University of Essex
Department of Computer Science

Ultrasonic Ranging for Model Helicopters

SURNAME	Naylor
OTHER NAMES	David
QUALIFICATION SOUGHT	MSc Embedded Systems & Robotics
TITLE OF PROJECT	Ultrasonic Ranging for Model Helicopters
SUPERVISOR	Prof Owen Holland
DATE	September 2006

ABSTRACT

Ultrasonic ranging systems that use arrays of conventional time of flight sonar sensors offer an effective, easy to use and low cost means of sensing the distance and direction of objects and surfaces in an environment. Ultrasonic ranging systems are commonly used on many different types of vehicles, however, currently available systems are not suitable for use on small scale model electric helicopters, due to the unique constraints encountered when using this type of aerial vehicle.

An ultrasonic ranging system is therefore developed for use on model helicopters, using an array of SRF10 miniature ultrasonic ranging devices and a novel Real-time Ultrasonic Firing Algorithm for Crosstalk Elimination (RUFACE). The new algorithm is based on the same principles as the existing Error Eliminating Rapid Ultrasonic Firing (EERUF) algorithm but the firing sequences are calculated dynamically in real-time. The RUFACE algorithm is shown to effectively eliminate crosstalk, while the flexibility resulting from the dynamic firing sequences allows for additional benefits such as an adaptive firing rate.

This project dissertation is in accordance with Examinations Regulations 6.12 and 6.13.

Table of Contents

List of Figures

List of Tables

Appendices

Acknowledgments

I would like to express my gratitude to everyone whose help and support have contributed to the success of this project.

My supervisor, Prof Owen Holland, for introducing me to the topic and providing guidance and advice throughout.

Renzo De Nardi for his expert help with the technical side of the project.

Greg Willatt for helping with the construction of the system and the testing platform.

1. Introduction

Ultrasonic ranging systems that use arrays of conventional time of flight sonar sensors offer an effective, easy to use and low cost means of sensing the distance and direction of objects and surfaces in an environment. Ultrasonic ranging systems are commonly used on many different types of vehicles, however, their use on small scale model electric helicopters has been limited to only single sensor systems used for measuring altitude (5)(6). Currently available multi-sensor systems are not suitable for use on small scale model electric helicopters.

The aim of this project is to make multi-sensor ultrasonic ranging available to small scale model electric helicopters using easily available, off-the-shelf components by developing an embedded ultrasonic ranging system that is suitable for use on this type of vehicle.

A model helicopter presents a challenging platform for an ultrasonic system due to the unique constraints encountered when using this type of aerial vehicle. For example, the system has to be small and light weight to minimise negative effects on the helicopter's flight. A helicopter is capable of moving in any direction and therefore ranging must be provided in all directions.

The need for model helicopter ultrasonic ranging stems from the UltraSwarm project (7)(8), currently under development at the University of Essex, that aims to have autonomous flocking model helicopters in an indoor arena. The UltraSwarm helicopters require ranging information for obstacle avoidance and position localisation.

The ultrasonic system is implemented on a microcontroller connected to an array of sonar devices. The SRF10, advertised as the “world's smallest ultrasonic ranger”, was the first choice of sonar devices considered for the sensor array for its small size and weight. A detailed investigation is undertaken to determine whether the SRF10 is suitable for this purpose.

One of the challenges of using multiple ultrasonic devices is the problem of crosstalk. After looking into a number of approaches that address the problem of crosstalk, the Error Eliminating Rapid Ultrasonic Firing (EERUF) algorithm(1) was found to be the most feasible option. The disadvantage of EERUF is that it uses predetermined, fixed sensor firing schedules and this does not allow for much flexibility. EERUF can effectively detect and filter crosstalk but its fixed firing schedules restricts its ability to adapt its firing rate to prevent crosstalk happening in the first place.

A novel algorithm, Real-time Ultrasonic Firing Algorithm for Crosstalk Elimination (RUFACE) has therefore been developed that is based on the same principles as EERUF but calculates the firing sequences dynamically in real-time. The RUFACE algorithm is shown to effectively eliminate crosstalk, while the flexibility resulting from the dynamic firing sequences allows for an adaptive firing rate.

An ultrasonic system that is capable of eliminating crosstalk requires precision firing of the sonar devices. The developed system achieves this using a sophisticated design that coordinates the actions of the sonar devices dynamically allowing for flexible firing sequences and scalability.

2. Background

2.1. The UltraSwarm Project

The need for ultrasonic ranging on model helicopters stems from the UltraSwarm project (7)(8) currently under development at the University of Essex that aims to build an indoor flocking system using small electric helicopters. The UltraSwarm helicopters require onboard multi-directional ranging for obstacle avoidance and position localisation.

Ultrasonic sensors have been selected for their low cost, ease of use, small size and, most importantly, light weight as the maximum weight that can be carried by the model helicopter is a major constraint.

2.2. Ultrasonic Ranging

A time of flight ultrasonic range finder is a sonar sensor used to determine the distance to an object or surface in front of it. This is achieved by emitting a short burst of ultrasonic waves that get reflected off any objects in their path back towards the device where they are detected by a receiver. By measuring the time the ultrasonic waves take to return to the device, the distance to the nearest object can be calculated using the speed of sound through air according to the following simple equation:

$$distance = \frac{speed\ of\ sound \times time}{2}$$

The speed of sound through air is not constant and is effected by factors such as the temperature, humidity, atmospheric pressure and wind. However, in an indoor environment the variation of these factors is small enough to be considered negligible. The speed of sound can therefore be accurately approximated at room temperature from the following equation (2)(10):

$$speed\ of\ sound = \sqrt{nRT}$$

$$= 343.026 \mathrm{m}/s$$

with :

$n = 1.4\ (adiabatic\ constant\ of\ air)$

$R = 287 \mathrm{m}^2/s^2\ K\ (gas\ constant)$

$T = 292.85 \mathrm{K}\ (20^o C\ in\ Kelvin)$

The direction of the detected objects can be determined from the direction the ultrasonic device is pointing but this is only an approximation as the beam of ultrasonic waves emitted expands as the waves propagate, producing a beam angle from the point of emission. Any objects that fall within this angle will reflect the ultrasonic pulse and the object will be detected. The direction of the detected object is only known to be somewhere within the beam angle.

The accuracy of the direction measure can be increased by using an array of multiple ultrasonic devices each pointing an incremental few degrees apart so that their ultrasonic beams are overlapping. In this case the same object is likely to be detected by multiple consecutive ultrasonic devices in the array as can be determined by similar consecutive range readings. The direction of the device in the middle can then be taken to be the most accurate measure of the direction to the object. The problem with adding more ultrasonic devices is that the levels of interference are increased.

2.2.1. Specular and Interference Errors

Although ultrasonic range finders are capable of providing accurate ranging data, specular and interference errors that result in incorrect readings are commonly encountered when using this type of sensor (11).

Specular errors occur if the incident angle of the emitted beam to the perpendicular of the surface or object is greater than the beam angle. This scenario is illustrated by *fig 1*.

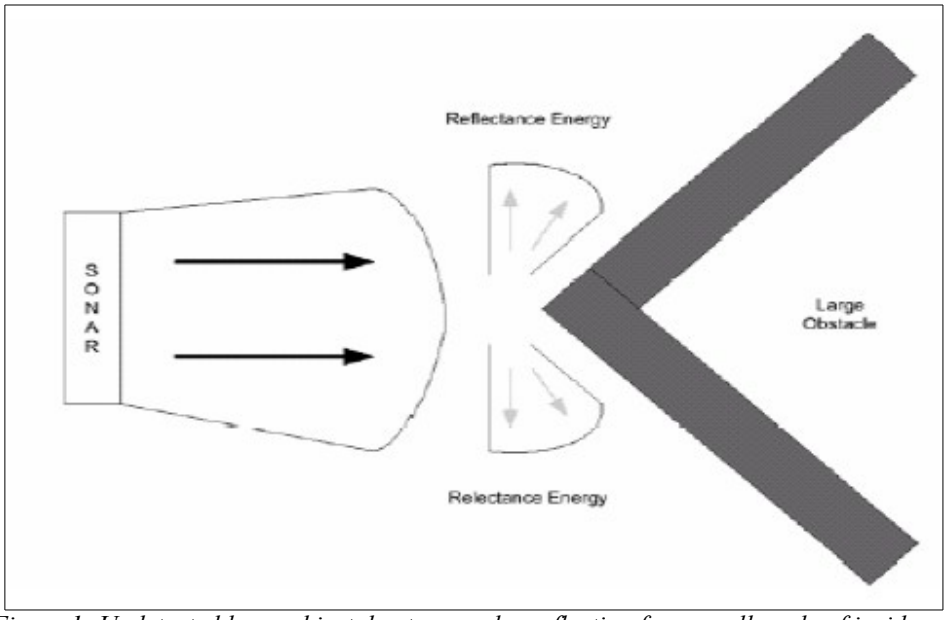

Figure 1: Undetected large object due to specular reflection from small angle of incidence
(Modi, S. Comparison of three obstacle avoidance methods for an autonomous guided vehicle, pp27)

As can be seen the ultrasonic waves are reflected off the surface or object at an angle and will not return to the ultrasonic receiver. The data from the sensor will indicate that there is no object in front of it (2). In another scenario the waves may be reflected off multiple surfaces before returning to the ultrasonic receiver. Due to the indirect route travelled by the waves they will take longer to return to the receiver and the sensor readings will indicate the object to be further away than it really is (11). Specular errors are a consequence of the same property of ultrasonic waves that makes them suitable for ranging, they are highly reflective. There is therefore not much that can be done to prevent specular errors from occurring, however sensors that have wide beam angles are less susceptible to these errors.

The second problem, of errors resulting from interference, is a major issue when using ultrasonics. Interference occurs when multiple ultrasonic devices are used together and one sensor detects the echo produced by another sensor. This type of interference is known as crosstalk (1) and is demonstrated by *fig 2*. Interference can also occur from other sonar sources such as ultrasonic sensors on other robots (1)(3). During its waiting period, before its echo is received, the ultrasonic device is susceptible to this interference and it may interpret the noise as its emitted pulse's echo. If this occurs, the ultrasonic device will provide an incorrect range reading that is less than what is would have been if it had received its correct echo. The higher the rate that the sensors fire, the more the total amount of ultrasonic waves present in the environment and therefore, the more crosstalk occurs.

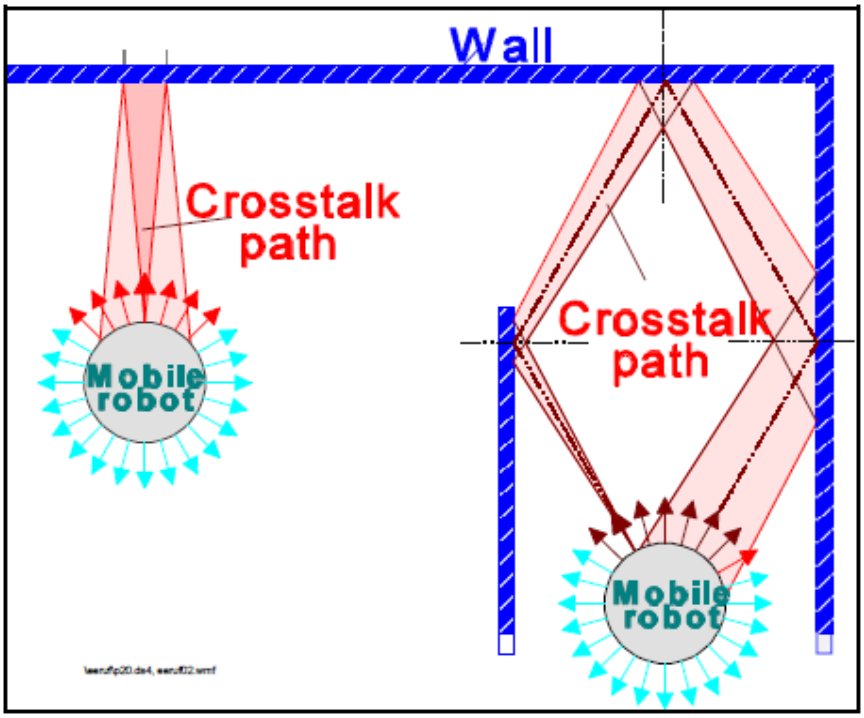

Figure 2: How crosstalk from onboard sensors is generated
(Borenstein, J et al. Error Eliminating Rapid Ultrasonic Firing for Mobile Robot Obstacle Avoidance, pp4)

2.2.2. Error Filtering and Noise Reduction

If a high rate of firing is required it is therefore necessary for the ultrasonic ranging system to reduce the effects of crosstalk. A number of methods have been proposed for this purpose, such as:

"Double Pulse Coding" as proposed by Heale and Kleeman (3)(4) has each sensor fire two pulses in quick, accurate succession. The time between the two pulses is unique for each sensor and only echoes received by the sensor, the unique time apart, are considered.

"Pseudo-Random Coding" is claimed by Jörg and Berg (11) to eliminate crosstalk and noise by having each individual sensor emit bursts of pseudo-random sequences with sharp autocorrelation functions that do not correlate with sequences from other sensors. This allows the individual sensors to identify their echoes using a matched filter technique.

Both the double pulse coding and pseudo-random sequence methods require the use of specialised sonar hardware. As one of the objectives of the project is to use easily available, off-the-shelf components, the model helicopter ultrasonic ranging system uses standard time of flight ultrasonic devices that do not provide the functionality required by these methods. Therefore a solution for

filtering interference based on these methods is not possible on the proposed system.

Borrenstein and Koren's (1) “Error Eliminating Rapid Ultrasonic Firing for Mobile Robot Obstacle Avoidance” (EERUF) is a method for eliminating crosstalk that uses a combination of the Comparison of Consecutive Readings” method and alternating delays.

With the comparison of consecutive readings method, two consecutive readings from the same sensor are compared and if the difference is greater than a certain threshold the readings are filtered. Borrenstein and Koren have shown this method to be effective at eliminating external ultrasonic noise, caused for example by other robots, as the probability of receiving two consecutive erroneous readings that are similar is small. This method is, however, not effective when it comes to noise generated from crosstalk as this causes consecutive errors that are the same (1, pp6). For a detailed explanation of why crosstalk results in errors that are the same, see section 5.1.1.

Introducing an alternating delay before each ranging operation is initiated produces consecutive errors resulting from crosstalk that are different. This allows for errors resulting from crosstalk to be detected and discarded by the comparison of consecutive readings method.

Borrenstein and Koren claim that this method is effective at significantly reducing crosstalk errors even when there are multiple ultrasonic systems operating in the same environment (1, pp1). For this reason and that this method can be used with standard time of flight ultrasonic devices, EERUF was initially selected for use on the helicopter ultrasonic ranging system to be developed. However, it soon became clear that EERUF has a number of limitations.

EERUF requires a number of timing parameters to be configured such as the alternating delay times, the error threshold for which readings will be accepted and the time interval between different sensors firing. Finding acceptable timing parameters as stated by the EERUF authors is “not quite trivial” and is dependent on a number of factors such as the number, configuration and properties of the sensors used. Once the timing parameters have been selected they are used to construct a firing schedule. The consequence of having a predetermined firing schedule is that the system lacks flexibility. If different timing parameters are required, for example if the minimum required delay between sensors firing is changed, or if the number of sensors used is changed, the firing schedule must be recalculated. EERUF can effectively detect and filter crosstalk but its fixed firing schedules restricts its ability to adapt its firing rate to prevent crosstalk happening in the first place. Borrenstein and Koren have proposed an “adaptive scheduling” (1, p11) approach where

multiple fixed firing schedules are created with different firing rates and an appropriate firing schedule is then selected according to the rejection rate. However, this approach has its limitations. A fixed firing schedule can only approximate the optimal firing rate for any given situation and certain situations may arise where none of the fixed firing schedules are appropriate.

A novel "Real-time Ultrasonic Firing Algorithm for Crosstalk Elimination" method (RUFACE) has therefore been developed for use on the ultrasonic ranging system. RUFACE is based on the same principles as EERUF and uses the comparison of consecutive readings method, however instead of using preconfigured alternating delays, the firing sequences are calculated dynamically in real-time.

The RUFACE algorithm is shown to effectively detect and filter crosstalk and allows for non-scheduled, asynchronous and non-sequential firing of the sensors. This flexibility allows for an adaptive solution where the amount of crosstalk that is occurring is monitored and the timing parameters adjusted accordingly in real-time so that firing rate is always at its optimum. The details of the RUFACE algorithm are presented in chapter 5.

2.2.3. Comparison of Consecutive Readings Rejection Threshold

Due to the random nature of the environmental noise, the probability of two incorrect readings, that have been subjected to interference, being the same is very low. On the other hand two correct consecutive readings from an ultrasonic device will always be very similar if not the same. This is the case, at least, if the ultrasonic device and the environment are stationary as the echoes will be reflected off the same surface. If, however, there is movement, either by the ultrasonic device, or the environment, the distance between the ultrasonic device and the reflective surface will differ from one reading to the next relative to the speed of the movement. The faster the movement, the greater the difference will be between two correct consecutive readings for the same firing rate but this difference will decrease as the firing rate increases as there is less time to move between firings.

When there is movement, the comparison of consecutive readings method is still effective as long as the ratio between the rate of firing consecutive pairs and the speed of the movement results in pair differences that are within the rejection threshold value. The selected rejection threshold value should therefore be high enough to satisfy this constraint. For example if an ultrasonic device is attached to a model helicopter that has a maximum speed of 1m/s and the period of time between consecutive firings is 10ms, then the helicopter can move a maximum distance of 1cm between

readings. The rejection threshold should therefore be at least 1cm so that consecutive correct readings when the helicopter is moving at 1m/s do not get filtered.

Unfortunately the larger rejection threshold value, the more erroneous readings get through, and the less effective the comparison of consecutive readings method is at filtering out noise. It is therefore important that the firing rate be as high as possible so that a low rejection threshold can be used.

3. System Requirements

The purpose of this chapter is to detail the specific requirements of the ultrasonic ranging system and its sensors.

In order to determine these requirements it is necessary to first consider the specific domain in which the system will function. This includes:

- **The helicopter platform**. The helicopters used by the UltraSwarm project are customised versions of the Hirobo XRB (*fig 3.*). The Hirobo is a co-axial rotor, small, electric, model helicopter. The important properties of the Hirobo that have been taken into consideration are given in *table 1.*

- **The environment**. The environment in which the helicopter will operate is an indoor arena. The arena is cylindrical in shape with a diameter of 12 metres and height of 6 metres. The arena has plastered walls, a metal floor, one large glass window and a number of smaller glass windows.

- **The intended purpose**. The helicopters are to be used for the UltraSwarm project which intends to have multiple helicopters flocking. The ultrasonic ranging system is to be used to provide each helicopter with ranging data for obstacle avoidance and position localisation. There will be multiple helicopters in operation at the same time and they will be in close proximity to each other.

Figure 3: The Hirobo XRB model helicopter

Rotor Diameter	350 millimeters
Height	140 millimeters
Weight	170 grams
Payload Capacity	40 grams

Table 1: Hirobo XRB properties

3.1. Ranging System Requirements

With the above mentioned specific domain considerations in mind the following requirements have been determined for the ultrasonic ranging system:

- The total weight of the system must not exceed 40 grams. A major constraint is that the system must be light weight and small to minimise negative effects on the flight of the helicopter from gravity and drag. It is therefore necessary to use small and light component and to use as few components as possible while still satisfying all other requirements.

- The system must have low power drain so that the duration the helicopter can operate between charges is maximised using as small and light a battery as possible.

- A helicopter is an aerial vehicle that can move freely in all directions. Therefore objects must be detected in all directions. Ideally this includes in an upwards direction but as the helicopter's rotors are in the way this is not required.

- The ranging system must be able to function with multiple systems in operation at the same time and within close proximity to the other ranging systems.

3.2. Sensor Requirements

With the above mentioned specific domain considerations in mind the following requirements have been determined for the ultrasonic sensors:

- Acoustic noise, mechanical vibration and air turbulence from the helicopter's rotors increase the amount of interference experienced by the ultrasonic receivers (5). The system must therefore be able to effectively filter the noise produced by this interference from the ranging data. The ultrasonic devices used must able to operate under these conditions and produce ranging data that is of a high enough quality that after filtering has occurred the rate of ranging fulfils the previous requirement.

- The minimum range at which range readings can be made must be at most 150mm. The radius of the helicopter's rotor is 175mm and therefore the helicopter body on which the ultrasonic devices are mounted cannot get closer than this distance to a surface or object without crashing. It is therefore not necessary for the ultrasonic device to be able to read ranges less than this distance.

- The maximum range at which range readings can be made must be at least 6000mm. The indoor arena has a diameter of 12000mm. Therefore the maximum distance the helicopter can be from all sides of the arena is 6000mm if it is in the centre. Assuming the ultrasonic devices point in all directions, as required above, having the maximum range above this distance ensures that at least two ultrasonic devices will always be able to provide range readings. This is necessary for position localisation.

- The maximum firing rate of each ultrasonic device must be 20Hz or

more. The helicopter has a maximum speed of 1m/s, therefore a firing rate of 20Hz ensures the maximum distance the helicopter can move between readings is 50mm. This is sufficiently less than the helicopter's rotor radius, of 175mm, for obstacle avoidance to be performed.

- For the same reasons as the above requirement, the ultrasonic device must produce ranging data that is accurate to 50mm for any given distance.

- The average variation between consecutive range readings when the distance is constant must be 10mm or less. The method used for filtering crosstalk and other interference is the comparison of consecutive readings method. Therefore it is important that consecutive readings are consistent.

- The ultrasonic sensors must be able to provide accurate and consistent range readings, as specified above, off all surfaces encountered in the indoor arena, i.e. the plastered walls, the metal floor and the glass windows.

4. The SRF10 Ultrasonic Range Finder

The SRF10 is the time of flight ultrasonic ranging device that has been selected for the sonar array of the ultrasonic ranging system.

Voltage	5V
Current	15mA Typical, 3mA Standby
Frequency	40KHz
Range	30mm - 6000mm
Connection	Standard I^2C Bus
Units Range reported in	uS, cm or inches
Size	32mm w x 15mm d x 10mm h
Weight	5 grams
Beam Angle	72^o

Table 2: SRF10 Manufacturer's Specification

The manufacturer's specifications, shown in *table 2.* above, were evaluated and the SRF10 was selected for the following reasons:

- It is light weight (5g) and the total weight of the system is a major constraint if it is to be used on model helicopters.
- It is small in size (32mm w x 15mm d x 10mm h) and this is important to minimise the effects of drag on the helicopter during flight.
- Its current drain is low (15 mA Typical, 3 mA Standby). The less the power drain on the helicopter's battery the better as the helicopter will be able the fly for longer without requiring a larger, heavier battery.
- It provides a standard I^2C interface which allows multiple devices to be connected and addressed on a single bus.
- It has a wide beam angle (72^o), which means that fewer devices need to be used to provide ranging data over a given area. Having fewer devices is desirable as this reduces the total weight of the system.

Figure 4: SRF10 Ultrasonic Range Finder

An additional feature of the SRF10 is that it has software adjustable maximum analogue gain and maximum range settings which means it can be calibrated for the environment in which it is used. The details of these settings are discussed in the next section.

4.1. SRF10 Maximum Analogue Gain and Range Setting

The SRF10 has adjustable maximum analogue gain and maximum range settings which effect the properties of the device. These settings are adjusted by sending commands to the SRF10 device over the I^2C bus.

The analogue gain value of the SRF10 is the inverse of the amplitude threshold at which the receiver registers that an echo has been received. Straight after a pulse has been fired the ultrasonic receiver is at its lowest analogue gain value and the amplitude threshold is therefore high. This means that only very strong echoes will register. This value is then increased steadily over time until the maximum analogue gain setting is reached. Therefore as the time passes after the pulse is fired, weaker echoes will register. This technique is commonly used by ultrasonic sensors and is based on the reasoning that the amplitude of echo decreases over time as it propagates through the air. Therefore the longer an echo takes to return to the receiver the weaker the echo will be when it is received and the amplitude threshold of the receiver must be reduced for the echo to be registered. Having a variable analogue gain value reduces the effects of interference as only echoes above the amplitude threshold are registered.

The maximum analogue gain setting limits the maximum level that the analogue gain of ultrasonic receiver can increase to. Higher maximum gain setting allow the SRF10 to register weaker echoes and therefore increases the maximum range but the device will be more sensitive to noise and interference. There are 16 analogue gain setting and *table 3.* gives the corresponding maximum gain

value for each setting.

Gain Register Setting	*Max Analogue Gain Value*
0	40
1	40
2	50
3	60
4	70
5	80
6	100
7	120
8	140
9	200
10	250
11	300
12	350
13	400
14	500
15	600
16	700

Table 3: SRF10 gain settings and corresponding max gain value

The maximum range setting controls the amount of time the SRF10 will listen for an echo to be received after transmission. The further away an object or surface is to the device the more time the echo will take to return to the receiver. Increasing the maximum range setting therefore increases the maximum range of the SRF10. The disadvantage of a high maximum range setting is that the SRF10 waits for longer before firing again if an echo is not received.

The SRF10 maximum range setting is an 8 bit register that when set to its maximum decimal value of 255 will listen for echoes for 65mS. Therefore each incremental setting of the register from 0 adds an additional 65mS / 256 = 0.254mS to listen for. Using a speed of sound constant of 343.026m/s, the distance travelled by the echo in 0.254mS is 87.13mm. This is divided by 2 to get the maximum range, as the echo makes a round trip, to give a distance of 43.56mm. Therefore each range setting increment adds 43.56mm to the maximum range.

4.2. SRF10 Testing

In addition to the manufacturer's specifications, the SRF10 has been extensively tested in the indoor arena to to ensure that it satisfies the sensor requirements specified in chapter 3. For this purpose a customised testing platform was constructed, the details of which are discussed in chapter 6. A large number of sensor readings were recorder (*appendix 2*) with single and multiple SRF10s at various:

- distances to the arena surfaces
- angles to the arena surfaces
- firing rates
- SRF10 maximum analogue gain settings
- SRF10 maximum range settings

The results of the testing showed that the SRF10 performs well and easily meets all the requirements. However, the SRF10 proved to be susceptible to the common problem of crosstalk when multiple devices were used with a high firing rate.

This is demonstrated by the following two graphs (*fig 5.* and *fig 6.*) that show a comparison between the results obtained at 1 metre from the same SRF10 with only a single SRF10 firing and then with multiple SRF10s firing. The data displayed in the graphs is the mean of each of 100 range readings taken with increasing delays between the SRF10s firing from 0ms to 10ms.

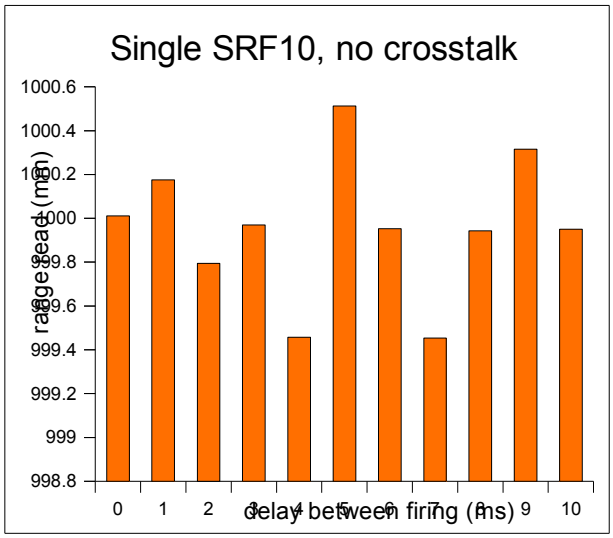

Figure 5: Single SRF10 firing with no crosstalk

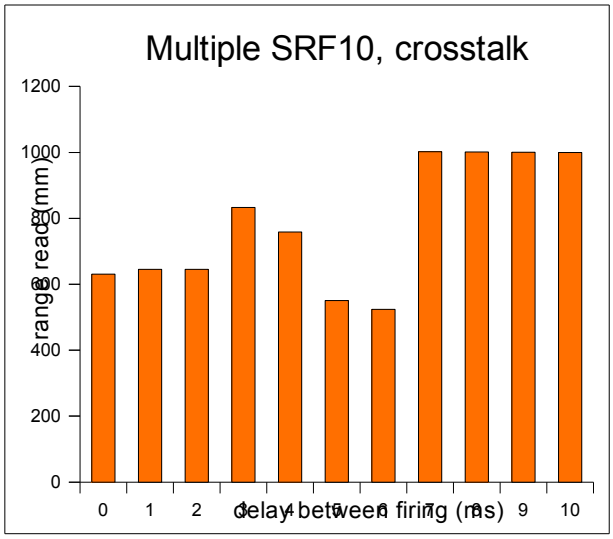

Figure 6: Crosstalk with multiple SRF10s firing

As can be seen when there is only one single SRF10 firing the range readings are consistently correct but when multiple devices are firing the range readings are incorrect for delays below 7ms as a result of crosstalk.

4.3. Beam Angle

The ultrasonic pulses transmitted by the SRF10 travel in the direction that the SRF10 is pointing. As the pulses are sound waves, they will also be subject to a certain degree of expansion or spreading out as they propagate through the air. Therefore at any given distance from the source the ultrasonic waves will have a certain width with the energy of the waves being the highest at the centre and decreasing to zero in an outward direction. Objects that are within the width of the ultrasonic waves will reflect the waves but only objects that are closer to the centre of the waves will produce echoes that are strong enough to be detected by the SRF10's receiver. The width in which objects are detected is known as the beam width. The beam width can be seen to form an angle from the SRF10 and all objects that fall within this angle are detected. This is known as the beam angle. Due to the small surface area of the SRF10 ultrasonic transmitters the beam width of the emitted pulses produce a large beam angle as is shown in *fig 7*.

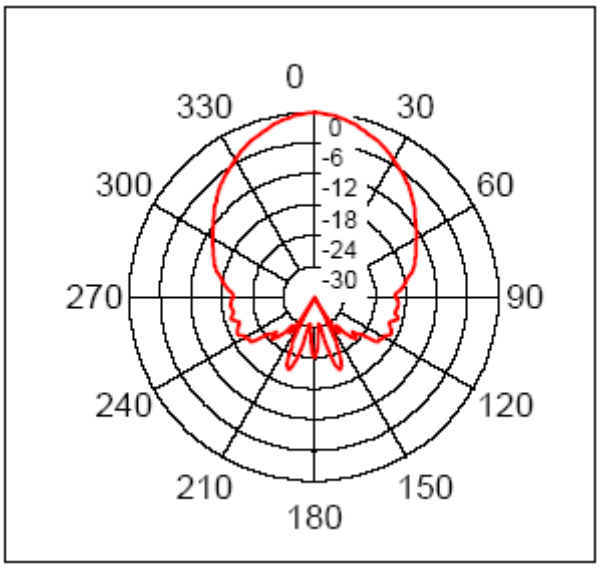

Figure 7: The beam pattern in dB of the SRF10, taken from the manufacturer's data sheet

Any objects within the arc of the beam angle will be detected by the SRF10. This creates a viewing angle of approximately 72º. A single SRF10 is therefore well suited to sensing for objects in a wide area but not at accurately determining the direction of the detected objects as the direction of a detected object is only known to be between -36º and 36º from the direction the sensor is pointing.

4.4. SRF10 Sensor Array Configuration

The requirements from chapter 3 specify that the ranging system must be able to detect objects in all directions, except in an upwards direction. It is therefore necessary to use multiple SRF10s pointing in different directions to form an array of sensors that sense in all directions. Taking advantage of the wide beam angle of the SRF10 it is possible to do this with only five devices that are placed at 90º apart facing forwards, backwards, left, right and downwards.

As using as few devices as possible is beneficial for minimising the overall weight of the system, this is the sensor array configuration that has been selected for the ultrasonic ranging system. The drawback of this configuration is that the direction of the detected objects can only be coarsely estimated, however, for the initial stages of development, this is not considered an issue as more devices can be added at a later stage if this becomes necessary.

4.5. SRF10 Conclusion

The SRF10 ultrasonic ranging device has been evaluated and found to be suitable for use on the

helicopter ranging system. A sensor array configuration of SRF10s has been selected with five devices positioned at 90º to each other.

However, the SRF10 was found to be susceptible to crosstalk when multiple devices are fired at a high rate. It is therefore necessary for the ultrasonic ranging to employ a method to reduce the effects of this crosstalk and this is the topic of the next chapter.

5. RUFACE a Novel Ultrasonic Firing Algorithm

This chapter details the RUFACE ultrasonic firing algorithm that has been developed. The theory behind the algorithm is discussed and it is shown how this can be used to calculate sensor firing times on the fly that allow for crosstalk interference to be detected.

The main features of the algorithm are as follows:

- Detectable crosstalk errors. The sensors are fired at precise times that ensure any errors resulting from crosstalk can be detected and therefore filtered.

- Non-scheduled firing. The are no predetermined, fixed firing schedules. All firing times are determined dynamically in real-time. The flexibility offered by this approach is shown to be beneficial for a number of reasons.

- Asynchronous firing. The operations of firing the sensors and waiting for their echoes to be received are not synchronised. After a sensor fires, the next sensor is scheduled to fire as soon as possible, regardless of when the first sensor receives its echo.

- Non-sequential firing. The sensors do not fire in a fixed order. As soon as an ultrasonic device receives it's echo, it is queued to fire again. Therefore sensors that are closer to objects will fire more rapidly than those further away. This provides more ranging data where it is needed, as the ranging information for closer objects is more important for obstacle avoidance.

- Adaptable firing rate. Having non-scheduled firing means the firing rate can easily be changed at any time to any value. The rate that readings are being filtered is a measure of how much crosstalk is occurring. A higher filtering rate indicates that the firing rate must be reduced and a low filtering rate indicates the firing rate can be increased.

5.1.1. Why Crosstalk is not Detectable

Consider a simple ultrasonic ranging system with two pulse-echo sonar devices, A and B, The sonars fire sequentially, one after the other, after a constant delay of T_{delay} has passed from the time the firer's echo has been received. The sequence of events over a period of time for such a system is a follows:

1. A_{fire1} $(Sonar\ A\ fires\ pulse\ 1)$
2. A_{echo1} $(Sonar\ A\ receives\ echo\ 1)$
3. T_{delay} $(Constant\ delay\ applied)$
4. B_{fire1} $(Sonar\ B\ fires\ pulse\ 1)$
5. B_{echo1} $(Sonar\ B\ receives\ echo\ 1)$
6. T_{delay} $(Constant\ delay\ applied)$
7. A_{fire2} $(Sonar\ A\ fires\ pulse\ 2)$
8. A_{echo2} $(Sonar\ A\ receives\ echo\ 2)$
9. T_{delay} $(Constant\ delay\ applied)$
10. B_{fire2} $(Sonar\ B\ fires\ pulse\ 2)$
11. B_{echo2} $(Sonar\ B\ receives\ echo\ 2)$

If it is the case that sensor B is receiving crosstalk from sensor A and the sensors are stationary, the situation that arises as illustrated in the timing diagram below (*fig 8.*):

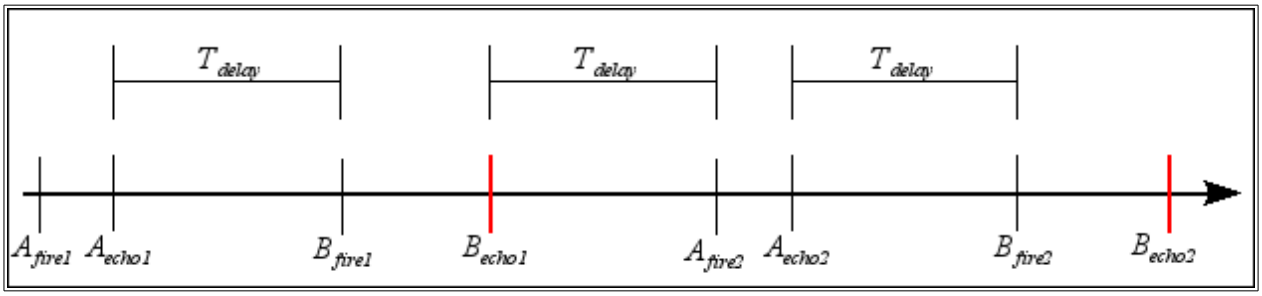

Figure 8: Two ultrasonic devices firing with constant delay, with B receiving crosstalk from A

The following observations can now be made:

1. The times $A_{echo1} - A_{fire1}$ and $A_{echo2} - A_{fire2}$ are equal because the sensors are stationary and the ultrasonic pulse will always have to travel the same distance to and from A, and therefore take the same time.

2. As a result of 1. and T_{delay} being constant, B always fires at the same time after A fires.

3. As a result of 2. and if B is receiving crosstalk from A, after B fires it receives A's echo after the same amount of time has passed. This is because the sensors are stationary and the pulse emitted from A will travel along a constant path to B, and therefore take the same time.

4. As a result of 3. the times $B_{echo1} - B_{fire1}$ and $B_{echo2} - B_{fire2}$ are equal

Having $B_{echo1} - B_{fire1}$ equal to $B_{echo2} - B_{fire2}$ is a problem because this results in B continuously producing erroneous readings that are equal and therefore not detectable by the comparison of consecutive readings filter.

To simplify the concept, the example given above is for a stationary system. If the sensors are moving, the times for $A_{echo1} - A_{fire1}$ and $A_{echo2} - A_{fire2}$ will differ and therefore the times $B_{echo1} - B_{fire1}$ and $B_{echo2} - B_{fire2}$ will differ by the same amount. The size of this difference is relative to the speed of the movement and the size of T_{delay} . If the size of this difference is less than the comparison of consecutive readings rejection threshold, the same problem exists and the erroneous readings produced by B will not be detected.

5.1.2. Making Crosstalk Detectable

Now consider a second example, illustrated by the following timing diagram (*fig 9.*), that is the same as the example in the previous section except that the delay before B_{fire2} has been shortened by the time T_{Δ} .

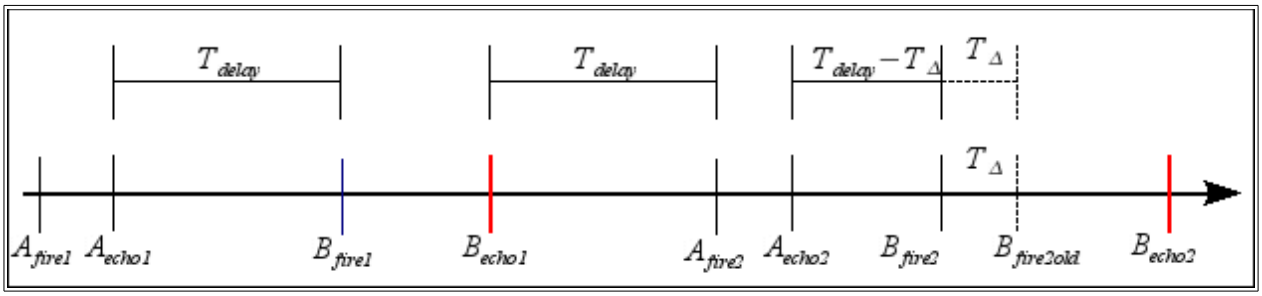

Figure 9: Two ultrasonic devices firing with variable delay, with B receiving crosstalk from A

As can be seen the effects of shortening T_{delay} by T_{Δ} are as follows:

1. The time between A firing its first pulse and B firing its first pulse is $A_{echo1} + T_{delay}$ and the time between A firing its second pulse and B firing its second pulse is $A_{echo2} + T_{delay} - T_{\Delta}$,

and these now differ by T_Δ

2. As a result of 1. B now fires its second pulse at a different time, i.e. $B_{fire2} = B_{fire2old} - T_\Delta$ but because B is receiving crosstalk from A, the B_{echo2} time remains the same

3. As a result of 1. and 2. it is now the case that $B_{echo1} - B_{fire1} = B_{echo2} - B_{fire2} + T_\Delta$ and the times for $B_{echo1} - B_{fire1}$ and $B_{echo2} - B_{fire2}$ differ by T_Δ

The erroneous readings produced by B now differ by T_Δ and these are therefore detectable by the comparison of consecutive readings method as long as T_Δ is greater than or equal to the comparison of consecutive readings' rejection threshold. It should be noted that lengthening the original constant delay before B_{fire2} by T_Δ would have the same effect and the times for $B_{echo1} - B_{fire1}$ and $B_{echo2} - B_{fire2}$ would also differ by T_Δ .

In other words, in order to make B's erroneous readings resulting from crosstalk detectable we must ensure that the times for $B_{echo1} - B_{fire1}$ and $B_{echo2} - B_{fire2}$ differ by at least T_Δ . As shown by the example above this can be achieved by changing the delay time before the first time B fires and the second time B fires by T_Δ . This can be stated by the following equation:

$$\left|(B_{fire1} - A_{echo1}) - (B_{fire2} - A_{echo2})\right| \geq T_\Delta \qquad (1)$$

It is more convenient to state this equation in terms of the sensors' firing times only. It is possible to do this as a result of the time between A firing and receiving its echo always being the same. This is given by

$$A_{echo1} - A_{fire1} = A_{echo2} - A_{fire2}$$

and therefore

$$(B_{fire1} - A_{echo1}) - (B_{fire2} - A_{echo2}) = (B_{fire1} - A_{fire1}) - (B_{fire2} - A_{fire2})$$

which can be substituted into equation (1) to give

$$\left|(B_{fire1} - A_{fire1}) - (B_{fire2} - A_{fire2})\right| \geq T_\Delta \qquad (2)$$

This equivalence is illustrated by the following timing diagram (*fig 10.*)

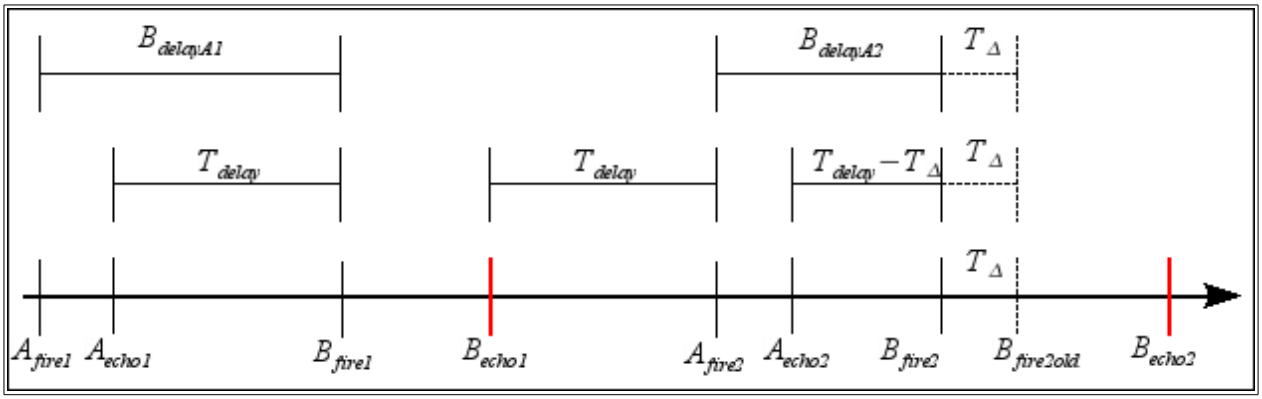

Figure 10: Firing delay equivalences

The EERUF algorithm ensures that equation (2) is satisfied by applying predetermined alternating delays. However, as has been discussed in chapter 2, this approach has limited flexibility. A new approach has therefore been developed that satisfies equation (2) by calculating a suitable value for B_{fire2}, dynamically, in real-time, directly after A_{fire2} has occurred. The details of this approach are presented in the next section.

5.2. A New Approach: Calculating a Suitable Next Firing Time, on the Fly

Assuming that A_{fire2} has just occurred, we would like to calculate the minimal, suitable time to fire B_{fire2} such that equation (2) from the previous section is satisfied. Equation (2) can be rewritten by removing the absolute value brackets as follows

$$\begin{aligned} &(B_{fire1}-A_{fire1})-(B_{fire2}-A_{fire2}) \geq T_\Delta \\ &\qquad OR \\ &(B_{fire2}-A_{fire2})-(B_{fire1}-A_{fire1}) \geq T_\Delta \end{aligned} \qquad (3a)$$

therefore

$$\begin{aligned} &B_{fire2} \leq B_{fire1}-A_{fire1}+A_{fire2}-T_\Delta \\ &\qquad OR \\ &B_{fire2} \geq B_{fire1}-A_{fire1}+A_{fire2}+T_\Delta \end{aligned} \qquad (3b)$$

Equation (3b) provides two ranges of values in which B_{fire2} may fall, but we must also take into account time that has already passed by including another constraint. If the current time, directly

after A_{echo2} has occurred, is given by, $T_{current}$, then

$$B_{fire2} \geq T_{current} \tag{4a}$$

It is also necessary to allow for the minimum delay between firing to be specified so that the firing rate can be controlled. Therefore if T_{min_delay} is the desired minimum delay constant, this can be applied to equation (3a) to give

$$B_{fire2} \geq T_{current} + T_{min\_delay} \tag{4b}$$

The combination of equation (3b) and (4b) provides the range of possible values that can assigned to B_{fire2} that will satisfy equation (2) from the previous section and therefore allow crosstalk errors to be detected. This range is illustrated by the diagram below (*fig 11.*).

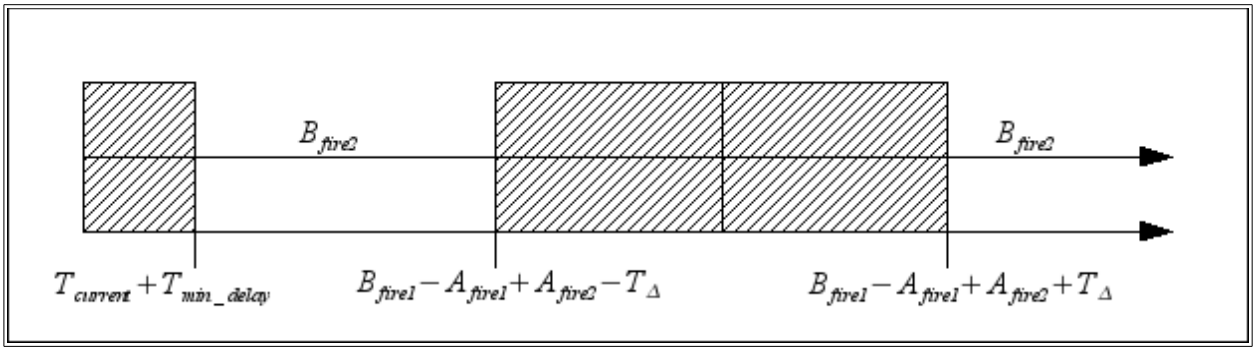

Figure 11: Possible times for B's next firing shown by clear space

The time chosen for B_{fire2} from the range of possible values should always be the soonest possible, to ensure that the delay between firing the sensors is kept to the minimum. With this in mind the value for B_{fire2} can be determined by applying the following formula

let $minAllowed = B_{fire1} - A_{fire1} + A_{fire2} - T_{\Delta}$
and $maxAllowed = B_{fire1} - A_{fire1} + A_{fire2} + T_{\Delta}$

$$\begin{aligned} &IF\ minAllowed < T_{current} + T_{min\_delay} < maxAllowed \\ &THEN \\ &\quad B_{fire2} = maxAllowed \\ &ELSE \\ &\quad B_{fire2} = T_{current} + T_{min\_delay} \end{aligned} \tag{4}$$

The formula presented above provides a way to calculate the next firing time for sensor B that

ensures that if crosstalk occurs from sensor A, it can be detected, but this is only for the specific example given. A more generalised solution is presented in the next section.

5.2.1. An Algorithm for Calculating Non-scheduled Firing Times

In order to extend the formula, presented in the previous section, for calculating the next firing time, into a generalised form that can be used to calculate the next firing time of any sensor in a system with any number of sensors, it must be taken into consideration that every sensor is susceptible to crosstalk from every other sensor. In addition a sensor is also susceptible to crosstalk from itself by receiving old echoes from its own previous firings and these errors should also be detectable.

A generalised form that takes these additional factors into consideration will now be presented by defining $S_i last$ and $S_i previous$ to be the last time fired and previous time fired, before the last time, respectively, for every sensor S_i from the set of all sensors. The next firing time to be calculated for sensor k is defined as $S_k next$. It is still the case that the sensors fire sequentially, one after the echo of the other sensor has been received. The following timing diagram (*fig 12.*) illustrates these definitions.

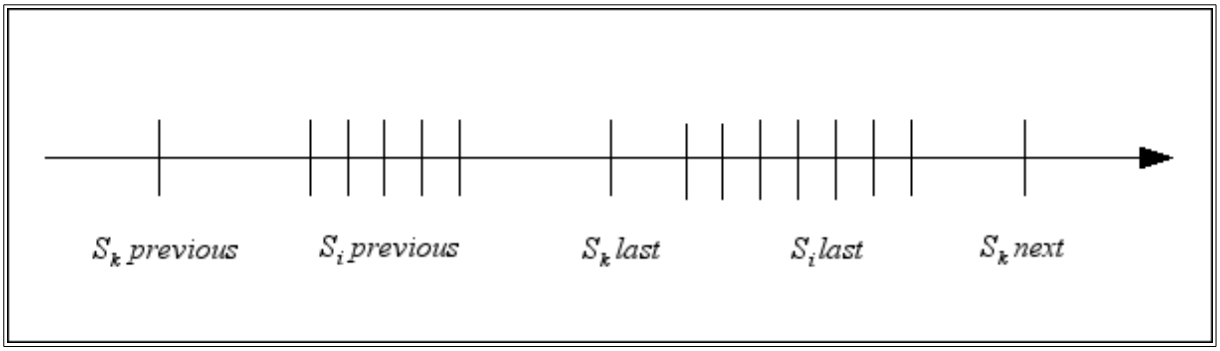

Figure 12: Defining previous, last and next terms

Recall from section 6.2.2. that to ensure crosstalk from sensor A can be detected the following equation must be satisfied

$$\left|(B_{fire1} - A_{fire1}) - (B_{fire2} - A_{fire2})\right| \geq T_{\Delta} \qquad (2)$$

This equation can now be extended to the general form to ensure that crosstalk can be detected from any sensor S_i in the set of all sensors used, using the previous, last and next definitions as follows

$$\forall\ S_i \in all\ sensors:\ \left|(S_k next - S_i last) - (S_k last - S_i previous)\right| \geq T_\Delta \qquad (5)$$

We must now find the minimum value for $S_k next$ that satisfies equation (5) for all S_i in the set of all sensors used. If we use a similar approach as was used in the previous section using formula (4) to obtain B_{fire2} , we can define the following generalised formula:

1. *foreach* $S_i \in all\ sensors$

   let $minAllowed_i = S_k last - S_i previous + S_i last - T_\Delta$
   and $maxAllowed_i = S_k last - S_i previous + S_i last + T_\Delta$

1.1. *IF* $minAllowed_i < T_{current} + T_{min\_delay} < maxAllowed_i$ (6)
THEN
$S_k next_i = maxAllowed_i$
ELSE
$S_k next_i = T_{current} + T_{min\_delay}$

The result of this formula is i $S_k next_i$ values. The $S_k next_i$ value to select for $S_k next$ is the minimum possible that satisfies equation (5) for all S_i in the set of all sensors used. However there is a problem with this approach. If for a certain sensor, $minAllowed_i \geq T_{current} + T_{min\_delay}$ evaluates to true, $S_k next_i$ will be set to $T_{current} + T_{min\_delay}$ and the $maxAllowed_i$ value will be discarded. Now if $S_k next_i$ is not selected as the value for $S_k next$ as a result of it not satisfying equation (5) for another sensor, it becomes necessary to also consider $maxAllowed_i$ as the possible best value for $S_k next$, i.e. the minimum possible that satisfies equation (5), but the $maxAllowed_i$ has already been discarded. For this reason, formula (6) does not provide a satisfactory generalised method for determining the next firing time. For a generalised solution that takes into account these problems, the following algorithm is defined:

1. $S_k next = T_{current} + T_{min\_delay}$
2. $foreach\ S_i \in all\ sensors$
   - 2.1. $minAllowed_i = S_k last - S_i previous + S_i last - T_\Delta$
   - 2.2. add $minAllowed_i$ to $minAllowed\ list$
3. $foreach\ minAllowed_i \in minAllowed\ list$
   - 3.1. $if\ S_k next > minAllowed_i$
     - 3.1.1. remove $minAllowed_i$ from the $minAllowed\ list$
     - 3.1.2. $maxAllowed_i = S_k last - S_i previous + S_i last + T_\Delta$
     - 3.1.3. $if\ maxAllowed_i > S_k next$
       - 3.1.3.1. $S_k next = maxAllowed_i$
       - 3.1.3.2. GOTO 3.
4. $S_k previous = S_k last$
5. $S_k last = S_k next$

Figure 13: An algorithm for calculating the next firing time

The algorithm begins by setting $S_k next$ to the minimum time that it can possibly be, i.e. $T_{current} + T_{min\_delay}$. Now an iteration is started where the $minAllowed_i$ times are calculated and evaluated against $S_k next$ for every sensor. If the current value of $S_k next$ is greater than any of the $minAllowed_i$ value then $maxAllowed_i$ is calculated. At this point $minAllowed_i$ is no longer a possible value for $S_k next$ and it can be discarder. If $maxAllowed_i$ is greater $S_k next$, $S_k next$ is set to $maxAllowed_i$. Now because the value for $S_k next$ has changed, it is necessary to reevaluate any remaining $minAllowed_i$ times, that have not been discarded, and if the new $S_k next$ value is greater than any of them, the iteration is repeated. Finally the algorithm ends by updating the $S_k previous$ and $S_k last$ firing times so that these values are correct for future algorithm evaluations. When the algorithm terminates, $S_k next$ will be set to the minimum possible value that satisfies equation (5). The algorithm is efficient because at worst, the iteration will only have to be repeated i times and $minAllowed_i$, and $maxAllowed_i$ are only calculated once for each sensor.

5.2.2. Asynchronous, Non-sequential Firing

One of the biggest advantages of having dynamic timing is that it offers a great deal of flexibility. We are no longer constrained by predetermined, fixed firing sequences and are therefore free to take advantage of the benefits of non-sequential firing. The following section defines the meaning of the terms asynchronous and non-sequential and discusses their benefits. It is then shows how the algorithm presented in the previous section can be extended to allow for asynchronous, non-sequential firing.

In a conventional, simple ultrasonic firing system the sensors fire in a fixed order one after another. After a sensor fires the system waits for its echo to be received and then after a certain delay the next sensor in the fixed order is fired. In a more advanced system, such as EERUF, this process can occur asynchronously, i.e. after a sensor fires, the system does not wait for its echo to be received and the next sensor in the fixed order is immediately scheduled to fire after a certain delay. A separate parallel process then monitors for the received echoes. This approach is superior to a synchronous one as it allows the sensors to be fired more rapidly as their pulse-echo operations are pipelined.

Another advantage of the asynchronous approach is that it allows for non-sequential firing. Non-sequential firing means the sensors do not fire in a fixed order. Non-sequential firing is desirable because it allows the sensors that are reading smaller ranges to fire more rapidly than those that are reading ranges further away. This is beneficial because the smaller distances read are often of more importance than larger distances for example when the ranging data is being used for obstacle avoidance.

In an asynchronous system, if it is the sensor's turn to fire again but its echo has not yet been received, the system can postpone the firing of the sensor to the next round and instead fire the next sensor in the fixed order that is ready. As this allows sensors to be skipped, it may seem, at first, that this would allow the sensors to fire non-sequentially but because the sensors are still being processed in a fixed order, but this is not the case. The next sensor fired is the next sensor in the fixed order that has received its echo and is ready to fire but this sensor may not be the first sensor that became ready to fire. For example consider the case where there are three sensors, A, B and C that are processed in the fixed order of A, B, C and then A again. A possible scenario is that A, B and C are fired before any echoes are received. Then the first sensor to receive its echo is C and this is followed by B receiving its echo. If at this time the system is processing A, it will skip the firing

of A as it has not yet received its echo and move onto B. Now because B has received its echo, it is again scheduled to fire and will therefore fire before C even though C was the first sensor to receive its echo.

In a truly non-sequential system, as soon as a sensor receives its echo and becomes ready to fire it is put on a queue of sensors waiting to fire. The sensor at the front of the queue is then selected for the next firing. This ensures that the next sensor fired is always the one that first became ready to fire and the sensors fire in the same order that they receive their echoes.

It would be relatively easy to implement such a scheme on a simple ultrasonic ranging system that uses a constant delay between firings as the order that the sensors fire does not matter. However it would not be possible to implement non-sequential firing on a system that uses predetermined variable delays such as EERUF. This is because the predetermined variable delays have been calculated for a fixed firing schedule and the algorithm would not function correctly if the sensors did not fire in the correct sequence. When the variable delays are calculated dynamically this is no longer a problem as the correct firing time for the next sensor fired is recalculated every time the sensor fires, ensuring crosstalk can be detected, regardless of the firing order.

The algorithm presented in the previous section for determining the next firing times is for a synchronous, sequential system. However, the algorithm can also be used, as is, for an asynchronous system by simply calculating the next firing time of the next sensor directly after the last sensor fires, instead of waiting for the last sensor's echo to be received before doing so. Therefore $T_{current}$ is now set to the current time directly after the last sensor fires and T_{min_delay} now represents the actual delay between sensors firing, regardless of the time taken to receive the echoes. This does not have any other effect on the algorithm because as shown in section 5.1.2. by equation (2) and *fig 8.* this scenario is equivalent to the delays occurring after the echo is received and the principles on which the algorithm are based still hold.

The algorithm can also be extended to allow for non-sequential firing by defining $S_k lastDelay_i$ to be the last recorded delay time between the last time S_k and S_i fired for a given sensor S_k and all sensor S_i in the set of all sensors used. The resulting algorithm that uses $S_k lastDelay_i$ is the RUFACE algorithm and is defined as follows:

1. $S_k next = T_{current} + T_{min\_delay}$
2. *foreach* $S_i \in all\ sensors$
   - 2.1. $minAllowed_i = S_k lastDelay_i + S_i last - T_\Delta$
   - 2.2. add $minAllowed_i$ to $minAllowedList$
3. *foreach* $minAllowed_i \in minAllowed\ list$
   - 3.1. *if* $S_k next > minAllowed_i$
     - 3.1.1. remove $minAllowed_i$ from the $minAllowedList$
     - 3.1.2. $maxAllowed_i = S_k lastDelay_i + S_i last + T_\Delta$
     - 3.1.3. *if* $maxAllowed_i > S_k next$
       - 3.1.3.1. $S_k next = maxAllowed_i$
       - 3.1.3.2. GOTO 3.
4. *foreach* $S_i \in all\ sensors$
   - 4.1. $S_k lastDelay_i = S_k next - S_i last$
5. $S_k last = S_k next$

Figure 14: The RUFACE algorithm

The record of the last delays, $S_k lastDelay_i$, gets updated at the end of the algorithm when the next firing time for S_k has been determined. The algorithm therefore only takes into account the previous delays from the other sensors, from the last time sensor S_k fired. The other sensors can therefore fire an arbitrary number of times, in an arbitrary order before sensor S_k fires again and the algorithm is still able to ensure that equation (5) from the previous section is satisfied.

The value of T_Δ is a configurable setting of the RUFACE algorithm. The RUFACE algorithm ensures the delays always differ by at least T_Δ . This parameter of the RUFACE algorithm is therefore referred to as the delay difference setting. The other configurable setting it the minimum delay setting. RUFACE ensures the delays between sensors firing is at least this minimum delay.

In theory, the RUFACE algorithm offers an efficient method for dynamically determining the next firing time of any sensor in a non-scheduled, asynchronous and non-sequential manner, while still ensuring that crosstalk between any of the sensors can be detected. In order to determine the effectiveness of the RUFACE algorithm on the ultrasonic ranging system requires testing and this is the topic of the next section.

5.3. RUFACE Testing

Ideally all the potential benefits of the RUFACE algorithm would be tested extensively. Unfortunately, due to time constraints, this is beyond the scope of this project. At this stage, the part of the RUFACE algorithm that is most important for the model helicopter ultrasonic ranging system is its ability to detect and filter crosstalk errors. This has therefore been tested by the following procedure:

The ultrasonic ranging system is positioned at a distance of 1 metre from the indoor arena wall, facing forwards at an angle of 40^{o} to the perpendicular, and at a height of 2 metres off the ground. This is a position in which a large amount of crosstalk was found to occur.

At first only the forward facing ultrasonic device is enabled and 100 readings are taken with a minimum delay setting of 30ms to ensure that no crosstalk occurs. These readings are therefore known to be correct and are used as a reference to check the correctness of the readings from the tests that follow.

All five ultrasonic ranging devices are then enabled and the RUFACE algorithm is disabled. 100 readings are then taken from the forward facing ultrasonic device for each minimum delay settings from 0ms to 12ms with increments of 1ms. The RUFACE algorithm is then enabled and the previous test is repeated for each delay difference setting of the RUFACE algorithm from 0.1ms to 2.0ms with increments of 0.1ms.

5.3.1. Analysis and Results

Looking at the range data obtained with a minimum delay of 5ms and the RUFACE algorithm disabled, as illustrated by the graph in *fig 15.,* the effects of crosstalk can clearly be seen when comparing the results from a single SRF10 firing and multiple SRF10s firing. When only the single forward facing SRF10 is firing it produces range readings that are consistently correct. However when multiple SRF10s are fired together the range readings produced by the forward facing SRF10 are incorrect as a result of crosstalk, and, as predicted by the theory presented at the beginning of this chapter, the incorrect readings are almost always the same. These erroneous readings are therefore not detectable and this clearly demonstrates why an algorithm such as RUFACE is required.

The graph in *fig 16.* shows the same test but this time with the RUFACE algorithm enabled with a delay difference setting of 1ms. As can be seen the erroneous readings resulting from crosstalk when multiple SRF10s fire together are now very much different from one reading to the next. This means that these errors can be filtered by the comparison of consecutive readings filter.

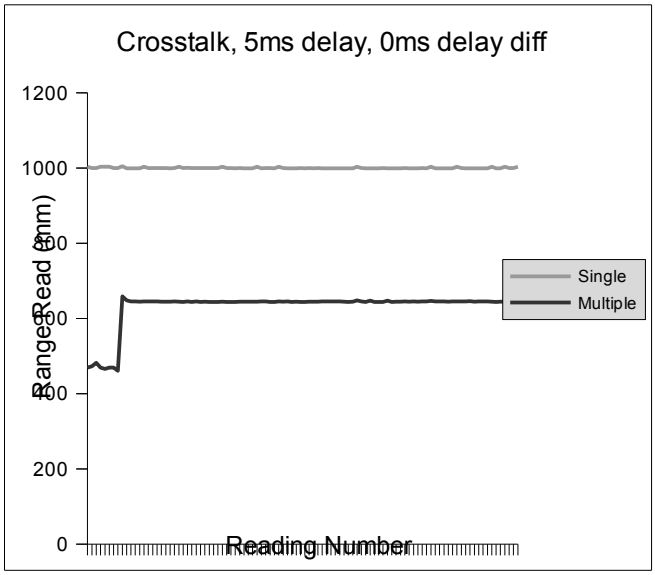

Figure 15: Crosstalk effected range readings without RUFACE

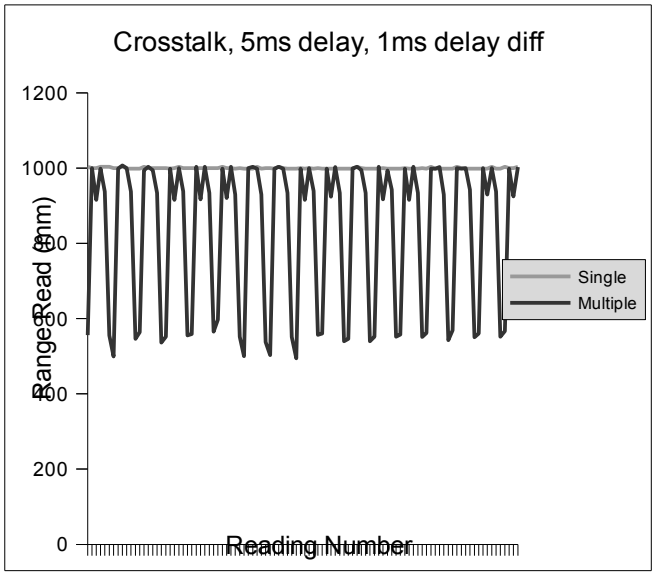

Figure 16: Crosstalk effected range readings with RUFACE

The comparison of consecutive readings filter is then applied to the range data with a rejection threshold of 20mm. The percentage of incorrect readings that are not filtered are shown in *Table 4.* A reading is considered incorrect, in this case, if it differed from the mean of the range data

obtained from the reference test by more than 20mm. The columns represent minimum delay settings from 0ms to 12 ms and the rows represent delay difference settings from 0ms to 2.0ms. The first row, with a delay difference of 0 ms, show the percentage of incorrect readings with the RUFACE algorithm disabled.

As can be seen the percentage of incorrect readings that do not get filtered is very high for low minimum delay settings. It is only after the minimum delay is at 10ms and higher that the percentage of incorrect readings drops to 0. It will be shown, shortly, that the reason for this is that there was no crosstalk when a minimum delay of 10ms or more was used. The remaining columns show the percentages of incorrect readings with the RUFACE algorithm enabled, as the delay difference setting is increased to 2.0ms in increments of 0.1ms. As can be seen there is an immediate, dramatic drop in the percentage of incorrect readings that do not get filtered, even with a delay difference setting of only 0.1ms. For higher delay difference settings, above 1.5ms, almost all incorrect readings are filtered.

	0	1	2	3	4	5	6	7	8	9	10
0	98.99	100	100	100	100	54.55	38.38	0	62.63	0	0
1	22.22	19.19	19.19	21.21	41.41	55.56	15.15	0	0	0	0
2	23.23	18.18	15.15	25.25	16.16	28.28	0	0	0	0	0
3	16.16	14.14	14.14	11.11	25.25	17.17	3.03	0	0	0	0
4	19.19	15.15	16.16	8.08	18.18	16.16	2.02	0	0	0	0
5	15.15	16.16	16.16	8.08	26.26	12.12	0	0	0	0	0
6	9.09	9.09	11.11	8.08	4.04	7.07	0	0	0	0	0
7	4.04	10.1	6.06	10.1	0	10.1	0	1.01	0	0	0
8	13.13	13.13	13.13	16.16	2.02	4.04	0	0	0	0	0
9	8.08	9.09	1.01	1.01	2.02	0	0	0	0	0	0
10	10.1	13.13	9.09	7.07	4.04	9.09	0	0	0	0	0
11	7.07	11.11	6.06	0	0	11.11	0	0	0	0	0
12	6.06	10.1	13.13	0	2.02	0	0	0	0	0	0
13	3.03	4.04	4.04	1.01	0	0	0	0	0	0	0
14	10.1	1.01	0	0	0	3.03	0	0	0	0	0
15	0	0	0	0	0	1.01	0	0	0	0	0
16	0	0	0	0	1.01	0	0	0	0	0	0
17	1.01	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0
19	2.02	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0

Table 4: Percentage incorrect readings with RUFACE after filtering
The columns are with various delay settings in ms, the rows are with various delay difference settings in ms

The results of *table 4.* can be more clearly demonstrated when the percentage of incorrect readings are shown alongside the percentage of correct and filtered readings as shown by the graphs of *fig 17.* and *fig 18.* With these graphs the percentage of incorrect, correct and filtered readings are

stacked together and the sum of these three values is always 100 percent. The first graph represents the range data taken from the forward facing SRF10 with all other SRF10s firing and the RUFACE algorithm disabled. As can be seen, almost all the readings are incorrect and are not filtered when the minimum delay setting is low. It is only after the minimum delay setting goes higher than 9ms that crosstalk no longer occurs and all the readings are correct. The important observation here is that hardly any of the incorrect readings are filtered which demonstrates that crosstalk is not detectable under normal circumstances.

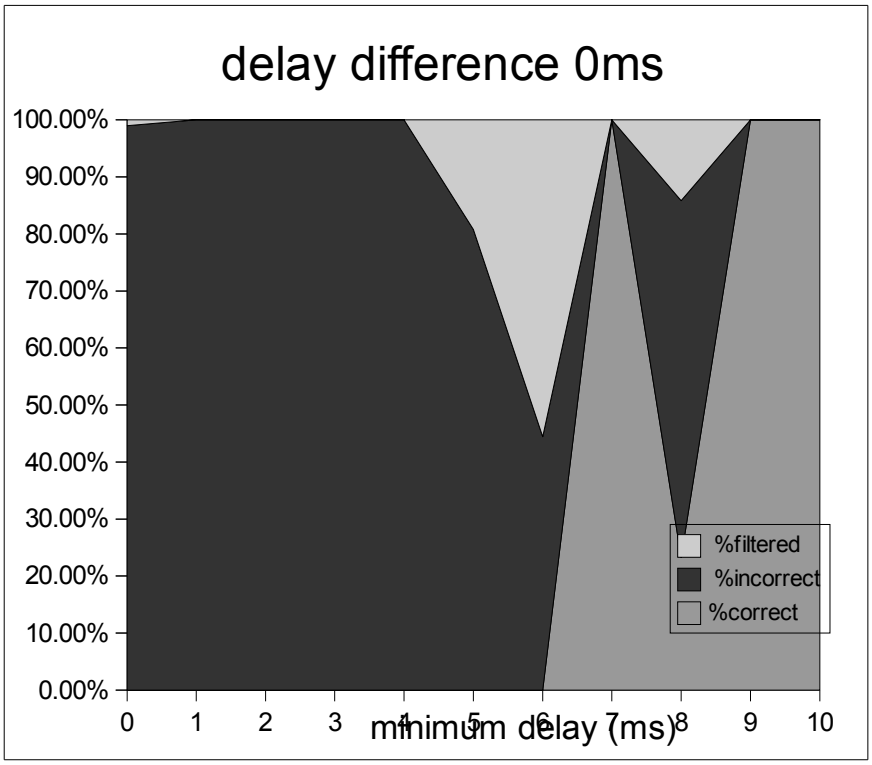

Figure 17: Percent correct, incorrect and filtered without RUFACE

The second graph represents the range data taken from the forward facing SRF10 with all other SRF10s firing and the RUFACE algorithm enabled with a delay difference setting of 1.0ms. The graph clearly shows that there is a dramatic increase in the number of incorrect readings that are filtered. Although there are slightly more correct readings at lower minimum delay settings with the RUFACE algorithm enabled as compared to with it disable, it is again only after the minimum delay setting goes higher than 9ms that crosstalk no longer occurs and all the readings are correct. Therefore the RUFACE algorithm makes crosstalk detectable but does not directly reduce the amount of crosstalk occurring. However, it will be shown shortly that the RUFACE algorithm can be used indirectly to reduce the amount of crosstalk occuring.

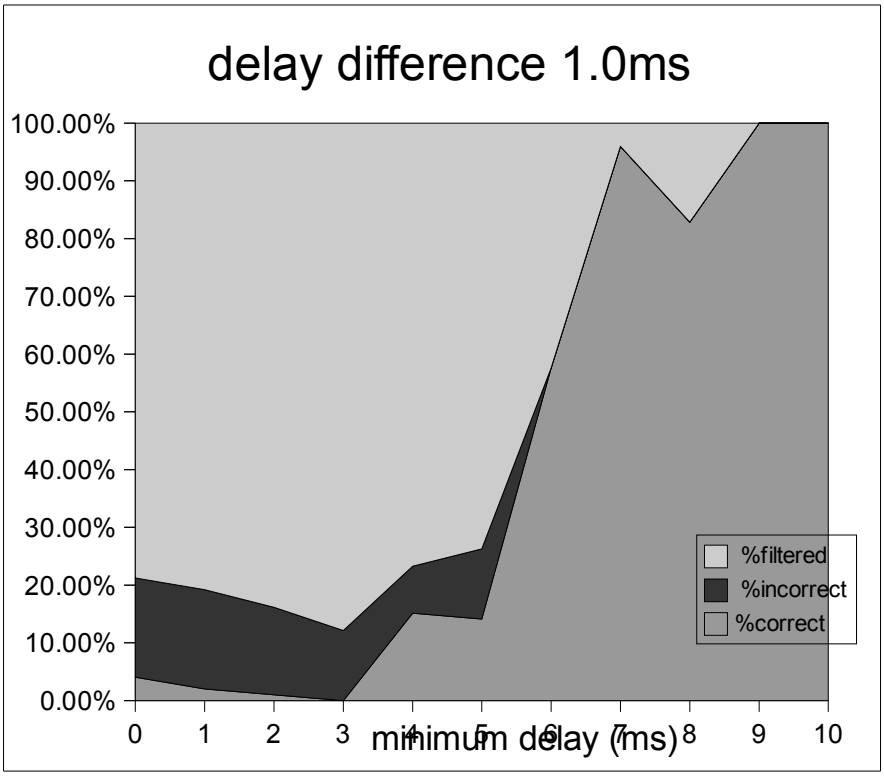

Figure 18: Percent correct, incorrect and filtered with RUFACE

As can be seen by the graphs, the optimum minimum delay setting, in this particular case, is 9ms when all the readings are correct and crosstalk is no longer occurring. However it is important to emphasise that differentiating between correct and incorrect readings in these graphs is done by comparing the reading to the mean of the readings taken from the reference test, which is known to be correct. Under normal circumstances it would not be possible for the ultrasonic ranging system to determine if a reading is correct or incorrect. The only information available is the percentage of readings that are filtered by the comparison of consecutive readings method.

When the RUFACE algorithm is disabled, very few readings get filtered, and the ultrasonic ranging system has no way to determine that 9ms is the current optimum value for the minimum delay. However, when the RUFACE algorithm is enabled, a large percentage of the readings get filtered before the minimum delay increases to the optimum setting of 9ms. It would therefore be possible for the ultrasonic ranging system to determine the optimum minimum delay by monitoring the percentage of readings that get filtered. The flexibility provided by the RUFACE algorithm allows the minimum delay setting to be changed at any time. The process of monitoring the percentage of filtered readings and adjusting the minimum delay setting accordingly can therefore occur on a continuous basis, resulting in an adaptive firing rate solution.

5.4. RUFACE Conclusion

When using an array of time of flight ultrasonic sensors a major problem encountered is crosstalk. However, by firing the sensors at precise times the crosstalk can be detected and filtered. Existing methods such as EERUF (1) achieve this by using specially designed, predetermine, fixed firing schedules. A new approach is shown to be possible by calculating crosstalk detectable sensor firing times dynamically in real-time.

The firing times are therefore non-scheduled and this allows for a dynamic firing rate that can be modified at any time. An adaptive firing rate based on the rate that readings are filtered is therefore possible. Higher filtering rates indicate a lower firing rate is required and lower filtering rates indicate a higher firing rate is possible.

In addition, the approach has been extended to allow for asynchronous, non-sequential firing, where sensors that are closer to objects fire more rapidly than sensors that are further away from objects. More ranging data is therefore produced where it is needed. The result is a novel ultrasonic firing algorithm called RUFACE which is suitable for use on the helicopter ultrasonic ranging system to effectively eliminate crosstalk. In addition, the RUFACE is based on the same principles as EERUF which is claimed to allow multiple ultrasonic system to operate in the same environment (1, pp1). Therefore using RUFACE will allow multiple helicopters to operate at the same time, which is a requirement stipulated in chapter 3.

6. Implementation

This chapter details the platform, design, implementation and testing of the ultrasonic ranging system. The system has been designed to meet all the requirements specified in chapter 3.

6.1. System Architecture

From a high level perspective, the ultrasonic ranging system is comprised of an ultrasonic embedded system and a client, as shown in *fig 19*.

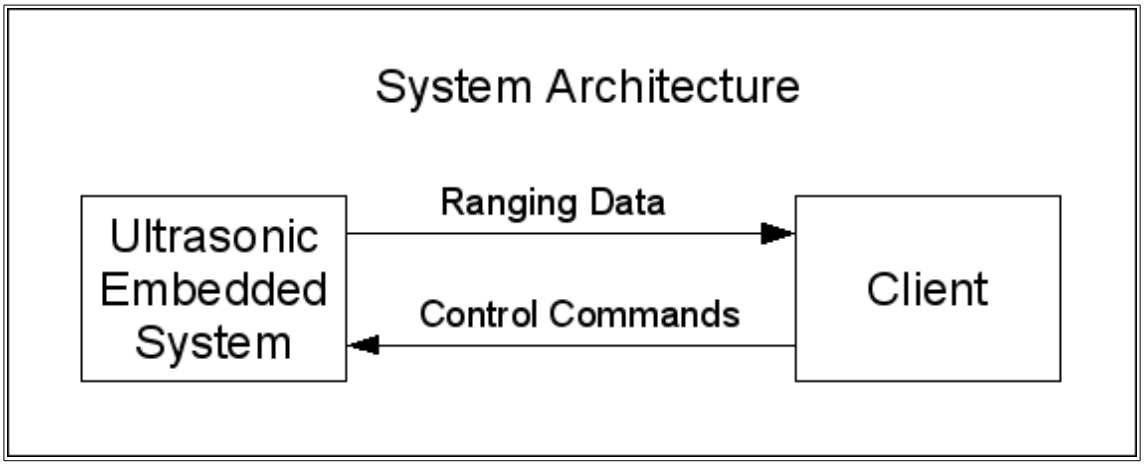

Figure 19: The system architecture

The SRF10 ultrasonic devices are contained within the ultrasonic embedded system, which provides the client with the ranging data as it becomes available from the sensors. The client is able to control the operation of the ultrasonic embedded system by sending it control commands. The ultrasonic embedded system accepts the following commands:

- Start all SRF10 devices firing.
- Stop all SRF10 devices firing.
- Start any individual SRF10 device firing.
- Set the minimum delay between sensors firing.

- Set the minimum delay difference between the firing delays. This is the T_{Δ} parameter of the RUFACE algorithm.
- Set the analogue gain setting of any or all SRF10 sensor(s)
- Set the maximum range setting of any or all SRF10 sensor(s)

All the ranging data obtained from the SRF10 devices is tagged with an identifier of the SRF10 device that produced the reading and forwarded directly to the client. All post-processing of the data, for example filtering by comparison of consecutive readings, occurs at the client.

This architecture allows the ultrasonic embedded system to be independent of the client which has a number of benefits. Different clients, for example a test system on a personal computer, or an obstacle avoidance system on a model helicopter, can be used without making any modifications to the ultrasonic embedded system which simply provides the client with the raw ranging data. The ultrasonic embedded system does not have to be concerned with what the data is being used for. The overhead of post-processing the data is offloaded from the ultrasonic embedded system's microcontroller to the client where it can be dealt with by a more powerful processor.

6.2. The Development Platform

The SRF10 devices are controlled by a microcontroller that issues firing commands to the desired SRF10 device at the desired rate and reads the corresponding range output in milliseconds over an I^2C bus. The microcontroller is connected to a client via a serial connection and all ranging and timing output is sent to the client for processing and analysis.

6.2.1. Hardware Platform

As detailed in chapter 5, the ultrasonic sensor array is comprised of five SRF10 ultrasonic devices, positioned at 90^o to each other with the devices facing, forwards, backwards, left, right and downwards.

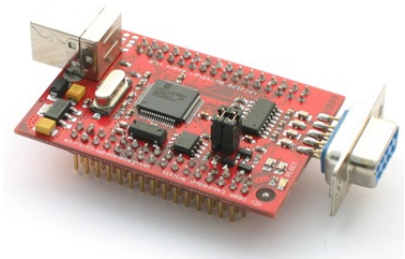

Figure 20: The Embedded Artists' LPC2138 Quickstart Board

The ultrasonic embedded system is implemented on the Embedded Artists' LPC2138 Quickstart Board (*Fig 20*). This board has been selected for the prototype ultrasonic system for the following reasons:

- It incorporates the Philips ARM7TDMI LPC2138 microcontroller. This is similar to the type of microcontroller selected for use on the model helicopters by the UltraSwarm project, which are also based on the ARM7 core. It will therefore be easy to port a system that runs on the Philips ARM7TDMI LPC2138 microcontroller onto the helicopters' microcontrollers when required at a later stage. The features of the Philips ARM7TDMI LPC2138 microcontroller that are relevant to the ultrasonic embedded system are the following:
  - A CPU clock speed of 58,9824 MHz which provides more than enough processing power to run the embedded ultrasonic system and the RUFACE algorithm.
  - An I^2C module that provide an interface for communicating with and controlling the SRF10 devices.
  - A UART module that provides an interface for communicating with the client.
  - A Timer module that allows the firing of the SRF10 devices to be accurately timed, which is necessary to effectively implement the RUFACE algorithm.

- A PWM module for generating PWM frequencies, used for controlling the servo motor used for the testing platform that is discussed below.

- It offers convenient development platform as it provides an RS232, DSUB-9 connector that makes it easy to upload the ultrasonic embedded system's program code onto the microcontroller's flash program memory. The RS232 port is also connected to microcontroller's UART interface and this allows the test client to connect to the ultrasonic embedded system using a standard serial data cable.

- It provides an I^2C bus with connectors pins for connecting the SRF10 devices to the microcontroller's I^2C module.

The test client is a laptop computer that connects to the ultrasonic system via a standard serial data cable

6.2.2. Software Platform

The ultrasonic system was programmed using C for the low level libraries for fine grained control of the microcontroller and its hardware modules. C++ was used to program the high level components as objects as this simplified the design and therefore allowed for a more powerful design. This results in a hierarchical architecture with the higher level modules using the lower level modules to complete their tasks. The source code for the ultrasonic system is given in *appendix SC1*.

The compiler used was GCC with the GNUARM toolchain extension because it is freely available and allowed standard Linux development tools to be used

The Test client was programmed in Java to allow for rapid development so more time could be spent on the ultrasonic system. The source code for the client is given in *appendix SC2*.

6.2.3. Testing Platform

The UltraSwarm helicopters were not yet fully functional at the time of development of the

ultrasonic ranging system. The prototype system was therefore built on a testing platform. The testing platform consists of a polycarbonate box in which the microcontroller is housed. The five SRF10s are mounted centrally on each the box's external surfaces, except the top of the box as this is the same configuration of SRF10s that will be used for the sensor array on the UltraSwarm helicopters. A servo motor has been attached to the top of the box, and the servo motor is attached to the base of a wooden pole. The servo motor has a 180° range and this allows the box to be accurately rotated to a specific angle, which was necessary for testing. The wooden pole is attached to an elevated, wheeled base which allows the testing platform to moved around to any location in the arena. Photographs of the testing platform can be seen in *appendix 5*.

6.3. Ultrasonic Embedded System Design

In the field of computer science, modularity is a fundamental principle that is the key to producing good system designs. The ultrasonic system is therefore decomposed into distinct modules as shown in the following block diagram (*fig 21*).

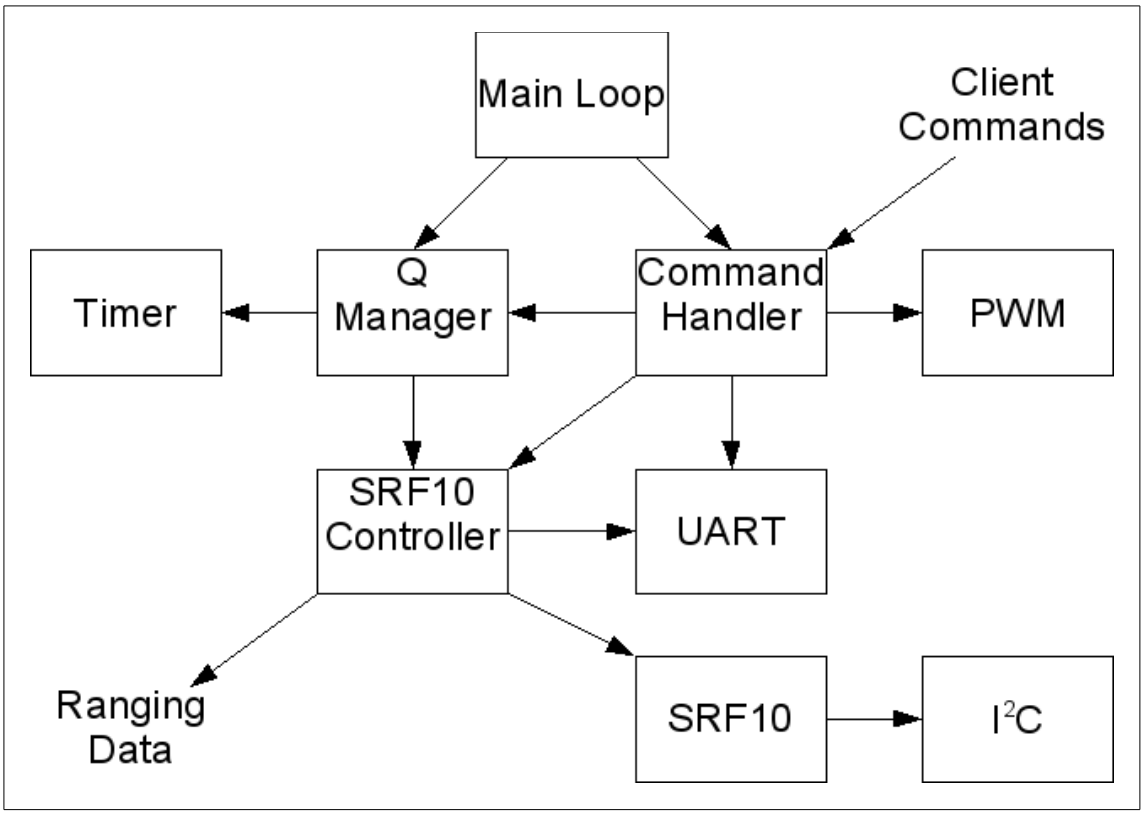

Figure 21: The system modules and their organisation

Each of these modules and their interaction will now be discussed in the sections that follow.

6.3.1. Low Level Modules

The I^2C, UART, Timer and PWM modules provide simple interfaces to the associated microcontroller hardware modules. This allows the higher level modules, presented below, to use these hardware modules without having to be concerned with the low level hardware details.

6.3.2. Command Handler

The command handler receives the client command via UART and performs the necessary actions. The list of available commands was presented in section 6.1. The command handler also, on request, provides the client with information about the current state of the system. This information includes the following:

- The currently set minimum delay between firing
- The currently set minimum delay difference between sensors
- Information about SRF10 devices installed on the system including:
  - The I^2C address of the device
  - The device identifier
  - The currently set analogue gain value
  - The currently set maximum range value

The Command Handler is allowed to execute one step per iteration of the main loop. During this step the Command Handler checks to see if any characters have been received over UART, indicating the client would like to send a request. If this is the case the Command Handler follows a simple ASCII character based protocol and sends an acknowledgement to the client and then waits for the client's request. On receipt of the acknowledgement the client knows the Command Handler is ready to receive the request and sends it over UART. The Command Handler then receives the client's request and executes the necessary action. Program control is then returned to the main loop.

6.3.3. SRF10

The SRF10 module is comprised of multiple object instances, one per physical SRF10 device, that are responsible for communicating with each of the physical SRF10 devices over I^2C. The following commands can be issued:

- **doRange** – instructs the SRF10 device to initiate its ranging sequence by firing a pulse.
- **doPoll** – checks whether the SRF10 has received its echo after firing.
- **doRequestRangeData** – issued before doReadRange to set the internal address of the register to be read to that of the range data.
- **doReadRangeData** – instructs the SRF10 to send the range data over the I^2C bus and then read the range data from the I^2C bus.
- **setMaxRange** – set the maximum range of the SRF10 device.
- **setGain** – set the maximum analogue gain of the SRF10 device.

6.3.4. SRF10 Controller

Each of the SRF10 object instances are owned by an SRF10 Controller object instance. There is therefore an SRF10 Controller for each physical SRF10 device. The SRF10Controller has the following responsibilities:

- Managing the state of its SRF10 object.
- Managing the commands that the SRF10 object sends over the I^2C bus.
- Managing the timing for firing its associated SRF10 device.
- Forwarding range data obtained from the SRF10 device to the client via UART

The SRF10 Controller cycles its SRF10 object's state when certain events occur as illustrated by the following diagram (*fig 22*).

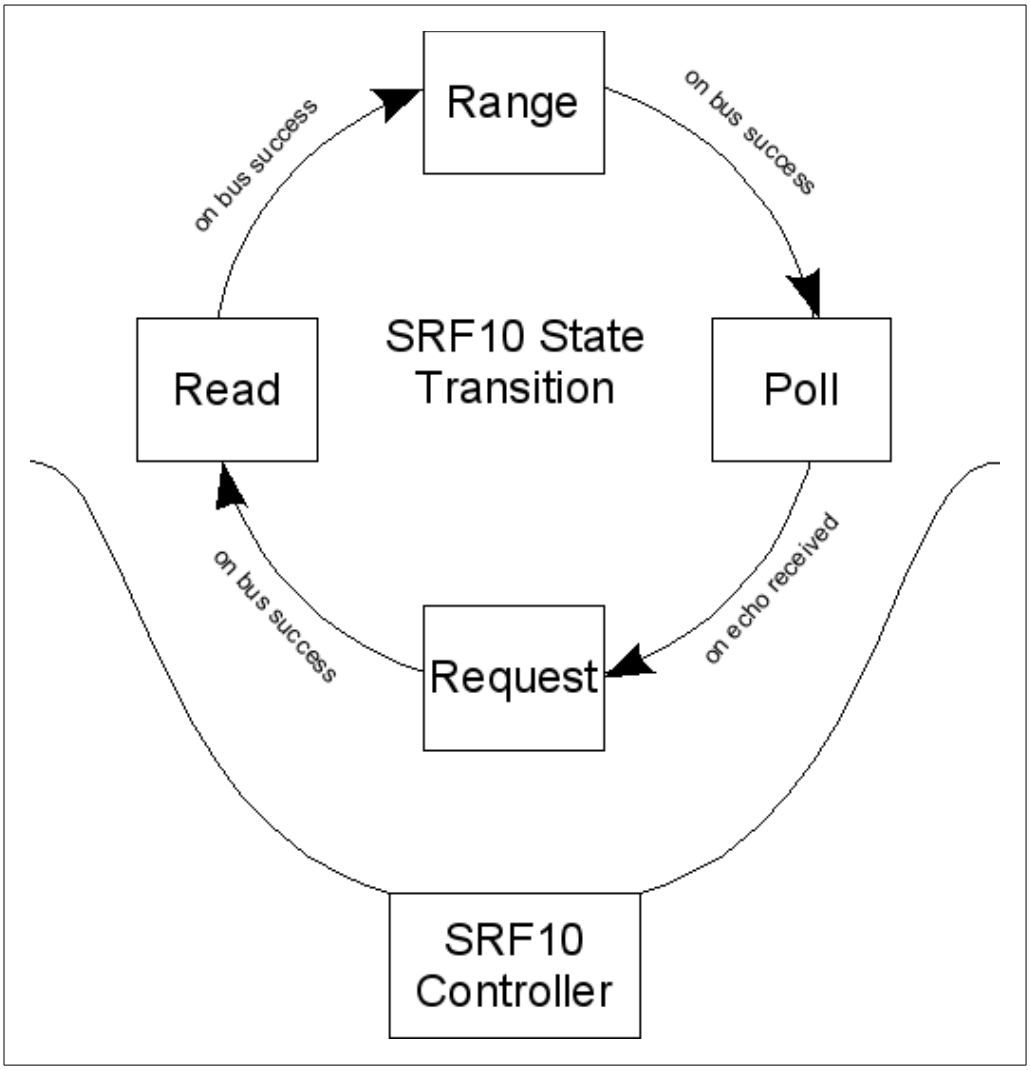

Figure 22: SRF10 State Transition

The SRF10 Controller instructs its SRF10 object to issue commands to the the physical SRF10 device over the I^2C bus for performing the necessary actions associated with each state. This ensures that the correct commands are issued to the SRF10s in the correct order. The request, read and range states are progressed when the associated command has been successfully executed. If there is an I^2C failure the state is not progressed and the associated command is sent again at the next available opportunity. The poll state is progressed when it has been determined that the SRF10 has received its echo, until then the poll command is issued repeatedly at every available opportunity.

The SRF10 Controller also schedules the timing of firing of the SRF10 using timer interrupts. Whenever the range state is entered the appropriate delay is calculated and a timer interrupt

scheduled. As soon as the timer interrupt occurs the I^2C range command is issued. If there is an I^2C failure a new delay time is calculated and the process is repeated. The delay time is calculated using the RUFACE algorithm, if it is enabled, else the delay is calculated using the minimum delay setting only. When calculating the value that the timer interrupt will be set to, the time that has already passed from the last sensor firing is taken into consideration and subtracted from the desired delay. For example, if the minimum delay setting is 10ms, but 3ms have already passed since the last sensor fired, then the timer interrupt will be set to occur in 7ms. This allows for more precise scheduling of sensor firings which is necessary for the RUFACE algorithm to be effective.

6.3.5. Q Manager

There is the problem that there are multiple SRF10 Controllers requiring the use of a single timer and a single I2C bus. The Q Manager coordinates and synchronises the SRF10 Controllers' actions by using queues. This allows the resources to be shared and the SRF10 Controllers to operate independently of each other. This is illustrated by the following diagram (*fig 23*).

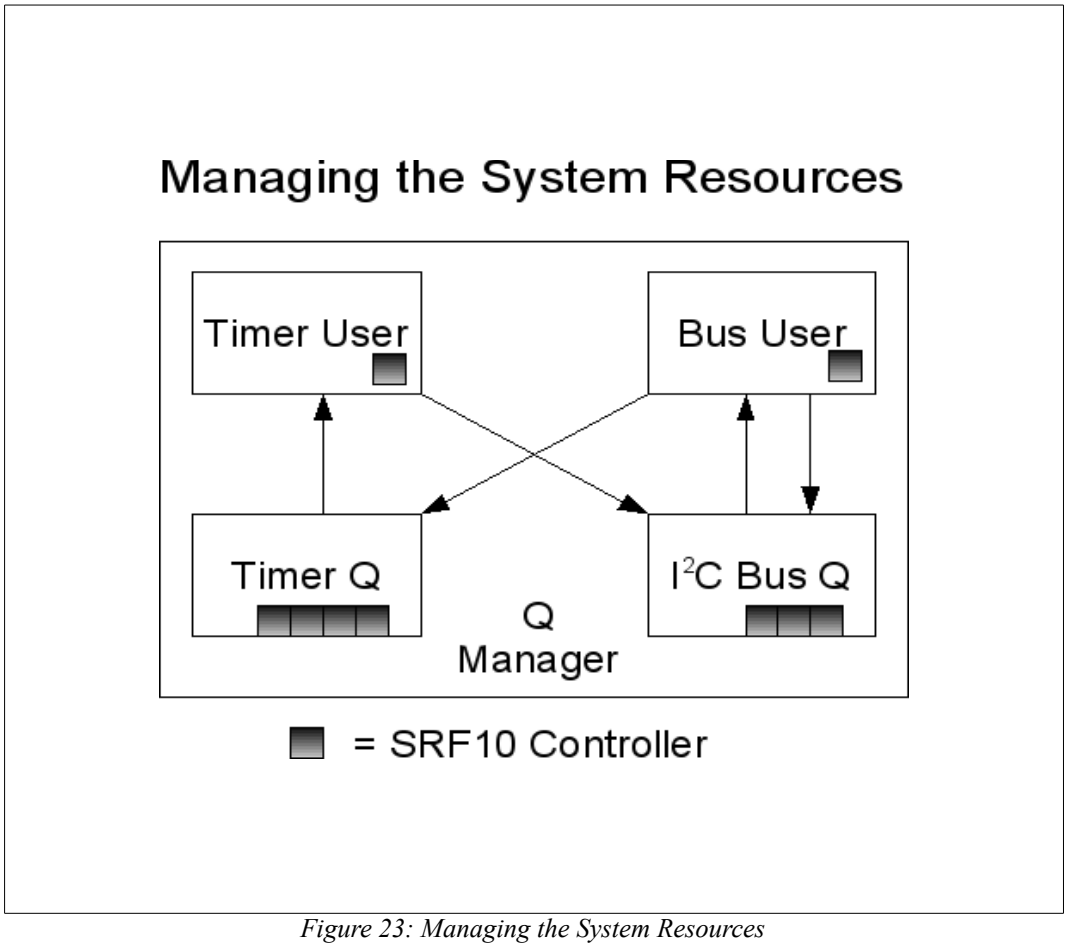

Figure 23: Managing the System Resources

SRF10 Controllers waiting to use the I^2C bus are in the bus queue and SRF10 Controllers waiting to use the timer are in the timer queue. As soon as the resource becomes available, the Q Manager is notified by an interrupt and the controller at the front of the queue gets set as the resource user which gives it mutually exclusive access to the resource. Once a controller has finished using the resource is is moved to the appropriate queue according to its state and the next SRF10 Controller is set as the resource user. SRF10 Controllers on the timer queue are waiting to use the system timer to schedule and interrupt. When the timer interrupt occurs this signals that the SRF10Controller wants to fire its SRF10 device. The SRF10Controller is therefore moved to the front of the bus queue so that the firing command can be issued over the I^2C bus as soon as possible. This ensures that the actual time the SRF10 fires is as close to the calculated time as possible which is necessary for the RUFACE algorithm to be effective.

The approach of using queues allows the system to easily be extended and have additional SRF10 devices connected with only minor modification.

Although interrupts are used to trigger events and coordinate the actions of the SRF10 Controllers, all actions are executed serially within the main program loop to minimises the time spent executing interrupt service routines. The Q Manager is allowed to execute one step per iteration of the main loop. During this step the Q Manager checks to see if any interrupts have occurred that indicate an SRF10 Controller has finished using a resource. If this is the case the SRF10 Controller that was using the resource is moved to the appropriate queue depending on its state. The next SRF10 Controller in the queue is set as the resource user. The program control is then passed to this SRF10 Controller so it can execute its next command and use the resource. The program control is then returned to the main loop.

6.3.6. Main Loop

The main loop is a continuous loop that drives the system by executing the entry functions of the Q Manager and Command Handler modules once per iteration where they perform their required tasks, if any, and return the program control back to the main loop.

6.4. Problems Encountered

Implementing the system proved to be a demanding and challenging task. The actions of the multiple SRF10 devices have to be flawlessly synchronized and coordinated and for the RUFACE

algorithm to be effective this has to be done with precision timing.

A major problem that was encountered was that the SRF10s intermittently crashed, i.e. stopped responding indefinitely. The problem was difficult to reproduce. Eventually it was found that it only happened if an I^2C command were issued immediately after the effected SRF10 has fired but the cause of the problem is still unclear. A temporary solution has been employed by pausing the I^2C bus, i.e. not allowing any commands to be issued to any SRF10s, for a period of 1.25ms after an SRF10 fires. Unfortunately this does reduce the maximum firing rate capable by the system but is necessary for the sake of stability. It is shown below that even with the 1.25ms pause, the system's maximum firing rate is still higher than required.

6.5. Software Testing

Each software module of the system has been tested in isolation to ensure they function correctly independently. This was done with the use of test scripts that simulated input to the module and the resulting output was checked for correctness. System testing was performed by outputting debugging information while the system ran. The debugging information was then analysed to ensure the system functions correctly as a whole.

6.6. Implementation Testing

The implementation was tested by outputting the firing times of all the sensors as they fired over the UART interface where they were read by a laptop connected to the system via a serial data cable. 250 sequential firing times each for various minimum delay and delay difference settings were captured and stored in data files (*appendix 4*). The timing data was then analysed to ensure the system works correctly and that the sensors fire at the correct times both with and without the RUFACE algorithm enabled.

6.6.1. Analysis and Results

The delay times between firings were extracted from the timing data by calculating the difference from a firing time to the previous firing time. This data was then analysed to determine if the actual delay times between sensors firing is correct. *Table 5.* shows the minimum, maximum, average and standard deviation of the delay times for various minimum delay settings, from 0ms to 50ms, with the RUFACE algorithm disabled.

	minimum	maximum	average	std deviation
0	2.31	4.36	3.36	0.66
1	2.31	4.36	3.36	0.65
2	2.31	4.36	3.36	0.65
3	3.27	3.44	3.35	0.06
4	3.35	4.62	4	0.55
5	3.93	6.08	5.01	0.31
6	4.99	7.03	6.01	0.18
7	6.83	7.19	7.01	0.02
8	8	8.02	8.01	0
9	9	9.02	9.01	0
10	10	10.01	10.01	0
20	20	20.01	20.01	0
30	30	30.01	30.01	0
40	40	40.01	40.01	0
50	50	50.01	50.01	0

Table 5: Firing delays with various minimum delay settings (time in ms)

The results show that the system is not able to accurately maintain the set delay for minimum delay settings below 5ms. This is because these settings are being effected by the system's maximum firing rate which is limited due to factors such as:

- Pausing the I^2C bus for 1.25ms after an SRF10 fires to prevent it crashing
- I^2C bus latency when sending SRF10 commands
- UART latency when sending range and timing data to the client

For minimum delay settings of 5ms to 7ms the system is able to maintain the required delay between sensors firing within a low deviation and above 7ms the actual delays are equivalent to the minimum delay settings. The deviation for 5ms and above is considered acceptable which means the system is capable of firing with accurate delays to a maximum firing rate of 200Hz.

The delay times produced, when the RUFACE algorithm is enabled, were analysed for various minimum delay and delay difference settings. *Table 6.* provides the results of calculating the minimum delay difference that occurred for each setting combination from the time data. The columns are for delay difference settings from 0ms to 5ms and the rows are for minimum delay settings from 0ms to 10ms.

	0	1	2	3	4	5
0	0	0.03	0.89	1.97	2.51	4.52
1	0	0.06	0.56	1.97	2.61	4.62
2	0	0.01	0.89	1.97	2.62	4.51
3	0	0.01	1.33	1.96	3.15	3.98
4	0	0.16	1.17	2.16	3.17	4.17
5	0	0.99	1.99	2.99	3.99	4.98
6	0	0.99	1.99	2.99	3.99	4.99
7	0	0.99	1.99	2.99	3.98	4.98
8	0	0.99	1.99	2.99	3.98	4.99
9	0	0.99	1.99	2.99	3.98	4.99
10	0	0.99	1.99	2.99	3.98	4.99

Table 6: Actual delay difference times with various delay difference settings (time in ms)

For similar reasons as above, the delay differences are not accurate for minimum delay settings below 5ms as the resulting firing rate is above the system's maximum. However, for minimum delay setting of 5ms and above, the system is able to ensure the required delay differences are produced. Therefore the system is capable of running the RUFACE algorithm to a maximum firing rate of 200Hz.

6.7. Implementation Conclusion

A fully functional embedded model helicopter ultrasonic ranging system has been successfully implemented using easily available off-the-shelf components, that fulfills all the requirements specified in chapter 3.

The system incorporates a sophisticated, elegant design that coordinates the actions of the sensors dynamically which allows for flexible firing sequences and scalability. Although only five SRF10 ultrasonic ranging devices are used at this stage, the design allows more SRF10 devices to be easily connected, if this becomes necessary, with only minor modifications.

The system implements the RUFACE algorithm and is therefore capable of eliminating crosstalk errors, allows multiple ultrasonic systems to operate in the same environment and is capable of supporting an adaptive firing rate.

The cause of the SRF10 crashes is still unknown and a temporary solution has been employed of pausing the I^2C bus for a period of 1.25ms after an SRF10 fires. Unfortunately this does incur a performance penalty, but the system has been shown to still be able to fire its sensors with accurate timing up to a maximum firing rate of 200Hz. There are five SRF10s, which means that each sensor can fire up to a rate of 40Hz which is double the specified rate required. The system is therefor capable of supporting ten SRF10 sensors within the requirements.

7. Conclusion

The goals of the project have been met. The project has demonstrated that current technologies can be applied to provide ultrasonic ranging capabilities to small scale model helicopters by developing a fully functional embedded ultrasonic ranging system.

The developed ultrasonic ranging system currently uses an array of five SRF10 ultrasonic sensors but is capable of supporting up to ten SRF10 devices. The SRF10 ultrasonic device has been shown to be suitable for the intended purpose but that it is also susceptible to crosstalk when multiple SRF10 are fired in quick succession. The system therefore employs a method for eliminating crosstalk.

The problem of crosstalk and the theory behind why it is not detectable under normal circumstances has been thoroughly investigated. The result is the development of a new algorithm, the Real-Time Ultrasonic Firing Algorithm for Crosstalk Elimination (RUFACE). RUFACE is based on the Error Eliminating Rapid Ultrasonic Firing (EERUF) algorithm (1) but extends this approach using the novel method of calculating crosstalk detectable firing sequences dynamically in real-time. RUFACE has been shown to be an effective alternative to existing crosstalk eliminating approaches and has the additional benefits of allowing for non-scheduled firing, non-sequential firing and an adaptive firing rate.

7.1. Project Management

This scope of this project has been wide and it has been comprised of many challenging tasks. The successful completion of the project, required careful planning and project management to effectively manage the short amount of time available and coordinate the many tasks. The individual tasks of the project were identified at the start and a project plan was produced and followed (*appendix 1*). The project content, source code and test data were managed using the CVS version control system.

7.2. Future Work

Unfortunately the UltraSwarm helicopters were not yet operational at the time of development of the ultrasonic ranging system. It was therefore not possible to test the newly developed ranging

system on a helicopter. The tests carried out on the testing platform showed the system to be effective but in order to really know how well the system performs it must be tested on a functional helicopter. The next step of the project is therefore to connect the system to a helicopter and perform detailed testing and analysis with the helicopter flying in the arena.

An unresolved issue for which a temporary solution has been employed is the problem of the SRF10s intermittently crashing. Unfortunately the temporary solution incurs a penalty of a time delay and lowers the maximum rate that the SRF10s fire. Although the system is still able to fire at a suitably high rate, ideally the cause of this problem will be found so that the time delay can be removed and the SRF10s can be fired at even higher rates.

The initial test results of the RUFACE algorithm showed it to be very effective at eliminating crosstalk, however, due to the scope of the project and time constraints, the testing has been limited at this stage. The flexibility of RUFACE algorithm provides new benefits over existing approaches and it is therefore believed that the algorithm could potentially be of value to the field of ultrasonic ranging. Before this can be said for sure the algorithm must be tested extensively by performing a detailed analysis of the timing sequences it generates under various conditions as well as testing the ability of the algorithm to scale as more ultrasonic devices are added.

References

(1) Borenstein, J. and Koren, Y. *Error Eliminating Rapid Ultrasonic Firing for Mobile Robot Obstacle Avoidance*. University of Michigan. [online]. 1995, [Accessed 8 September 2006]. Available from World Wide Web: <http://www.eecs.umich.edu/~johannb/paper32.pdf>.

(2) Modi, S. *Comparison of three obstacle avoidance methods for an autonomous guided vehicle*. University of Cincinnati. [online]. 2002, [Accessed 9 March 2006]. Available from World Wide Web: <http://reeses.mie.uc.edu/theses/theses2002/sachinthesis_ver2.pdf>.

(3) Kleeman, L. *Real Time Mobile Robot Sonar with Interference Rejection*. Monash University. [online]. [Accessed 10 March 2006]. Available from World Wide Web: <http://www.ecse.monash.edu.au/centres/irrc/LKPubs/sensrev.PDF>.

(4) Heale, A. and Kleeman, L. *A Sonar Sensor with Random Double Pulse Coding*. Monash University. [online]. [Accessed 8 September 2006]. Available from World Wide Web: <http://www.ecse.monash.edu.au/centres/irrc/LKPubs/FinalRandomSonar.PDF>.

(5) Corke, P. Sikka, P. and Roberts, L. *Height Estimation for an Autonomous Helicopter*. CSIRO Manufacturing Science & Technology. [online]. [Accessed 10 March 2006]. Available from World Wide Web: <http://www.ri.cmu.edu/events/iser00/papers/corke.pdf>.

(6) Danko, T. Kellas, A. and Oh, P. *Robotic Rotorcraft and Perch-and-Stare: Sensing Landing Zones and Handling Obscurants*. Drexel University, Philadelphia PA. [online]. [Accessed 8 September 2006]. Available from World Wide Web: <http://prism2.mem.drexel.edu/~paul/papers/dankoIcar2005.pdf>.

(7) De Nardi, R. Holland, O. Woods, J. and Clark, A. *Swarmav: A swarm of miniature aerial vehicles*. Proceedings of the *21st Bristol International UAV Systems Conference*. University of Essex. [online]. April 2006. [Accessed 8 September 2006]. Available from World Wide Web: <http://gridswarms.essex.ac.uk/publications/bristoluav21.pdf>.

(8) Holland, O. Woods, J. De Nardi, R. and Clark, A. *Beyond swarm intelligence: The Ultraswarm*. Proceedings of the *IEEE Swarm Intelligence Symposium (SIS2005), June 2005*. University of Essex. [online]. April 2006. [Accessed 8 September 2006]. Available from World Wide Web: <http://gridswarms.essex.ac.uk/publications/SIS2005copyright.pdf>.

(9) Ohya, A. Ohno, T. and Yuta, S. *Obstacle Detectability of Ultrasonic Ranging System and Sonar Map Understanding*. University of Tsukuba. [online]. [Accessed 8 September 2006]. Available from World Wide Web: <http://www.roboken.esys.tsukuba.ac.jp/~ohya/pdf/RAS1996-ONO.pdf>.

(10) Bank, D. *A Novel Ultrasonic Sensing System for Autonomous Mobile Systems*. Research Institute for Applied Knowledge Processing (FAW). [online]. [Accessed 8 September 2006]. Available from World Wide Web: <http://www.morpha.de/download/publications/FAW-SENSORS2002.pdf>.

(11) Jörg, K. and Berg, M. *First Results in Eliminating Crosstalk & Noise by Applying Pseudo-Random Sequences to Mobile Robot Sonar Sensing*. Kaiserslauten University. IEEE/RSJ Int. Conf on Intelligent Robots and System.[online]. 1996. [Accessed 8 September 2006]. Available from World Wide Web: <http://ag-vp-www.informatik.uni-kl.de/Leute/kwj/Papers/IROS96.pdf>.

(12) SRF10 Ultrasonic range finder Technical Specification. [online]. [Accessed 8 September 2006]. Available from World Wide Web: <http://www.robotstorehk.com/srf10tech.pdf>.

Appendices