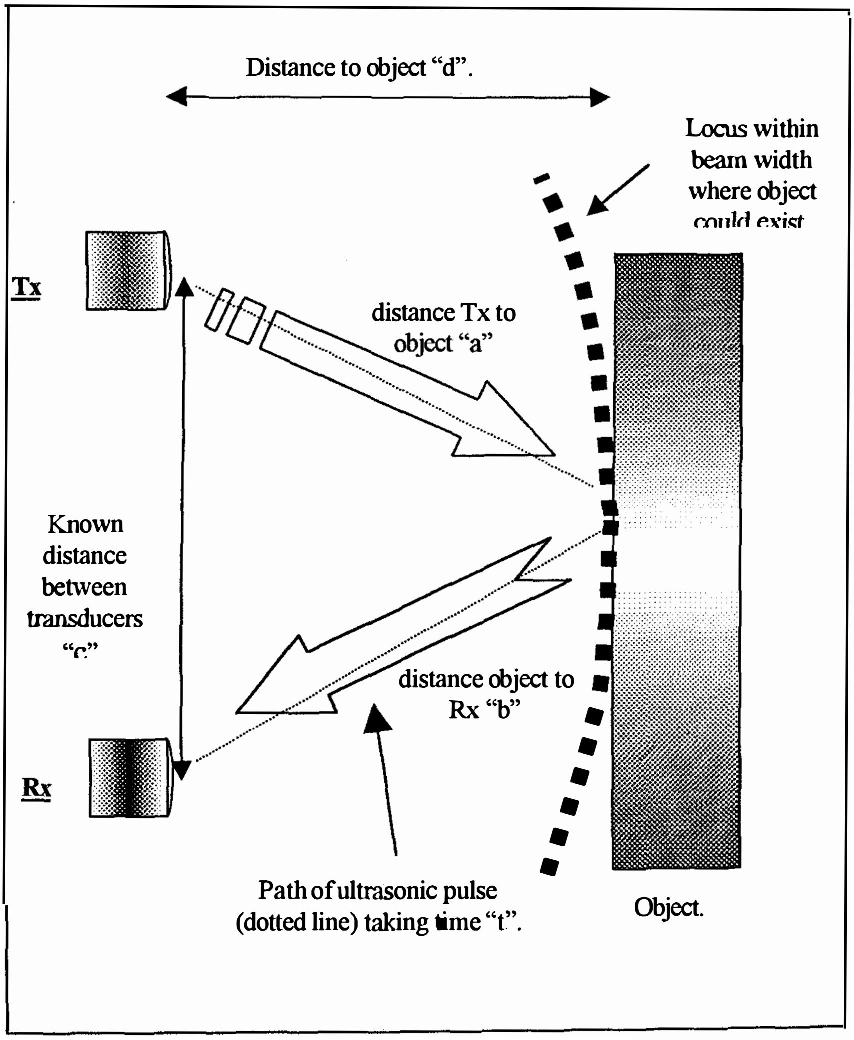


Figure 4.8. Object location using a combination of TOF range and triangulation.

Equation 4.2: Equations for calculating the position of an object using triangulation.

$$d = \sqrt{a^2 - (c - e)^2)}$$

$$e = \sqrt{b^2 - d^2}$$

Where : a = distance Rx to object,
b = distance Tx to object,
c = separation of Tx to Rx,
d = distance from sensor line to object,
e = offset from Rx to object.

An alternative method for estimating the position of an object was to relate the measurements "a" and "b" to a lookup table to define the location of the object. A lookup table could be an efficient method for a micro-controller-based sensor system.

Time of flight sensors can generally detect an object but the sensors tend to have limited directional capabilities due to divergence of the transmitted Beam [Sanders *et al.* (1998)]. The location of the object may be estimated to be lying on a locus described by an arc with radius equal to range, the sides of the arc defined by beam width. Accurate beam widths for transmitters used in this work were obtained experimentally inside an anechoic chamber and an example is shown in figure 4.9. Symmetry is assumed about the transmitter axis. The data was gathered by measuring an isometric power line around an operating transmitter. Figure 4.10 shows a test underway in the anechoic chamber.

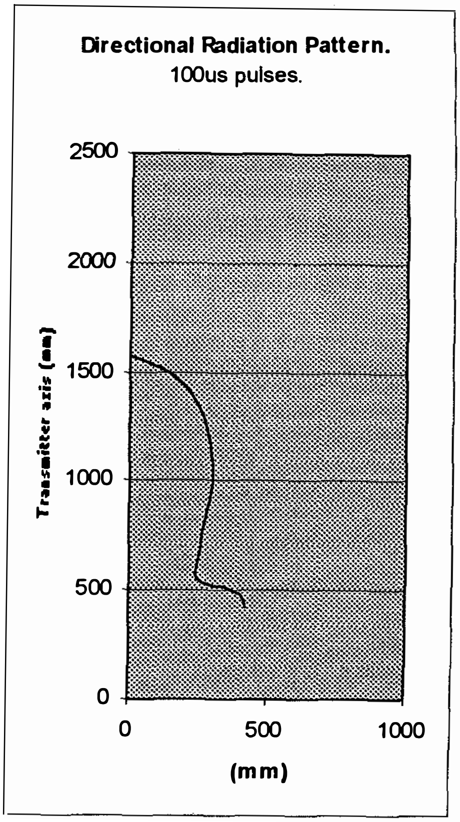


Figure 4.9. An experimentally derived beam pattern.

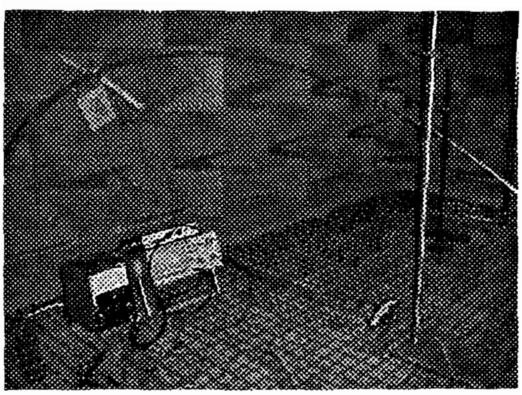


Figure 4.10. A test inside the anechoic chamber.

Figure 4.9 shows a directional radiation pattern with the transmitter pointing along the y-axis. Half of the beam pattern is shown. Figure 4.9 shows the beam pattern obtained when the transmitter was mounted in a hard mounting. The transmitter was also tested under the same conditions but this time the mounting was faced with 5mm of polyurethane foam as shown in figure 4.11. The pattern on figure 4.9 shows a side lobe starting at 500 mm from the sensor face. The foam facing on the mounting reduced the

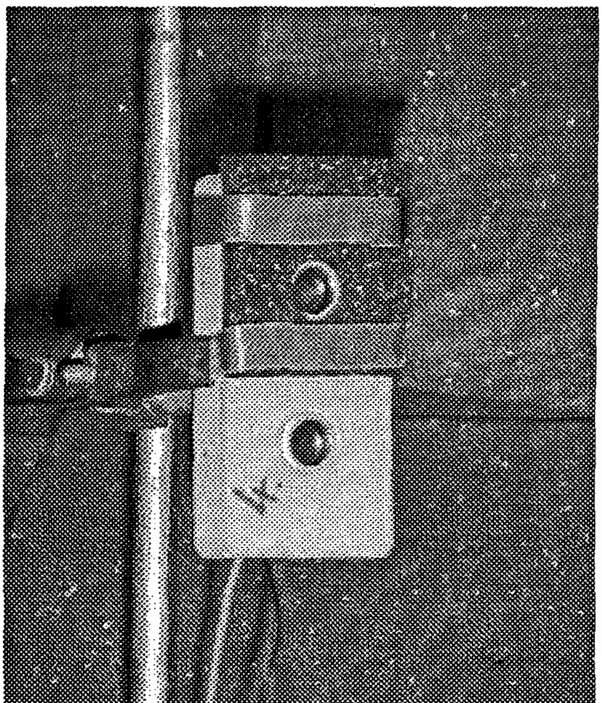


Figure 4.11. A sensor mounting with a 5mm thick polyurethane foam facing.

lobe but made the beam wider. The beam plot of the foam-mounting test is shown in figure 4.12. The wider beam is consistent with predictions using Huygens principle where each point along an advancing wave front may be considered as a new source of sound [Frederick, (1965)]. As the wave front exits through the foam layer then the wave fronts from these sources combine and form a new wave front which continues in the same direction as the original, except at the edges. Here the sound forms a wave front which originated from the corner and this wave front radiates in all directions. This explains why sound is diffracted around solid edges and narrow beams are difficult to create. Figure 4.13 shows the effect of Huygen's principal.

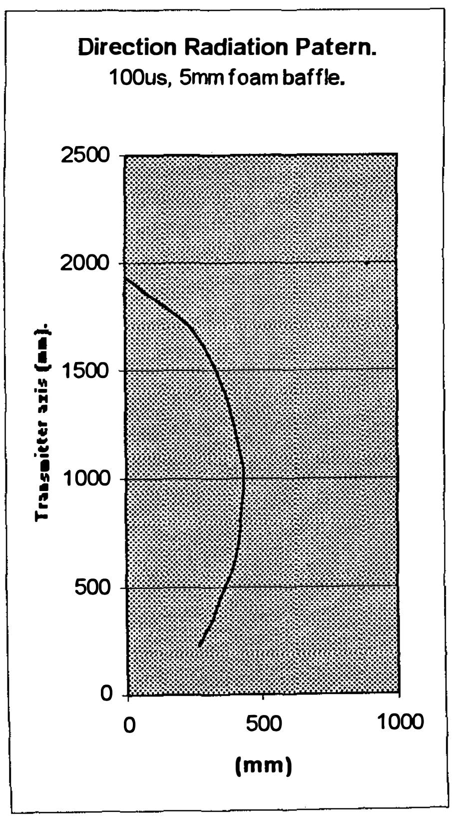


Figure 4.12. An experimentally derived beam pattern with a foam facing fitted to the front of the array.

Side lobes could affect performance. Side lobes are the effect produced by constructive interference patterns and appeared as fingers of higher intensity sound radiating from the transmitter. Side lobes caused objects to be detected to the side of a sensor. This caused the sensors to suggest a detection in front of the sensor when the object was to one side. The addition of a foam facing to the sensor mounting eliminated the sharp edges that promote the formation of lobes by disrupting the interference patterns. Although the beam pattern was wider in the case of figure 4.12, the lobes were no longer prominent

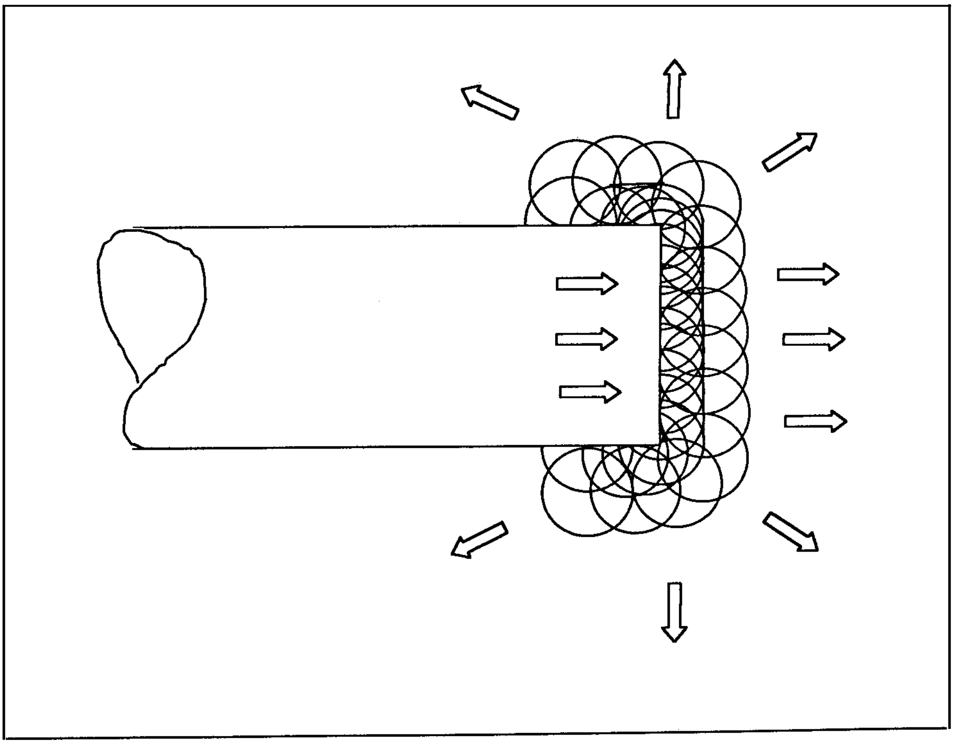


Figure 4.13. The effect of the Huygen's pipe principal

which made the beam more consistent and easier to model than the pattern seen in figure 4.9.

4.5 Detecting an object.

A prerequisite to using the new ultrasonic range finders successfully was to detect the presence of an object. Once the presence of an object had been established, the distance to it could be calculated. It could be difficult to detect an object as ultrasonic data was noisy and prone to random spikes picked up from the environment. To investigate ultrasonic TOF, an experimental test rig was created. The test rig is shown in figure 4.14. It consisted of a "PCL-815 Lab-card" input/output board connected to an 80486 base personal computer. Ultrasonic transmitter and receiver boards were constructed

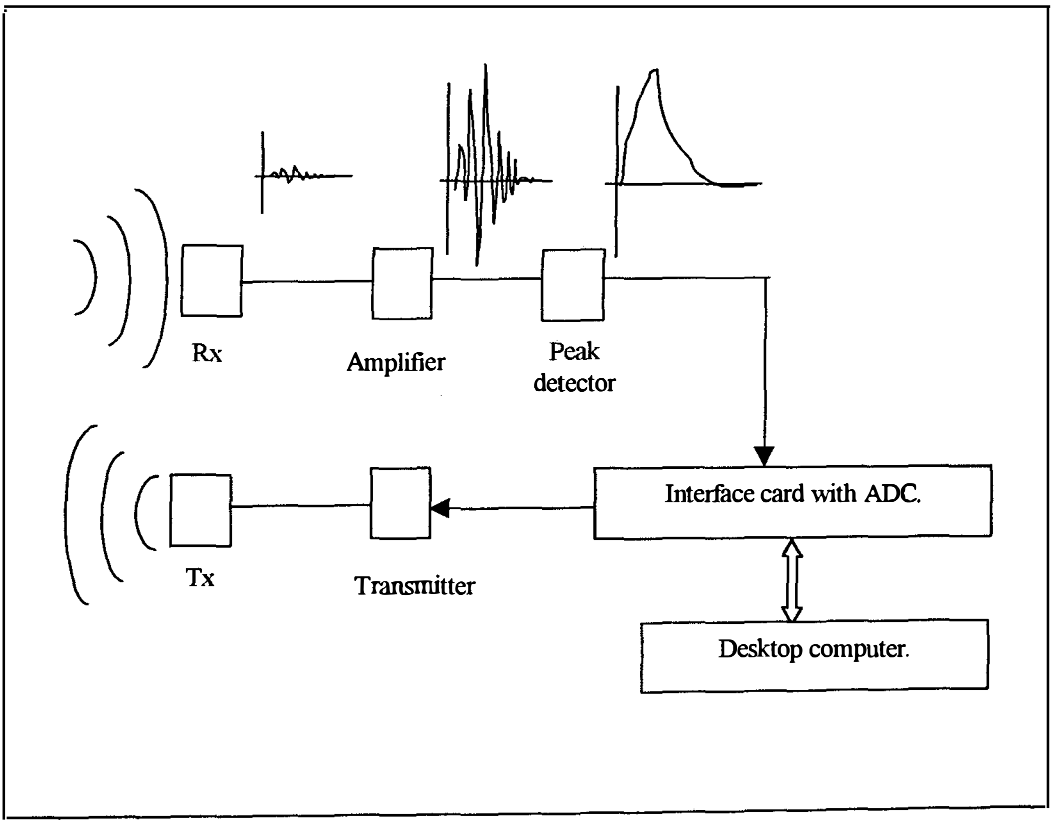


Figure 4.14. Block diagram of the test rig.

and connected to the Labcard.

A detection was usually associated with a sudden rise in the voltage measured at the receiver. The receiver voltage returned to a reference level after the pulse reflected from the object had passed. Several characteristics of the reflected pulse were considered when selecting a detection algorithm. These included:

(a). The amplitude of the receiver signal exceeding a pre-set value.

(b). A recognition of the shape of the pulse.

(c). The gradient of the leading edge of the pulse exceeding a pre-set value.

A typical receiver voltage signal is shown in figure 4.15. A 100-nanosecond pulse of ultrasonic energy was fired at a 110mm diameter plastic tube at a range of 2.2m. The plastic tube was 1m long and standing vertically on the floor. The sensor transducers were mounted 500mm from the floor. The receiver voltage was smoothed (using a low

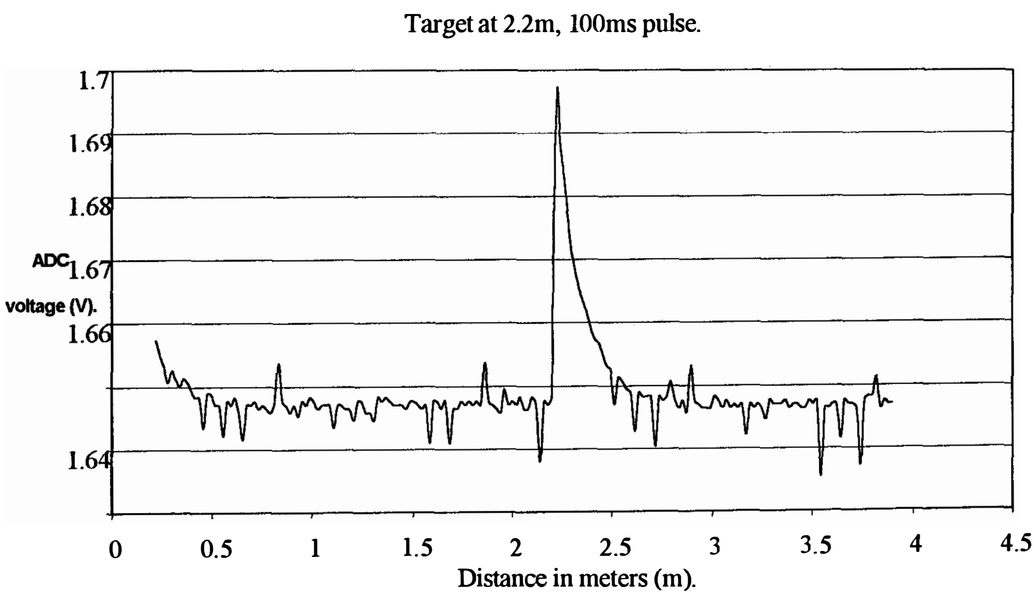


Figure 4.15. A receiver voltage plot.

pass filter) and sampled at 8.333KHz by an analogue to digital converter (ADC) on the Labcard. The sampled signal was stored to a file for analysis later.

The pulse of returned energy is prominent on the data trace. Noise can be seen along the reference level (set at 1.65V) before and after the pulse was received. The trace can be seen to fall from 1.66V at 300mm to 1.65V (reference) at 700mm. This was due to the receiver circuit recovering from cross talk that occurred as the pulse was fired.

A detection method that used a threshold voltage would be suitable for this data set. The detection threshold could for example be set at 1.67V for an error free detection of an object at 2.2m. A disadvantage of this method was that the amplitude of the pulse tended to reduce as the distance increased or as the object became smaller. In these situations the received pulse could become lost in the noise around the reference level, or if the threshold level was set too low, noise could appear as a detection.

A method for detecting the shape of the pulse (method (b)) could be an accurate way to sort the pulse from the noise. By inspection of the shape of the pulse it can be seen that the pulse had a steep rise (attack) and a gentle decay. The noise however tended to have a more gentle attack and an equal decay gradient. The most obvious attribute of the pulse was the attack gradient and method (c) suggests the use of the gradient to detect an object. Figure 4.16 shows a plot of the gradient of the trace shown in figure 4.15. The gradient of the rising face of the pulse can be seen and it reaches a peak of 400 volts/second. This can be compared to the highest gradient attributed to noise at 100 volts/second. An added advantage of the gradient method was that the reference voltage of the detector was not critical. Using this method, detector boards could be exchanged without having to recalibrate the reference voltage.

Method (c) was reliable and convenient and was selected as the detection strategy for this work.

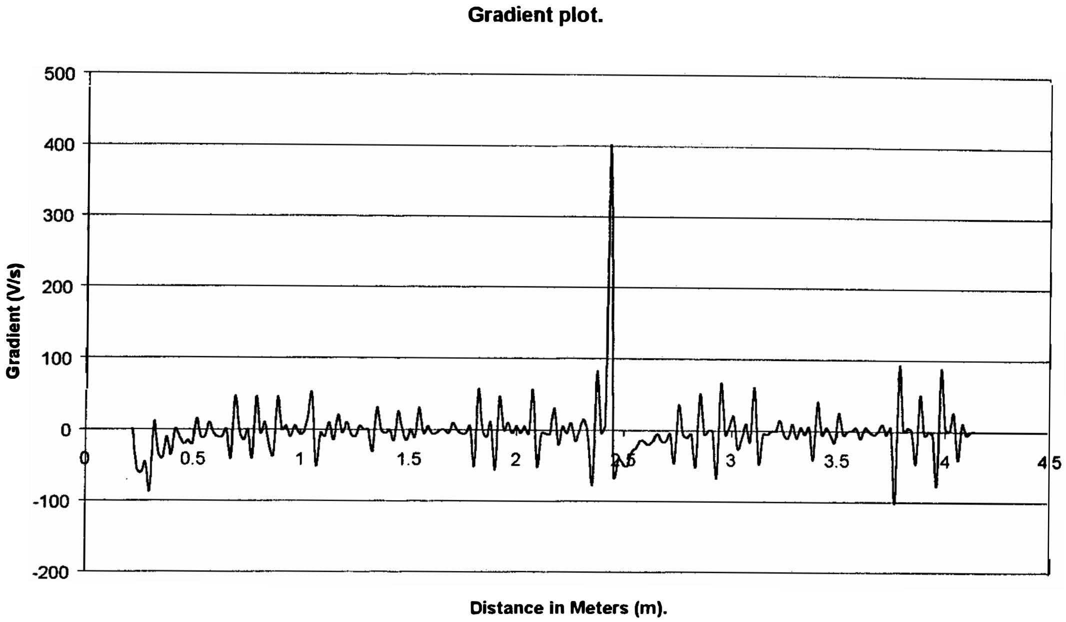


Figure 4.16. A gradient plot of a receiver voltage.

4.6 Sensor configurations used in this work.

The range finders employed 40KHz ultrasonic transmitters and receivers mounted in pairs. A pulse of ultrasonic energy was transmitted. The pulse of energy was reflected off objects in its path. Some of the reflected energy was returned to the receiver. A sensor transducer pair working in isolation was of limited effectiveness and had a limited coverage. A single sensor pair did not have a beam width wide enough or the resolution required to cover the frontage of the wheelchair. For safe navigation, the area immediately in front of the wheelchair and the areas to the front left and right were important. Either the sensor pair needed to move [Langner (2000)] or several sensor pairs were required to sense these areas and for enough data to be gathered for safe navigation. Tests were conducted with arrays of one and more sensor pairs. These tests are described in this Chapter and results of the tests are presented in Chapter 6.

4.6.1 A single sensor array.

Reliable measurement of range was possible from a minimum of 75 mm from the front of the transducers to a maximum of 2m. Longer ranges were sometimes possible with a large target with a good reflecting surface.

Figure 4.17 shows the simple single sensor array that was used as an example and shows a typical beam pattern for a single pair of transducers. A single-sensor grid was

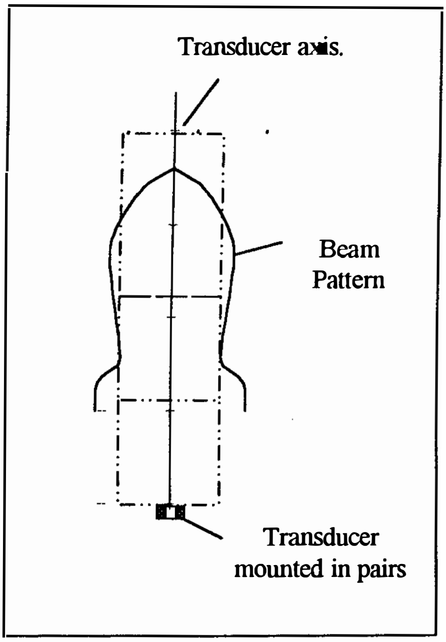


Figure 4.17. A single sensor array beam pattern.

sufficient for demonstrating a histogramic method for improving sensor reliability. The use of histogrammic mapping is described in section 5.4. The number of cells selected can be dependent on the application. It was simple to modify the software to vary grid parameters and size.

A characteristic of ultrasonic transmitters is the divergence of the beam and their tendency to exhibit side lobes. It was impossible for a single sensor range finder to provide angular information and the location of an object was often uncertain. Figure 4.18 shows a good reflector located within a powerful side lobe of the transmitted beam. The dashed line shows an arc of the possible location of the object. An object with poor reflection characteristics would not necessarily be detected if it were in a similar location to the good reflector.

For this work two configurations were tested; (a) three sensors in line abreast across the front of the wheelchair, and (b) four sensors mounted on the front corners of the wheelchair. The configurations are shown in figures 4.19a and 4.19b. The results are

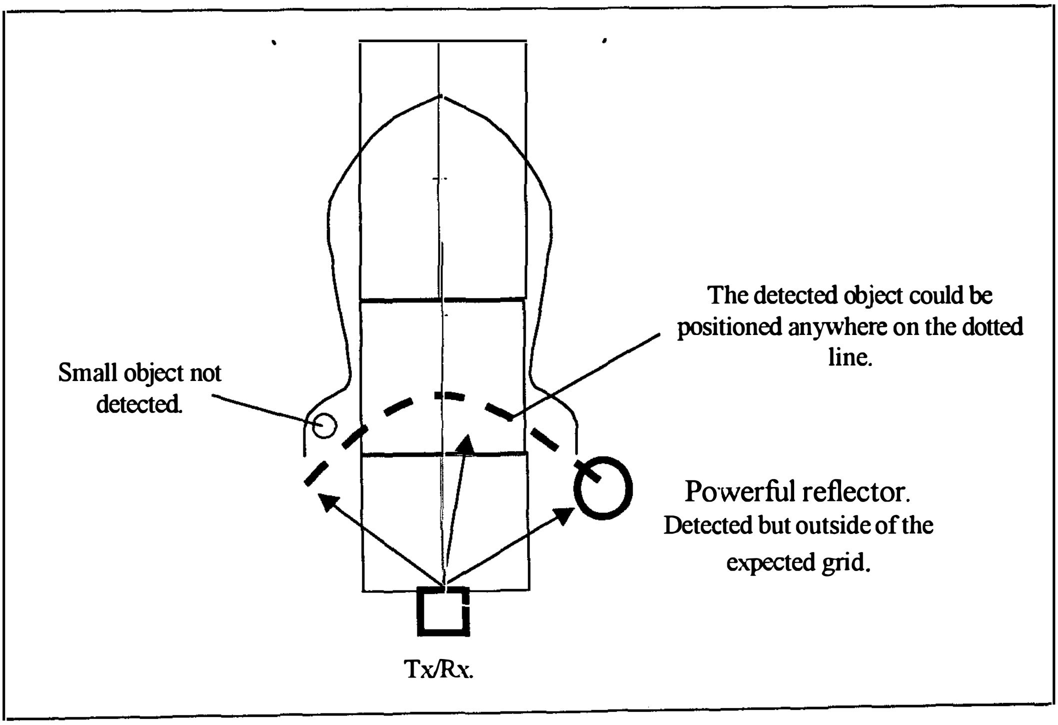


Figure 4.18. An example of unreliable detection due to beam pattern side lobes.

described in Chapter 6.

4.6.2 Multi-sensor arrays.

A sensor range finder array acting in isolation had limited usefulness. More information could be gathered about the environment if more than one range finder was used. Three sensor pairs were mounted as part of an array which created a nine-element grid in front of the sensors. Figure 4.19a shows the three-sensor array. It can be seen that grids and the beam pattern overlap. However it could not be assumed that an object detected in one grid square would be detected by the corresponding transducer in an overlapping grid. Detections were reinforced where detection in one grid was matched by a corresponding detection in an adjacent grid. Conversely where an object appeared in one grid but not the adjacent one less confidence existed in the accuracy of the detection.

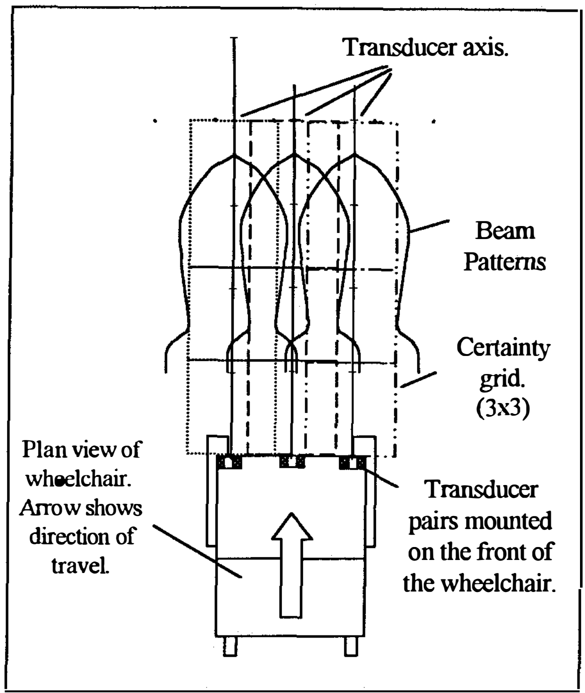


Figure 4.19a. The three sensor array beam patterns.

Figure 4.19b shows the four-sensor array that was tested. In this configuration, there were many more overlapping grid elements. This increased the possible of reinforcing or reducing the belief that a grid element was occupied or not. Disadvantages of this configuration were the extra processor time taken to acquire grid data and apply the comparing algorithms. Also as there were more sensor pairs, it took longer to activate and read all of the transducer sets. It was also noted that little more information was available from this array than the simpler three-sensor array shown in figure 4.19a. The four-sensor array was not selected for use on the prototype intelligent wheelchair.

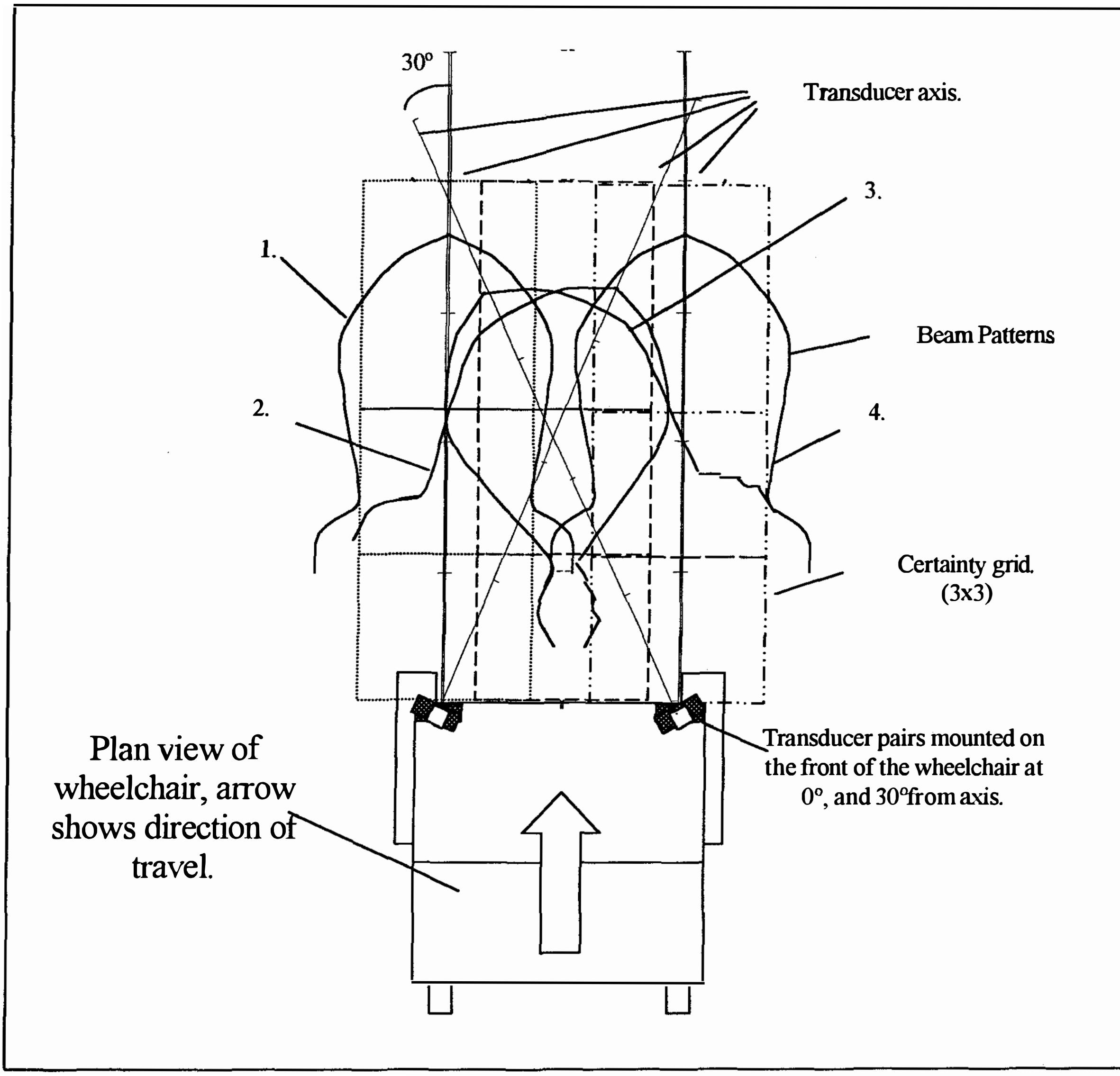


Figure 4.19b. The four-sensor array beam pattern.

A representation of the environment in front of the sensor array was then available for analysis. Confidence existed in the accuracy of the grid as the histogrammic mapping reduced the effects of noise on the array. Path planning algorithms could then be used to form a strategy and suggest safe routes [Sanders, (1993 & 1998)].

4.7 The micro-controller circuits.

Hardware was created to provide a platform for the creation and testing of novel intelligent wheelchair control algorithms. Some of the initial prototyping and experimental work was completed using a Philips micro-controller. In order to take advantage of "in-system-programming" (ISP), flash memory, greater speed and an integrated programming environment, the target processor was changed to an ATMEL AVR micro-controller. The Philips based experimental test rig was used in some of the early work to create suitable object detection algorithms for the sensors created as part of the work described in this Dissertation. The later prototype micro-controller hardware used some of the object detection algorithms created during the initial experimental work and was interfaced to the wheelchair controller. The later prototype hardware was faster, more powerful and easier to program than the initial experimental test rig.

4.7.1 The Philips based experimental test rig.

Timing and control functions were carried out by a micro-controller. Early work used a Philips 87C752 micro-controller. The 87C752 was an 80C51-based 8-bit micro-controller. It contained 2KByte EPROM, 64 byte RAM, 21 I/O lines, a 5 channel multiplexed ADC (analogue to digital converter) and an 8-bit PWM output. This made it a suitable choice for the new controller. Exposing the "quartz lid" erasing window to an ultraviolet lamp for 15 minutes could erase the EPROM. It was necessary to place the micro-controller chip into an ultraviolet erasing device each time the micro-controller code needed to be changed. This process required the removal of the micro-controller from the target circuit board for each program modification.

The 87C752 was used in an early prototype of the sensors. A test PCB was manufactured at the University that gave access to all the micro-controller pins and provided a crystal oscillator circuit for the micro-controller. The PCB is shown in figure 4.20

The multiplexed 5 channel ADC was used to sample the output of the receiver circuit. The sensors were interfaced to an 80486-based desktop computer during the development stage. The test rig is shown in operation in figure 4.21. The Philips based test rig was useful during the development of sensor algorithms.

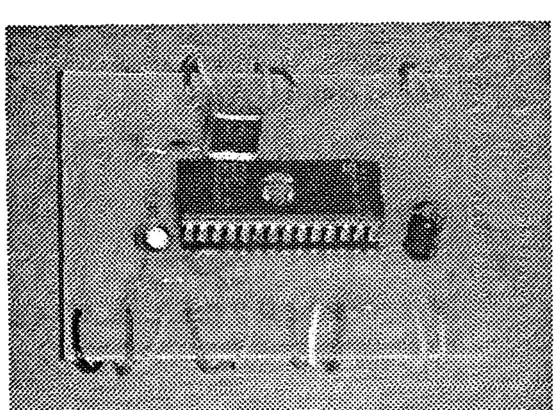


Figure 4.20. The experimental circuit board.

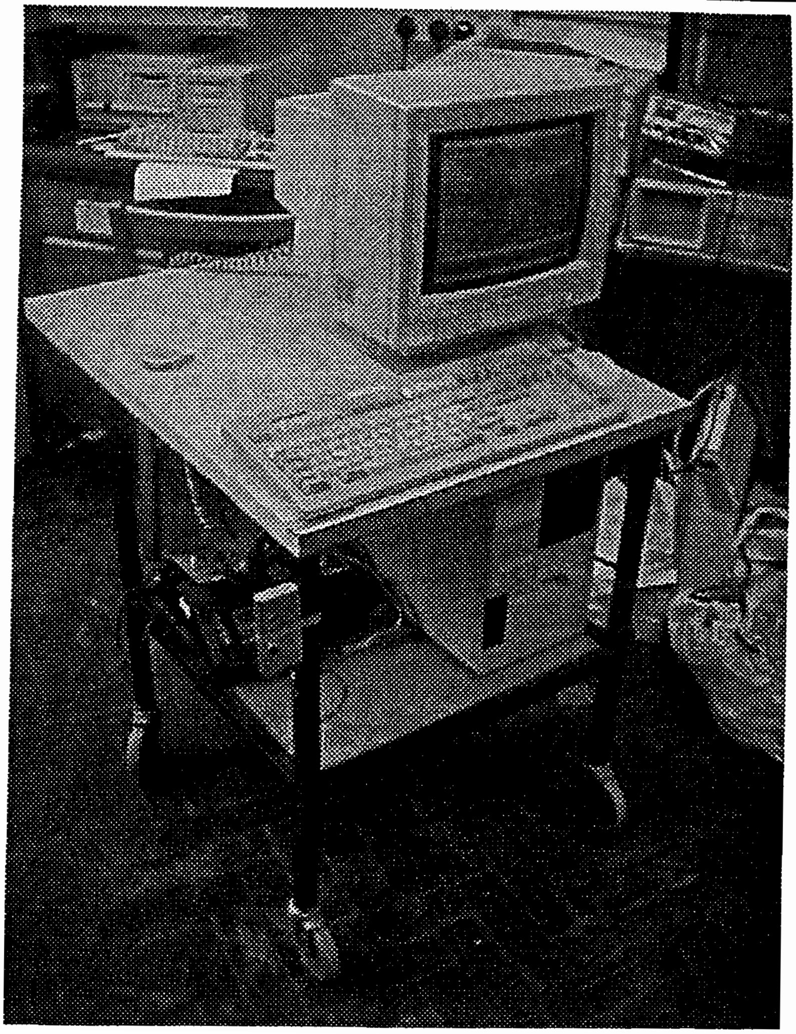


Figure 4.21. The Philips based experimental sensor system.

4.7.2 The ATMEL based prototype.

A decision was made during this work to standardise all work on AVR enhanced RISC micro-controllers in order to take advantage of high level language tools for embedded micro-controllers available at the University. The AVR enhanced RISC micro-controllers were based on a RISC architecture that has been developed to take advantage of semiconductor integration and software capabilities [Myklebust (1996)].

An ATMEL AT90S8515 was selected as it featured a serial port, 8K bytes of "in-system" re-programmable flash memory and sufficient input/output lines. A major asset of this processor was the ease of programming through the "in-system" programming (ISP) capability that allowed rapid prototyping and testing of software code. A programmable serial UART allowed convenient access to and input of data to internal registers during development and testing. A Serial Peripheral Interface (SPI) was available onboard. Peripherals attached to this serial port could be daisy chained allowing potential for expansion. Many features of the micro-controller were redundant at this stage of development, making it flexible for future applications.

The adoption of the '8515' micro-controller improved the speed of the prototype intelligent wheelchair controller in comparison to the experimental test rig. The program space (ROM) allowed up to four times the amount of code to be used. Although the time spent on converting to the ATMEL architecture slowed the rate of progress for the research, the benefits were sufficient to outweigh the disadvantages.

4.7.3 The analogue to digital converter.

A suitable analogue to digital converter (ADC) was required to sample the receiver signals. A MAX1112 was selected as it would interface easily with the micro-controller. The MAX1112 was an 8-bit, 8-channel ADC. Communication with the micro-controller was possible through the on-board SPI compatible interface. An 8-channel device was desirable as it gave the capability for 4 ultrasonic sensor channels, 2 joystick sensor channels and 2 motor-voltage monitoring channels.

4.8 The SPI interface.

The communications between the micro-controller and the ADC (sensors) were routed through a serial peripheral interface (SPI). SPI was a convenient way to interface an ADC to the micro-controller. Connections are shown in figure 4.22. The SPI required four lines to operate. MISO (master in slave out), MOSI (master out slave in), SCLK (serial clock) and CS (chip select). The SPI was designed to provide a simple serial interface. As the serial clock was operated, data was "clocked" out of the MOSI and into the data input of the peripheral. At the same time, data was "clocked" out of the

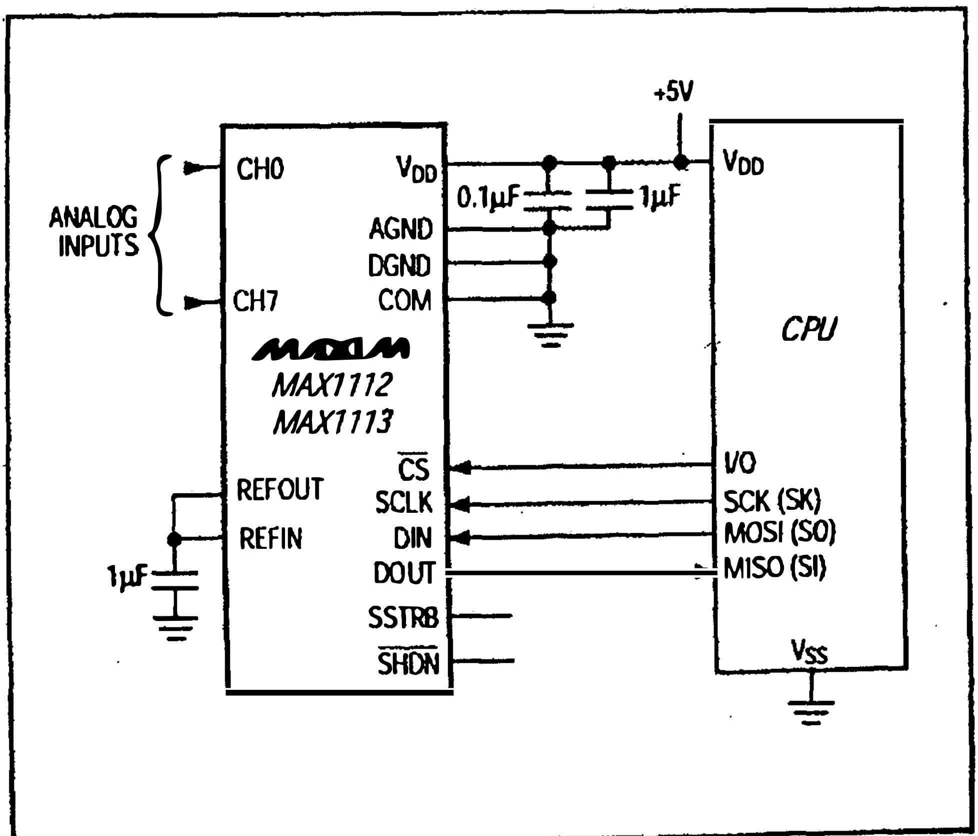


Figure 4.22. The connections from the micro-controller to the ADC.

peripheral's data output and into the MISO. The data exchange would continue until the serial clock was stopped. Chapter 6 describes the results from using the SPI and MAX ADC.

4.9 Wheelchair controller voltage modification, the PWM interface.

In order to change the trajectory of the wheelchair, it was necessary to make changes to the motor control voltages. A method was required to reduce or possibly increase the motor control voltage. If this was done, the wheelchair could be made to stop, turn, slow down or the radius of turn could be altered to make a tighter of wider turn. A number of circuits for changing the motor control voltages were considered. These included using an SPI compatible digital to analogue converter (DAC) or the pulse width modulation (PWM) outputs of the micro-controller.

An interface was created for connection to the dual PWM outputs in favour of using an SPI compatible DAC. The PWM outputs of the ATMEL 8515 micro-controller were a convenient way to produce a DC voltage of magnitude 0 – 5V. As the output from the micro-controller was PWM it was not a smooth DC voltage and needed to be conditioned before it was of sufficient quality to be mixed with the original joystick voltage. As a PWM voltage is a pulse of varying duration, repeated at a fixed time interval, the controller would not accept an un-smoothed PWM voltage as a "safe" input voltage and would "trip", assuming a fault condition. The raw PWM voltage needed to be approximated to a joystick quality DC voltage before it was mixed with the joystick voltage.

A simple smoothing circuit was created and fed to a level shifting circuit, the circuit is shown in figure 4.23. This made the PWM voltage acceptable to the inputs of the controller. The level shifting circuit was added to make best use of the PWM range. A certain PWM value could be set as the "null" value. At this value (0V) the PWMs did not change the input to the controller. The PWM value could be change to increase the controller voltage or to decrease it. The wheelchair motors were now controllable and the joystick inputs could be over-ridden or negated by altering register settings in the micro-controller.

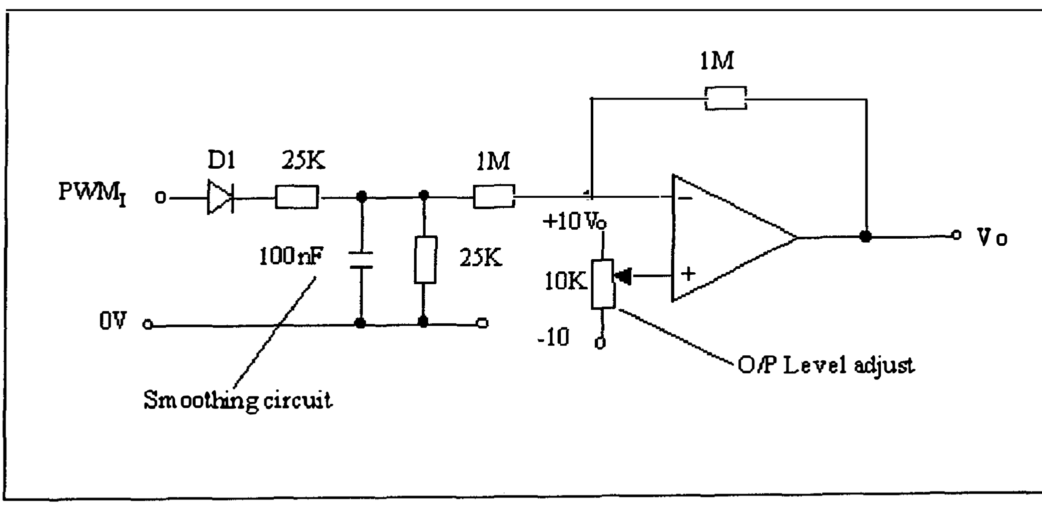


Figure 4.23. The PWM interface.

4.10 The joystick.

A Penny and Giles potentiometric joystick was used in this work. The joystick consisted of two potentiometers which provided information on the joystick position. In addition to the two wiper voltages from the potentiometers, the joystick required three reference voltages from the wheelchair controller (Hi Ref and two independent supplies of Lo Ref). The two (left and right) wiper voltages were called JS LT Wiper and JS RT Wiper.

The wiper connections were disconnected and re-routed to a summing amplifier. The joystick control signals could be modified by summing them with a voltage generated by the new micro-controller. The output of the summer was connected to the input of the wheelchair controller. Two ADC channels connected to the summer inputs monitored the joystick position. The new connections are shown in figure 4.24.

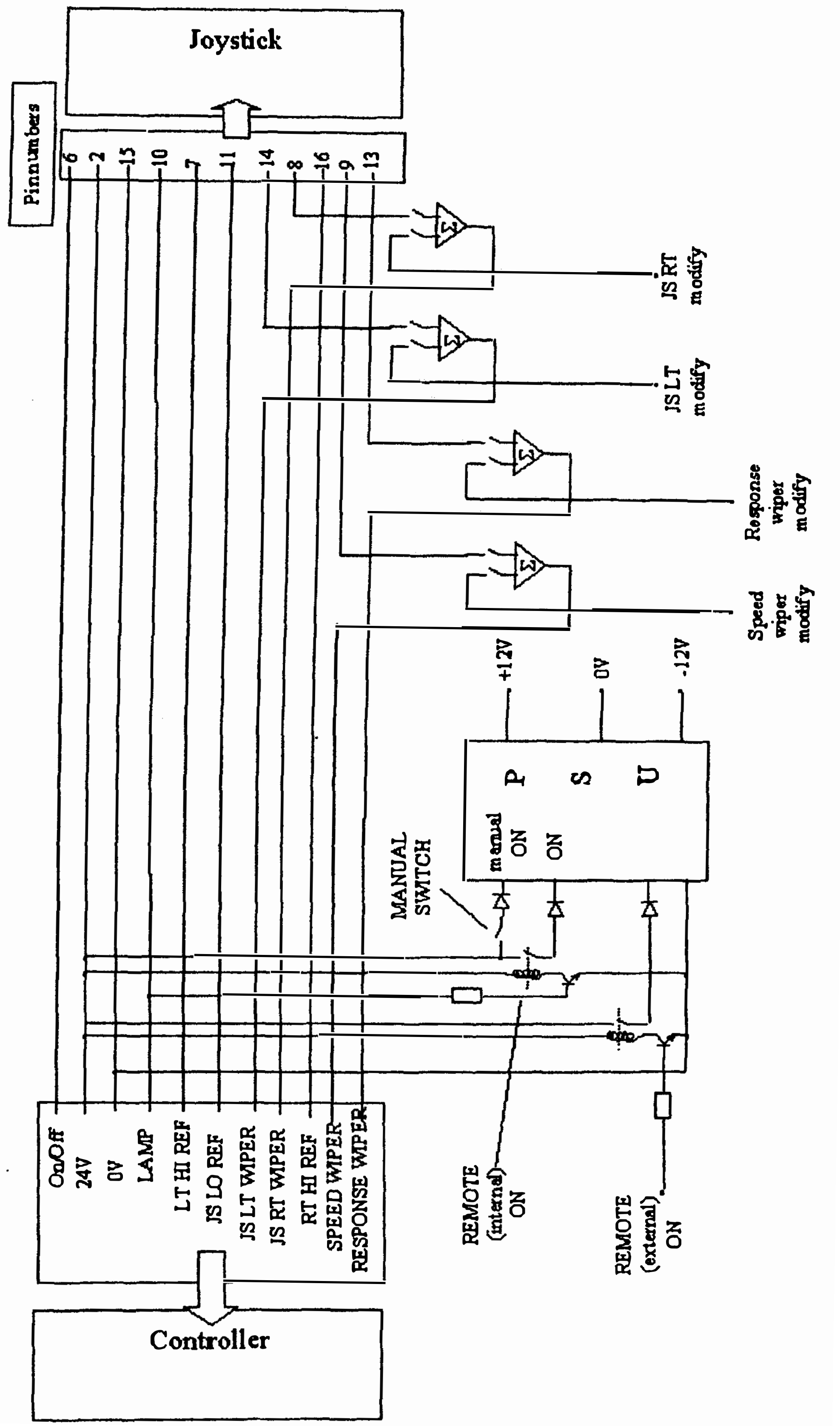


Figure 4.24. The wheelchair interface.

4.11 The prototype wheelchair.

When each of the components of the prototype intelligent wheelchair hardware had been tested separately, the prototype was constructed as shown in figure 4.25. A user could, if a suitable seat was fitted, drive the wheelchair. Clinical trials were not conducted during this work so a user was usually simulated by the author walking next to the wheelchair or riding on the rear cover. The link between the wheelchair and the joystick had been severed and the computer processed all the control information. The sensors could be activated and interrogated by the computer. The computer could be programmed to modify the path of the wheelchair. Alternatively the joystick control data could be processed and sent to the wheelchair controller without modification. In

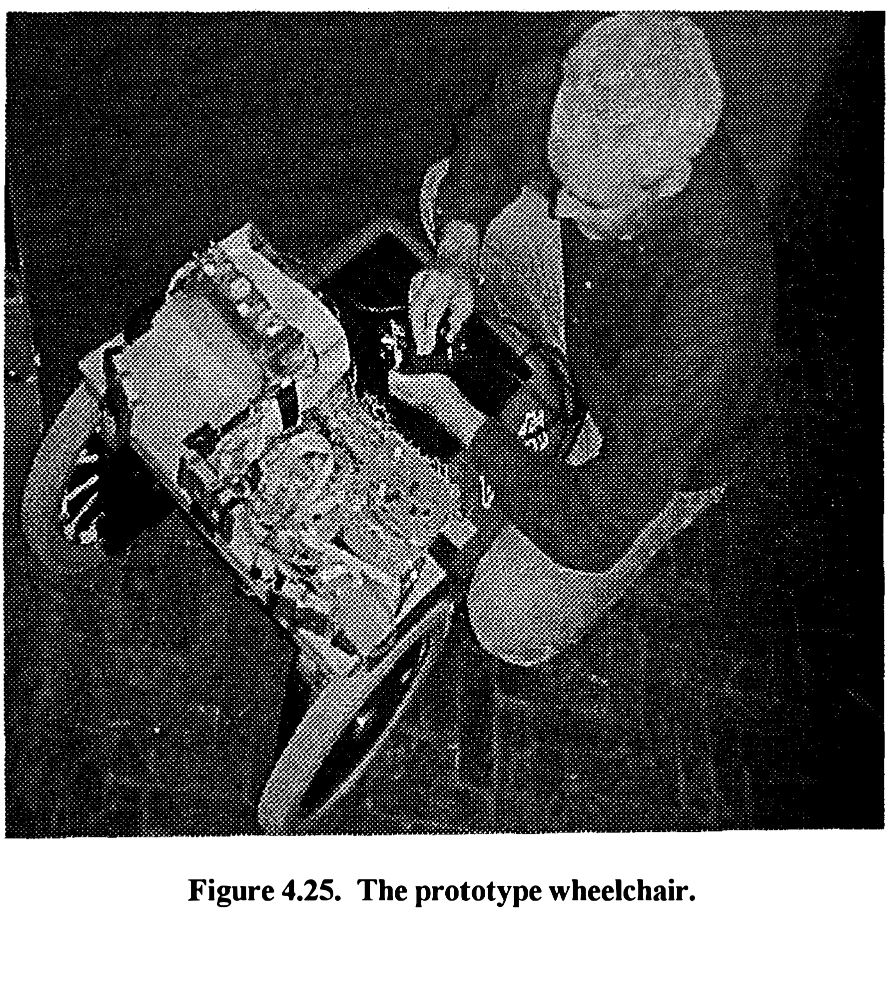


Figure 4.25. The prototype wheelchair.

this case the wheelchair responded to joystick inputs as if it was an unmodified wheelchair.

The next Chapter considers the new Expert Systems and Chapter 6 considers the results from work carried out on this hardware.

Chapter 5

Novel expert systems to generate modified paths.

The intelligent wheelchair interface created during this work was fitted between the existing joystick and the wheelchair controller as shown in figure 5.1. The first prototype contained a Philips 80C51 based micro-controller as described in Chapter 4 but this was later upgraded to an ATMEL AT90S8515 micro-controller. They were programmed to read the ultrasonic

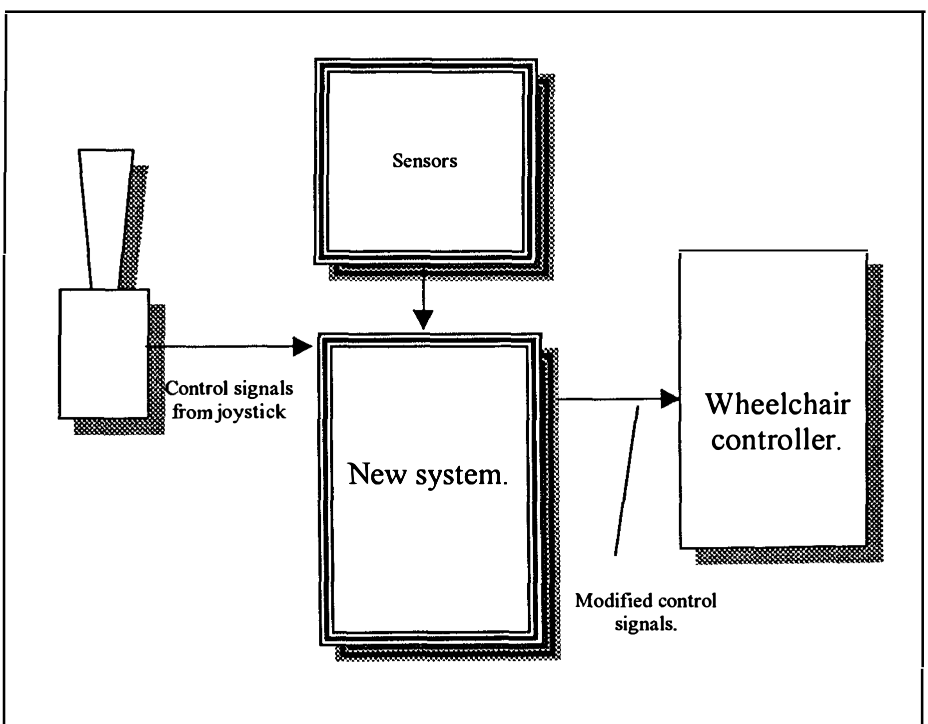


Figure 5.1. The modified wheelchair system.

sensors, read the joystick and modify the joystick signal fed to the controller if required. Novel algorithms created during this work generated the modification signals.

The novel algorithms recognised situations where the user may have needed assistance in steering the wheelchair. Expert systems were used to identify potentially hazardous situations and to recommend safe courses of action. The micro-controller code was written as a series of modules, which were then integrated into the main micro-controller program. As each module was created it was tested and debugged before it was added to the program structure. Conflicts and timing issues were tested, detected and removed as the code was integrated. This approach allowed problems to be identified and rectified early and before they became too embedded in software and made their detection difficult.

5.1. Micro-controller programming environments.

Some of the initial experimental work was completed using a Philips microcontroller. The target processor was changed to an ATMEL AVR micro-controller in order to take advantage of in-system-programming (ISP), flash memory, greater speed and an integrated programming environment. Two sets of prototype computer programs were created using the ATMEL micro-controller. The first used AVR BASIC as the high level programming language and the second used ANSI C.

5.1.1 Early experimental test rigs.

An experimental test rig was created using a PHILIPS S87C752 micro-controller for some of the initial prototyping and investigation work. A DS-750 Development Tool from Ceibo was used to program the PHILIPS micro-controller when writing code in assembly language.

The DS-750 was serially linked to a desktop Personal Computer and could emulate the microcontrollers using either the built-in clock oscillator or any other clock source connected to the microcontroller. This allowed algorithms to be simulated in the virtual reality environment described in Chapter 3 and then emulated on the development tool.

Emulation was carried out by programming an 87C752 microcontroller with the user software and an embedded monitor program. Three working modes were available: real-time, simulator and simulator plus. In the real-time mode the user software was executed transparently and without interfering with the microcontroller speed. Breakpoints could be added to stop program execution at a specific address. In the simulator modes, an additional microprocessor was used to take control of the 87C750 lines and to simulate its operation but this did not work in real-time and tended to be slow. The simulator mode allowed access to all the microcontroller functions (1/0, timers, etc.) and interacted with the hardware according to the user software execution or directly by means of emulator commands sent from the host computer.

The combination of the two available working modes allowed easier debugging of hardware and software functions.

Early versions of the ultrasonic sensors were developed on the experimental test rig using the Philips micro-controller. Later the target processor was changed from the PHILPS S87C752 to the ATMEL AT90S8515.

5.1.2 The first set of prototypes.

The first prototypes were created using an ATMEL micro-controller and AVR BASIC. This was in order to take advantage of in-system programming, flash memory, greater speed (nearly one instruction per clock cycle) and the Integrated Programming Environments available for the ATMEL range of AVR micro-controllers. An Integrated Development Environment from Silicon Studio (AVR Basic) was tested and selected as a suitable development environment. The AVR BASIC Toolset contained a comprehensive suite of code development tools for the Atmel AVR RISC micro-controller family. The package included a powerful Integrated Development Environment (IDE) which encompassed a BASIC compiler, macro assembler, editor and hex generator.

The AVR BASIC language allowed code to be written in a high level language, while still retaining the fast execution speed of assembler. The code was compiled from a BASIC source program into optimised AVR assembly instructions ready to be programmed into an AVR micro-controller device. It quickly became apparent that there were problems with AVR BASIC, as code did not function correctly or consistently. The manufacturer identified problems with the compiler code and further versions were released in an attempt to rescue the package. Considerable time was expended with AVR BASIC in an attempt to work around the software bugs built into the AVR BASIC programming environment. The decision was made to abandon AVR BASIC in favour of a more reliable development package.

5.1.2 The second set of prototypes.

A second set of prototypes were created using the ATMEL micro-controller and "The IAR Systems Embedded Workbench". ATMEL AT90 Code was created using The this environment which was an integrated development and debugging environment. Object code was produced and downloaded to the host processor using Atmel's AVR In- System-Programming (ISP) Development Tool for the AVR family of micro-controllers. The software allowed devices to be programmed with the most common file formats including all those generated by Atmel's AVR Assembler, the IAR Systems Embedded Workbench and a selection generated by IAR's AA90 Assembler.

The ISP software allowed free manipulation of both the EEPROM & Program Memory areas of the devices. This was very useful for applications where reading and editing the EEPROM areas was required.

5.2. The IAR integrated programming environment.

The IAR Systems Embedded Workbench was a flexible integrated environment for developing applications for a variety of different target processors [IAR systems (1999)]. It provided a convenient Windows interface for rapid development and debugging. Support for a number of different target processors could be added to the Embedded Workbench, and different target processors could be specified on a project by project basis.

The tools included a fast compiler, an efficient linker, a librarian, a syntax highlighting text editor, an automatic Make facility, and an optional "C-SPY" debugger. The tools are explained in more detail later in this section.

The Embedded workbench had the following features, [IAR systems (1999)]:

- Windows 95 and Windows NT compatible.
- Hierarchical project representation.
- Intuitive user interface, taking advantage of Windows 95 features.
- The Make utility recompiled, reassembled, and linked files only when necessary.

- Full integration between the Workbench tools and editor.
- Support of drag and drop features.
- Comprehensive hypertext help.

5.2.1 C compiler

The IAR Systems C Compiler for the AT90S family of microprocessors offered the standard features of the ANSI C language, plus many extensions designed to take advantage of the AT90S-specific facilities. The compiler was supplied with the IAR Systems Assembler for the AT90S, with which it was integrated, and shared linker and librarian manager tools.

5.2.2 Assembler

The IAR Systems AT90S Assembler was a powerful relocating macro assembler with a versatile set of directives. The assembler incorporated a high degree of compatibility with the microprocessor manufacturers own assemblers; to ensure that software originally developed using them could be transferred to the IAR Systems Assembler with little or no modification.

5.2.3 XLINK Linker

The IAR Systems XLINK Linker's function was to convert one or more relocatable object files produced by the IAR Systems Assembler or C Compiler to machine code for a specified target processor. It supported a wide range of industry-standard loader formats, in addition to the IAR Systems debug format used by the C-SPY high level debugger.

XLINK supported user libraries, and loaded only those modules that were actually needed by the program that was to be linked.

The final output produced by XLINK was an absolute, target-executable object file that could be programmed into an EPROM, downloaded to a hardware emulator, or run directly on the host using the IAR Systems C-SPY debugger.

5.2.4 XLIB Librarian.

The IAR Systems XLIB Librarian enabled manipulation of relocatable object files produced by the IAR Systems Assembler and C Compiler.

5.3 The ultrasonic sensors.

A simple prototype sensor system was created for the work described in this Dissertation. An ultrasonic rangefinder was chosen for its simplicity, rugged construction and low cost. To assist in the creation of suitable detection algorithms, a test rig was constructed so that tests could be conducted and different algorithms tried. The algorithms created from the tests were used in a micro-controller based prototype. The micro-controller controlled the sensor transmitter and receiver boards and processed the sensor data. The raw sensor data was processed to improve the reliability of the range data by mapping the sensor data onto a histogram certainty grid. The grid represented specific areas in front of the wheelchair.

5.3.1 Code for the prototype sensor test rig.

The sensors were prototyped using an 80486 desktop personal computer, an input/output card and interfaces. The prototype code was written in QuickBasic 4.5. The use of a high level language during the initial stages of development allowed rapid modification and debugging of new code. The sensor prototyping hardware has been described in Chapter 4. The prototyping code was used to experiment with different sampling rates, detection algorithms, pulse lengths, and target ranges. The results were saved as a file on a disc, imported to

Microsoft EXCEL and could be displayed on the computer monitor.

Test data were analysed using a QuickBasic program and then as part of the work described in this Dissertation. The program examined the sampled data and was used to create and develop the detection algorithms "off-line". The algorithms were later converted to low-level code and implemented in a micro-controller.

The prototyping code flow diagram is shown in figure 5.2 and a data analysis (object detection) program flow diagram is shown in figure 5.3. The QuickBasic 4.5 program listing, a sample test file, a data analysis program, and a raw data graph and gradient plot are included in Appendix A. The prototype was useful as it allowed system parameters to be selected for the micro-controller sensor prototype whilst using high level user interfaces (personal computer keyboard and monitor).

The hierarchical structure of the code is shown in figure 5.4. This is similar to the general hierarchical structure described by Sanders (1993) and Stott (1997). The supervisory level controlled the frequency and order of execution for the lower levels and could interface to other devices to allow integration of the expert system with other wheelchair controllers and rehabilitation equipment.

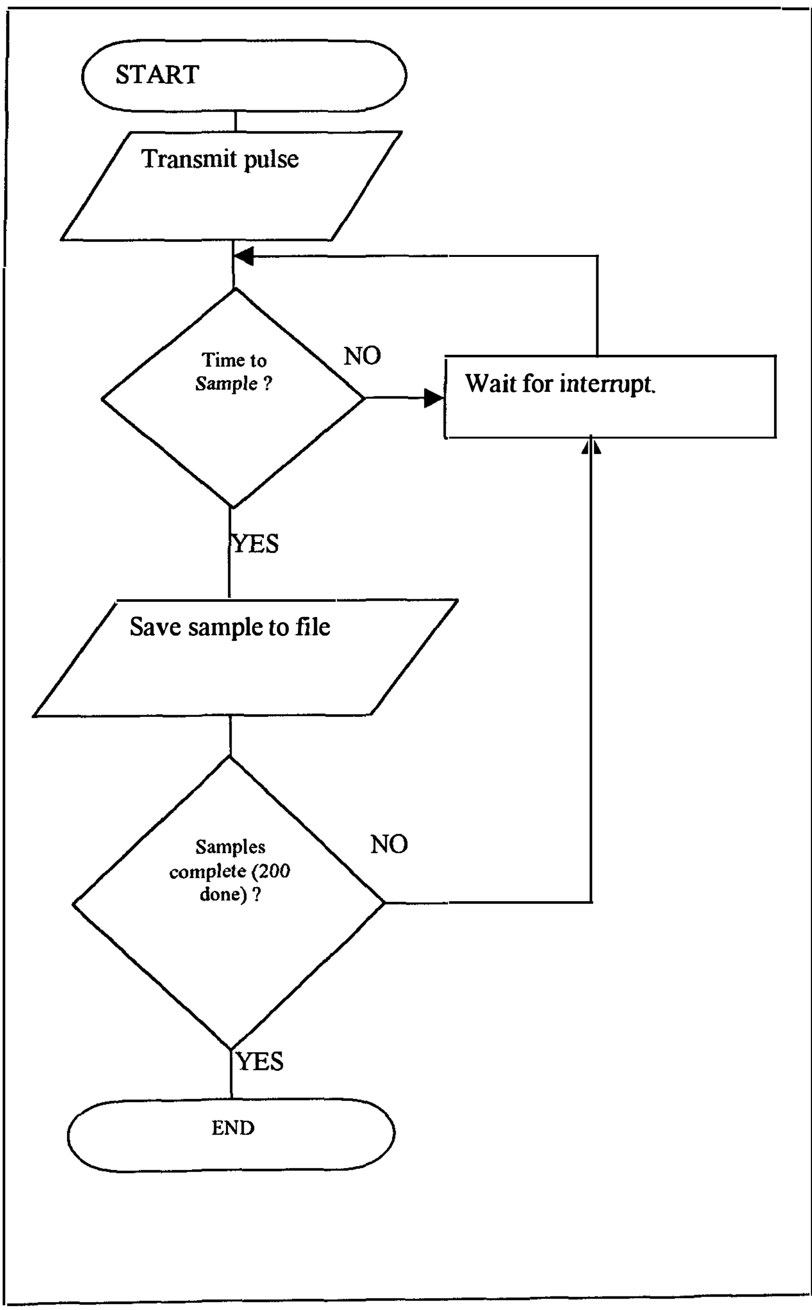


Figure 5.2. The flow diagram for the test rig code.

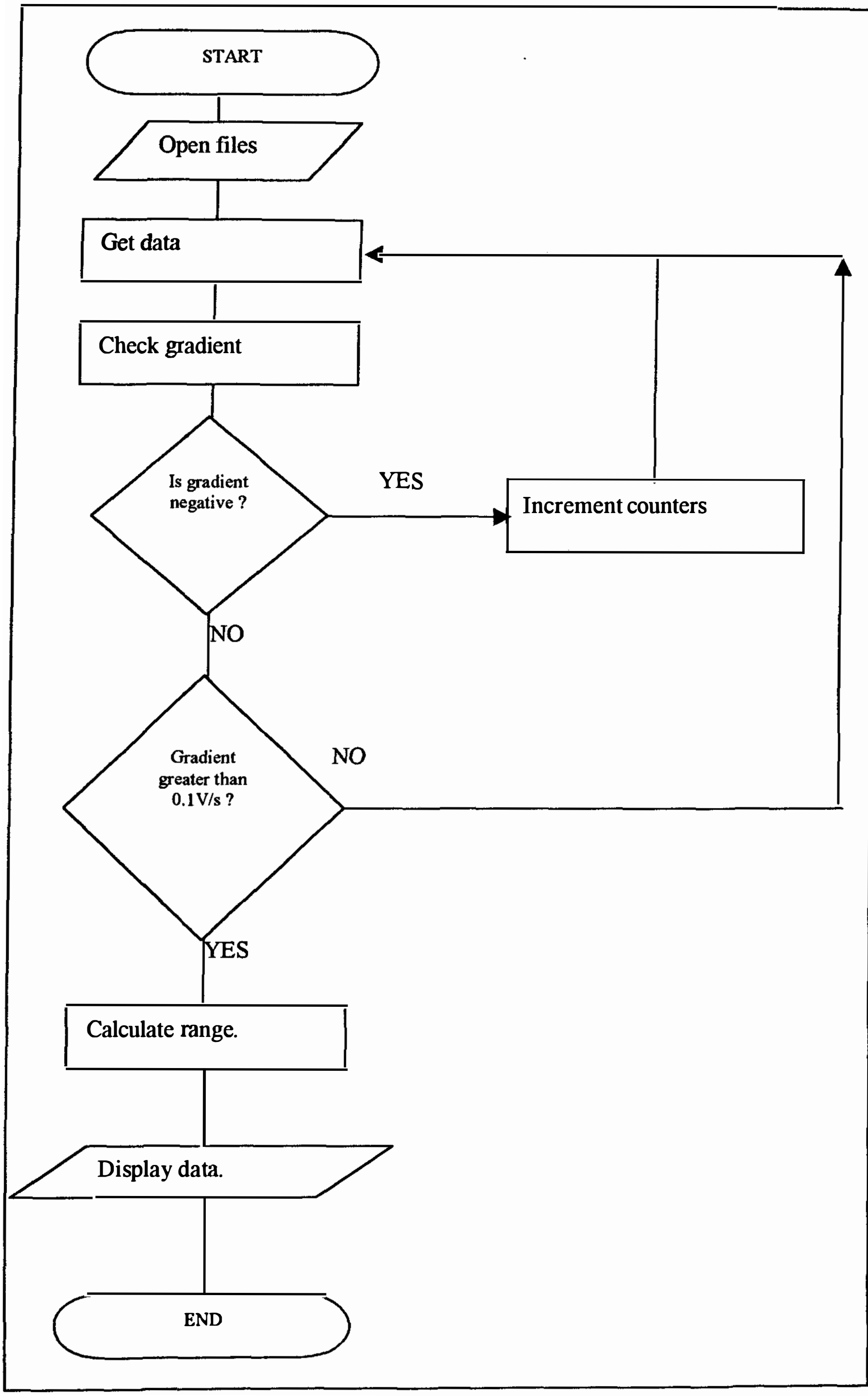


Figure 5.3. The flow diagram for the object detection algorithm.

5.3.2 The adaptive range finder.

Simple range finders were created that used a single pulse of ultrasonic energy to detect objects. The transmitters are described in Chapter 4 and it can be seen that the transmitter required a pulse of 3ms duration to reach maximum output power. A long pulse therefore contained more energy and so remained capable of detecting an object at a greater range. If the speed of sound in air is assumed to be 330m/s, the physical length of a 3ms pulse of sound is 0.99m. Allowing for the pulse to leave the transmitter, bounce off an object and return to the receiver, the minimum range that a 3ms pulse could be used for was 0.5m. At shorter ranges the pulse energy required was reduced. Pulse lengths of 10us, 100us, 500us and 1ms

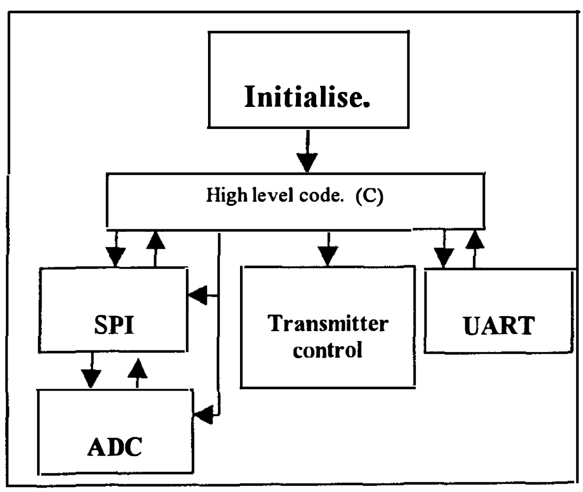


Figure 5.4. Hierarchical structure of the code.

were examined during tests and experiments. A range finder prototype was created to automatically switch between pulse widths as the range changed, and if there was no object detected, the range finder would “hunt for a detection by systematically raising the pulse length to increase the range. A flow diagram of the “adaptive” range finder is shown in figure 5.5. Work on the adaptive range finder was discontinued in favour of simpler implementations because multiple targets tended to appear and disappear as the wheelchair moved around and it was sometimes difficult for the adaptive range finder to lock on to a target.

5.4 Histogrammic mapping.

Ultrasonic sensors tended to be noisy and return misreads. Methods for filtering out misreads were considered to improve the reliability of detections. A method based on Histogramic In-Motion Mapping [Borenstien and Koren (1991)] was selected to improve sensor reliability.

Array elements representing the area in which an object was detected were incremented by a higher number, for example, three. Other array elements were decremented by a lower number, for example, one. The arrays typically had a maximum value of 15 and a minimum value of zero. Figure 5.6 shows an example of the simple three-element histogrammic representation of the environment and the position of an object in the third element causing that element to ramp up. An object occupying a grid element would cause that element to quickly ramp in value to the maximum. Random misreads in the other elements incremented that element temporarily, but the value of the false reads were decremented each time the sensors acquired more data. If the object moved to a different element, the new element quickly ramped up to its maximum value and the old element ramped down to the noise level. A reliable range could be acquired within half a second. Figure 5.7 shows the structure of the histogrammic object detection process.

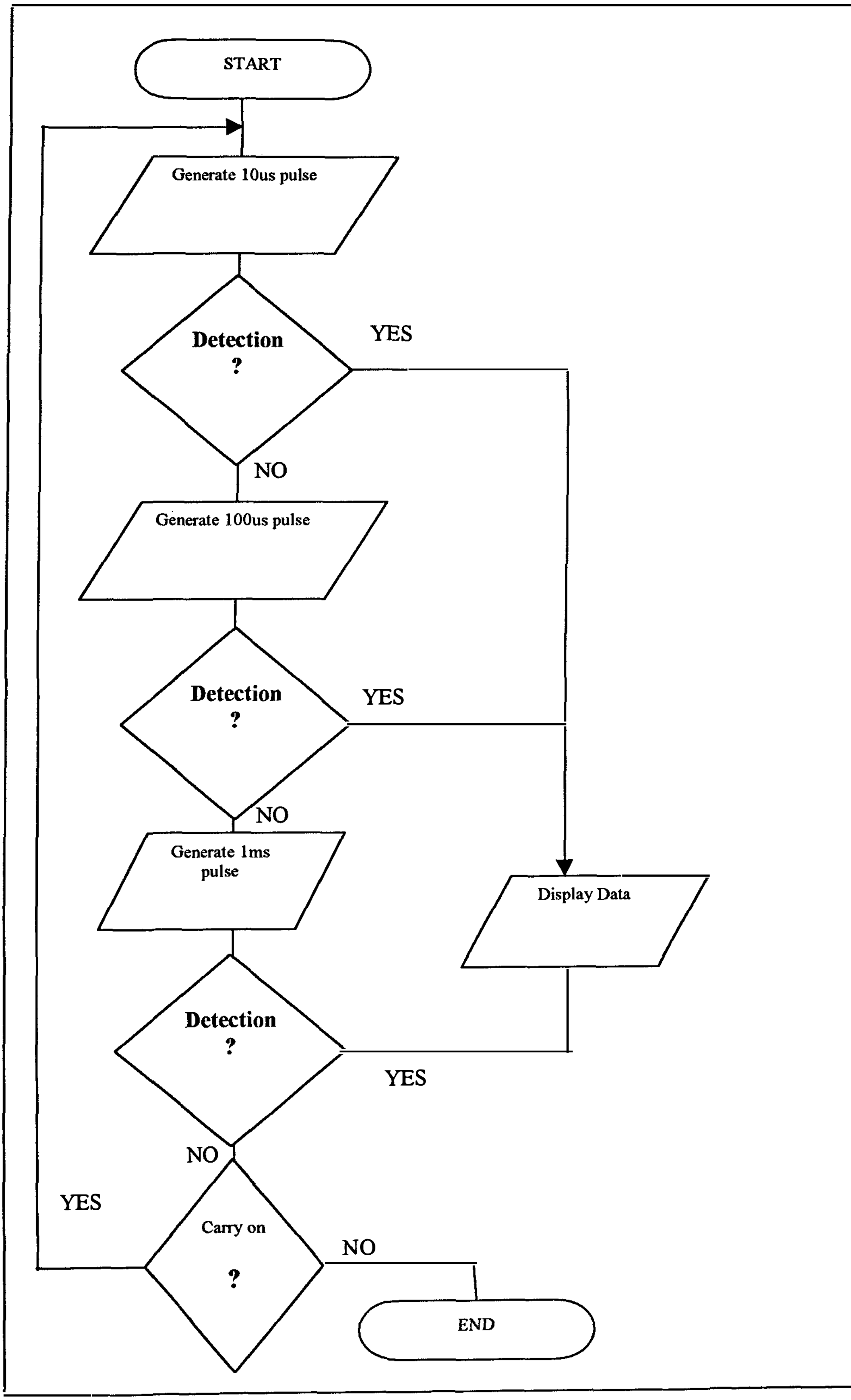


Figure 5.5. Flow diagram of the adaptive range finder code.

5.4.1 The sensor representations.

The volume in front of each sensor was divided into a simple grid of three volumes as shown in figure 5.8. In two dimensions, the areas were denoted as near, middle and far and were a simple representation of the ultrasonic beam. They were stored as an array in the micro-controller memory. When a range was returned, it was classified as being in the near, middle or far grid element. This is similar to the classifications used by Sanders

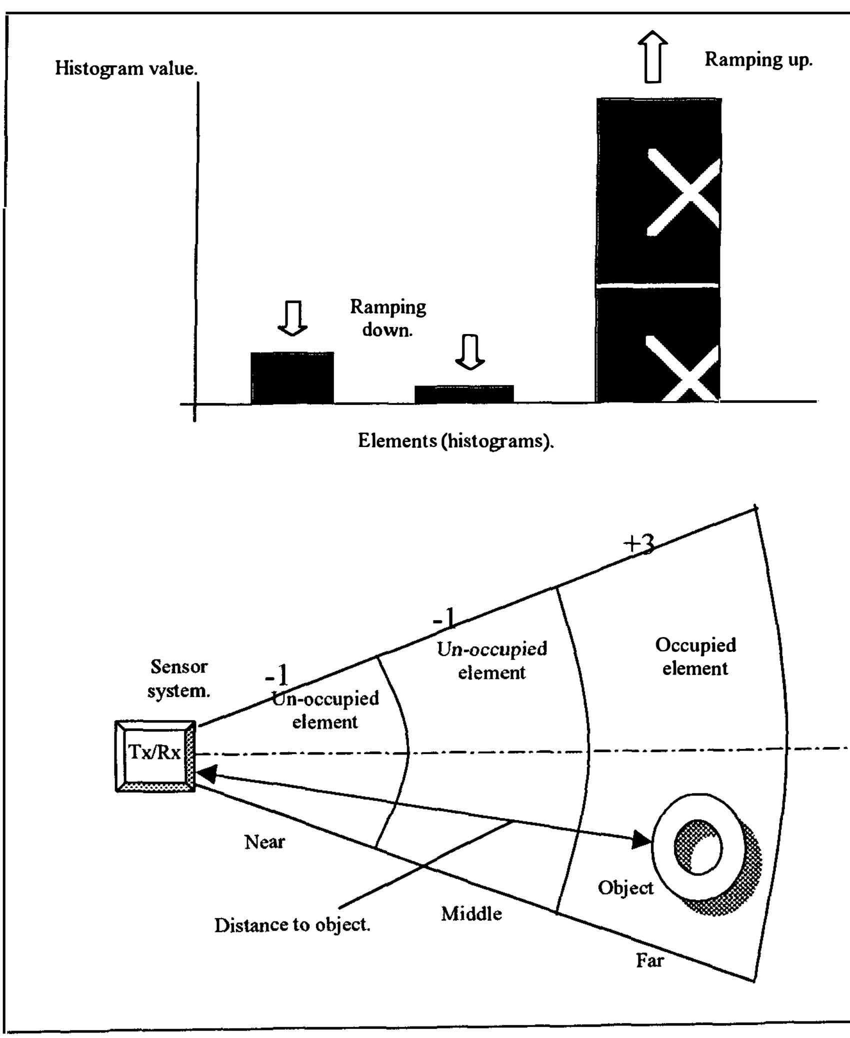


Figure 5.6. A representation of the sensor histograms.

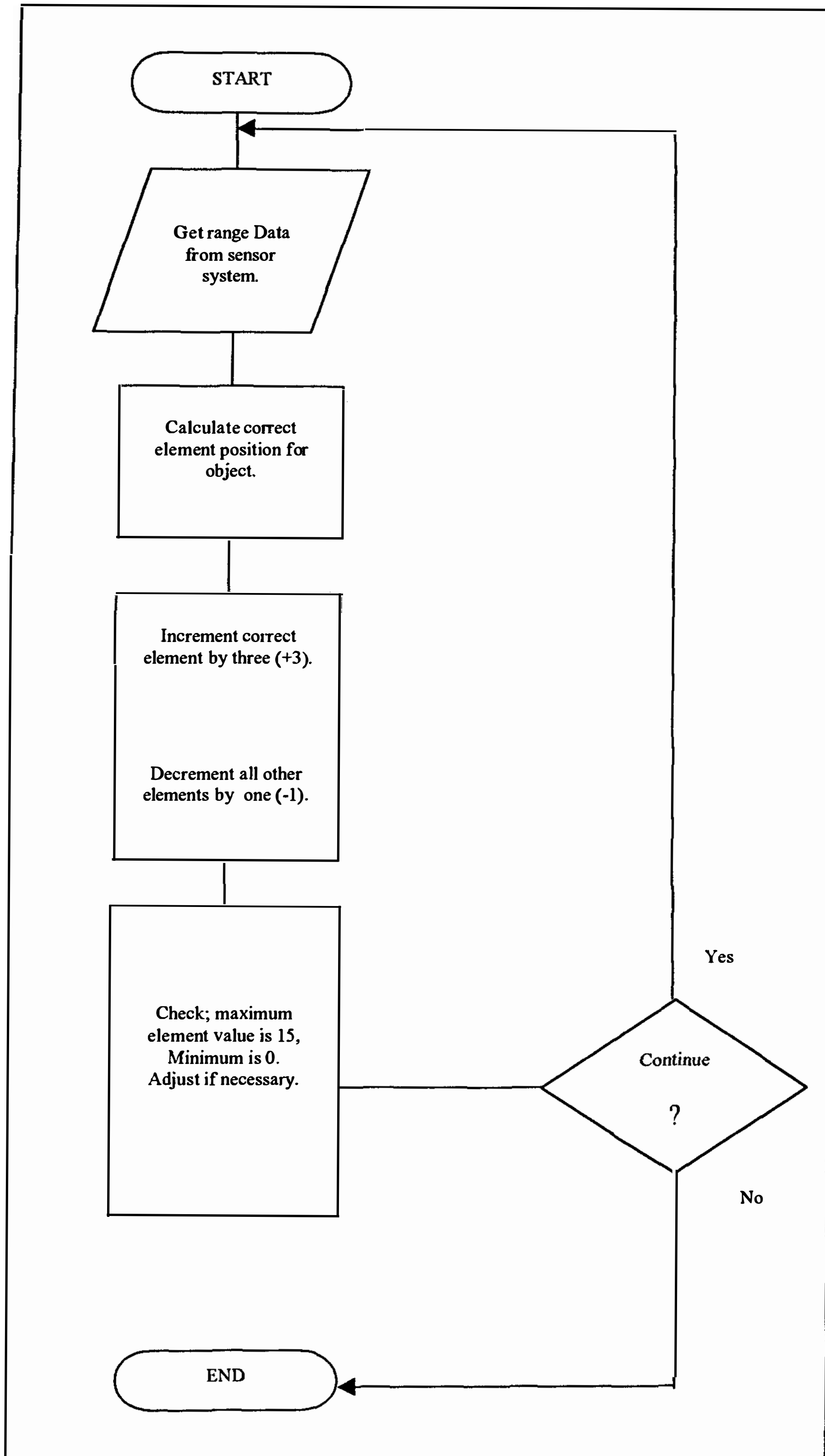


Figure 5.7. The structure of the histogrammic object detection process.

(1995). Three range finders were used on the wheelchair. The sensors were mounted so that their beams swept the area in front of the wheelchair. The arrangement of sensors is shown in figure 5.9. An explanation of the sensor algorithms is given in Section 5.9.4.

5.5 Pulse width modulation (PWM) code.

The ATMEL AT90S8515 was equipped with a dual PWM output. The PWM channels were used as the outputs to the trajectory modification circuitry. Two channels were needed because a separate channel controlled each wheelchair motor. The code to modify the PWM outputs is included in Appendix B1.

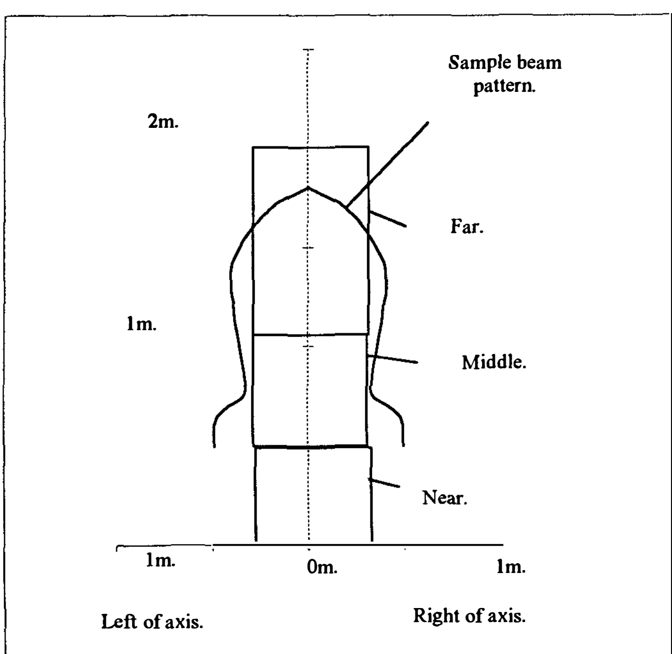


Figure 5.8. A single sensor representation.

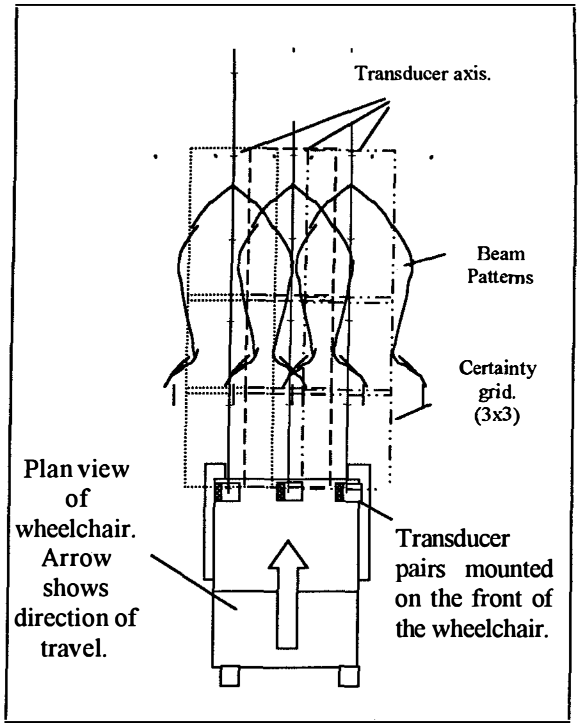


Figure 5.9. Representation of a three-sensor array.

5.6 Serial Peripheral Interface code.

The Serial Peripheral Interface (SPI) was a shift register that serially transmitted data bits to other SPI's. During a data transfer, one SPI compatible device acted as the "master" which controlled the data flow, while the other(s) acted as "slave" which had data shifted into and out by the master. The role of master could be shifted between devices as required and one master could simultaneously shift data into multiple slaves. For a full explanation of the SPI hardware, refer to Chapter 4.

The ATMEL AT90S8515 micro-controller acted exclusively as the master during this work, with the MAX1112 analogue to digital (ADC)converter being the only slave. The SPI was used to "clock" sampled sensor data into the micro-controller for analysis. Clocking a control byte into the data in (DIN) of the ADC started a conversion of the MAX1112 ADC. The control byte format is shown in figure 5.10. The ADC conversion-timing diagram shown in figure 5.11 shows the movement of data through the SPI during an ADC conversion. As the control byte was clocked into the ADC on the DIN, a dummy byte RB1 was clocked into the micro-controller. This was ignored. Two more bytes, this time containing zeros were then clocked into the ADC. In return, two bytes of data were clocked from data out (DOUT) of the ADC into the micro-controller. These two bytes were called RB2 and RB3. RB2 and RB3 contained the sampled data from the sensors although in order to extract the data, some shifting was required to re-align the data correctly. The code to access an ADC value is shown in Appendix B2.

BIT 7 (MSB)	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0 (LSB)
START	SEL2	SEL1	SEL0	UNI/$\overline{BIP}$	SGL/$\overline{DIF}$	PD1	PD0

BIT	NAME	DESCRIPTION
7 (MSB)	START	The first logic "1" bit after $\overline{CS}$ goes low defines the beginning of the control byte.
6 5 4	SEL2 SEL1 SEL0	Select which of the input channels are to be used for the conversion (Tables 1 and 2).
3	UNI/$\overline{BIP}$	1 = unipolar, 0 = bipolar. Selects unipolar or bipolar conversion mode (Table 4).
2	SGL/$\overline{DIF}$	1 = single ended, 0 = differential. Selects single-ended or differential conversions. In single-ended mode, input signal voltages are referred to COM. In differential mode, the voltage difference between two channels is measured. See Tables 1 and 2.
1	PD1	1 = fully operational, 0 = power-down. Selects fully operational or power-down mode.
0 (LSB)	PD0	1 = external clock mode, 0 = internal clock mode. Selects external or internal clock mode.

Figure 5.10. Description of the MAX1112 analogue to digital converter control byte.

5.7 Algorithms to interpret the joystick.

A Penny and Giles Potentiometric joystick was fitted to the wheelchair. It was a standard wheelchair joystick and contained two potentiometers to provide two channels of joystick information. A representation of the joystick output is shown in figure 5.12. The joystick position could be read by an analogue to digital converter (ADC) as a set of Cartesian co-ordinates. These co-ordinates are shown on figure 5.12 as the true voltage and as the values that the ADC would convert them to. Cartesian co-ordinates were not a convenient way to express the position of the joystick. The co-ordinates did not contain information on the direction and magnitude of the joystick signal. In order to interpret the joystick data in a more convenient manner, the Cartesian co-ordinates were converted to Polar form. Standard mathematical functions were used from the C libraries for arctan and square root functions for the Cartesian to polar conversion. This representation was easier to use with switches and other input devices.

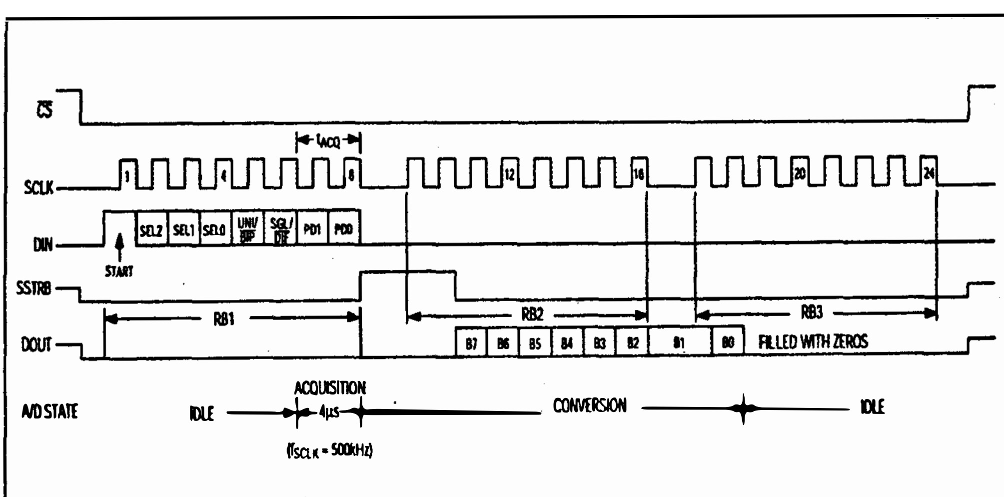


Figure 5.11. The SPI timing diagram.

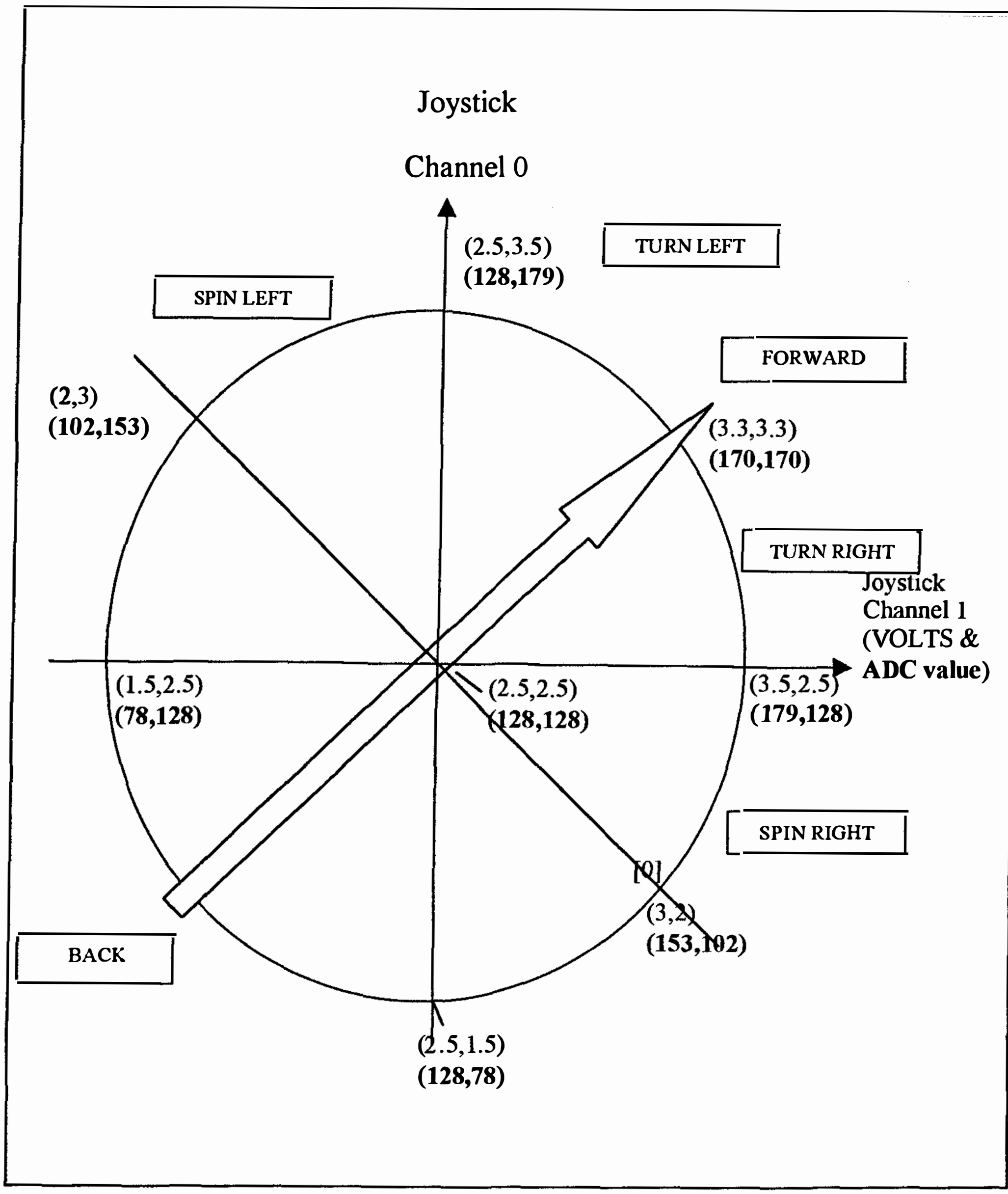


Figure 5.12. Map of joystick voltage and ADC values.

The joystick output was to be integrated to allow a level of confidence in the users intention to be assessed. There were two outputs of interest from the joystick; the direction and the requested speed. The joystick provided data in the form of two voltage channels. The two voltages were expressed as a Cartesian co-ordinate. Cartesian co-ordinates were not suitable for the integration technique so the joystick data was converted to polar co-ordinates using trigonometrical functions and Pythagoras' theorem. The joystick data was now in the form; $|J|\angle\theta$. where $|J|$ was the magnitude or how far the joystick had been pushed, and $\angle\theta$ was the angle that the joystick was being pushed. The magnitude could be integrated simply as it was a scalar quantity. The angle of the joystick introduced a directional element which could not be integrated. The joystick angular position was quantfied so that the approximate direction of the joystick could be estimated. This allowed the algorithms to measure the length of time that the joystick had been held in a consistent direction which helped the expert system to identify the wishes of the user. The joystick angles were defined as;

Spin left, **1.54 – 2.36 radians**

Spin right, **5.50 – 6.28 radians**

Turn left, **0.89 – 1.54 radians**

Turn right, **0.00 – 0.69 radians**

Forward, **0.69 – 0.89 radians**

Reverse. **2.36 – 5.50 radians**

Stop, **magnitude<16**

The sectors are indicated on figure 5.12. Two example joystick positions are shown in the figure for Turn left and Spin right. Reverse was not considered for this work as sensors were not fitted to the rear of the wheelchair.

The value for the angle and magnitude for the joystick were calculated using:

```
argument = JS0/JS1;          //opposite over adjacent (ATAN)

angle = atan(argument);      // joystick direction in radians
```

The magnitude was calculated using:

```
magnitude = sqrt((JS0*JS0)+(JS1*JS1));
```

where JS0 and JS1 were the Cartesian co-ordinates with the origin centred on the joystick stop position.

Magnitude and angle were then used to calculate the sector that the joystick was occupying. The example code listed in Appendix B3 used printfu (a simplified print function that used less memory space) to indicate that the code had correctly calculated the sector for the joystick. The printfu function could be replaced by any function to react to the joystick position for testing or operation of the wheelchair.

The position and confidence of the joystick could be expressed as an array as shown in figure 5.13. Each joystick sector contained two array values. A value of "Angle Confidence" of between 0 and 15 indicated the certainty that the joystick was being held in that sector. A value of "Magnitude" indicated the position of the joystick with regards the speed that was demanded of the wheelchair. A histogrammic representation could then be used as a pseudo-integrator. If the joystick was held in a certain position, the array element relating to that position could be incremented to raise its overall value. All other array elements could then be decremented to reduce their importance. The array with the highest value could be used as the latest and most confident position for the joystick to be in. The joystick occupying a joystick array element would cause that element to quickly ramp in value to the maximum. Random joystick action in the other elements incremented that element temporarily, but the value of the false reads were decremented each time new data

Angle	Angle confidence	Magnitude
Spin left	0 - 15	0 - 15
Turn left	0 - 15	0 - 15
Forward	0 - 15	0 - 15
Turn right	0 - 15	0 - 15
Spin right	0 - 15	0 - 15
Stop	0 - 15	0 - 15

Figure 5.13. The joystick position and confidence array.

was acquired. If the joystick moved to a different element, the new element quickly ramped up to its maximum value and the old element ramped down to the noise level or zero.

The joystick position could be represented as a histogram where the highest histogram element represented the most likely direction for the user to be indicating as the desired direction. The histogram elements decayed rapidly if the joystick was moved from the sector that the element represented and the new element ramped up more slowly. An example histogram representation for the joystick is shown in figure 5.14. This method allowed for conditions such as cerebral palsy where a clear signal could not be identified.

The code to build and decay the joystick histograms was contained in module JSArray and is included in Appendix B4. The code tested the joystick position and angle, and indicated which sector the joystick was occupying. The appropriate element of the "angle confidence" (Aconf) was then increased by magnitude 40. All Aconf elements were then decreased in magnitude by 30 to decay the un-occupied elements. The occupied element was therefore subject to an increase of 10 magnitude and all other elements were subject to a decrease in magnitude of 30. This allowed the histogram elements to decay rapidly and build in value more slowly. A joystick array element was able to increase to its maximum value of 225 in a minimum time of 0.5 seconds (approximately) and decay to zero in approximately 170ms. The ramping and delay weighting factors were determined experimentally by driving the wheelchair with several different weighting factors in operation. The delay induced in the response of the wheelchair by the weighting factors could be dependent on the individual user and was not considered as part of the research described in this Dissertation. In practice they would be set by a rehabilitation engineer or clinical care staff.

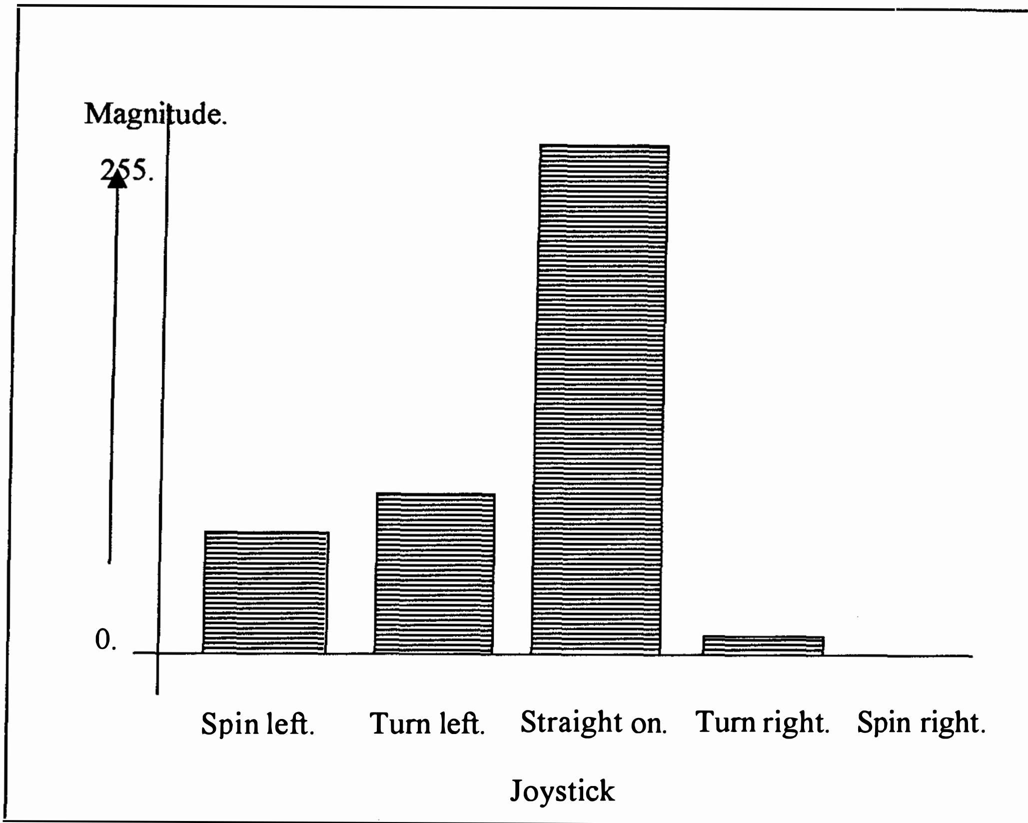


Figure 5.14. A representation of the joystick using histograms.

5.8 Creating an intelligent expert system.

To create an expert system, the expert knowledge of the process or problem had to be acquired. A domain expert is a person who has accumulated a great deal of skill in handling problems in a particular area called the domain of expertise. An expert powered wheelchair driver and an expert rehabilitation engineer or clinician could be a domain expert. Knowledge engineering is the extraction of useful knowledge from domain experts for use by computers.

5.8.1 Knowledge acquisition.

Knowledge acquisition is the process of extracting knowledge from the expert sources. This knowledge could be organised and structured to be used by the problem solving mechanisms of an expert system. This process was described in Section 2.8.4, Knowledge Representation.

To an expert powered wheelchair driver, driving a powered wheelchair in an unstructured environment was relatively easy if the driver understood the concepts involved with mobility. That was; the concept of space, obstructions and movement into empty space. The knowledge required to drive a powered wheelchair to most people is intuitive. A little time is usually needed to familiarise oneself with the response of the expert system. Some people are more naturally dextrous and could learn to drive in less time than others. To acquire the knowledge of wheelchair driving for this work, the wheelchair was driven by the author and the knowledge was transferred to the expert system directly.

Subsequent meetings with experts in rehabilitation technology involved discussion of specific problem areas, focusing on why particular decisions were made and their effects on the user. Three categories of knowledge acquisition techniques were considered [Marik and Vlcek(1992)]:

- Interviewing.
- Problem dependent systems for knowledge formalisation.
- Induction.

The large amount of time required to acquire the knowledge and to develop the rules could have been a problem. It had been estimated that interview based techniques produced between two and five units of knowledge per day [Jackson (1990)]. This was commonly known as the knowledge acquisition bottleneck [Feigenbaum(1977)]. There were a number of reasons for this:

- Specialist fields had their own Jargon' which the knowledge engineer needed to understand.
- The problem of defining a mathematical theory or deterministic model to represent the knowledge elicited.
- The knowledge engineer had to learn how to apply the expert knowledge obtained and in which situations different aspects of knowledge were relevant.
- The large volume of knowledge.
- The availability of the expert sources.

Buchanan *et al*(1983) described knowledge acquisition in terms of five stages of development: Identification; Conceptualisation; Formalisation; Implementation; Testing. Reviews of knowledge acquisition research can be found in Neale(1988), Hudson (1997) and Arciszewski & Rossman(1992).

Once the knowledge acquisition process had been completed the knowledge could be analysed before implementation. Wellinga and Breuker(1986) described an approach based on five levels of analysis: Knowledge Identification; Knowledge Conceptualisation; Epistemological Analysis; Logical Analysis; Implementational Analysis. Distinction was made between identification, conceptualisation and formalisation as in the analysis by Buchanan *et al*(1983). An alternative knowledge level analysis was called Ontological Analysis [Alexander *et al*(1986)]. This approach described the knowledge in terms of entities, the relationships between these and the transformations that could take place by the performance of defined tasks.

An Expert System is a type of knowledge-based system, and deals with matters of realistic complexity that normally require a considerable amount of human expertise.

An Expert System must be capable of justifying solutions to convince the user that it has made a valid choice. The expert system used in this work could justify itself by having fewer collisions than the manually controlled powered wheelchair.

Rules were intended as generative rules of behaviour. Given some set of inputs the rules determined what the output should be.

5.8.2 An example rule-based expert program.

As an exercise a simple rule based program was written in QBASIC. A "Number puzzle game", (similar to the "sliding tile game familiar to many children) solving program was written to test AI program techniques. The program used a "hill climbing" algorithm to find a solution. A pre-programmed puzzle matrix was included, along with a target array. As the program was stepped towards a solution, the space was "moved" within the matrix under a set of rules until the example matrix matched the target matrix. To change the target matrix, the matrix was manually re-keyed. This example program is listed at Appendix C1

A problem that could be typical with control systems occurred during program execution. The hill-climbing algorithm failed to find a solution to some target matrix configurations. A local minima was found but the solution algorithms became trapped (in a valley to continue the hill climbing metaphor) and the complete solution was not be found. Extra algorithms needed to be created to detect local minima and force a solution possibly by creating a pseudo random set of moves.

5.8.3 The first prototype intelligent wheelchair Expert System.

The novel expert system created during this work augmented the user of the wheelchair. It was important that the intelligent wheelchair operated in real time in order to assist the user. There were two real time inputs to the expert system; the users input (a joystick for example) and the sensors. The user indicated a speed and direction for the wheelchair. The sensors gathered information about the environment. A sensor expert then analysed the sensor information and made a recommendation for a path that would prevent collisions. The data inputs often conflicted. Another expert, called the Fuzzy Mixer considered both inputs and was responsible for the final outputs to the motor controller. Joystick monitor was responsible for interpreting the wishes of the user. Variables such as joystick position and consistency were examined by the joystick monitor to assess the desired wheelchair trajectory. An example of a simple rule based Expert system is shown in figure 5.15.

This basic structure was used to create an initial framework for a rule based Expert system for the work described in this Dissertation. A major difference between the example of Figure 5.15 and the requirements of this work was that there would be an additional demand source applied to the Expert system. As well as a user, sensors fed information to the expert system. These two sets of data were often in conflict. The Fuzzy mixer was responsible for apportioning levels of control between these data. The Expertise of the Expert System was expressed by altering the path of the wheelchair. Methods by which the user could veto the suggestions of the Expert System are discussed later in this Chapter.

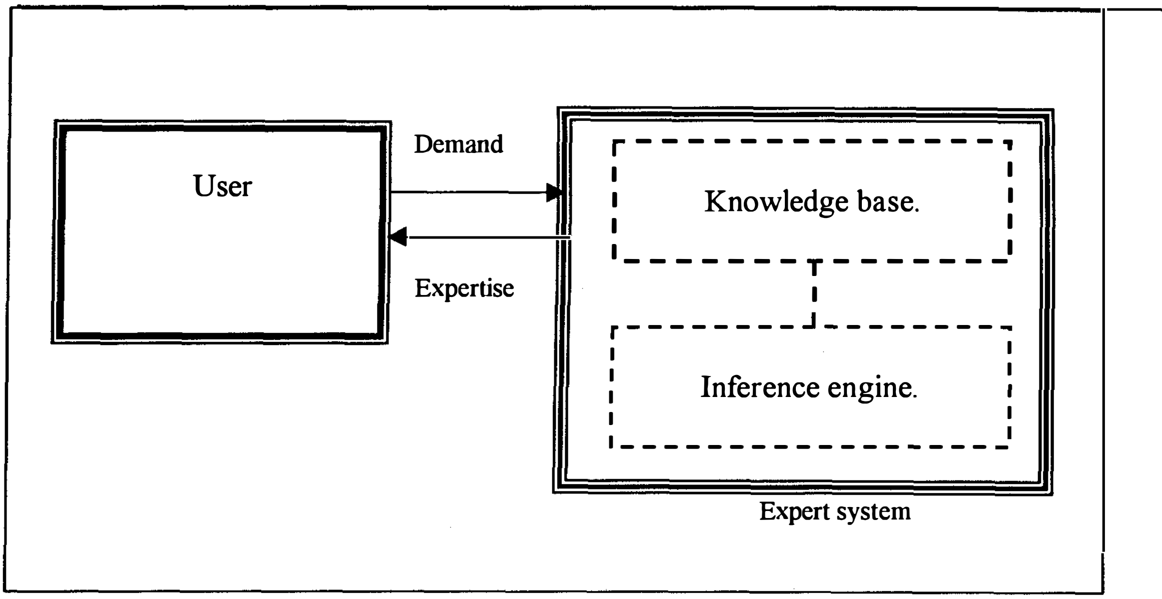


Figure 5.15. An example of a simple rule based Expert system.

The novel expert control systems described in this Dissertation were initially implemented on a simple rule based Expert system. The structure of this implementation is shown in Figure 5.16.

The first prototype computer program consisted of the following components:

(a) Fuzzy mixer.

(b) Joystick monitor.

(c) Sensor Expert.

(d) Doorway.

(a). Fuzzy mixer The Fuzzy mixer apportioned control effort between joystick and sensors. It matched the joystick and sensors recommendations, examined conflicts and kept the controller voltage within parameters. The fuzzy mixer was able to override any input with "Proximity Stop", generated by "Doorway". The Proximity Stop function was a failsafe anti-collision mode that stopped the wheelchair from crashing. Fuzzy mixer received information (or advice) from Sensor expert, Joystick monitor and Doorway.

Fuzzy mixer took joystick confidence values and the sensor information and mixed them. Low joystick confidence meant that the sensor information was more important if the wheelchair was to avoid obstacles and drive safely in the direction approximately set by the joystick. High confidence in the joystick, meaning that it accurately reflected the wishes of the user resulted in the sensor data having less influence on the trajectory. When the sensor byte was zero (000000) or the sensor expert was recommending the same course of action as the joystick indicated, joystick confidence was maximum (1) and the user had complete control. If the joystick was directed to make the wheelchair hit a wall, joystick

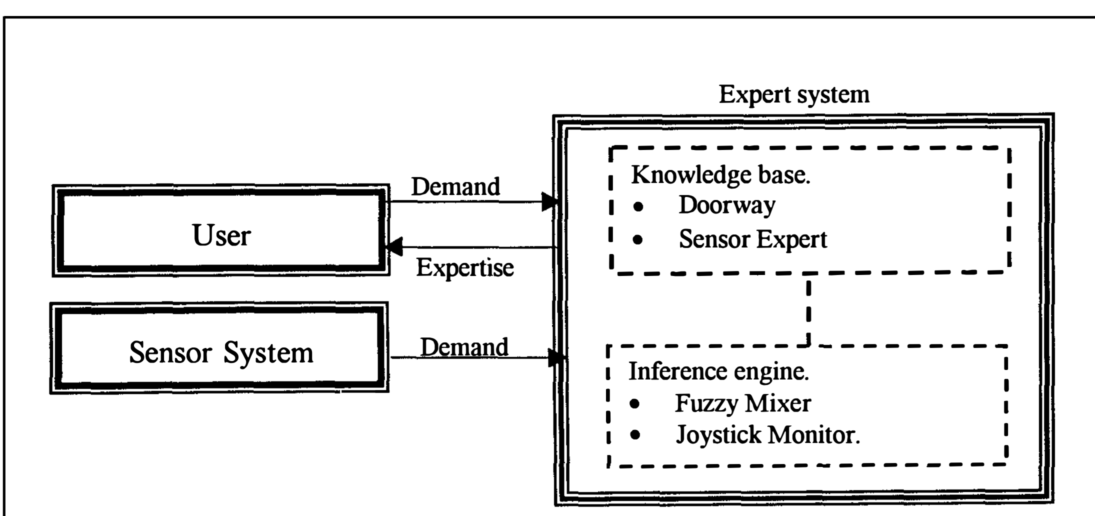


Figure 5.16. The new systems implemented on a simple rule based Expert system.

effect was initially reduced but if joystick position was reinforced or held in the same position, the joystick confidence increased and the wheelchair was allowed to move towards the wall. When it reached a pre set distance, the proximity stop activated and the wheelchair stopped near to the wall.

(b) Joystick monitor. Joystick monitor checked for changes in the joystick position and consistency. A steady joystick position indicated a desire to go there. A joystick made to move randomly about may have indicated an unsure or out of control driver and the expert system needed to rely more on the sensors to help to steer the wheelchair.

(c) Sensor Expert. The Sensor Expert applied knowledge of sensor combinations. Sensor Expert created a sensor grid and information was extracted from it. The Sensor expert was only concerned with making a recommendation to the expert system on the course of action that would take the wheelchair away from an object or would prevent the wheelchair from hitting it. The Sensor Expert would not consider the wishes of the user.

(d) Doorway. Doorway was a program that extracted information from data supplied by Sensor expert. It was an object avoidance program that avoided objects through a "distance function" algorithm. Doorway was over-ridden or allowed to affect the trajectory of the wheelchair by the Fuzzy mixer. The distance to an object measured by sensors "left" and "right" was used to determine how the wheelchair should react to the environment. A similar technique was used by Stott *et al*(1995) to steer an early prototype intelligent wheelchair and guide a wheelchair through a doorway. This was later described in Stott (1997).

The information from the joystick was combined with the information from the sensors, so that:

Output(left) = Input(left) - F(right)

Output(right) = Input(right) - F(left)

where output() was the resultant voltage for the wheelchair controller, Input() was the voltage from the joystick, and F() was the distance function value generated by the sensors. All the variables were vector quantities, having two values, one for each wheel (left and right). This output was presented to the wheelchair controller driving the wheels.

"Doorway" was effective at turning the wheelchair away from the nearest object, slowing the wheelchair down smoothly as it became closer to objects and in centralising the wheelchair between two objects such as the frames of a door.

5.9 The first prototype algorithms.

To assist a user of a powered wheelchair, algorithms were created to mix data from the joystick and the ultrasonic sensors. Algorithms were also created to interpret the information from the sensors and the joystick. The interpretation of the data was important as it influenced the way in which the expert system was able to react to the changing position of the wheelchair and needs of the user.

5.9.1 The relationship between the sensors and the joystick.

The fuzzy mixer controlled the relationship between the joystick and the sensors. As described in Section 5.9.3, the fuzzy mixer apportioned control to the joystick or sensors depending on the environmental conditions or the wishes of the user. The instantaneous relationship could be; all joystick and no sensors, all sensors and no joystick or somewhere between the two extremes. This is demonstrated in figure 5.17.

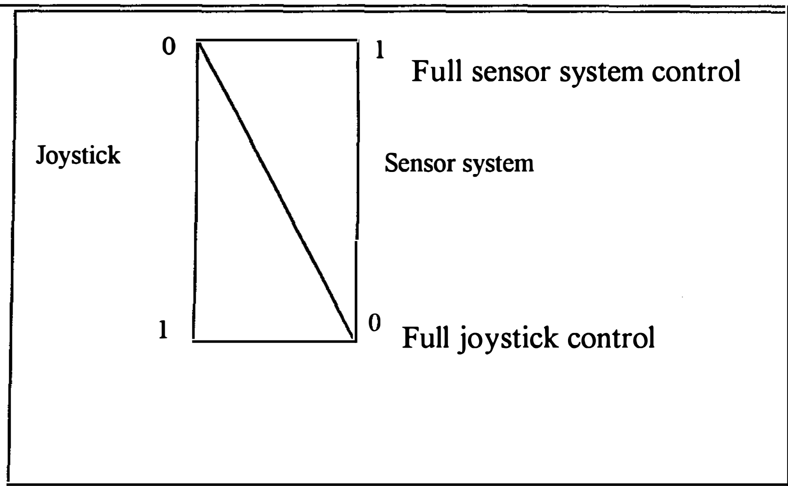


Figure 5.17. The relationship between the joystick and the sensor input.

5.9.2 Fuzzy mixer expert algorithms.

The fuzzy mixer was constantly assessing the two inputs. Control algorithms apportioned control between two inputs. The algorithms are shown below.

TargetLeft = (((JS0*Aconf[Joysticksector])+((TargetLeft-125)*(255-Aconf[Joysticksector]))/255)+125;

TargetRight = (((JS1*Aconf[Joysticksector])+((TargetRight-125)*(255-Aconf[Joysticksector]))/255)+125;

Where;	TargetLeft/Right	= Desired controller voltages.
	JS0/1	= Actual joystick values.
	Aconf[]	= Joystick confidence value.

The algorithms used distance functions to create the Target values for the left and right controller voltages. The distance functions are shown below.

TargetLeft = 2.5*result[1] + 110;

TargetRight = 2.5*result[0] + 110;

Where; result[] = instantaneous range from the sensors.

The result[] from the sensors was scaled (*2.5) and a constant (110) added. This converted the result (sensor) data to a form compatible with the Target (ADC) data.

The joystick was capable of defining the user input accurately and this was important for driving the wheelchair. The expert system needed to recognise the position of the joystick in order to make an assessment of the wishes of the user. To do this the joystick map was divided into sectors: Forward, Turn right, Turn left, Spin right, Spin left, Stop and Back. The arrangement of the sectors is shown in figure 5.12. The position of the sectors was variable by re-coding. It is possible that this could be achieved automatically as part of an adaptive diagnostic program or manually by a user. The sector location could change in response to a user's condition changing such as a change in the mobility of the wrist for instance or a change in the user's cognitive abilities.

Factors to increase joystick confidence (**Aconf**[]).

Joystick agrees with the sensors,

Joystick held in a steady position (consistent),

Joystick position increased against sensor action.

Factors to decrease joystick confidence.

Joystick – sensor conflict,

Joystick not steady.

5.9.3 Assessing the joystick consistency.

If the average joystick position was calculated in real time, a smoothed joystick voltage waveform would be created. The average voltage would have been near to the instantaneous value but if the instantaneous voltage was rapidly changing, the instantaneous value would usually be substantially different to the average value. The instantaneous voltage in the case of figure 5.18 swung each side of the average and hence:

Usually, Actual voltage $\neq$ Average voltage

This showed a lack of consistency for the joystick operation and therefore the joystick confidence was lower.

In the case of figure 5.19 the instantaneous voltage was similar to the average voltage. This showed a higher level of control for the joystick user or a better understanding of how to drive the wheelchair in the situation present at that time. In this case, the joystick confidence was increased.

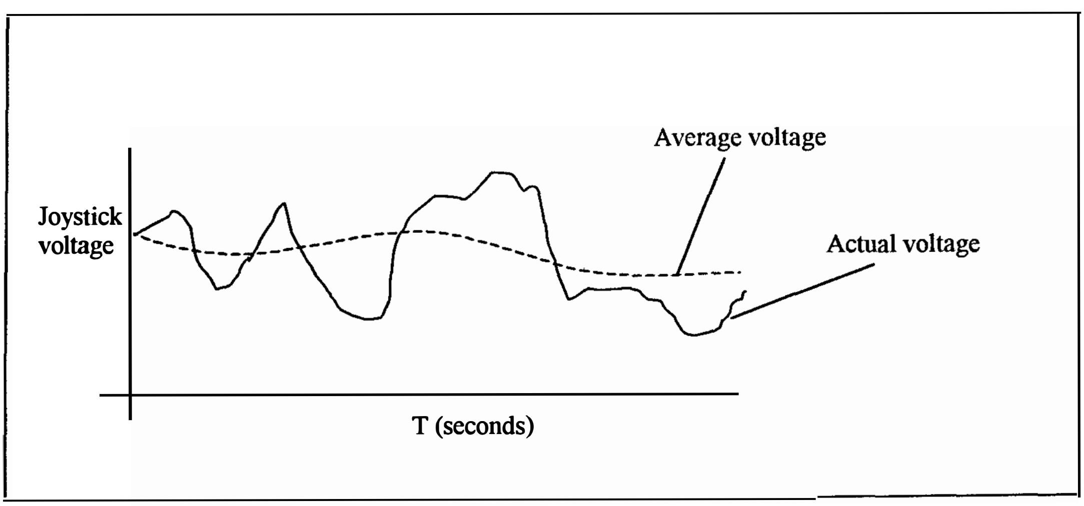


Figure 5.18. A unsteady joystick input.

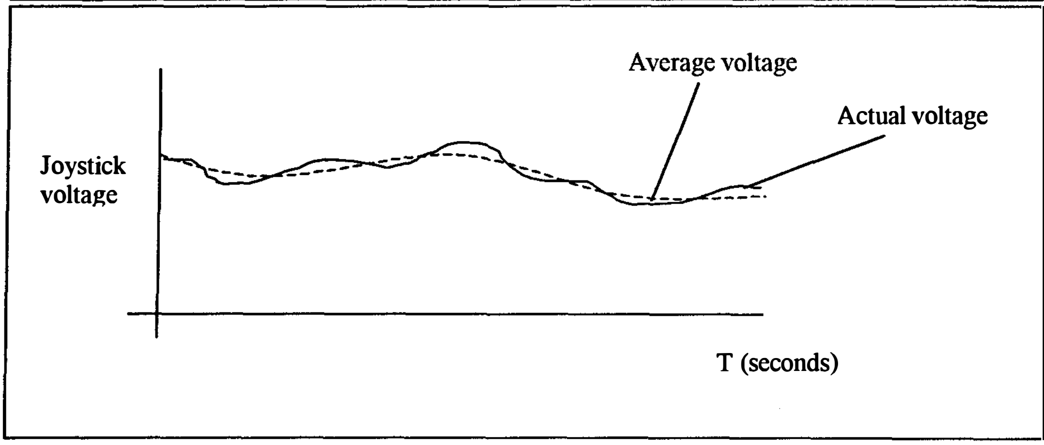


Figure 5.19. A steady joystick input.

It was clear that in order to assess the wishes and accuracy of the user a method was needed which allowed the joystick expert to monitor the joystick position. Simple averaging was a possibility but an Integration technique was created, examined and tested to improve the output from the joystick monitor.

5.9.4 Sensor expert algorithms.

The Sensor expert applied a set of algorithms to the sensor byte generated from the sensor information. There were seven actions that were considered in this work for the wheelchair to take. They were:

"Nothing"	meaning carry on under user control,
"Stop"	collision is imminent, stop immediately,
"Slow"	approaching a potentially dangerous situation, slow down,
"Turn left"	a gentle turn away from an object,

"Spin left" sharp turn away,

"Turn right" a gentle turn away from an object,

"Spin right" sharp turn away.

A full list of Sensor Byte possibilities and the action required by the wheelchair in order to minimise the chances of a collision are shown in Appendix D. The Sensor Byte was constructed from consideration of the Sensor Byte mapping which is a list of all possible combinations of sensor array configuration. A Sensor Expert Rule Set was extracted from the mapping and this is shown in Appendix E.

The six bit Sensor Byte was created from the three sensor arrays. Each sensor array had two bits to represent the position or not of an object within the array. The sensor Byte configuration is shown in figure 5.20. The six bit byte contains array bits "Right" the two least significant bits (bits 0,1)and working right to left, "centre" (bits 2,3) and "Left"(bits 4,5). Bits 6 & 7 were spare. The array bits were expressed as zero to three (2 bit binary), the numerical representations being:

0 no detection for this array,

1 detection in "far" array element,

2 detection in "middle" element,

3 detection in "near" element.

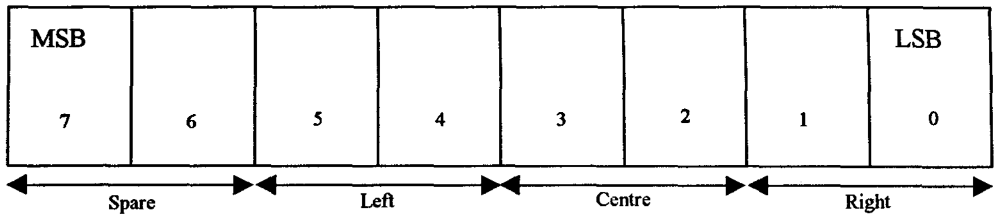


Figure 5.20. The sensor byte configuration.

These numerical operators were used to search the Sensor Byte for object configurations so that the Sensor Expert could recommend an action. The sensor expert algorithms were based on recognition of patterns in the Sensor Byte and are shown in the flow diagram (figure 5.21).

The Sensor Byte classification can be found in Appendix D and may be referred to in order to assist in the understanding of the interpretation algorithms. The two bit numerical operators were examined in isolation from each other and simple algorithms were developed. The algorithms detected numerical patterns in the Sensor byte that indicated a course of action to be recommended. The recommendations and algorithms are listed in table 5.1.

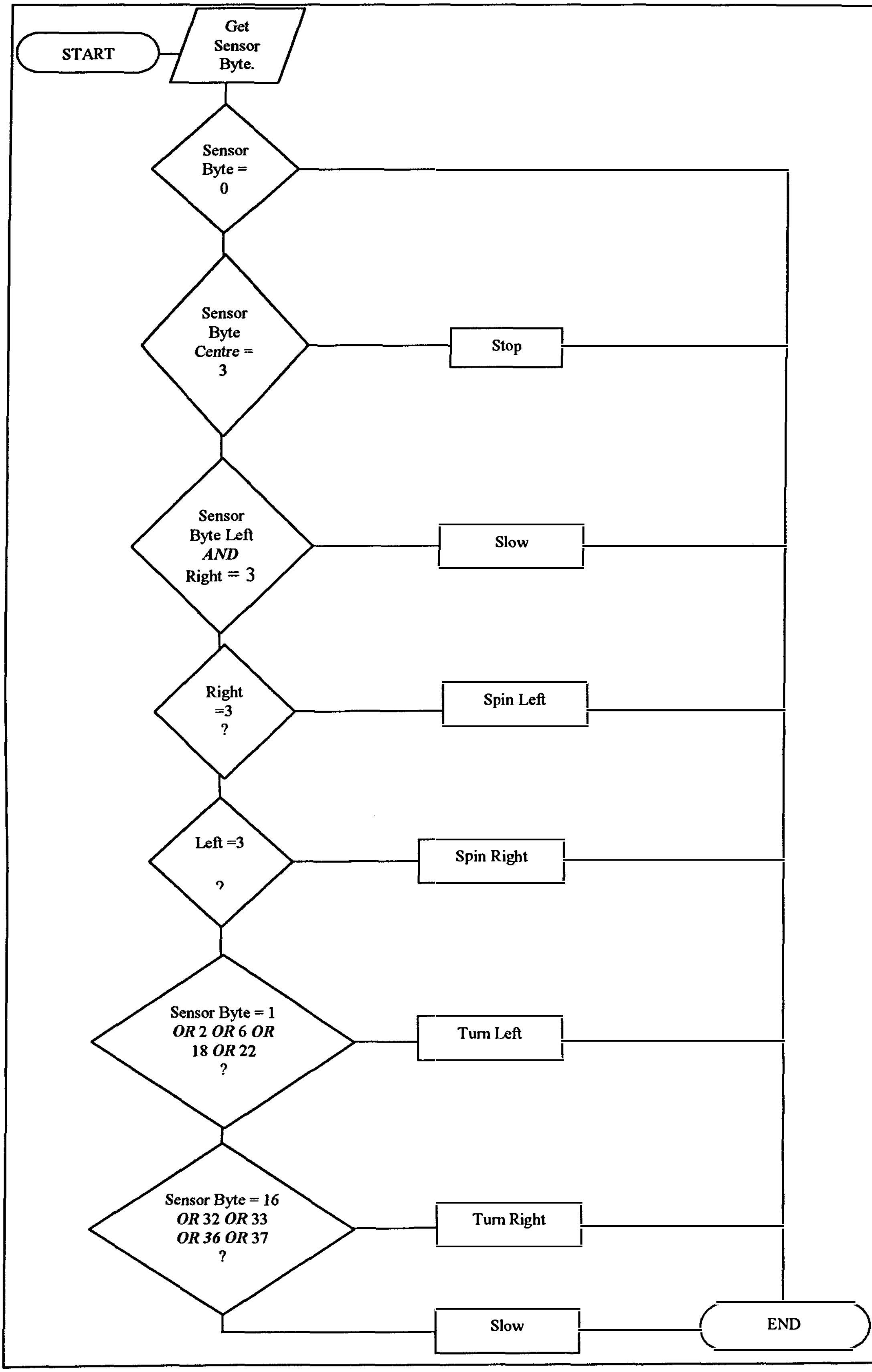


Figure 5.21. The sensor expert algorithm flow diagram.

Test	Action	Remarks
Sensor byte = 0	Nothing	Sensors have not detected anything.
"Centre" = 3	Stop	Object close and in the centre of the wheelchair's path.
"Right" = 3 ***AND*** "Left" ≠ 3	Spin Left	Object close and near the wheelchair's right hand front corner.
"Left = 3 ***AND*** "Right ≠ 3	Spin Right	Object close and near the wheelchair's left hand front corner.
Sensor byte = 1 ***OR*** 2 ***OR*** 6 ***OR*** 18 ***OR*** 22	Turn Left	Object near the wheelchair's right hand side in the wheelchair's path.
Sensor byte = 16 ***OR*** 32 ***OR*** 33 ***OR*** 36 ***OR*** 37	Turn Right	Object near the wheelchair's left hand side in the wheelchair's path.
ELSE	Slow	Wheelchair's path blocked but will not collide yet.

Table 5.1. The sensor expert algorithm.

5.10 Testing the first prototype Expert System.

The first test program was downloaded to the prototype wheelchair-mounted hardware described in Chapter 4. The intelligent powered wheelchair was tested by driving it in an unstructured but uncluttered environment. The response of the wheelchair was slow and inflexible. Parts of the program structure were redundant and were removed for the second prototype program. Figure 5.22 shows the wheelchair during a doorway passage test. The wheelchair has stopped due to phenomena known as "local minima". The distance functions on the wheelchair have prevented the wheelchair from passing through the doorway as the sides of the wheelchair have reached the minimum allowable distance from an object. The distance function algorithms could be adjusted to reduce their effect and

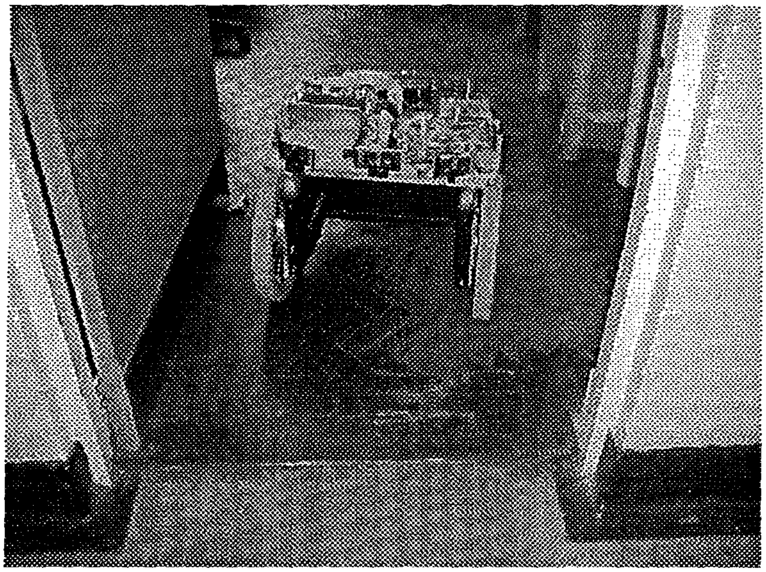


Figure 5.22. The wheelchair attempting an unassisted doorway passage.

allow the wheelchair to move closer to an object. This would have allowed the wheelchair to move through the doorway. However, the response of the wheelchair was sluggish which did not allow it to turn the away from obstructions. Figure 5.23 shows the wheelchair and its path as it attempted to move away from a wall despite a simulated user demanding it to go straight on. The white tape indicates the path the wheelchair had to take to reach that position. The response of the expert system was not sufficient to turn the wheelchair in time and a collision occurred.

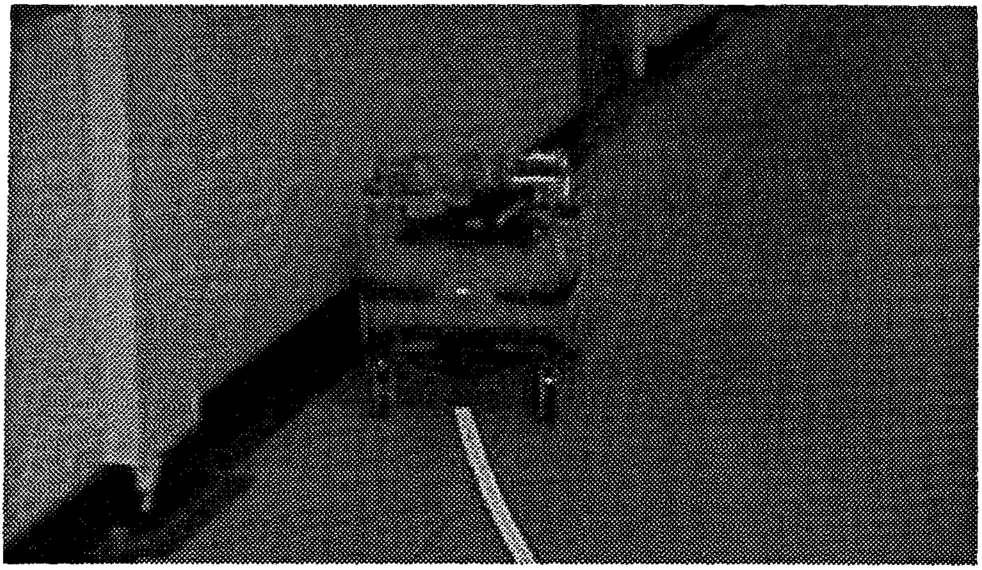


Figure 5.23. The wheelchair during an unsuccessful collision avoidance manouvre.

5.11 The second prototype intelligent wheelchair expert system.

The second prototype expert system was created to simplify some of the processes created for the first prototype. A diagram of the second system is shown in figure 5.24. Code was written in modules and they are described below:

MainCode( ) (C module) was the main program that controlled the program flow and scheduled all major events.

GetRange( ) (C module) was called when the expert system required range data from the sensors. A variable "Transmitter" specified the channel of the sensor to be read. GetRange( ) called module BuildArray( ) to store the range data in an array to be converted into the "sensorbyte" format.

BuildArray( ) (C module) created an array of sensor data gathered by GetRange( ). The

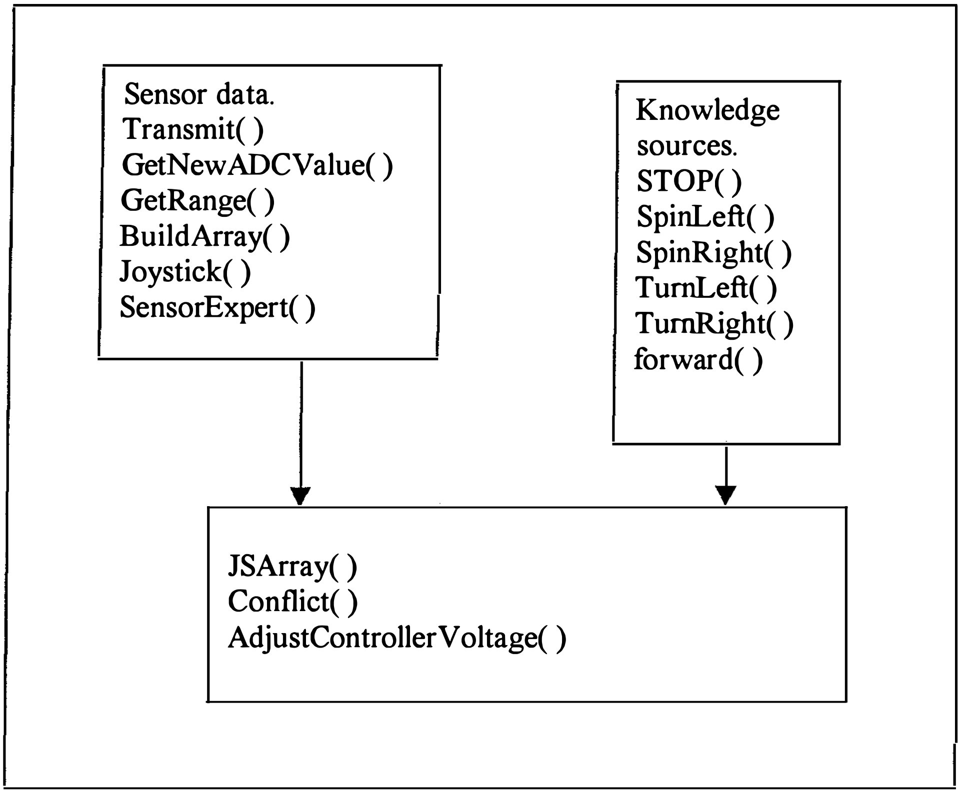


Figure 5.24. The second prototype expert system.

array was an expression of the range data as histograms. The histogram data minimised the effect of noise and random miss-reads. The histogram array was used to create the "sensorbyte" data for the interpretation of the sensor data.

Transmit( ) (assembly code module) fired a single ultrasonic transmitter when active. The transmitter channel was defined as a variable passed with the call statement. The length of the transmitter pulse was regulated by an interrupt set by a timer. The timer was activated at the beginning of the pulse and when it timed out, an interrupt was generated. The interrupt handling routine turned the transmitter off.

Joystick( ) (assembly module) read the status of the joystick outputs and the inputs to the wheelchair controller. On activation, the module used the ADC to read the two controller channels and the two joystick channels.

Checkswitch( ) (assembly module) checked the status of an external toggle switch that was connected to an input port. The switch was used to prevent the main program from looping, the PWM outputs were turned off and the wheelchair reverted to manual control.

Sensorexpert( ) (C module) decoded the sensorbyte value to a course of action for the wheelchair to take. Boolean tests were performed on sensorbyte to interpret the probable best course of action for the wheelchair in order for it to navigate between objects safely. Patterns existed in the sensorbyte data that could be tested for. Once a pattern had been detected, a recommendation was made.

forward( ) (C module) defined control voltages that caused the wheelchair to drive forwards.

Spinleft( ) (C module) defined control voltages that caused the wheelchair to spin left.

Spinright( ) (C module) defined control voltages that caused the wheelchair to spin right.

Turnleft( ) (C module) defined control voltages that caused the wheelchair to turnleft.

Turnright( ) (C module) defined control voltages that caused the wheelchair to turn right.

STOP( ) (C module) defined control voltages that caused the wheelchair to stop.

normal( ) (C module) defined control voltages that caused the wheelchair to drive without interference from the expert system.

AdjustControllerVoltage( ) (C module) adjusted the controller voltages to the required voltage. An algorithm was applied which rapidly adjusted the voltage to the wheelchair controller.

JSArray( ) (C module) created an array from the joystick data. The sampled joystick channels were converted to a Polar form. The Polar data was used to convert the numerical values of joystick position to an expression of joystick position by sector. Six sectors were used to define the position of the joystick; stop, spin left, spin right, turn left, turn right and forward. Back was not considered, as sensors were not mounted onto the rear of the wheelchair.

conflict( ) (C module) considered the sensor recommendation and the joystick position. If the joystick and the sensors agreed on the direction that the wheelchair should be travelling, there was no requirement for any in changes to the trajectory. If there was a conflict between the sensors and the joystick, the trajectory of the wheelchair was modified.

GetADCvalue( ) (assembly module) used to activate the ADC via the SPI interface. The ADC channel was specified as a variable passed to the module. The value of the ADC channel was passed back to the calling function upon termination of the module.

A simplified Blackboard framework was used as the program structure which was similar to the structure of the Hearsay II Blackboard shown in figure 5.25 and described in Section 2.8. As the computing power and program space was limited by the low cost electronic hardware used in this work, certain parts of the Hearsay structure were not included in the simplified Blackboard framework. Figure 5.26 shows the simplified Blackboard framework that was used for this work. The program was working in real-time with a reduced set of knowledge sources and so the Agenda was not required to prioritise a list of

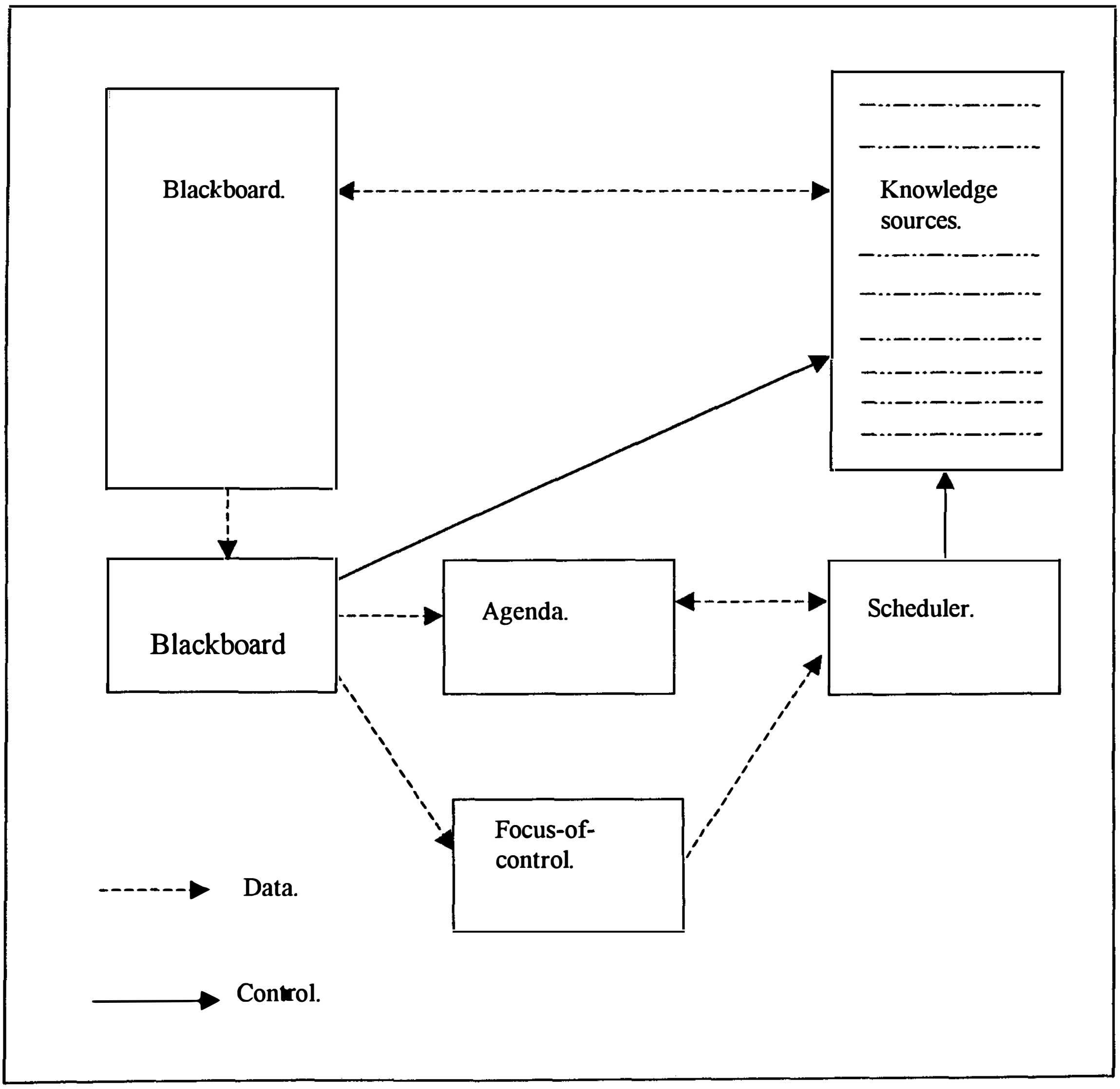


Figure 5.25. The Hearsay II Blackboard structure.

Reproduced from Sanders et al. (2000).

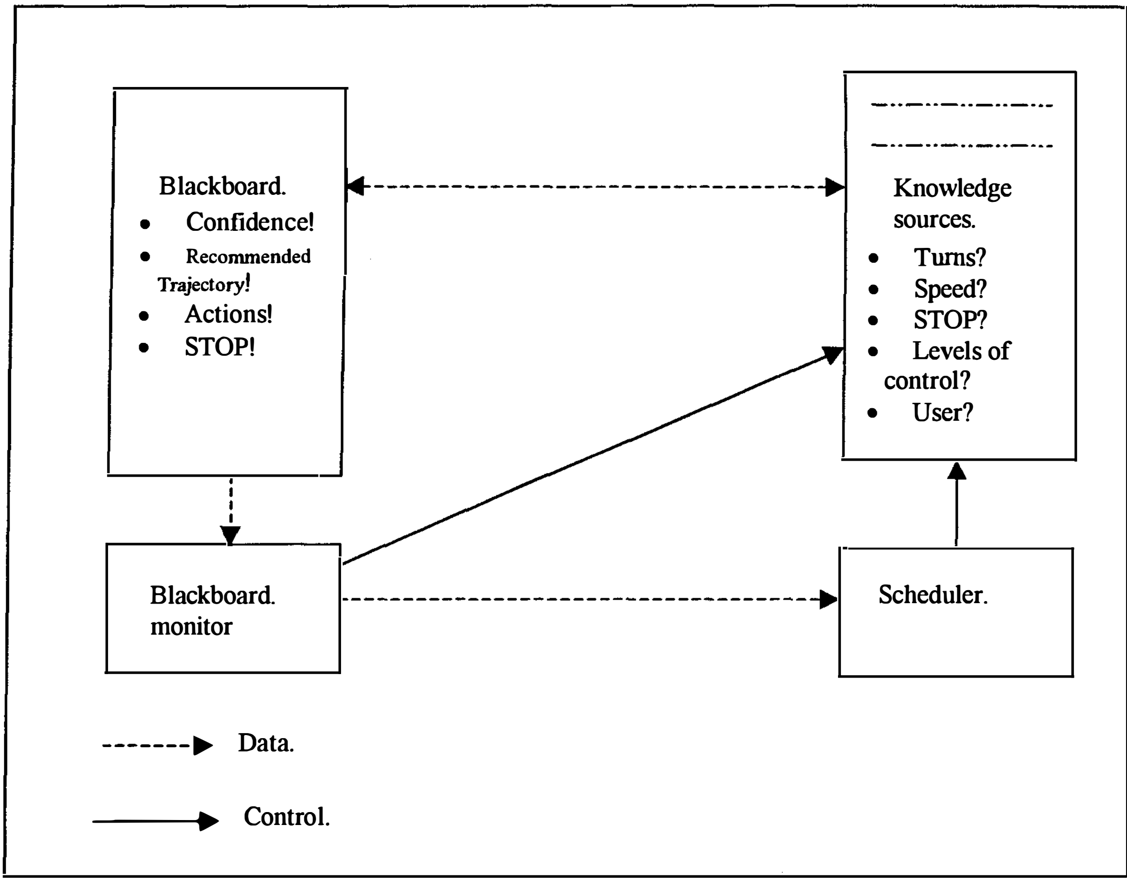


Figure 5.26. The Simplified Blackboard structure.

actions. These were done as part of a polled loop, called from the real-time task under execution. There was a limited set of knowledge sources for this application and hence the Focus-of-control was removed, as it was redundant. The simplified Blackboard framework has a minimal structure to streamline its operation for real-time operation and to minimise the amount of computing power and memory space needed.

The program was easier to control in this structure as all of the main modules communicated with the blackboard (MainCode). The Blackboard monitor checked for changes in the data configuration. Often, no action was needed and control was not passed to any other expert. If the configuration of the data indicated that an action was necessary, control was passed to the knowledge sources for the appropriate course of action to be initiated. The scheduler passed the necessary data to the knowledge sources. Algorithms were simplified to speed up response and predictability. The code including these algorithms is included in Appendix B5.

The Simplified Blackboard Expert System was integrated with the other program modules created during the work described in this Dissertation. The final program structure is shown in Figure 5.27. The user input data through the joystick which was interpreted by the joystick monitor. The joystick monitor passed the interpreted data to the Fuzzy mixer

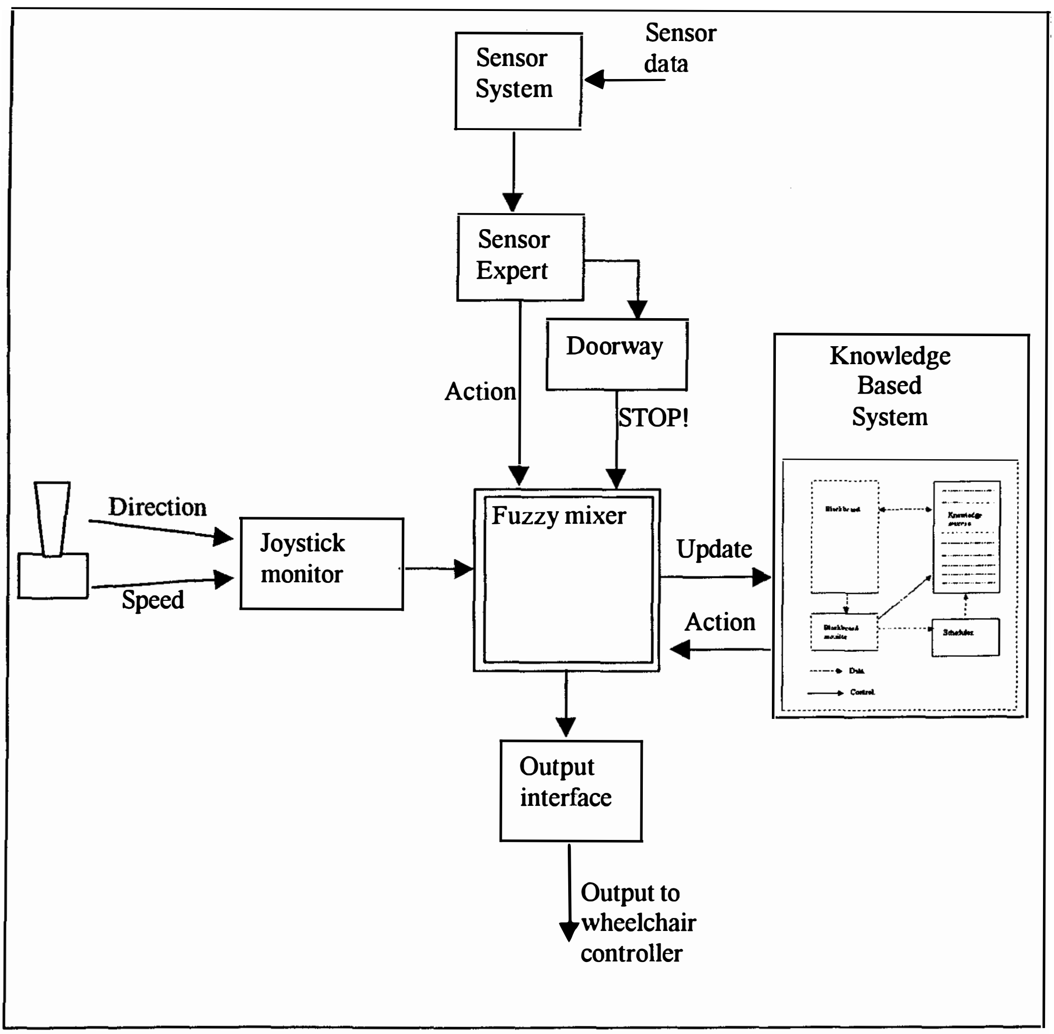


Figure 5.27. The new modules integrated with the Expert System.

as direction, speed and confidence values. Sensor data, acquired by the sensors was read into the sensor expert. The sensor expert filtered the data and applied interpretation algorithms to the data and passed the sensor data to the Fuzzy mixer as a recommendation on how to drive without collision. Sensor data was monitored by the Doorway program which checked for urgent situations where the wheelchair should be stopped immediately as a collision was imminent. The filtered and conditioned data from the Joystick monitor and the Sensor Expert was then referred to the Knowledge Based System. Excluding a "panic" situation where the action would be to stop the wheelchair, the Knowledge Based System made a recommendation on the new trajectory. The new trajectory information was fed back to the Fuzzy mixer where the sensor data was mixed with the user data. The weighting of the data was derived from the confidence values of the sensor and user information. The mixed data was sent to the output interface and on to the wheelchair controller.

Where the confidence in the user was high, the fuzzy mixer tended to ignore the sensor recommendations and allow the wheelchair to be driven mainly under user control. If the sensor information conflicted with the user information, the sensor information was used in proportion to the confidence in the user data. This could vary between the expert system ignoring but continuing to monitor the user data, and the sensors being disregarded and the user having total control.

As an example, in the case where the joystick and the sensor expert were indicating "forward", the expert system set the trajectory as straight-ahead. The sensors were still interrogated to determine the distance that the wheelchair was from the nearest object. The speed of the wheelchair was reduced as the wheelchair became close to an object.

The algorithm used the SpinLeft/SpinRight command to turn the wheelchair. Although the controller voltage settings were set to the spin values, the expert system tended to apply the spin settings for the minimum amount of time required to turn the wheelchair. The wheelchair very rarely performed a "spin" manoeuvre in this mode as the controller settings had returned to a "forward" mode. The application of a spin manoeuvre for a limited time simulated the user moving the joystick completely to one side in order to execute a turn. Users were observed tp use exagerated movements of the joystick, even to performe gentle manoevres.

An example of an algorithm that prevented the wheelchair from driving quickly into an obstruction during a TurnRight manouvre is included in Appendix B6. The code for the TurnLeft case was similar to the TurnRight but the Target values were reversed.

5.12 Testing the second prototype expert system.

The second prototype was downloaded in to the wheelchair-mounted hardware described in Chapter 4. The intelligent wheelchair algorithms were tested by driving the wheelchair in an unstructured but uncluttered environment. The response of the wheelchair was fast enough for the wheelchair to navigate itself along a corridor and align itself with a doorway with the joystick held in the forward position. The path that the wheelchair took indicated that the sensor expert was recommending trajectory changes to the wheelchair controller. Figure 5.28 shows the wheelchair during a doorway passage test. The white tape trailing the wheelchair shows the trajectory of the wheelchair when the joystick was held in the forward position. Figure 5.29 shows the wheelchair and its path as it navigated along a corridor, the wheelchair algorithms were effective in suggesting a path that avoided the walls of the corridor and stayed approximately in the centre of the corridor.

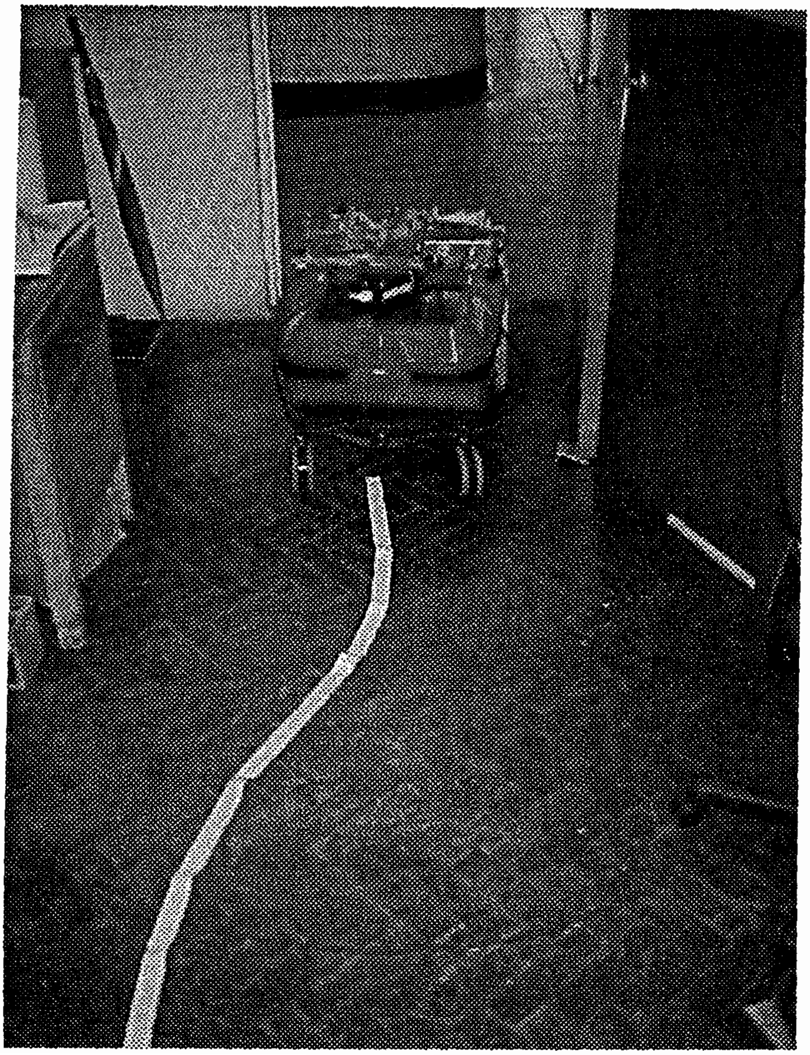


Figure 5.28. The wheelchair negotiating a doorway.

5.13 Conclusion.

The adoption of a simplified Blackboard framework for the program structure simplified the creation of the prototypes. When operating a joystick controlled vehicle, users tend to use large deflections of the joystick to manoeuvre the vehicle. The dynamics of the wheelchair controller and the physical dynamics of the wheelchair make the

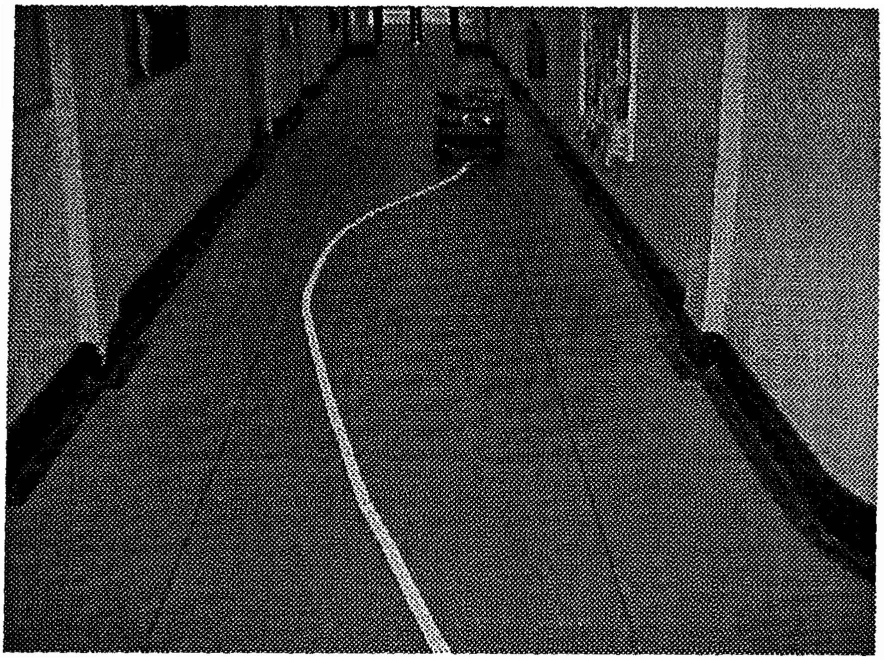


Figure 5.29. The wheelchair navigating along a corridor.

application of large deflections of the joystick a suitable method for accurate control of the wheelchair. Small deflections caused a sluggish reaction from the wheelchair or the inputs were ignored completely by the wheelchair controller. Large changes in the controller input voltages caused smooth changes to be made to the wheelchair trajectory.

The code was written in a mixture of high and low level languages and compiled to a single machine level file. This file was loaded into non-volatile memory in a micro-controller. The use of an integrated programming environment with access to high level editing and de-bugging tools assisted in the creation of the prototypes. A modular structure was adopted to simplify program construction and minimise duplication of code. The structure of the final expert system was similar to a Blackboard type framework. However the similarities were limited by the size of the micro-controller memory of the on-board real time experts which ruled out the creation of complicated data structures and large amounts of program code.

Chapter 6 describes the results of the tests of the expert algorithms and other program modules.

Chapter 6.

RESULTS.

Chapter 4 described the creation of the new circuit boards, interfaces other hardware and Chapter 5 the creation of the novel algorithms and software. This Chapter describes the results of testing the hardware and software. The parts for the intelligent wheelchair were built separately and tested in isolation for their functionality before being joined together. Tests were performed during construction of the prototype to ensure that conflicts were not created when the components were joined.

Intermediate tests were made during the construction of the prototype wheelchair. Final tests were made with human operators once the intelligent wheelchair had been constructed. The results of the tests are presented. Section 6.1 considers the hardware, 6.2 considers the Virtual Reality work and 6.3 considers the expert system and controller algorithms.

6.1 The Hardware.

The Bobcat II wheelchair had been used as a research vehicle for other research projects by the Author and other researchers. It was returned to its original form for this work. The hardware described in previous Chapters was then fitted to the wheelchair. The work was carried out in the following order:

- A prototyping printed circuit board (PCB) was created so that a Philips 87C752 micro-controller could be programmed, tested and connected to prototype circuit boards. The Philips micro-controller prototype PCB is shown in Figure F1 of Appendix F.

- Another prototype board was created for the ATMEL 90AT8515 microcontroller. The PCB is shown in Figure F2 of Appendix F.

- Transmitter and receiver boards were built. These are shown in Figure F3 of Appendix F.

- A prototyping PCB for the MAX1112 analogue to digital converter was created. Shown in Figure F4 of Appendix F. The ultrasonic sensors were connected to the micro-controller circuits and the sensor data was read and recorded.

- An interface was created to connect the microcontroller pulse width modulation (PWM) circuits to the wheelchair interface. The PWM interface is shown in Figure F5 of Appendix F.

- Micro-controller boards were connected to read the joystick and provide feedback to the PWM interface.

- An interface that had been used for other work was modified and used as a hardware mixer for data from the micro-controller and the joystick. The modified interface is shown in Figure F6 of Appendix F.

At all stages tests were made, the results recorded and relevant results are presented in the following Sections.

6.1.1 An analysis of the joystick.

The joystick was a sealed unit. To accurately simulate the joystick for other research, Stott (1997) had already mapped the electrical properties of the joystick. By analyzing the electrical response to movements of the joystick an equivalent circuit was obtained. The equivalent joystick circuit diagram is shown in Figure 6.1. The Joystick consisted of two high reference voltage lines, a low reference line and two wipers. The values of the internal resistors were found to be critical as they were checked by the Penny and Giles PG8 controller as part of the safety procedure during startup. It was important to maintain the correct reference voltages too and the wiper voltage could never be allowed to move higher than the high-reference or lower than the low-reference voltages. The consequence for allowing any parameters to fall

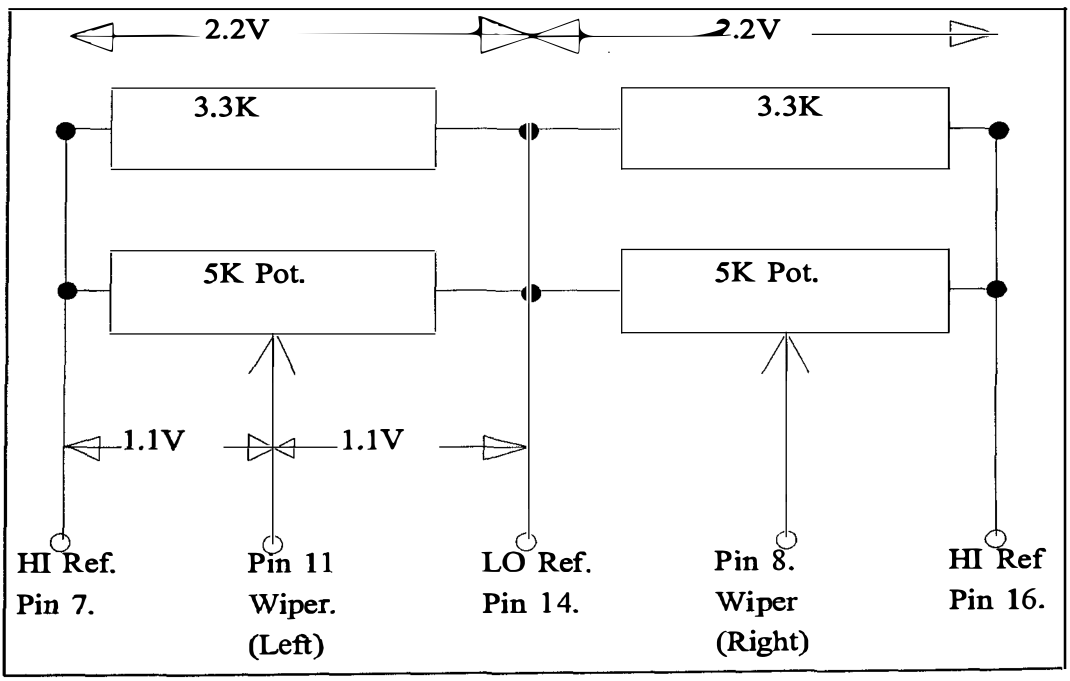


Figure 6.1. The equivalent circuit of the joystick.
Reproduced from Stott (1997).

outside the specified values was that the controller would either fail to switch on or would trip-out if the fault occurred during operation of the intelligent wheelchair.

Position of joystick.	Voltage from joystick wiper. (V)	Integer returned from ADC.
Full speed forwards.	3.50	194
Slow (forwards). (Wheels just begin to turn)	2.63	146
Stop.	2.45	136
Slow (backwards). (Wheels just begin to turn)	2.20	122
Full speed backward.	1.50	83

Table 6.1: ADC data read in relation to joystick position.

The differential voltage between the low and high references was found to be 2.2V. The stop position for the wheelchair was the midpoint or 1.1V between the joystick wipers and the references. The values were not precise and a tolerance of ±0.11V existed around the mean stop position. The wiper voltages relative to the references are shown in Table 6.1. The table shows the voltage values for one joystick wiper. The response of both wipers was similar and Stott (1997) reported that the data in the table was valid for either the left or the right hand wiper.

6.1.2 Reading the joystick by the computer.

During the novel work described in this Dissertation the joystick was connected to the controller interface as described in Chapter 4. A MAX1112 ADC was used to sample the joystick voltages. The sampled signal was converted to an integer and stored. The relationship between the joystick voltage and the value of the integer generated by the ADC is shown in Figure 6.2. The ADC was also used to sample the ultrasonic sensors and the voltage that was finally input to the controller after the joystick signal had been modified. The data is shown in tabular form in Table 6.2.

6.1.3 Data to control the wheelchair motors using the computer.

The PWM interface was used to control the motors of the wheelchair by sending the appropriate control signals to the wheelchair controller as discussed in Chapter 4. The PWM outputs of the microcontroller converted an integer held in an internal register to

Joystick position	Wiper Voltage Right channel (pin 8), ADC values in brackets.	Wiper Voltage Left channel (pin 11), ADC values in brackets.
Full speed (forwards)	3.50 (194)	3.50 (194)
Turn left	3.74 (205)	2.63 (146)
Spin left	3.50 (194)	1.70 (94)
Turn right	2.63 (146)	3.74 (205)
Spin right	1.70 (94)	3.50 (194)
Stop	2.39 – 2.51 (132 – 139)	2.39 – 2.51 (132 – 139)

Table 6.2: Joystick wiper voltages relative to the joystick position.

a pulse-width modulated DC voltage. The PWM interface converted the PWM voltage to a smoothed DC voltage proportional to the PWM value. The output from the PWM interface could be "level-shifted" to alter the null value of the output. The null value or 0V output allowed the joystick to control the wheelchair without interference from the PWM interface circuits. The PWM interface could be set up so that the corresponding

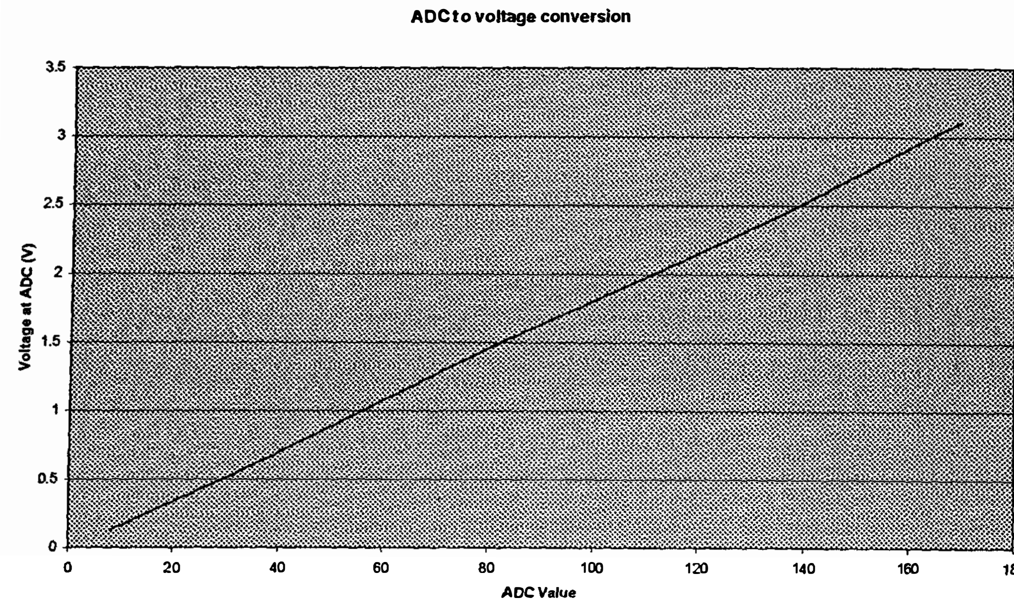


Figure 6.2. A graph of ADC values returned relative to voltage measured.

output voltage range was manually adjustable to suit the tests underway. During tests, the PWM output was set up so that a PWM register value of zero did not cause modification of the joystick signal. Figure 6.3 shows the output of the PWM interface to a range of micro-controller inputs.

It was important for the output voltage to remain within the range 1.4V – 3.6V, as these were the maximum and minimum voltages expected from the joystick. Should the voltage to the controller differ from the expected range the controller assumed that a fault had occurred and shut down. The wheelchair would perform an emergency stop and would not operate again until the joystick wiper voltages were returned to the stop values and the controller was manually reset by the operator.

6.1.4 Testing the sensors.

The ultrasonic sensors are described in detail in Chapter 4. The ultrasonic range finder could detect large objects with a surface that was a good reflector of ultrasonic energy.

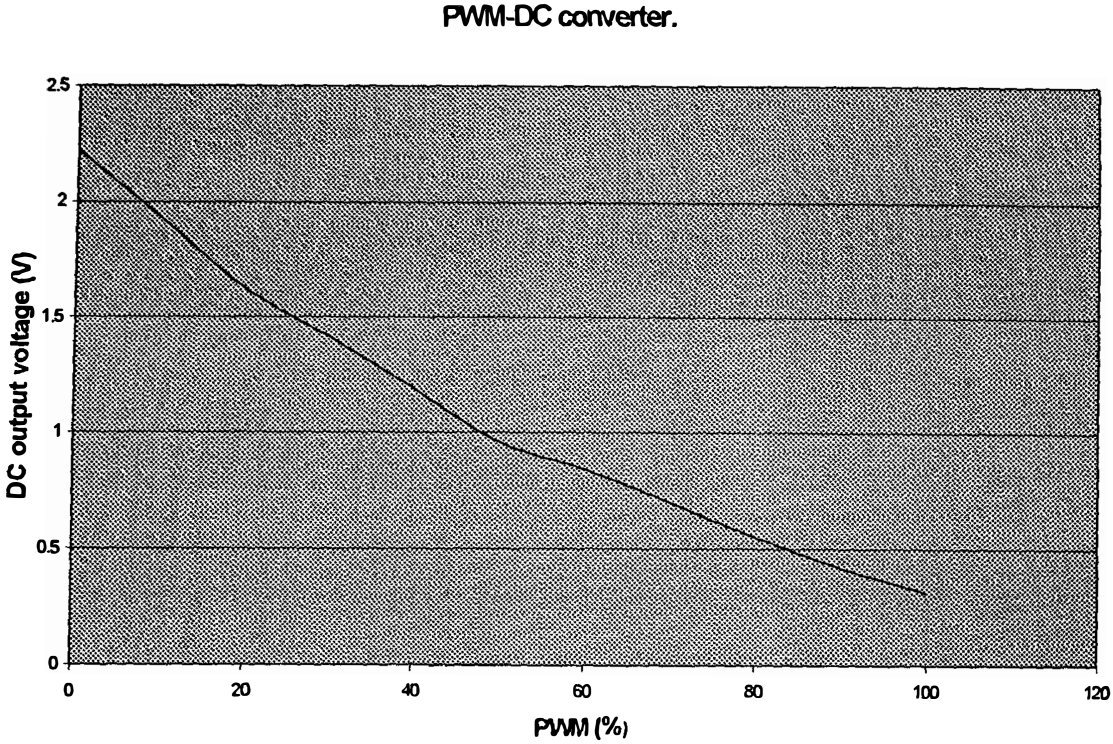


Figure 6.3. The output of the new PWM interface.

Generally, glass, metal, cardboard and wood were all good reflectors. Poor reflectors included targets with small surface areas such as chair and table legs and soft materials such as clothing and fabric covered cushions. An ideal target presented a hard smooth surface with a large area at 90^0 to the sensor axis. Poor reflecting targets were generally detected when the distance from the sensor to the target was small. Very small targets such as a single chair leg (15mm diameter steel tube) were sometimes not detected due to their inability to reflect sufficient energy to the receiver.

Sensor range tests. Time of flight sensors of this type tend to be subject to a minimum range. In this case this was due to the length of the pulse of ultrasonic energy emitted by the transmitter. Transmitter and receiver circuits were constructed and tested. The transmitter and receiver were mounted side by side so that the receiver would detect a transmitted pulse almost immediately. The micro-controller did not start to read the receiver until the transmitter had finished transmitting the pulse. This prevented the receiver detecting the pulse before it had been fully transmitted.

Range was determined by sampling the receiver signal after a pulse of ultrasonic energy had been fired towards a target. A software algorithm detected a returned pulse echo by analysis of the sampled data. The number of samples taken before a detection was made related to the range of the target. Some example results from initial range tests are shown in Table 6.3. In the test represented in the table, three sensor pairs were tested by firing them in turn at a target that was progressively moved away. The target consisted of a 1m by 0.5m flat board held at right angles to the sensor axis. The tabulated results showed that the sensors were capable of detecting increments of approximately 66mm. This was an acceptable resolution for this application. It would have been possible to increase the resolution by increasing the sampling rate, but that would have caused an increased workload for the microcontroller and no significant advantages.

Distance (mm)	Sensor 1	Sensor2	Sensor 3
100	2	1	1
200	3	3	2
300	4	4	3
400	6	6	5
500	7	7	6
600	7	8	8
700	9	9	9
800	11	11	11
900	12	12	12
1000	14	14	14
1200	17	17	17
1400	20	20	20
1600	23	23	23
1800	26	26	26
2000	30	29	30
2200	32	33	32
2400	34	35	34
2600	37	38	37
2800	40	40	40

Table 6.3. The results of the range tests.

Triangulation tests. The sensor pairs could be separated by up to 500mm as they were mounted along the front of the wheelchair. This made it possible to triangulate between the transmitter of one transducer pair and the receiver of another. This was tested in a similar manner to the range tests. The transducer pairs were mounted 250mm apart and a 110mm plastic cylinder of length 1m was placed in front of the sensors. The cylinder was typically moved on a grid pattern to known positions and the actual distance to the sensor transducers was measured. A 100μs pulse was fired from the transmitter. The microcontroller calculated the distance to the object. The test was carefully conducted so that the object was always the nearest and best reflector to the sensor transducers. The actual (physically measured) and sensor derived (calculated from sensor data) data are shown in Table 6.4. Figure 6.4 shows the same data plotted as a scale plan view. The crosses on Figure 6.4 represent the position of the target as measured by the sensors. The target was moved in 100mm increments across and away from the sensors in a grid pattern with the exception of tests. As the range increased, the reliability of the sensors was reduced and the target became difficult to detect.

Test positions1, 2,5,6,7,8,11 and 12 were successfully detected using time of flight and triangulation algorithms. For the time of flight tests, the pulse was transmitted from transmitter at "4" and received by the receiver at the same position. The distance of the target from "4" could be calculated from this information. Where triangulation was used, the pulse was transmitted from position "4" but received with the receiver at position "1". With the time of flight and the triangulation, the approximate position of the target could be estimated.

	Sensor 1	Sensor 4	Sensor 1	Sensor 4
	Sensor distance (mm)	Sensor distance (mm)	Actual distance (mm)	Actual distance (mm)
1	132	264	140	290
2	198	198	150	200
3	-	198	250	150
4	-	264	300	250
5	264	330	250	300
6	330	396	280	420
7	396	462	330	470
8	396	462	350	400
9	-	396	400	360
10	-	528	480	450
11	528	528	450	490
12	528	594	470	570

Table 6.4. The results of the triangulation tests.

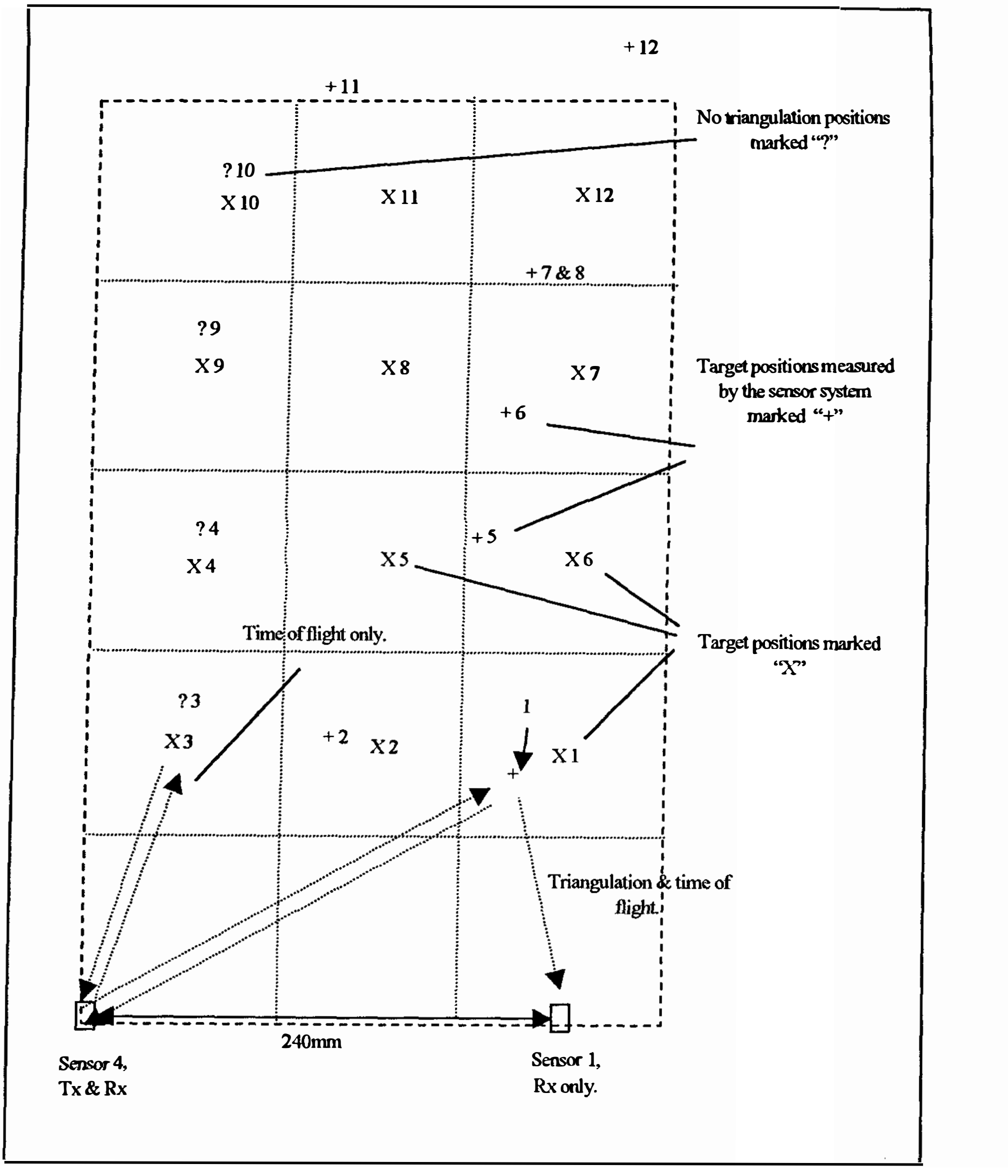


Figure 6.4. Schematic representation of a triangulation test.

Tests 3, 4, 9 and 10 did not successfully use triangulation. Time of flight sensing worked well with these tests as the target was almost directly in line with the sensors. However the angular sensitivity of the receiver at position "1" was not great enough for the triangulation method to operate successfully for these tests.

It can be seen from Figure 6.4 that the triangulation algorithms were more accurate at lower ranges. As the target range increased the error in the target position increased.

The width of the target was 110mm and although it was presenting a rounded surface to the sensors, it could not be assumed that the reflected ultrasonic energy was using the shortest path to the receiver. Also, the resolution was 66mm, and not designed to accurately pinpoint a target. The sensors were required to indicate the presence of an object if present, within a known volume. By increasing the pulse length it was possible to increase the range. For this test however, it was sufficient to use shorter ranges. A series of tests were conducted in the laboratory to test the sensors in a configuration that was more typical for ultrasonic range finders. An example is the use of the 110mm diameter plastic cylinder (height 1m) as a target for the sensors. The tests were described in Chapter 4. For tests detailed here the paired receiver and transmitter were horizontally separated by 40mm. The micro-controller code for the tests is shown in Appendix C2. An example of data from one of the range tests is shown in Figure 6.5. The data was converted to a gradient plot for the detection algorithms. The plot shown in Figure 6.5 was converted to a gradient plot and is shown in Figure 6.6. The voltage plot shown in Figure 6.5 was from a "good" reflector and the returned pulse of ultrasonic energy is prominent. If the returned pulse was of minimal power or was swamped by noise, the gradient plot was easier to decode. It was discovered by experimentation that the gradient of the returned pulse was generally steep. The random noise however was of lower gradient and could be ignored. An advantage of the gradient detection algorithms was that the signal was easier to decode with simple algorithms.

That made it suitable for the application described in this Dissertation, which was implemented on a micro-controller.

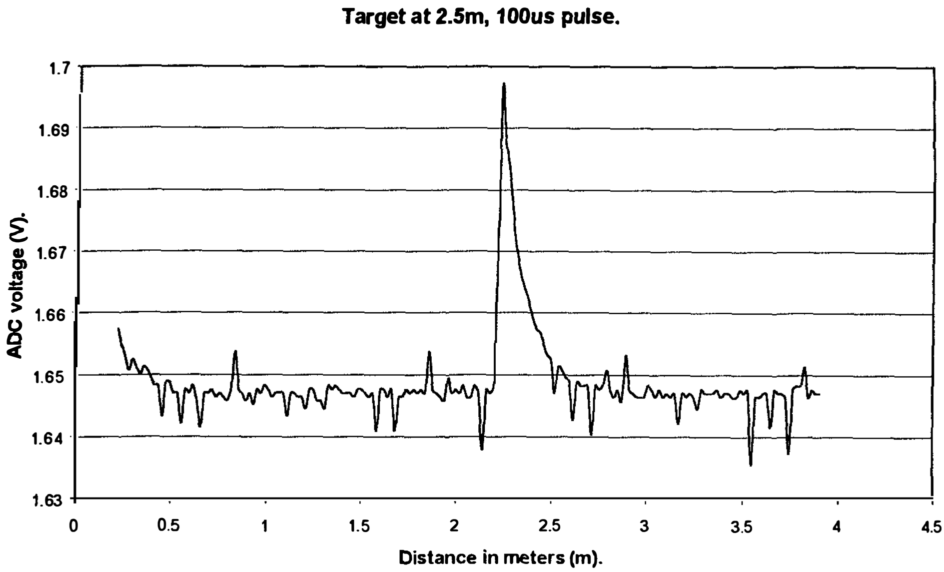


Figure 6.5. A typical voltage plot from a range test.

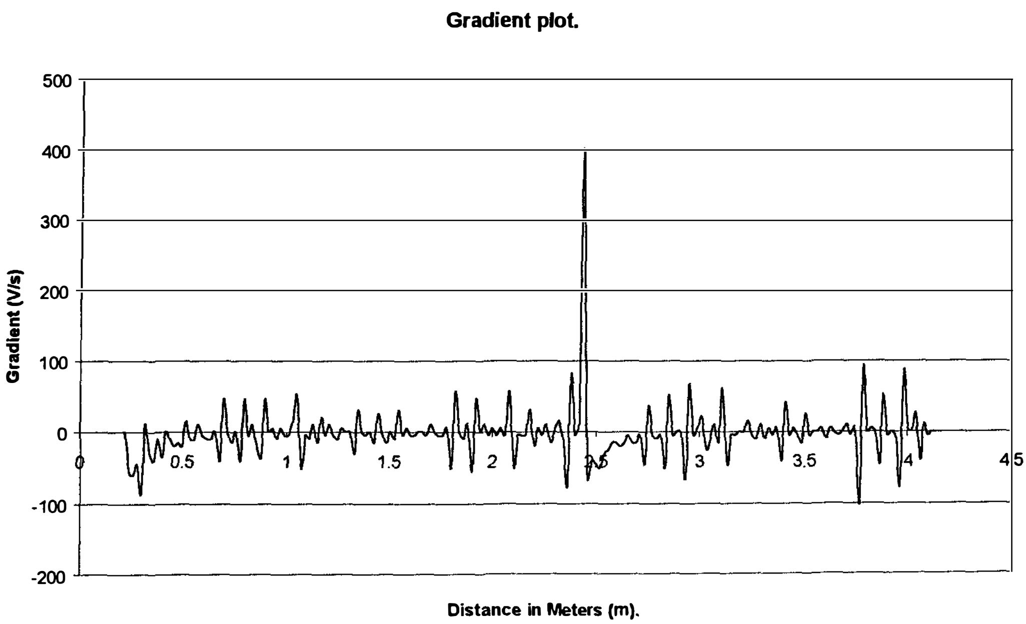


Figure 6.6. The voltage plot shown in Figure 6.5 converted to a gradient plot.

A target was placed in front of the transmitter and receiver pairs so that the ultrasonic energy was reflected back to the receiver. Ultrasonic sensors of this type typically had a beam divergence of 15^0- 30^0. A test was devised to establish the beam pattern and divergence angle of the sensor transducers used in this work. Schematically a cone shape could have represented the beam. In reality the beam was not symmetrical or regular. Manufacturers data for the transmitter used is shown at Figure 6.7. The directional radiation pattern showed a regular "balloon" shape in front of the transmitter and irregular "lobes" of reduced power at the sides. An interesting characteristic was low attenuation up to 15^0 each side of the axis (the power being measured at the same radial distance from the transmitter). At an angle of 30^0 the power had attenuated by 10 dB and at 46^0 by 20 dB. It was noted that the angle of divergence for the sensors could be minimized by a reduction in sensitivity. Also the use of short duration pulses tended to narrow the beam and reduce the effect of the lobes due to the larger number of harmonic frequencies which cause destructive interference at the edges of the beam. This is discussed in Chapter 4. Low sensitivity would only register a relatively powerful detection which would probably be a reflection from a target within the 0dB-attenuation

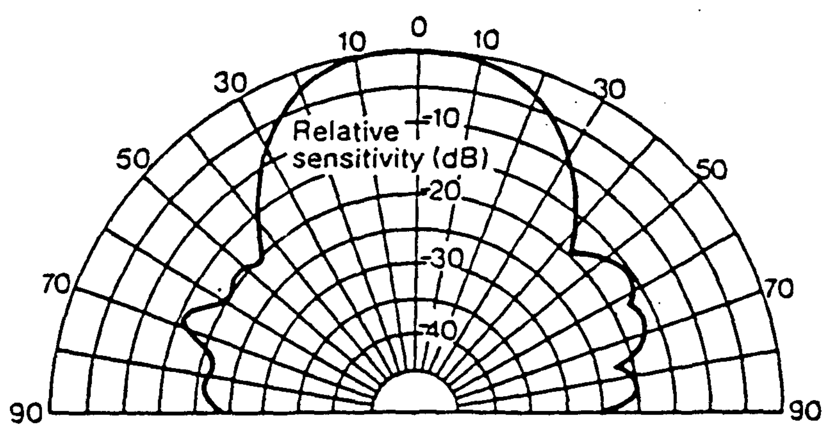


Figure 6.7. Manufacturers radiation plot for the ultrasonic sensor transducers.

band (± 15^0 from axis). A disadvantage of low sensitivity would have been the failure of the system to detect poor reflections, even on the axis of the sensor system. Reliable detections of all objects within the known volume swept by the sensors were necessary to allow the expert systems to safely guide a user. User confidence would be reduced for an intelligent wheelchair that did not reliably detect objects in the path of the wheelchair.

The sensors used for this work tended to use the low attenuation part of the energy envelope. To understand more about the characteristics of the sensors, tests were carried out in an anechoic chamber. The chamber was lined with sound absorbent foam cones to reduce the effect of reflections and ambient noise interfering with the tests. An ultrasonic transmitter was positioned inside the chamber and used to transmit pulses of known duration. An ultrasonic receiver was placed in front of the transmitter and used to record the power of the ultrasonic pulse at known positions in front and to the side of the transmitter. The results were recorded and a crude beam pattern was produced from the results. This was discussed in section 4.4.2.

6.1.5 Sensor strategies.

Several sensor configurations were tested to. These included three sensors mounted on the front of the wheelchair facing forward and four sensors mounted on the front of the wheelchair in a configuration to test some triangulation algorithms. The configurations are shown in Chapter 4 Figures 4.17 and 4.19. The number of samples that could be acquired by the MAX1112 ADC was limited and that prevented cascaded sensor readings from being taken. Sensors could be activated and read sequentially, however, it was possible to use novel sensor reading strategies to try to maximise the effectiveness of a sensor array.

6.1.6 Initial configuration consisting of four sensors.

Originally it was intended to activate one transmitter and sample several receiver channels in sequence. The Analogue to Digital Converter (ADC) could only sample one channel at a time and so in order to approximate to simultaneous sampling, the channels had to be rapidly sampled in a sequence. If more than one channel could be read simultaneously, range finding using both time-of-flight and triangulation could be used. Figure 6.8 shows the configuration of the four-sensor array tested on the wheelchair. The sensors were positioned so that the beams overlapped and this made the configuration suitable for triangulation. Table 6.5 shows Zones of the four-sensor array and the sensors that swept them. Table 6.6 shows possible sensor operations to examine all the Zones with the sensor array. It can be seen from Table 6.6 that a transmitter could be fired and data obtained from more than one receiver.

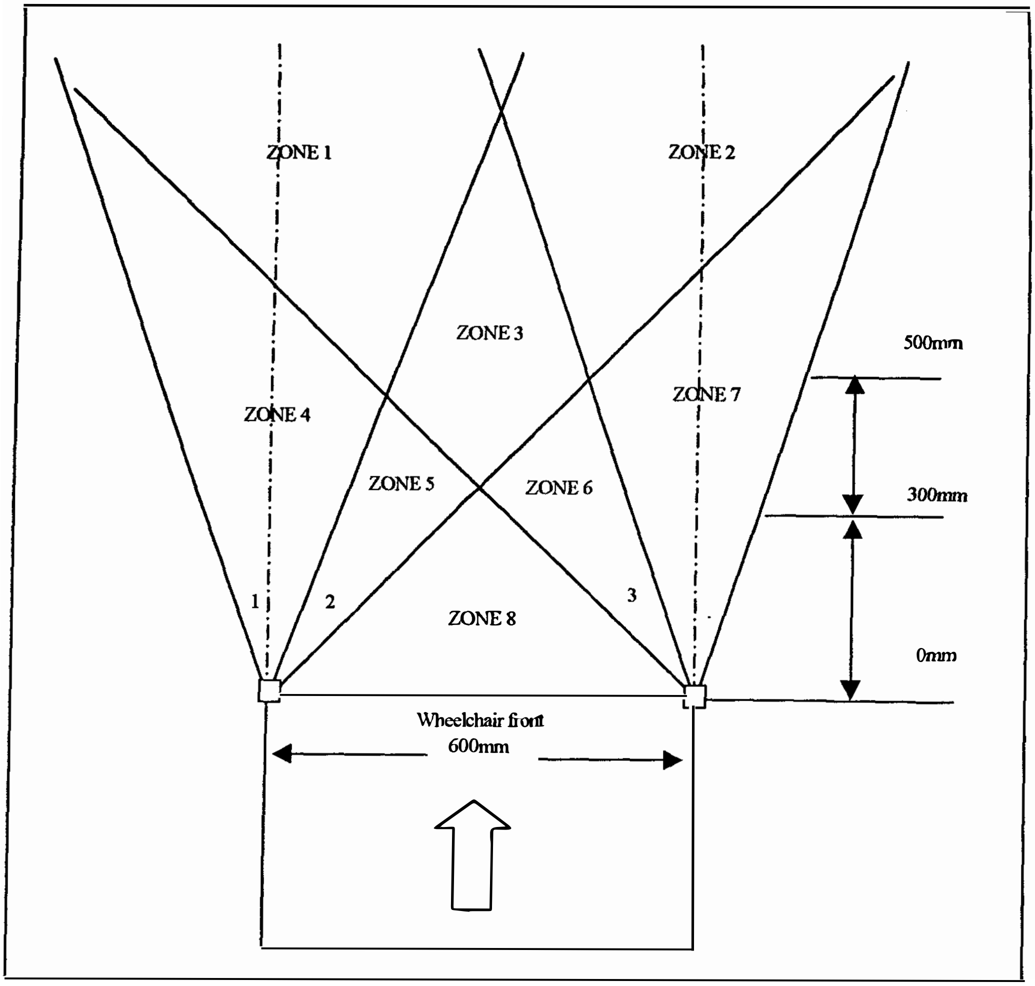


Figure 6.8. The four-sensor array configuration.

Triangulation and ranging algorithms were used with the data from the four-sensor array. The data from different receivers were mixed together to reinforce the belief that an object was or was not present in the corresponding zone. Zones such as Zone 3 in Figure 6.8 were swept by more than one sensor and so the certainty value that an object existed in this Zone was increased faster than a Zone such as Zone 4 that was swept by only one sensor. Zone 8 was close to the wheelchair front and not swept by any sensors. This area was a dead zone to the array and it was noted that this was a vulnerable area.

Zone	Triangulation between sensors	Time of flight sensor(s)
1	1 and 3	1 and 3
2	2 and 4	2 and 4
3	2 and 3	2 and 3
4	none	1
5	none	2
6	none	3
7	none	4
8	none	none

Table 6.5. The sensor coverage of the Zones shown in Figure 6.8.

Fire sensor	Sample receivers	Target zones
1	1 and 3	1 and 4
2	2,3 and 4	2,3 and 5
3	1,2 and 3	1,3 and 6
4	2 and 4	2 and 7

Table 6.6. Sensor strategies for triangulation and range finding operations with the four-sensor array.

The Zones shown in Figure 6.8 were stored in the micro-controller as simplified histograms as discussed in Chapter 5. As each of the Zones were swept by sensors a different number of times, the histogrammic weightings were altered to reflect the frequency of the Zone update. This prevented some Zones from ramping up or down more rapidly than others. The algorithms used for the 4-sensor array are shown below.

Range finders:

Sensor 1. Checked Zones 1 & 4.

Sensor 2. Checked Zones 2,3 & 5.

Sensor 3. Checked Zones 1,3 & 6.

Sensor 4. Checked Zones 2 & 7.

Triangulation:

Sensor 2 triangulated with sensor 3.

Sensor 3 triangulated with sensor 2.

All Zones that were occupied were incremented by 3.

All Zones were decremented (-1) to decay array.

Decrement Zones 1, 2 & 3, to decay middle Zones quicker.

Return all out of range (>15 or <0) Zones to tolerance.

The algorithm's "Zones" were mapped to a certainty grid or array. The algorithm increased the value of any array values that corresponded to an object detected in a Zone. The array positions that were swept by two sensors ramped up at twice the speed as other array positions. These positions were decayed at a faster rate to compensate.

Testing the four-sensor array. A grid of 200mm squares was drawn on the floor of the laboratory. The transmitter and receiver were placed at the intersection of two grid lines in a suitable position and this position was marked 0. All measurements were taken from position 0. A 110mm diameter plastic cylinder, 1m in length was used for the target. The target was placed vertically in front of the sensors with the surface normal to the sensor axis at all times during the test. Figure 6.9 shows a test being conducted. The centerline of the cylinder was placed over the intersection of the grid lines. The target was moved around in front of the sensors from one grid intersection to the next until all intersections had been tested. A schematic of the tests is shown in Figure 6.10. The test was conducted with 100mm squares up to 500mm from the sensors and 300mm squares from 300mm to 3m from the sensors. The Figure shows the coverage of the sensor array by shaded squares marked with an X. Squares where there was no detection were left blank. The squares up to 100mm from the sensor array were within the minimum sensing range of the system and were too close to the sensors

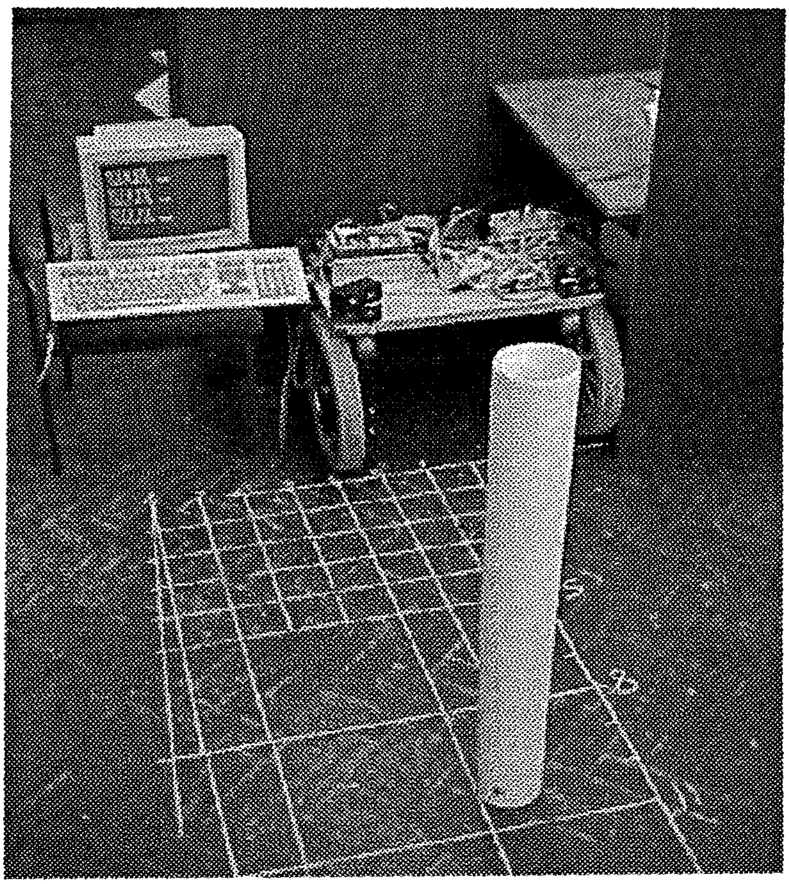


Figure 6.9. A four-sensor array test.

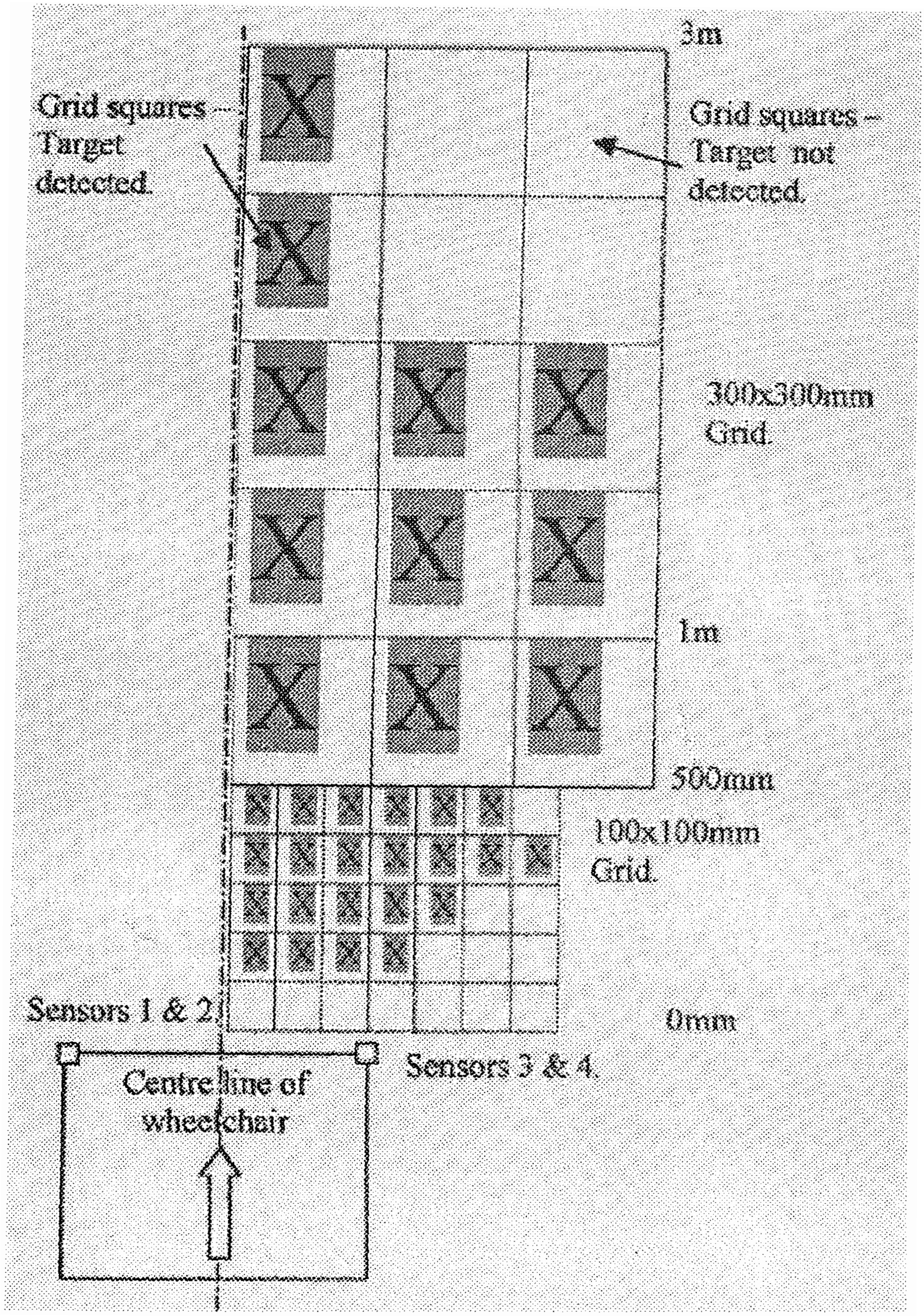


Figure 6.10. A schematic of the four sensor test.

to be tested. The sensor transducers were mounted to ensure that this area was behind the front edge of the wheelchair's driving wheels. The coverage of the sensor system demonstrated a classical balloon shaped area and this represented the coverage of the four-sensor array. The maximum range of all sensor pairs in the array was limited by software to speed up the program cycle time and because greater ranges were not required on an intelligent wheelchair designed to operate within a dwelling or other "cluttered" building.

6.1.7 Later configuration consisting of three sensors.

Chapter 4 described the orientation of the three-sensor array for the intelligent wheelchair. Figure 4.11 showed how the sensors were mounted on the wheelchair. The three-sensor configuration allowed simpler software design than the four-sensor configuration as the amount of data from the array was lower and during some tests, there were no triangulation operations to perform. Figure 6.11 shows the configuration of the three-sensor array. The sensors were positioned so that the beams overlapped, which made this configuration suitable for triangulation. Table 6.7 shows the transmitting and receiving strategies used during some tests. It can be seen from Table 6.8 that a transmitter could be fired and data obtained from more than one receiver. The data from different receivers were mixed together to reinforce the belief that an object was present in the corresponding zone. Similar algorithms to the four sensor configuration algorithms were used to enhance or reduce the value of individual zones. The algorithms used for these tests are shown below:

Range finders:

Sensor 1. Checked Zone 3.

Sensor 2. Checked Zones 4.

Sensor 3. Checked Zones 5.

Triangulation:

Sensor 1triangulated with sensor 2.

Sensor 2 triangulated with sensor 1 & 3.

Sensor 3 triangulated with sensor 2.

All Zones that were occupied were incremented by 3.

All Zones were decremented (-1) to decay array.

Decrement Zones 1, & 2 to decay middle Zones quicker.

Return all out of range (>15 or <0) Zones to tolerance.

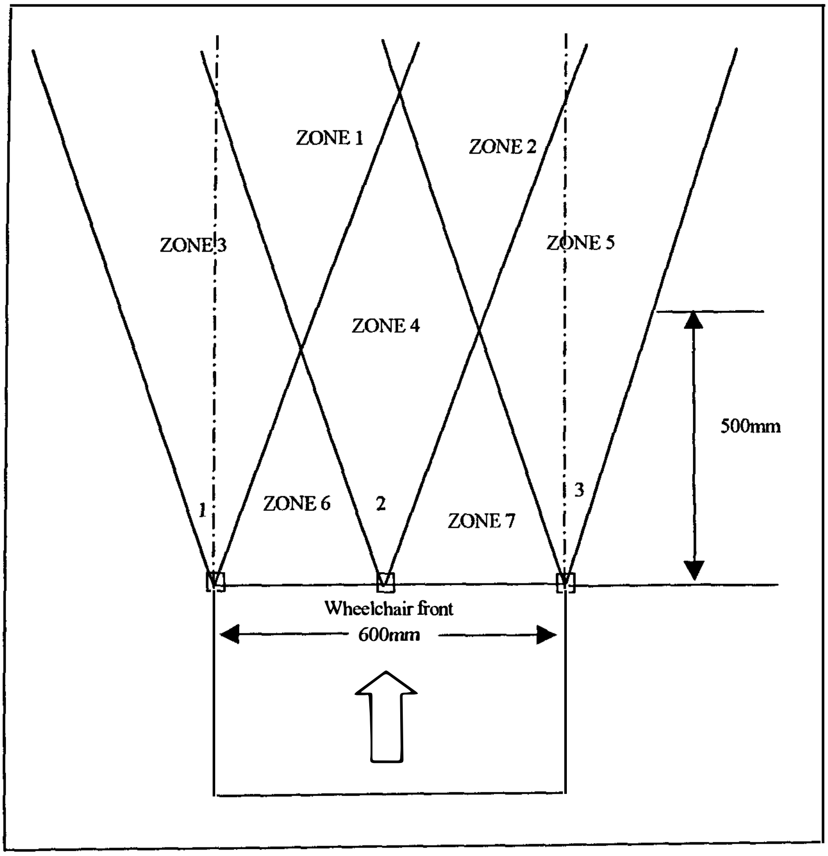


Figure 6.11. The three-sensor array configuration.

As was found when investigating the four-sensor array, it was not possible to simultaneously sample more than one ADC channel. Interrupt over-run occurred and

Zone	Triangulation between sensors	Time of flight sensor(s)
1	1 and 2	1 and2
2	2 and 3	2 and 3
3	none	1
4	none	2
5	none	3
6	none	none
7	none	none

Table 6.7. Experimental transmitting and receiving strategies.

the microcontroller crashed as the serial link between the ADC and the micro-controller was not able to transfer data at the speed required for multi-sensor sampling. In order to perform triangulation between sensors, it was necessary to perform each test separately. Acquiring sensor data was unacceptably slow. The triangulation operations were abandoned at this stage, as they did not add sufficiently to the accuracy or effectiveness of the sensors. The three-sensor array was used in the time of flight range finder mode with the three sensor pairs acting independently. The simple three-sensor configuration is shown in Figure 6.12. Beam patterns have been overlaid to indicate the detection areas of each of the sensors. A 3x3-certainty grid has also been included on the Figure. The simple three-sensor configuration was found to be sufficient.

A comparison of the sensor configurations. In the case of the four-sensor array, a complete sensor sweep, which included all possible transmitters to receiver combinations, required a total of ten transmitter pulses. Transmitters 1 and 3 would fire twice each and transmitters 2 and 4 would fire three times each. In the case of the three-

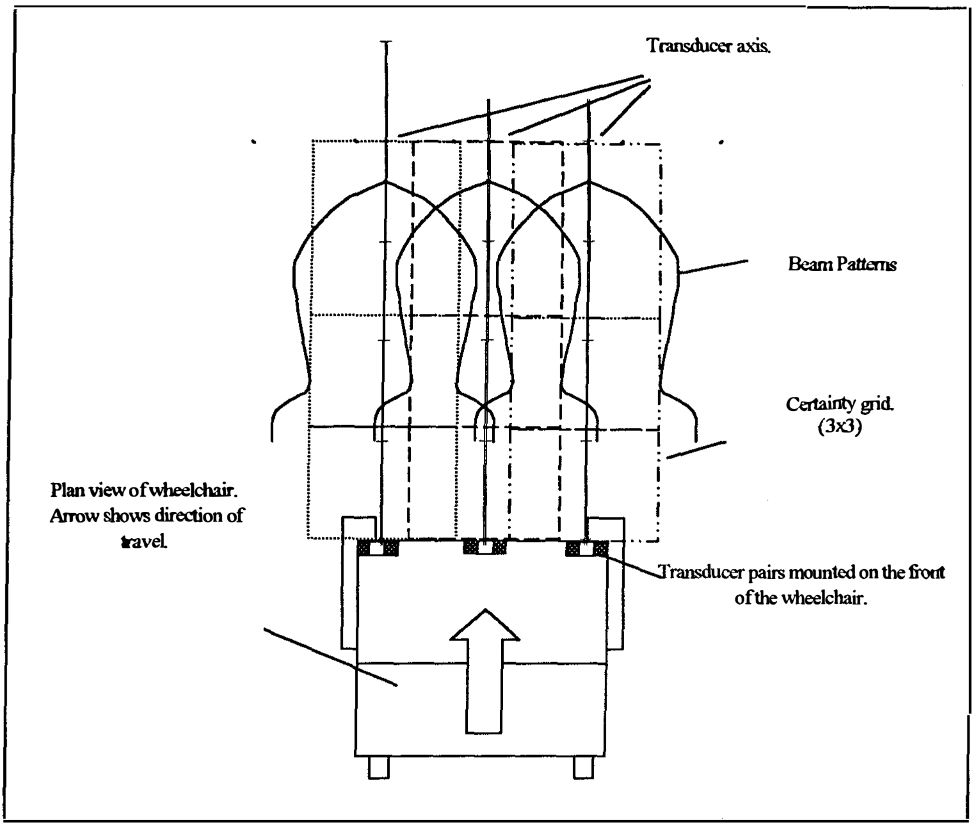


Figure 6.12. Array of three sensors.

sensor array, a total of seven transmitter pulses were required to complete the sensor sweep. It was found that the amount of time required for sensor operation was too high. For ten transmitter operations a time of approximately 400ms was required. This affected the operation of the wheelchair, as the sensor update time was too slow. Tests also revealed that the four-sensor array working at two sweeps per second, but with higher data output, was no more effective than the three-sensor array working at ten sweeps per second, but a lower data output. The simpler and faster three-sensor configuration which used time of flight range finding was selected as the most suitable configuration for this application. The time per sweep was lower and the amount of processing required to represent the environment was reduced.

The final sensor configuration. It was found that the sensor array was more effective if the outer two sensors were angled out from the axis of the wheelchair by about 15 degrees which was the angle of divergence of the beam of ultrasonic energy from the axis of the sensor transducer. This maximised the energy available at each side of the wheelchair and enabled more reliable detection of objects at a greater range. It was more likely that the expert system would steer the trajectory away from objects on the peripheral of the wheelchair's path. Objects on the axis of the wheelchair were usually too close to avoid and the expert system would only have to reduce speed and stop the wheelchair. Using this configuration more reliable detection of navigational features was possible such as doorposts during doorway passages and walls when the wheelchair was driving in a corridor. The final sensor configuration is shown in Figure 6.13.

This configuration was similar to the SCAD sensor array described by Langner (1995) where a single centrally mounted sensor was used that swept from side to side.

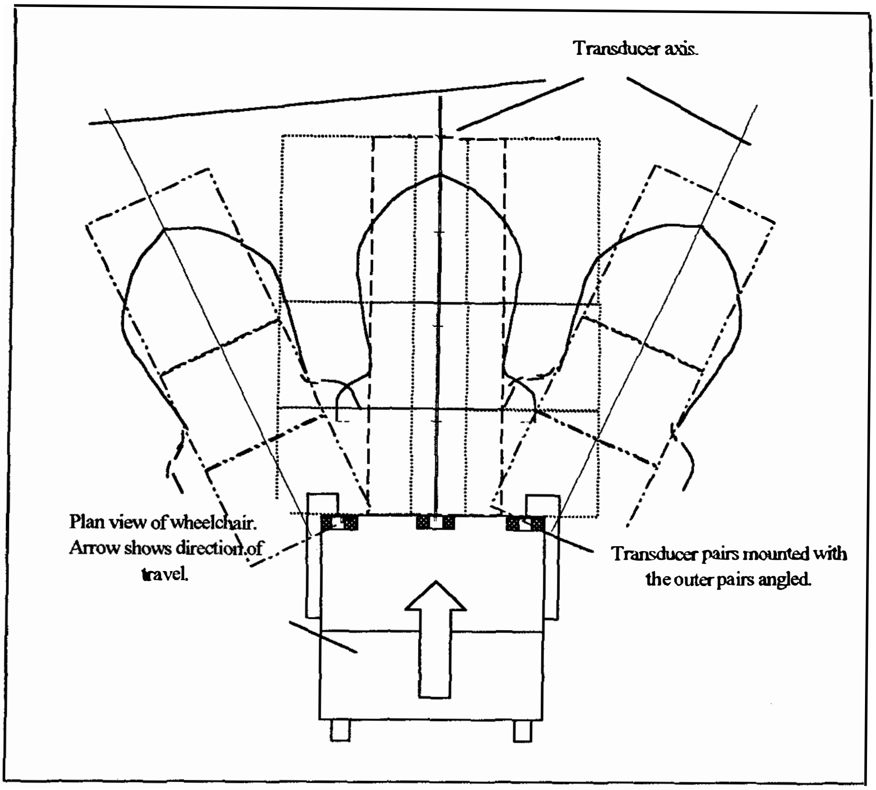


Figure 6.13. The final configuration of the sensors.

6.2. Virtual reality.

The virtual reality simulations were tested and compared. The earlier simulation running on the Superscape (version 3.50) was viewed from a 15 inch computer monitor whilst for IGRIP, a 21 inch higher resolution monitor was available. Both simulations demonstrated a simulation of a powered wheelchair. IGRIP was more useful as it was capable of dynamic simulation and was equipped with powerful functions to assist the programmer to create a realistic simulation of a real-world environment. Superscape, as well as being older programming environment was not as suitable for simulation of a powered wheelchair. The version available for this work was more suitable as a graphical simulation tool than for simulation of dynamic processes.

Superscape VRT 3.50. Some results from the tests of Superscape simulation are shown in Chapter 3. It was possible simulate a powered wheelchair and other objects in Superscape to be visually realistic. Figure 3.6 in Chapter 3 shows the view from the driving seat of the V-wheelchair whilst negotiating a traffic cone in VR. The virtual wheelchair dynamics were difficult to simulate which resulted in the virtual wheelchair moving in a simplistic manner. It was possible to accelerate the virtual wheelchair gradually but it still did not realistically simulate the momentum and characteristics of the real wheelchair. It was difficult to detect and identify other objects in the virtual world. The most effective method used in the work with Superscape was to know the position of objects by their global co-ordinates and to test the wheelchair position. The positions of all the objects in the virtual world needed to be recorded and the possibility of the virtual wheelchair colliding with any of them calculated. If an object were identified as being in the path of the virtual wheelchair, avoiding action could be calculated and executed. When more than one object existed within the virtual world, the code became complex. For ease of description a single object (the traffic cone) was used as an example. However it was not practical to attempt to simulate a real environment using this method.

The IGRIP virtual reality simulation. The virtual model of the virtual environment created by the *IGRIP* software demonstrated a high quality graphical simulation. An example of the virtual wheelchair is shown in Figure 6.14. The images generated by IGRIP were smooth and appeared where necessary to have rounded surfaces. Texture could be added to surfaces. Many facets used to build a round smooth surface.

The creation of a detailed and geometrically complex image increased the workcell size and reduced the simulation efficiency. A balance was needed to ensure a realistic representation and an efficient simulation.

To move the virtual wheelchair around within the virtual world required simple kinematics functions to be added to the main chassis. The virtual wheelchair was programmed to move by the user using a mouse-input device. By pressing the middle, left and right buttons on the mouse, the virtual wheelchair moved forward, left or right.

A virtual sensor array was created that could identify and calculate the range from the virtual wheelchair to an object. The virtual sensor array was required to accurately

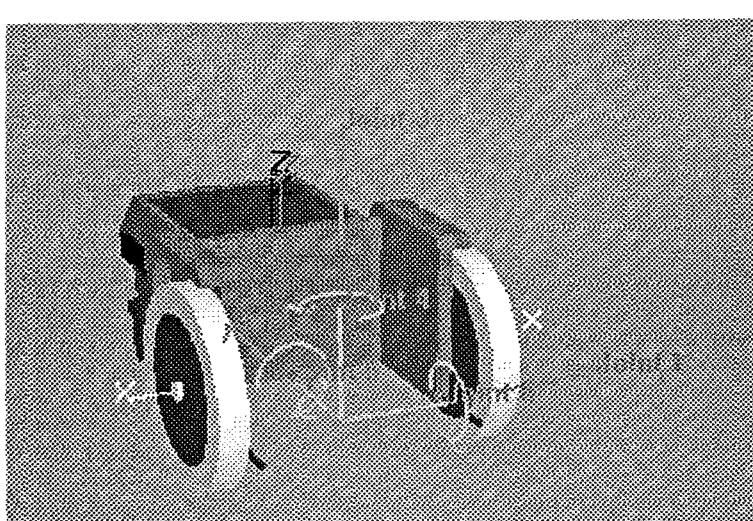


Figure 6.14. The virtual reality powered wheelchair.

simulate the real ultrasonic sensors. The IGRIP "raycast" function was used which could detect and identify any object along a vector specified by the programmer. The real sensors were simulated by grouping raycast vectors so that they covered an area similar to the beam characteristics of the ultrasonic sensors. The range of the vectors was limited so that the maximum range of the sensors were also simulated. Figure 6.15 shows the virtual sensors with the raycast vectors radiating from the sensor positions of the virtual wheelchair.

The virtual sensors were updated every 200 distance units which tended to limit effectiveness when virtual objects were near to the virtual wheelchair. Also the raycast vectors were infinitely narrow and should a small object be positioned within the range of the virtual sensors yet not be bisected by a vector, it would not be detected. As the virtual wheelchair was usually moving, virtual objects would sometimes be "caught" by the sensors and then lost again as they moved out of the path of the vector. This was

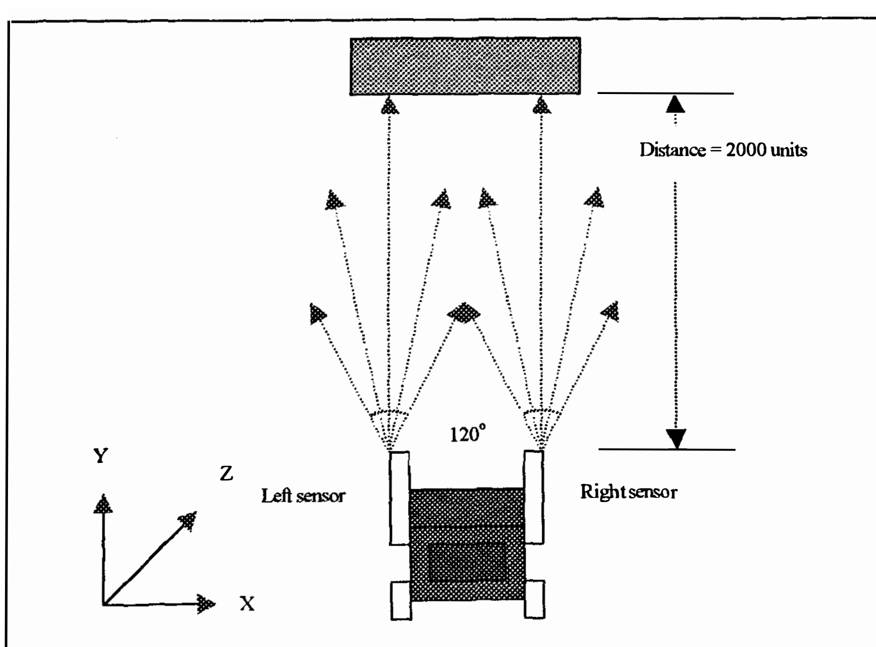


Figure 6.15. The virtual wheelchair with the virtual sensor system fitted.

similar to the real wheelchair as it moved through the environment and the ultrasonic sensors missed objects due to specular reflections or absorption.

The virtual wheelchair was capable of being controlled by a user and of detecting objects in its own environment with its own sensors. Algorithms were programmed into the controller to test by simulation before implementing them on the real wheelchair.

Distance functions were implemented on the virtual wheelchair by using the virtual sensors to calculate the distance to the nearest virtual objects and then reducing the maximum speed of the virtual wheelchair in proportion to the distance measured. The distance functions included:

(a). A step function.

(b) A proportional function.

(c) An obstacle avoidance function.

These are discussed.

(a). A step function:

```
if (dist_r > 2000) then
        f = f - 5
        MOVE JOINT 2 BY f NOSIMUL
else
        s = s - 2.5
        MOVE JOINT 2 BY s NOSIMUL
endif
```

When the dist_r (distance between the sensor and the nearest obstacle) was greater than 2000 units, the wheelchair moved forward by maximum speed of 5 units per simulation, else it slowed down by half of the maximum speed.

(b). A proportional distance function:

```
if (dist_r <= 2000 ) then

        s = s – ((dist_r – 200) / 360)
```

When the distance was calculated to be less than or equal to 2000 units, the wheelchair speed gradually decreased to a stop when the distance had reduced to 200 units.

Figure 6.16 shows the algorithm for the proportional control of virtual wheelchair speed.

(c). An object avoidance function.

```
if (dist_r <= 2000 ) then
        s = s – ((dist_r – 200) / 360)
        l_c = l_c + ((2000 – dist_r) / 10000)
        MOVE JOINT 2 BY s SIMUL
        MOVE JOINT 4 BY l_c NOSIMUL
```

When the distance (dist_r = distance from the right hand sensor to the nearest object) was calculated to be less than or equal to 2,000 units, the forward speed of the virtual wheelchair was gradually decreased and turned to the left (or right) of the initial path.

The IGRIP simulation was flexible and had the potential to be realistic and a convincing representation a real powered wheelchair. The Graphic Simulation Language (GSL) allowed complex dynamic and strategic programs to be written into the virtual wheelchair control code. The visual representations that could be created added a feeling of realism for the user. An enclosed user interface could have added to this. A prototype powered wheelchair simulator has been described and some intelligent algorithms have been applied to the simulation. IGRIP was a capable simulation package that would be suitable for testing intelligent wheelchair algorithms in an intelligent wheelchair simulator. An intelligent wheelchair simulator would also be suitable for training potential intelligent wheelchair users.

6.3 The expert system and controller code.

Chapter 5 described the creation of the innovative and novel software. The software was written in modules and applied to the hardware as the wheelchair was progressively modified. This procedure ensured the software modules could be tested in isolation before they were integrated with others. Software modules were written in C or as low

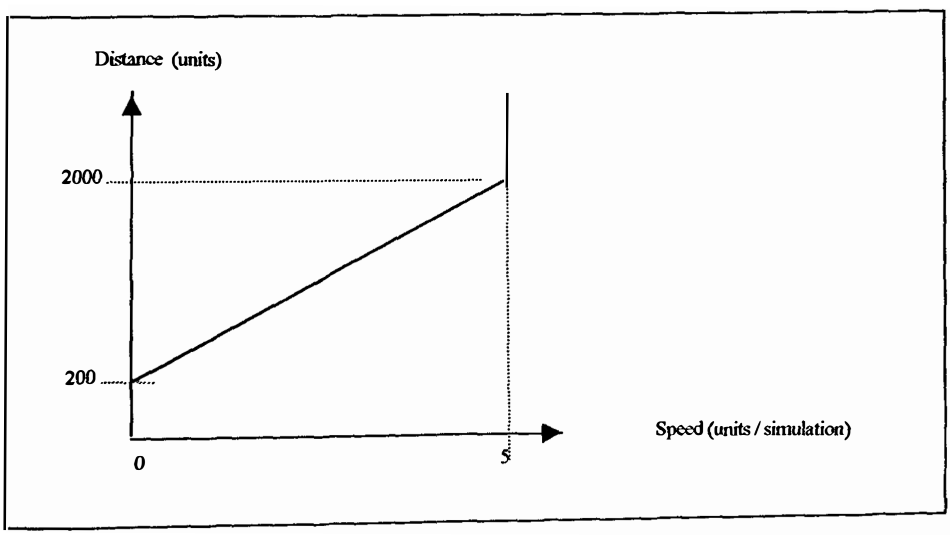


Figure 6.16. The proportional distance function algorithm.

level assembly language. Generally, higher level control functions were written in the higher level language and tasks or functions concerning low level functions of the hardware or requiring accurate timing were written in assembly language.

6.3.1 Analogue to digital conversion of data.

The joystick, controller and sensor data provided information in the form of an analogue signal. The joystick had two analogue channels and these were read during each cycle of the main program. The joystick needed only one analogue to digital conversion per channel to provide an indication of the joystick position. Each time new joystick or controller voltage data were required the MAX1112 analogue to digital (ADC) routine was called and an ADC conversion initiated. The sensors were capable of providing more information. There were up to four sensor channels used during the work and each was read during each cycle of the main program. The same procedure was used for the sensors as was used for reading the joystick data. The sensor data consisted of an analogue signal from the receiver. The receiver signal was sampled and analyzed in real time by the micro-controller code. The sampling process was completed when the analysis of the data stream revealed that the sensors had possibly detected an object.

6.3.2 Controlling the digital outputs.

Depending on the sensor configuration used, three or four bits of an eight-bit input/output (IO) port of the micro-controller were used to control the operation of the transmitters. The duration of the output pulse to fire the transmitter was accurately controlled by interrupt driven timing code in the micro-controller. A digital output from the 8515 micro-controller was pulled high when the sensor was to be activated and remained high for a predetermined time. Whilst the digital output was high the transmitter was active and a pulse of ultrasonic energy was transmitted. As there were up to four channels for the sensors, the procedure was repeated for the other channels.

6.3.3 Analyzing the data.

Three sets of data were acquired from the intelligent wheelchair. These were:

(a). The data from the joystick.

(b) The voltages of the inputs to the wheelchair controller.

(c) The sensor data.

These are discussed.

(a). The joystick data consisted of two integer values that indicated the position of the joystick. As the user of the wheelchair positioned the joystick the data indicated the required motion of the wheelchair.

(b). The sensor data consisted of a recommended direction for the wheelchair to steer. The software detected possible objects from the data supplied by the sensors. The algorithms used to guide the wheelchair mixed the data from the sensors and the joystick. The algorithms were described in Chapter 5.

(c). The wheelchair controller input voltages were monitored in order to provide some feedback for the PWM settings.

The data from the sensors and the joystick were successfully interpreted and analyzed. This resulted in successful wheelchair test runs where the wheelchair trajectory was modified by the expert system and the wheelchair was guided around obstacles in its path. Figure 6.17 shows the intelligent wheelchair negotiating a doorway.

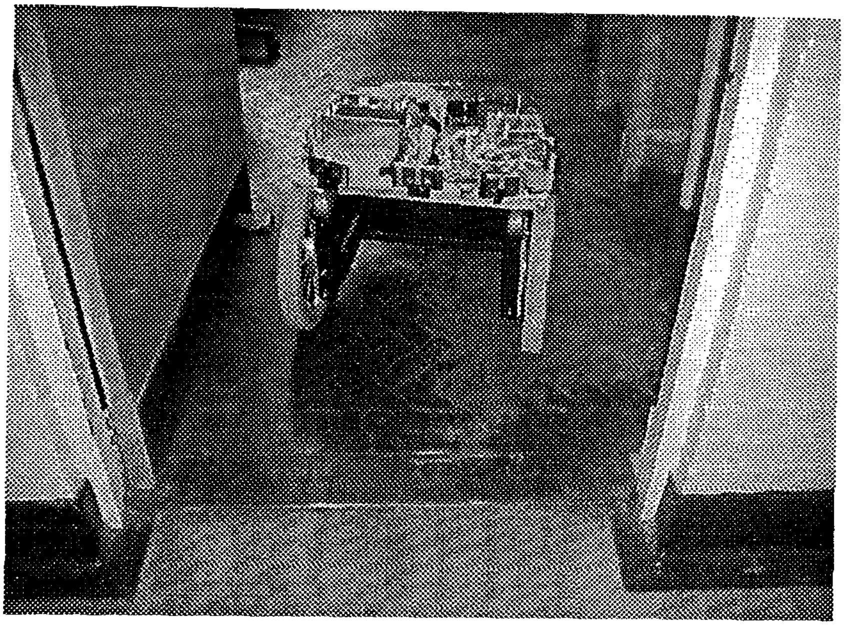


Figure 6.17. The new wheelchair system negotiating a doorway.

6.3.4 The intelligent wheelchair in unstructured environments.

The intelligent wheelchair was tested in the laboratory in a variety of environments including a corridor. Some test runs were conducted with the wheelchair joystick held constantly forward and these are described as an example. If the user of the intelligent wheelchair held the joystick forward consistently, the expert system attempted to drive the wheelchair forward and this included driving around obstacles automatically. The sensor expert suggested trajectory changes and drove the wheelchair without colliding with obstacles. Where the other algorithms were to be tested or the sensor expert was to be over-ridden by the user input, the joystick was manually controlled.

An example of a test run. In a typical test the wheelchair was allowed to suggest a suitable trajectory. An example was when the wheelchair was driving towards a doorway but automatically moved away from a desk. The wheelchair automatically altered its trajectory to move away from the desk and into free space. A typical result is shown in Figure 6.18. The Figure shows that the wheelchair had moved away from the desk, before it was stopped for the purposes of demonstration. The tape on the floor was positioned behind the wheelchair to indicate the path taken. Having moved away from the desk, the tape trail indicated that the intelligent wheelchair had moved to the left to avoid the open door. When the edge of the door had been passed, the system

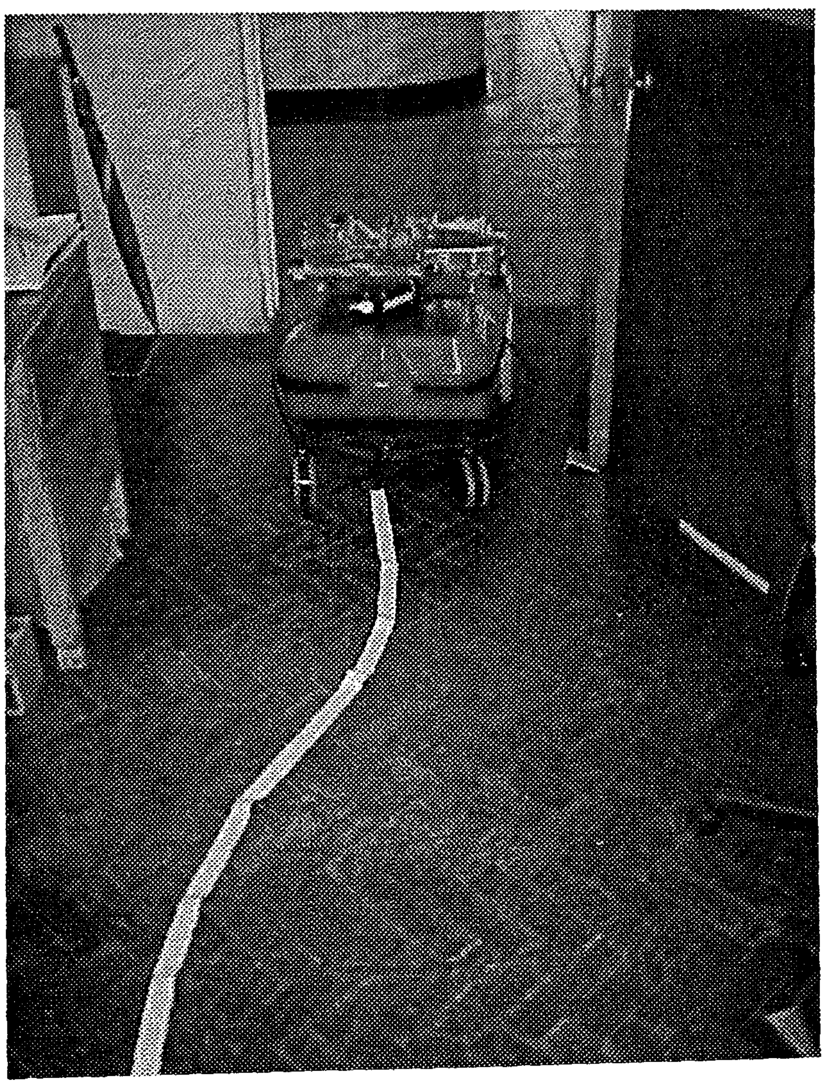


Figure 6.18. A typical wheelchair trajectory during a test.

detected the wall on the left of the doorway. The intelligent wheelchair had then steered to the right, which aligned the wheelchair with the doorway space. The tape was laid for demonstration purposes but was positioned as accurately as possible on the path of the wheelchair.

The joystick being held in the "forward" position simulated an inexperienced or unskilled user attempting to indicate a rough direction. The joystick paddle was secured in the forward position as shown in Figure 6.19 and this was almost a worst case.

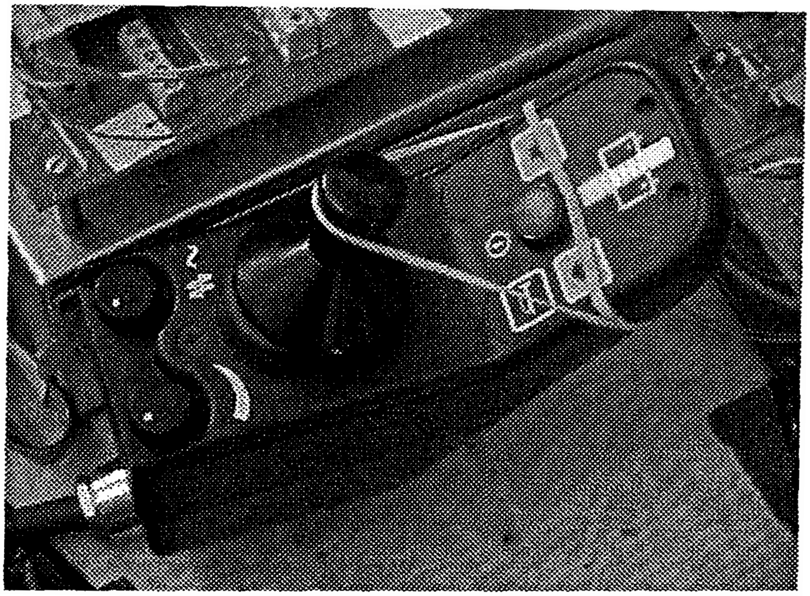


Figure 6.19. The joystick secured in the "forward" position by an elastic band.

With the joystick secured the expert system algorithms were not subjected to "user interference" which could have made some results difficult to quantify and compare. If a user was able to assist more in steering then the trajectories were even safer. The joystick can be seen resting on the rear deck of the wheelchair in Figure 6.18.

Testing with "distance function" obstacle avoidance algorithms. The system was tested with a set of simple proportional distance functions for object avoidance. Similar distance functions were described in Section 5.8.3d. The distance functions were designed to progressively reduce the controller voltage and hence the motor speeds as the wheelchair moved closer to an object. The algorithms were capable of steering the wheelchair away from an object by reducing the speed of the driving wheel on the opposite side of the wheelchair from the obstacle. The wheelchair was able to gently steer away from objects such as a wall in a corridor. Figure 6.20 shows the intelligent wheelchair system during a test run in a corridor. The wheelchair has driven close to a wall and has stopped. A simple algorithm was used during the tests to stop the wheelchair when any sensor detected an object nearer than a pre-set distance. This

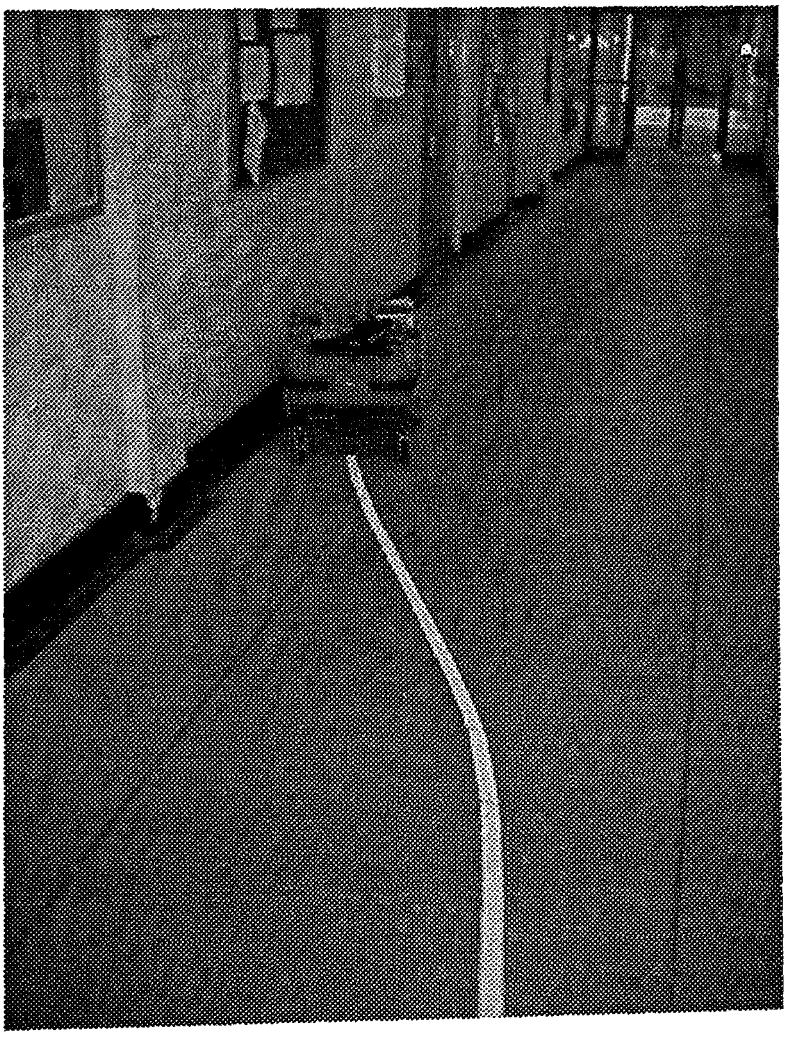


Figure 6.20. An unsuccessful corridor run.

algorithm was to protect the wheelchair from damage during tests where the wheelchair was directed on collision courses on purpose. In a real situation the user would be expected to provide an input to explain what they wanted to do. As can be seen in Figure 6.20, the wheelchair failed to avoid the wall, as it did not turn steeply enough. This was due to the setting of the variables in the navigational algorithms and these could be adapted depending on the ability of the wheelchair user. As the wheelchair approached the wall the distance algorithms reduced the speed of the motors in proportion to the distance from the sensors to the wall. The left-hand sensor was closer to the wall so the right hand motor reduced speed more than the left-hand motor. This resulted in the wheelchair beginning to turn to the right. The tape on the floor behind the wheelchair indicates this. The right hand sensor had also detected the wall as the beam width of the sensors made large targets such as the wall in this case detectable by the edge of the beam. This invoked the right hand sensor distance function to reduce the speed of the left-hand motor. Although the right hand motor speed was reduced more than the left-hand motor speed, any reduction in the left-hand motor speed would tend to negate the effect of the right hand. Figure 6.21 shows this effect for a similar scenario. The use of simple distance function algorithms was also tested during doorway tests. A doorway test presented a different problem for the expert system as objects (the doorposts) were detected on both sides of the wheelchair. Unless the doorway was unrealistically wide, the distance functions tended to slow both wheelchair drive motors down and could stop them when the wheelchair was just entering the doorway. The wheelchair was not able to continue through the doorway. This situation was a local minima where the algorithms had forced the wheelchair to remain in one position and had no way to extract the wheelchair from it. Figure 6.22 shows the wheelchair in a local minima position at the entrance to a doorway. The distance from

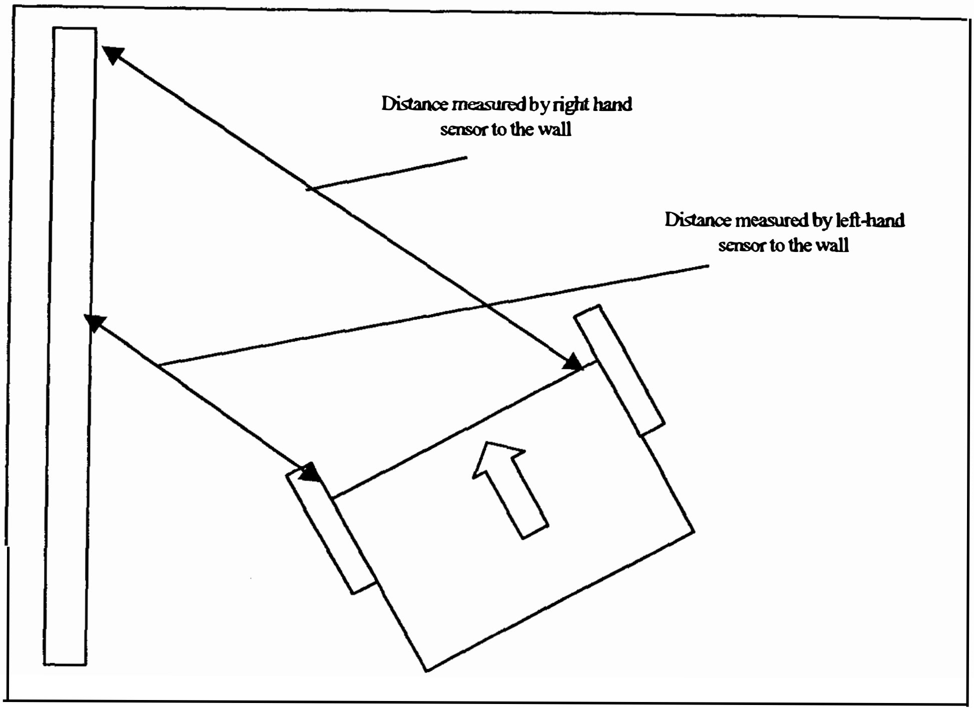


Figure 6.21. The reduced steering effect of distance functions against large objects.

the left sensor to the left doorframe is similar to the distance from the right sensor to the right doorframe. In this case distance measurements were small enough to overcome the demand for forward motion by the joystick-input and stop the wheelchair motors. The wheelchair remained in this position until the user intervened by adjusting the input. These results indicated that the application of a navigation strategy based on simple mathematical functions may have disadvantages. Preventing the wheelchair from driving within a preset distance of an object did not allow for the required manouvre space for the wheelchair and considered distance as of primary importance function. A powered wheelchair has a primary function of transportation, and this must also be considered when proposing novel control strategies for a powered wheelchair. Expert systems have been used where mathematical solutions were not always the most effective answer to a problem and this was considered valid for this work.

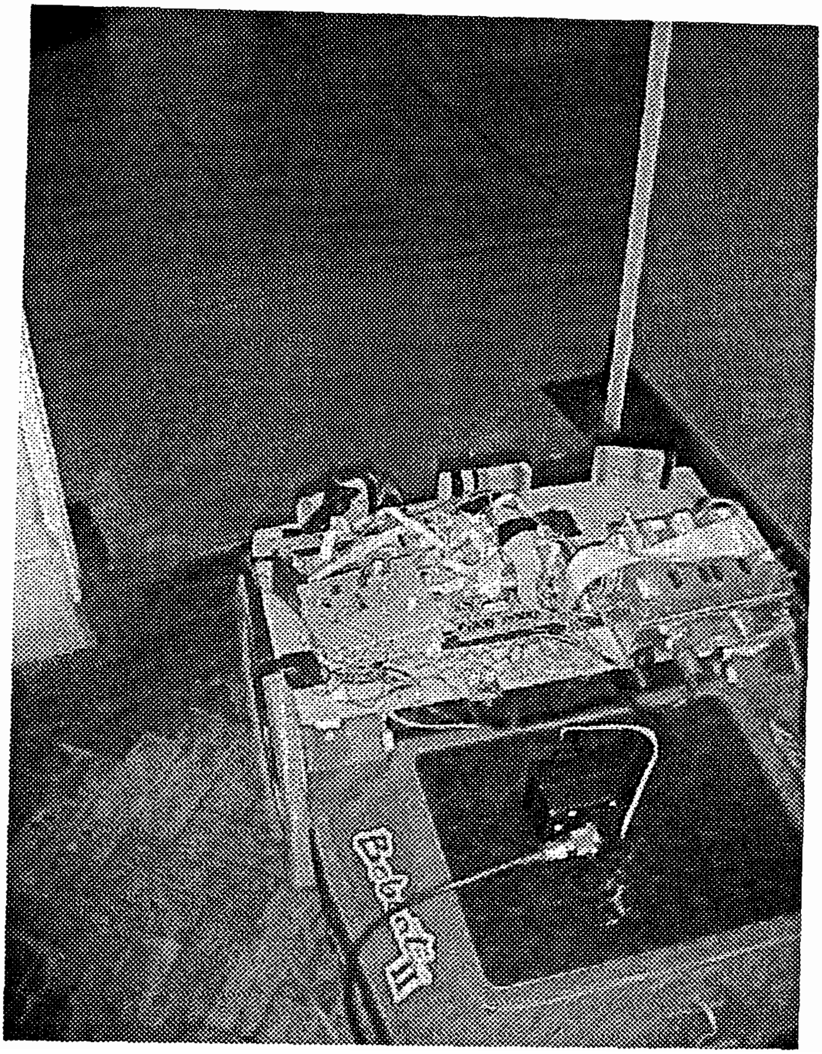


Figure 6.22. The wheelchair stuck in a local minima position.

Testing some expert algorithms. To avoid problems associated with local minima and subsequent non-compliance with the wishes of the user, expert algorithms were investigated as a method for navigational assistance for a powered wheelchair user. Simple expert algorithms were tried and these were discussed in Chapter 5.

Whilst driving a powered wheelchair, some users find it difficult to correct the wheelchair trajectory to avoid objects in the path of the wheelchair. Algorithms were tested that assisted in the safe navigation of a powered wheelchair when the wheelchair user was driving through a cluttered area. A typical algorithm is shown:

```
if (Joystick == forward)

        {if (SensorDistanceLeft<SensorDistanceRight)

                SteerRight;

        if (SensorDistanceLeft>SensorDistanceRight)

                SteerLeft:

        }

        else Forward;
```

These simple algorithms emulated a user steering a powered wheelchair. If an object threatened to collide with the right hand side of the wheelchair, the expert system would help the user to steer left and vice versa. The algorithms were tested by driving the intelligent wheelchair through a doorway and along a corridor. Typical results for a doorway test were shown in Figure 6.18 and a typical corridor test is shown in Figure 6.23.

The wheelchair was driven on a collision course with the wall from position 1 on Figure 6.23. Whilst the intelligent wheelchair was driving in free space, the expert algorithm did not alter the trajectory from that demanded by the joystick so from position 1 to position 2 the wheelchair drove forward. As it approached 3, the sensors detected the wall. The distance from the left-hand sensor array to the wall was less than the distance to the right hand sensor array. The expert algorithm caused the wheelchair to turn to the right and away from the wall. In the absence of any correction or other inputs from the user, the wheelchair continued in a forward direction until at 4 it steered to the left in response to the presence of the right hand wall of the corridor.

The intelligent wheelchair made numerous test runs along a corridor rapidly acquired the middle of the corridor. Figure 6.24 shows a different test run in the same corridor. The

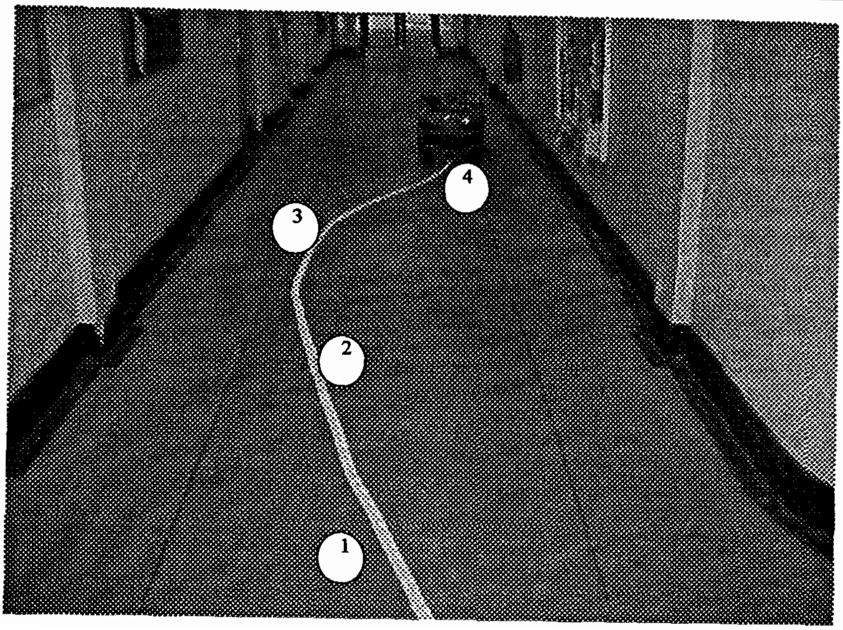


Figure 6.23. The intelligent wheelchair under test in a corridor.

expert algorithms showed consistency in the trajectories that were taken. The expert system was designed to bend the trajectory away from obstructions to take some of the burden of control from the user. A wheelchair user who was proficient at driving a powered wheelchair along a corridor in a straight line would not benefit from a navigational assistance system of this type unless they became tired. The intelligent wheelchair was mainly aimed at the severely disabled who have little fine motor control skills or spatial awareness.

On some test runs, the wheelchair turned away from the wall and continued turning for longer than the minimum time required to avoid the wall. The extended turn time was sometimes due to the wall in that position reflecting more ultrasonic energy back to the receivers. This was due to features such as notice boards cluttered with paper notices that presented a random reflecting surface to the sensors. If the wheelchair turned too much, it could present itself at a steep angle to the opposite wall of the corridor. In these situations, the expert system required an input from the user to direct the

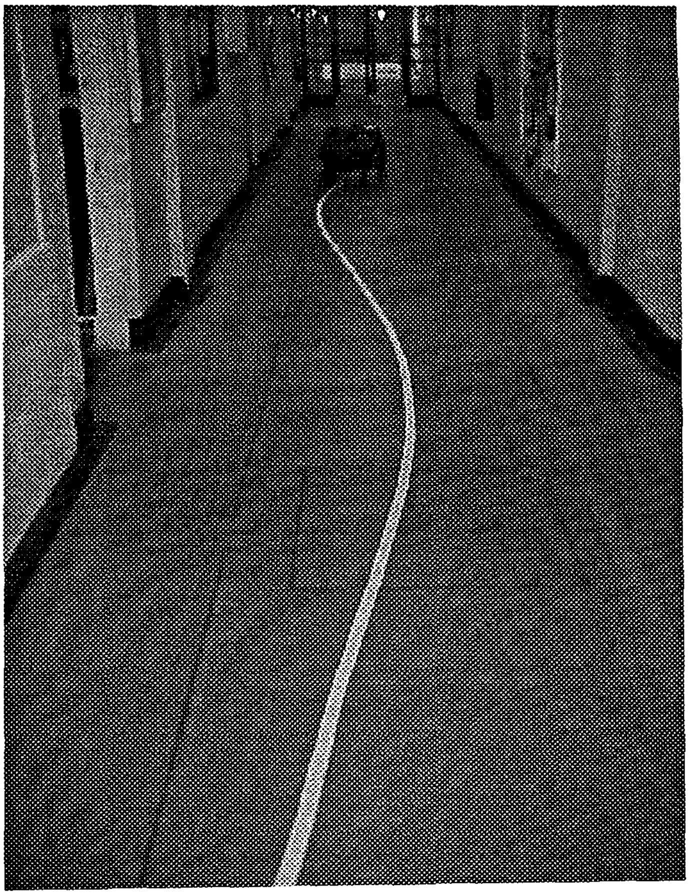


Figure 6.24. A typical corridor wheelchair test run.

wheelchair back on course.

The user override algorithms. In order to allow the user to veto the intelligent wheelchair control algorithms, the user needed to have the option to override the expert algorithms. The algorithms to measure the level of confidence in the joystick position were discussed in Chapter 5. The joystick position was monitored and if an indication could be obtained from the user then it was allowed to overcome the effect of the Sensor Expert recommendation. Figure 6.25 shows the wheelchair that had been driven up against a wall. If the joystick was unsteady and the expert system could not be confident about the intentions of the user, the wheelchair would not move until the expert system was confident about an indication from the joystick. If the value of the

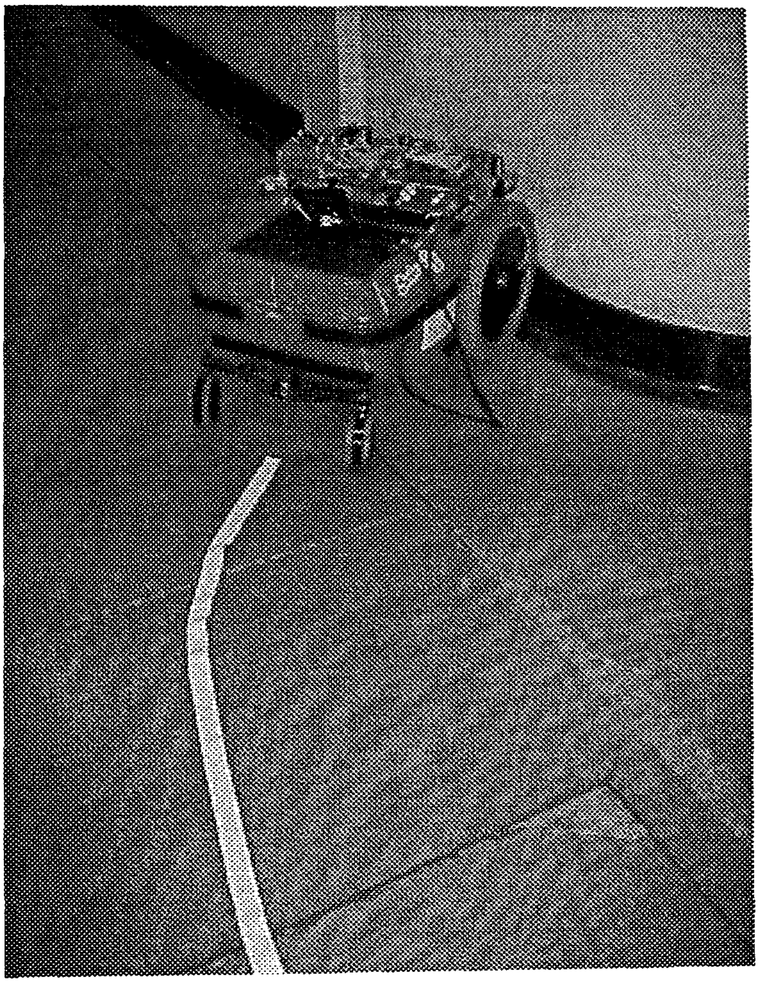


Figure 6.25. The wheelchair system deliberately driven up to a wall.

joystick had ramped up (which indicated a positive decision by the user), the wheelchair would attempt to drive in the direction indicated by the user. Even if this meant trying to drive into the wall. Algorithms prevented the full power of the wheelchair from being applied to the motors as the wheelchair motors were limited when objects were close. However the wheelchair could still be used to push objects out of the way if the user wished to. Figure 6.26 shows the wheelchair position after the joystick was held in the forward position with the wheelchair's front against the wall. The wheelchair has been powered so as to force itself around so that it was square on to the wall. A function that allowed the user to push objects may be suitable for some users, algorithm tailoring may

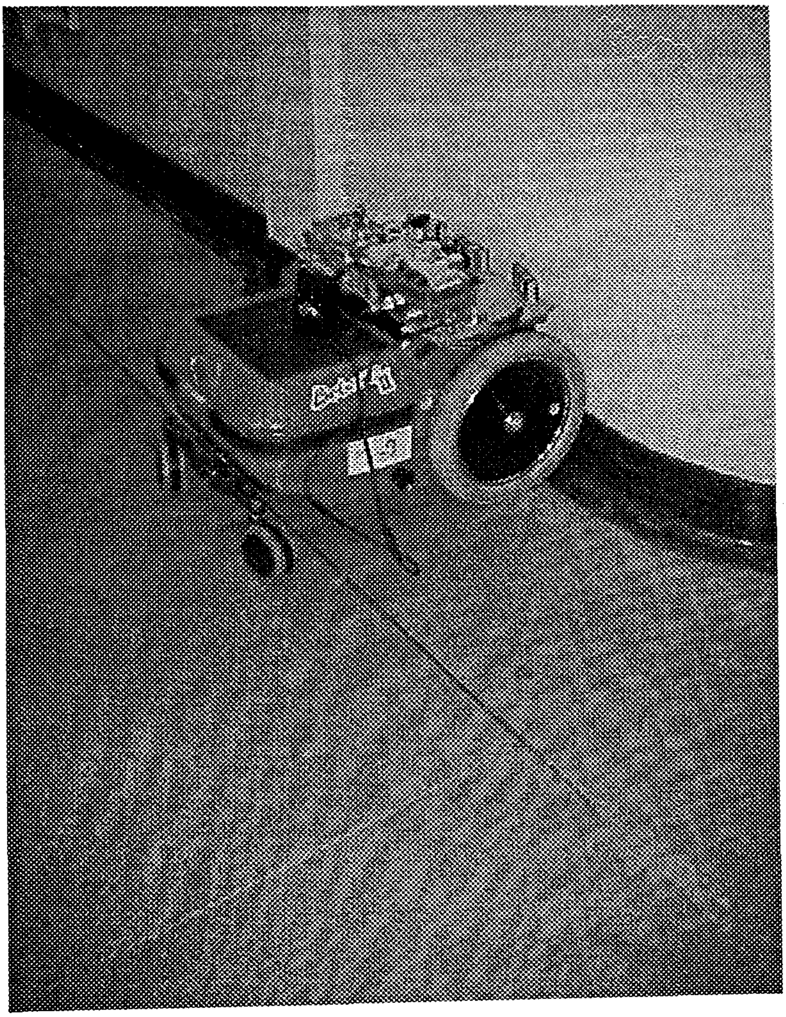


Figure 6.26. The wheelchair system forced into the wall by constant joystick demand.

be important to fit individuals requirements dependent on their driving skills and demeanor.

Similar algorithms were used to ensure that the wheelchair was able to turn towards an object when the expert system was attempting to turn it away. The user involvement was intuitive as, if the wheelchair was turning away from the direction the user wished to turn, the joystick could be pushed to contradict the sensor expert recommendation and the expert system could be quickly overridden.

Modifying the expert system involvement. It was possible to modify the amount by that the expert system affected the trajectory of the wheelchair. The mixing algorithms could be altered to involve the expert system from 0 to 100%. If the intelligent wheelchair was operating at 100% reliance on the expert system it was implied that the user involvement was 0%, or the user was just a passenger. There may be occasions when this is desirable such as when a user is inexperienced in powered wheelchair driving. The wheelchair could be programmed to perform a series of automatic movements for the user to gain experience of the feeling of motion.

It was more likely and within the philosophy of this work that user involvement would change over time. The effect of the sensor expert and mixing algorithms could be adjusted and this was tested. Within the code, weighting factors could be altered to reduce the effect of the expert system. It was found that the effect of reducing the expert system involvement would not necessarily increase the user's level of involvement. Even with the intelligent wheelchair operating normally, the user was still involved with maneuvering the wheelchair. The expert system attempted to assist the user but when the user was operating the joystick correctly the expert system did nothing. The user remained involved whilst driving the wheelchair and needed to concentrate on maneuvering. If the user did not drive the wheelchair into the approximate position and orientation to complete the driving operation, the expert system would stop the wheelchair rather than take over and drive the user in a direction that they did not want. The biggest effects that the intelligent wheelchair expert system had were whilst the wheelchair was being driven in a straight line and also when the wheelchair was close to objects and risking a collision. During fine maneuvering operations, the user was almost in total control as the expert system could not predict and then try to override what the user was trying to do.

6.4 Examples of variations of the parameters.

The selection for the weighting factors or degrees of influence that the novel algorithms were attributed was an important part of this work. Steering algorithms using a distance function have been discussed in this Chapter. Other algorithms that had effects on the overall response of the intelligent wheelchair were; sensor algorithms, joystick interpretation algorithms and the algorithms to mix different sets of data. It was found through experimentation and prediction that some of the algorithms needed to be tuned so that they operated with their maximum effectiveness. This Section discusses the algorithms that had a direct effect on the navigational and assistive behaviour of the intelligent wheelchair.

6.4.1 The effect of changing the steering algorithms.

During early tests, distance functions were used to demonstrate a method of steering the intelligent wheelchair. This has been discussed in Section 6.2.3 and Figure 6.20 illustrated the effect of distance functions that were too weak to oppose the effect of the joystick input and so could not prevent the wheelchair from driving into the wall. Figure 6.27 shows a simple linear distance function and indicates the effect that altering the weighting of the function had on the response of the intelligent wheelchair. Making the responses non-linear changed the effect of the distance function and improved the scenario where the wheelchair was driving toward a wall. Non-linear distance functions tended to prevent the wheelchair from being driven close to an object as the effect of the algorithm increased out of proportion to the distance to the object. This is shown on figure 6.28. Initially, although the object had been detected in the path of the wheelchair, the effect of the distance function algorithm was minimal until the

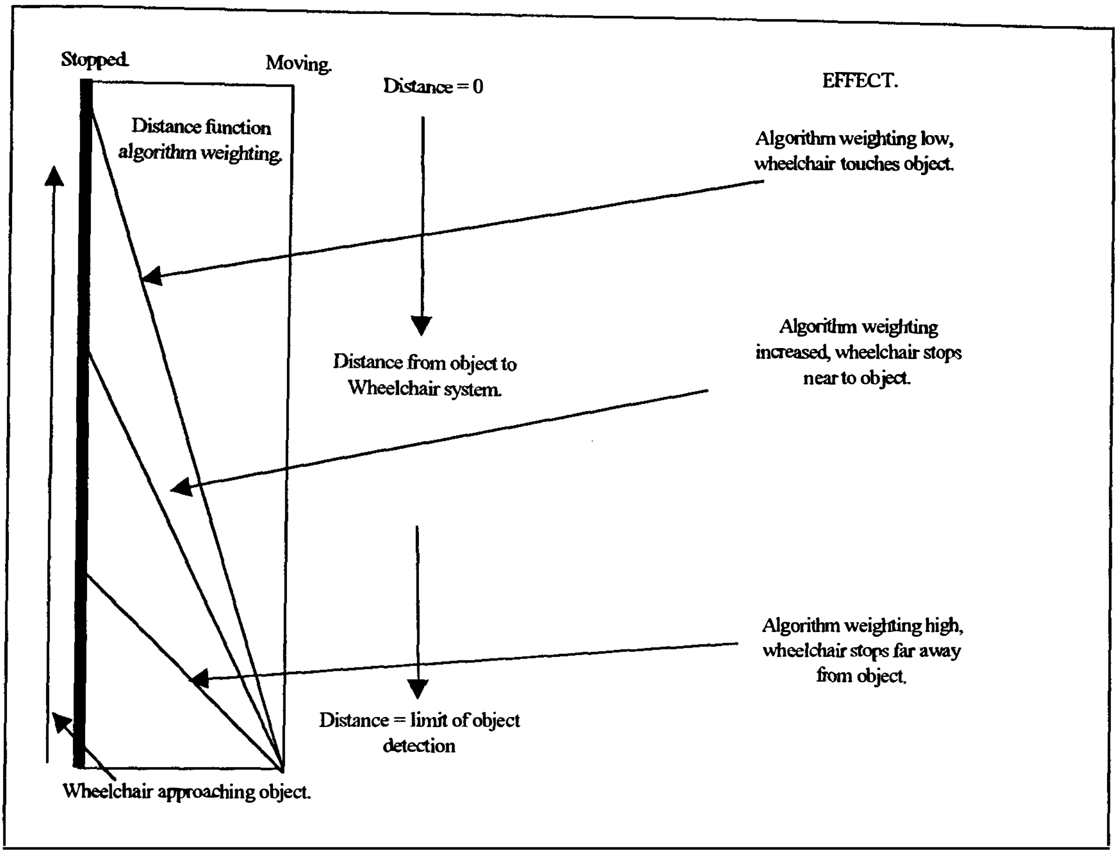


Fig 6.27. The effect on the wheelchair performance when the distance algorithm weighting is changed.

wheelchair was closer to the object. The effect of the algorithms was to reduce the speed of the wheelchair more rapidly near to the object and allow the wheelchair to move more quickly when the distance to the object was greater. This allowed the wheelchair to move closer to an object but the expert system was still able to have some effect on the trajectory and speed of the wheelchair. The local minima effect was still a problem (although less so) when trying to navigate between two objects such as a doorway. Figure 6.22 showed the wheelchair stuck in local minima and unable to pass through the doorway. A proportional distance function algorithm was improved by making the distance functions compare the distances between the front left and front right sensors and steering left or right away from the nearest object. This algorithm never stopped the wheelchair but caused the wheelchair to wiggle between objects. The comparison algorithm was discovered by observing a wheelchair user manouvring in a

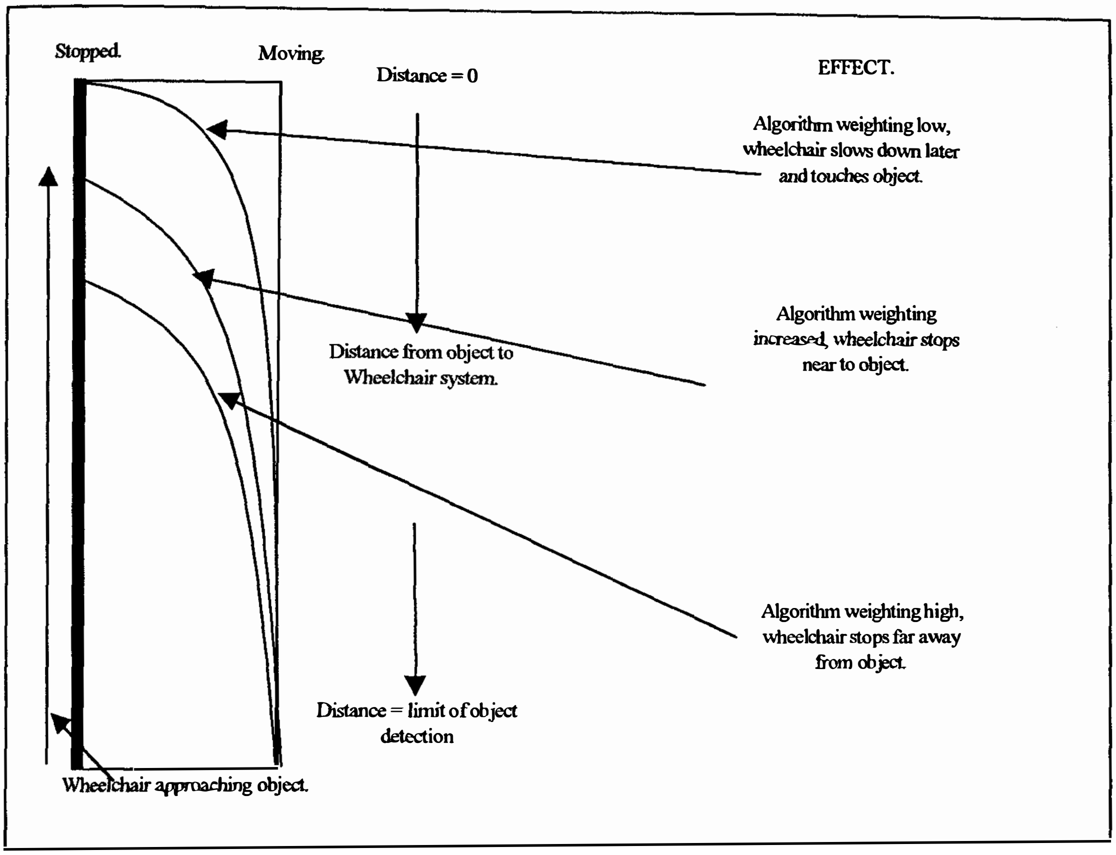


Fig 6.28. The effect on the wheelchair performance when the non-linear distance algorithm weighting is changed.

cluttered environment. It was noticed that the user was moving the joystick in large, side to side sweeps to move the wheelchair precisely through a difficult manouvre. The comparison algorithms used distance functions, but the distance functions were used to steer the wheelchair by stopping one driving wheel or the other, never both together.

6.4.2 Variations in the Sensor expert algorithms.

As discussed in Section 5.4.1, and shown in Figure 5.6, the sensor "picture" was built up by ramping up and ramping down histograms. The histogrammic representation was less prone to mis-reads and noisy data. Each histogram represented and area in front of the sensor array that could contain an object. The speed that the histogram ramped up and down was critical. If the ramping up was too slow, the sensors did not object an object until the wheelchair was close to it and the intelligent wheelchair may not have enough time to avoid the object. However, the sensors were less effected by noise.

If the ramping was too quick, noise and mis-reads were recognised as possible detections and the purpose of the histogrammic method was lost. The sensors became unreliable and reported "ghost" objects.

Figure 5.7 shows that where detection was suspected, the appropriate element was incremented by three, then all elements were decremented. Therefore the suspected occupied element ramped up to a maximum value in half the time that an element could ramp down from the maximum.

After the sensor information had been interpreted, the sensor expert was able to recommend a change in trajectory to the fuzzy mixer if appropriate. Sensor expert contained a list of actions to suggest for every set of sensor data that was possible. The initial Sensor expert algorithm suggested actions that ware designed to progressively alter the wheelchair trajectory. The trajectory was changing too slowly and often resulted in a collision as shown in Figure 5.22. It was found with later algorithms that Sensor expert should suggest a more positive action that caused the wheelchair to turn sharp left for example. These novel expert algorithms tended to instruct the wheelchair to "spin left or right" by stopping the appropriate drive wheel. These algorithms resulted in a trajectory for the wheelchair that was more successful at avoiding objects. Figure 5.25 shows a typical trajectory for the intelligent wheelchair using the novel expert algorithms.

6.4.3 Altering the joystick interpreting algorithms.

The joystick data was presented as a set of Cartesian values by the joystick. This data format was difficult to interpret and so it was converted to polar form in the micro-controller. A Histogrammic algorithm was used to interpret the intention of the user and each histogram represented a desired direction for the wheelchair to move, derived from the angle of the joystick paddle. The magnitude or required speed of the wheelchair was derived by the position of the joystick paddle. A joystick histogram array element took 0.5 seconds to reach the maximum value and 0.17 seconds to decay from the maximum to zero. Slowing the response of the joystick algorithms made interpretation of the users intentions more accurate, but a slow response from the joystick reduced the confidence of the user in the intelligent wheelchair. If the intelligent wheelchair was slow to respond to the joystick input, the perceived link between the joystick and the wheelchair movements was broken. This made the control of the wheelchair difficult for a user who expected the wheelchair to respond promptly to joystick inputs.

6.4.4 Changing the Fuzzy mixer algorithms.

The Fuzzy mixer was created so that the information from the joystick and the sensors could be mixed. The proportion that the two data were mixed in was critical to the performance of the intelligent wheelchair. Figure 5.16 shows how the joystick and sensor information could be apportioned between the joystick and the sensors. The confidence algorithms controlled the decision over how much control should be apportioned between the user and the expert system. These algorithms are described in Section 5.9.2. Reducing the amount of influence the expert system had on the control of the wheelchair made the wheelchair behave as a normal powered wheelchair as the joystick had the biggest influence on the control of the wheelchair. If the expert system were allowed to assume more control of the wheelchair, the wheelchair behaved more as an intelligent asistive wheelchair system. Using the confidence parameter, the fuzzy mixer altered the importance of the user and the expert system to reflect confidence in the user, or the need for the expert system to act to provide a safer trajectory for the wheelchair to follow.

6.5 Discussion.

The intelligent wheelchair was tested in controlled conditions in and outside of the laboratory, but always indoors. Specific wheelchair maneuvers were used as complex examples for the tests. A door passage was selected as it was considered to be a typical maneuver that a wheelchair may perform on a regular basis. A door passage maneuver contains many elements of other maneuvers. A doorway was considered to be a space between two objects; the doorposts. In a domestic environment the floor could be scattered with objects of varying sizes. The wheelchair should pass through the space between them to safely navigate.

To move through an environment with random obstructions was a similar problem to driving along a corridor. The intelligent wheelchair was required to react to objects appearing in front of the wheelchair and to move in the direction of least clutter. The behavior was described as reacting to an algorithm that demanded the wheelchair move into open space. The intelligent wheelchair demonstrated successful assistive doorway passages. The algorithms used were simple and targeted at navigating through gaps in obstructions where the expert system needed to make decisions on which algorithm to apply. Switching between algorithms was automatic and seamless as production rules were selected in response to sensor and user inputs.

The work presented in this Dissertation has demonstrated that it is possible to assist a wheelchair user to navigate a wheelchair without taking control away from the user. A minimum number of sensors have been used and the accurate mapping of the environment was avoided although the wheelchair would not always manage a collision free journey depending on the skill of the user and the settings of the variables in the algorithms. The addition of close range sensors and algorithms for recovering the wheelchair from close proximity situations might be required for safer operation, although the intelligent wheelchair has only been created to assist a user and could not replace the "common sense" required for safe wheelchair driving.

Chapter 7 contains a more detailed discussion of the results and a consideration of future work.

Chapter Seven

DISCUSSION

7.1 The purpose of this work.

The prototype was created to test new algorithms to help a disabled wheelchair user to navigate a wheelchair through an environment such as a home. Novel local planning methods were selected to assist the operator in steering the vehicle. The expert system attempted to find a suitable trajectory that were close to the vector requested by the operator but which moved away from close objects. Algorithms simulated an expert powered wheelchair driver and suggested a route that would prevent the wheelchair driving close to objects. The algorithms were weighted by the wishes of the user. If the expert system detected that the user was unsure or inconsistent, the on-board expert system was given more importance in the selection of the wheelchair route. If the user had been consistent in the joystick inputs, the wheelchair user was given more importance in the selection of the wheelchair path. The system could be over-ridden if the user opposed the expert system and was consistent in the use of the joystick. Data from ultrasonic sensors were processed and combined with the information from the joystick. The system used this combined information to assist the operator in steering the vehicle.

The intelligent wheelchair prototype was autonomous as the intelligent wheelchair equipment was all fitted to the wheelchair chassis. This allowed realistic testing operation during tests. Low cost and simple electronic circuits were built and innovative software using a simplified expert system and novel algorithms demonstrated an innovative and useful approach to assisting a powered wheelchair user.

7.2 A summary of achievements.

The following were created specifically for the intelligent wheelchair prototype:

- A novel expert system was created to modify the wheelchair trajectory based on a simplified Blackboard Architecture.
- This novel expert system was be implemented on a simple low cost hardware platform using a minimal production rule set.
- Simple but innovative algorithms were created which were effective for controlling an assistive wheelchair system.
- Innovative histogrammic maps were used to generate confidence in sensor and user data.
- Innovative training methods and simulations of novel algorithms were created using virtual reality in Superscape and IGRIP.

This work demonstrated that:

- Simple expert systems can be implemented on low cost hardware.
- An effective system to assist a powered wheelchair user can be created by reducing the production rule set.
- Elaborate, processor hungry algorithms are not always necessary for simple assistive systems.

7.3 Conclusions and recommendations for future work.

The wheelchair performed repetitive navigation tasks with the joystick held in the forward position under test conditions. The wheelchair was not always successful in choosing the correct path through a doorway or along a corridor and would sometimes collide with an object or get stuck against a wall. However in the event of a collision, the expert system had often slowed the wheelchair down prior to the impact. It was more likely that the expert system was successful in providing navigational assistance to the user and the numbers of collisions were reduced.

Suggested areas for future work:

- Close collaboration with a wheelchair controller manufacturer so that the intelligent wheelchair code can be embedded in the wheelchair control firmware. This will provide access to data regarding the motor speeds for example, which may be useful for odometric analysis. The response of the intelligent wheelchair to stimuli will be improved also as the delay between a change in a control signal sent to the controller and the reaction to it can be eliminated.

- The commercial viability of an intelligent wheelchair system should be assessed.

- Experimentation with the weighting strategies applied to the sensor and user data could improve the response of the intelligent wheelchair. Some work has been completed using distance functions with linear and squared relationships. Further improvements may be possible with adaptive distance functions where speed and historical use and reaction data can be considered.

- Clinical trials are needed so that the usefulness of the intelligent wheelchair system can be analysed by medical staff and recommendations for the advancement of the project identified.

- The intelligent wheelchair also has potential as a diagnostic and monitoring tool to assess and monitor the health and condition of the user through the patterns of use of the wheelchair. This potential aspect of the work has not been investigated and will require extensive clinical trials to assimilate suitable data.

- Many powered wheelchairs are rear wheel drive. The intelligent wheelchair prototype created for this work has only been tested on a front wheel drive wheelchair. The intelligent wheelchair prototype should be tested on other powered wheelchair platforms and the algorithms adapted if necessary to create a more generic assistive system.

The prototype intelligent wheelchair presented in this Dissertation was created to assist wheelchair users and potential wheelchair users. Simple algorithms were used to demonstrate assistance to the user at a low level. Relatively high levels of assistance could have been possible with this system at the expense of real time operation. The improvements suggested could speed new systems up and add more intelligence. The new systems could be made relatively cheaply and be flexible enough to interface directly to existing wheelchairs.

Appendix A.
The QuickBasic 4.5 program listing for sampling a sensor test using the 80486 based test rig. A sample test file from a test showing 200 samples is listed. A data analysis program listing which was used to test object detection algorithms is also listed, and a raw data graph generated from the test file and gradient plot from the same data are included.

The QuickBasic 4.5 program listing for sampling a sensor test using the 80486 based test rig.

```
'program name : 4sin.asm.
'an output pulse on DOO initialises a sensor range test.
'the result of the test should appear on D1 low byte.

SCREEN 9
COLOR 15, 1

OUT &H22D, 0                      ' turn off Tx
               'Set up the IO board.
param%(0)  = 0                    ' Board number
param%(1)  = &H220                ' Base I/O address
param%(2)  = 1                    ' DMA level
param%(3)  = 1                    ' DMA level
param%(4)  = 2                    ' IRQ level : IRQ2
param%(5)  = 2                    ' Pacer rate = 2M/(2 * 120)=8.333KHz
param%(6)  = 120                  'or max range = 3.96m
param%(7)  = 0                    ' Trigger mode, 0 : pacer trigger
param%(8)  = 0                    ' Non-cyclic
param%(13)  = 0                   ' must set to 0.
param%(17)  = 0                   ' Overall gain code, 0 : +/- 5V
param%(31)  = 1                   'one DI conversion
param%(32)  = 0                   'DI port 0
'
FUN% = 3                          ' FUNCTION 3
CALL PCL812(FUN%, SEG param%(0))     ' Func 3 : Hardware
initialization
IF param%(45) <> 0 THEN PRINT "DRIVER INITIALIZATION FAILED !": STOP

DO                                ' Turn on Tx
OUT &H22D, 1                      'set pulse width x
FOR i = 0 TO 1: NEXT i
(1=0.2ms,50=2.5ms)
OUT &H22D, 15                     'turn off Tx pulse
SLEEP 1                           'wait 1 second.
result = INP(&H226)
LOCATE 10, 10: PRINT "the result of the test is____"; result
LOCATE 6, 20: PRINT "continue with a new test?: <Y>es or <N>o?
"
A$ = INKEY$
IF A$ = "y" OR A$ --"Y" THEN GOSUB resett
LOOP UNTIL A$ <> ""
END
```

The following QuickBasic 4.5 code contains an algorithm that detected the gradient change in the sensor data that indicated the presence of an object.

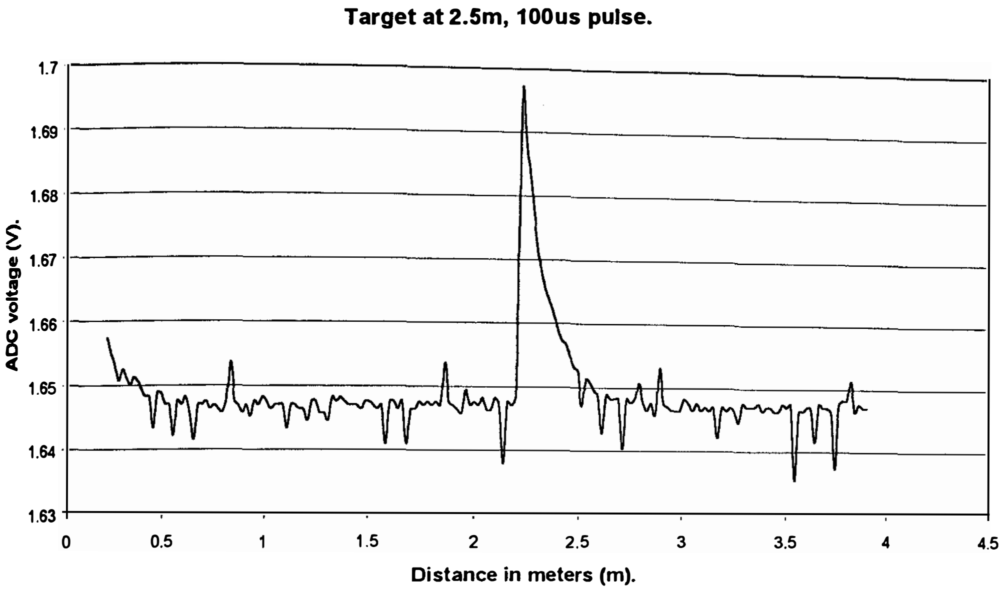
```
'A program for analysing sensor data from ultrasonic tof
'(TIME OF FLIGHT)sensors.                    Ian Stott 11/8/98

SCREEN 9                                     'Prepare arrays.
DIM dat(400)
DIM voltage (400)
DIM distance(400)
DIM gradient(400)
                                             'Extract data from file.
OPEN "c:/qbfiles/sensor-l/00-00-12.txt" FOR INPUT AS #1
FOR i = 1 TO 400
INPUT #1, dat(i)
NEXT i
                                             'Convert ADC data to voltage.
FOR i = 2 TO 400 STEP 2
voltaqe(i) = (dat(i) - 2030) /417

NEXT
                                             'Convert sample number to distance

FOR i = 1 TO 399 STEP 2
distance(i) = (i + 16) * .01

NEXT
                              'Calculate the qradient of each part of the graph.

FOR i = 4 TO 400 STEP 2
qradient(i) = voltaqe(i) - voltaqe(i - 2)

NEXT
                    'Check for large negative spikes (noise) and
                    'Calculate the place where the qradient is more than O.lv/sample

FOR i = 2 TO 400 STEP 2
        IF gradient (i -2)< -.1 THEN GOTO jump
        IF qradient(i) > .1 THEN GOSUB detected
jump:

NEXT i

LOCATE 10. 10: PRINT "no detections."
END

detected:
PRINT "something detected at"; distance(i + 1); "m"
END
```

Sample test file of data collected from a sensor test.

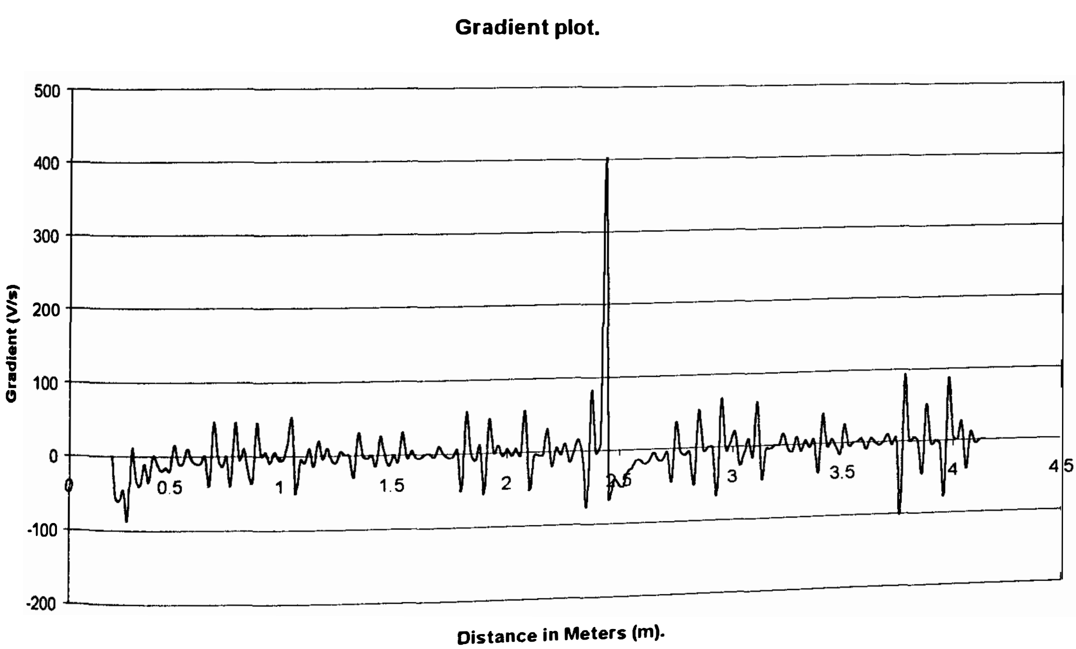
200 samples collected at 8333Hz.

1,2774
2,2762
3,2769
4,2746
5,2739
6,2733
7,2729
8,2718
9,2720
10,2718
11,2716
12,2714
13,2708
14,2709
15,2707
16,2705
17,2706
18,2704
19,2705
20,2703
21,2702
22,2708
23,2700
24,2704
25,2700
26,2702
27,2704
28,2699
29,2699
30,2698
31,2701
32,2693
33,2700
34,2700
35,2700
36,2686
37,2700
38,2700
39,2700
40,2700
41,2703
42,2701
43,2700
44,2699
45,2700
46,2693
47,2700
48,2701
49,2700
50,2699
51,2700
52,2700
53,2700
54,2701
55,2714
56,2700
57,2699
58,2700
59,2702
60,2694
61,2699
62,2700
63,2700
64,2700
65,2700
66,2700
67,2700
68,2699
69,2701
70,2700
71,2698
72,2701
73,2700
74,2701
75,2698
76,2702
77,2699
78,2700
79,2704
80,2699
81,2697
82,2700
83,2701
84,2694
85,2699
86,2700
87,2699
88,2696
89,2701
90,2701
91,2701
92,2700
93,2701
94,2701
95,2701
96,2701
97,2699
98,2691
99,2699
100,2700

101,2701
102,2698
103,2700
104,2698
105,2876
106,2948
107,2922
108,2880
109,2866
110,2852
111,2861
112,2854
113,2877
114,2850
115,2827
116,2810
117,2788
118,2779
119,2769
120,2757
121,2742
122,2743
123,2736
124,2732
125,2726
126,2728
127,2719
128,2716
129,2715
130,2711
131,2711
132,2709
133,2709
134,2706
135,2705
136,2702
137,2703
138,2702
139,2702
140,2697
141,2701
142,2701
143,2702
144,2701
145,2702
146,2701
147,2700
148,2700
149,2700
150,2694
151,2698
152,2701
153,2699
154,2698
155,2701
156,2701
157,2700
158,2700
159,2707
160,2701
161,2700
162,2699
163,2700
164,2691
165,2699
166,2700
167,2701
168,2701
169,2692
170,2700
171,2699
172,2701
173,2696
174,2700
175,2700
176,2700
177,2699
178,2711
179,2700
180,2701
181,2700
182,2700
183,2701
184,2700
185,2699
186,2700
187,2702
188,2697
189,2700
190,2699
191,2700
192,2688
193,2699
194,2701
195,2700
196,2698
197,2698
198,2701
199,2500
200,0



Receiver voltage plot created from the data listed.



A gradient plot of the data listed.

Appendix B1 .

Code to modify the PWM outputs:

```
SetPWM:                          //run this to set motor speed

    CPI    R16,0                 //select channel
    BRNE   chanB                 //not PWM Ch A
    MOV    R18,R20               //make copy of PWMValue
    CPI    R18,0x01              //is value negative?
    BRGE   AllokL                //Branch if more than 1
    LDI    R20,0x01              //don't let R20 be less than 0

AllokL:                          //R20 must be >1 by now

    OUT    OCR1AL,R20            //set PWM Left value
    RJMP   endit

chanB:

    CPI    R16,1                 //select channel
    BRNE   endit                 //not PWM Ch B
    MOV    R18,R20               //make copy of PWMValue
    CPI    R18,0x01              //is value negative?
    BRGE   AllokR                //Branch if more than 1
    LDI    R20,0x01              //don't let R20 be less than 0

AllokR:                          //R20 must be >1 by now

    OUT    OCR1BL,R20            //set PWM Right value

endit:                           //channels should be set now.

RET
```

Appendix B2. GetNewADCvalue, used to access the ADC via the SPI.

```
GetNewADCvalue:
        ANDI    R16,0X07              ;Mask out Higher bits
        LSR     R16                   ;Rotate bit 0 into carry
        BRCC    Sel2Clear

#if (Device_Number =1113) /* 4 chan */
        ORI     R16,B'00000010                    ;set bit

Sel2Clear:
        LSL     R16

#else   /* 8 Chan Device */
        ORI     R16,B'00000100                    ; set bit

Sel2Clear:
#endif
        SWAP    R16                   ;move Chan num to high nibble
        ORI     R16, Control_Byte     ;R16 now holds the control byte
        CBI     PORTB,PB4             ;Set SS Low

        OUT     SPDR,R16              ;output Control Byte
        CLR     R18                   ;Clear R18 ready for next Tx

WaitSPIF1:
        IN      R17,SPSR              ;Read SPI Status reg into R17
        CPI     R17,LOW(0)
        BRGE    WaitSPIF1             ;wait for R17 to become negative (i.e.
SPIF =1)
                                      ;Control Byte now fully Clocked out

        IN      R19,SPDR              ;Read in SPI Data (Spoof read - ignore
this!)
        OUT     SPDR,R18              ;start clocking in next word

WaitSPIF2:
        IN      R17,SPSR              ;Read SPI Status reg
        CPI     R17,LOW(0)
        BRGE    WaitSPIF2             ;wait for R17 to become negative (i.e. SPIF =1)

        IN      R16,SPDR              ;Read in SPI Data
        OUT     SPDR,R18              ;start clocking in next word

        LSL     R16                   ;Pre process R16
        LSL     R16

WaitSPIF3:
        IN      R17,SPSR              ;Read SPI Status reg
        CPI     R17,LOW(0)
        BRGE    WaitSPIF3             ;wait for R17 to become negative (i.e. SPIF =1)

        IN      R18,SPDR              ;Read in SPI Data
        SBI     PORTB,PB4             ;Set SS Low

        SWAP    R18                   ;Get the two bytes to the low nibble
        LSR     R18                   ;
        LSR     R18                   ;Shift to Bit 0,1
        OR      R16,R18               ;Mix R16, R18
        RET

        END
```

Appendix B3

The test code using printfu (a simplified print function that used less memory space) to indicate that the code had correctly calculated the sector for the joystick. The printfu function could be replaced by any function to react to the joystick position for testing or for driving the wheelchair.

```
if (magnitude<16)
  printfu(STxt);   //joystick in the stop position

else if ((angle>=0.48)&&(angle<0.98)&&(JS1>10))
  printfu(FTxt);   //angle was straight ahead

else if ((angle>=0.98)&&(angle<1.58)&&(JS1>10))
  printfu(TLTxt);  //angle was turn left

else if ((JS0>10)&&(angle>=-1.58)&&(angle<-0.785))
  printfu(SLTxt);  //angle was spin left

else if ((JS0>5*JS1)&&(JS0>10)) //covers infinity bit of ARCTAN
  printfu(SLTxt);  //angle was spin left

else if ((JS0>10)&&(JS1>10))
  printfu(TRTxt);  //angle was turn right

else if ((angle>=-0.785)&&(JS1>10)&&(JS0<10))
  printfu(SRTxt);  //angle was spin right

else printfu(BTxt);       //The joystick was pulled into reverse.
     printfu(EndTxt);     //End of test.
```

Appendix B4.

The code to build and decay the joystick histograms was contained in module JSArray which is listed below.

```
if (magnitude<20)                                        //joystick in stop position

else if ((angle>=0.68)&&(angle<0.88)&&(JS1>10))
        sector=0;                                        //angle was straight ahead

else if ((angle>=0.88)&&(angle<1.58)&&(JS1>10))
        sector=1;                                        //angle was turn left

else if ((JS0>10)&&(angle>=-1.58)&&(angle<-0.785))
        sector=2;                                        //angle was spin left

else if ((JS0>5*JS1)&&(JS0>10))                          //covers infinity bit if ARCTAN
        sector=2;                                        //angle was spin left

else if ((JS0>10)&&(JS1>10))
        sector=3;                                        //angle was turn right

else if ((angle>=-0.785)&&(JS1>10)&&(JS0<10))
        sector=4;                                        //angle was spin right

else
        sector=5;                                        //dummy sector

//increase sector confidence

if (sector!=5)
Aconf[sector]=Aconf[sector]+40;

Joysticksector=5;                                        //reset
highest = 0;
for (j=0;j<5;++j)
{
   Aconf[j]=Aconf[j]-30;                                 //decay all sectors
   if (Aconf[j]<0)                                       //min is 0
      Aconf[j]=0;
   if (Aconf[j]>255)                                     //max is 255
      Aconf[j]=255;

   if (Aconf[j]>highest)                                 //test for highest JS conf
        {
        highest = Aconf[j];                              //most confident sector
        Joysticksector = j;
        }
}
```

Appendix B5.

Algorithms were simplified to speed up the response of the system and to make the system more predictable.

In the case where the joystick and the sensor expert were indicating "forward", the system set the trajectory as straight-ahead. The sensor system was still interrogated to determine the distance that the wheelchair was from the nearest object. The speed of the wheelchair was reduced as the wheelchair became closed to an object;

```
if (Middlesensor<10)
        {
        TargetLeft=150;                  //Slow the wheelchair down
        TargetRight=150;                 //still driving forward
        }

if (Middlesensor<7||Rightsensor<7||Leftsensor<7)
        {
        TargetLeft=145;                  //Slow down more, the
        TargetRight=145;}                //wheelchair is close to an object.
        if (Aconf[Joysticksector]>200)   //The joystick is constantly forward.
                {
                TargetLeft=150;          //The user must want to carry on
                TargetRight=150;         //let the wheelchair touch the object if required.
                }
        }

break;

case 5:                                  //sensors said STOP!
STOP();
if (Aconf[Joysticksector]>200)
        {
        TargetLeft=150;                  //Sensors said STOP but the joystick still
        TargetRight=150;                 //indicates FORWARD. Move slowly forward.
        }

break;
default:                                 //sensors were go left or right
if (Leftsensor>Rightsensor)              //Take avoiding action by spinning
    Spinleft();                          //the wheelchair away from the object.
if (Rightsensor>Leftsensor)
    Spinright();

break;
```

Appendix B6.

The algorithms prevented the wheelchair from driving quickly into an obstruction. The code shown below would be used in the case where the user was holding the joystick in a right turn position when the sensors had detected an object in the area to the right of the wheelchair.

```
        //Example; the sensors said "object right" and the joystick indicated "go right"

if (Middlesensor<10)                       //Objects are near.
        {
        TargetLeft= 180;                   //Slow the wheelchair down
        TargetRight=146;                   //still driving to the right
        }

if (Middlesensor<7||Rightsensor<7||Leftsensor<7)
        {
        TargetLeft=150;                    //Objects are close.
        TargetRight=145;                   //Slow down to crawl to touch the object.
        }
```

The algorithm did not stop the wheelchair completely, but allowed the user to approach and touch the object with consistent use of the joystick.

Appendix C

```
'Number puzzle game solution program, by Ian Stott.
'This program was written to test A.I. program techniques and uses
'a "hill climbing" algorithm to find a solution.
'A preprogrammed puzzle martix is included, change the 'array to change
the problem.

Setup:

CLS
COLOR 15, 1
DIM goalarray(3, 3) DIM startarray(3, 3) DIM temDarravl3. 3\

count = 0
'sets up the Goal puzzle matrix
goalarray(1, 1) = 7
goalarray(2, 1) = 6
goalarray(3, 1) = 5
goalarray(1, 2) = 8
goalarray(2, 2) = 0
goalarray(3, 2) = 4
goalarray(1, 3) = 1
goalarray(2, 3) = 2
goalarray(3, 3) = 3

'sets up the Start puzzle matrix
startarray(1, 1) = 7
startarray(2, 1) = 0
startarray(3, 1) = 5
startarray(1, 2) = 1
startarray(2, 2) = 6
startarray(3, 2) = 4
startarray(1, 3) = 2
startarray(2, 3) = 8
startarrav(3. 3} = 3

GOSUB display

DO

GOSUB evaluate function oldevaluation = eval eval = 0
```

```
'make a temporary copy of the start array
FOR x = 1 TO 3: FOR y = 1 TO 3
temparray(x, y) = startarray(x, y)
NEXT: NEXT

    GOSUB slider
    GOSUB success
    GOSUB display

waiter:
a$ = INKEY$: IF a$ = "" THEN GOTO waiter

    GOSUB blankspace
count  = count + 1

LOOP UNTIL whoopee = 1

END

evaluatefunction:
eval = 0
FOR x = 1 TO 3
FOR y = 1 TO 3
IF startarray(x, y) = 0 THEN GOTO over
IF startarray(x, y) <> goalarray(x, y) THEN eval = eval + over:
NEXT: NEXT

FOR xg = 1 TO 3: FOR yg = 1 TO 3
FOR xs = 1 TO 3: FOR ys = 1 TO 3
IF startarray(xs, ys) = goalarray(xg, yg) THEN eval = eval + (ABS(xs -xg)
+ ABS(ys -yg))
evalll = evalll + 1
NEXT: NEXT: NRXT : NF,XT

RETURN

tempevaluatefunction:
eval = 0
FOR x = 1 TO 3
FOR y = 1 TO 3
IF temparray(x, y) = 0 THEN GOTO overtoo
```

```
IF temparray(x, y) <> goalarray{x, y) THEN eval = eval + 1
overtoo:
NEXT: NEXT
FOR xg = 1 TO 3: FOR yg = 1 TO 3
FOR xs = 1 TO 3: FOR ys = 1 TO 3
IF temparray(xs, ys) = goalarray(xg, yg) THEN eval = eval + (ABS(xs -xg)
ABS(ys -yg})
NEXT: NEXT: NEXT: NEXT
tempeval = tempeval + 1
RETURN

display: 'sets up a start and goal matrix on the screen CLS
LOCATE 7,
20 x = 1
FOR y = 1 TO 3
PRINT startarray(y, x);
NEXT y
LOCATE 6, 20
x = 2
FOR y = 1 TO 3
PRINT startarray(y, x);
NEXT y
LOCATE 5, 20
x = 3
FOR y = 1 TO 3
PRINT startarray(y, x);
NEXT V

LOCATE 14, 20
x = 1
FOR y = 1 TO 3
PRINT goalarray(y, x)
NEXT y
LOCATE 13, 20

x = 2
FOR y = 1 TO 3
PRINT goalarray(y, x);
NEXT y
LOCATE 12, 20
x = 3
```

```
FOR y = 1 TO 3
PRINT goalarray(y, x);
NEXT y
LOCATE 16, 20: PRINT count
LOCATE 25, 5: PRINT "press a key to step through the program."
RETURN

slider:
'This subroutine moves the space around the array to try to solve the
'problem.

'make a temporary copy of the start array
FOR x = 1 TO 3: FOR y = 1 TO 3
temparray(x, y) = startarray(x, y)
NEXT: NEXT

'move space down
IF yspace -1 < 1 THEN GOTO jump2
temparray(xspace, yspace) = temparray(xspace, yspace) - 1
temparray(xspace, yspace- 1) = 0
     GOSUB tempevaluatefunction
IF eval < oldevaluation THEN GOTO save

jump2:

'move space right
IF x space + 1 > 3 THEN GOTO jumpl
'rnake a temporary copy of the start array
FOR x = 1 TO 3: FOR y = 1 TO 3
temparray(x, y) = startarray(x, y)
NEXT: NEXT

temparray(xspace, yspace) = temparray(xspace + 1, yspace)
temparray(xspace + 1, yspace) = 0
     GOSUB tempevaluatefunction
IF eval < oldevaluation THEN GOTO save

jumpl:

'move space left
IF x space -1 < 1 THEN GOTO jurnp3
```

```
'-make a temporary copy of the start array
FOR x = 1 TO 3: FOR y = 1 TO 3
temparray(x, y) = startarray(x, y)
NEXT: NEXT

temparray(xspace, yspace) = temparray(xspace -1, yspace)
temparray(xspace- 1, yspace) = O
     GOSUB tempevaluatefunction
IF eval < oldevaluation THEN GOTO save jump3:


'move space up
IF yspace + 1 > 3 THEN GOTO jump4
'make a temporary copy of the start array
FOR x = 1 TO 3: FOR y = 1 TO 3
temparray(x, y) = startarray(x, y)

NEXT: NEXT

t~parray(xspace, yspace) = temparray(xspace, yspace + I )
temparray(xspace, yspace + 1) = 0
     GOSUB tempevaluatefunction
IF eval < oldevaluation THEN GOTO save jump4:


save:
'save temp array as itr is a good move, this becomes our start array next
'itteration.
FOR x = 1 TO 3: FOR y = 1 TO 3
startarray(x, y) = temparray(x, y)
NEXT: NEXT

RETURN

success:
FOR X = 1 TO 3: FOR y = 1 To 3
IF startarray(x, y) <> goalarray(x, y) THEN RETURN
NEXT: NEXT
whoopee = 1
RETURN

blankspace
```

```
FOR x = 1 TO 3: FOR y = 1 TO 3
IF startarray(x, y) = O THEN
x space = x yspace = y
END IF

NEXT: NEXT:
RETURN
```

Appendix D.

Sensor Byte by action classification.

Spin Left

000011	03
000111	07
001011	0B
010011	13
010111	17
011011	1B
100011	23
100111	27
101011	2B

Turn Left

000001	01
000010	02
000110	06
010010	12
010110	16

Stop

001100	0C
001101	0D
001110	0E
001111	0F
011100	1C
011101	1D
011110	1E
101100	2C
101101	2D
101110	2E
101111	2F
111100	3C
111101	3D
111110	3E
111111	3F
011111	1F

Spin Right

110000	30
110001	31
110010	32
110100	34
110101	35
110110	36
111000	38
111001	39
111010	3A

Turn Right

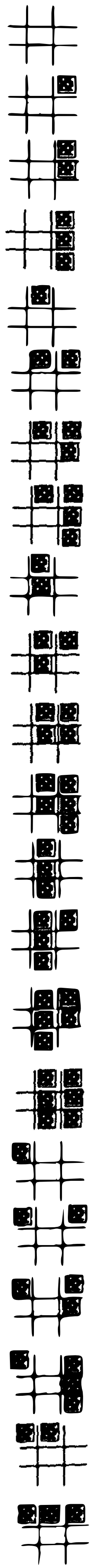
010000	10
100000	20
100001	21
100100	24
100101	25

Slow

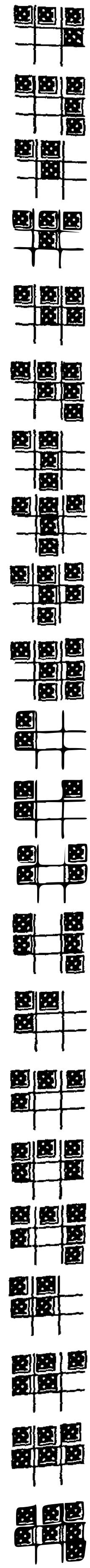
000100	04
000101	05
001000	08
001001	09
001010	0A
010001	11
010100	14
010101	15
011000	18
011001	19
011010	1A
100010	22
100110	26
101000	28
101001	29
101010	2A
110011	33
110111	37
111011	3B

Appendix D. Sensor Byte mapping.

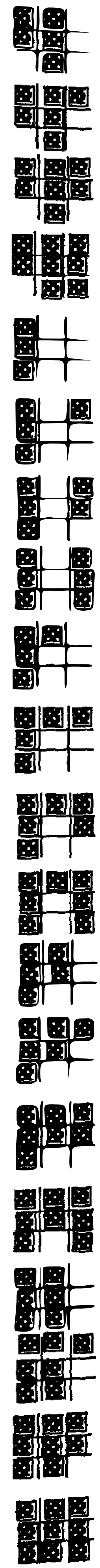
000000	00
000001	01
000010	02
000011	03
000100	04
000101	05
000110	06
000111	07
001000	08
001001	09
001010	0A
001011	0B
001100	0C
001101	0D
001110	0E
001111	0F
010000	10
010001	11
010010	12
010011	13
010100	14
010101	15



010110	16
010111	17
011000	18
011001	19
011010	1A
011011	1B
011100	1C
011101	1D
011110	1E
011111	1F
100000	20
100001	21
100010	22
100011	23
100100	24
100101	25
100110	26
100111	27
101000	28
101001	29
101010	2A
101011	2B



101100	2C
101101	2D
101110	2E
101111	2F
110000	30
110001	31
110010	32
110011	33
110100	34
110101	35
110110	36
110111	37
111000	38
111001	39
111010	3A
111011	3B
111100	3C
111101	3D
111110	3E
111111	3F



Appendix E. Sensor Expert Rule Set.

Sensor Byte.		Action.
000000	00	Nothing
000001	01	Turn Left
000010	02	Turn Left
000011	03	Spin Left
000100	04	Slow
000101	05	Slow
000110	06	Turn Left
000111	07	Spin Left
001000	08	Slow
001001	09	Slow
001010	0A	Slow
001011	0B	Spin Left
001100	0C	Stop
001101	0D	Stop
001110	0E	Stop
001111	0F	Stop
010000	10	Turn Right
010001	11	Slow
010010	12	Turn Left
010011	13	Spin Left
010100	14	Slow
010101	15	Slow
010110	16	Turn Left
010111	17	Spin Left
011000	18	Slow
011001	19	Slow
011010	1A	Slow
011011	1B	Spin Left
011100	1C	Stop
011101	1D	Stop
011110	1E	Stop
011111	1F	Stop
100000	20	Turn Right
100001	21	Turn Right
100010	22	Slow
100011	23	Spin Left
100100	24	Turn Right
100101	25	Turn Right
100110	26	Slow
100111	27	Spin Left
101000	28	Slow

Sensor Byte.		Action.
101001	29	Slow
101010	2A	Slow
101011	2B	Slow
101100	2C	Stop
101101	2D	Stop
101110	2E	Stop
101111	2F	Stop
110000	30	Spin Right
110001	31	Spin Right
110010	32	Spin right
110011	33	Slow
110100	34	Spin Right
110101	35	Spin Right
110110	36	Spin Right
110111	37	Slow
111000	38	Spin Right
111001	39	Spin Right
111010	3A	Spin Right
111011	3B	Slow
111100	3C	Stop
111101	3D	Stop
111110	3E	Stop
111111	3F	Stop

Appendix F. Prototype circuit boards created for the new work.

Figure F1. The Philips micro-controller prototype PCB.

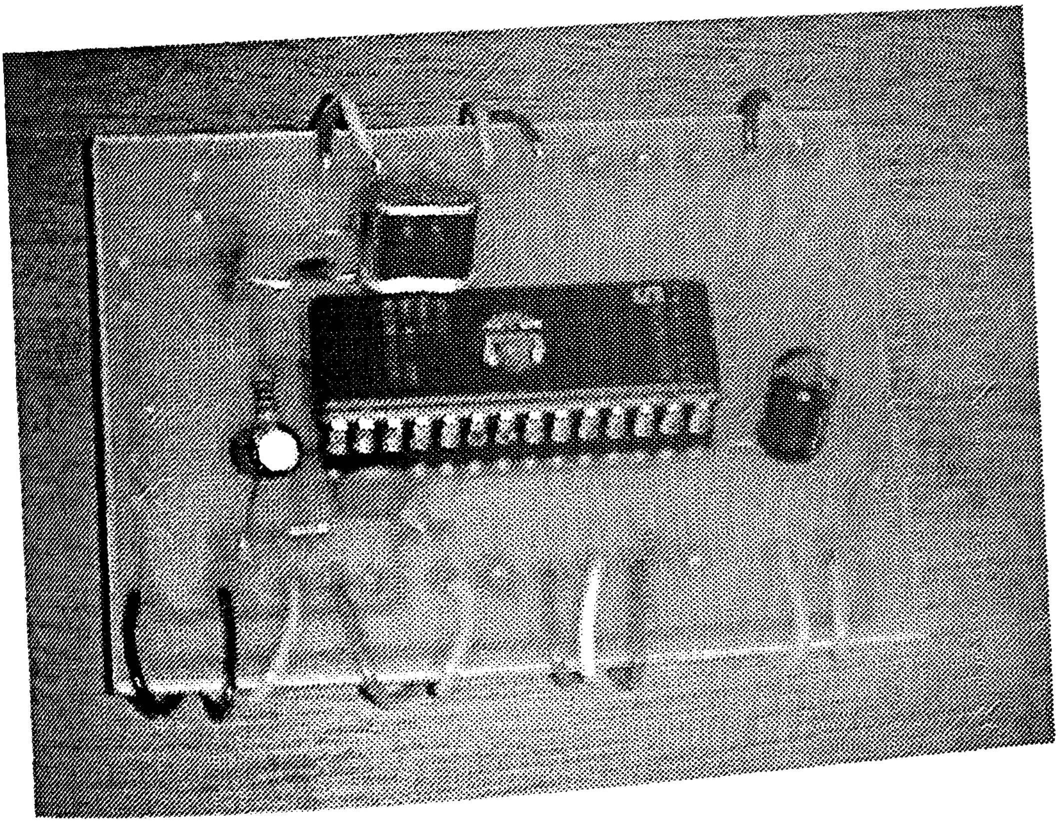


Figure F2. The ATMEL 90AT8515 prototype system board.

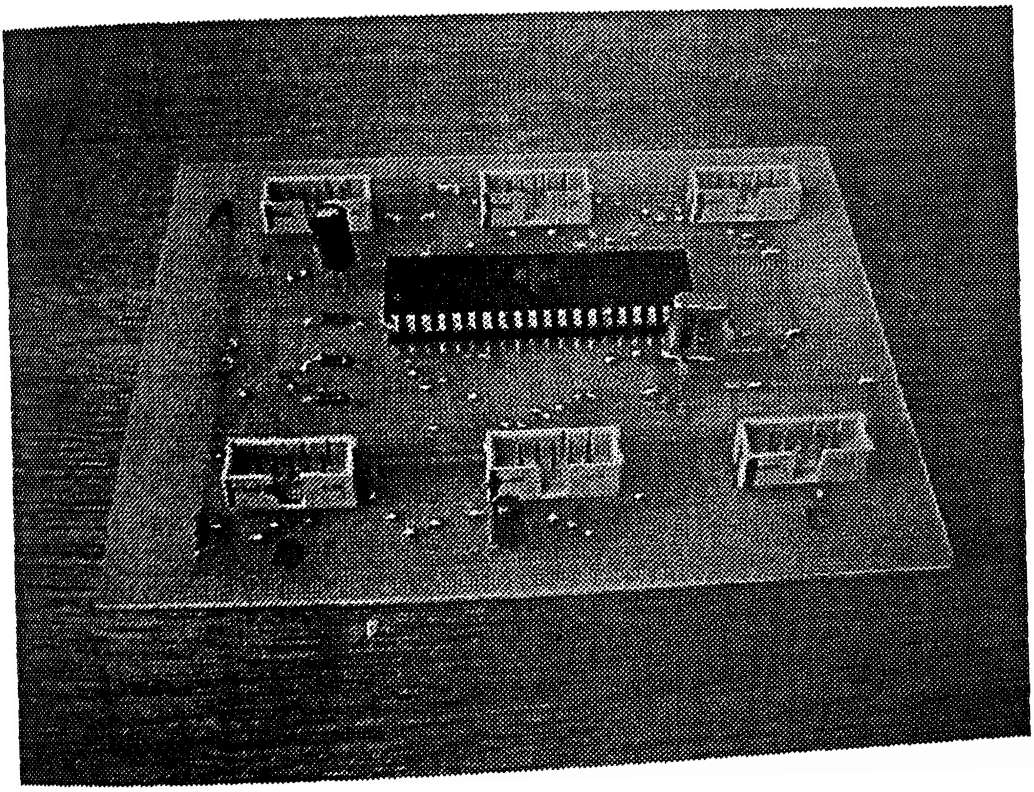


Figure F3. The new transmitter and receiver boards.

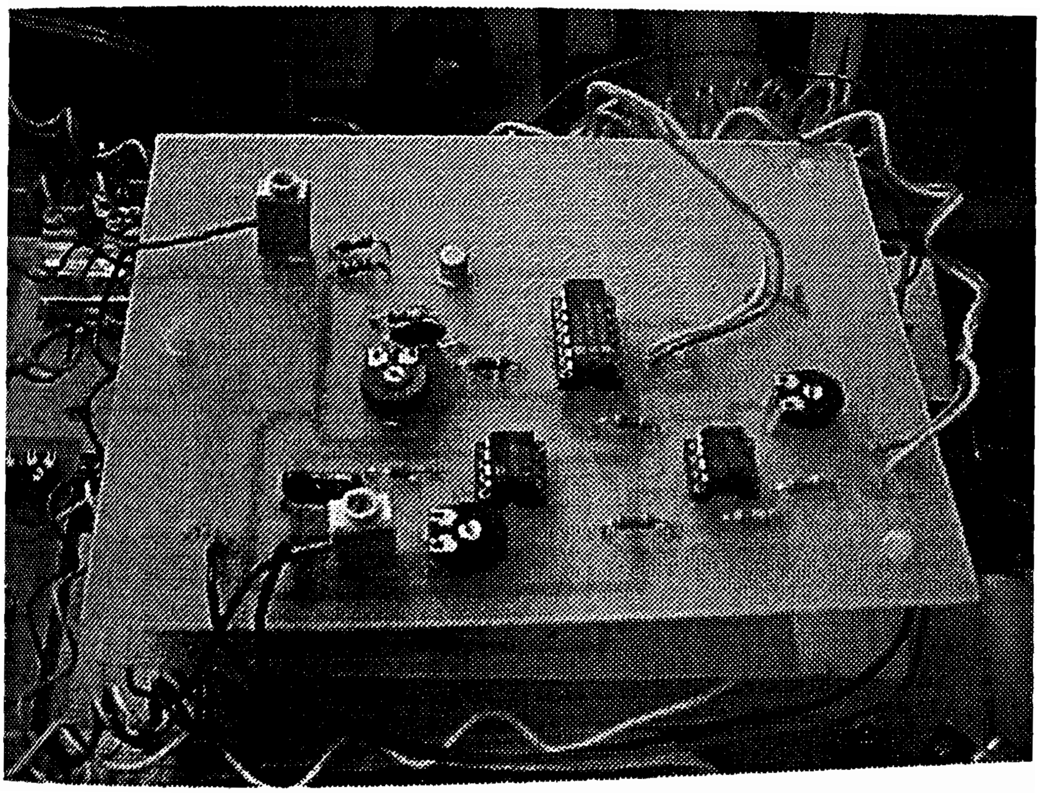


Figure F4. The new prototyp PCB for the MAX1112 analogue to digital converter.

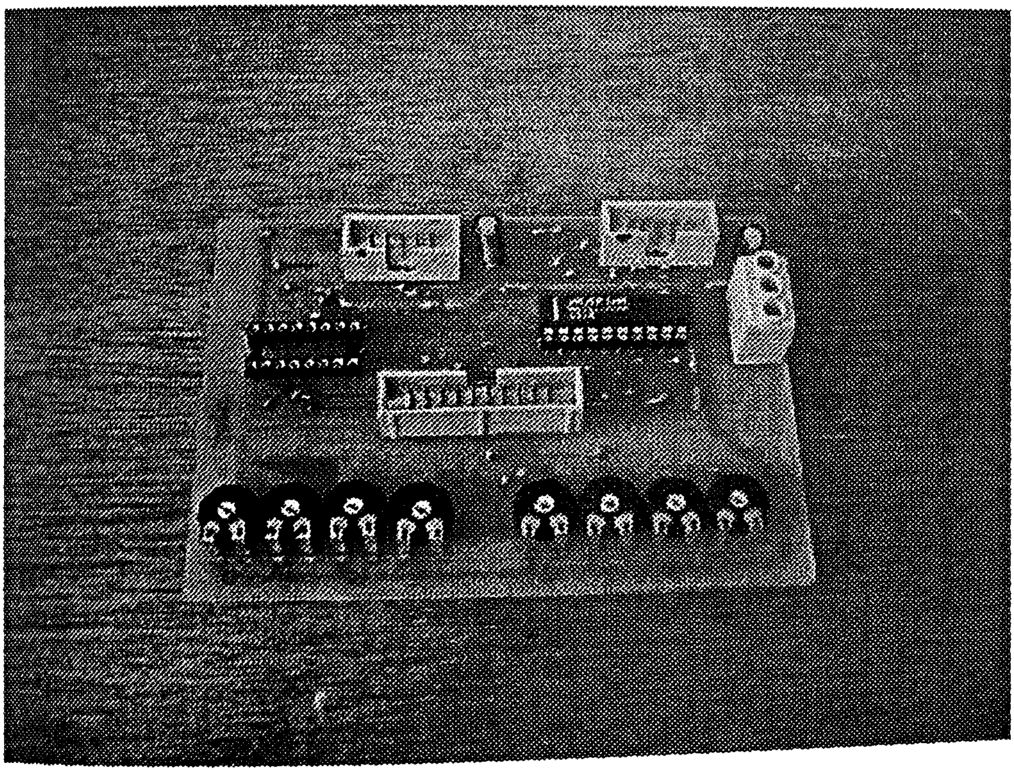
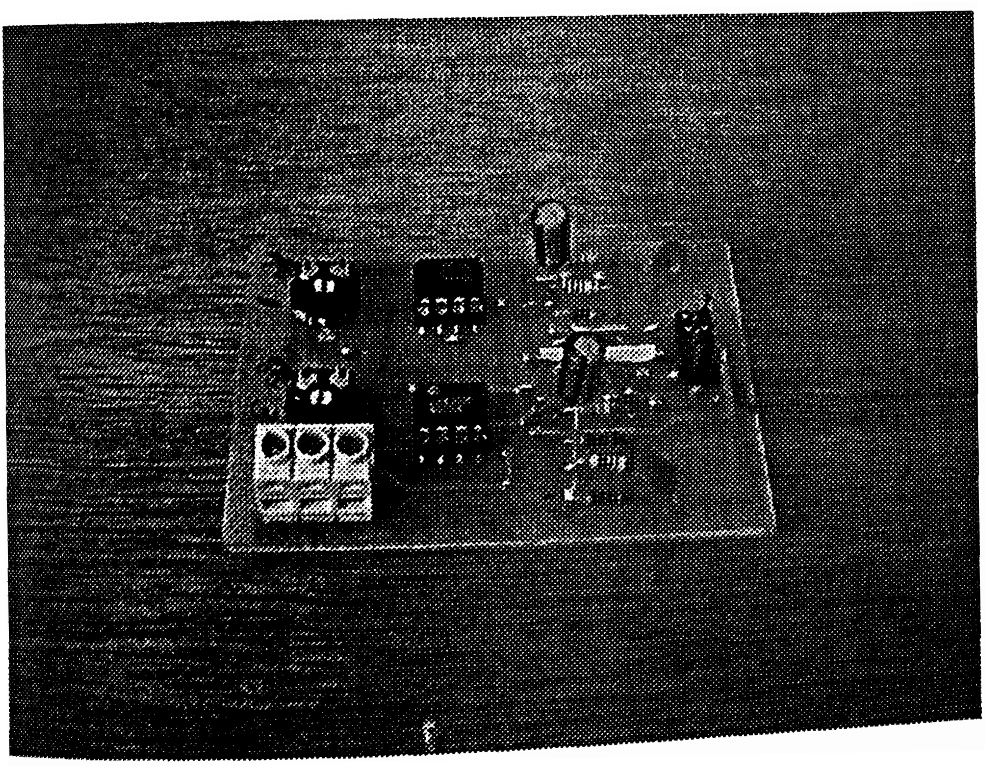
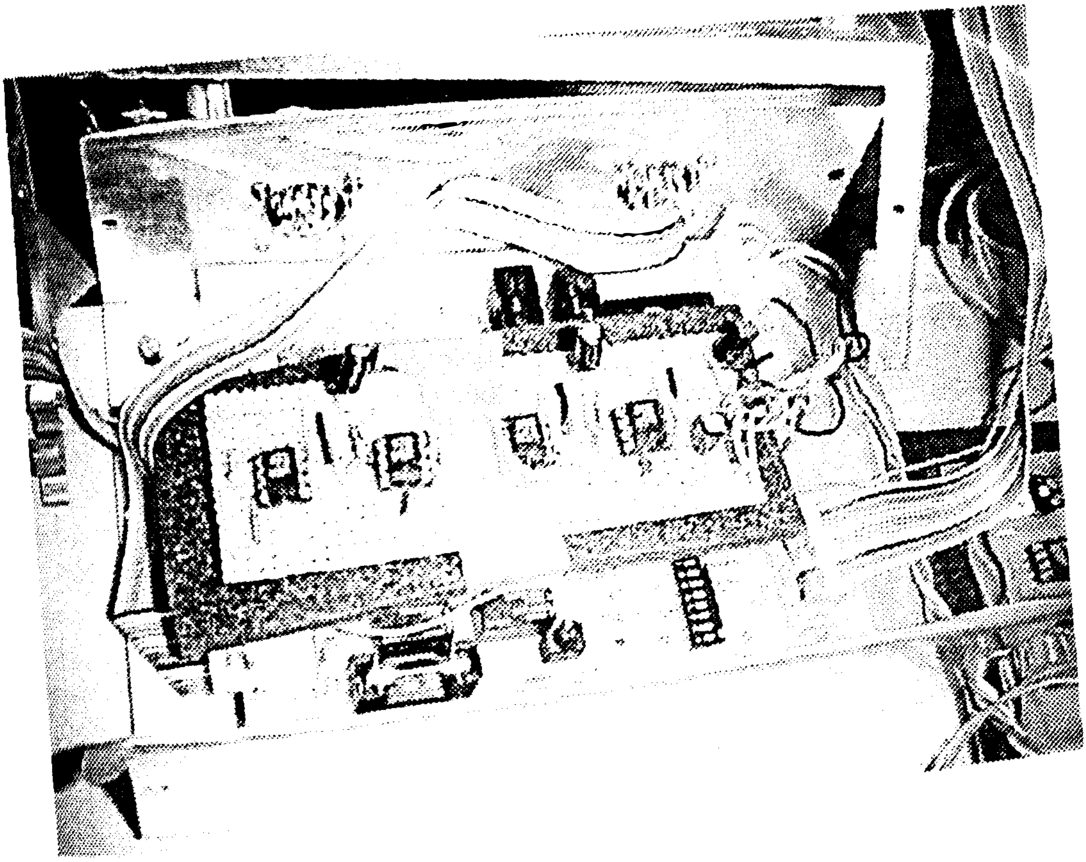


Figure F5. The new interface created to connect the microcontroller pulse width modulation (PWM) circuits to the wheelchair interface.



igure F6. An interface that had been used for other work was modified and used as a ardware mixer for data from the micro-controller and the joystick.



REFERENCES

ADDIS TR (1996), "Alien Intelligence", Inaugural Lecture, University of Portsmouth, 21 November 1996.

ALEXANDER J.H., FRIELLING M.J., SHULMAN S.J., STALEY J.L.,REHFUSS S. & MESSICK S.L. (1986). "Knowledge Level Engineering: Ontological Analysis", Proc. Nat. Comf. Artif. Intel., pp963-968.

APPLETON E. & WILLIAMS D.J. (1987). "Industrial Robot Applications". Open University Press Robotics Series. ISBN 0 335 1406 9.

ARCISZEWSKI T. & ROSSMAN L. (1992). "Knowledge Acquisition in Civil Engineering", ISBN: 0872628647.

BALDING N.W. (1987) "Real Time Path Planning for Robot Arms", PhD Thesis Durham University.

BEATTIE (1993) "The SENARIO (Sensor Aided Intelligent Wheelchair Navigation) Project (TIDE Project 1047", IEE Colloquium on New Developments in Electric Vehicles for Disabled Persons, (Professional Group P6) , Digest Paper No 1995/055, IEE, London, 17th. March 1995, pp 2/1 – 2/4

BEATTIE (1995). "The SENARIO (sensor aided intelligent wheelchair navigation) project (tide project 1047)", digest paper no 1995/055, IEE Colloquium On New Developments In Electric Vehicles For Disabled Persons". Professional Group P6 IEE, London, 17th March 1995, pp 2/1 - 2/4.

BEER J. (1994) "Autonomous climbing robots at the University of Portsmouth" Electrotechnology. vol.5, no.4; Aug.-Sept. 1994; p.6-9

BELL D., LEVINE S., KOREN Y., JAROS L. & BORENSTEIN J. (1993) " An Identification Technique for Adaptive Shared Control in Human-Machine Systems", Proc. 15th. Intl. Conf. of IEEE Engineering in Medicine & Biology, 1993, pp 1299-1300.

BELL D., BORENSTEIN J., LEVINE S., KOREN Y. & JAROS L. (1994) "An Assistive Navigation System for a Wheelchair based upon Mobile Robot Obstacle Avoidance", IEEE Proc Conf. on Robotics & Automation, 1994, pp 2018-2022

BEVAN N.R., COLLIE A.A., WHITE T.S. & LUK BL (1994). "Automation and Robotics in Construction XI" Proceedings of the 11th International Symposium on Automation and Robotics in Construction (ISARC). Elsevier, Amsterdam, Netherlands; 1994; xiv+716 pp.p.609-16

BORENSTIEN J. & KOREN Y. (1991). "Histogramic in motion mapping for mobile robot obstacle avoidance". IEEE tran on robotics and Automation, vol. 7. No 4. pp 535-539.

BOSTEL A.J. & SAGARM V.K. (1996) "Dynamic Control Systems for AGV's", Computing & Control Engineering Journal, August 1996, Vol. 7, No. 4, pp 169-176.

BOURHIS G., MOUMEN K., PINO R., ROHMER S. & PRUSKI A. (1993), "Assisted Navigation for a Powered Wheelchair", Proc. IEEE Intl. Conf. on Systems, Man & Cybernetics, 1993, pp 553=558.

BOURHIS G. & PINO P.(1996), "Mobile Robotics and Mobility Assistance for People with Motor Impairments: Rational Justification for the VAHM Project", IEEE Trans. on Rehabilitation Engineering, Vol.4 No.1, pp 7-12.

BRIENZA D.M. & ANGELO J. (1996). "A Force Feedback Joystick And Control Algorithm For Wheelchair Obstacle Avoidance". Disability And Rehab.Vol.18, No.3, pp123-129.

BROOKS R.A. (1983)(a) "Find Path for a Puma-Class Robot", Proc AAAI, Washington, USA, pp40-44.

BROOKS R.A. (1983)(b) "Solving the Find path Problem by Good Representation of Free Space", Proc of IEEE, Trans on Systems, Man and Cybernetics, Vol SMC-13, pp 190-197

BROOKS R.A. (1983)(c) "Planning Collision Free Motions for Pick and Place Operations", Int Journal of Robotics Research, Vol 2, No 4, pp 19-44.

BROOKS R. (1986) "A Robust Layered Control System for a Mobile Robot", IEEE Journal of Robotics & Automation, Vol. 2, No. 1, 1986, pp 14-23

BROOKS R.A. (1991a). "Intelligence Without Reason". Proc. of Int. Joint Conf. On Artificial Intelligence. (1991). pp569-595.

BROOKS R. (1991b) "Intelligence without Reason", Proc. 12th. UCAL Sydney,Australia 1991, pp569-591)

BROOKS R.R & IYENGAR S.S. (1997). "Real-time distribut d sensor fusion for time-critical sensor readings". Optical Engineering, Vol. 36. No. 3. March 1997. pp767-779.

BROOKS & LOZANO-PEREZ (1983) "A Subdivision Algorithm in Configuration Space for FindPath with rotation", Proc of Int Joint Conf AI, Karlsruhe (Germany), pp 799-806.

BUCHANNON B.G., BARSTOW D., BECHTEL R., BENNETT J., CLANCEY W., KULIKOWSKI C., MITCHELL T.M. & WATERMAN D.A. (1083). "Constructing an Expert System", Building Expert Systems, Addison Wesley, Reading MA.

BUCHBERGER M., JORG K. & PUTTKAMER E. (1993) "Laser Radar & Sonar Based World Modelling & Motion Control for Fast Obstacle Avoidance of the Autonomous Mobile Robot MOBOT-IV" Proc. IEEE Int. Conf. Rob. Autom. Atlanta, GA, 1993 pp 534 -540

BURROWS M.S., COLLIE A.A. & CURRY A. (1991) "Inspection of a Nuclear Reaction Pressure Vesel by vacuum attached and magnetically attached mobile robots". 4th Int. Symp. oln Offsjore, Robotics and artificial intelligance, Marseille, France, 11 – 12 Dec 1991.

BUTLER (1984). "Motorized wheelchair driving by disabled children". Archives of physical medicine and rehabilitation. 1984. Part 65 pp 95 - 97.

BUTLER (1986). "Effects of powered mobility on self - initiated behaviors of very young children with locomotor disability" Proceedings of conf on Developmental medicine and Child Neurol. 1986 Part 28 pp325 - 332.

CALL CENTRE (1993 a). "Evolution and use of CALL Centrer Smart Wheelchairs 1987-1993", Call Centre, University of Edinburgh, 1993, pp1-10.

CALL CENTRE (1993 b). "An Introduction to the Call Smart Wheelchair", Call Centre, University of Edinburgh, 1993, pp1-15.

CANNY J. (1985). "A Voronoi Method for the Piano-Movers problem", Proc of IEEE Int Conf on Robotics and Automation, St Luis, USA, pp 530-535.

CANNY J. (1986). "Collision Detection for Moving Polyhedra", IEEE Trans on Pattern Analysis and Machine Intelligence, Vol PAMI-8, No 2.

CANNY J. (1987). "Exact Solution of Some Robot Motion Planning problems", AI Lab MIT, Computer Sci Div, Univ Cal, USA pp 505-513.

CARROLL J.M., (1997). "Human-computer interaction: psychology as a science of design". Int. Jnl. Human-Computer studies (1997) 46, pp501 – 522.

CHANG C.C. & SONG K. (1996). "Sensor-Based Motion Planning of a Mobile Robot in a Dynamic Environment". Proc of the IEEE IECON Int. Conf. On Ind. Equipment.1996. pp766-771.

CHASE J. & BAILEY D.M. (1990). "Evaluating the Potential for Powered Mobility". The American Jnl. Of Occupational Therapy. Vol. 44. No. 12. December 1990. pp1125-1129.

CHUNRUI-ZHANG & SIYAL M.Y. (2000). "A new segmentation technique for classification of moving vehicles". IEEE 51st Vehicular Technology Conference Proceedings, Piscataway, NJ, USA; pp. p.323-6 vol.1.

CLIFF D, HARVEY I AND HUSBANDS P (1995), "Evolutionary Robotics", IEE Colloqium on the Design and Development of Autonomous Agents, p1/1-1/3.

COLLIE A.A., COOKE D.S., HEWER N.D., LOCKE D.A. & LUK B.L. (1994) "PIROV-Pipe Insertion Remotely Operated Vehicle for inspecting nuclear reactor internals" Proceedings EURISCON '94. European Robotics and Intelligent Systems Conference. Univ. Bristol, Bristol, UK; 1994; 3 vol. xx+1828 pp.p.1156-61 vol.3

COLLINS T.J. & KAUZLARICH J.J. (1998); "Directional Instability of Rear Caster Wheelchairs", Journal of Rehabilitation Research & Devt.,Vol. 25, No. 3, 1998, pp1 – 18

COOKE D. (1999) "Tractive mechanisms for wall climbing robots" PhD, Uni of Portsmouth.

COPELAND J. (1993). "Artificial Intelligence: a philosophical view". Blackwell Publishers, ISBN 0-631-18385-X.

CORKILL D.D , LESSER V & HUDLICKA E. (1982). "Unifying Data Directed and Goal Directed Control: An Example and Experiments". Proc. AAAI-82 , pp143-147.

COX E (1987), "Fuzzy Systems Handbook", Kluwer Academic Publishers, ISBN 0-89838-149-5.

CRAIG I., NISBET P., ODOR P. & WATSON M. (1995). "Tools for Living: Design principles for Rehabilitation Technology". CALL Centre, University of Edinburgh, 4, Buccleuch Place, Scotland. 1995.

CRANE J.M. (1995). "PG8 Series Digital Powerchair Controller". Proc of 1st Symposium on Powered Vehicles for Disable Persons. Uni of Portsmouth Aug 95. Pp 22-34.

DAS Y., RUSSELL K., KIRCANSKI N. & GOLDENBERG A.A. (1999). "An articulated robotic scanner for mine detection: a novel approach to vehicle mounted systems". Proceedings of the SPIE. The-International-Society-for-Optical-Engineering. vol.3710, pt.1-2; pp.1396-404.

DE PENNINGTON et al (1983). "Geometric Modelling: A contribution towards intelligent robots", Proc 13th Int Symp on Industrial Robots, April 17th - 21st, Chicago. pp 7.1 - 7.15.

DENEB (1999) . "IGRIP". A design evaluation and off-line programming tool for robotic work cells. Product information can be found on-line at - http://www.deneb.com/

DEVITO F.A. (1996). "Intelligent wheelchair/nursing home system". Proc of the Int. Symp. on automotiv. tech. and automation (29th). Florence, Italy, Automotive Automation Ltd. June 1996, pp279-285

DILLON B. (1995). "A general-purpose Multiple Master Multiple Slave intelligent interface for the rehabilitation environment." Proc of the 1st Symp on Powered Vehicles for Disabled Persons. University of Portsmouth. ISBN 0-9527524-0-9. pp35-42

DODD G. & ROSSOL L. (1979). "Computer vision and sensor-based robots". Plenum Press. ISBN 0 306 40305 6.

DONALD B.R. (1984). "Motion Planning with Six Degrees of Freedom", MIT AI Lab, Tech Report No 791.

DONALD B.R. (1987). "A Search Algorithm for Motion Planning with six degrees of Freedom", Artificial Intelligence No 31, pp 295-351.

DOUGLAS J. & RYAN M., (1987). "A pre-school severely disabled boy and his powered wheelchair: a case study". Child: care, health and development, 1987, 13, 303 – 309

DUBINS L.E. (1957). "On curves of Minimal Length with Prescribed Initial and Terminal Positions and Tangents", American Journal of Mathematics, Vol. 79, pp 497-516.

EDLINGER T. & PUTTKAMER E. (1994) "Exploration of an Indoor Environment by an Autonomous Mobile Robot", Proc. IEEE/RSJ Int. Conf. Intelligent. Robotic. Systems., Munich, Germany 1994, pp 1278 - 1284

ELFES E. (1987) "Sonar Based Real-World Mapping and Navigation", Autonomous Robot Vehicles, Springer-Verlag, 1990, pp. 233 – 249

ENGLEMORE R.& TERRY A. (1979). "Structure and Function of the CRYSALIS System" Proc IJKAI, pp250-256.

ERMAN L., HAYES-ROTH F., LESSER V. & REDDY D. (1980). "The HEARSAAY II Speech Understanding System: Integrating Knowledge to resolve Uncertainty". Computing Surveys, Vol.12, No 2, pp213-253.

FEIGENBAUM E.A. (1977). "The Art of Artificial Intelligence: Themes and Case Studies of Knowledge Engineering". Proc.5th Int Joint Conf. Artif. Intel. pp1014-1029.

FREDERICK J.R. (1965). "Ultrasonic Engineering". Published by J Wiley & Sons Inc. Cat card No. 65-14257. USA.

FOREMAN, N. (1993). "In a virtual world". Special Children, 1993 March , 64, pp 28-29.

FRANKLIN J.A & SELFRIDGE O.G (1990) "The perceiving robot: What does it see? What does it do?". Proceedings. 5th IEEE International Symposium on Intelligent Control 1990 (Cat. No.90TH0333-5). IEEE Comput. Soc. Press, Los Alamitos, CA, USA; 1990; 2 vol. xxii+1297 pp.p.146-51 vol.1

FU K.S., GONZALES R.C. & LEE C.S.G. (1987). "Robotics: Control, Sensing, Vision and Intelligence. ISBN 0 07 100421 1.

GALER M. (1995). "Driver Vehicle Interaction and Human Factors". Smart Vehicles: J.P. Pauuweleussen, H.B. Pacejka (eds.) pp381-387.

GALT S. (1999). "Soft Computing Based Motion Control For An Eight-Legged Robot". PhD. Uni of Portsmouth.

GARIBOTTO G. (1999). "Alerting mechanisms in attentive vision systems". Human and Machine Perception 2. Emergence, Attention, and Creativity. Kluwer Academic, Dordrecht, Netherlands; 1999; pp. p.99-105.

GILBERT & JOHNSON (1985). "Distance functions and their application to robot path planning in the presence of obstacles". IEEE Journal of Robotics and Automation, Vol RA-1, No 1, pp 21-30.

GOMI T. & IDE K.,(1996) "The Development of an Intelligent Wheelchair", Proc.of IEEE Intelligent Vehicles Symposium , 1996, pp 71-75

GOODWIN M.J. (1992) (M.Phil) "A Flexible Manufacturing System for Education", University of Portsmouth, UK

GOODWIN M.J. (1997a) "A Data Fusion Algorithm for Wheelchair Navigation", Proc. ICORR'97 Intl. Conf on Rehabilitation Robotics, April 1997, pp.131-134, ISBN 1-85790-034-0

GOODWIN M.J.(1997b) "Navigational Assistance for Disabled Wheelchair Users", Journal of Systems Architecture, Vol.43, pp.73-79.

GOODWIN M.J. (1998) "Navigational Assistance Using Short Term Memory", Proc. 29th. Intl. Symposium on Robotics (ISR98) NEC Birmingham, pp. 149-151.

GOODWIN M.J. & SANDERS D.A. (2000). "New Intelligent Systems for a Bobcat II wheelchair". Proc of 3rd Symp on Powered vehs for Disabled Persons. pp46-49. ISSN: 1472-7633.

GOUZENES L. (1984). "Strategies for Solving Collision-Free Trajectories Problems for mobile and manipulator robots", Int Jnl of Robotics Research, No 3.4, pp 51-65.

GRAY J.O. (1996). "Recent Developments in Advanced Robotics and Intelligent Systems", IEE Computing and Control Engineering Journal, December 1996, p267-76.

GUEY J.C., BELL M.R. & COFFEY J.T. (1997). "An Information-theoretic Approach to the Design of a Distributed Cascade of Sensors". Jnl. of the Franklin Inst, Vol. 33B, No. 4, pp 707-736

GUNDERSEN R.W., SMITH S.J. & ABBOTT B.A. (1996). "Applications of virtual reality technology to wheelchair remote steering systems" Proc of the 1st Euro. Conf on Disability, Virtual Reality and Ass Technologies. U.K. ISBN 0 7049 1140 X. pp 47-56.

GUTMANN J.S. & SCHLEGEL C., (1996). "AMOS: Comparison of Scan Matching Approaches for Self-Localization in Indoor Environments, 1st Euromicro Workshop on Advanced Mobile Robots (EUROBOT '96). IEEE Comput. Soc. Press, Los Alamitos, CA, USA; pp. p.61-7.

HAMMOND R.A., NEWMAN D.S. & YEH Y.C. (1989). "On Fly-By-Wire Control System And Statisticasl Analysis Of System Performance".

HARTLEY J. (1987). "guided vehcles are moving with the times". The FMS Magazine, Vol 5(3), July 1987, pp126 - 128.

HAUPT RL AND HAUPT SE (1998). "Practical Genetic Algorithms", John Wiley and Sons, ISBN 0-471-18873-5.

HAVLIK S., (1998) " Robotic Systems for Humanitarian De-mining Operations" ", Proc. 29th. Intl. Symp. on Robotics, April 1998, NEC, Birmingham ,UK..

HEWER N.D., SANDERS D.A. & TEWKESBURY G.E. (1994). "Control of a robot using bounded, smoothed, incremental and stepping variable structure control" Proceedings EURISCON '94. European Robotics and Intelligent Systems Conference. Univ. Bristol, Bristol, UK; 1994; 3 vol. xx+1828 pp.p.1131-47 vol.2

HEWER N.D. (1995) "Aspects of Design for Inspection Robots in a Nuclear Environment". Mhil. University of Portsmouth.

HINKEL R., KNIERIEMAN T. & PUTTKAMER V. (1988). "MOBOT II - An autonomous mobile robot for indoor applications". 19th Int. Symp. and Exhibition on Robots. 1988 Sydney, Australia. pp 491 - 504.

HOLVE R. & P. PROTZEL. (1996) "Reverse Parking of a Model Car with Fuzzy Control" Proc. 4th Euro Congress on Intelligent Techniques and Soft Computing -EUFIT'96, Aachen, Germany, Sept. 1996, pp. 2171-2175.

HUDSON A.D.(1997) "A New Blackboard Framework Applied to a HRAR Desifn Program" PhD. University of Portsmouth.

HWANG K.Y (1988)(PhD) "Robot Path Planning using Potential Field Representation", University of Illinois, USA.

IAR SYSTEMS (1999). "IAR Systems Web site". Information on the IAR Tools and products can be found at http://www.iar.com/.

INMAN D.P., LOGE K. & LEAVENS J. (1997). "VR Education and Rehabilitation". Communications of the ACM. August 1997/Vol40, No8, pp 53-58.

JACKSON P. (1990). "Introduction to Expert Systems", 2nd Edition. Addison-Wesley, Wokingham. U.K.

JIMINEZ A.R., CERES R. & PONS J.L. (2000) "A vision system based on a laser range-finder applied to robotic fruit harvesting". Machine-Vision-and-Applications. Vol.11, no.6. p.321-9.

JORG K.W. & BERG M. (1996). "First results in eliminating crosstalk & noise by applying pseudo-random sequences to mobile robot sonar sensing". IEEE proceedings of EUROBOT 96, 1996. pp 40 - 45.

JOHNSON B.W., & AYLOR J.H. (1985). "Dynamic modelling of an electric wheelchair". IEEE Trans on industry applications. Vol. IA-21, No5, Sept/Oct 1985, pp1284-1293

KAM M., ZHU X. & KALATA P. (1997) "Sensor fusion for Mobile Robot Navigation". Proc. of the IEEE, Vol. 85, No. 1, Jan 1997, pp 108-119.

KANTROWITZ, (1995). "FAQ: Expert System Shells." http:/www.cs.cmu.edu:8001/web/groups/AI/html/faqs/ai/expert/part1/faq.html.

KAY L.(1999A). "Sonar-now a viable alternative to vision". Sensor Review , Published by MCB University Press. Vol.19, No.3. pp178-179.

KAY L. (1999B). "ACTFM acoustic spatial sensing technology: its use by blind persons and robots". Sensor Review, Published by MCB University Press. Vol. 19, No 3, 1999, pp. 195-201.

KEZHONG H., HAIHANG S., MUHE G. & HONG W. (1996). "Research of Intelligent Mobile Robot Key Techniques". Proc of the IEEE Int. Conf. on Ind. Tech., 1996, pp503-507.

KHATIB O. (1986) "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", Intl. Journal of Robotic

Research, Vol. 5, No. 1, pp 90-98

KLEEMAN L. (1999). "Real-time mobile robot sonar with interference rejection". Sensor Review. MCB Press, Vol 19, No 3. p214-221.

KORZENIOWSKI K.A. (1996). "Using supervised learning to fuse sensor data for surface tracking in complex environments". Proc. Of SPIE. Vol. 2911. 1996. pp2-8

KWITOWSKI A.J., BRAUTIGAM A.L. & MONAGHAN W.D. (1995) "Teleoperated continuous mining machine for

KUMAR V.R. & WALDRON K.J. (1988) "Force distribution in closed kinematic chains". IEEE-Journal-of-Robotics-and-Automation. vol.4, no.6; Dec. 1988; p.657-64

KUMAR V. & WALDRON K.J. (1989) "Suboptimal algorithms for force distribution in multifingered grippers". IEEE-Transactions-on-Robotics-and-Automation. vol.5, no.4; Aug. 1989; p.491-8

LANGNER M.C. (1995) "The Development of Special Mobility Systems at Chailey Heritage", Proc. 1st. Symp.on Powered Wheelchairs, Univ..of Portsmouth, Aug. 1995, pp12-17, ISBN 0-9527524-0-9

LANGNER M.C. (2000). "Mobile and Interactive Equipment at Chailey Heritage School", Proc. 3rd symp. Powered Vehs for the Disabled. pp35-37. ISSN: 1472-7633.

LEARMOUNT D. (1991). "Fly-By-Wire: The Verdict". Electronics World and Wireless World. July 1991. pp593-595

LEE J.D. (1994). "Trust self-confidence, and operators' adaptation to automation." Int journal Human-Computer Studies. Vol 40. pp 153-184.

LEE J.D. & MORAY M. (1992). "Trust, control strategies and allocation of function in human-machine systems". Ergonomics. Vol. 35, No 10, pp 1243-1270.

LESSER D.D., FENNELL R., ERMAN L. & RAJ-REDDY D. (1975). "Organisation of th HEARSAY II Speech Understanding System". IEEE trans Acoustics, Speech and Sig Processing. ASSP-23, pp 11-24.

LESSER D.D.& CORKILL V. (1983). "The Distrubuted Vehicle \Monitoring Testbed: A Tool For Investigating Distributed Problem Solving Networks". Blackboard Systems, Addison-Wesley.

LESSER V,. CORKILL D.D.. WHITEHAIR R. & HERNANDEZ J. (1989). "Focus of Control Through Goal Relationships". Black board Arch and Apps. Academic press.

LEVINE P., BELL D., & KOREN Y. (1994) "NavChair: An example of a Shared Control System for Assistive Technologies", IEEE Transactions on Computers, Vol. C-32, No 2, pp108-120.

LINDSTEDT G. (1996). "Recognition using frequency sonar sweep". Proc. Eurobot 96, IEEE Computer Society, pp 46-53. ISBN 0-8186-7695-7.

LITTLE C. (1997). The Intelligent Vehicle Initiative. "Advancing Human-Centered Smart Vehicles". Public Roads. 1997, 61/2 (Sept./Oct.), pp18-25.

LOZANO-PEREZ T. & WESLEY (1979) "An Algorithm for Planning Collision Free Paths among Polyhedral Obstacles", Communications of the ACM, Vol. 22, No. 10, pp 560-570..

LOZANO-PEREZ T. (1990) "Autonomous Robot Vehicles", Springer-Verlag, ISBN 0-387-97240-4, pp. vii - xi

LUK et al. (1988). "Stochastic Force Sensing applications in Robotics". Proc of Euromicro 88, Zurich, Pub by N. Holland, Sept 88, pp 419-423.

LUK B.L., COLLIE A.A. & BILLINGSLEY J. (1991) "Robug II: An intelligent wall climbing robot" Proceedings. 1991 IEEE International Conference on Robotics and Automation (Cat. No.91CH2969-4). IEEE Comput. Soc. Press, Los Alamitos, CA, USA; 1991; 3 vol. xxxix+2843 pp.p.2342-7 vol.3

LUK B.L., COOKE-DS; COLLIE-AA; WHITE-TS (1994) "Robug III-a semi-intelligent teleoperated walking and climbing robot for disordered hazardous environments" Proc EURISCON '94. European Robotics and Intelligent Systems Conf. Univ. Bristol, Bristol, UK; 3 vol. xx+1828 pp. p.1162-8 vol.3.

MACHINE DESIGN. (1992) "Fly-by-wire just the beginning". New trends. Machine Design Magazine. Jan. 23, 1992. pp12-15.

MADHANI A. & DUBOWSKY S. (1992). "Motion planning of mobile multi-limb robotic systems subject to force and friction constraints" Proceedings. 1992 IEEE International Conference on Robotics And Automation (Cat. No.92CH3140-1). IEEE Comput. Soc. Press, Los Alamitos, CA, USA; 1992; 3 vol. xxxix+2819 pp.p.233-9 vol.1

MATIJEVIC. J.. (1996). "Mars Pathfinder Microrover - Implementing a Low Cost Planetary Mission Experiment", Proceedings of the Second IAA International Conference on Low-Cost Planetary Missions, John Hopkins University Applied Physics Laboratory, Laurel, Maryland, April 1996.

McCLOY D. & HARRIS D M (1986). "Robotics: An introduction". Open University Press robotics Series. ISBN 0 335 154302 6.

McILVAINE PARSONS H (1996) "A Visionary Project for Drivers". Ergonomics in Design,Jan. 1996, pp11-14.

McKERROW P.J. (1991). "Introduction to Robotics". Addison-Wesley Publishing Company. ISBN 0 201 18240 8.

MCKERROW PJ & HARPER NL.(1999). "Recognising leafy plants with in-air sonar". Sensor Review, Published by MCB University Press. Vol.19, No.3. pp202-206.

MERIDIAN (1995). "Good For You", a health magazine programme. Available for view from the University of Portsmouth video archives, Ref. V404.

MOOREHEAD S., SIMMONS R., APOSTOLOPOULOS D., AND WHITTAKER W.L. (1999) "Autonomous Navigation Field Results of a Planetary Analog Robot in Antarctica," International ymposium on Artificial Intelligence, Robotics and Automation in Space, June, 1999.

MORAVEC H.P. (1983) "The Stanford Cart and the CMU Rover", Proc. IEEE, Vol. 71 No.7, pp.872-884.

NELSON P.J., VERBURG G., GIBNEY D. & KORBA L. (1990). "The Smart Wheelchair: A Discussion of the Promises and Pitfalls". Proc. Of RESNA 13th An. Conf. Washington D.C. U.S.A. (1990). pp307-308.

MORRISON, J. AND NGUYEN, T., "On-Board Software for the Mars Pathfinder Microrover", Proceedings of the Second IAA International Conference on Low-Cost Planetary Missions, John Hopkins University Applied Physics Laboratory, Laurel, Maryland, April 1996.

MOSHE K., XIAOXUN Z. & KALATA P. (1997). "Sensor Fusion for Mobile Robot Navigation". Proc. Of the IEEE. Vol. 85. No. 1. January 1997. pp 108-119.

MURPHY R.R., GOMES K. & HERSHBERGER D. (1996) "Ultrasonic data as a function of robot velocity". Proc of the SPIE. Int. soc. for Optical Engineering. Vol. 2905. 1996. pp 114 - 126.

MUTAMBARA A.G.O. (1996). "Non-linear sensor fusion for a mobile robot". Proc of the SPIE. Int. Soc. for Optical Engineering. Vol. 2905. 1996. pp 102 - 113.

MYKLEBUST G. (1997). "Parallel design of a RISC core and its C compiler". Electronic-Product-Design. vol.18, no.4; April 1997; p.21-2, 25

NASA (1998). "Advanced Subsonic Technology Project. Fly-By-Light/Power-By-Wire Element". http://www.lerc.nasa.gov/Other_Groups/AST/fblpbw.htm

NASA FACTS. (1998). "F-8 Digital Fly-By-Wire". Internet http//www.dfrf.nasa.gov/EAO/FactSheets/F_8DFBWFACTS.html.

NEALE I.M. (1988). "First Generation Expert Systems: A Review of Knowledge Acquisition Methodologies", Knowledge Eng. Rev. Vol.3, No.2, pp105-145.

NELSON P.J., VERBURG G., GIBNEY D. & KORBA L. (1990). "The Smart Wheelchair: A Discussion of the Promises and Pitfalls". Proc of RESNA 13th An Conf. Washington DC USA pp307-308.

NEWELL A., (1982). "The Knowledge Level". Artif Intel. Vol.18, pp87-127.

NII H.P., FEIGENBAUM E., ANTON J. & ROCKMORE A.J. (1982). "Signal to Symbol Transformation: HASP/SIAP Case Study", AI Magazine, Vol.3, No.1, pp23-35.

NII H.P. (1986a). "Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures". AI Magazine, Vol.7, No.2, pp38-53.

NII H.P. (1986b). "Blackboarcd Systems: Blackboard Application Systems, Blackboard Systems form a Knowledge Engineering Perspective"" AI Magazine, Vol.7, No.3, pp82-106.

NISBET P.D., ODOR J.P. & LOUDON I.R. (1988). "The CALL Centre smart wheelchair". Proc. of 1st Int. Workshop on Robotic Applications to Healthcare. June 1988. pp 9.1 - 9.10.

NISBET P. (1995) "Learning Through Smart Wheelchairs", Proc. 1st. Symp.on Powered Wheelchairs, University.of Portsmouth, Aug. 1995, pp 46-53, ISBN 0-9527524-0-9

NISBET P. (1996) "Integrating assistive technologies: current practices and future possibilities". Medical-Engineering-&-Physics. vol.18, no.3; April 1996; p.193-202

NISBET P. (2000) "The Smart Wheelchair". Proc. 3rd symp. Powered Vehs for the Disabled, pp50-54. ISSN: 1472-7633.

ORIN D.E. & OH S.Y. (1981) "Control of force distribution in robotic mechanisms containing closed kinematic chains". Transactions-of-the-ASME.-Journal-of-Dynamic-Systems,-Measurement-and-Control. vol.103, no.2; June 1981; p.134-41

PARSONS H. M. (1996). "A Visionary Project for Drivers". Ergonomics in Design. January 1996. pp11-14.

PARNICHKUN M. & SAMADI V. (1999). "High-accuracy-long-range measurement by automatic multi-mode measurement scheme based laser range finder" IEEE International Symposium on Intelligent Signal Processing and Communication Systems. Signal Processing and Communications Beyond 2000. King Mongkuts Inst. Technol, Bangkok, Thailand. pp. p.411-14.

PG8 Series Digital Powerchair Controller, PG8 Remote Interfaces (Potentiometric). (1995). Engineering Notes, Penny & Giles Drives Technology Ltd. Dorset.

PIEPER D.L 1969(PhD) . "The Kinematics of Manipulators under Computer Control", Stanford (USA).

PRUSKI A. & BOURHIS G (1992). "The VAHM Project: a Cooperation Between an Autonomous Mobile Platform and a Disabled Person". Proc. of the 1992 IEEE Int. Conf on Robotics and Automation. pp 268-273.

PURCELL D.W. & HUISOON J.P. (1993). "A Curved Array Transducer For Autonomous Vehicle Navigation", Proc of the Acoustic Sensing And Imaging Conf. Publication No. 369, IEE 1993. pp32-37.

QED (1995) (Sales Literature) "New Horizons for People with Special Needs", Quest Enabling Designs Ltd.Ability House, 242 Gosport Road, Fareham, Hants., PO16 0SS

RAO S. & KUC R. (1989). "INCH: An intelligent wheelchair prototype", Proc of the 15th North East Bioengineering Conf (1989) USA, pp35 - 36.

RASOL Z. (2001). "A New Automatic System for Simple Spot Welding Tasks". MPhil. University of Portsmouth.

RILEY D. (1990). "Data Abstraction and Structures" Boyd and Frazer.

SANDERS D.A., HERRING D., NAGHDY F. & BILLINGSLEY J. (1987) "Advances in adaptive robot path control using information from servo amplifier currents". Third International Conference on Electrical Machines and Drives (Conf. Publ. No.282). IEE, London, UK; 1987; xi+370 pp.p.300-5

SANDERS DA (1993). "Making complex machinery move: automatic programming and motion planning", a post graduate text published by Research Studies Press ISBN 0-86380-141/2, and by J Wiley & Sons Inc ISBN 0-471-93793/2.

SANDERS D.A. (1993) "Making Complex Machinery Move: Automatic Programming and Path Planning", Research Studies Press ISBN 0-86380-141/1, and J. Wiley ISBN 0-471-93793/2.

SANDERS DA (1998). "Route Planning". Intelligent Mobility,8, published by PCR, pp 44-49, ISBN: 0-9527524-1-7.

SANDERS D.A. & HUDSON A.D. (2000) "A Specific blackboard expert system to simulate and automate the design of high airlift reactors", IMACS, No.53, Elsevier Science B.V. pp41-65.

SANDERS D.A., HUDSON A.D., TEWKESBURY G.E. & MASON S. (2000). "Automating the design of high-recirculation airlift reactors using a blackboard framework". Expert Systems with Applications. 18. Elsevier Science, pp231-245.

SANDERS DA & STOTT IJ (1999) "A new prototype intelligent mobility system to assist powered wheelchair users". Industrial Robot; an international journal, Vol 26, No 6, pp 466 - 475, ISSN: 0143-991X.

SANDERS DA. STOTT IJ AND TEWKESBURY GE. (1998) "Strategies to assist a person in navigating a powered vehicle through a doorway". Proceedings of Advanced Robotics - Beyond 2000, Chapter 11 "Mobile Robots I", NEC Birmingham, pp 19 – 21

SCHILLING R.J. (1990) "Fundamentals of robotics; Analysis and Control". Prentice-Hall International. ISBN 0-13-334376-6.

SEBASTIAN J.M., GARCIA D., SANCHEZ MORENO F.M. & GONZALEZ J.M. (1999). "Inspection system for machine parts with three-dimensional characteristics. Proceedings-of-the-SPIE--The-International-Society-for-Optical-Engineering. vol.3836. pp.236-43.

SHORTLIFFE E.H. (1976). "Computer based Medical Consultations: MYCIN". Elsevier, Ney York.

SONG K. & CHEN C. (1996). "Application of Heuristic Asymmetric Mapping for Mobile Robot Navigation Using Ultrasonic Sensors". Jnl. Of Intelligent and Robotic Systems. Vol. 17. pp 243-264. 1996.

SONG S.M. AND WALDRON K.J. (1989), "Machines that walk: the adaptive suspension vehicle", MIT Press.

STANTON D, WILSON P, & FOREMAN N. (1996). "Using virtual reality environments to aid spatial awareness in disabled children". Proc of 1stEuro. Conf. Disability, Virtual Reality & Assoc. Tech. Maidenhead, UK. ISBN 07049-1140-X. pp 93 - 101.

STAUFFER R.N. (1984) "Robots Assume Key Role in Electronics Manufacture" Manufacturing Engineering Vol. 92 No.4, April 1984, pp 84-87.

STONE, H. W., (1996) "Mars Pathfinder Microrover: A Low-Cost, Low-Power Spacecraft", Proceedings of the 1996 AIAA Forum on Advanced Developments in Space Robotics, Madison, WI, August 1996.

STOTT I.J. (1997) "Computer Assistance for Users of Powered Wheelchairs", M.Phil. Thesis University of Portsmouth.

STOTT IJ (1998). "A review of advanced and intelligent wheelchair research". Intelligent Mobility,8, published by PCR, pp 4-12, ISBN: 0-9527524-1-7.

STOTT IJ & SANDERS DA (1996). "Modification of a Wheelchair Trajectory by the Integration of Ultrasonic Sensor Data". Proc. of 1st Symposium on Powered Wheelchairs for Disabled Persons, Uni of Portsmouth, published by PCR, pp 4-7. ISBN 0-95275240-9

STOTT IJ & SANDERS DA (1998). "Advanced intelligent wheelchairs". Intelligent Mobility, 5, published by PCR, pp 23-31, ISBN: 0-9527524-1-7.

STOTT IJ, SANDERS DA AND GOODWIN MJ. (1997) "A software algorithm for the intelligent mixing of inputs to a tele-operated vehicle". Journal of Systems Architecture, Vol 43, No 1, pp 67 - 72.

STOTT IJ & SANDERS DA, (2000a), ""The use of virtual reality to train powered wheelchair users and test new wheelchair systems" International Journal of Rehabilitation Research, Vol 23 (issue 4), p. 321-326

STOTT I.J & SANDERS D.A. (2000b). "New powered wheelchair system for the rehabilitation of some severely

disabled users". The International Journal Of Rehabilitation Research. Vol. 23(3) ISSN: 0342-5282..

STOTT I.J., SANDERS D.A. & TEWKESBURY G.E. (2000) "Low cost ultrasonic sensors for tele-operatd vehicles". Sensor Review: the international journal of sensing. Vol.20, no.3, pp227-234, ISSN: 0260-2288.

TAYLOR C. (1995). "DX Mobility Systems". Proc of 1st Symposium on Powered Vehicles for Disable Persons. Uni of Portsmouth Aug 95. Pp 54-56.

TESHIGAWARA M; SHIBATA F & TERAMOTO H. (1989) "High resolution (0.2 mm) and fast response (2 ms) range finder for industrial use in air". Proceedings of IEEE Ultrasonics Symposium (Cat. No.89CH2791-2). New York, USA; vol 1. pp.639-42.

TERRY A. (1988). "Using Explicit Strategic Knowledge to Control Expert Systems", Blackboard systems, Addision-Wesley pp159-188.

TEWKESBURY G.E. (1994). "Design using Distributed Intelligence Within Advanced Production Machinery", PhD University of Portsmouth. U.K.

TEWKESBURY G.E. & SANDERS D.A. (1995) "Task programming applied to powered wheelchairs" Jnl. of intelligent mobility 1. PCR pp7-10.

THE FMS MAGAZINE. (1987). "Dexion develops its own AGV". The FMS Magazine, News, April 1987, pp59.

TSENG C.S. (1987)(PhD) "Rapid Generation of Collision-Free paths for Robot Manipulators with simulation", University of Florida, (USA).

UDAPA S.M. (1977) "Collision Detection and Avoidance in Computer Controlled Manipulators", Ph.D. Thesis, California Inst. Of Technology, USA.

VERBURG G., SNELL E., PILKINGTON M. & MILNER M (1984). "Effects of powered mobility on young handicapped children and thier families". Proc. of 2nd Int. Conf. of the rehabilitation engineering Soc. of N. America. June 1984. Ottowa. pp172 - 173.

VIJAYKUMAR R (1987), "Motion Planning for Legged Locomotion Systems on Uneven Terrain", PhD, Ohio State University.

WAKAUMI H., TSUKAGOSHI Y., MIURA T. & YAMAUCHI F (1989). "AGV guidance systems uses magnetic lattice lanes". The FMS Magazine, Vol. 7(1), January 1989, pp35 - 38.

WARWICK K (1997), March of the Machines, Century.

WELLINGA B.J. & BREUKA A.J., (1986) "Models of Expertise". Proc. 7th European Conf. Artif. Intl. pp 306-318.

WIKER S.F. (1993). "The human/robot interface". Aerospace America. Oct 1993. pp30-33.

WILLIAMS T., LOWRANCE J., HANSON A & RISEMAN E (1977) "Model-Building in the VISIONS System", Proc IJCA 1-77, pp645-664.

WOOLHOUSE N.D. (1998). "Virtual Reality Simulation of a Computer Assisted Wheelchair". An undergraduate text, Department of Mech. Eng. Uni of Portsmouth, U.K.

YODER J.D., BAUMGARTNER E.T. & SKARR S B (1996) "Initial Results in the development of a guidance system for a powered wheelchair". IEE Trans. on Rehab.Eng. Vol.3, No. 4. Sept 96.

ZALILA-Z; FATENE-M; KADHI-R. (1998) "Fuzzy supervision in statistical process control". SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.98CH36218). IEEE, New York, NY, USA; vol.1. pp.638-43.

ZALZALA AMS AND MORRIS AS (1996). "Neural Networks for Robotic Control", Ellis Horwood, ISBN 0-13-119892-0.

ZWERN AL & GOODRICH GL. (1996). "Virtual computer monitor for visually-impaired users". Proc of 1st Euro. Conf. Disability, Virtual Reality & Assoc. Tech. Maidenhead, UK. ISBN 07049-1140-X pp 13 - 21.