Report

Literature Review

Indoor Localization

Indoor localisation technologies are a widely researched topic in recent years, with a focus on developing a method for widescale deployment. With the growing availability of advanced mobile devices and wireless infrastructure in public areas, accurate indoor localisation systems within places such as hospitals, shopping centres, warehousing etc. without the need for bespoke hardware is more feasible than ever. Although global positioning system (GPS) works extremely well for an open-air localization, it does not perform effectively in indoor environments due to the disability of GPS signals to penetrate in-building materials {~3~}. Being able to pinpoint the location of a wireless devices requires higher location resolution for indoor environments than in outdoor applications {~1~}. In this paper we will examine some of the approaches implementing solutions for the indoor localization problem and some of the main issues in implementing such systems. We focus on a solution that uses ultrasonic audio signals to attempt to locate a device with millimetre accuracy. There is no system that can be used for all applications under all environmental conditions. From the point of view of usability and accuracy, it is preferable to use a system that performs localization by using the propagation delay time of electromagnetic waves, based on a principle like that used in GPS {~4~}.

Ultrasonic is a well-known ideal candidate for indoor positioning that relies on the time of flight (TOF) scheme [Fischer et al. 2008]. The key idea is to use an ultrasonic transceiver to emit and detect ultrasonic signals {~9~}. Ultrasound is already commonly used for distance measuring in ultrasound parking assist. By recording the time it takes for an ultrasonic signal to travel between a transmitter and receiver, it is possible to compute the distance given the medium traveling speed. Using to represent the speed of sound together with the time of flight, , the distance can be calculated using The TOF of each vector between one of the anchors and the mobile device is measured through its time of arrival (TOA) by finding the peak of a correlation result {~10~}.  
  
Time difference of arrival (TDOA) works similarly to the system described above, but uses a radio synchronization packet followed by an ultrasonic pulse. The distance between nodes is calculated by the travel time of the ultrasound signal. Within a localised system, this measurement can be performed between a device and at least three beacons and the position of the mobile device can be found using trilateration{~9~}. These systems are made up of a number of fixed nodes or beacons and mobile nodes, where the mobile nodes try to calculate their position based on the known positions of the fixed nodes.

One disadvantage of this system is the considerable amount of fixed position nodes needed, which increases the setup cost {~30~}. Ultrasound can provide high localization precision using this system however, it suffers from the line-of-sight restrictions. Given the system requires the mobile device to communicate with at least 3 fixed nodes to get an accurate measurement, the placement of fixed nodes must account for obstacles like desks and chairs to communicate with the mobile devices. Beacon placement thus becomes challenging for ultrasound-based indoor localization in environments with various obstacles {~28~}. Determining the node positions also requires manual calibration which is time consuming, since each anchor has to be measured individually, and fault-prone, because of inaccurate measurement methods and human error {~22~}.

The angle of arrival (AOA) technique uses an array of receivers to evaluate the incoming reception angle. Calculating the location of the source is done by combining the angles of different receivers {~30~}.   
Relative Received signal strength (RSSI) is a WiFi based indoor localization system which uses the signal strength of radio communication to estimate the distance between devices {~30~}. The main idea is to measure a set of signals signatures, known as fingerprints, based on different locations in the area of interest and build a fingerprint database. The location is then estimated by mapping the measured fingerprints against the database. This approach requires a considerable manual effort to build the fingerprint database and the resulting system is relatively inflexible to changing environments {~32~}. RSSI is not well suited to tracking users in real time, due to the lengthy time taken to calibrate for channel propagation parameters {~12~}.  
{~31~} and {~32~} propose methods of tracking users movements within a space with wearable devices embedded into person or user motions from mobile phones.

Implementing accurate indoor localization faces a number of problems technologically to implement. A good solution should be low cost, scalable, robust and easy to deploy. It should also be able to cope with changing environments. The localisation accuracy depends mainly on four factors: Accuracy of the range measurements, location errors of the anchor nodes and geometric configuration of the system. There also exist trade-offs among the positioning accuracy, computational complexity, cost and power consumption {~20~}. {~1~} finds the deployment cost of these various systems to be one of the largest problems to overcome. Out of 22 solutions compared, the average setting and calibration time is 5 hours for two rooms covering 300 square meters. This may be unrealistic and intrusive when deploying these localization systems in large deployment sites like shopping malls. {~4~} proposes an indoor localization which is realized with as few initial references as possible, based on the idea of iterative multilateration. When such a localization method is used, deterioration of localization accuracy due to no line-of-sight signals and to accumulated errors is a problem.  
  
{~28~} describes many of the issues in deploying an ultrasound based localization system. In general, ultrasonic wave emission is usually directional, which introduces difﬁculties in orienting the transceiver precisely. Only when the listener is inside the transmitter’s beam pattern, can the listener derive distance measurement value from the beacon. One solution to this issue is described in {~26~}. They developed a 2-D isotropic ultrasound transmitter with a beam width of 360° using an array of eight ultrasound transducers placed in a round body. The angle of aperture of one transducer is 45°. By using eight ultrasound transducers, the coverage range can be increased by a factor of 8. {~28~} also determines that more beacons are required grows linearly with the size of the area of interest.

Digital signal processing

In this section, we will examine the fundamentals of digital signal processing which will form the basis of all future work. We will see how signals can be created, transmitted, processed and analysed to allow us to build the localisation system. This section begins by looking at the Fourier transform, which can be used to express signals as functions of their component parts. We will then show how correlation is used to compare signal similarity, which has applications in signal detection. Following on from this we examine how this signal analysis can be used practically in a localisation system to find a device’s location. Following on from this we will examine some modulation techniques which are used in signal transmission, and finally some methods for synchronisation between devices within the system.

The Fourier Transform

It can be shown that signals can be described by a sum of sinusoidal components of various amplitudes, frequencies and phases. As we process these signals we need a way to determine the values of these components that make up the signal automatically and accurately.

The Fourier transform of a function of time f(x) is a complex valued function of frequency, whose absolute value is the magnitude of frequency component present in the signal, while the complex argument represents the phase shift of the component. The transform is referred to as the frequency domain representation, and is represented on a magnitude spectrum.

To show what we mean by a frequency domain representation, let us look at the example below. We have a sine wave with the frequency . The graph on the left shows the time-domain representation of the signal. On the right is the frequency domain representation of the same signal. It has a single sinusoidal component with an amplitude of 2 and a frequency of .

|  |  |
| --- | --- |
|  |  |

If we add more signals to the sinusoid in the time domain, we can see the results in the frequency domain below. We have added 2 additional sine waves to the initial wave giving us the final signal in green. By analysing the frequency domain, we can see the amplitude and frequency of each of the signal components within the Composite signal.

|  |  |
| --- | --- |
|  |  |

The frequency domain is showing us the frequencies where the energy of the signal is contained. The figure below is good visual representation of how the two are interlinked, which will be useful when we show what is happening mathematically.

For signal processing, we use the Discrete Fourier Transform (DFT) which deals with discrete functions of time:

Where , is the signal we are analysing, is the number of samples in the signal that is being analysed. is the sample number, which is the index of the time domain representation of the signal. Similarly, is the index into the sequence of frequency values that will be returned. This is referred to as the bin number.

Using Euler’s formula, we can rewrite the complex exponential in terms of sine and cosine functions, which will make it easier to visualise how this applies to signals. This results in the following:

The signal we are analysing is being multiplied by a set of cosine waveforms, and summed to give the real terms in the complex components in the results, while also being multiplied by a set of sine waveforms to give the imaginary terms. This process of multiplication and summation is known as correlation, which we will look at in detail later. Suffice to say correlation is a measure of similarity between two signals, or a measure of the presence of one signal in another. This is essentially what the DFT is doing.

If we substitute into the formula above, we get the following:

When , , cos(0) = 1 and sin(0) = 0, so the formula becomes:

When , becomes a cosine waveform with 1 cycle over samples and becomes a sine waveform with 1 cycle over samples. These signals are then multiplied with the original signal and the values are summed to give the associated real and imaginary term value of the frequency bin, . When , the signal is multiplied by waveforms with 2 cycles over samples to give the values for , and so on up to .

|  |  |  |
| --- | --- | --- |
| k |  |  |
| 0 |  |  |
| 1 |  |  |
| 2 |  |  |
| … | ... | … |

These waveforms are called the analysis basis functions, and when we multiply these with the signal we are analysing, , we get a set of measurements from 0 to of the signals similarity with the cosine and sine waves, which indicates the presence of the cosine and sine waves within the signal .

As an example, we will use a 2Hz cosine signal , and set our sample rate, .

When we multiply this with the analysis basis functions above we get the following results:

|  |  |  |
| --- | --- | --- |
| k |  |  |
| 0 |  |  |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| … | ... | … |

Then, summing the values of each of the samples, we get the real and imaginary terms for the results of each of the frequency bins, which can be plotted in the frequency domain.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | … |
|  | 0+i0 | 0+i0 | **8+i0** | 0+i0 | … |

What is interesting about these results, is the only non-zero bin value lies in the bin where the basis function contains the same number of complete cycles as the signal being analysed. Another way of looking at is the value will be non-zero if the signal contains that base waveform.

We can apply the same process to signals comprised of multiple sinewaves and expect similar results. Using the signal and again set our sample rate, , we get the following results:

|  |  |
| --- | --- |
|  |  |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | |  | 0 | 1 | 2 | 3 | … | |  | 0+i0 | 4+i0 | **16+i0** | 0+i0 | … | | |

Once again, we see the non-zero values at the components containing the base waveforms. The values of the magnitudes in the frequency domain appear to be much larger than in the time domain, however this is simply a result of the sampling frequency we have chosen. If you wish to scale the frequency domain amplitudes, simply divide by .

We will now introduce a phase shift into the signal being analysed to see how this affects the transform. Using the signal and again set our sample rate, . Once again, we run it through our analysis basis functions, but this time we start to notice values appearing in the imaginary terms.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | … |
|  | 0+i0 | 0+i0 | **6.23-i10.256** | 0+i0 | … |

Now, let’s try to verify these values. First, we need to scale the values by , giving us a value ~1.5575+i2.564. A cosine wave with a phase shift can be written in terms of a cosine wave plus a sine wave, . Knowing this, the term can be rewritten as :

And this is what the DFT is telling us. The signal contains both a cosine component represented by the real value of the frequency bin value, and a sine component, represented by the imaginary term. The magnitude of this complex number, when scaled to account for the sample period, is equal to the amplitude of the original waveform, and the angle of this complex number in radians is the phase shift.

So far, we have looked at cases where the signals we are analysing have had an integer multiple number of cycles within the sampling period. However, things get a bit trickier when this is not the case.

Let’s look at a signal with a frequency of 2.4Hz, . When we run this through the transform with the sample period of 8 we get the results below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | … |
|  | 5.59 + i0 | 5.589 + i3.147 | 9.045 + i14.635 | 9.045 – i8.033 | … |
| Magnitude | 5.59 | 6.414 | 17.205 | 12.097 |  |

What we can see is the spectral energy is spread throughout the bins. This is known as spectral leakage. If we plot this in the frequency domain, we can see that a significant amount of the spectral energy is contained between bins 2 and 3.

The shape of this spectral leakage is well defined, and we will explore it later. Remember that the DFT is used to determine similarity between signals, but our signal does not exactly match the frequency of any of the analysis basis functions as the number of cycles is not an integer. However, there is a lot of similarity with the functions used for k = 2 and k = 3, as the frequency is between these two values. Because our signal doesn’t not have an integer number of complete cycles within the sample period, it is impossible to contain the spectral energy in a single bin, so we will always see spectral leakage in some form.

There are a number of techniques to make the results of spectral spreading more predictable and manageable. The first we will look at is known as zero-padding. As we have seen, signal being analysed is correlated against a set of analysis basis functions to produce a set of frequency bin values. The frequencies of these basis functions contain integer multiples of cycles that fit exactly into the sampling window. If the number of cycles in the of one of the sinusoidal components of the waveform matches exactly the number of cycles in one of these functions, the spectral energy with be contained in that one bin value. However, when the signal has a frequency that does not have an integer number of cycles in the sample period, there is no exact match and we get a spread of energy across the frequency bins, as we have seen above. Having more samples will increase the resolution of the frequency domain giving the DFT more opportunities of finding a signal which is an exact match for the signal frequency being analysed and pinpoint where the spectral energy lies. With zero-padding, the number of samples being analysed can be increased by appending a large number of zero valued samples to the end of the signal.

Let’s take the following example where we have a signal of 2Hz, with 1000 samples. We can see in the frequency domain there is a single spike, at bin number 2 and all other values are zero. We can also see that the magnitude of this bin is 250, which is the amplitude of the signal multiplied by half the number of samples. Now when we take the same signal, but increase the frequency to 2.5Hz, we see a spread of energy in the frequency domain, with most of the energy between bins 2 and 3, as should be expected, as the frequency of the signal lies between 2 and 3 cycles. We also notice that the magnitude of the maximum is much lower than it was then we matched the frequency exactly.

We now will zero-pad this signal with 1000 samples with 0 values and analyse the new signal. In the time domain, we can see our signal now has 2000 samples, with the original signal of 2.5 cycles up to sample 1000, and 1000 zero values. In our frequency domain plot, we now see a large, bell shape, known as the main lobe. Its maximum value is centred at bin 5, which corresponds to the 2.5Hz frequency of our signal. If the signal had multiple sinusoidal components, there would be a main lobe for each component. What we also see is the magnitude in back up to 250, as it was in the example with 2Hz over 1000 samples. However, we also notice a number of side lobes, which have been introduced as a result of the zero-padding. These can be problematic in accurately analysing the frequency spectrum data. We can reduce the size of these side lobes by using a process known as windowing.

In the example above, we zero padded by the length of the original signal. This is the minimum amount of zero-padding allowed. It worked in the example above as 2.5Hz over the period of 1000 samples did not have an integer value of cycles, but when we doubled the length of the sample period, there would be 5 full cycles of the waveform, so this matched the basis function associated with bin 5. If we use a value of 2.2Hz, zero-padding by 1000 samples would result in the broad spectral spread of energy we saw before. The solution in this case would be to zero-pad by a factor of 5 to allow the DFT find a waveform with the same number of cycles over the sample period as the signal. In practice, we do not know the values of the frequencies we are looking for, so the approach is to zero pad by as large a number as possible. This gives the DFT the greatest chance of finding the waveform that matches exactly with the frequency of the signal being analysed.

Windowing is applied to a signal to improve the behaviour in the frequency domain.  
it is achieved by multiplying the signal to be analysed by a window function.   
Zero-padding is itself a form of windowing, where the window being applied is a rectangular window, which has a value of 1 for the first N samples, and zero for values less than 0 and greater than N. Our wave form is a continuous signal with the frequency 2.5Hz. This can be represented on a magnitude spectrum as a spike at the bin positions representing 2.5 Hz. When zero-padding, we multiply our wave by the rectangular window described above. This rectangular window has a frequency spectrum associated with it, with both positive and negative frequencies. When we multiply in the time domain, the effect in the frequency domain is a process called convolution. When we convolve the waveform associated with the window waveform, the result is this spectral shape associated with the rectangular window appearing everywhere there is a spike associated with the original frequency. This is introducing a large amount of high frequency components in the side lobes. If we had two sinusoids of similar frequencies in our signal close together, we can see the interference of spectral noise can lead to data that can be easily misinterpreted.

The ideal form of DFT would be an infinitely long sample period. In practical terms, we are restricted only by computing power, so we append a very large number of zeros. Going back to our 2.2Hz example, we will append 99000 zeroes, which for our purposes will act as the “infinite” resolution DFT response. Looking the frequency domain, we see that main lobe, with the peak at the bin associated with the 2.2Hz, and number of side lobes. If we plot the magnitude spectrum for the same signal padded with 9000 zeroes. We see that the samples align exactly with those from the “infinite” response. Doing this again with 4000, and 0 zeros appended, we see that all the values in the frequency bins are all aligned to the “infinite” curve. The shape of the response from the DFT with N samples will be a sampled subset of the infinite response.

What we try to achieve with windowing, is to use a function that has a spectral shape that is more manageable for analysis purposes. If we consider all There are many well defined window functions which can give us predictable responses in the frequency domain. Let’s take the Hanning window for an example.  
eoafuenaoufnaef BLEEDIN GRAPHS OOOOOOOOOO

The Hanning window can be described by the following equation:

It looks like this in the time domain. As we see, it has zero values at the edges and a peak in the centre. The magnitude spectrum of this window has a wide main lobe, and small side lobes. So, what we expect to see when we apply this window to the signal, is this shape to appear at each frequency component in the original signal.

With windowing, we set the window to the same sample width as the signal we are analysing, and then multiply the signal

The resultant shape in the frequency domain is the bell-like shape we saw while zero-padding, but the side lobes have been greatly reduced. We also notice the main lobe is twice the width of the main lobe we saw with zero padding. We can now determine the amplitude, frequency and phase information of the sinusoid by analysing the magnitude spectrum. First, we need to find the bin with the maximum value. In this case, it is \*\*\*BINNUMBER\* and the magnitude is 125, which is half again of what it was when we were able to match a signal without the windowing. When using the Hanning Window specifically, we can divide the magnitude value of the maximum value by the length of the signal divided by 4. This depends on the window function being used. The angle of the maximum gives us exactly the phase of the sinusoid. To get the frequency, multiply the bin number by the sampling frequency and divide by the number of samples.

We have seen that zero padding and windowing are useful in more accurately determining where spectral energy lies, but they come with overhead and a trade-off must be found that maximises efficiency of processing, while also maximising accuracy.

Correlation

Correlation can be used to the measure of how similar two signals, x and y, are to each other. Mathematically, it can be defined as follows:

When dealing with signals that are zero valued up to sample n=0, and zero values for all samples greater than N-1, we can rewrite the equation as:

We will run through a basic example to illustrate the process of correlating signals.  
Take the signals x, y and z below, with their respective plots:

|  |  |
| --- | --- |
| x = [ 2 1 -1 3 ]  y = [ 3 2 0 3 ]  z = [ -1 3 2 1 ] |  |

We then use the equation above to calculate the correlations:  
Corr­­­­­x,y = x[0]y[0] + x[1]y[1] + x[2]y[2] + x[3]y[3] = (2)(3) + (1)(2) + (-1)(0) + (3)(3) = 6 + 2 + 0 + 9 = 17  
Similarly, we can find Corr­­­­­x,z = 2 and Corr­­­­­y,z = 6

The results can be interpreted as the larger the result, the more similar signals are. As can be seen from the graph, the signal x is more like y than it is z. Correlation is a method to tell how similar signals are automatically, without needing to plot.   
However, this basic form of correlation runs into issues when the signals we are interpreting have different energy levels.

Let’s see what happens if we change one of the values in z to be much larger:

|  |  |
| --- | --- |
| x = [ 2, 1, -1, 3 ]  y = [ 3, 2, 0, 3 ]  z = [ -1, 30, 2, 1 ] |  |

We can see that the signal z is now much less similar than the others, but the correlation yields the following results:  
Corr­­­­­x,y = 17, Corr­­­­­x,z = 29 and Corr­­­­­y,z = 60  
This is not what is expected of a correlation result. A larger value should imply more similarity. For more accurate results we need a way to scale the correlation function by a factor related to the energy levels we are looking at. Thus, we can use the normalised correlation function:

The numerator of the expression above is the correlation function as above, however the lower determines the energy in the signals and scales the result. Taking the second example above we can determine the following:

These results are in keeping with what you would expect from correlation, where a higher value implies a more similar signal. Normalised correlation will always return values from -1 to 1.   
You would get a value of 1 if you compared a signal with itself, and a value of -1 if you compared a signal with and inverted version of itself, i.e. x[n]\*-1

There are practical applications to using both approaches, and it depends on what is trying to be achieved.

Cross Correlation   
Now that we have an understanding of what correlation is, we can look at practical uses within signal processing. Cross correlation is the measurement of the similarities of two signals at different lag positions. It is defined for discrete functions f and g as:

where denotes the complex conjugate of . It is much easier to consider it in practical terms. Let us look at the cross correlation of the two signals below, x and y.

|  |  |
| --- | --- |
| x = [ 2, 1, -1, 3 ]  y = [ 2, 0, 3, 1 ] |  |

We show signals x and y at lag position 0. That is to say sample 0 of the signal x, x[0], is vertically aligned with sample 0 of the signal y, y[0]. To calculate the correlation when there is no time lag, we use the same formula from earlier. We can say the correlation measure at a lag of 0 is, (2)(2) + (1)(0) + (-1)(3) + (3)(1) = 4.

We can now shift the sequence y one sample to the right, we see the signals at a lag position 1, y[0] is now aligned to x[1], and so on. We now take the correlation of the vertically aligned samples to find the correlation measure at lag position 1: (1)(2) + (-1)(0) + (3)(3) = 11

|  |
| --- |
|  |

We continue to shift to the right until we have calculated all of the overlapping lag positions, as we complete the correlation sequence:

|  |  |
| --- | --- |
|  |  |

Just as we shift to the right to find the positive lag, we should also shift to the left to find the similarity measure at the negative lag values, to build up the full correlation sequence:

|  |  |  |
| --- | --- | --- |
|  |  |  |

Now we have the correlation measurements for these signals at all lag positions, giving us the following results:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Lag Position | -3 | -2 | -1 | 0 | 1 | 2 | 3 |
| Correlation measure | 2 | 7 | 2 | 4 | 11 | -2 | 6 |

Remember that correlation is a measurement of similarity, so what these results tell us is that signals x and y were most similar at lag position 1.

This result can be used to determine a time delay between two signals which can be used to measure the distance between points.

What is Autocorrelation?   
Autocorrelation is simply the cross correlation of a signal with itself. When dealing with one dimensional, real sequences, autocorrelation will have a peak at a lag of zero, and its size will be the signal energy.

Taking the example of the signal x above, cross correlating the signal with itself yields the following correlation sequence:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Lag Position | -3 | -2 | -1 | 0 | 1 | 2 | 3 |
| Correlation measure | 6 | 1 | -2 | 15 | -2 | 1 | 6 |

Autocorrelation has some interesting properties for use in digital signal processing. The autocorrelation of a periodic function is itself periodic, with the same period. This can be used to determine frequencies or pitches of musical tones. In a noisy waveform, autocorrelation can be used to reduce the effects of that noise.

Measuring distances using audio

REF [1] Time-of-Flight Measurement Techniques for Airborne Ultrasonic Ranging for reference  
[2] Fast processing techniques for accurate ultrasonic range measurements

Measuring distance using ultrasonic signals offers an inexpensive solution to the indoor localisation problem. Measurements of the distance of an object from fixed landmarks can be used to calculate the precise location of that object. The basic idea is to send an acoustic signal through the air from a transmitter to a receiver. The time it takes for the signal to reach the receiver is known as the time of flight (TOF), which can be used to measure the distance by the following equation:

Where is the distance, is the time of flight and is the speed of sound in air. Many ultrasonic applications use a single transducer, which both emits and receives, to calculate the distance to an object by emitting a signal and measuring the time it takes for an echo to be received. The distance is calculated similarly to the calculation above, but having the result, as the signal has travelled the distance to the object and back; . For this comparison, we will consider the measurements are using the second technique, which allows us to know when the signal was sent, although in more complex ranging systems that use independent transmitters and receivers, synchronisation becomes an important factor, which we will examine later.

The accuracy of the distance measurement depends on the accuracy of the measurements of the TOF and the speed of sound in air. Choosing the measurement technique of the TOF depends on a number of factors such as cost, ease of implementation, environment. We will examine some common methods used for these TOF measurements. Methods are usually compared by the accuracy, or error in the distance measurement, repeatability, or the variance in repeated measurements, cost of implementation and performance under noise.

Time Domain methods with single frequency signals

The first method we look at is thresholding. A signal is sent, and the time of flight is the time it takes for the amplitude of the echo signal to surpass a certain threshold. The threshold is usually selected to be well above the noise standard deviation. This method is relatively straightforward and does not require complex circuitry or calculations, and can be implemented with inexpensive transducers. Its limitation is it naturally introduces a bias into the measurement. If the threshold level is set to low, noise interference can cause false positive to be detected. Increasing the threshold will improve detection of real echoes, however the time it takes for the incoming signal amplitude to surpass the threshold will be delayed further. This is especially true if too low a sampling frequency is chosen. Although noise and sampling frequency selection are issues that all TOF measurements must deal with.

Curve fitting is a method of TOF estimation that attempts to fit a parabolic curve to the echo signal envelopes leading edge to provide a measurement without bias. This uses threshold estimate as above, but then fits a parabolic curve in the form where is the estimation of the result of simple thresholding and is estimated from a second derivative approximation around this threshold point. A nonlinear least-squares method is then applied to fit the curve, and the vertex of the parabola is used as the measurement of the TOF.

Sliding window is a method that can be used to make detection more robust. A window of length N is slid through the echo signal one sample at a time. As the window slides through the sample, it counts the number of samples which exceed the threshold value. If this count exceeds second threshold the signal is considered present and the TOF estimate is produced.

Cross-correlation is an unbiased measure of TOF. The echo signal is correlated with a matched filter that contains the waveform and the delay will be the peak. This method has a few drawbacks in comparison with those outlined above. In real-time processing, the entire echo must be observed before the correlation process can being which can add a significant delay to producing the estimate. It is also computationally much more complex than the previous methods. However, this method should significantly reduce noise interference.

A study performed by [2] shows that Correlation gives by far the best results in terms accuracy, which is what would be expected, however the less complex methods offer some acceptable performance results at much lower cost. Curve fitting performing best when it came to bias and total error, whereas sliding window performed best with standard deviation.

Other methods of measurement

Time difference of arrival (TDOA) can be used as part of a multilateration system to predict the location of an object. A transmitter sends a signal which is received at receiver stations 1 and 2, which are at known locations. When both receivers have received the signal, the can cross-correlate them to determine the time shift between the two waves which is the difference in time it took the signal to at each station. This time shift can be used in equation 1 to get a measure of distance. As both stations are fixed, we now have an infinite number of points along a curve that satisfy the transmitters location. If we had a second pair of stations, we would get a second curve of possible locations that intersects the first. This produces a small number of locations that the transmitter could be.   
The ultrasonic waves from a small mobile device with an ultrasonic transmitter are received by the receiver array. The ultrasonic receiver array and the mobile device are synchronized by a wireless connection. The listener can then derive the distance from the beacon by multiplying the ultrasound propagation time by the speed of sound. Since the location of the receiver is known when the mobile device was deployed, the listener must be located at the surface of a sphere that is centred at the beacon and with a radius of the derived distance from the beacon to the listener {~28~}. By multiplying the measured propagation delay time by the speed of sound, the distance between the mobile device and each of the ultrasonic receivers is derived. Since the locations of the individual receivers are accurately given beforehand, the location of the mobile device can be derived three-dimensionally by solving a set of simultaneous equations involving the measured distance to each receiver and the locations of the receivers. {~4~} However, this does require the system to be synchronised for accurate measurements between devices.

Another approach using TDOA is described in {~9~}. Distance is measured using the difference in time-of-ﬂight of RF signals and US signals. A RF and US signal are sent simultaneously from a transmitter to a receiver. The RF signal travels much faster than the US signal. Over a distance of 10m, the radio signal takes ~30ns and the ultrasound signal, will take ~30ms. As the RF signal is much lower, the TDOA can be measured as the difference in time from when the RF signal arrives to when the US signal arrives.   
Unfortunately, the speed of sound is not constant. Indoors it varies mainly with temperature and can be approximated using where T is the temperature in degrees Centigrade.

The multilateration approach is popular in indoor localisation systems, however its accuracy is dependent on many factors such as the devices in the system being synchronised, interference from noise or multi-path interference, inaccuracies in the measurement of the locations of the fixed devices.

Angle of Arrival (AoA) is used to determine the direction a received signal arrives at an array. It is measures by taking the TDOA between elements within the array. For example, let’s consider an array of two microphones separated by half the wavelength of an incoming sinusoidal wave. If the wave was emitted from directly in front of the array, there would be no phase difference between the two measured waves. However, if the wave was emitted from the right of the array, the microphones would receive the signals half a wavelength apart, resulting in a phase difference on 180. If this was used alongside the multilateration system we described earlier, we could immediately disregard many of the values of the curve produced.

Synchronisation

The earlier discussions on measuring time delays when calculating distance assumed that the signal was sent from a single transceiver, that both sends and receives the signal on a single circuit. In the localisation solutions, it is more likely the transmitters and receivers are distributed. This presents a new issue. For accurate measurements of TOF, TDOA or any other technique, we need to be confident that each module in the distributed system is synchronised to within a reasonable degree of accuracy.

In a distributed system, each node has its own physical clock. These clocks are based on crystal oscillation counters which generate many interrupts per second. The clock in the node will tick on each timer interrupt.

The issue is that two devices will hardly ever agree, as the clocks will oscillate at slightly different frequencies. If we take UTC time, , to be the perfect clock, the physical clock on nodes, , usually run faster or slower. This is known as drift, and can be measured using the . Ideally, which means the clock is perfectly in sync with UTC time. If , the clock is fast, and , the clock is slow. When we read two disagreeing clocks at the same time, the difference in values is known as the skew. The aim of synchronisation is to keep the skew between clocks bound to within an acceptable constant of drift, , such that . As we are attempting to use the time difference in signals to measure distance, even a skew of 1ms could lead to errors in measurement of around 0.35m. To ensure the time difference between any two clocks in the system never exceeds a maximum value of , The clocks need to be synchronised every seconds.

There are several methods of achieving system wide synchronisation.

External synchronisation requires each node to synchronise its clock with an authoritative external source. The MSF is a 60kHz radio signal which is dedicated to broadcasting the current UTC which can be decoded by radio-controlled clocks for synchronisation. Similarly, GPS receivers can be used to synchronise with UTC. These solutions, do provide accurate measurements, but are not a practical solution for simple distributed systems.

A much simpler solution is to have a dedicated time server within the system, which all other devices can synchronise with.  
Each node can ask the time server for an accurate time periodically, and adjust its clock accordingly. This requires an accurate measure of the round-trip delay for the node to receive the updated time.

Cristian’s algorithm is a simple implementation of this method. A client node sends a request to the timing server for the current time at T­0. The message is received at the server after a delay for transmission across the network. The server processes the request and sends the current time, t, back to the client, who receives the time at T1. Knowing nothing else about the network, the client will set its clock to the estimated value of . Provided the network delays are small, this is a reasonably accurate measurement. If we know the minimum delay in sending messages across the network, , we can improve the accuracy of the result by limiting the range for the answer to . This is a simple approach to the problem, but has a major drawback if the time server fails, the entire system will fail. It also relies on the time server having a UTC synchronized clock.

An alternative approach is to let the time server be the master, and periodically poll all nodes in the system for their local time. Taking an average time across the system, tell each slave node by how much they need to adjust their clock. This is a completely self contained system that uses relative time adjustments from a single point, so there is no need for external synchronisation.

The Berkeley algorithm is an implementation of this idea. The master polls each node and estimates the delay for each node. Once all nodes have responded, the master will take the average time of all clocks in the system, including its own, and calculate the average. This accounts for individual clocks tendencies to run fast or slow. It then sends the offset for each local clock to adjust. The algorithm will exclude local times whose skew is too great, to compute a more tolerant average. As this is completely contained within the system, if the master fails for any reason, any other node can take over. The selection of the new master node is done by a process called election. In a system where all distributed processes have knowledge of every other node, but not whether the node is active or not, the process with the largest process id should become the new coordinator.

When a coordinator fails, the process whose timer expires first will broadcast to all other devices an election message. If it has the highest process id it will declare itself the new coordinator, or else send an election message to all higher processes. If it receives a response from a higher ID, it waits to receive a coordinator message from another device.

If it receives a message from a lower process, it will reply and restart the election process {CSD-86-275}

Ring election algorithm. When a process, P, notices the coordinator is down, it passes the Election message with its own id as part of a list. The next process passes this message to the next process in the ring, with its own id appended to the list and so on until P receives the Election message with its own id in the list of processes. It then selects the largest process as the new coordinated and sends a new Coordinator message with the new coordinators id and the list of active processes.

Network Time Protocol (NTP) enables clients across the internet to be accurately synchronised to UTC. It is a client-server architecture based on UDP message passing. It can provide reliable service even with lengthy losses of connectivity and allows clients to offset typical drift rates by synchronising frequently. It provides a reliable service by having redundant paths and servers and provides protection against interference.

It arranges the network into groups of sub-networks, known as strata. Nodes on the first strata are connected directly to the synchronisation source. The second strata contains nodes connected to the first strata and so on.

There are several NTP modes. Multicast uses one node to periodically send time information to all other nodes on the network. The nodes assume a small transmission delay and adjust their clock. This method is lower in accuracy and only suitable to high speed LANs. Procedure-call mode is similar to Cristian’s algorithm. A service will accept requests from clients. This yields more accurate results than multicast. Symmetric mode is used, generally by servers in the same strata in the system, to improve synchronisation over time. A node may be connected to a number of higher strata servers as well as peers on the same stratum.

In procedure-call and symmetric modes, messages are exchanged in pairs between servers, denoted by m and m’ for the sent and received messages respectively. For these message pairs, NTP calculates the offset estimate between the two clocks, the delay time between the messages, and the filter dispersion, which is an estimate of the quality of the results based on the accuracy of the server’s clock and consistency of the network transit time. This is calculated using the algorithms based on the most recent measurements of offset and delay between the client and server. Time servers will communicate with multiple peers and eliminates peers with unreliable data.

NTP messages contain a lot of information about the nodes, including clock precision, stratum, reference timestamps for when the clock was last set and poll intervals. This information can all be used as part of validating messages.

Getting the offset estimate in NTP is similar to what was used in Cristian’s algorithm. The NTP message m is timestamped at , when it is sent from the client. It is then timestamped again at , when it is received at the server. The server will then send the response m’ which is timestamped at times and by the server and client when it is sent and received, respectively. The calculation of offset is defined as , and the round-trip delay by . These values are collected and statistically analysed to rule out outliers and the clock frequency is adjusted gradually according to the best estimate of the offset.

SNTP is a simplified version, or rather, a subset of NTP. It operates in procedure-call or multicast mode, but does not use the statistical methods in NTP to adjust the clock gradually. It is used in systems where the root node is the server and clients are leaf nodes (stratum 1).

Synchronisation Signals

[1] Analysis of the Frequency Offset Effect on Zadoff-Chu Sequence Timing Performance

[2] Zadoff-Chu Coded Ultrasonic Signal for Accurate Range Estimation

So far we have looked at generic sinusoidal waves for transmission across the network. When these signals are cross-correlated with an expected waveform, we can see a peak at the lag position where the two signals overlap, giving us a measure for the delay time. What we can also see is a wide sideband of peaks which get larger the closer to the true peak. If we were to take the measurement for the lag be the maximum value of the waveform, we can see how a noisy signal could lead to incorrectly choosing the maximum.

There are signals with autocorrelation properties which mitigate against this. One of the properties of Additive white Gaussian noise (AWGN) is that the auto-correlation values for any non-zero delay, i.e. where the signals do not align perfectly, is effectively zero. It also has zero cross-correlation with any other AWGN waveform. This is a very useful property in peak detection. Pseudo-random noise (PRN) sequences also display similar autocorrelation properties. At zero time-delay there is a peak, and at non-zero time-delays the values are very small. These PRN signals also carry timing information as well, which is extremely useful in distributed systems to coordinating device transmissions.

In wireless transmission, a PRN sequence is used in setting up connections to detect and synchronise devices. A wireless access point will transmit a synchronisation signal. A matched filter in devices check incoming frames for this PRN sequence and aligns its local clock to the access point’s and sends back a signal, which the access point then scans for the PRN sequence, detecting the devices timing and instructs the device to adjust its transmit timing to account for round trip propagation.

So far, we have determined that a sequence with good autocorrelation properties are useful in time-of-arrival detection. However, another useful property of these sequences is having zero or very low cross-correlation with the same signal at any delay. A complex PRN sequence has a periodic autocorrelation of where N is the period of the PN sequence. Therefore, cyclically shifted PN sequences have a correlation with the original sequence.

Gold Code - A Gold code, also known as Gold sequence, is a type of binary sequence, used in telecommunication (CDMA)[1] and satellite navigation (GPS).[2] Gold codes are named after Robert Gold.[3][4] Gold codes have bounded small cross-correlations within a set, which is useful when multiple devices are broadcasting in the same frequency range. A set of Gold code sequences consists of 2n − 1 sequences each one with a period of 2n − 1. Gold codes are used in GPS. The GPS C/A ranging codes are Gold code of period 1,023.

A Zadoff-Chu Sequence is a complex-valued sequence with some very useful properties in signal transmission. It is given by the equation

Where is the length of the sequence.

When is odd, the sequence is periodic  
2. If is prime, the Discrete Fourier Transform of a Zadoff–Chu sequence is another Zadoff–Chu sequence conjugated, scaled and time scaled.

{\displaystyle X\_{u}[k]=x\_{u}^{\*}({\tilde {u}}k)X\_{u}[0]} X\_{{u}}[k]=x\_{{u}}^{{\*}}({\tilde {u}}k)X\_{{u}}[0] where {\displaystyle {\tilde {u}}} {\tilde {u}} is the multiplicative inverse of u modulo {\displaystyle N\_{\text{ZC}}} N\_{{\text{ZC}}}.

3. The auto correlation of a Zadoff–Chu sequence with a cyclically shifted version of itself is zero, i.e., it is non-zero only at one instant which corresponds to the cyclic shift.

4. The cross-correlation between two prime length Zadoff–Chu sequences, i.e. different values of {\displaystyle u,u=u\_{1},u=u\_{2}} u,u=u\_{1},u=u\_{2}, is constant {\displaystyle {\sqrt {N\_{\text{ZC}}}}} {\sqrt {N\_{{\text{ZC}}}}}, provided that {\displaystyle u\_{1}-u\_{2}} u\_{1}-u\_{2} is relatively prime to . [2]

Zadoff-Chu is used in 3gPP LTE services for both synchronisation and random access preambles   
Zadoff–Chu sequences are an improvement over the Walsh–Hadamard codes used in UMTS because they result in a constant-amplitude output signal, reducing the cost and complexity of the radio's power amplifier.[3]

Supposing we had a single nodes whose location is unknown is communicating with multiple fixed point nodes whose locations are known.

The orthogonal nature of the ZC signals means that multiple cyclically shifted signals can be combined and sent simultaneously in a single transmission. If each receiver had a matched filter to look for the signal with a particular phase shift, the other signals in the transmission would not be detected. This can also

[2]

*Zadoff-Chu sequence is a polyphase complex valued sequence, named after Solomon A. Zadoff and D.C. Chu [15],[16]. Zadoff-Chu sequences are constant amplitude zero auto-correlation (CAZAC) sequences, where the autocorrelationiszeroforanynonzerolag.AZadoff-Chusequence sk of length N can be written in the following form [16]: sk = ei↵k,k =0 ,1,2,...,N1, (1) where ↵k is given by: ↵k =(M⇡k2 N , if N is even M⇡k(k+1) N , if N is odd (2) where M and N are integers and M is coprime to N. The received signal is modeled as: y[k]=↵x[kd]+w[k], (3) where ↵ is an attenuation factor, x[k] is the transmitted signal, d is the delay between transmitting and receiving the signal*

*and w[k] is an additive white Gaussian noise (AWGN). If the transmitted signal is an odd-length Zadoff-Chu sequence, y[k] can be written as: y[k]=↵ei⇡ M N (kd)(k1d) + w[k],k=1 ,2,...,N (4) The cross-correlation of x with y is given by:*

*(x⇤y)[n] ,*

*N1 X m=0*

*x⇤[m]y[m + n] (5)*

*The cross correlation function r[n] is given by:*

*r[n]= N1 X k=0*

*x⇤[k]y[k + n]*

*=*

*N1 X k=0*

*ei⇡ M N k(k+1)⇣↵ei⇡ M N (k+nd)(k+1+nd) + w[k + n]⌘ Considering the noiseless case and with some manipulations we can reduce the summation to:*

*r[n]=↵ei⇡ M N (nd)(n+1d)*

*N1 X k=0*

*ei⇡ M N 2k(nd) (6)*

*Taking the absolute value of the cross-correlation function, we have: |r[n]|=|↵| N1 X k=0 ei⇡ M N 2k(nd) = |↵| ei2⇡M(nd) 1 ei2⇡ M N (nd) 1 = (0 if n 6= d |↵|N if n = d note that M and N are coprime. The peak of the cross correlation is at the TOF (n = d). The same derivations can be implemented to even-length Zadoff-Chu sequences. Since the signal bandwidth BW = 2 Tsym, where Tsym is symbol duration, the minimum symbol duration depends on the available bandwidth. The hardware used in this work has a total bandwidth of 7 kHz, limiting the symbol duration not to be less than 0.286 ms. The sequence in this work has been chosen to have a length of 15 symbols (N = 15) and a symbol duration of 0.3125 ms, resulting a sequence duration of 4.688 ms. For continuous transmission, this sequence is repeated P times, where each single sequence is referred to as a block.*

*Assuming multi-paths reception, the received signal in (3) can be written as*

*y[k]=*

*L X i=1*

*↵ix[kdi]+w[k]. (7) A cross-correlation between a reference block and a window of the received signal is applied to estimate the TOF of the ﬁrst block. The peak associated with the direct path is not necessarily the highest peak, multi-paths can add up constructively and result in a higher peak. However, the direct path peak is always the ﬁrst signiﬁcant peak to arrive. Cross correlating the reference transmitted block with the received blocks results in peaks at d1, d2, ..., dL, with d1 <d i, i =2 ,3,...,L. In order to ﬁnd the peak corresponding to the TOF of the ﬁrst block, an early peak search is applied where the highest peak is located, then a search for all the earliest peak with amplitude greater than a threshold of 0.6 of the highest peak is performed. This threshold was tested experimentally and proved to be high enough to avoid noise, and low enough for direct path detection. Once the TOF, d1, of the direct path has been estimated, the distance to the target is given by D = d1 vs fs , (8) where vs is the speed of sound in air. Figure 2 illustrates this step.*

Modulation.

A key aspect of wireless communication is modulation. This is where a carrier signal which is going to be sent from the transmitter across the network is modified in some way to be able to send information in a more efficient manner. We will explain the basic concept of modulation, as well as looking at some methods used in digital communication to transmit data. The most well-known forms of modulation are frequency modulation (FM) and amplitude modulation (AM) which are used in radio broadcast. These work by taking a baseband signal, in the case of radio broadcast this will be audio signals, and varying the carrier signal’s frequency or amplitude proportionately to the baseband signal.

Let’s take a baseband audio signal , which is to be transmitted using a sinusoidal carrier signal , which has a frequency and an amplitude . If we assume to be a continuous sinusoid which has a frequency and an amplitude , which can be assumed to be limited to the range ±1. The result of combining these two waves in frequency modulation is given by   
where . represents the frequency deviation from the carrier signal frequency and is the sensitivity of the frequency modulator. This can be written in the general form

In this equation is the instantaneous frequency.

What this results in is a signal which varies its frequency around a carrier frequency , which results in a waveform like below. Notice as the values of the baseband signal rise, the frequency of the modulated signal increases. The result of this in the frequency domain is a peak at the carrier frequency with sidebands roughly contained to the deviation frequency.

Digital data can also be sent in this manner using a process called amplitude-shift keying (ASK) or frequency-shift keying (FSK). As the information we are sending is usually a binary stream of data, we can assign values of amplitude or frequency to the symbols 1 and 0. The simplest form of ASK is on-off keying (OOK), where a 1 is represented by the presence of a signal for a determined length of time, and a 0 by no signal over that same time. Binary FSK (BFSK) works by transmitting binary information using a pair of discrete frequencies to transmit data where a 1 or zero is represented by frequencies of the carrier frequency plus and minus some frequency deviation. This can be achieved by using a single oscillator which means the phase of the generated sinusoids are continuous.

Gaussian FSK (GFSK) uses a Gaussian filter to smoother the transitions of the base data, which reduces sideband power, and therefore interference with neighbouring channels. This method’s smoother transitions increase the probability of intersymbol interference. GFSK is used in many areas of wireless communication, including Bluetooth.

Minimum-shift keying (MSK) is a form the FSK that uses a frequency deviation of 0.25 of the carrier frequency. This results in the high and low waveforms differing by half the carrier period. This has a particularly efficient spectral response compared to other forms of FSK. Gaussian MSK (GMSK) is similar to standard MSK, but again applies a Gaussian filter to the data stream before applying the frequency modulation. The deduces both sideband power, and results in narrower phase shift angles, but it requires higher power to reliably transmit data, compared to other methods. GMSK is used in GSM phone standard as well as the maritime navigation system AIS.

Audio FSK (AFSK) is a modulation technique which represents binary data by changes in pitch of audio data on the baseband signal and then modulated using conventional methods like FM for transmission.

Another method of modulation is to use phase-shift keying (PSK), which represents symbols using differences in phase. As with most modulation techniques, PSK required a demodulator designed specifically for the symbol set being used. Binary PSK (BPSK) uses 2 phases, separated by 180° representing binary 1 and 0.

Quadrature PSK (QPSK) represents 4 binary symbols using waveforms separated by 90° in phase. This can achieve the same data rate as BPSK but halves the bandwidth and maintains the same bit error rate (BER). However, this method uses twice the power and the implementation requires more complex transmitters and receivers.

Offset QPSK is a variant on QPSK, but processes the odd and even bits separately, so that the phase never changes by more than 90° at a time, which reduces amplitude fluctuations.

ACTUALL MOTHERFUCKING WOOOOORK

RPi Experiments:

Configuration:  
Brief introduction to raspberry Pi. Its current range of models.  
More detailed look at RPi3  
Soundcard we have chosen and other hardware.

Brief introduction to Python. Why we use it. What’s good.

Configuring the modules used for signal manipulation and analysis.   
PyAudio  
Numpy  
SciPy  
MatPlotLib  
SoundDevice  
Soundfile

<https://python-sounddevice.readthedocs.io/en/0.3.8/index.html>  
<http://pysoundfile.readthedocs.io/en/0.9.0/>

Audio Processing  
Playing audio can be done in many ways. The most straightforward is to create an array with audio data, and use the SoundDevice (sd) library’s play() function along with the sample rate:

sd**.**play(myarray, samplerate)

This will play the audio to the default device, unless specified.   
A file can be loaded from disk and stored in an array with SoundFile (sf) library’s read() function:

myarray, samplerate **=** sf**.**read(‘filename.wav’)

sd**.**play(myarray, samplerate)

This method will also read the audio file’s sample rate.

Recording audio can also be done using SoundDevice using the rec() function:

duration **=** 10.5 *# seconds*

myrecording **=** sd**.**rec(int(duration **\*** fs), samplerate**=**fs, channels**=**2)

This will record audio from the default input device for the duration specified and store it as the Numpy array myrecording

These basic methods are useful for simple applications, but require the entire audio file being played to be stored in memory. For very large files this can use all available resources

Streams

Audio Input

Input and output simultaneously: Wire, transceiver.

Signal generation: Tone, Multiple tones/chords, Stereo signals

Correlation: Cross-correlation, Autocorrelation, noisy correlation

DFT: Single tone, multiple tones, phase, spectral spread, zero-pad, window

TOA: send to receive.

TDOA: loopback, send to receive

FM Broadcast