# AUCS: Communication between Autonomous Vehicles using Ultrasonic Sensors, or Autonomous Ultrasonic Communication System.

Thomas Hansen
*UW Madison*

Ajene Johnson
*UW Madison*

## Abstract

As we move forward in time, an increasing number of cars are expected to be either semi or fully autonomous, a trend we are seeing from major car manufacturers and startups such as Tesla, Uber, and Waymo. These cars usually have ultrasonic and visual sensors on them, and some such as Uber and Waymo additionally use Lidar to help map the world around them. Our system takes some of the pre existing sensors, namely ultrasonic sensors, and is able to communicate basic intentions between cars to allow them to more easily communicate whether they want to move into a lane, change speeds, or perform other basic functionality.

## 1 Introduction

Currently, if a vehicle tries to merge into a lane in front of your semi-autonomous car, you'll likely experience a quick jerking motion as your vehicle adjusts to the new vehicles speed and position. If you ride in a Waymo vehicle, you will experience long waits as you attempt to merge into traffic or change lanes, as the car needs to be certain the merge will be safe. Autonomous vehicles currently aren't very good at predicting the movements of other cars, or optimizing ride comfort, which is where our system comes in. We hope to alleviate this issue by providing autonomous vehicles the intended movement of surrounding vehicles, similar to the types of communication humans already participate in when they wave someone in or preemptively slow down. Most autonomous communications systems, such as CarSpeak, attempt to share a map of the world with all the cars around it, allowing other cars to see a shared map of all objects and things around it [3]. AVR attempts to share large point clouds of their surroundings to other cars, dealing with bandwidth issues and large processing requirements [2]. Our system communicates similarly to how people do, telling other cars around it what actions it intends to make, and waiting for them to change their behavior so it may perform the action. This is more similar to a driver waving another car past compared to telling it what is sees, and is less of a risk to the driver and the car since the information shared is only recorded as received once the receiving car changes its behavior in accordance with the message.

Autonomous Ultrasonic Communication System, or AUCS, is an important change of direction for inter-car communication, because instead of attempting to share the world around it and transmit complex point clouds between cars, it simplifies the entire process. Cars communicate between each other using hardware on the car that is already present, and require less significant changes to the driving algorithms to function. Cars send out a request to perform a function to the cars around it, and it continues to wait for an opening to perform the action. If an opening appears, it may use it, and if a car responds and moves because of the message, it may take the opening as well. Our system doesn't require a new allocation of bandwidth and doesn't break down if other cars in its vicinity do not have the system. AUCS improves rider quality, creating a more pleasant driving experience and preventing accidents where predictive information could be used in collision avoidance.

## 2 Related Work

Our contributions provide a new insight into communication between autonomous vehicles. Most other related work involves sharing visual data and point clouds between vehicles. Ours communications uses preexisting ultrasonic sensors, and only shares intents, not acknowledgements or behavior. This makes it less vulnerable to threats and more similar to how humans behave while driving on the road. Our major contributions to this paper are as follows

- Communication on bands typically reserved for ultrasonic distance detection.
- Is able to receive messages and interpret them using ultrasonic bands

- Receiving messages despite ambient interference from ultrasonic sensors
- Creating a list of action codes relevant to car motion

### 2.1 Ultrasonic Sensors in Autonomous Vehicles

There already exists a precedent for using ultrasonic sound in autonomous vehicles. As stated previously, semi-autonomous and fully-autonomous vehicles have ultrasonic sensors already installed to measure their proximity to their surroundings and prevent collisions. For example, Tesla's Enhanced Autopilot system has twelve ultrasonic sensors surrounding the car, each with a range of about 8 meters. These systems, in their current implementation, are only beneficial to a single car. By implementing AUCS, these systems can transmit data that will make autonomous driving beneficial to other cars on the road. Implementing AUCS would have a low overhead because autonomous vehicles already have a setup that allows them to determine the signals location, since ultrasound is currently used to map out the surrounding area.

### 2.2 Ultrasonic Sensors in Communication

Cellphones are currently taking advantage of ultrasonic beacons in multiple ways. Google Chromecast, a wireless device that plugs into a TV's HDMI port, pairs with cellphones and computers by emitting ultrasonic beacons to send a pin to users' devices. This establishes a connection between the device and Chromecast to take place, without being beholden to the owners WiFi network. This is done as a security measure to ensure the user connecting to the device is in fact in the room nearby, and sends a number of inaudible audio tones that can be registered on the phones microphone, verifying the user [4].

Another popular use of ultrasonic emissions is cross-device tracking [6]. Cross-device tracking is the action of tracking users across multiple devices by using ultrasonic signals emitted by one device and received by another device's microphone. One utilization of cross-device tracking is advertising. Companies will inject ultrasonic beacons into their online ads, which will be recognized by the user's cellphones, and deliver relevant advertisements to the users' phones.

The United States Army has looked into using ultrasonic communication as an Identification of Friend-or-Foe (IFF) technique [1]. The authors say that ultrasonic communication is highly feasible for systems that do not require a high throughput, and that multipathing effects can be turned into a geographic advantage when sending data at lower bit rates. This is perfect for AUCS because we do not need to transfer data at high bit rates and only simple information needs to be sent, as is the case with IFF.

## 3 Methodology

### 3.1 Frequency Shift Keying

In order to produce and interpret data using ultrasonic sensors, we created a protocol that utilizes the Frequency Shift Keying (FSK) technique. FSK works by establishing a carrier frequency, then modulating the frequency to represent different bits. Our specific implementation was Binary FSK. In this implementation of FSK, the carrier frequency is modulated to a higher frequency in order to represent a "1" or the mark frequency. A "0" is represented by modulating the signal to a lower frequency, which is known as the space frequency.

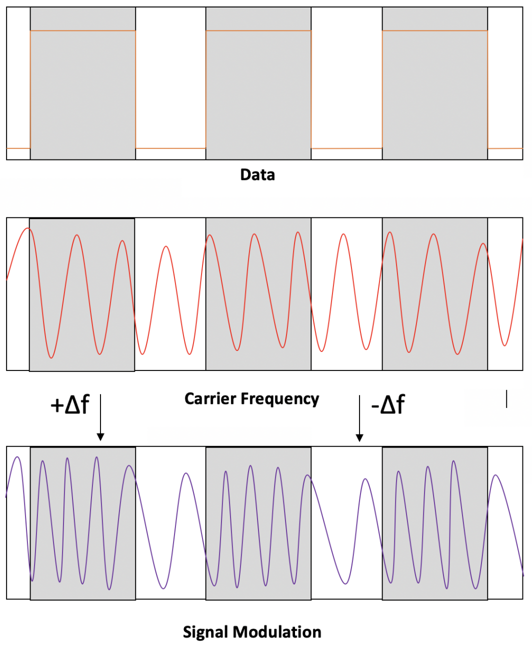

Figure 1: The first diagram is the data packet that will be sent. The second diagram is the carrier frequency that will be modulated to represent the bits. The last diagram is what the carrier frequency looks like after it is modulated to higher and lower frequencies.

### 3.2 Protocol

The AUCS protocol transmits packets that are 18 bits in length. The first three bits are the preamble, which allows receivers to recognize that a packet is being transmitted. The preamble is "101" because this will allow receivers to distinguish which frequency represents a "1" bit and which is a

| Action | Code | Distance |
|---|---|---|
| Right turn | 0b0001 | |
| Left turn | 0b0010 | |
| Change lane (right) | 0b0011 | |
| Change lane (left) | 0b0100 | |
| Merge (right) | 0b0101 | |
| Merge (left) | 0b0110 | |
| Offramp | 0b0111 | |
| Temporarily change lanes (incident) | 0b1000 | |
| Speed (speeding up) | 0b1001 | Not used |
| Speed (slowing down) | 0b1010 | Not used |

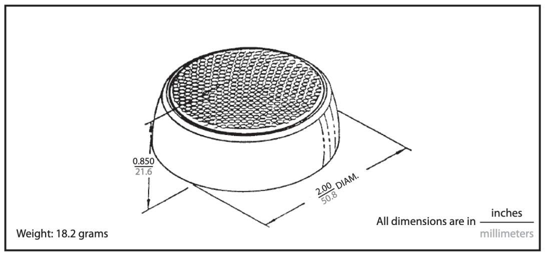

Figure 2: The KSN 1193A piezoelectric speaker from Piezo Source [5].

"0" bit. Autonomous vehicles will constantly listen for the preamble, which is possible because there is little concern for power conservation due to the large size of car batteries.

The next three bits represent the version number of the protocol. It is important to know the version of the protocol because a different version might mean that the receiver has to change the way it interprets a data packet, and if the version number is invalid the packet is dropped and we assume there was an error in transmission or reception. The next four bits represent the action that the transmitting car will take. Actions vehicles might take are turning, merging, and changing speeds, and are represented as predefined codes transmitted as a positive integer. The final eight bits represent the distance from the transmitting vehicle to the area where the vehicle will enact the action, where the distance is measured in meters. This final section is all zeros if the action doesn't require a distance attribute.

Below we have a table of the different values we used, including whether they relied on the distance metric and what the intent is (with limited information as to why they're doing it).

## 3.3 Architecture

In order to transmit data, we used a piezoelectric speaker, as illustrated in Figure 2, which has a frequency response ranging from 5kHz to 30KHz. This speaker was connected to a computer through the auxiliary port and the computer ran a program that generated AUCS data packets. The receiver was another computer that listened to the transmission with its built-in microphone and translated the signal into bits.

## 3.4 Signal Interference

There are two primary sources for signal interference in our system. One is from other cars using the AUCS protocol, and the other is from cars using ultrasonic bands to get relative distances to other cars.

Our first method of preventing interference is a band pass filter to remove lower frequency road noises and noise from the surrounding area. The second primary method of preventing signal interference is transmitting signals at random frequencies, where we attempt to detect the high and low frequency signals (representing a 1 and 0 bit) by checking frequency differences first, instead of looking for particular frequencies. This should be a relatively safe method of transmitting signals, and although we expect interference, we think it should be manageable. We can calculate the probability of any two signals interfering by using the following equation:

$$1 - \frac{n!}{n^k (n-k)!}$$

- Where n = number of frequency slots to communicate over,
- k = number of vehicles trying to communicate in the same airspace

This suggests that over a frequency range of 48 kHz to 50 kHz (n=2000), if you had 4 cars attempting to communicate their own signal, we'd experience a .3% chance of a signal collision, requiring a re-transmit. Although our system acts like slotted ALOHA, in the sense that it communicates over certain bands or different frequencies, it does not listen for interference to see if it must send again. The algorithm expects many transmissions until the intended action has been performed, so if collision occurs it will simply collide and re-transmit as if it didn't collide at all.

As other cars attempt to calculate the distance between them and other vehicles, they ping random frequency's out from the sensor and listen back for the response. This is occurring over the same bandwidth we're attempting to communicate on, potentially causing a problem. However, this sort of interference shouldn't be a problem because the pings from other cars are short enough that their relative power compared to the intended signal on an FFT over the signal period should be negligible, and because our system can safely ignore them even if they're near the frequencies we're listening for. The signals can show up if there are no other signals occurring at that point, however the probability of the random pings faking the preamble of a message, let alone the header, and doing so repeatedly is abysmally small, and poses no real threat to the user.

### 3.5 Doppler Shift

Lastly, we had to take into account the possibility of the Doppler Shift negatively impacting our performance. This might shift a frequency up or down depending on the direction the two vehicles are traveling. Given an initial frequency of $f_0$, we can calculate the new frequency we might see as the following:

$$f = f_0 \frac{c \pm v_r}{c \pm v_t}$$

Where $v_r$ is the velocity of the car receiving and $v_t$ is the velocity of the car transmitting.

Since the absolute frequency isn't known, cars are free to shift frequency due to the Doppler Shift. The problem that arises is when one is accelerating while transmitting and changes frequencies by the time it has finished sending the signal. This shouldn't be a problem however due to how short our packet lengths are, as the resulting frequency difference would be minimal.

There additionally might be changes in the packet length, however, due to the Doppler Shift. An 18 bit packet having each bit transmitted over the course of 1ms is expected to take 18ms to transmit. If one car was moving at 60mph and another was moving at 45, each packet could be interpreted by the faster car as having a total length of 18.62ms, where each bit only takes .98ms to transmit. Although at lower speed differences the issue is negligible and the packets are able to be read, this poses a major problem for sustained error, meaning one car repeatedly reads a packet incorrectly and changes its behavior based on the wrong information. This could easily be corrected at the receiver by including information about the other car's speed and adjusting the length of each bit it expects to read. We can identify which car is sending the signal by cross referencing the point cloud with the sensor we received the signal on, and adjusting for the time to read each bit. This method proved to be very successful in accurately reading transmitted packets.

## 4 Design

The overall design of the system can be broken down into two parts: the transmitter and the receiver. The transmitter constructed a signal based off of randomly selected high and low frequencies, while the receiver had to interpret this high and low frequency, and then translate it into a code and distance.

### 4.1 Transmitter

The transmitter simply took a high and low frequency, and turned it into a set of points which it then produced sound for. Notably, it tracks the phase of each frequency and resumes the next frequency at that phase to improve the resulting audio from the speaker and reduce noise. The algorithm is approximated by the following.

```
generateSignal(dataSet)
    lf = rand_low_freq
    hf = rand_low_freq + frequency_shift

    bits = [1,0,1,0,0,1,dataSet]
    freqs = ones(length(bits)) .* lf

    % assign 1 bits to high frequency
    for 1 = 1:length(bits)
        if bits(i) == 1
            freqs(i) = hf
        end
    end

    % add frequencies to signal
    for i = 1:length(packet)
        t = 0:1/fs:bit_duration;
        f = sin(2*pi*packet(i).*t + phase);

        % update phase
        phase = phase + 2*pi*packet(i)*bit_duration;

        % add in next position
        ff(start_leg:end_leg) = f;
    end
end
```

### 4.2 Receiver

Our receiver worked by plotting the most powerful frequencies and then finding patters in this data to distinguish the high and low frequencies, as well what data was being sent. Knowing the anticipated length of time for each bit sent, we calculated an FFT over each of those lengths of times, found the most prominent frequency, and assigned that to a 0 or 1 bit. Once we verified the preamble and header to be correct, we were able to use the rest of the FFT data to find each of the proceeding bits and translate the data packet into the intent of the nearby car. In practice, we'd continue to receive multiple of these packets every second, and would be able to cross check them to remove bit error.

## 5 Evaluation

When evaluating AUCS, we collected data in a controlled environment using a PiezoSource speaker and a MacBook

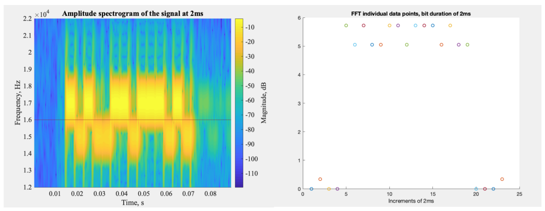

Figure 3: On the left is the power plot of an example set of bit data. On the right is the data plotted out by our FFT algorithm, which was then successfully able to recreate the signal.

2016 microphone, all done at lower frequencies due to the physical limitations of the MacBook and speaker.

## 5.1 Length of Bit Signals

The length in time of each bit was important to how well the receiver was able to interpret a data packet. Our goal was to send packets as fast as possible in order to avoid Doppler Shift effects, while also making sure that the packet was still readable. We tried having the receiver sample at .2ms per bit, but upon creating the spectrogram we found that .2ms was much too fast for the receiver, as can be seen in Figure 5. We found that making the data packet transmit bits at 2ms intervals allowed for a reasonably fast transmission rate that still allowed the receiver to distinguish between a 0 and a 1 bit, which is demonstrated in Figure 4.

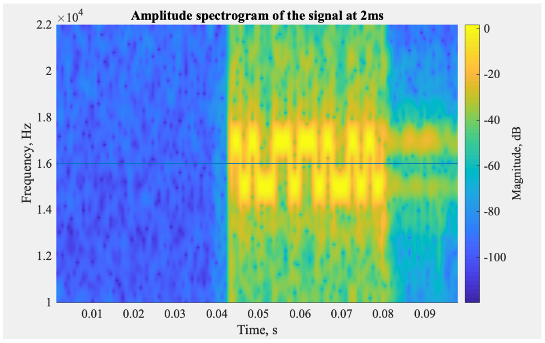

Figure 4: Above is a signal generated with data, where each bit was transmitted over 2ms. This allowed us to create high resolution bit maps where the bits are distinguishable from each other.

## 5.2 Successful transmission

We had high levels of success when transmitting in a controlled environment, such as the packet in Figure 3, and did

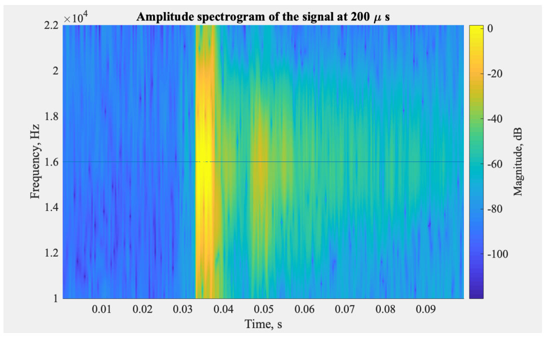

Figure 5: Our .2ms signal, where you can see the signal strengths mixing. This is primarily due to the lack of resolution in our microphone, preventing us from communicating over faster bitrates

not fail to read or transmit a bit while operating within the range of our microphone. We performed a variety of tests, where each bit was sent with a duration of 2ms and each packet would take 36ms to transmit. This gave it a total throughput of 500 bits per second, or 62.5bytes per second.

## 5.3 Transmitting with interference

In the second half of our evaluation we attempted to send signals with 'pings' from ultrasonic speakers that were trying to calculate distance from nearby cars. Although there is variation in lengths of time and frequencies used by different manufacturers, we approximated these 'pings' by adding higher and lower frequencies to the generated data, and then introduced noise by outputting it through the speaker and listening through a microphone. We found that pings relatively small in size compared to the signal were largely ignored and had little impact on our data.

## 6 Future Work

Something we'd be interested in is testing data with OEM hardware. The microphone we used to implement the receiver could only receive signals up to around 20 kHz, and the maximum frequency that our speaker could transmit at was 30 kHz. Using an ultrasonic sensor from a Tesla or other semi-autonomous vehicle would allow for better insight into exactly what roadblocks we would face when attempting to communicate between vehicles. Ultrasonic sensors in autonomous vehicles operate around 48 kHz, so for our experiments we operated at a lower frequency than most of the automotive world.

The transmitter and receiver that we tested AUCS with were both stationary. Since AUCS will be implemented in cars, we should create an environment that is closer to how

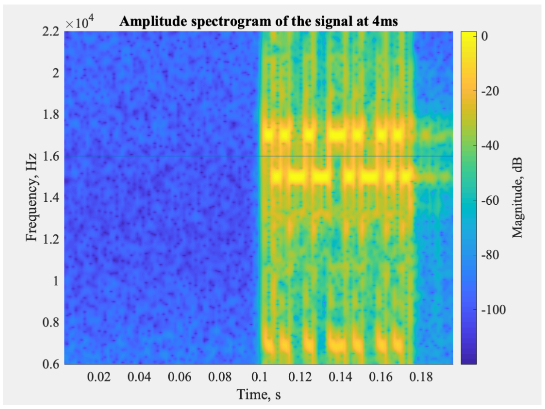

Figure 6: This is a 4ms signal with multiple disturbances in the preamble, approximating interference from distance signals.

AUCS would be used by cars. In a future set of testing we could hook up ultrasonic speakers to two cars and test them on the road, to see if we can communicate in a real world environment.

## 7 Contributions

Thomas implemented the transmitter code to successfully emit sound at varying frequencies. Ajene discovered FSK and came up with he name AUCS. Both partners worked on the receiver, experimenting with how fast the receiver was able to translate the signals into bits. Both partners decided how to implement the protocol and we were fortunate enough to take feedback and help from our Mobile Networking class.

## References

[1] David Tofsted, Sean OBrien, S. D. E. C. S. E. An examination of the feasibility of ultrasonic communications links.

[2] Hang Qiu, Fawad Ahmad, F. B. Avr: Augmented vehicular reality. *ACM MobiSys* (2018).

[3] Pascal Getreuer, Chet Gnegy, R. F. L. R. A. S. Carspeak: A content-centric network for autonomous driving. *ACM Sigcomm 20*, 6 (2012), 1277 – 1290.

[4] Patrick de Niet, S. H. Chromecast: Security analysis of the guest mode. *University of Amsterdam* (2015).

[5] Source, P. Ksn 1192a-1193a.

[6] Vasilios Mavroudis, Shuang Hao, Y. F. F. M. C. K., and Vigna, G. On the privacy and security of the ultrasound ecosystem. *Proceedings on Privacy Enhancing Technologies 2017*, 2 (2017), 95–112.