OMNIDIRECTIONAL IMAGE SENSING FOR AUTOMATED GUIDED VEHICLE

PETRUS JOHANNES SWANEPOEL

Dissertation submitted in fulfilment of the requirements for the degree

MAGISTER TECHNOLOGIAE: ENGINEERING: ELECTRICAL

in the

School of Electrical and Computer Systems Engineering

of the

Faculty of Engineering, Information and Communication Technology

at the

Central University of Technology, Free State

Supervisor: Mr B. J. Kotze

Co-supervisor: Dr H. Vermaak

Bloemfontein

April 2009

DECLARATION

I, PETRUS JOHANNES SWANEPOEL, identity number [redacted], and student number 20235836, do hereby declare that this research project which has been submitted to the Central University of Technology Free State, for the degree MAGISTER TECHNOLOGIAE: ENGINEERING: ELECTRICAL, is my own independent work and complies with the Code of Academic Integrity, as well as other relevant policies, procedures, rules and regulations of the Central University of Technology, Free State, and has not been submitted before by any person in fulfilment (or partial fulfilment) of the requirements for the attainment of any qualification.

..... …….

SIGNATURE OF STUDENT DATE

ACKNOWLEDGEMENTS

I would like to thank the Lord for giving me the opportunity to complete my Master's degree. Thanks to my family for their support and all their prayers while I was completing this task.

I would also like to express my gratitude to the following individuals in particular:

- Ben Kotze, for all his advice and ideas and for all the useful discussions I had with him.
- Tian Bothma, for all the tips that he gave me regarding C# and Jean Janse van Rensburg for his help in setting up networks.
- The Central University of Technology, Free State, for the provision of research facilities and financial support.
- To all the guys in the Lab whom I have not mentioned here. Thank you for all the jokes and laughs, you made it fun to be involved in research.
- Dr Vermaak for his help with my final documentation.

SUMMARY

Automated Guided Vehicles (AGVs) have many different design specifications, although they all have certain design features in common, for instance they are designed to follow predetermined paths, and they need to be aware of their surroundings and changes to their surroundings. They are designed to house sensors for navigation and obstacle avoidance.

In this study an AGV platform was developed by modifying an electric wheelchair. A serial port interface was developed between a computer and the control unit of the electric wheelchair, which enables the computer to control the movements of the platform.

Different sensors were investigated to determine which would be best suited and most effective to avoid collisions. The sensors chosen were mounted on the AGV and a programme was developed to enable the sensors to assist in avoiding obstacles.

An imaging device as an additional sensor system for the AGV was investigated. The image produced by a camera and dome mirror was processed into a panoramic image representing an entire 360^{o} view of the AGV's surroundings. The reason for this part of the research was to enable the user to make corrections to the AGV's path if it became stuck along the track it was following.

The entire system was also made completely wireless to improve the flexibility of the AGV's applications.

OPSOMMING

Geoutomatiseerde geleide voertuie (GGV) word ontwikkel om verskeie take te verrig. Daar is sekere algemene ooreenkomste tussen GGV's, byvoorbeeld hulle word ontwerp om vooropgestelde roetes te volg, hulle moet bewus wees van hulle omgewing, en hulle moet veranderings in hulle omgewing waarneem. Hulle word ontwerp om sensors wat behulpsaam is met navigasie te huisves en om hindernisse te vermy.

'n Elektriese rystoel is verander en aangepas om as 'n GGV platform te dien in hierdie studie.

'n Seriepoort-kommunikasiestelsel tussen die rekenaar en die beheereenheid van die rystoel is ontwikkel. Dit stel die rekenaar in staat om die bewegings van die platform te beheer.

Navorsing is gedoen op verskillende sensors om te bepaal watter die geskikste sou wees om botsings te vermy.

'n Beeldherkenningstoestel is ook as 'n bykomende sensoriese sisteem vir die GGV ontwikkel. 'n Kamera is gebruik, en met behulp van 'n bolvormige spieël, skep dit 'n panoramiese (360º) beeld vir die GGV.

Hierdie deel van die ontwikkeling is gedoen sodat die gebruiker, te enige tyd, verstellings aan die koers van die GGV kan doen indien dit hindernisse teëkom.

Die hele sisteem word ook deur afstandbeheer bestuur om meer buigsaamheid aan die GGV se gebruike te verleen.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AC	Alternating Current
AGV	Automated Guided Vehicle
Ah	Ampere hour
BLDC	Brushless Direct Current
CNC	Computer Numerical Control
DC	Direct Current
GGV	Geoutomatiseerde Geleide Voertuig
GUI	Graphical User Interface
HMI	Human Machine Interface
LSB	Least Significant Bit
ms	Millisecond
MSB	Most Significant Bit
OFDM	Orthogonal Frequency-Division Multiplexing
PC	Personal Computer
PWM	Pulse Width Modulation
RPM	Revolutions Per Minute
RF	Radio Frequency
USB	Universal Serial Bus
V	Volt
WLAN	Wireless Local Area Network

Chapter 1

Introduction to Omnidirectional Viewing Automated Guided Vehicle

Robots have become an integral part of our day-to-day lives, something which most people are unaware of. Robots have been implemented to replace humans in many areas in factory production lines. They can be used for tedious, repetitious and dangerous jobs. Their precision is unmatched, and they have the ability to produce duplicates.

Automated Guided Vehicles (AGVs) are designed to perform operations without human guidance. They are used in a wide variety of industrial applications.

Typical industries where AGVs are used.

- Automotive, see Figure 1.1 [1]
- Printing
- Chemicals and plastics, see Figure 1.2 [1]
- Hospitals
- Food and beverages, see Figure 1.3 [1]

- Pharmaceuticals
- Warehouse and distribution centres
- Paper
- Manufacturing, see Figure 1.4 [1].

Figure 1.1 shows an AGV used to transport heavy products, which is the task of most AGVs. The AGV shown uses a wire guidance system. A wire is placed underground, and a sensor is used to determine where the AVG is and the route it has to drive to follow the wire.

Figure 1.1: AGV used in automotive parts assembly line

AGVs make tasks that could be dangerous to humans less hazardous, as illustrated in Figure 1.2, where an AGV is transporting dangerous chemicals.

Figure 1.2: AGV used in the chemical industry

Figure 1.3 demonstrates the tireless working ability of an AGV: the demand for food is so great that the AGV must keep on producing processed food without stopping.

Figure 1.3: AGV used in the food and beverage industry

Figure 1.4 shows an AGV that uses laser guidance. A laser mounted on the AGV measures distances between itself and reflective markers to determine its position.

Figure 1.4: AGV used in the manufacturing industry

The advantages of using AGVs in industry are [2]:

- Reduction of transport damages
- Reduction of transport costs
- Improved efficiency of systems
- Greater system flexibility (can easily make adjustments to tasks to be performed)
- Greater profitability.

In this project, a modified electric wheelchair was used as an AGV platform. The appropriate sensors were mounted on it for object avoidance. The integration of a camera system for imaging of the AGV surroundings was considered. A wireless system for communication was also installed.

1.1 AGV Definition

An AGV is an unmanned, self-propelled vehicle like a mobile robot, which is equipped with an onboard computer that stores path and machine control instructions for the steering and forward and backward drive of the machine, powered by an electric motor and batteries [3].

1.2 Development of the AGV

The AGV has been widely integrated in industrial applications since its introduction in 1953[4].

Several guidance systems are currently used in the steering and manoeuvring of the AGV. Initially guidance was via a system that utilised a wire placed under the surface on which the vehicle was travelling, called wire guidance or colloquially termed “Smart Floors and Dumb Vehicles”.

Since there is an ever-growing demand for more intelligent AGVs, it is worth investigating and improving their design.

1.3 Problem statement

AGVs lack information on their surroundings. This information is needed to enable them to navigate and to observe changes in the environment. The images from an AGV must be transmitted wirelessly to a computer. A method that is quick enough to do this has to be

found, and a method to control and analyse the data has to be developed.

The camera images as well as the commands used to operate and control the AGV must be transferred using the same communication system.

1.4 Project objectives

- Develop an AGV platform for test purposes.
- Utilise a high-resolution camera as a sensory unit to capture images of the AGV's surroundings mounted facing a conical mirror.
- Develop software to process and analyse the acquired image to convert the 360° image to a panoramic view.
- Determine the correct control commands using the panoramic image to best navigate the AGV around its environment.

1.5 Research method

The research was divided into different objectives starting with the development of a spherical or conical mirror, used in conjunction with a camera to capture a 360° image. This captured disc-like image had to be converted into a panoramic view on a horizontal plane. Various image-processing algorithms were experimented with and the best possible shape and size of the mirror were selected.

MATLAB® was used to assist in the software development process. With MATLAB® it is possible to simulate the mathematical transforms to be used to convert the disc-like image

into a panoramic view [5]. Once the experimentation and simulations were done, a program was developed with C# that performs the same functions and tasks as an executable program on a Personal Computer (PC) platform [6].

A variety of communication techniques were explored to determine which is the most suitable. The wireless bandwidth capability to transmit and receive the panoramic images and control instructions between the AGV and PC had to be investigated. Possible communication media included Bluetooth, Wireless Local Area Networks (WLAN) to mention just some examples of systems that were examined. WLAN was the major focus of this research as the communication medium of choice, because it has the ability to change its access points when moving out of range of a specific access point. When indoors, 802.11b signals can travel as far as 46 metres (150 feet). Outdoors, the 802.11b range is over three times greater- 92 metres [7]. Using radio frequency (RF) technology, WLANs transmit and receive data wirelessly in a certain area. This allows users in a small zone to transmit data and share resources, such as printers, without physically connecting each computer with cords or wires.

1.6 Structure of chapters: summary

Chapter 2 discusses the theory behind different components and methods that can be used to construct and control the functions of an AGV. Chapter 3 shows how the AGV evolved from a wheelchair as well as the construction of interfaces between the computer and ultrasonic sensor kits, infrared sensors and the control unit of the platform. The method for setting up wireless communication between the AGV and a control computer is also discussed. Chapter

4 examines how to obtain a full 360° view of a room from a single camera and the process of transforming the image into a more user-friendly interpretation. Chapter 5 shows all the test results of the AGV and the imaging system. A published article is reproduced in Appendix F.

1.7 References

1. FMC TECHNOLOGIES, 2007, AGV-Automated guided vehicles systems [online], FMC TECHNOLOGIES, Available at http://www.fmcsgvs.com/index2.htm [Accessed 1 April 2009]
2. FRAUNHOFER IML, 2008, Automated guided vehicles [online], FRAUNHOFER IML, Available at http://www.iml.fraunhofer.de/1524.html [Accessed 1 April 2009]
3. PATENT STORM, 11 July1989, US Patent 4846297 Automated guided vehicle [online], Available at http://www.patentstorm.us/Patents/4846297/fulltext.html, [Accessed 1 April 2009]
4. SAVANT AUTOMATION, 2008, History of Automatic Guided Vehicles Systems [online], Available at http://www.agvsystems.com/res/history.htm, [Accessed 1 April 2009]
5. USER'S GUIDE, Image Processing Toolbox, Version 4, The Mathworks, May 2003
6. Deitel, H.M., Deitel, P.J., Visual C#®2005, How to Program, Second Edition, Pearson International, London, 2006
7. ABOUT.COM, 2008, Wireless / Networking [online], ABOUT.COM, Available at http://compnetworking.about.com/cs/wirelessproducts/f/wifirange.htm [Accessed 1 April 2009]

Chapter 2

Omnidirectional Image and Automated Guided Vehicle Technologies

In this chapter the development of an AGV is discussed. Technologies that are essential for control, navigation and obstacle avoidance are investigated, and a practical method of constructing an omnidirectional viewing device is examined. The different components and systems are investigated to determine which is most effective and practical for implementation in an AGV.

2.1 Platform

The development of a platform on which the AGV is based is dependent on the application requirements of the environment in which it will be operating, as an AGV is not limited by size or capability.

Bearing this in mind, the next step is to decide whether it would be possible to modify an existing mechanism to suit the requirements of the environment and purpose for which the AGV is being developed, or whether it would be more effective to construct a platform from the start.

The platform is just a structure on which the individual components are mounted to form the

complete AGV. It is a starting point in the construction of an AGV [1].

Figure 2.1 depicts an electric wheelchair converted into a suitable platform.

Figure 2.1: Wheelchair modified into a platform

2.2 Propulsion and steering

Steering refers to the collection of components that allow a vehicle to follow a desired course [2].

Propulsion refers to the act of moving a carrier of people and goods over a distance [3].

The biggest challenge of building an AGV is the manoeuvring and control of the AGV. Therefore the following decision has to be made: does the AGV have a separate propulsion system and steering mechanism, or a single wheel that steers and propels, or does it have two bidirectional driving wheels that act as the propulsion and steering system?

The method adopted in many electric wheelchairs is two fixed, bidirectional driving wheels with two, or in some cases one, free-turning wheel that can rotate freely on the axis perpendicular to the ground on which it is travelling. When both drive wheels rotate in the same direction, the wheelchair moves either forward or backwards depending on the direction of the rotation of the wheels. When the wheels rotate in opposite directions to one another the wheelchair will turn as depicted in Figure 2.2 (seen from above).

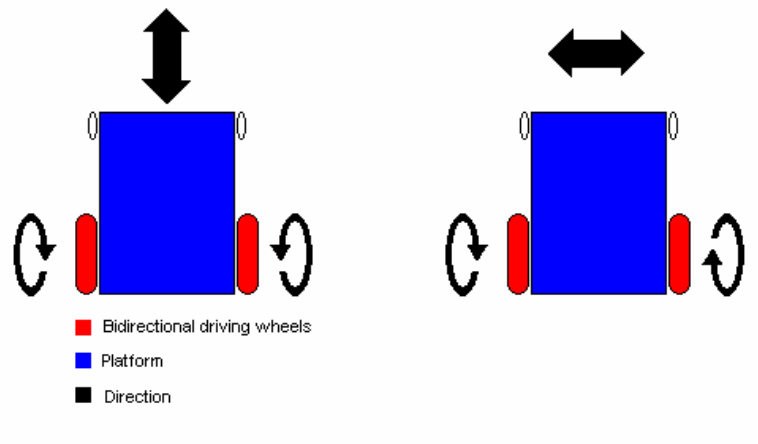


Figure 2.2: Steering direction

2.3 Motors

Different motors can be used for propelling an AGV, for example direct current (DC), alternating current (AC), brushless DC (BLDC) motors, servo motors and stepper motors [4, p.15].

The different motors and motor ratings affect the circuit used to drive the motors. When designing an AGV to transport payloads with a weight of 150 to 200kg, the most suitable choice would be a large DC motor. Motors of this size exceed the maximum amount of

current that can be supplied directly by a micro controller.

DC motors are not suitable for positioning unless some kind of position feedback is added. DC motors can be made to run in the opposite direction by changing the polarity of the current supplied to them. These motors are optimised to run at a fixed, usually high revolutions per minute (RPM). To reduce the running speed, the ratio of a gearbox that might be connected to it has to be high or the supply voltage has to be lowered, but the result is a reduction in the torque of the motor, so a different method has to be adopted.

Figure 2.3 shows an example of a DC motor and its ratings. It is obvious that the motor has a fixed RPM rating.

Figure 2.3: DC motor and its rating

2.4 Speed control

There are different methods to control the RPM of a DC motor, for example a gearbox with a ratio that would suit the application, voltage regulation or Pulse Width Modulation (PWM) [4, p.16].

2.4.1 Voltage regulation

Changing the voltage across a motor using a series resistor is an effective but inefficient method. As the resistance of the variable resistors increases, the voltage across the motor decreases and the total amount of current in the circuit will decrease causing the motor to lose torque.

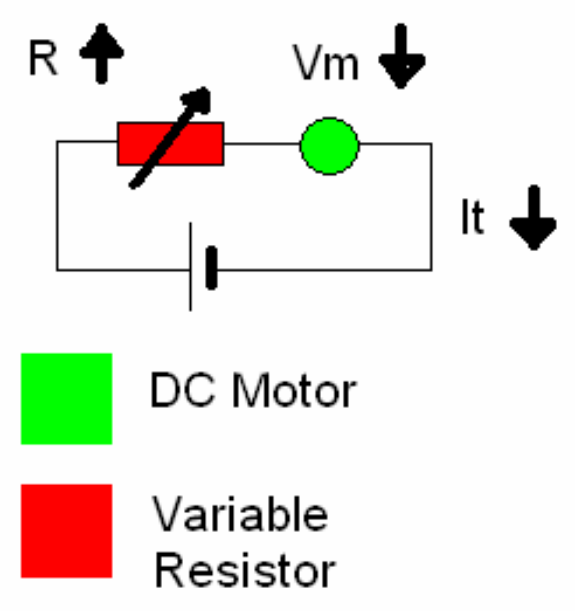


Figure 2.4: Diagram of a series resistor

2.4.2 Pulse width modulation

Pulse width modulation works by switching DC voltage to a square wave at a very rapid rate. It alternates between supply voltage and zero, imparting a series of pulses to the motor. If the

pulses are quick enough, the motor will run at a constant speed because of the momentum of the armature or flywheel. By adjusting the duty cycle of the square wave it is possible to change the motor's speed.

2.4.2.1 Duty cycle

The duty cycle is the width of the pulses as illustrated in Figure 2.5. It is adjustable.

[4, p.17].

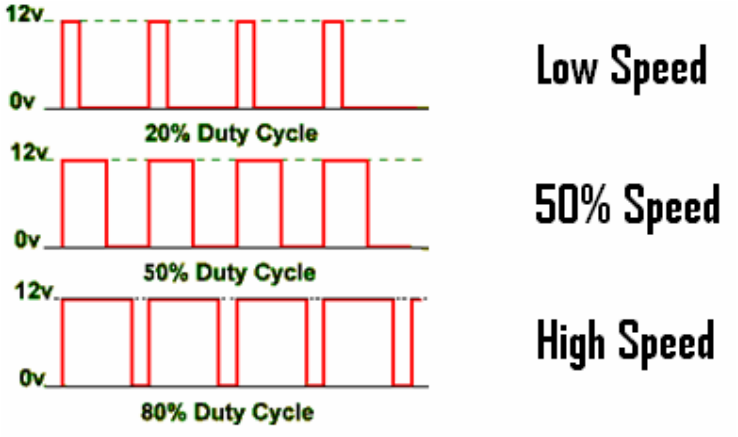


Figure 2.5: Duty cycle

There are a few advantages of using PWM, for instance the transistors are either fully on or off and not partly activated, and therefore less power is wasted as heat [5]. With a suitable circuit there are very little losses across the output transistor, and this means that the top end of the control range gets nearer to the supply voltage as compared to linear regulation. By

using PWM a motor can be turned at much lower speeds than one using an equivalent steady voltage supply.

The disadvantage of using PWM is that the pulses may be audible, especially at low revolutions.

2.5 Sensors

When an AGV is built it is very important for the AGV to be made aware of its surroundings. It must therefore be equipped with sensors that are able to pick up changes in the surroundings and report back to the AGV-the sensors thus act like feelers.

A sensor is a device that is used to measure a physical quantity and convert it into a signal that can be read by an observer or by an instrument. For example, a mercury thermometer converts temperature into expansion and contraction of the mercury, which can be read on a calibrated glass tube.

There are several different methods to determine proximity, including:

- Phase shift measurement
- Triangulation
- Time-of-flight
- Absolute interferometry

The three most common proximity sensors are laser, infrared and ultrasonic sensors.

Proximity sensors can be used when [6, p.2]:

- The object naturally transmits a signal
- The object has its own transmitter
- The signal is reflected – a signal is transmitted to the object and then the same signal is received back.

Ultrasonic sensors work similarly to sonar. They use the time-of-flight method in which they send out signals and determine an object's proximity by measuring the time taken between sending the signal and receiving an echo. This is possible by knowing the speed at which the transmitted sound wave travels in the medium in which the object is found, for example air or water.

$$Distance = speed \times time \quad (1)$$

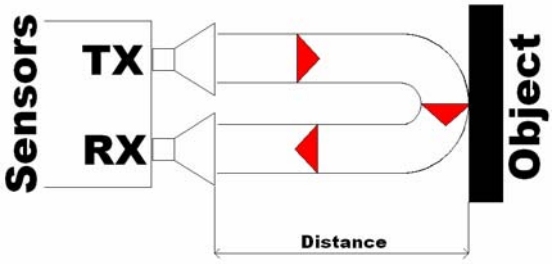


Figure 2.6: AGV with ultrasonic sensors

Figure 2.6 indicates that the sound wave has to travel to the object and back again. For object avoidance the distance to the object is important, so the distance that the wave has travelled has to be divided by two. Therefore the equation is now as follows:

$$\left.\begin{aligned} Distance &= \frac{speed \times time}{2} \\ Distance &= \frac{c \times t}{2} \end{aligned}\right\} \quad (2)$$

Figure 2.7 shows an ultrasonic sensor.

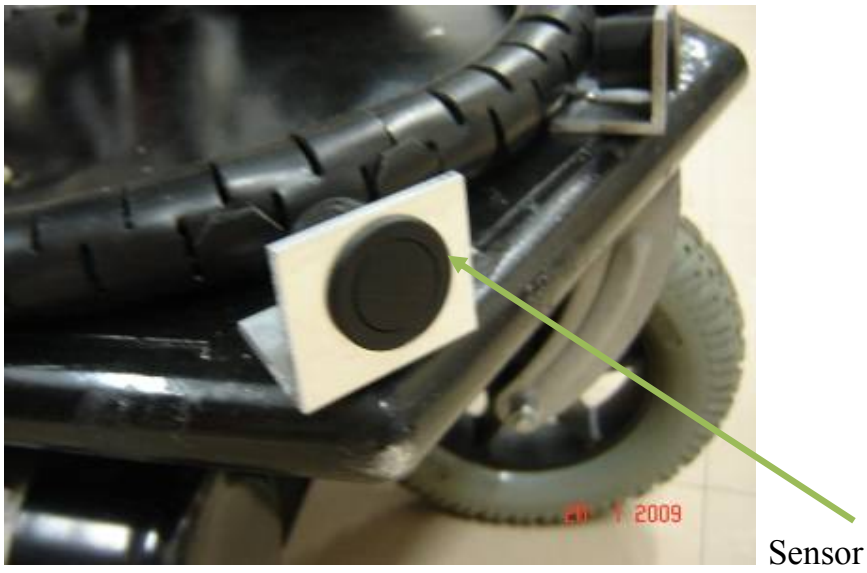


Figure 2.7: Ultrasonic sensor

2.5.1 Speed of sound (*c*)

Sound is a vibration that travels through a medium as a wave. Speed is the distance covered in a unit of time [7]. In dry air at 20°C the speed of sound is 343 m/s. The velocity of sound can be affected by environmental conditions such as temperature and humidity. For instance when the humidity increases the air becomes denser, thus increasing the velocity of the sound wave [8].

In general, the speed of sound *c* is given by:

$$c = \sqrt{\frac{C}{\rho}} \tag{3}$$

Where:

C is the coefficient of stiffness

p is the density

Thus the speed of sound increases with the stiffness of the material, and decreases with the density. For general equations of state, if classical mechanics is used, the speed of sound *c* is given by

$$c^2 = \frac{\partial p}{\partial \rho} \tag{4}$$

The approximate speed of sound in dry air (0% humidity) in metres per second ($\mathbf{m \cdot s^{-1}}$), at temperatures near 0 °C, can be calculated from:

$$c_{air} = 331.3 + (0.606 \times \vartheta) m.s^{-1} \quad (5)$$

The temperature in degrees Celsius (°C).

2.6 Navigation

Navigation is a process of planning a route whereby an aircraft, vehicle or vessel is controlled. All navigation techniques involve locating the vessel's position and comparing it to known patterns or known surroundings.

The general problem of AGV navigation can be summarised as three questions [9, p.10]:

- Where am I?
- Where am I going?
- How will I get there?

Dead reckoning is a navigational method whereby the present position is determined by projecting direction and speed travelled from a known position. The dead reckoning position is only an approximation as it does not make provision for any other external influences [10].

2.7 Communication

Communication is the process whereby information is conveyed over a medium. Communication requires that all parties understand a common communication language. Communication possesses a few major dimensions [11]:

- Content (what type of things are communicated)
- Source, emissary, sender or encoder (by whom)

- Form (in which form)
- Channel (through which medium)
- Destination, receiver, target or decoder (to whom)
- The purpose or pragmatic aspect.

2.7.1 Bluetooth

Bluetooth is a wireless protocol using a short-range communication technology that facilitates data communication over short distances. It has a 2.4GHz to 2.485GHz radio frequency bandwidth.

Bluetooth provides a way to exchange information between wireless devices such as cell phones, computers and digital cameras.

Bluetooth modules are able to communicate at distances ranging from 1 metre up to 100 metres at a rate of 921K Baud (RS232) [12].

2.7.2 WLAN

WLAN is a communication method that utilises a spread-spectrum or Orthogonal Frequency - Division Multiplexing (OFDM). This gives the user mobility, i.e. the ability to move around within a broad coverage area and still be connected to the network [13].

The benefits of WLAN include:

- Convenience

- Mobility
- Productivity
- Deployment
- Expandability.

Users are able to access network resources from nearly any location within their primary networking environment as well as maintain a nearly constant affiliation with their network as they move from place to place. Figure 2.8 shows a basic Wi-Fi range diagram.

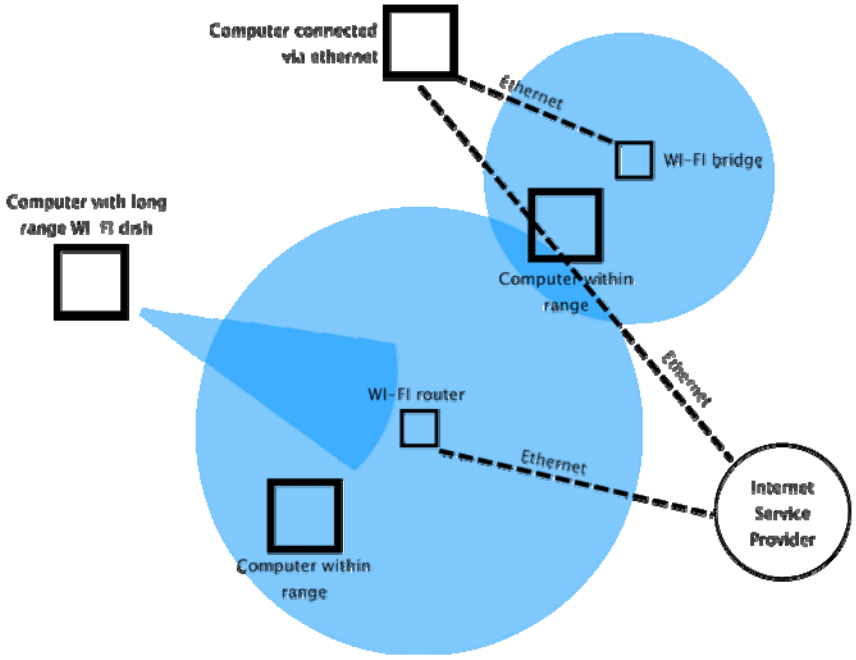


Figure 2.8: Wi-Fi range diagram

2.8 Imaging system

Like an array of sensors, it is possible to take multiple images of an AGV's surroundings and through video processing give a controller a clear picture of what obstacles are surrounding the AGV.

To produce a full 360^0 image of a specific point it is possible to mount multiple cameras directed in different directions to take one continuous image of the surroundings of that point, or a single camera can be used and simply rotated to the different positions and new images taken to form one continuous image as illustrated in Figure 2.9.

Figure 2.9: Multiple images to form a panoramic image

2.8.1 Using a single camera for an omnidirectional view

An image consists of light that is reflected off the object viewed, therefore if the light that is reflected off all the objects in a 360^o range is focused into a single camera lens it is possible to produce an image that represents an omnidirectional view of the surroundings [14].

An image of an entire room can be produced by mounting a camera so that it faces directly up at a round shaped mirror as shown in Figure 2.10.

2.9 Summary

Using the technologies and systems discussed, it would be possible to construct a fully functional AGV. Many different components and methods could be used other than those that have been investigated in this chapter.

2.10 References

1. ENCARTA, 2008, Platform [online], ENCARTA, Available at http://encarta.msn.com/dictionary_/platform.html [Accessed 1 April 2009]

2. WIKIPEDIA, 2008, Steering [online], WIKIPEDIA, Available at http://en.wikipedia.org/wiki/steering [Accessed 1 April 2009]

3. WIKIPEDIA, 2008, Propulsion [online], WIKIPEDIA, Available at http://en.wikipedia.org/wiki/Vehicle_propulsion, [Accessed 1 April 2009]

4. BOJE E.P., Intelligent AGV with navigation, object detection and avoidance in an unknown environment, CUT, Free State, Bloemfontein, 2006

5. BARR M, Introduction to Pulse Width Modulation, Embedded Systems Programming, Volume 14, Number 10, http://www.embedded.com/2001/0109, 2001

6. ALHAJ ALI S.M., Technologies for Navigation in Unstructured Outdoor Environments, Dissertation for PhD submitted to the University of Cincinnati, http://www.eng.uc.edu/~elhall/papers.html, 2003

7. MARSHALL B., An Introduction to Robot Sonar, www.robotbuilder.co.uk/resources/articles/138.aspx, 2005

8. AL-SUDANI M. Dr, WANG WEI L., Uber-Navigation Robot (UR), Exploring Innovation in Education and Research (iCEER-2005), http://www.iaalab.ncku.edu.tw/iceer2005/Form/PaperFile/99-0010.pdf, Taiwan, 2005

9. BORENSTEIN J., EVERETT H.R., FENG L., Where am I? Sensors and Methods for Mobile Robot Positioning, 1st edition, USA, University of Michigan, 1996

10. WIKIPEDIA, 2008, Dead reckoning [online], WIKIPEDIA, Available at

http://en.wikipedia.org/wiki/Dead-reckoning [Accessed 1 April 2009]

11. WIKIPEDIA, 2008, Communication theory [online], WIKIPEDIA, Available at http://en.wikipedia.org/wiki/communication-theory#Communication_theory_Framwork, [Accessed 1 April 2009]

12. BLUETOOTH, 2008, Bluetooth [online], Available athttp://www.bluetooth.com/Bluetooth/Technology/Basics.htm [Accessed 1 April 2009]

13. INTINI A.L., Orthogonal Frequency Division Multiplexing for Wireless Networks, http://www.create.ucsb.edu/ATON/01.01?OFDM.pdf, 2000

14. BAKERS. and NAYARS.K, A Theory of Single- Viewpoint Catadioptric Image Formation, Carnegie Mellon University, Columbia University, International Journal of Computer Vision, Vol. 35,No. 2,1999,pp. 1 – 22, 1999

Chapter 3

Development of the AGV platform

3.1 Evolution of the AGV platform

The vehicle chosen as the structure on which the AGV was based in this project is an electric wheelchair as shown in Figure 3.1.

Figure 3.1: Electric wheelchair

Its seat dimensions can be adjusted from 40cm to 50cm in width, its weight capacity is 200kg, and it is powered by two 12 volt (V), 50 Amperes per hour (Ah) lead-acid batteries

connected in series. The driving mechanism is two bidirectional 24V, 25 A direct current (DC) motors. It comes with a 24V, 3Ah charger. On full charge, the batteries will last between 4 to 6 hours, depending on the condition of the surface on which it is travelling [1].

The platform shown in Figure 3.2 is the modified wheelchair.

Figure 3.2: Platform after first modification

There are distinct advantages to a platform this size, for instance it can carry large payloads and there is ample space to house the electronics. However, in this case the physical size of the platform was too large, and it was found to be inefficient and not easily manoeuvrable. This resulted in a re-evaluation of the platform design. As shown in Figure 3.3, the platform was modified again and the result was a considerably smaller platform with only three wheels. The advantage of this design is that the platform now has improved traction which

will reduce slippage and navigational errors in the future.

Figure 3.3: Version two of the modified wheelchair

3.2 Controlling the AGV platform

The wheelchair is equipped with a joystick (Figure 3.4) connected to a control unit that houses the electronics for the control of the motors.

Figure 3.4: Joystick controller

The desired direction of movement of the wheelchair is achieved by having the motors turn in the necessary directions by moving the joystick. The joystick unit transmits the appropriate voltage values to the controller that houses the PWM electronics for the motors. This, in turn, causes the motors to turn in the correct direction and at the right speed for the wheelchair to move and manoeuvre.

3.2.1 Operation of the joystick

The output values supplied by the joystick unit range from 0V to 5V as indicated in Figure 3.5. The unit has four wire outputs, one for forward and backwards motion, one for right and left steering, one that is used as a reference voltage of 2.5V, and the fourth as a common ground.

When the joystick is in the centre position the output voltage from the unit is 2.5V, which has to match the reference voltage of 2.5V. When the joystick is moved forward the output voltage increases to greater than the reference voltage of 2.5V. The opposite applies: moving the joystick into the backwards position causes the output voltage to drop below 2.5V. The distance that the joystick is moved determines the amount of change to the output voltage [2].

The graphs in Figure 3.5 show the expected voltages according to the amount of movement of the joystick.

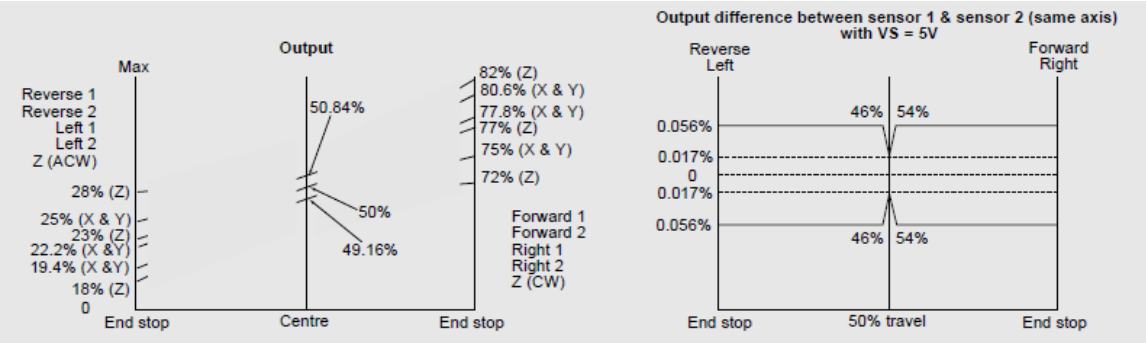


Figure 3.5: Joystick graphs from data sheet

3.2.2 Digital-to-analog conversion for platform control

For the platform to be controlled by a computer, it was necessary to construct an interface that would enable the computer to be linked to the platform's controls.

The control unit was left on the platform, but was modified so that a microcontroller could be plugged into the unit, thus giving control of the platform to a computer.

The block diagram in Figure 3.6 shows how the manual controller and the computer interface circuit are connected to the motor controller. This allows them both to control the platform's movements.

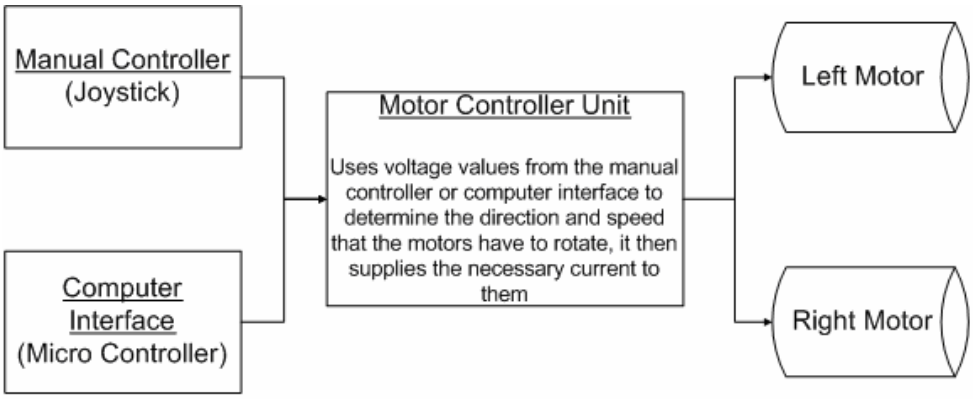


Figure 3.6: Modification of the motor controller unit

To connect the microcontroller to the controller unit, the voltage values from the joystick had to be matched to ensure proper operation and that no damage to the controller unit would occur. The circuit used to do this is an R-2R ladder network connection as shown in Figure 3.7.

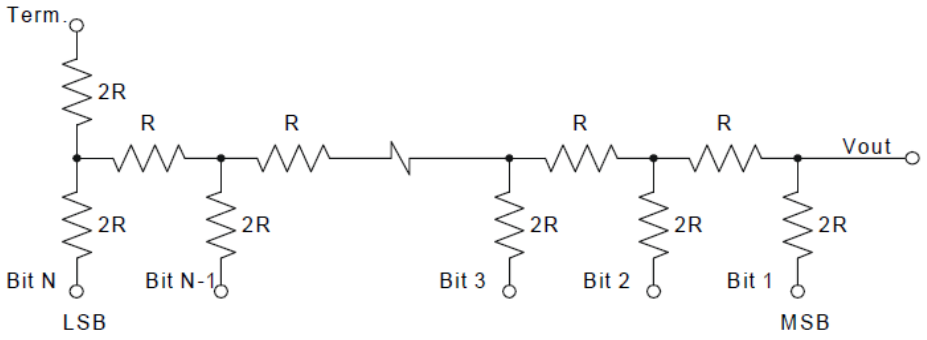


Figure 3.7: R-2R ladder of N bits network

Depending on the number and location of the bits switched to a reference voltage (Vr) or ground, Vout will vary between 0 volts and Vr. Vr is the desired maximum output voltage. This means that the bit values will either be 0V or Vr. Vr is connected to the circuit at the first terminal, as indicated in Figure 3.7 by the word Term. If all the inputs are connected to ground, 0 volts is produced at the output; if all the inputs are connected to Vr, the output voltage approaches Vr; if some inputs are connected to ground and some to Vr then an output voltage between 0 volts and Vr occurs. These inputs (also called bits) range from the Most Significant Bit (MSB) to the Least Significant Bit (LSB). As the names indicate, the MSB, when activated, causes the greatest change in the output voltage, and the LSB, when activated, will cause the smallest change in the output voltage. If the bits (or inputs) are

labelled bit 1 to bit N, the output voltage caused by connecting a particular bit to Vr with all other bits grounded is:

$$Vout = \frac{Vr}{2N} \tag{6}$$

where N is the bit number. For bit 1, Vout =Vr/2, for bit 2, Vout = Vr/4, etc. Note that since bit 1 has the greatest effect on the output voltage it is designated the Most Significant Bit. Since an R/2R ladder is a linear circuit, we can apply the principle of superposition to calculate Vout. The expected output voltage is calculated by summing the effect of all bits connected to Vr. For example, if bits 1 and 3 are connected to Vr with all other inputs grounded, the output voltage is calculated by:

$$Vout = \left(\frac{Vr}{2}\right) + \left(\frac{Vr}{8}\right) \tag{7}$$

which reduces to:

$$Vout = \frac{5Vr}{8} \tag{8}$$

The R/2R ladder is a binary circuit. The effect of each successive bit approaching the LSB is 1/2 of the previous bit. If this sequence is extended to a ladder of infinite bits, the effect of the LSB on Vout approaches 0. Conversely, the full-scale output of the network (with all bits connected to Vr) approaches Vr [3].

The control interface is shown in the block diagram in Figure 3.8.

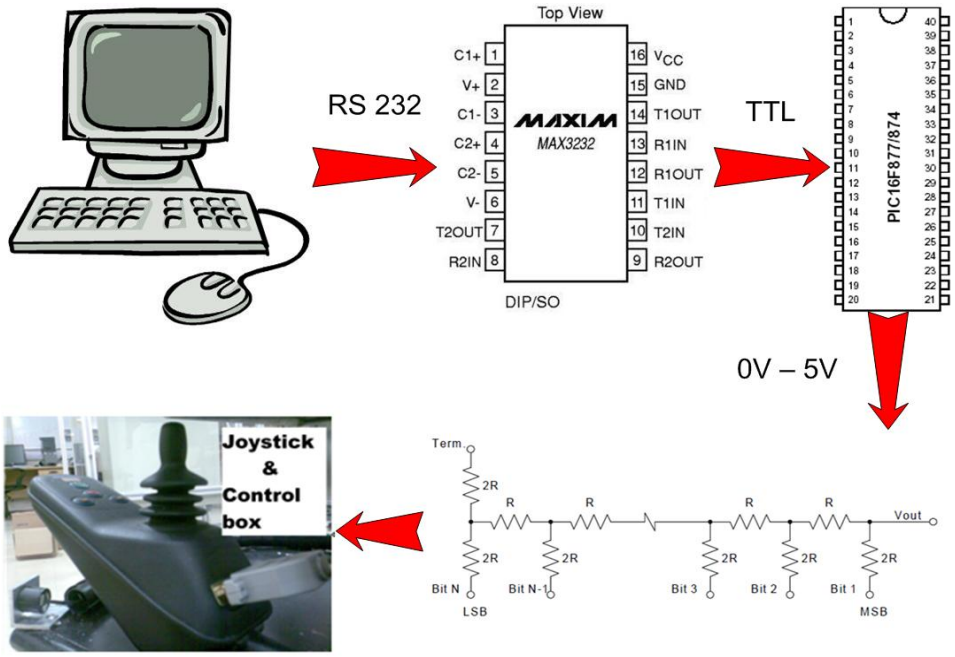


Figure 3.8: Control diagram

Figure 3.9 shows the interface circuit between the computer and the control unit of the platform.

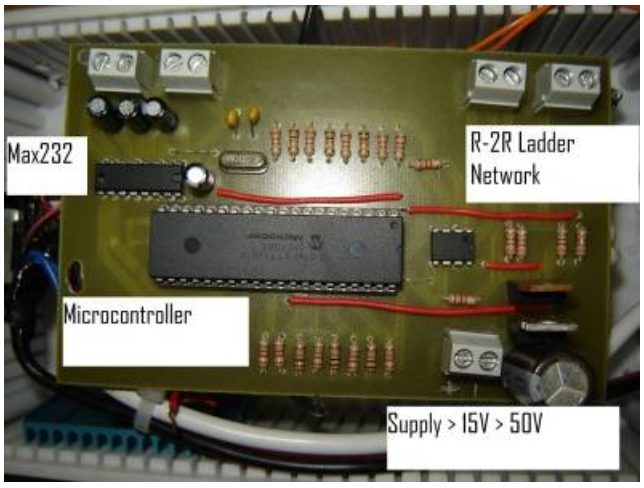


Figure 3.9: Joystick and microcontroller interface circuit

3.3 Ultrasonic kit

The ultrasonic kit is similar to those found in automotive parking applications. They are used to determine distance to ensure that no damage occurs to the vehicle when it is backing up. The kit is shown in Figure 3.10.

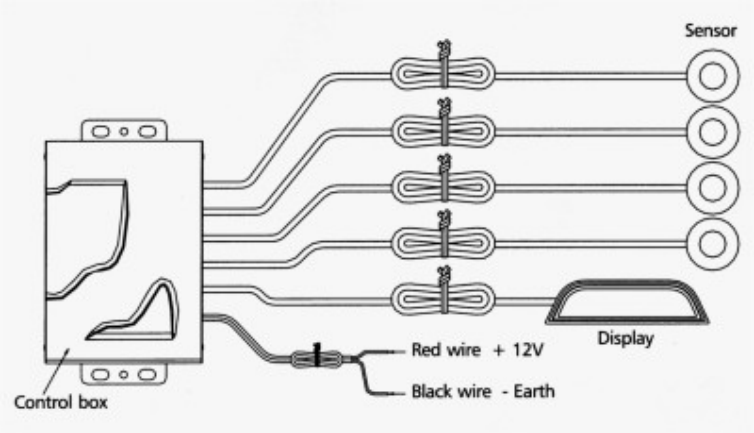


Figure 3.10: Parking sensor kit diagram

The information about the sensors is extracted from the data wire between the control box and the display. The format of the information as well as the Baud rate is determined by connecting a logic analyser to the data wire and the ground of the control box. Other information about the transmitted data can be determined, for example, the time between data words and number of data words before being refreshed, etc. The information, which is transmitted in TTL/CMOS format, can be converted to RS232 using a MAX232 and read using the serial port and hyper terminal. For the information to appear in a form that has any meaning to a user it requires a 30 HEX to be added to each word of data. This is because 0

represents 30 HEX on the ASCII map. The information is sent in a range lower than that of the standard ASCII map and appears as random symbols depending on the font set for the hyper terminal. For example if 0X00 is transmitted, the decimal value will be zero and the character will be a random symbol such as a "." or a "®". If 0X30 is added to the data word the resulting character will be a "0" and therefore more meaningful to the user.

Figure 3.11 is a screen shot of the program used with a logic analyser. It is clear from this figure that the Baud rate of the information sent is 9600 bits per second in this case, thirteen words of data are sent every 13ms and the information is refreshed every 220ms.

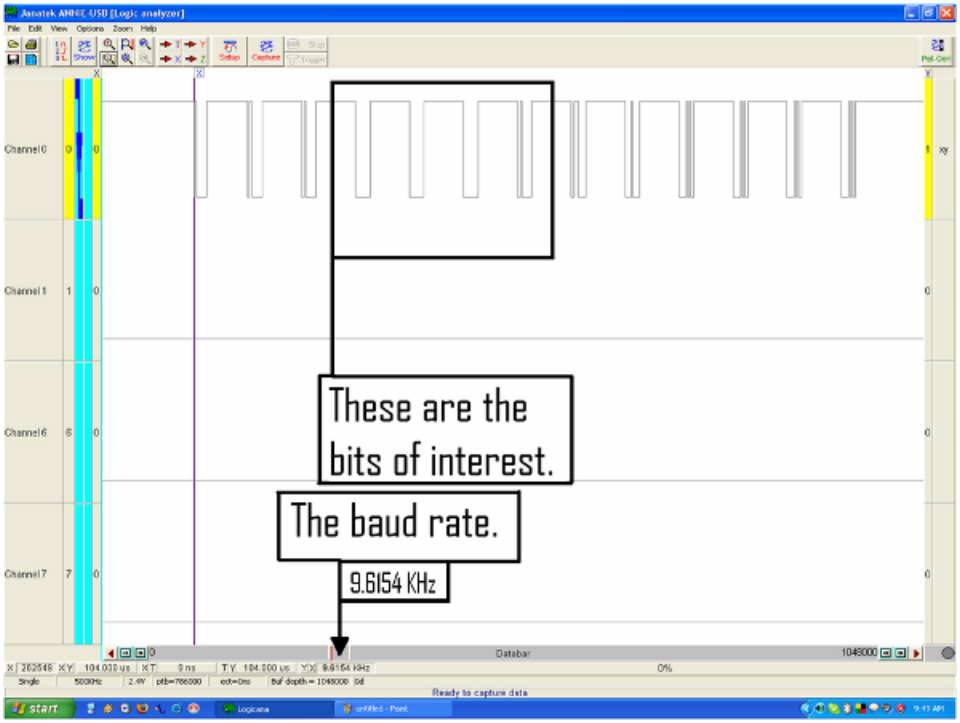


Figure 3.11: Screen print of the logic analyser

3.3.1 Interpretation of the digital information

Of the thirteen words of data, the first three represent the value of the closest sensor irrespective of which one it is. The first word represents the distance in metres, the second the distance in centimetres and the third the distance in millimetres. The four data words that follow each represent an individual distance of a sensor, and the rest of the data are reserved for future use.

The control box has four sensors, and if more are required it is possible to add more control boxes. For this to be done a system was designed to separate the information into a suitable protocol because the information is identical from both control boxes and needs to be distinguished so that there is no confusion as to which sensor is being analysed. In this system three microcontrollers are used to separate the information of the two control boxes, namely a PIC16F877 and two PIC16F628 microcontrollers.

3.3.1.1 Receiving data from the control box

The data are sent from the control box to a PIC16F628. The microcontroller is programmed to read the data into its serial port and store each of the thirteen individual words in their own registers and refresh them as new information is received. This is done by programming the microcontroller with a reset timer so that after every word of data has been received the timer is reset. If the timer overflows, the microcontroller is programmed to receive the next thirteen words of data as a refreshed set and to update the information stored in the registers. This is better understood from Figure 3.12.

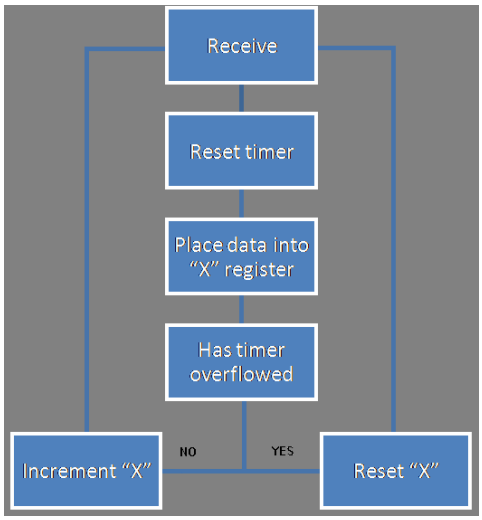


Figure 3.12: Timer routine to sort information

3.3.1.2 Transmitting the stored information to the PIC16F877

As it is not possible to have two different serial sources transmit to one receiving microcontroller unless it has two separate serial ports, the information is sent in parallel from the PIC16F628 to the PIC16F877. This is done by programming the PIC16F628 to have an interrupt service routine from which the PIC16F877 can retrieve the data. The interrupt service routine turns off the receiving serial port and waits to be toggled. With each falling edge of the incoming clock pulse from the PIC16F877 the next data word is transmitted until all the information has been sent, and it then exits the interrupt routine, turning on the serial port again so that the data collection from the control box can continue. Figure 3.13 explains this section more clearly. Figure 3.14 shows the data collector microcontroller (PIC16F877) and the two serial receiver microcontrollers (PIC16F628). Figure 3.15 shows how the different microcontrollers are connected.

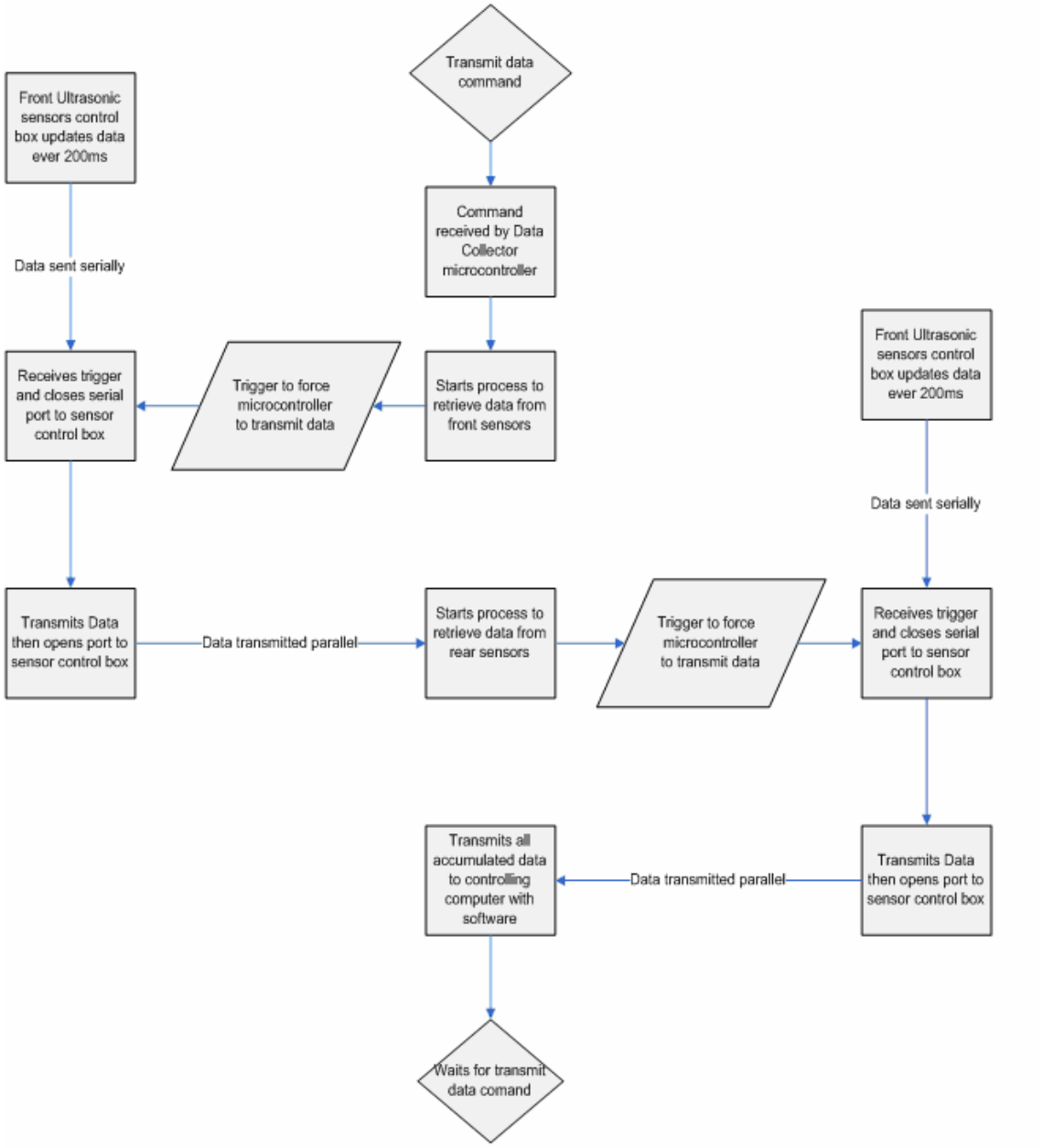


Figure 3.13: Flow diagram of the data collection process

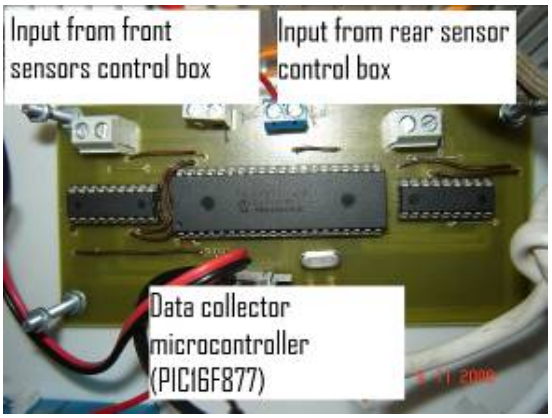


Figure 3.14: Data collection circuit

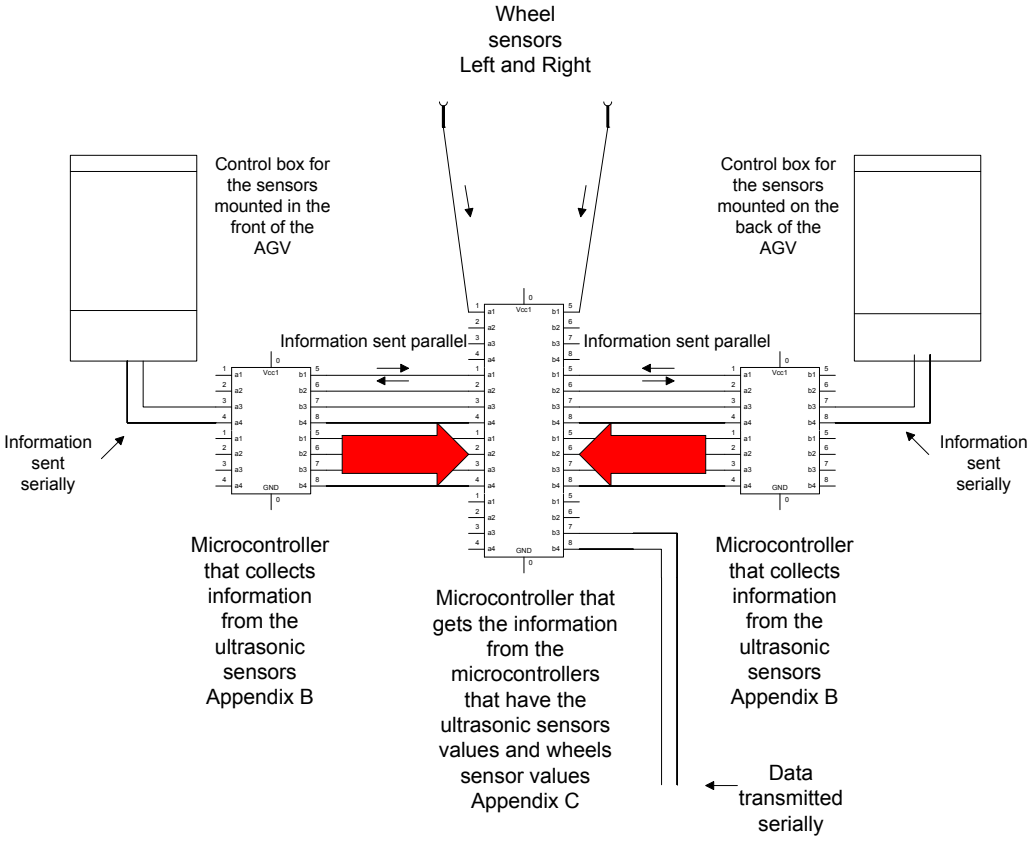


Figure 3.15: Basic layout of circuit in Figure 3.14

3.4 Wheel sensors for navigation

For the AGV to have some sort of navigation, its position in space has to be determined. Dead reckoning is a method for determining the position of the AGV by knowing the starting position and the distance and direction that each driving wheel has moved, assuming there are no errors caused by slippage and friction. For the amount of movement of the driving wheels to be determined, feedback from the wheels is obtained by placing an infrared proximity sensor on the gearbox of the driving wheels as shown in Figure 3.16. The proximity sensor is triggered when objects pass in front of it, and in this case the objects are the spokes of the wheels.

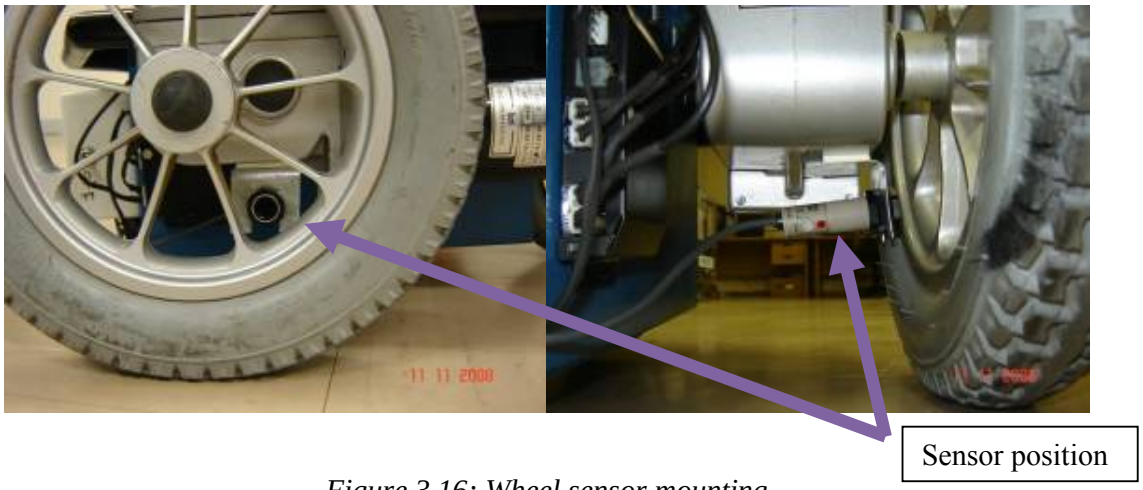


Figure 3.16: Wheel sensor mounting

3.4.1 Distance of the spokes

To determine the distance that each spoke represents, the diameter of the wheel is measured and multiplied by Pi (π) and then divided by ten as there are nine spokes. However, one of the

spokes is counted twice as it is the beginning and the end of the revolution.

$$U = \pi D \tag{9}$$

The distance between each spoke is 108mm, so for every spoke counted an additional 108mm can be added to the new predicted position for the dead reckoning navigation system.

3.4.2 Turning angle represented by each spoke

The distance between the driving wheels has a direct effect on the angle that the platform turns when one wheel is stationary and the other rotates. It was therefore adjusted so that when one wheel moves one spoke's distance forward (108mm) and one remains stationary the entire platform turns ten degrees (10°) on the path it is following. One of the characteristics of the wheelchair controller is that when it turns it does not just have one wheel rotate forward and the other remain stationary, but instead has both wheels rotate in opposite directions in order to shorten the turning circle. This means that if one wheel were to rotate while the other remained stationary, nine spokes would be counted for the platform to turn 90°. In this project only four on one wheel and five on the other are counted for the platform to turn 90°. Figure 3.17 shows the effect that the distance between the wheels has on the turning platform.

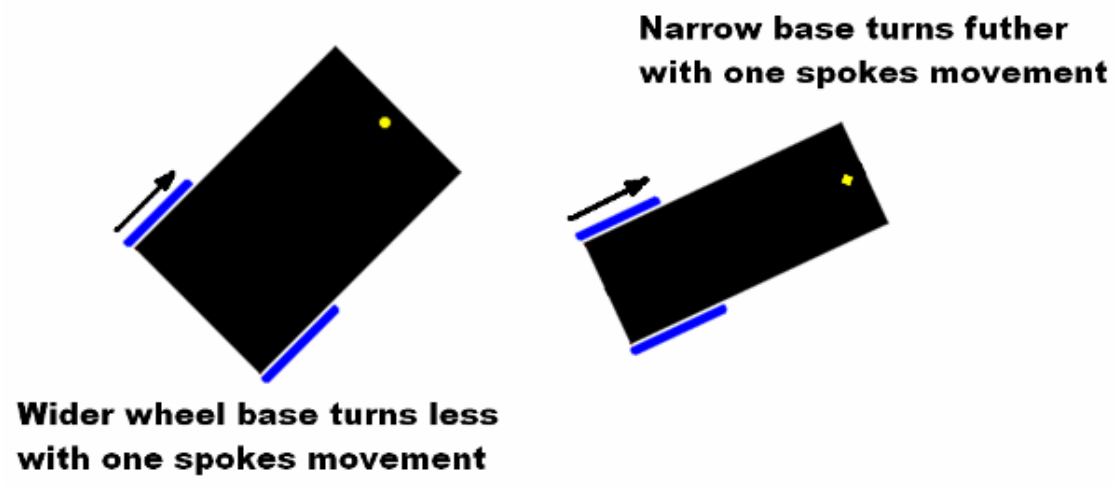


Figure 3.17: The effect of the distance between the wheels on the degree of turning

3.4.3 Software for dead reckoning

The software language used for this project is C# [4]. To determine the position of the AGV, feedback from sensors placed on the wheels of the AGV are used to draw a line on the screen. When the AGV moves, a line on the computer screen also moves, the ends of the line representing each wheel. The draw function of the program is able to draw a line between two coordinates. The coordinates are determined by using the fact that when a sensor increments, the angle of the AGV changes by 10° either in a positive or a negative direction as indicated in Figure 3.18. If the length of the representing line and the angle of the line on the Cartesian plane are known, the coordinates of the new end of the line can be determined using the SIN and COS mathematical functions as in Figure 3.18. The process is as follows: when a forward command is sent to the AGV, the sensors will be incrementing, and by doing so the line on the screen moves away from a starting position selected by the user. The opposite applies for backwards movement. When the AGV is stationary and receives a

command to turn left or right, only then do the wheels rotate in opposite directions, and therefore in this case the one angle of the one side of the line decrements while the other increments, depending on the direction in which the AGV is turning. When the AGV is moving forward or backwards and turning, the wheels do not move in opposite directions- the one wheel just moves slower than the other depending on the direction of turning.

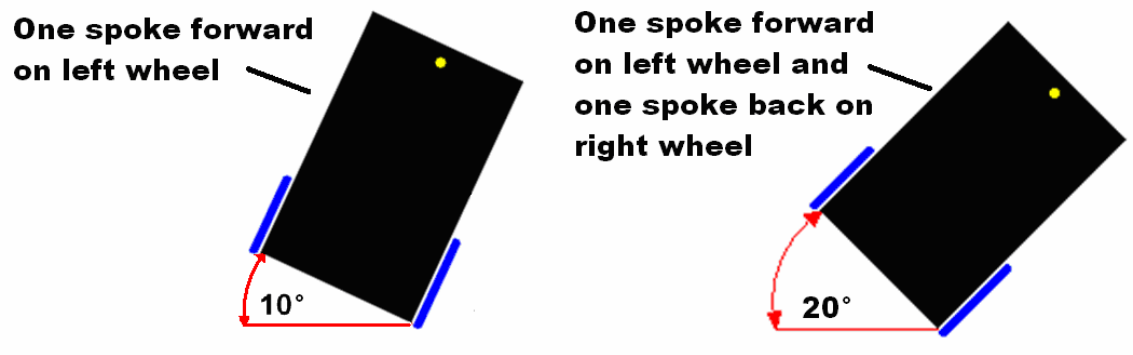
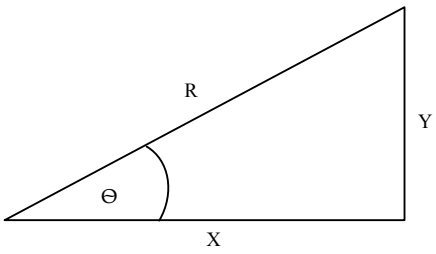


Figure 3.18: Difference between one-wheel movement and two-wheels movement

Figure 3.19 shows how the new coordinates are calculated in the software program.



$$X = R\cos\theta$$

$$Y = R\sin\theta$$

Figure 3.19: Rotation movement and coordinate calculation

3.5 Summary of the controlling circuits

The complete circuit shown in Figure 3.20 interfaces with the external controller of the AGV, collects data from the ultrasonic sensors mounted on the AGV and detects the AGV's movements.

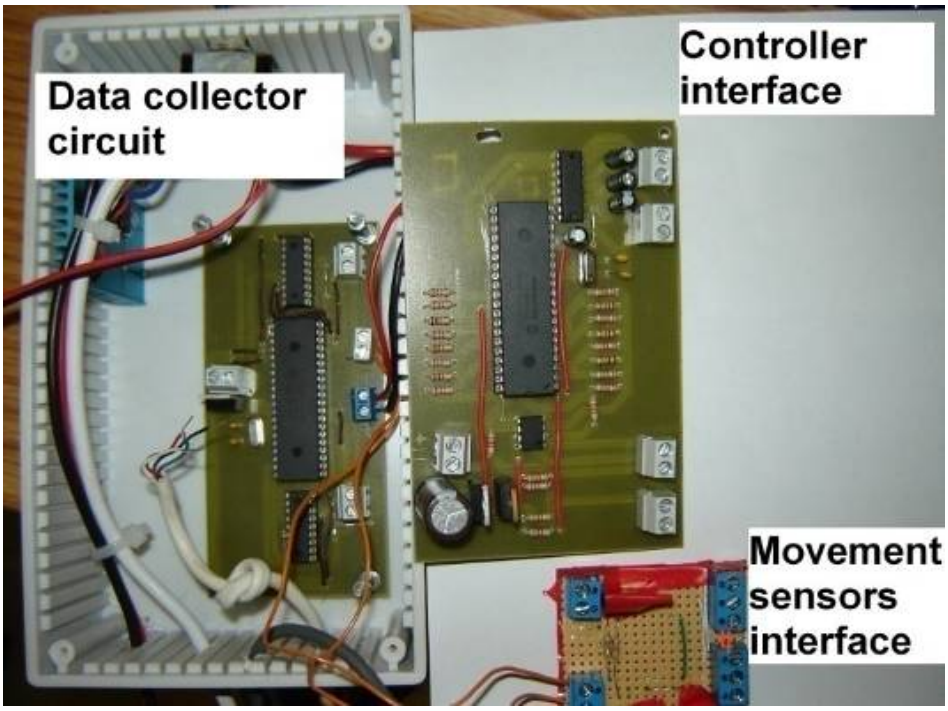


Figure 3.20: Complete control circuit

3.6 Wireless modules for data and control communication

The project has greater flexibility as a wireless system. Figure 3.21 shows a complete wireless local area network. By default the kit is set so that many serial bridges can connect to one central hub (access point) and they are issued with their own IP addresses to avoid any clash of information.

Figure 3.21: Airborne™ wireless LAN (module) evaluation and design kit

For cable replacement using Airborne™ products so that there is a direct connection between the controlling computer and the platform, an adhoc connection is set up. The settings of the access point are accessed by connecting the access point to a computer and allowing it to issue the serial bridges with a new IP address. When the Internet Explorer web browser is opened and the address http//(the IP assigned by the access point) is accessed, the next window that pops up asks for a user name and password. Both of these remain the same until changed by the user. They are: username: *dpac* and password: *dpac*. This gives the user access to the settings of the serial bridge. The following tables outline the steps to setup a wireless cable replacement connection between two serial ports using the Airborne™ products. This connection relies on a type of peer-to-peer wireless network called an AdHoc. This network type does not require an access point. An alternative method to setup the AdHoc network is to connect the access point to a serial port by using HyperTerminal sending the CLI commands listed in the following tables [5].

Table 3-1: Slave configuration and set-up

	Description	Setting	Webpage	CLI Command
1	Set the SSID of the unit to the name of the AdHoc network	AdHoc Network Name	Network	`wl-ssid AdHocNetwork`
2	Set network type to AdHoc (Infrastructure is default)	AdHoc	Network	`wl-type p`
3	Set AdHoc Channel	1	Network	`wl-chan 1`
4	Disable DHCP	Disable	Network	`wl-dhcp 0`
5	Assign a static IP	192.168.10.150	Network	`wl-ip 192.168.10.150`
6	Assign a network mask	255.255.255.0	Network	`wl-subnet 255.255.255.0`
7	Enable the Direct tunnel	Enable	Serial	`wl-tunnel 1`
8	Assign the tunnel port (8023 is the default and there is no need to change it)	8023	Serial	`wl-tunnel-port 8023`
9	Set the tunnel mode to TCP (this is default) - Assumes setting up a TCP/IP connection between the devices	TCP	Serial	`wl-tunnel-type tcp`

10	Configure the serial port settings to match the attached system	Baud Rate = 9600
		Data Bits = 8
		Parity = None
		Flow Control = Hardware (RTS/CTS)
		Stop Bits = 1
			Serial	`Bit-rate 9600`
				`data-bits 8`
				`parity n`
				`flow h`
				`stop-bit 1`

11	Set serial default mode to LISTEN (CLI is default)	Listen	Serial	`serial-default listen`

12	Save the settings and restart the unit.	Save and Restart	Reset	`commit`
				
				`restart`

Table 3-2: Master configuration and set-up

	Description	Setting	Web or ACC page	CLI Command
1	Set the SSID of the unit to the name of the AdHoc network	AdHoc Network Name	Network	`wl-ssid [AdHoc Network Name]`
2	Set network type to AdHoc (Infrastructure is default)	AdHoc	Network	`wl-type p`
3	Set AdHoc Channel	1	Network	`wl-chan 1`
4	Disable DHCP	Disable	Network	`wl-dhcp 0`
5	Assign a static IP (Slave address + 1)	192.168.10.151	Network	`wl-ip 192.168.10.151`
6	Assign a network mask	255.255.255.0	Network	`wl-subnet 255.255.255.0`
7	Set the Primary LAN Server IP Address to match the slaves static IP address	192.168.10.150	Serial	`wl-tcp-ip 192.168.10.150`
8	Set the LAN Server port to match the tunnel port on the slave	8023	Serial	`wl-tcp-port 8023`

9	Configure the serial port settings to match the attached system	Baud Rate = 9600
		Data Bits = 8
		Parity = None
		Flow Control = Hardware (RTS/CTS)
		Stop Bits = 1
			Serial	`Bit-rate 9600`
				`data-bits 8`
				`parity n`
				`flow h`
				`stop-bit 1`

10	Set serial default mode to PASS (CLI is default)	Pass	Serial	`serial-default pass`

11	Save the settings and restart the unit.	Save and Restart	Reset	`commit`
				
				`restart`

As long as the slave device is on and waiting for the connection, the master will boot and establish a TCP/IP connection with the slave. The slave will accept the connection and a serial-to-serial data tunnel will be established between the two units. Once the tunnel is established data can be sent between the two devices.

3.7 Conclusion

The entire AGV is controlled by a computer software program through an RS-232 serial port across a WLAN. It receives commands sent from a controlling computer and returns information about its current status and the action that it is performing. For example, if the AGV were driving forward it would return an ‘F’ character in its protocol to the controlling computer, and with this information and information from the wheel sensors the software program determines the path the AGV is following. If the AGV were found to be standing still for a certain amount of time it would mean that a technician would have to investigate a possible problem with the AGV. For this purpose an imaging system could be installed in the AGV enabling the technician to solve the problem by means of a remote human machine interface (HMI). This solution is discussed in Chapter 4.

3.8 References

1. Mr.WHEELCHAIR, 2008, Products – Powered Wheelchair [online], Available from http://www.mrwheelchair.co.za/prods_powered.htm, [Accessed 1 April 2009]
2. PENNY+GILES, 2008, Industry robotics joystick controllers [online], Available at http://www.pennyandgiles.com/products/products.asp?strAreaNo=402_9&intElement=1237&intIndustry=17, [Accessed 1 April 2009]
3. PLAY – HOOKEY, 2007, Digital to Analog Conversion [online], Available at http://www.play-hookey.com/analog/d2a_convertor.html, [Accessed 1 April 2009]
4. Deitel H.M., Deitel P.J., 2006, Visual C#® 2005, How to program, Second Edition, Pearson International, London
5. QUATECH, 2008, Products - 802.11b/g serial and Ethernet wireless solutions [online], Available at http://www.quatech.com/catalog/airbornedirect_80211bg.php, [Accessed 1 April 2009]

Chapter 4

Imaging system using a single viewpoint and a dome mirror

This chapter describes the vision system of the project: the type of camera used, the distance of the camera from a reflective dome, how the software program is written and what processes occur to accomplish the image processing which converts the image from the camera into a panoramic image.

4.1 Camera

To obtain high-quality images with high speed and high resolution, a webcam would not be adequate for the requirements of the project. The camera shown in Figure 4.1 is a Basler A601fc. It is a high-quality camera with a resolution of 656 x 491, and can stream images at sixty frames per second but typically streams at thirty frames per second.

Figure 4.1: Basler FireWire camera

4.2 Camera connections

The camera connects to a computer using a FireWire IEEE 1394 cable. There are two types of IEEE 1394 connectors that are typically used: a six-pin connector on the camera, and a four-pin connector as found on many desktop and laptop computers. Figure 4.2 shows the back of the Basler camera as well as the pin numbers of the IEEE 1349 socket. The IEEE 1349 socket is used to supply power to the camera and to interface video data and control signals [1, p.16].

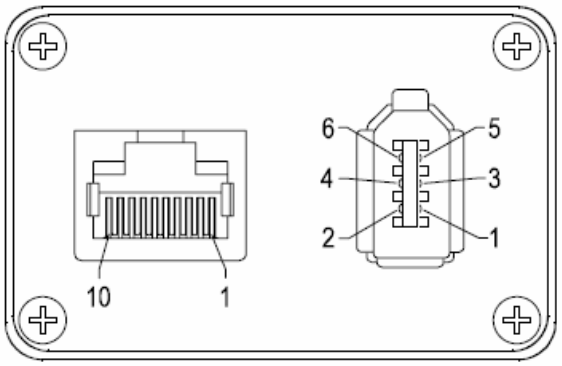


Figure 4.2: Back of the camera with pin out numbers

Table 4-1 gives the pin assignments of Figure 4.2 as well as the necessary voltage levels needed to supply the camera with power.

Table 4-1: Pin connections for the different pin connectors

4-pin connector	6-pin connector	9-pin connector	Name	Description	color of wire in cable
	1	8	Power	Unregulated DC; 30 V no load	white
	2	6	Ground	Ground return for power and inner cable shield	black
1	3	1	TPB-	Twisted-pair B, differential signals	orange
2	4	2	TPB+	Twisted-pair B, differential signals	blue
3	5	3	TPA-	Twisted-pair A, differential signals	red
4	6	4	TPA+	Twisted-pair A, differential signals	green
		5	A shield		
		7		-	
		9	B shield		
Shell			Outer	cable shield	

Six-pin to four-pin connector converters are available. The difference between the two is that the four-pin connector does not make provision for a ground and a power input pin. Figure 4.3 shows how all the different connectors' pin outs are numbered.

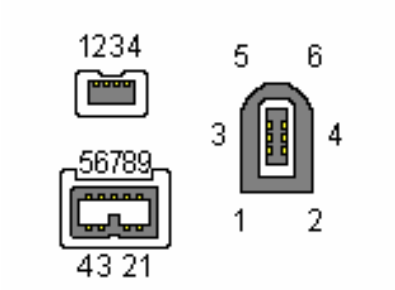


Figure 4.3: 4-pin, 6-pin or 9-pin IEEE1394 (FireWire)

For this project a laptop computer is mounted on the AGV as the controlling computer. The computer uses a four-pin connection, therefore a cable was made up using a four-pin and a six-pin connector. The connections are the same as the ones shown in Table 4-1. The power supply for the camera is tapped from the universal serial bus (USB) port of the laptop. The pin numbering for the USB port is shown in Figure 4.4 and the pin assignments are given in Table 4-4.

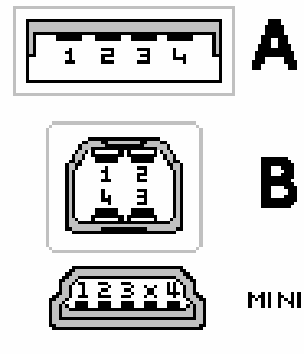


Figure 4.4: USB pin numbering

Table 4-2: USB pin assignments

Pin one side	Signal	Description	Pin other side
1	USB Vcc (Vbus)	usually RED, wire should be 20-28 AWG	1
2	USB Data -	usually WHITE, wire should be 28 AWG	2
3	USB Data +	usually GREEN, wire should be 28 AWG	3
4	GND	usually BLACK, wire should be 20-28 AWG	4

The cable used in the project is shown in Figure 4.5. The image shows two USB sockets. The reason for using two sockets is to supply power to the camera without overloading one of the USB ports. The USB sockets are connected in parallel and then to pins one and two as in Table 4-2.

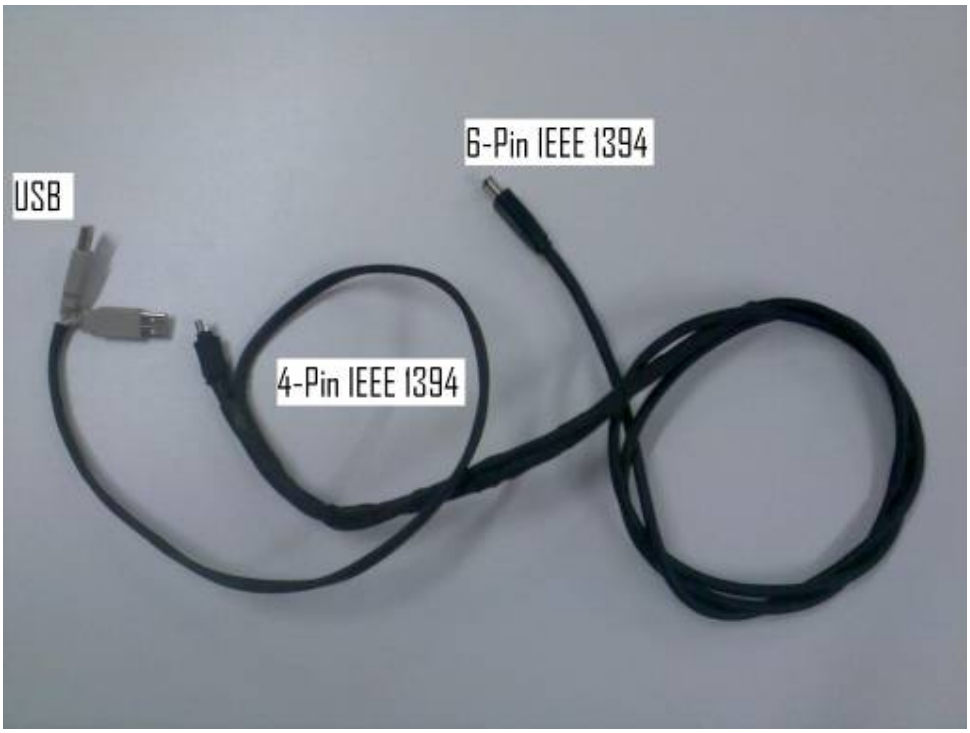


Figure 4.5: 6-pin to 4-pin IEEE 1394

4.3 Reflective dome

To produce an image that renders an entire 360° view, a round reflective surface is required. For the project a dome mirror was fabricated using a solid rod of aluminium cut on a Computer Numerical Control (CNC) lathe. Figure 4.6 shows the process of fabricating the round mirror. The first photo (top left) shows the raw material before the fabrication process. The rod of aluminium was placed in a CNC lathe where a large portion was cut away to form the rounded tip as seen in the fourth photo. The last photo shows the mirror once it has been removed from the CNC lathe and it has been buffed and polished. The diameter of the rod is 100mm and the rounded tip has a radius of 50mm. Determining the size of the rod is discussed in Section 4.4.

Figure 4.6: Fabrication process of the dome mirror

4.4 Camera position

To get an image that represents a full 360° view of the camera's, surroundings the camera was mounted on a stand facing directly upwards pointing at the dome-shaped mirror as depicted in Figure 4.7.

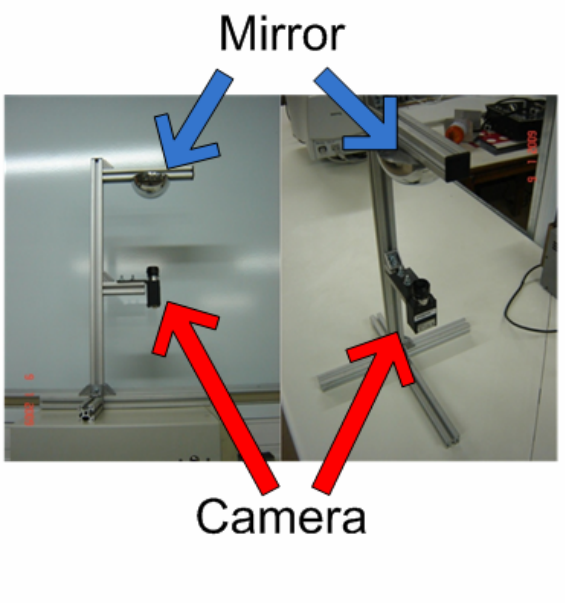


Figure 4.7: Camera and mirror stand

To get the camera correctly focused, it had to be decided whether the camera should be focused on the mirror or on the image in the mirror, and both methods were tested. The first method was carried out by sticking a small piece of sticky tape with a letter written on it to the mirror. The camera was then focused on the letter on the sticky tape. This meant that the camera was focused on the mirror and not on the images in the mirror. The images reflected in the mirror were found to be hazy and the mirror appeared far away in the image. The tape was then removed and the camera refocused, this time on the reflection of the camera's lens in the mirror. This meant that the camera was now focused on the images in the mirror. By using this method it was found that the mirror could be placed closer to the camera without losing any image quality. The reason for this is that the focal length is halved because the camera is focused on the images and not on the mirror as illustrated by Figure 4.8.

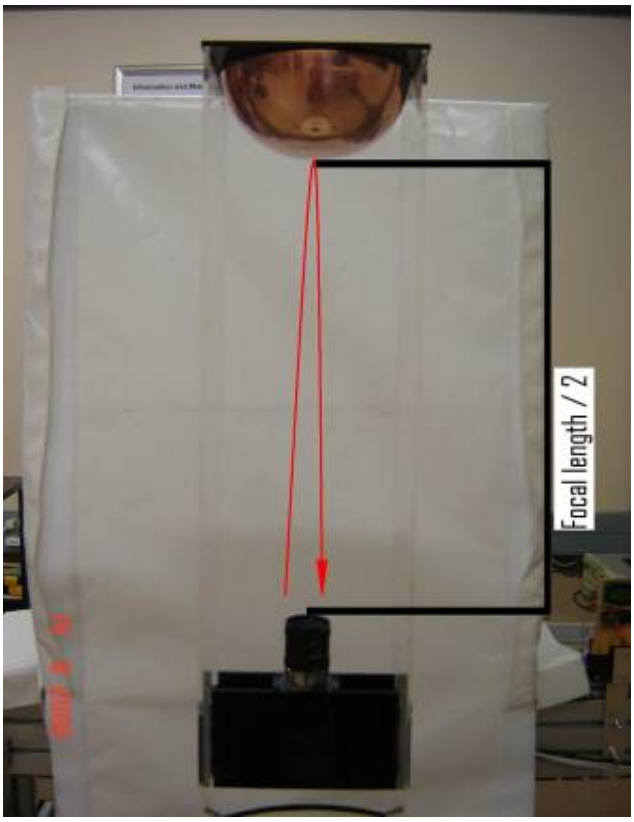


Figure 4.8: Focal length

The dome mirror's diameter was determined by placing a piece of paper at the same distance as the focal point of the image, and the outer perimeter of the image was drawn with a pen while looking at a streaming image from the camera. With a 12mm lens, the closest focal length of which is 0.3m, the image size from the top of the image to the bottom is 95mm, therefore a 100mm rod was used to make the dome mirror. It was moved away from the lens until it fit perfectly into the image as depicted in Figure 4.9.

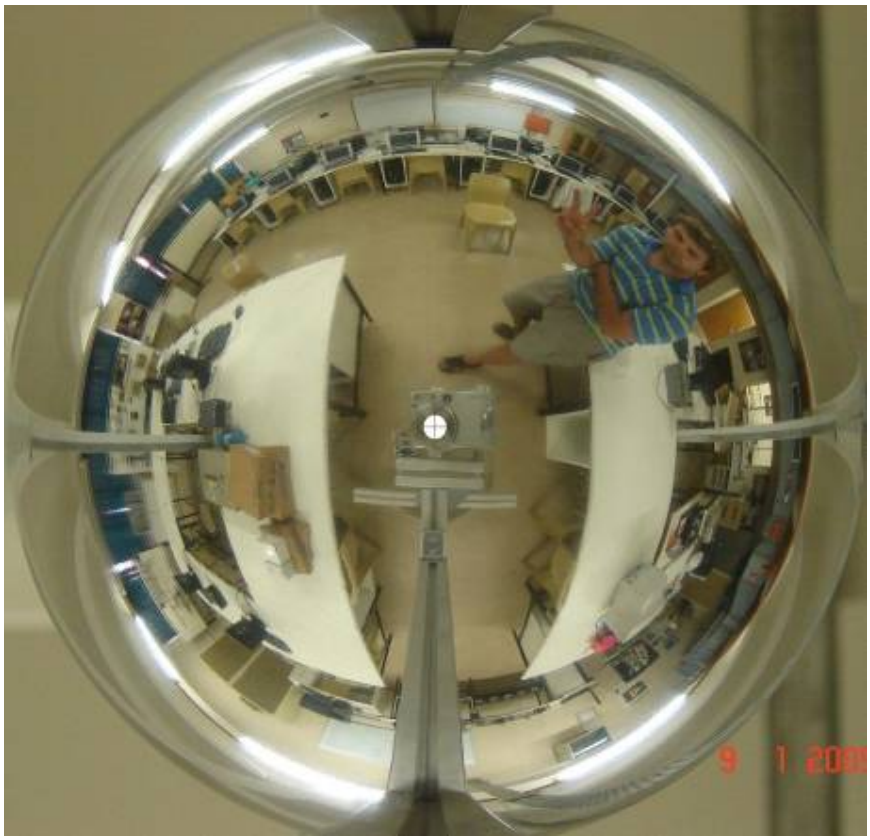


Figure 4.9: Image from the dome mirror

4.5 Camera settings

The camera used can be fitted with lenses that allow the amount of light entering the camera to be adjusted for situations where less light is needed. The shutter speed of the camera can also be set, as can the sensitivity to the three primary colours so that unwanted light can be filtered out. The camera software that was downloaded from Basler's website has a tab that automatically sets the colour balance as illustrated in Figure 4.10. This image also shows the settings of the camera.

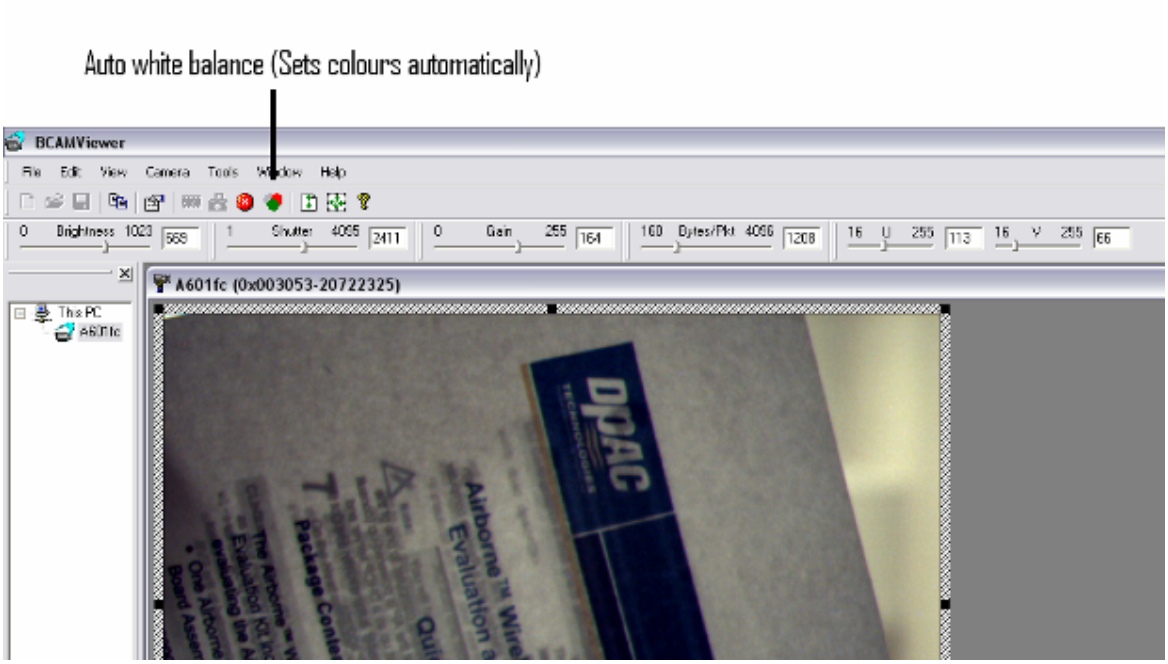


Figure 4.10: Bcam screen shot

Once the image is set so that the user can see all the necessary detail, the settings are read from the program shown in Figure 4.10, and another program transfers the settings to the camera driver on the computer. The program used to set the driver file of the computer is Amcap, which allows the user to set the area if interest as well as the colour balance but the U (amount of blue) and V (amount of red) values must be read from the BCAM Viewer program as mentioned earlier. The steps to be followed are shown in the following series of images.

Figure 4.11: Filter driver setting, step one

Once the Amcap program has been opened the options tab must be selected where the video capture filter settings must be selected as shown in Figure 4.11. This will bring up the video driver properties shown in Figure 4.12, and in this window the settings must be set to the same as those in the Bcam program. In Figure 4.10, the shutter setting can be changed to suit the user, as shown in Figure 4.13. Once the settings have been set the camera will filter the image the same way until the user changes the settings. Therefore if the camera moves to an area with less light it would not self-adjust and the image would appear darker. It is possible to set the camera to let in more light so that the different areas of light do not have such a significant effect on the image. Figure 4.12 shows how to set the camera resolution.

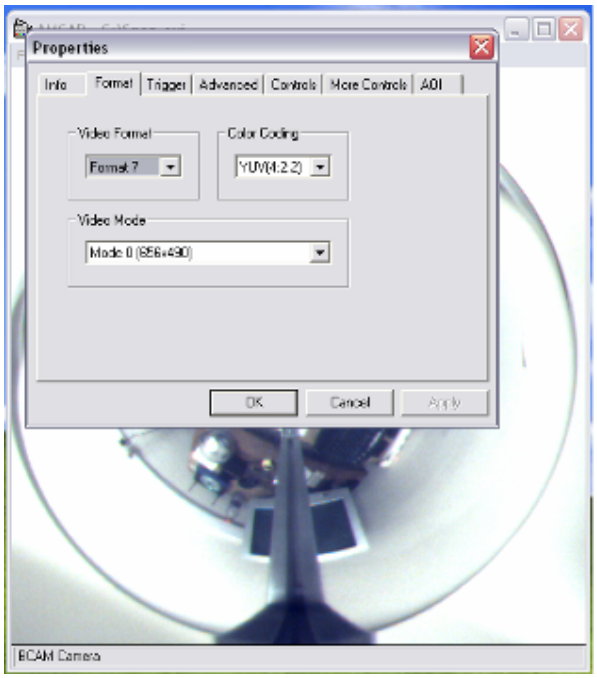


Figure 4.12: Camera resolution setting, step two

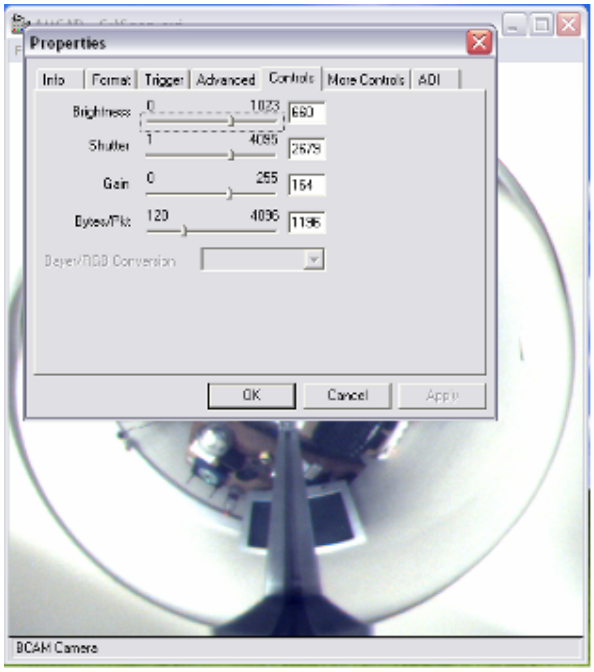


Figure 4.13: Light balance setting, step three

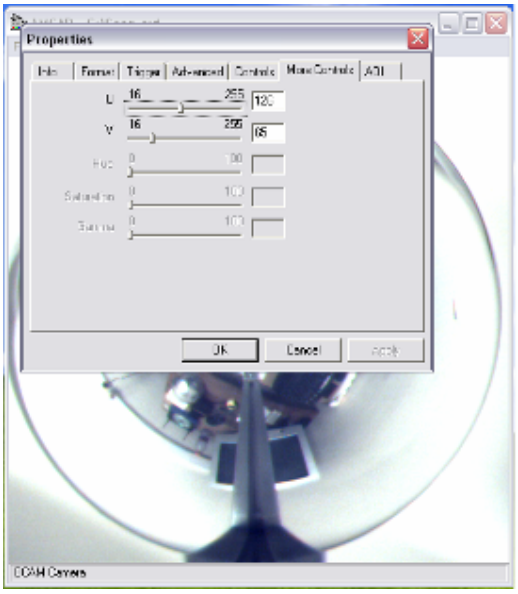


Figure 4.14: Colour balance setting, step four

Figure 4.15 shows how the area of interest is set.

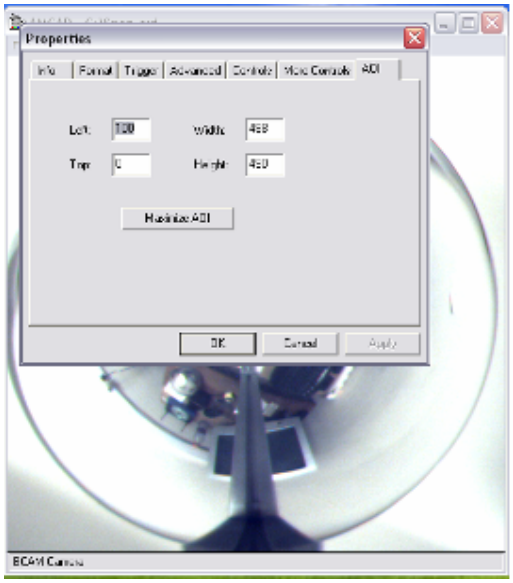


Figure 4.15: Area of interest setting, step five

In the case of the project, the image starts 100 pixels from the left of the image, which is set in the textbox labelled **Left**. It then spans 488 pixels to the right which is set in the textbox labelled **Width**.

4.6 Image processing

When looking at the image in Figure 4.9, it is difficult for the human eye to determine what is being seen. One of the goals of the project was to process the image so that a panoramic image could be extracted from the round image of the camera. It represents everything that is being viewed by the camera. To accomplish this, the pixels that make up the image would have to be relocated to form an entirely new image. The figures below show the theory behind a possible solution to transforming a camera image into a panoramic image. The coloured dots in Figure 4.16 represent the pixels that make up the camera image. The pixels then have to be relocated from their positions to the new positions shown in Figure 4.17.

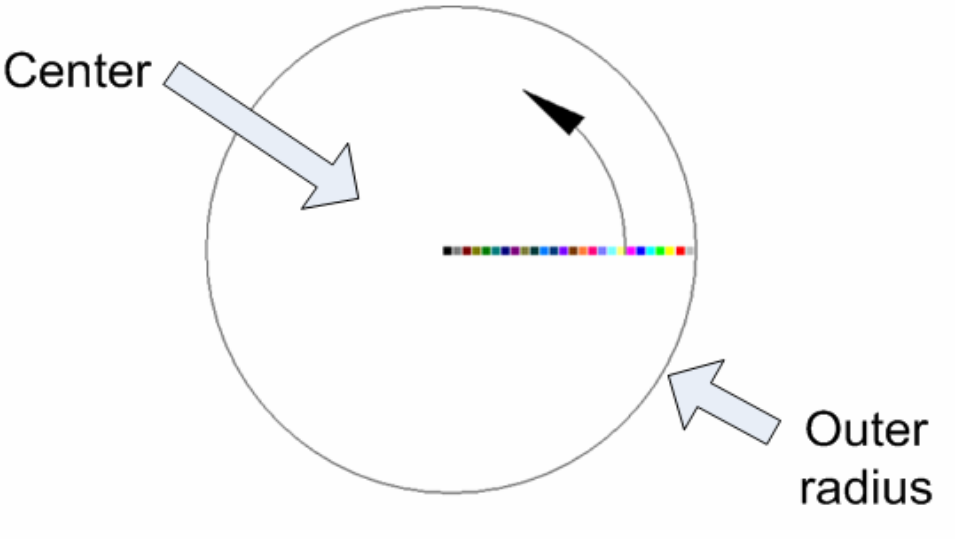


Figure 4.16: Camera pixel view

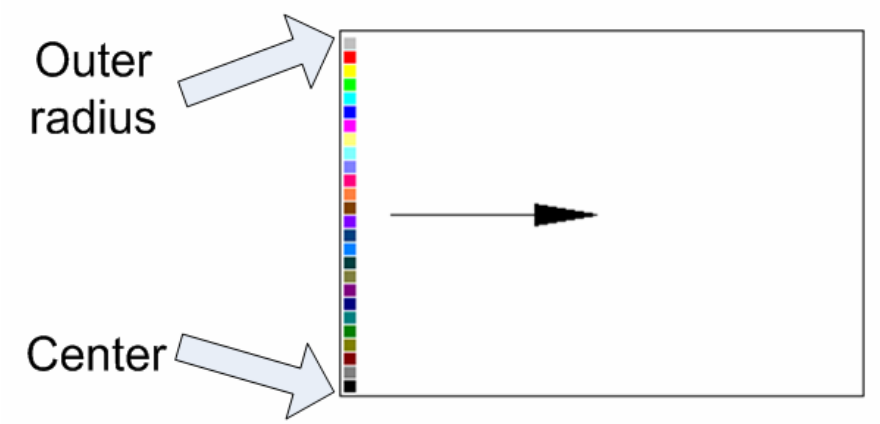


Figure 4.17: Predicted image after processing

4.7 Using MATLAB® to test the theory of producing a panoramic image

To test the theory discussed in Section 4.6, a program was written using MATLAB® [2] to relocate the pixels of an image to form a new panoramic image. A still image of the dome was taken and the centre of the image determined, and from there the pixels were moved as discussed in Section 4.6. The program which was developed is shown in Appendix A, and the results of the image processing are given in Section 5.9, Figures 5.3 and 5.4. The test proved that the theory is a viable solution to the problem, and the next challenge was to make the images stream so that they represent a 360^{o} image in real time.

4.8 Software

To combine all the newly designed systems, a software program was written in C# [3]. The program accesses the serial port for receiving feedback from the AGV and sends commands

to the AGV's controller to manipulate its movements. The IEEE 1394 port is also accessed with the program so that the images from the camera can be received and processed into a panoramic image in real time. Figure 4.18 shows a screen shot of the software written. In the figure there are three additional labels: ***Sensor values*** shows where the feedback of the proximity sensors can be seen. ***Wheel position*** indicates where the feedback from the wheel sensors can be seen. As the AGV moves a path is drawn on the screen, which indicates how the AGV has moved in space. ***Processed image*** indicates where the new processed image is rendered on the screen. The software also allows the user to drive the AGV manually by using the NUM PAD of a standard keyboard. On the NUM PAD button 8 moves the AGV forward, 4 turns it left, 6 turns it right and 2 makes it reverse. If all the buttons are released a command is sent to the AGV to stop. Appendix B shows the code used for the image processing.

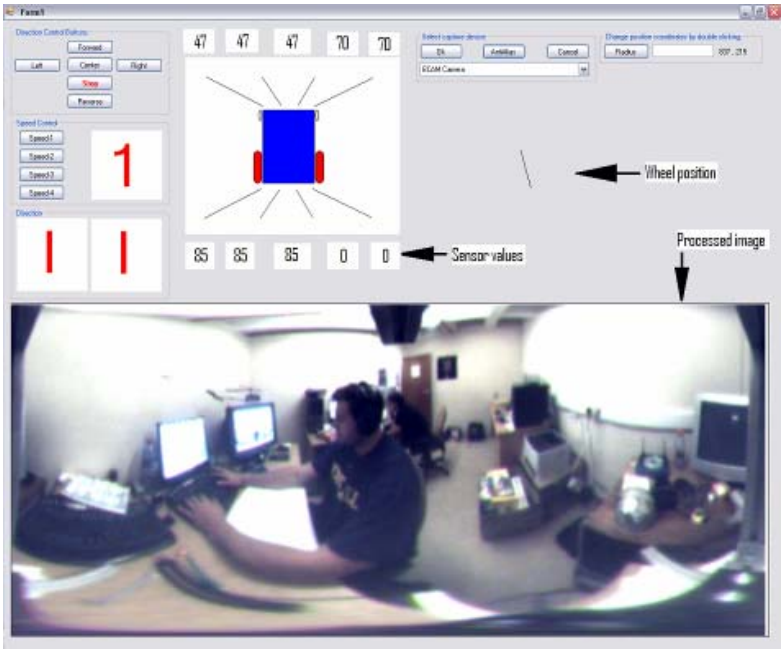


Figure 4.18: Screen shot of written software

One of the features that was added to the software is an anti-alias filter which smoothes out the detail in the image so that future image processing can be done, for example edge detection and image recognition. Figure 4.19 shows the effect of the anti-alias filter on the processed image.

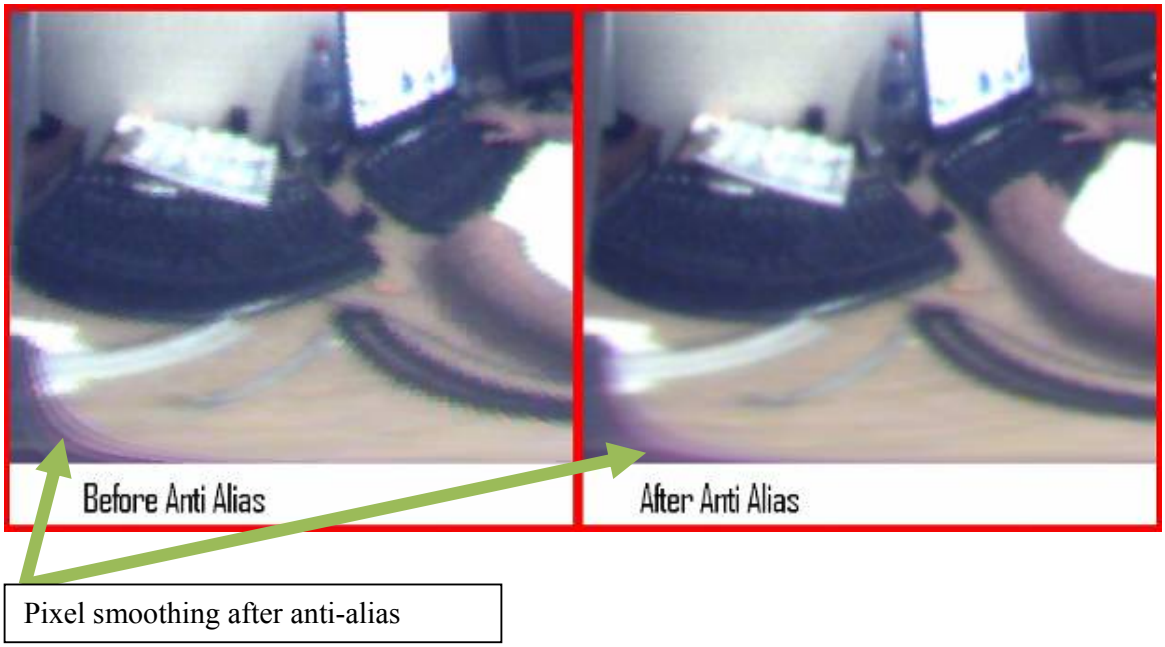


Figure 4.19: Anti-alias filter

4.9 Using the camera system as a sensor

The software discussed in the above section was written so that movement around the AGV and the extent of the movement taking place can be picked up. The sensing ability of the camera is triggered using the sensors mounted on the front in the middle of the AGV. When the AGV gets too close to an obstacle, the sensors trigger the motion detection and stops the AGV, which ensures that a collision with a person in the way will not occur. The AGV will

not continue driving until the movement has stopped and the sensors are clear. To detect movement an image is taken and then set as a background. The next image taken is then placed over the previous image and the two are subtracted from one another. The pixels that change are then filled in with red pixels, showing where the moment in the image is taking place. This is better understood from Figure 4.20.

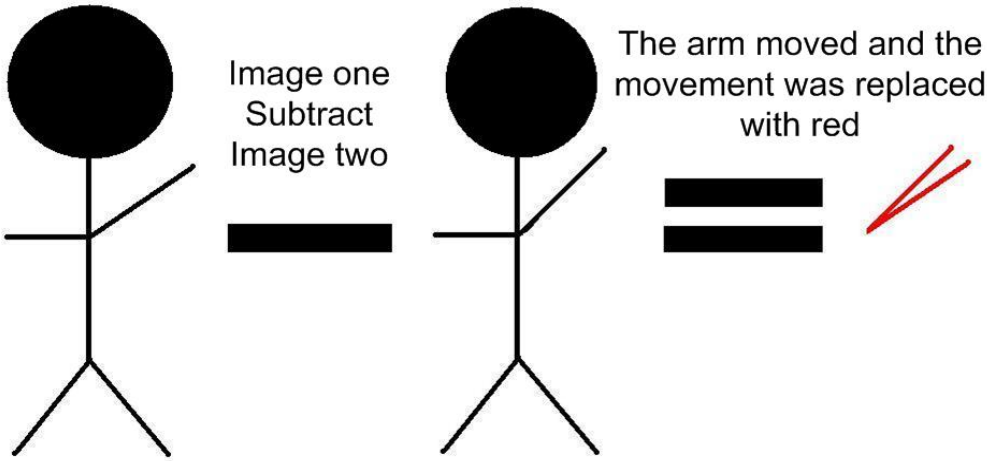


Figure 4.20: Motion detection method

The pixels that change are counted and this gives the amount of movement that is taking place in the image. If there is a large amount of movement it is possible that many objects are moving around the AGV, or simply that there is a single object moving in close proximity to the AGV. Figure 4.21 shows movement in the image as well as the amount of movement as indicated by the arrows in the figure. The code is given in Appendix B.

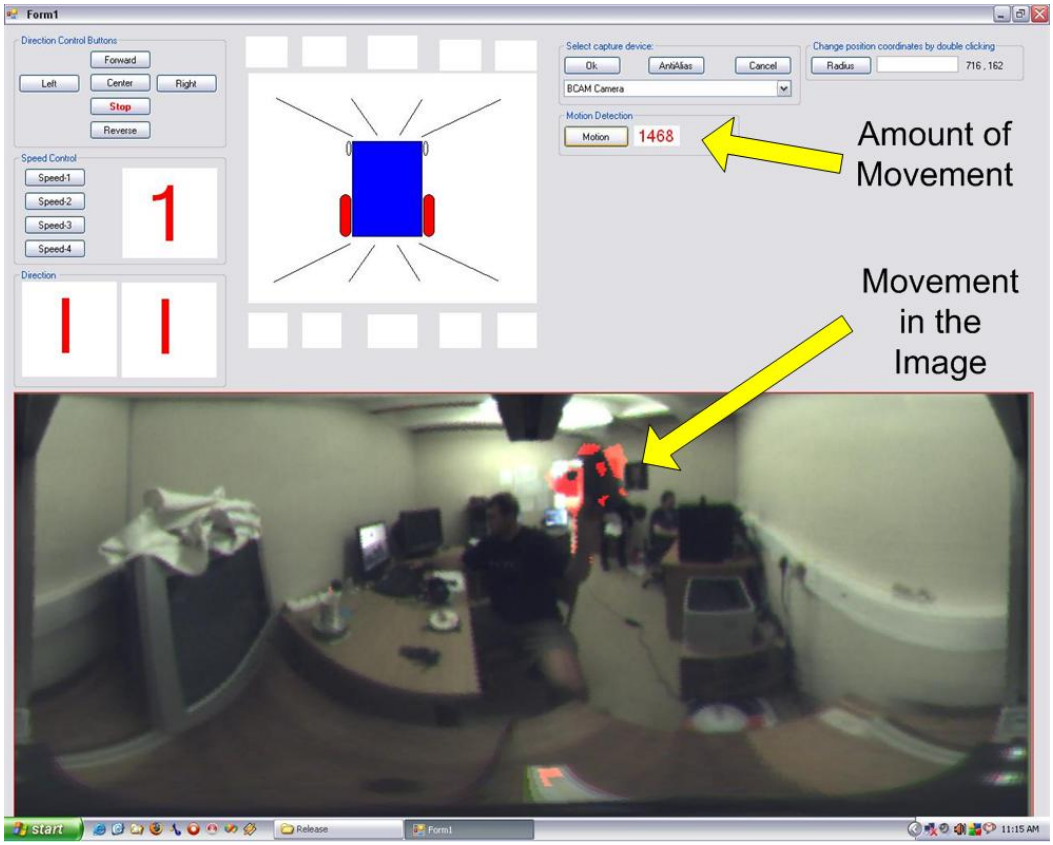


Figure 4.21: Screen print of motion detection in the software developed

The software was written with a threshold level on the amount of movement in the image. This threshold depicts an amount of pixels changed from one frame to another. Once the amount of movement drops below that threshold level the AGV continues along its path, from a standstill position.

4.10 Wireless communication

It is possible to use the wireless module discussed in Chapter 3 to control the AGV, although it cannot be used to replace the IEEE 1394 communication. This problem is solved by placing

a laptop computer on the AGV to run the software discussed in Section 4.7. This allows the software to use the serial and IEEE 1394 ports of the laptop computer while transmitting all the information on the laptop's monitor to a remote computer through the wireless LAN. To connect from a remote computer to the laptop on the AGV, Windows XP's "Remote desktop connection" program was used.

4.11 References

1. BASLER VISION TECHNOLOGIES, 2008, Brochures and Data Sheets [online], Available at http://www.baslerweb.com/indizes/download_index_en_17852.html, [Accessed 1 April 2009]

2. User's Guide, Image Processing Toolbox, Version 4, The Mathworks, May 2003

3. Deitel, H.M., Deitel, P.J., 2006, Visual C#® 2005, How to program, Second Edition, Pearson International, London

Chapter 5

Results of AGV platform tests

This chapter discusses the tests on the AGV which were performed in a passage, simulating real-life scenarios. There were many objects that would influence the AGV's reactions to its surroundings. Examples are students suddenly crossing its path or a dustbin in the way making the passage irregularly shaped, etc. [1, p.22].

5.1 Driving results

For final testing the AGV was placed in a straight passage with no doors and ending in a T-junction. The passage walls were 1900mm apart and the passage was 11 metres long (Figure 5.1). The passage was used as it resembles a scenario that this specific AVG would have to contend with.

Figure 5.1: Passage used for testing

The AGV was set to run in a straight line with its object avoidance sensors on to prevent it from colliding with the walls along the way. The AGV moves at a speed of one metre every 15 seconds when it is set to speed level one. It was placed in the passage as shown in Figure 5.1 and programmed to drive down the passage using the sensors mounted on it to make any corrections to its path if needed. Figure 5.2 shows how the values in Table 5-1 were compiled.

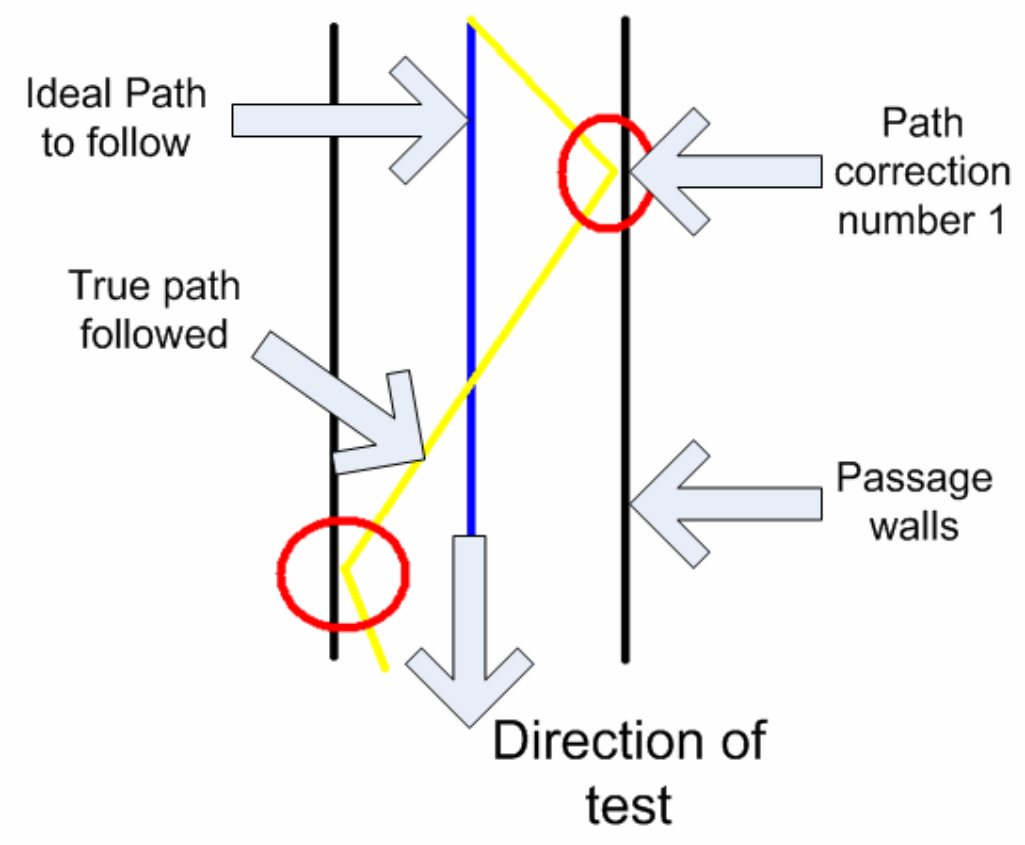


Figure 5.2: Explanation of test performed

The track correction time was calculated by taking the time to complete the ideal path and subtracting it from the time to complete the true path.

The AGV's time to cover the 11 metres varied very little over five tests (see Table 5-1).

Table 5-1: Results over five runs

Run number	**Time to complete run (minutes)**	**Number of track corrections**	**Track correction time (seconds)**	**Percentage time to correct track error**
1	3	11	15	9%
2	3:15	10	30	18%
3	3:12	11	27	16%
4	2:58	12	13	7%
5	3:14	12	29	17.5%
Average	3:04	11	23	13%

An observation reflected that when the speed was set to double the original test speed, the number of track corrections reduced by a half. The reason for this is that the AGV tends to drift to the right and when the forward speed is increased the drift speed remains the same,

reducing the movement to the right. The movement is a result of the reference voltage setting in the digital-to-analog interface circuit. The reference voltage is needed to simulate the joystick's output voltages as discussed in Section 3.2.1.If it is set too low it tends to drift to the left and if it is set too high it tends to drift to the right. If there was movement the AGV would not resume its movement until all movement around it had stopped.

5.2 Decision priority

The AGV's software was developed to give priority to the right, which means that if the AGV were to drive into a corner and the sensors all returned the same information, the software program would make the decision to turn right regardless. The AGV was placed in a position where it would drive directly into a corner, and when its resulting actions were examined, it was found that of the ten times it was tested the AGV turned to the right each time and managed not to get stuck.

5.3 Positioning system

The AGV was driven manually to ensure the best results. First it was driven in a straight line for 5 metres and the wheels were turned so that both spokes would trigger the sensors at the same time at the beginning of the test. It was found that the system over five tests is as follows (Table 5-2).

Table 5-2: Positioning system test

Number of runs	Distance travelled according to software	Distance travelled (measured)	% error
1	5000 mm	4960 mm	0.8 %
2	5000 mm	5040 mm	0.8 %
3	5000 mm	5020 mm	0.4 %
4	5000 mm	4990 mm	0.2 %
5	5000 mm	4970 mm	0.6 %

It was found that if the AGV was sent a stop command after having travelled the five metres after testing, the momentum of the AGV before it stopped caused the error in the results table above. If the wheels were not moved into the correct starting position the error would increase, but by no more than 1%.

5.4 Accuracy of ultrasonic sensors

A feature of the ultrasonic sensors is that the individual distance readings of each sensor and the reading of the sensor closest to an obstacle are displayed. This reading changes every centimetre as the obstacle moves closer until it reaches one meter, and then it starts to change every millimetre. The sensor starts to pick up an obstacle 1.5 meters away. Table 5-3 shows the accuracy of the sensors.

Table 5-3: Sensor accuracy

Number of readings	Distance of obstacle from sensor according to software	Measured distance of obstacle from sensor
1	1.2 m	1.2 m
2	1 m	1.1 m
3	96 cm	96 cm
4	94 cm	92 cm
5	49 cm	49 cm

5.5 Strain of image transformation on processor

The system performance was monitored with Windows Task Manager when the image was not being transformed. With an Intel(R) Core(TM) 2 Duo 3GHz CPU the average usage is 3% when the image is not being transformed. When the image is being processed the CPU usage rises to 62%, and when anti-alias is applied to the processed image the usage rises a further 2% [2].

5.6 Effects of light on camera image

The balance of light entering the camera is essential for a clear image. When the camera system moved into an area with less light the image became substantially darker, and to correct this problem an automatic gain control program was written for the camera to

improve images. An alternate solution could be an external lighting source mounted on the AGV that supplements the light when it gets darker [3, p.45].

5.7 Effect of image processing on the software written

When the image processing is turned off the information about the AGV on the screen updates instantly, whereas when the image processing is on the information takes approximately three seconds to update. The software uses the readings seen on the screen to plot the path travelled and to avoid obstacles. Those processes still function correctly regardless of whether the image processing is on or off. This means that the information still updates in the program, and that the screen is only updated every few seconds.

5.8 Effects of the building on wireless signal range

Driving the AGV through the building showed the different effects of the building's construction on the signal quality of the wireless LAN. In places where the wall structures are 28cm thick and reinforced the signal quality dropped from -31dBm to -71dBm. In places where electric cables are embedded in the walls the signal strength became very weak, and dropped down to -91dBm. In an industry application where there are many electrical devices that generate electrical noise, precise detail would have to be obtained for the positioning of the wireless receivers and transmitters.

5.9 The resulting image after transformation using MATLAB®

Figure 5.3 shows the image received from the camera facing upwards towards the round reflective dome.

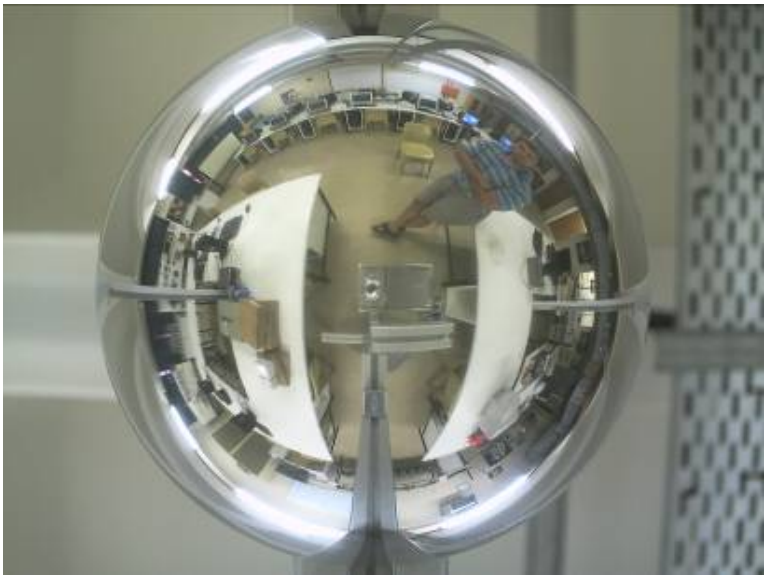


Figure 5.3: Image from camera before transformation

Figure 5.4 shows the image in Figure 5.3 after it was processed using the polar transformation.

Figure 5.4: Image from camera after transformation

The MATLAB® [4] software was used to develop the algorithms for the software to be written in C# language. The time taken to complete the image processing in the MATLAB® software was found to be too long without using the C-compiler. The C# software utilised the same image transformation process developed in the MATLAB® software. The MATLAB® algorithm was originally developed for still images only.

5.10 References

1. Saedan M., Wang Lim C., Marcelo H., 2006, Omnidirectional Image Matching for Vision-Based Robot Localisation, Department of Mechanical Engineering National University of Singapore

2. Intel(R) Core(TM)2 Duo 3GHz CPU, 2008, Datasheet [online], Intel, Available at http://download.intel.com/design/processor /datashts/313278.pdf, [Accessed 8 June 2009]

3. Viljoen V., Vermaak H.J., 2008 Vision-Guided Robotics in a Reconfigured Environment for an Integrated Component-Handling Platform, 2nd Robotics & Mechatronics Symposium, ISBN: 978-0-620-42463-9

4. User's Guide, Image Processing Toolbox, Version 4, The Mathworks, May 2003

Chapter 6

Summary and conclusions

The electric wheelchair was successfully transformed into an controllable automatic guided vehicle platform. The principles that emerged from this study can now be further developed and applied to develop a more efficient vehicle.

6.1 Summary

- Chapter 1

  An introduction to the study was given and the steps to be followed were discussed.

- Chapter 2

  The different technologies to be investigated were discussed and researched.

- Chapter 3

  The process of transforming an electric wheelchair into a platform on which an AGV could be based and controlling it wirelessly was discussed. Receiving information from sensors mounted on the platform to gather information on the AGV's surroundings was investigated.

- Chapter 4

  The fabrication of a round reflective mirror and the camera settings and the theory

behind the image processing were discussed.

- Chapter 5

  The tests on the AGV and some of its components were discussed and the results of the tests were shown.

6.2 Recommendations for further research

The aim of the project was to construct a stable, working AGV with an effective object avoidance system. There is room for improvement and it is possible to make the AGV more flexible in carrying out its tasks.

6.2.1 Sensor accuracy and reaction speed of the software

The sensors used were somewhat inaccurate for this application, and the track error correction time would be faster if they were as accurate as perhaps a laser sensor. The software would react quicker if the sensor information were accurate and would refresh more regularly.

6.2.2 Information transmission

The information transferred from the AGV to the controlling computer slows down as the image processing is turned on. If the image information could be compressed the transfer rate would increase and it would be easier for an operator to understand the problem from a

remote location.

6.2.3 Positioning system

The positioning system would be improved if the gaps between the spokes were smaller or if there were some sort of shaft encoder on the wheel axle to keep better track of where the AGV was.

6.2.4 Image recognition

It would be possible to remove most of the sensors on the AGV if image processing were improved so that obstacles and walls could be recognised and their distance from the AGV measured. The software could then make decisions to move the AGV around or away from the path interference.

6.3 Original contribution of this study

The accomplishments of this project can be summarised as follows:

- An AGV platform was developed
- An interface was built using a microcontroller to communicate control commands from a controlling computer to the platform
- Ultrasonic sensors were mounted on the platform for object avoidance and an interface was built to communicate the sensor information to a remote computer using

a microcontroller

- Sensors were mounted on the wheels and a program was written to determine where the AGV is in space
- Wireless modules were configured to communicate between the controlling computer and the AGV
- A reflective dome mirror was fabricated and installed as the vision system
- A camera was mounted facing the dome mirror to produce an image that shows the entire room
- A software program was written to transform the image from the camera into a panoramic image
- The entire system was tested over a wireless network and found to be working.

Appendix A

MATLAB® program steps to process a still image

```
clc % clear command window
clear % clear variables
A = imread('C:\My Documents\M studente\Piet\phot0003.jpg');
%dsc07356.jpg'); select picture for processing
A = A(1:2447,1:3263,:);%1436,1:1467,:); % select only a selected area
(y,x,:)  (26:744,8:720,:)
%B = A(45:200,30:100,:);
figure, imshow(A)

%I = rgb2gray(A);  %convert to black and white
%threshold = graythresh(I)*0.1; %get the theshold
%BW = im2bw(I,threshold);
%dim = size(BW);
%col = round(dim(2)/2)-200;
%row = find(BW(:,col),1);
%connectivity = 4;
%num_points = 18000;
%contour = bwtraceboundary(BW, [row, col], 'N', connectivity, num_points);
%++++++++++++
%figure, imshow(BW);
% calculate the location of the center and the radius
xc = 1552;%741;
yc = 1268;%703;
radius  = 1100;% 700;

% display the calculated center
%plot(xc,yc,'yx','LineWidth',2);
hold on;
plot(round(xc),round(yc),'yx','LineWidth',2);  %yellow centre
plot(round(xc+radius),round(yc),'r+','LineWidth',2); % red end

%plot the entire circle

% use parametric representation of the circle to obtain coordinates
% of points on the circle
%Xfit = radius*cos(theta) + xc;
%Yfit = radius*sin(theta) + yc;

%xxxxx Start trans xxxxxxxxx
%B = A(1:200,1:200,:);  % y x centre position

startradius = round(radius);%yc);
stopradius = 0;%round(yc-radius);
degrees = 0.1;
stopang = round(360/degrees);
for thetac = 0:1:stopang      %148
    rst = 0;
```

```
for rsteps = startradius:-1:stopradius  %rsteps = startradius:-1:stopradius
 Ypix = round((rsteps*sin(thetac*degrees/180*pi))+(yc));
 Xpix = round((rsteps*cos(thetac*degrees/180*pi))+(xc));
 %plot(Xpix,Ypix,'c.','LineWidth',1);
 %pause;
 %tranf((radius+1)-rsteps,thetac+1,:)=A(Ypix,Xpix,:);
 %upright  rst = rst + 1;
 %tranf((startradius-rsteps)+1,(stopang-thetac)+1,:)=A(Ypix,Xpix,:);
%upright mirror tranf(rst,(stopang-thetac)+1,:)=A(Ypix,Xpix,:);
 %upright mirror
    end
end
figure
imshow(tranf)
```

Appendix B

C# program written for the controlling computer

```
publicvoid ProcessFrame(refBitmap b)
        {



if (Form1.mot)
            {
if (backgroundFrame1 == null)
                {
// create initial backgroung image
                    backgroundFrame1 = grayscaleFilter.Apply(b);

// get image dimension
                    width = b.Width;
                    height = b.Height;

// just return for the first time
return;
                }

Bitmap tmpImage;



// apply the grayscale file
                tmpImage = grayscaleFilter.Apply(b);

// set backgroud frame as an overlay for difference filter
                differenceFilter.OverlayImage = backgroundFrame1;

// apply difference filter
Bitmap tmpImage2 = differenceFilter.Apply(tmpImage);



                // lock the temporary image and apply some filters on the
                locked data
                bitmapData = tmpImage2.LockBits(newRectangle(0, 0, width,
                height),
ImageLockMode.ReadWrite, PixelFormat.Format8bppIndexed);

// threshold filter
                thresholdFilter.ApplyInPlace(bitmapData);


// erosion filter
Bitmap tmpImage3 = erosionFilter.Apply(bitmapData);
```

```
// unlock temporary image
                tmpImage2.UnlockBits(bitmapData);
                tmpImage2.Dispose();

// calculate amount of changed pixels
                whitecount = CalculateWhitePixels(tmpImage3);


// dispose old background
                backgroundFrame1.Dispose();
// set backgound to current
                backgroundFrame1 = tmpImage;

// extract red channel from the original image
Bitmap redChannel = extrachChannel.Apply(b);

//  merge red channel with moving object
                mergeFilter.OverlayImage = tmpImage3;
Bitmap tmpImage4 = mergeFilter.Apply(redChannel);
                redChannel.Dispose();
                tmpImage3.Dispose();

// replace red channel in the original image
                replaceChannel.ChannelImage = tmpImage4;
Bitmap tmpImage5 = replaceChannel.Apply(b);
                tmpImage4.Dispose();

                b.Dispose();
                b = tmpImage5;

            }
```

The section above is the code for the motion detection

```
int nWidth = b.Width;
int nHeight = b.Height;

FloatPoint[,] fp = newFloatPoint[nWidth, nHeight];

Point[,] pt = newPoint[nWidth, nHeight];
Point mid = newPoint();
            mid.X = nWidth / 2;
            mid.Y = nHeight / 2;
double angel, offsetangel;

int x = 0, y = 0;

double newX, newY;



            offsetangel = (Math.PI * -90) / 180;


for (double angell = 0; angell < 486; ++angell)
```

```
            {
for (double radius = 0; radius < (nHeight - 1); ++radius)
                {

                    y = Convert.ToInt16(((nHeight - 1) - radius));
                    x = Convert.ToInt16(angell);

                    angel = -(((Math.PI * (angell / 1.35)) / 180) +
                    offsetangel);


                    newX = mid.X + ((radius / 2) * Math.Cos(angel));
if (newX > 0 && newX < nWidth)
                    {
                        fp[x, y].X = newX;
                        pt[x, y].X = (int)newX;
                    }
else
                        fp[x, y].X = pt[x, y].X = x;


                    newY = (mid.Y + ((radius / 2) * Math.Sin(angel)));
if (newY > 0 && newY < nHeight)
                    {
                        fp[x, y].Y = newY;
                        pt[x, y].Y = (int)newY;
                    }
else
                        fp[x, y].Y = pt[x, y].Y = y;
```

The section above is the code for the image processing

```
                }

            }
if (Form1.Allis)
            {

                OffsetFilterAntiAlias(b, fp);

            }
else
            {
                OffsetFilterAbs(b, pt);

            }

        }



// Calculate white pixels
privateint CalculateWhitePixels(Bitmap b)
        {
int count = 0;
```

```
// lock difference image
BitmapData data = b.LockBits(newRectangle(0, 0, width, height),
ImageLockMode.ReadOnly, PixelFormat.Format8bppIndexed);

int offset = data.Stride - width;



unsafe
            {

byte* ptr = (byte*)data.Scan0.ToPointer();




for (int y = 0; y < height; y++)
                {
for (int x = 0; x < width; x++, ptr++)
                    {
                        count += ((*ptr) >> 7);

                    }
                    ptr += offset;
                }
            }
// unlock image
            b.UnlockBits(data);

return count;
        }



publicstaticbool OffsetFilterAntiAlias(Bitmap b, FloatPoint[,] fp)
        {
Bitmap bSrc = (Bitmap)b.Clone();

// GDI+ still lies to us - the return format is BGR, NOT RGB.
BitmapData bmData = b.LockBits(newRectangle(0, 0, b.Width, b.Height),
ImageLockMode.ReadWrite, PixelFormat.Format24bppRgb);
BitmapData bmSrc = bSrc.LockBits(newRectangle(0, 0, bSrc.Width,
bSrc.Height), ImageLockMode.ReadWrite, PixelFormat.Format24bppRgb);

int scanline = bmData.Stride;

            System.IntPtr Scan0 = bmData.Scan0;
            System.IntPtr SrcScan0 = bmSrc.Scan0;

unsafe
            {
byte* p = (byte*)(void*)Scan0;
byte* pSrc = (byte*)(void*)SrcScan0;

int nOffset = bmData.Stride - b.Width * 3;
int nWidth = b.Width;
```

```
int nHeight = b.Height;

double xOffset, yOffset;

double fraction_x, fraction_y, one_minus_x, one_minus_y;
int ceil_x, ceil_y, floor_x, floor_y;
Byte p1, p2;

for (int y = 0; y < nHeight; ++y)
                {
for (int x = 0; x < nWidth; ++x)
                    {
                        xOffset = fp[x, y].X;
                        yOffset = fp[x, y].Y;

// Setup

                        floor_x = (int)Math.Floor(xOffset);
                        floor_y = (int)Math.Floor(yOffset);
                        ceil_x = floor_x + 1;
                        ceil_y = floor_y + 1;
                        fraction_x = xOffset - floor_x;
                        fraction_y = yOffset - floor_y;
                        one_minus_x = 1.0 - fraction_x;
                        one_minus_y = 1.0 - fraction_y;

if (floor_y >= 0 && ceil_y < nHeight && floor_x >= 0 && ceil_x < nWidth)
   {
// Blue

p1 = (Byte)(one_minus_x *(double)(pSrc[floor_y * scanline + floor_x * 3]) +
fraction_x * (double)(pSrc[floor_y * scanline + ceil_x * 3]));

p2 = (Byte)(one_minus_x * (double)(pSrc[ceil_y * scanline + floor_x * 3]) +
fraction_x * (double)(pSrc[ceil_y * scanline + 3 * ceil_x]));

p[x * 3 + y * scanline] = (Byte)(one_minus_y * (double)(p1) + fraction_y *
(double)(p2));

// Green

p1 = (Byte)(one_minus_x * (double)(pSrc[floor_y * scanline + floor_x * 3 +
1]) +
fraction_x * (double)(pSrc[floor_y * scanline + ceil_x * 3 + 1]));

p2 = (Byte)(one_minus_x * (double)(pSrc[ceil_y * scanline + floor_x * 3 +
1]) +
fraction_x * (double)(pSrc[ceil_y * scanline + 3 * ceil_x + 1]));

p[x * 3 + y * scanline + 1] = (Byte)(one_minus_y * (double)(p1) +
fraction_y * (double)(p2));

// Red

p1 = (Byte)(one_minus_x * (double)(pSrc[floor_y * scanline + floor_x * 3 +
2]) +
fraction_x * (double)(pSrc[floor_y * scanline + ceil_x * 3 + 2]));
```

```
p2 = (Byte)(one_minus_x * (double)(pSrc[ceil_y * scanline + floor_x * 3 +
2]) +
fraction_x * (double)(pSrc[ceil_y * scanline + 3 * ceil_x + 2]));

p[x * 3 + y * scanline + 2] = (Byte)(one_minus_y * (double)(p1) +
fraction_y * (double)(p2));
                        }
                    }
                }
            }

            b.UnlockBits(bmData);
            bSrc.UnlockBits(bmSrc);



returntrue;
        }

/***********************************/


publicstaticbool OffsetFilterAbs(Bitmap b, Point[,] offset)
        {
Bitmap bSrc = (Bitmap)b.Clone();

// GDI+ still lies to us - the return format is BGR, NOT RGB.
BitmapData bmData = b.LockBits(newRectangle(0, 0, b.Width, b.Height),
ImageLockMode.ReadWrite, PixelFormat.Format24bppRgb);
BitmapData bmSrc = bSrc.LockBits(newRectangle(0, 0, bSrc.Width,
bSrc.Height), ImageLockMode.ReadWrite, PixelFormat.Format24bppRgb);


int scanline = bmData.Stride;

            System.IntPtr Scan0 = bmData.Scan0;
            System.IntPtr SrcScan0 = bmSrc.Scan0;






unsafe
            {
byte* p = (byte*)(void*)Scan0;
byte* pSrc = (byte*)(void*)SrcScan0;

int nOffset = bmData.Stride - b.Width * 3;
int nWidth = b.Width;
int nHeight = b.Height;


int xOffset, yOffset;


for (int y = 0; y < nHeight; ++y)
                {
```

```
for (int x = 0; x < nWidth; ++x)
                    {
                        xOffset = offset[x, y].X;
                        yOffset = offset[x, y].Y;

if (yOffset >= 0 && yOffset < nHeight && xOffset >= 0 && xOffset < nWidth)
                        {
p[0] = pSrc[(yOffset * scanline) + (xOffset) * 3];
p[1] = pSrc[(yOffset * scanline) + (xOffset * 3) + 1];
p[2] = pSrc[(yOffset * scanline) + (xOffset * 3) + 2];
                        }

                        p += 3;
                    }
                    p += nOffset;
                }
            }

            b.UnlockBits(bmData);
            bSrc.UnlockBits(bmSrc);

returntrue;

        }


    }
}
```

The section of code above places the pixels in their new positions

Appendix C

Assembler program of the microcontroller that collects information from the ultrasonic sensors

```
 #INCLUDE         <P16F627.INC>
 LIST        P=16F627

CBLOCK 0X20

            REG
            WTEMP
            STATEMP
            D1
            D2
            D3
            COUNT
       HOLDER
            PPS1
            PPS2
            PPS3
            PPS4
            PPS5
            PPS6
            PPS7
            SENT
            TEST

ENDC

 ORG          0X00

              GOTO       MAIN

 ORG         0x04

              MOVWF      WTEMP                    ;Store W register
              SWAPF      STATUS,W                 ;Context Saving
                   BCF   STATUS,RP0               ;Context Saving
              MOVWF      STATEMP
              BCF        STATUS,Z

              BTFSC      PORTB,0
              CALL       SENDER
              BTFSS      TEST,0
              CALL       RECIEVE
              CLRF       TEST

                   SWAPF STATEMP,W                ;Context Saving
              MOVWF      STATUS                   ;Context Saving
              SWAPF      WTEMP,F                  ;Context Saving
              SWAPF      WTEMP,W                  ;Context Saving
              BCF        PIR1,RCIF
```

```
            BCF         INTCON,INTF

            MOVLW       D'52'
            MOVWF       D3
            MOVLW       D'50'
            MOVWF       D2
            MOVLW       D'5'
            MOVWF       D1

            RETFIE


;;;;;;;;;;;;;;;;;;;;;;INIT;;;;;;;;;;;;;;;;;;;;;;

MAIN
            BCF         STATUS,RP1
            BCF         STATUS,RP0              ;BANK0
                 MOVLW B'10010000'              ;NO PARITY, NO OVERUN
                                                ;ADDRESS DETECTION
                 MOVWF RCSTA                    ;DON'T CARE, 8 BIT

            CLRF        INTCON

            MOVLW       0X07                    ;Turning the comparitors
off
            MOVWF       CMCON

            CLRF        PORTA
            CLRF        SENT

            BCF         STATUS,RP1
            BSF         STATUS,RP0              ;BANK1

            MOVLW       B'00000000'
            MOVWF       TRISA
            MOVLW       B'11111111'
            MOVWF       TRISB

            MOVLW       B'01111111'
            MOVWF       OPTION_REG

            BSF         PIE1,RCIE

            MOVLW       D'25'
                 MOVWF SPBRG            ;SETUP BAUD RATE FOR 9600 WITH 4MHZ
                                        ;INTERNAL CRYSTAL

                                        ;TO BE CONFIGURED AS TX AND RX PINS
                 MOVLW B'00100100'      ;SET TX STATUS NO 9TH BIT,TSR FULL
                 MOVWF TXSTA            ;HIGH ASY MODE,ASY MODE,ENABLE TX,
                                        ;8 BIT TX,SLAVE MODE

            BCF         STATUS,RP1
            BCF         STATUS,RP0  ;BANK0

            CALL        DELAY1
```

```
                BTFSS RCSTA,OERR  ; Important test to see
                GOTO  CARRY       ; if i recieved an error
                BCF   RCSTA,CREN  ; with a multiple string input
            NOP                   ; WILL PREVENT USART
            BSF       RCSTA,CREN


CARRY           MOVLW B'11010000'
            MOVWF     INTCON

            CLRF      TEST
            GOTO      CONTROL

;***********************************************;


RECIEVE         BTFSS RCSTA,OERR            ; Important test to see
                GOTO  YOU                   ; if i recieved an error
            BCF       RCSTA,CREN            ; with a multiple string
input
            NOP                             ; WILL PREVENT USART
                BSF   RCSTA,CREN            ; FROM FREAZING

YOU             BTFSS PIR1,RCIF             ;Before reading the RCREG,
                                            ;check should be made to
                                            ;determine whether new data
                                            ;has been received. When
                                            ;new data in the RCREG
                                            ;the RCIF bit in the PIR1
                                            ;register will be set.

            GOTO      RECIEVE
            BCF       PIR1,RCIF

                MOVF  RCREG,W               ;WEN NEW DATA IS LOADED
            MOVWF     REG



            BTFSS     HOLDER,0
            GOTO      P1
            BTFSS     HOLDER,1
            GOTO      P2
            BTFSS     HOLDER,2
            GOTO      P3
            BTFSS     HOLDER,3
            GOTO      P4
            BTFSS     HOLDER,4
            GOTO      P5
            BTFSS     HOLDER,5
            GOTO      P6
            BTFSS     HOLDER,6
            GOTO      P7

            RETURN

;***********************************************;
```

```
P1              MOVF  REG,W
             MOVWF    PPS1
             BSF      HOLDER,0
             RETURN
P2              MOVF  REG,W
             MOVWF    PPS2
             BSF      HOLDER,1
             RETURN
P3              MOVF  REG,W
             MOVWF    PPS3
             BSF      HOLDER,2
             RETURN
P4              MOVF  REG,W
             MOVWF    PPS4
             BSF      HOLDER,3
             RETURN
P5              MOVF  REG,W
             MOVWF    PPS5
             BSF      HOLDER,4
             RETURN
P6              MOVF  REG,W
             MOVWF    PPS6
             BSF      HOLDER,5
             RETURN
P7              MOVF  REG,W
             MOVWF    PPS7
             BSF      HOLDER,6
             RETURN

;***********************************************;

SENDER          BTFSS PORTB,3
             GOTO     SENDER

             BTFSS    SENT,0
             GOTO     S1
             BTFSS    SENT,1
             GOTO     S2
             BTFSS    SENT,2
             GOTO     S3
             BTFSS    SENT,3
             GOTO     S4
             BTFSS    SENT,4
             GOTO     S5
             BTFSS    SENT,5
             GOTO     S6
             BTFSS    SENT,6
             GOTO     S7

SENDER1         BTFSC PORTB,0
             GOTO     SENDER1
             CLRF     PORTA
             CLRF     SENT
             BSF      TEST,0
             RETURN

S1              MOVF  PPS1,W
```

```
            ;ADDLW      0X30
            MOVWF       PORTA
            BSF         SENT,0
S11              BTFSS  PORTB,3
            GOTO        SENDER
            GOTO        S11

S2               MOVF   PPS2,W
            ;ADDLW      0X30
            MOVWF       PORTA
            BSF         SENT,1
S22              BTFSS  PORTB,3
            GOTO        SENDER
            GOTO        S22

S3               MOVF   PPS3,W
            ;ADDLW      0X30
            MOVWF       PORTA
            BSF         SENT,2
S33              BTFSS  PORTB,3
            GOTO        SENDER
            GOTO        S33

S4               MOVF   PPS4,W
            ;ADDLW      0X30
            MOVWF       PORTA
            BSF         SENT,3
S44              BTFSS  PORTB,3
            GOTO        SENDER
            GOTO        S44

S5               MOVF   PPS5,W
            ;ADDLW      0X30
            MOVWF       PORTA
            BSF         SENT,4
S55              BTFSS  PORTB,3
            GOTO        SENDER
            GOTO        S55

S6               MOVF   PPS6,W
            ;ADDLW      0X30
            MOVWF       PORTA
            BSF         SENT,5
S66              BTFSS  PORTB,3
            GOTO        SENDER
            GOTO        S66

S7               MOVF   PPS7,W
            ;ADDLW      0X30
            MOVWF       PORTA
            BSF         SENT,6
S77              BTFSS  PORTB,3
            GOTO        SENDER1
            GOTO        S77
```

```
;***********************************************;

CONTROL          CLRF  TEST
                 BTFSS RCSTA,OERR        ; Important test to see
                 GOTO  DELAY             ; if i received an error
                 BCF   RCSTA,CREN        ; with a multiple string input
            NOP                    ;      WILL PREVENT USART
                 BSF   RCSTA,CREN        ; FROM FREEZING
            GOTO       CONTROL

;***********************************************;

TX               BTFSC PIR1,TXIF
            GOTO       SEND
            GOTO       TX
SEND             MOVWF TXREG
            RETURN


;***********************************************;

DELAY            MOVLW D'52'
            MOVWF      D3
            MOVLW      D'50'
            MOVWF      D2
            MOVLW      D'5'
            MOVWF      D1
            DECFSZ D1
            GOTO       $-1
            DECFSZ D2
            GOTO       $-5
            DECFSZ D3
            GOTO       $-9
            CLRF       HOLDER
            GOTO       CONTROL

DELAY1           MOVLW D'100'
            MOVWF      D3
            MOVLW      D'50'
            MOVWF      D2
            MOVLW      D'32'
            MOVWF      D1
            DECFSZ D1
            GOTO       $-1
            DECFSZ D2
            GOTO       $-5
            DECFSZ D3
            GOTO       $-9
            RETURN

;***********************************************;

END
```

Appendix D

Assembler program that obtains the information from the microcontrollers that have the ultrasonic sensor and wheel sensor values

```
 #INCLUDE         <P16F877.INC>
 LIST        P=16F877

CBLOCK 0X20

            REG
            WTEMP
            STATEMP
            D1
            D2
            D3
            SORT
            DIRECTIONFB
            DIRECTIONLR
            LEFT
            RIGHT
            WHEEL
ENDC

 ORG         0X00

             GOTO       MAIN

;;;;;;;;;;;;;;;;;;;;;;INIT;;;;;;;;;;;;;;;;;;;;;;

MAIN         BCF        STATUS,RP1
             BSF        STATUS,RP0  ;BANK1

             MOVLW      0X06
             MOVWF      ADCON1

             MOVLW      B'11111111'
             MOVWF      TRISB
             MOVLW      B'11111111'
             MOVWF      TRISD
             MOVLW      B'11110000'
             MOVWF      TRISA       ;BSF  TRISE,4

             BSF        PIE1,RCIE

             MOVLW      D'25'
                  MOVWF SPBRG       ;SETUP BAUD RATE FOR 9600 WITH 4MHZ
                                    ;INTERNAL CRYSTAL
                                    ;TO BE CONFIGURED AS TX AND RX PINS
                  MOVLW B'00100100' ;SET TX STATUS NO 9TH BIT,TSR FULL
                  MOVWF TXSTA       ;HIGH ASY MODE,ASY MODE,ENABLE TX,
```

```
                BCF   STATUS,RP1  ;8 BIT TX,SLAVE MODE
             BCF      STATUS,RP0  ;BANK0
                MOVLW B'10010000' ;NO PARITY, NO OVERUN ERROR, DISABLE
                                  ;ADDRESS DETECTION MODE;DISABLE CON RX
                MOVWF RCSTA       ;DON'T CARE, 8 BIT SELECTION, SERIAL
PORT

             MOVLW    B'00000000'
             MOVWF    INTCON


             CLRF     PORTA
             BSF      SORT,0
             BSF      SORT,1
             CLRF     WHEEL
             CLRF     LEFT
             CLRF     RIGHT


;*********************************************************************
************************************;

RUN               CALL  TEST
             CALL     MOVE
             GOTO     RUN


;*********************************************************************
************************************;


TEST         BTFSS RCSTA,OERR  ; Important test to see
             GOTO  RECIEVE     ; if i received an error
             BCF   RCSTA,CREN  ; with a multiple string input
             NOP               ; WILL PREVENT USART
             BSF   RCSTA,CREN  ; FROM FREEZING
RECIEVE      BTFSS PIR1,RCIF   ;Before reading the RCREG, a
                               ;check should be made to
                          ;determine      whether new data
                               ;has been received. When there is
                               ;new data in the RCREG register,
                               ;the RCIF bit in the PIR1
             RETURN ;register           will be set.

             BCF      PIR1,RCIF

                  MOVF  RCREG,W     ;WEN NEW DATA IS LOADED INTO RCREG
             MOVWF    REG

             BTFSS    SORT,0
             GOTO     FB

             BTFSS    SORT,1
             GOTO     LR


             MOVF     REG,0
             XORLW    '#'
```

```
             BTFSS      STATUS,Z
             RETURN
             CLRF       SORT
             RETURN


FB                MOVF  REG,0
             MOVWF      DIRECTIONFB
             BSF        SORT,0
             RETURN

LR                MOVF  REG,0
             MOVWF      DIRECTIONLR
             BSF        SORT,1
             CALL       SENDER
             RETURN

;*********************************************************************
************************************;
MOVE              BTFSC PORTA,4
             CALL       PORTA4SET

             BTFSS      PORTA,4
             CALL       PORTA4CLEAR

             BTFSC      PORTA,5
             CALL       PORTA5SET

             BTFSS      PORTA,5
             CALL       PORTA5CLEAR

             RETURN

PORTA4SET         BSF WHEEL,  0
             RETURN

PORTA4CLEAR       BTFSC WHEEL,0
             INCF       LEFT
             BCF        WHEEL,0
             RETURN

PORTA5SET         BSF WHEEL,  1
             RETURN

PORTA5CLEAR       BTFSC WHEEL,1
             INCF       RIGHT
             BCF        WHEEL,1
             RETURN

;*********************************************************************
************************************;

SENDER            MOVLW '#'
             CALL       TX
             MOVF       DIRECTIONFB,0
             CALL       TX
             MOVF       DIRECTIONLR,0
             CALL       TX
```

```
BSF         PORTA,0
CALL        DELAY
BSF         PORTA,1
CALL        DELAY

MOVF        PORTB,W
CALL        TX

BCF         PORTA,1
CALL        DELAY
BSF         PORTA,1
CALL        DELAY

MOVF        PORTB,W
CALL        TX

BCF         PORTA,1
CALL        DELAY
BSF         PORTA,1
CALL        DELAY

MOVF        PORTB,W
CALL        TX

BCF         PORTA,1
CALL        DELAY
BSF         PORTA,1
CALL        DELAY

MOVF        PORTB,W
CALL        TX

BCF         PORTA,1
CALL        DELAY
BSF         PORTA,1
CALL        DELAY

MOVF        PORTB,W
CALL        TX

BCF         PORTA,1
CALL        DELAY
BSF         PORTA,1
CALL        DELAY

MOVF        PORTB,W
CALL        TX

BCF         PORTA,0
```

```
BCF         PORTA,1

BSF         PORTA,2
CALL        DELAY
BSF         PORTA,3
CALL        DELAY

MOVF        PORTD,W
CALL        TX

BCF         PORTA,3
CALL        DELAY
BSF         PORTA,3
CALL        DELAY

MOVF        PORTD,W
CALL        TX

BCF         PORTA,3
CALL        DELAY
BSF         PORTA,3
CALL        DELAY

MOVF        PORTD,W
CALL        TX

BCF         PORTA,3
CALL        DELAY
BSF         PORTA,3
CALL        DELAY

MOVF        PORTD,W
CALL        TX

BCF         PORTA,3
CALL        DELAY
BSF         PORTA,3
CALL        DELAY

MOVF        PORTD,W
CALL        TX

BCF         PORTA,3
CALL        DELAY
BSF         PORTA,3
CALL        DELAY

MOVF        PORTD,W
CALL        TX
```

```
            BCF         PORTA,2
            BCF         PORTA,3

            MOVF        LEFT,0
            CALL        TX
            CLRF        LEFT

            MOVF        RIGHT,0
            CALL        TX
            CLRF        RIGHT

            RETURN

;***********************************************;

TX               BTFSC PIR1,TXIF
            GOTO        SEND
            GOTO        TX
SEND             MOVWF TXREG
            RETURN


;***********************************************;

DELAY            MOVLW D'5'
            MOVWF       D3
            MOVLW       D'5'
            MOVWF       D2
            MOVLW       D'5'
            MOVWF       D1
            DECFSZ D1
            GOTO        $-1
            DECFSZ D2
            GOTO        $-5
            DECFSZ D3
            GOTO        $-9
            RETURN
            END
```

Appendix E

Assembler program for the interface between the controlling computer and the platform control

```
 #INCLUDE         <P16F877.INC>
 LIST        P=16F877

CBLOCK 0X20

            REG
            WTEMP
            STATEMP
            D1
            D2
            D3
            DIRFB
            DIRLR
ENDC

 ORG         0X00

             GOTO       MAIN

 ORG        0x04

             MOVWF      WTEMP                   ;Store W register
             SWAPF      STATUS,W                ;Context Saving
                  BCF   STATUS,RP0              ;Context Saving
             MOVWF      STATEMP
             BCF        STATUS,Z

             CALL       RECIEVE

             SWAPF      STATEMP,W               ;Context Saving
             MOVWF      STATUS                  ;Context Saving
             SWAPF      WTEMP,F                 ;Context Saving
             SWAPF      WTEMP,W                 ;Context Saving
             BCF        PIR1,RCIF

             RETFIE

;*********************************************;

RECIEVE           BTFSS RCSTA,OERR        ; Important test to see
                  GOTO  YOU               ; if i received an error
                  BCF   RCSTA,CREN        ; with a multiple string input
             NOP                     ;      WILL PREVENT USART
                  BSF   RCSTA,CREN        ; FROM FREEZING
```

```
YOU              BTFSS PIR1,RCIF         ;Before reading the RCREG, a
                                         ;check should be made to
                                       ;determine      whether new data
                                         ;has been received. When there is
                                         ;new data in the RCREG register,
                                         ;the RCIF bit in the PIR1
                                       ;register      will be set.

             GOTO       RECIEVE

                  MOVF  RCREG,W     ;WEN NEW DATA IS LOUDED INTO RCREG
             MOVWF      REG

             CALL       CONTROL

             RETURN

;;;;;;;;;;;;;;;;;;;;;;;INIT;;;;;;;;;;;;;;;;;;;;;;;

MAIN         BCF        STATUS,RP1
             BSF        STATUS,RP0              ;BANK1

             CLRF       TRISA
             CLRF       TRISB

             BSF        PIE1,RCIE

             CLRF       TRISA

             MOVLW      D'25'
                  MOVWF SPBRG       ;SETUP BOUD RATE FOR 9600 WITH 4MHZ
                                    ;INTERNAL CRYSTAL
                                    ;TO BE CONFIGURED AS TX AND RX PINS
                  MOVLW B'00100100' ;SET TX STATUS NO 9TH BIT,TSR FULL
                  MOVWF TXSTA       ;HIGH ASY MODE,ASY MODE,ENABLE TX,
                  BCF   STATUS,RP1  ;8 BIT TX,SLAVE MODE
             BCF        STATUS,RP0  ;BANK0
                  MOVLW B'10010000' ;NO PARITY, NO OVERUN ERROR, DISABLE
                                    ;ADDRESS DETECTION MODE;DISABLE CON RX
                  MOVWF RCSTA ;DON'T CARE, 8 BIT SELECTION, SERIAL PORT

             MOVLW      0X08
             MOVWF      PORTA
             MOVWF      PORTB

             MOVLW      B'11000000'
             MOVWF      INTCON

             CLRF       DIRFB
             CLRF       DIRLR

             BSF        DIRFB,2
             BSF        DIRLR,2

             GOTO       START1

;***********************************************;
```

```
START1          CALL  DELAY
            CALL      DELAY
            CALL      DELAY
            CALL      DELAY
            CALL      DELAY

            CALL      DELAY1
            MOVLW     '#'
            CALL      TX
            CALL      DELAY1
            MOVLW     'S'
            CALL      TX
            CALL      DELAY1
            MOVLW     'C'
            CALL      TX

START           CALL  DELAY
            CALL      DELAY1
            MOVLW     '#'
            CALL      TX

            BTFSC     DIRFB,0
            CALL      FOR

            BTFSC     DIRFB,1
            CALL      BAC

            BTFSC     DIRFB,2
            CALL      STO

            BTFSC     DIRLR,0
            CALL      LEF

            BTFSC     DIRLR,1
            CALL      RIG

            BTFSC     DIRLR,2
            CALL      CEN

            GOTO      START

;***********************************************;

FOR             CALL  DELAY1
            MOVLW     'F'
            CALL      TX
            RETURN
BAC             CALL  DELAY1
            MOVLW     'B'
            CALL      TX
            RETURN
RIG             CALL  DELAY1
            MOVLW     'R'
            CALL      TX
            RETURN
LEF             CALL  DELAY1
            MOVLW     'L'
            CALL      TX
```

```
             RETURN
STO               CALL  DELAY1
             MOVLW      'S'
             CALL       TX
             RETURN
CEN               CALL  DELAY1
             MOVLW      'C'
             CALL       TX
             RETURN
;***********************************************;

TX                MOVWF TXREG
BEGIN             BTFSS PIR1,TXIF
             GOTO       BEGIN
             RETURN

;***********************************************;


CONTROL           MOVF  REG,0
            XORLW 'A'
            BTFSC STATUS,Z
            CALL  AA

            MOVF  REG,0
            XORLW 'B'
            BTFSC STATUS,Z
            CALL  BB

            MOVF  REG,0
            XORLW 'C'
            BTFSC STATUS,Z
            CALL  CC

            MOVF  REG,0
            XORLW 'D'
            BTFSC STATUS,Z
            CALL  DD

            MOVF  REG,0
            XORLW 'E'
            BTFSC STATUS,Z
            CALL  EE

            MOVF  REG,0
            XORLW 'F'
            BTFSC STATUS,Z
            CALL  FF

            MOVF  REG,0
            XORLW 'G'
            BTFSC STATUS,Z
            CALL  GG

            MOVF  REG,0
            XORLW 'H'
            BTFSC STATUS,Z
            CALL  HH
```

```
MOVF  REG,0
XORLW 'I'
BTFSC STATUS,Z
CALL  II

MOVF  REG,0
XORLW 'J'
BTFSC STATUS,Z
CALL  JJ

MOVF  REG,0
XORLW 'K'
BTFSC STATUS,Z
CALL  KK

MOVF  REG,0
XORLW 'L'
BTFSC STATUS,Z
CALL  LL

MOVF  REG,0
XORLW 'M'
BTFSC STATUS,Z
CALL  MM

MOVF  REG,0
XORLW 'N'
BTFSC STATUS,Z
CALL  NN

MOVF  REG,0
XORLW 'O'
BTFSC STATUS,Z
CALL  OO

MOVF  REG,0
XORLW 'P'
BTFSC STATUS,Z
CALL  PP

MOVF  REG,0
XORLW 'Q'
BTFSC STATUS,Z
CALL  STOPFB

MOVF  REG,0
XORLW 'R'
BTFSC STATUS,Z
CALL  STOPLR

RETURN
```

```
;*********************************************  FORWARD
*********************************************************************;

AA          MOVLW 0X09
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,0
            RETURN

BB          MOVLW 0X0A
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,0
            RETURN

CC    MOVLW 0X0B
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,0
            RETURN

DD          MOVLW 0X0C
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,0
            RETURN



;*********************************************  BACK
*********************************************************************;


EE          MOVLW 0X07
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,1
            RETURN

FF          MOVLW 0X06
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,1
            RETURN

GG    MOVLW 0X05
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,1
            RETURN

HH          MOVLW 0X04
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,1
            RETURN
```

```
;*********************************************  RIGHT
*************************************************************************;

II          MOVLW 0X09
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,1
            RETURN

JJ          MOVLW 0X0A
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,1
            RETURN

KK          MOVLW 0X0B
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,1
            RETURN

LL          MOVLW 0X0C
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,1
            RETURN


;*********************************************  LEFT
*************************************************************************

MM          MOVLW 0X07
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,0
            RETURN

NN          MOVLW 0X06
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,0
            RETURN

OO          MOVLW 0X05
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,0
            RETURN

PP          MOVLW 0X04
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,0
            RETURN
```

```
;*******************************************************************************
*****************************************;

STOPFB MOVLW      0X08
            MOVWF PORTB
            CLRF  DIRFB
            BSF   DIRFB,2
            RETURN

STOPLR MOVLW      0X08
            MOVWF PORTA
            CLRF  DIRLR
            BSF   DIRLR,2
            RETURN

;*******************************************************************************
*****************************************;

DELAY       MOVLW D'100'
             MOVWF      D3
             MOVLW      D'50'
             MOVWF      D2
             MOVLW      D'32'
             MOVWF      D1
             DECFSZ D1
             GOTO       $-1
             DECFSZ D2
             GOTO       $-5
             DECFSZ D3
             GOTO       $-9
             RETURN

DELAY1            MOVLW D'10'
             MOVWF      D3
             MOVLW      D'10'
             MOVWF      D2
             MOVLW      D'10'
             MOVWF      D1
             DECFSZ D1
             GOTO       $-1
             DECFSZ D2
             GOTO       $-5
             DECFSZ D3
             GOTO       $-9
             RETURN

;*******************************************************************************
*****************************************;


END
```

Appendix F

Article for the 2nd Robotics & Mechatronics Symposium

Omni directional image sensing for automated guided vehicle

Piet Swanepoel, Ben Kotze and Herman Vermaak

Department of Electrical & Computer Systems Engineering
Central University of Technology

South Africa
Plande2001@gmail.com

ABSTRACT - *An Automated Guided Vehicle (AGV) equipped with an omni directional imaging camera, giving the ability to view 360 degree area will be presented in this paper. This would give the AGV a unique capability of viewing the surroundings from a remote location. A modified electric wheel chair is used as platform, equipped with ultra sonic sensors and a Wireless Local Area Network (WLAN) for communication.The ultra sonic sensors are used to measure the distances in the AGV's close surroundings. All of these systems combined will produce a vehicle that carry heavy loads while viewing its surroundings, avoid obstacles in its way and be controlled remotely by another computer on the network.*

I INTRODUCTION

Some automated guided vehicles (AGV's) rely on their surroundings for decision making purposes when navigating through obstacles. An array of sensors that surround the AGV can be used for this purpose [1]. In such an application the sensors that are normally incorporated are ultra sound, infra red and touch sensors. Utilizing the surroundings by interpreting it through vision would give an added advantage to be used for navigation as touch or distance do not always give a clear picture of the obstacle in the AGV's path. Similar research utilizing cameras has been done and there are numerous options that become available for navigation of AGV's by having such images available for processing [2].

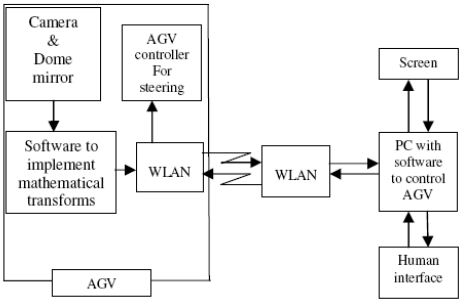


Figure 1: Block diagram of the complete AGV system

The block diagram in Figure 1 depicts the surroundings of the AGV, taken by a single camera facing a round dome mirror. The image is transformed into a panoramic view and then transmitted through a wireless medium like WLAN. This image is then sent to a remote computer, that will analyze the information received, sending the appropriate control instructions to the AGV. The total system will make provision for a human interface for communicating instructions to and information from the AGV that can be viewed as a 360° image interpreted by the AGV.

II AGV DEFINITION

Within this project the AGV is defined as an unmanned, self-propelled vehicle in the nature of a mobile robot, which is equipped with an onboard computer that stores path and machine control instructions for the steering, forward and backward drive of the machine, powered by an electric motor and batteries [3].

III DEVELOPMENT OF AN AGV

The AGV has been integrated into widespread industrial applications since its introduction in the 1950s [4].

There are a variety of guidance systems that are currently used in the steering and guidance of the AGV but initially it was up to a system that utilized a wire placed under the surface that the vehicle was traveling on for guidance, called wire guidance or crudely dubbed "Smart Floors and Dumb Vehicles".

Since there is an ever growing demand for more intelligent AGV's, it is worth investigating and improving their design, so more development will always be required.

IV DEVELOPMENT OF THE PROJECT PROTOTYPE

An Automated Guided Vehicle can be developed from any electrically driven and steered vehicle. Taking this into account a modified electrically controlled wheel chair would be the ideal platform for developing an Automated Guided Vehicle that could carry pay loads of up to two hundred kilograms as seen in Figure 2.

Figure 2: Modified electric wheel chair

It is extremely important for an unmanned vehicle to be aware of the immediate surroundings, as for the vehicle to collide with obstacles or even get stuck is unwanted, forcing human intervention. For this purpose sensors are mounted on the platform to make the system aware of any dangers in the vehicles vicinity [4].

V Ultrasonic Sensors

There are a large variety of sensors available on the market utilizing different methods of detection for example laser, infrared and ultrasonic sensors. The sensors that use light as a source of detection like, laser for example predominantly use a triangulation method where by the reflected light is returned at an angel to the transmitted light and so through the process of triangulation the distance or proximity of the object is determined. Figure 3, depicts the utilization of ultrasonic sensors on an AGV.

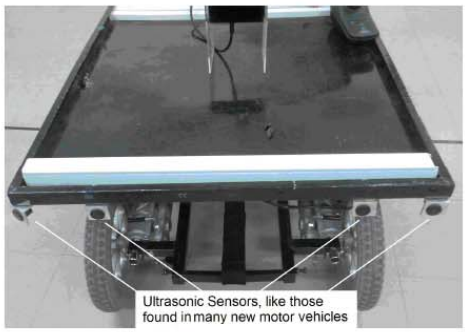


Figure 3: Ultrasonic sensors

Ultrasonic sensors use the time of flight method as seen in Figure 4, this method is bases on the knowledge that sound travels through a medium at a known speed, therefore the time take between the initial transmitted wave and the reflected wave is divided by two and multiplied by the speed of the sound to determine the distance of the obstacle.

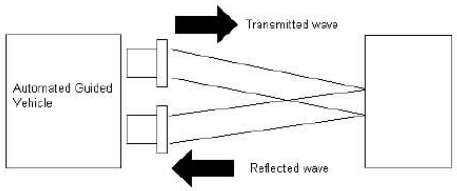


Figure 4: Time of flight

VI Navigation

Navigation is a process of planning a route by which an AGV can be controlled. All navigation techniques involve locating the AGV position and comparing it to known patterns or known surroundings.

Dead reckoning is a navigational method by which the present position is determined by projecting direction and speed traveled from a known position. The dead reckoning position is only an approximation as it does not make provision for any other external influences.

Wire guidance is a method by which a wire is placed approximately one inch below the floor surface with an electronic current at high frequency running through it, this induces a magnetic field around the wire. Next a device called a floor controller passes over it and uses the wires magnetic field to control the steering system so that the AGV does not deviate off the route along which the wire is placed. This system is expensive and the exact path has to be cut into the floor. The cut for a turn has to follow the radius curve of the AGV as otherwise the AGV will deviate from its path and would have to be reset.

Laser navigation is a more accurate method that utilizes small reflective targets that are mounted on the walls, posts and so forth, surrounding machines. A rotating laser light beam is mounted on the AGV that sources the light beams and a receiver that receives any reflected light beams. When the light transmitted is reflected, the distance and angle is measured to determine where the markers are in comparison with the A.G.V. The coordinates of the markers are then compared to the coordinates on the AGV s internal memory as to establish the exact position of the AGV.

Another technique is to place a solid-state gyroscope in the AGV. It detects any small deviations in the AGV s direction of travel. These small deviations are then compared to the mapped out coordinates in the AGV s internal memory and the necessary corrections are applied to the traveled route. There are small markers (magnets) installed in the floor as to correct any small errors that the gyro might have made.

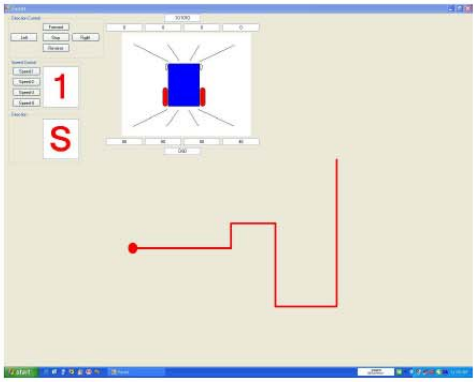


Figure 5: Human interface with control of AGV

As seen in Figure 5, the dead reckoning method is used. Knowing the speed and the time traveled the approximate position can be determined.

VII THEORY OF IMAGE TRANSFORMATION

A dome shaped mirror is to be mounted facing down toward the ground so that when facing the mirror an image depicting a 360̊ view of the horizontal plane would be observed. An AGV with the Omni directional sensor is shown in Figure 6.

Figure 6: AGV with Omni directional image sensor

The camera focused on the dome shaped mirror would render an image resembling the image reflected by a compact disc as seen in Figure 7.

Figure 7: Image rendered by camera

The image must be transformed into an image that is easily interpreted by humans on an interface.

The image s pixels have to be moved using a mathematical transform called a polar transform.

The polar transformation is a well known and widely used mathematical operation. The implementation of this transformation converts polar coordinates using radius and angle to Cartesian coordinates using horizontal and vertical axes. The effect of this operation is to convert circles to horizontal lines and rays radiating from the origin to vertical lines. The polar operation can be expressed mathematically as [5]:

$$[r\cos\emptyset, r\sin\emptyset] \rightarrow [x, y] \quad (1)$$

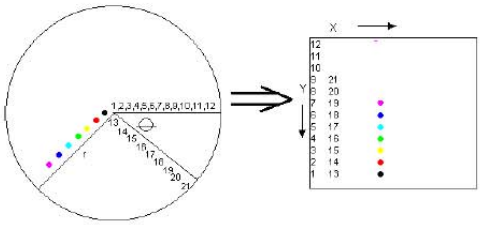


Figure 8: Polar Transform

Figure 8 is a graphical representation of the effect the polar transform has on the image rendered by the camera. The numbers seen in the images represent the position of the pixels in the original and new image after processing. The original image is the circular image.

VIII PROCESSING SPEED OF IMAGE

For every pixel moved a calculation is to be completed using many processing clock cycles. This is unwanted as it

slows down the reaction of the entire system. It makes therefore sense to investigate whether running the process and mapping the new coordinates after every mathematical calculations in a lookup table, will lessen the clock cycles to determine the new positions of the pixels. This will possibly improve the processing cycles drastically.

IX Communication

For the AGV to be completely wireless, a system has to be chosen that is flexible and has to have a high rate of data transfer. Wireless Local Area Network (WLAN) is tested to be used as it has a wide range and can be used through different network routers. WLAN's can be set up as ad-hoc systems, in other words, can have two modules that can only communicate with each other therefore it can replace a wire. In Figure 9 is an example of two modules that have been set up as an ad-hoc network for serial communication.

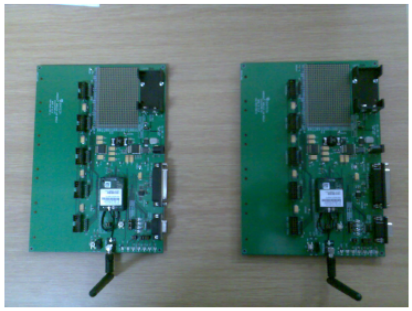


Figure 9: WLAN modules

Conclusion

A high resolution camera focused at a conic mirror will be utilized in conjunction with ultra sonic sensors. These devices will be used for detecting obstacles that do not reflect light, for example glass, to enable an AGV to get detailed information of its surroundings.

This is vital for the decision making process with respect to corrections in its navigation of a predetermined route. The Navigation method implemented is based on the dead reckoning navigation method.

References

[1] Fend M., Bovet S. and Halfner V.V., The Artificial Mouse – A Robot with Whiskers and Vision – Artificial Intelligence Laboratory University of Zurich, 2004.

[2] Fernandes J.C.A and Neves J.A.B.C., Using Conical and Spherical Mirrors with Conventional Cameras for 360° Panorama Views in single image – Department of Industrial Electronics Universidade do Minho, Department of Electrical Engineering Universidade Lusiada, Mechatronics, 2006 IEEE International Conference on, July 2006.

[3] Patent Storm US Patent 4846297 Automated guidance vehicle www.patentstorm.us/patents/4846297/fulltext.html , July 11,1989.

[4] Automated Guided Vehicle http://en.wikipedia.org/wiki/Automated_Guided_Vehicle.

[5] Leo S. Bleicher, Serial Polar Transformations Of Simple Geometries, San Diego, CA 92037, 2004.