Acoustic Shockwave-Based Bearing Estimation

János Sallai, Péter Völgyesi, Ákos Lédeczi, Ken Pence, Ted Bapty, Sandeep Neema, James Davis
Institute for Software Integrated Systems
Vanderbilt University
Nashville, TN, USA
akos.ledeczi@vanderbilt.edu

ABSTRACT

The paper presents a smartphone-based shooter localization system. As muzzle blasts are difficult to detect at longer distances and consequently present higher false detection rates, the system relies on shockwaves only. Each sensor uses four microphones to detect the Angle of Arrival and the length of the shockwave. This information, along with the sensor's own GPS coordinates, are shared among nearby smartphones. Assuming a known weapon type, it then proceeds to estimate the two possible projectile trajectory candidates for each sensor that are consistent with the observations in the horizontal plane of the sensors. A simple clustering algorithm identifies the correct projectile trajectory relying on as few as two sensors. The trajectory is then used to estimate the bearing to the shooter relative to each sensor. The paper presents the overall system architecture, the design of the sensor node that interfaces with the smartphone, the trajectory and bearing estimation algorithms, and the evaluation of the system based on a field experiment.

Categories and Subject Descriptors

C.2.4 [**Computer-Communications Networks**]: Distributed Systems; J.7 [**Computers in Other Systems**]: Military
General Terms: Design, Measurement, Performance
Keywords: Sensor networks, data fusion, acoustic source localization, shooter localization
Acknowledgments: The activities described in this paper have been supported by the DARPA Transformative Apps program. We are grateful for the contributions of the Nashville Metro Police Training Academy and DataBouy, LLC. We would like to thank the anonymous reviewers and our shepherd, Andreas Andreou, for the helpful comments that greatly improved the quality of the paper.

1. INTRODUCTION

A decade after the first prototype wireless sensor network-based acoustic shooter localization system was reported in the technical literature [14], there is still no deployed or commercially available system that utilizes a network of sensors. The state-of-the-art is still a standalone unit. The reluctance of the military to embrace a wireless solution is understandable since they require any deployed networking technology to be certified and secure. The current communications infrastructure simply does not lend itself to supporting low cost wearable networked sensors. However, a recent development promises to alleviate this problem. The DARPA Transformative Apps program is built around the idea that every soldier will carry a military certified smartphone in the near future. In addition to a number of innovative military apps, a common middleware platform, called AMMO for Android Mobile Middleware Objects, is being developed that will support a variety of communication protocols [19]. The communication needs of a networked shooter localization system will be easily supported by AMMO. The caveat is that the system will have to be based on smartphones.

The typical smartphone has all the components required by acoustic shooter localization: an audio channel, processing capabilities, wireless communications, GPS, and a display. In fact, it would be possible to build a shooter localization app, but the design tradeoffs and the closed nature of the hardware platform would result in subpar performance. First, the maximum sampling rate of the audio channel (48 kHz) would be satisfactory, but the audio stream is typically preprocessed by the on-board codec chip (noise reduction, filtering, etc.), erasing and distorting important signal features, especially the sharp rising edge of the shock wave. Also, most people carry their phones in their pockets or bags making acoustic event detection nearly impossible. Second, the low-cost GPS solutions in phones contain too large of positioning errors for shooter localization where sensor positions need to be known with about a meter accuracy [8]. Third, time synchronization is typically a must for networked source localization. However, application programs running on top of smartphone OSs do not have access to precise global time in spite of the availability of GPS. Also, the exact sampling times of individual audio samples are unavailable, as well, because of the unknown delays and jitter associated with the hardware (ADC, codec, I/O bus) and the software (kernel drivers, I/O and task scheduling, etc.). The solution is to build a sensor that offloads the detection, self-localization, and time synchronization tasks from the phone. (Note that our system ended up not needing time synchronization, but this was unknown at the time of hardware design, and a precise GPS was needed in any case.) Such a sensor can have its own microphone(s), sample at a higher rate, and run the shot event detection algorithm. It only needs to notify the phone when an event detection was made. The event can be tagged with its relevant features, such as Time of Arrival (TOA), length, etc., as well as the corresponding sensor location. The sensor-phone interface can be either Bluetooth or USB. The rest of the paper presents the design, implementation and evaluation of such a system.

2. RELATED WORK

Acoustic shooter localization has a long and rich history. Here we only summarize the results that are most relevant to our work.

When the typical rifle is fired, two acoustic events are generated: the muzzle blast and shockwave. The muzzle blast is associated with the explosion that occurs in the barrel of the gun that ejects the projectile through the muzzle at a high speed. When this speed is supersonic – true for the vast majority of rifles – the projectile generates an acoustic wavefront called a shockwave. This wavefront has a conical shape. The axis of the cone is the trajectory of the bullet. The angle is determined by the ratio of the speed of the bullet to the speed of sound (also called the Mach number). Being an acoustic phenomenon, the wavefront itself propagates at the speed of sound.

Figure 1 illustrates the shockwave propagation. Shooter **S** fires a bullet along the trajectory **SA**. Microphone **M** detects the shockwave front after a certain amount of time. This is the sum of the time the projectile needed to cover the distance **SA** at its supersonic speed and the time it takes for the shockwave to cover the **AM** distance, also called slant range, at the speed of sound. The distance **BM** is called the closest point of approach (CPA) or miss distance. It is worthwhile to note that in reality, the projectile constantly decelerates, so the section of the shockwave front shown in the figure is not a straight line, but slowly curves with the angle α continuously increasing. Finally, the muzzle blast travels the SM distance at the speed of sound (not shown in the figure).

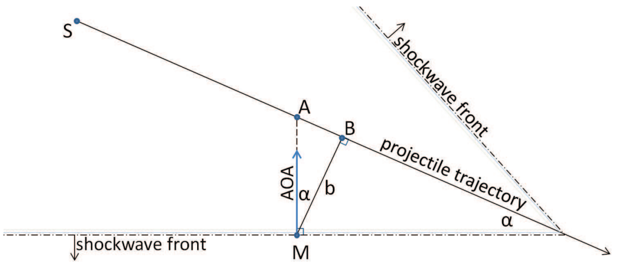


Figure 1: The geometry of a shockwave generated by a supersonic projectile originating at point S and observed at point M.

The signal shape of a shockwave is highly characteristic. It resembles the letter N with sub microsecond rise times and typical lengths of 100 to 600 microseconds. An empirical formula called the Whitham's equation [18], relates the shockwave length T to the CPA b, the Mach number M, the speed of sound c, and the projectile diameter (caliber) d and length l:

$$T = \frac{1.82 M b^{1/4}}{c(M^2-1)^{3/8}} \frac{d}{l^{1/4}} \approx \frac{1.82 d}{c} (\frac{Mb}{l})^{1/4} \tag{1}$$

Recently, Sallai et al. rearranged Whitham's equation by grouping together most parameters into a scaling factor k [12]:

$$b \approx kT^4 \tag{2}$$

They observed that multiple sensors detecting the same shot will necessarily have to have the same k (assuming a constant bullet speed near the sensors). They showed that measuring the shockwave length at multiple sensors is sufficient to estimate the trajectory of the projectile [12]. Interestingly, k also falls out of the solution providing accurate caliber estimation and weapon classification.

Previously, Duckworth et al. described an approach where shockwave TOAs measured by at least six sensors (either six widely distributed omnidirectional microphones or two small arrays of three microphones each) is sufficient to accurately determine the trajectory of the projectile and an approximate range to the shooter [6]. If muzzle blasts are also available, the range accuracy greatly improves. They developed a very precise ballistic model to achieve remarkable accuracy. However, the method relies on empirically derived caliber- and sensor-specific parameters for sensitivity and frequency response, making calibration necessary. Accurate measurement of the peak amplitude of the shockwave is also needed. Since nearby trajectories generate high energy shocks, microphones are often overdriven by them. Barger et al. overcomes some of these problems by introducing an accelerometer-based sensor replacing microphones altogether [2]. However, it is not clear how sensitive these methods are to sensor location and time synchronization errors.

Damarla et al. and Lindgren et al. demonstrated an innovative technique that relies on TDOA of the shockwave and muzzle blast on the same nodes only. That means that time synchronization is no longer required in the network [5, 9]. The main disadvantage of the method is that the muzzle blast needs to be detected on each node. Sensors that only detect either the shockwave or the muzzle blast cannot participate in the fusion at all. Also, the technique requires a known projectile speed along the trajectory and is computationally expensive. Recently, Ash et al. quantified what effect a constant bullet speed assumption has on the accuracy in [1].

Volgyesi et al. presented a wireless sensor network-based system in [17]. It relies on measuring both muzzle blast and shockwave AOAs on each sensor and TDOAs across the sensors. A single sensor alone can locate the shooter if it detects both events on at least three microphones each. The network fusion localizes the trajectory and shooter position, estimates the weapon caliber, and classifies the weapon at a remarkable accuracy. The main disadvantage of the system is that it cannot handle longer range shots that pass on one side of the sensor field. The problem is that if only one side of the shockwave is measured, a large number of various trajectories could explain the TDOA observations. If there are not enough muzzle blast detections in the network, the trajectory and the shooter cannot be localized with their approach.

3. DESIGN MOTIVATORS

As we have seen in the previous section, most acoustic shooter localization systems rely on both the shockwave and the muzzle blast. Unfortunately, the muzzle blast is not nearly as characteristic an acoustic event as the shockwave. Moreover, the shockwave typically originates near the sensors as they are usually placed near the protected area, since this is presumably where the shot is targeted. Muzzle blasts, on the other hand, travel much longer distances. Consequently, their energy dissipates and the signal shape gets distorted. Muzzle blast detection beyond a hundred meters quickly becomes unreliable and makes the system prone to false detections. Finally, the muzzle blasts of friendly return fire would be picked up by the sensors and would make it impossible for the sensor fusion to sort through the numerous detections. The shockwave of a soldier's shot, on the other hand, does not even reach its own sensor because of the geometry.

Hence, the most important design decision was to make the system rely on shockwaves alone. While the numerous advantages of this were described above, it comes at a steep price. Most significantly, without the muzzle blast, the range to the shooter is impossible to estimate accurately. (More precisely, it is possible to estimate by measuring the curvature of the shockwave due to the deceleration of the projectile, but it would require larger microphone spacing than is practical in a wearable sensor.) Fortunately, the bearing to the shooter is considered far more valuable information for the warfighters than the range.

The second design motivator of the system was the requirement to provide useful information relying on as few sensors as possible. The obvious consequence of this was to opt for multi-channel sensors. Having a mini microphone array provides AOA information. Having the shockwave AOA and its length makes it possible to estimate the miss distance and characterize the wavefront (see Section 5). This is almost enough to determine the trajectory of the bullet. Consider Figure 2.

Assume for a second that the speed of the projectile is known. As we have seen in Section 2, the bullet speed determines the angle of the shockwave cone and hence, the trajectory angle can be estimated from the shockwave AOA. Unfortunately, a second trajectory, called image trajectory, can generate the same shockwave observations, as shown in the figure. Therefore, a single sensor cannot determine the trajectory unambiguously. However, two sensors are enough to disambiguate the situation. See Section 5. Note that this discussion was restricted to 2D; the situation is more complicated in 3D, but the solution can be generalized. Nevertheless, the consequence of the design decisions is that the minimum number of sensors required is two.

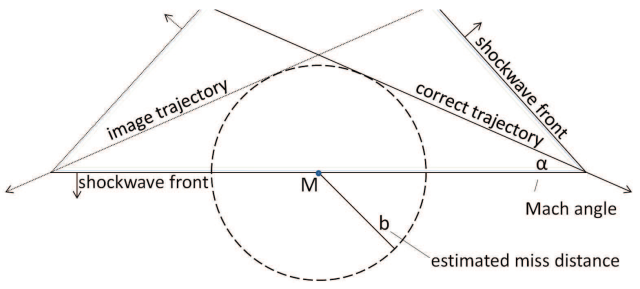


Figure 2: The trajectory ambiguity problem.

4. SYSTEM ARCHITECTURE

The intended application and concept of operation requires a truly distributed architecture, which is a significant departure from previous shooter localization systems [13, 3, 14, 17], in which a dedicated *base station* collects low-level measurements, computes the solution, and displays the results. Due to the low-bandwidth requirements—a single message per node when a shot is fired with sporadic status updates for tracking the position of other team members—and due to a low practical bound on the number of nodes within audible range of the trajectory, we could implement a simple and robust distributed model. In this model, all detection events with time and node orientation and position information are broadcast to all other nodes. Based on these broadcasted events and the local detections, each node independently calculates its own solution. Unless messages are lost, these independent results will be identical.

The conceptual system architecture is shown in Figure 3. An Android smartphone at each mobile node (soldier) provides three high-level services: it receives and broadcasts event detection messages, executes the localization algorithm, and displays the results. The smartphone provides multiple options for message distribution. Previously, we used IP/UDP multicast with dedicated WiFi access points, UDP message broker servers where 3G services are available, and a more sophisticated middleware layer providing broadcast services over traditional and custom radio interfaces. Such a custom RF interface might be the only viable option in remote locations with no reliable cellular coverage or where the service provider cannot be trusted.

Note that these tasks require significant CPU, memory and user interface resources, thus delegating these to the smartphone seems natural. Also, the more productive and refined developer ecosystem around the Android platform significantly shortens the time and effort for refining the

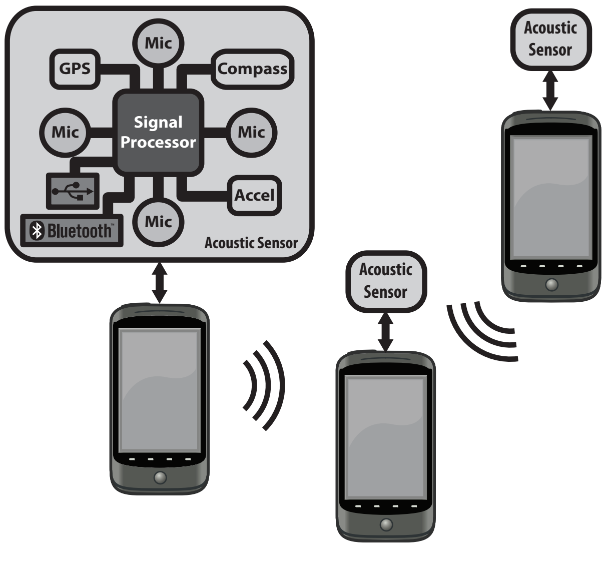


Figure 3: Conceptual system architecture.

most complex parts of our application. On the other hand, none of these tasks require real-time guarantees, an area where smartphone *apps* do not excel.

A custom sensorboard provides all real-time services of the application. First, it needs to sample and monitor multiple (four) audio channels for detecting potential shockwave signatures. Shockwave length and AOA (based on TDOA) measurements are more accurate using higher sampling rates than are typical in audio applications. This multi-channel, high-bandwidth, real-time signal processing task requires significant computational resources, which can be met with parallel architectures while staying within the limited power budget. Systems with similar requirements have used SRAM and Flash-based FPGA devices [14, 7, 15]. Alternately, the PSoC mixed-signal platform [4] is also a possible candidate for such tasks. The stream processing is easily handled by mainstream midrange FPGAs running at low (10 MHz) clock speeds, but the development time and effort is sometimes prohibitive to try out radically new ideas. The PSoC platform provides limited (re)configurable logic resources (Universal Digital Blocks) along with on-chip analog components for signal conditioning and sampling and an industry standard ARM Cortex-M3 processor core. While software development on this platform is significantly faster and more convenient, the PSoC platform does not scale well with the number of channels. Beyond two channels, the scheduling of the shockwave detection and other tasks and satisfying latency requirements becomes non-trivial.

These other tasks include communication with an onboard GPS and a 3D compass module for accurately tracking the current time, position, and orientation of the microphone array. Although our current localization algorithm does not require highly accurate timestamps of the events across nodes (only used for separating different shots), a timepulse signal provided by most GPS modules is a simple and reliable way of generating such timestamps. The position accuracy of the GPS module is more important and affects the overall accuracy of the solution. The 3D compass module uses MEMS accelerometers and magnetometers for calculating the actual heading, pitch, and roll angles of the array. Accuracy and refresh rate were the two critical parameters in selecting the compass module.

Building the connection between the sensor array and the smartphone is surprisingly challenging in this application. Although the communication channel does not have to provide real-time guarantees—all time critical information is captured and registered by the sensor node—the communication interface needs to provide a reliable yet physically flexible bridge between the endpoints. The potential technologies are limited by the built-in capabilities of the smartphone platform to USB, Bluetooth, or WiFi. The wireless interfaces provide the most flexible deployment models; however, these are also more vulnerable to eavesdropping or jamming. The USB interface requires a wired connection between the array and phone and may also enable a shared power supply solution. Unfortunately, the standard smartphone USB interface requires a host implementation on the other side, unnecessarily increasing the design complexity of the sensor node.

In typical deployment configurations, the sensor node is required to run from an independent power source for several hours during the entire length of a mission. The current architecture and detection logic offer only limited energy

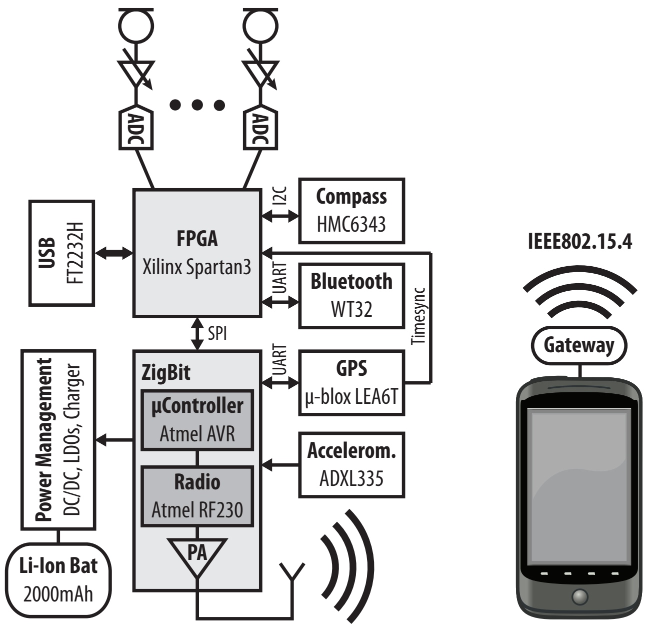


Figure 4: Current prototype system architecture.

saving opportunities via duty cycling. The channel detectors are enabled continuously—these need to find the first acoustic transient event with no advanced trigger. The GPS receiver needs to track the position and time of the node also. Here, energy saving measures can be implemented with a low-power MEMS accelerometer for detecting stationary periods when the GPS module can be powered off and falling back to internal timekeeping. Nonetheless, the sensor node requires a high energy density Li-Ion battery with a rated capacity comparable to ones in current smartphones (1000-2000 mAh).

Our current prototype implementation shown in Figure 4 follows the conceptual architecture described above. However, as it happens with most research prototypes, we had to make changes to and diverge from the ideal setup. The main factors affecting the proof-of-concept prototype were very limited time and the financial budget. To be able to build a working and deployable system, we decided to leverage an existing acoustic sensor board (Octopus) [16] and design an add-on circuit board with the components needed for the system concept. The original sensor board supports eight (8) independent channels, a high-speed USB interface, and a large PSRAM memory for instrumentation purposes, which results in a significantly higher power budget and larger physical size than is needed by the current application. Also, the two stacked circuit boards and the necessary wiring between these made the profile of the node significantly higher. Finally, to decouple the early development phases from the communication concerns of the smartphone network, we decided to use the existing IEEE 802.15.4-based RF solution readily available on the Octopus board. In this modified architecture, the smartphone still runs the sensor fusion and provides the user interface.

On the existing Octopus sensor board, we use only four (4) of the audio channels that have variable gain (44–65 dB) and 1 MSPS/12 bit sampling. The independent channels are processed by a Xilinx Spartan3 FPGA (XC3S1000) using cus-

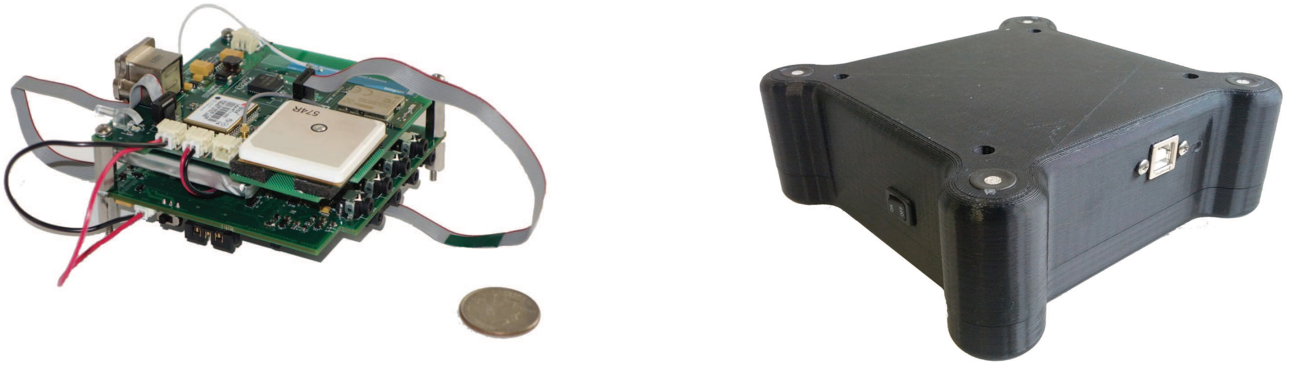

(a) (b)

Figure 5: The prototype sensor node assembly (a) and packaging (b).

tom IP cores developed for detecting acoustic shockwave signatures. The high-speed USB interface (FTDI FT2232H) on the Octopus board provides a real-time streaming interface to the audio channels. The ZigBit OEM module provides communication, system management, and limited computational resources. It includes an Atmel AVR (ATmega1281) microcontroller, an Atmel RF230 RF transceiver chip operating in the 2.4 GHz ISM band, and an integrated power amplifier (20 dBm) for increased communication range and noise immunity. Each smartphone (Dell Streak) is connected to a TelosB [11] node using a custom USB cable to use the non-standard USB host capability built into the Streak. The TelosB acts as a wireless gateway and forwards all packets to the phone.

Mobile and autonomous operation is supported by several key components on the support board. First, we integrated a u-blox LEA-6T GPS receiver, which provides accurate position estimates in standalone mode with satellite augmentation, an extremely accurate global time reference, and access to raw measurement data. Node orientation is tracked by a Honeywell HMC6343 3D compass module at 10 Hz and 0.1° resolution. This compass module combines a 3-axis magnetometer and a 3-axis accelerometer MEMS sensor with digital/analog support circuitry and software IP for calculating heading, pitch, and roll results from raw sensor measurements. These sensitive measurements are affected by bias and noise as soon as the compass module becomes a part of a real board assembly. The module provides a self calibration process, which we used successfully for compensating the bias of the magnetometers caused by close metallic/magnetic objects (PCB traces and planes, RF shields, and antennae). We also experienced significant measurement noise, which was caused by an inductor coil of a switching-mode DC/DC converter on the board. By changing the layout of this circuitry we managed to mitigate the interference and the noise.

Acoustic shockwave detection is not the intended application of the low-cost low-power miniature electret microphones we used in our prior work. These sensors typically lack in responsiveness and dynamic range to handle such short high energy transients. The high sound pressure level (SPL>150dB) of close trajectories is especially problematic due to the long recovery time of the mic (the JFET preamplifier inside the capsule) from overload. During this recovery interval the microphone is practically deaf, which prevents accurate shockwave length estimation and hinders the detection of simultaneous shots. For the current system we selected a more robust, water resistant microphone assembly from Knowles Acoustics (VEK-H-30108). This is a low-sensitivity model designed for gunshot or other impulse sensing, thus can handle more extreme SPL (124dB at 1% THD, 154dB at 10%) and have a flat frequency response between 100 Hz to 10 kHz. Our field experiments show that even these special sensors are easily overloaded by AK-47 (7.62 mm) bullets if the trajectory is within 5 meters. We experimented with using rubberized paint and custom designed plastic caps on top of the capsules for protecting the microphone element from overload, but these efforts resulted in markedly distorted signal shapes even with safe (distant) trajectories.

The support board also contains a Bluetooth transceiver module (BlueGiga WT32), which provides a high-level UART based interface to the FPGA for communicating with Bluetooth enabled devices. This alternative wireless interface was not used in our experiments, but can easily replace the TelosB gateway in the architecture in future deployments.

The power source of the sensor assembly is a 2000 mAh Li-Ion battery, enabling continuous operation for approximately 6 hours. The current drain (typically 300 mA) depends significantly on the state of the GPS module (acquiring, tracking, sleeping), the state of the Bluetooth transceiver, and the communication burden on the ZigBit radio (packet transmission rate). The support board integrates a USB-based Li-Ion charger and step-up/step-down regulators for supplying the power rails needed by the Octopus board.

The twin board architecture, required interconnects, battery, antennae, and microphones (Knowles Acoustics VEK-H-30108)) needed to be packaged for supporting mobile field deployments. We used a rapid 3D printing process to design and build a custom ABS plastic enclosure. This allowed for maximum flexibility in arranging the microphones and other layout sensitive elements (connectors, antennae). The internal board assembly and the final packaged node are shown in Figure 5. Note that the existing Octopus board and the 2-board assembly drove the physical dimensions of the package. We will be able to cut each dimension in half in a future revision with a single newly designed board. This will result in a smaller microphone array, but in our estimation, it will have only a minor effect on AOA detection accuracy.

5. SENSOR FUSION

The sensor fusion computes the trajectory estimate and bearing to the shooter from the following inputs: a) a set of at least two shockwave detections, each including TDOAs and durations from four microphones; b) sensor positions and 3D orientations; c) speed of sound (assumed to be a known constant); d) microphone geometry; and e) assumed weapon parameters: a weapon specific parameter k that is used to map shockwave length to miss distance (see Equation 2), which includes the projectile muzzle speed and a deceleration parameter.

The sensor fusion algorithm consists of six steps:

1. Calculate the shockwave AOA for each sensor;
2. Compute the miss distances for each sensor;
3. Attempt to compute the speed of projectile;
4. Compute the trajectory estimates for each sensor;
5. Estimate trajectory by fusion of per-node trajectory estimates;
6. Compute bearing estimates.

Steps 1 and 2 are fairly trivial. First, the AOA of the shockwave at each sensor is computed from the TDOAs at the microphones, using the known microphone geometry and a given speed of sound. The AOA is represented as a 3-dimensional vector of unit length. Since this AOA is initially computed in the node's local coordinate system, we rotate this vector using the corresponding 3D compass readings to convert them to an absolute (East-North-Up) coordinate system. For simplicity, we convert the AOAs to 2D by discarding the z (Up) coordinates. Then, in Step 2, the miss distances are computed using Equation 2, with a predefined weapon-specific constant k.

If the projectile passes the sensors on one side, the AOAs reported by all the sensors are close to parallel. However, if there are sensors on both sides of the trajectory, the sensors on the two sides detect different AOAs. Their difference is directly related to the cone angle of the shockwave front, and, in turn, to the speed of the bullet over the sensor network. We check for this condition in Step 3, and compute the bullet speed as follows. Assuming β_1 and β_2 are the two distinct shockwave angles of arrival, such that $0 < (\beta_1 - \beta_2) < \pi$, the cone angle α is computed as $\pi - (\beta_1 - \beta_2)/2$. From here, the bullet speed estimate is computed as $v_{bullet} = c/sin(\alpha)$, where c is the speed of sound. If the bullet speed could not be computed this way—that is, for one-sided shots—we rely on the assumed weapon-specific muzzle speed, deceleration, and assumed shooter range to compute a rough bullet speed estimate.

If the trajectory passes between the sensors, the direction of the trajectory can also be computed as $2\pi - (\beta_1 + \beta_2)/2$. Note that the image trajectories for these sensors are completely different and only the real trajectory estimates align.

In Step 4, for each sensor, we compute the two trajectory estimates using the miss distance, shockwave AOA, and bullet speed values. Consider Figure 6. Notice that the trajectory is tangent to the circle whose center is the sensor

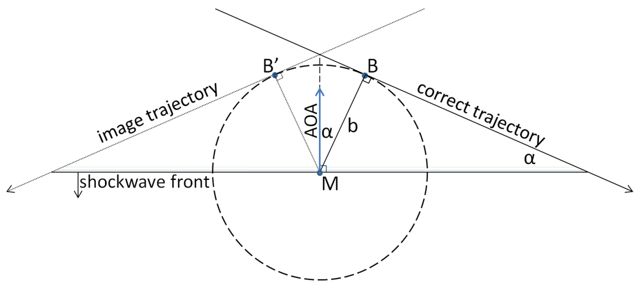


Figure 6: Computation of the two trajectory estimates.

position M and has a radius equaling the miss distance. The trajectory touches the circle at point B; therefore, the vector $\vec{MB}$ is normal to the trajectory. Also notice that the angle between the shockwave AOA vector and $\vec{MB}$ equals the cone angle α. From here, we can compute the position of point B by rotating the shockwave AOA vector clockwise by the angle α, and making its length equal to the miss distance. To get the direction of the trajectory, we need to rotate the vector by $\pi/2$ further clockwise. The trajectory solution is defined by point B and the trajectory's direction. Point B' and the second trajectory solution can be computed similarly, but with counterclockwise rotations.

If it was possible to compute the direction of the trajectory in Step 3, then the correct trajectory can be easily identified. We simply choose the one whose direction is close to the previously computed value.

If the trajectory angle is not available, we have to do some more work to eliminate the image trajectories. After completing Step 4 for all n number of sensors, we have $2n$ trajectory solutions available, of which n are incorrect.

We need to search for a consistent subset of trajectory solutions that *line up* with each other. Naturally, if a set of sensors detect the exact same shockwave direction of arrival, their respective true trajectory solutions will all be parallel. However, this is true for the image trajectory solutions, as well. This means that just by looking at the angles of the trajectory solutions it is not possible to identify the consistent subset. To further complicate things, we cannot assume that the trajectory solutions in a consistent subset are parallel: errors in the compass readings, as well as detection errors will result in slightly diverging trajectory solutions, which makes it hard to define a distance metric between them.

We observe that the even though the trajectory solutions in the correct subset may diverge far from the sensor field, they pass close to each other in the proximity of the sensors. This is not surprising, since, by construction, the the distance between a sensor M and point T on a corresponding trajectory is exactly the miss distance. Therefore, the metric we chose to score a subset of trajectories with similar angles is as follows. For all sensor-trajectory pairs in the subset, we compute the variance of the sensor-trajectory distances. The trajectory solution subset with the lowest variance is chosen as the correct one.

Once the correct subset of trajectory solutions is available, in Step 5 we compute the final trajectory solution by averaging the trajectories in the subset.

In general, two nodes are sufficient to identify the correct

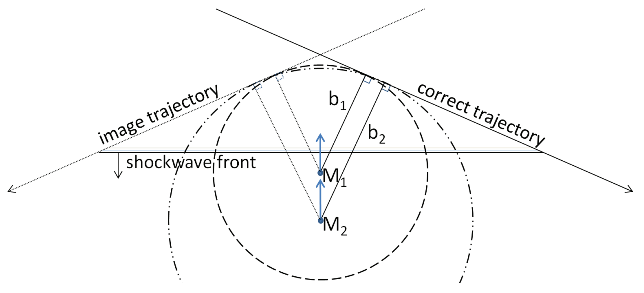


Figure 7: Degenerate geometry where it is not possible to distinguish between the true and the image trajectories.

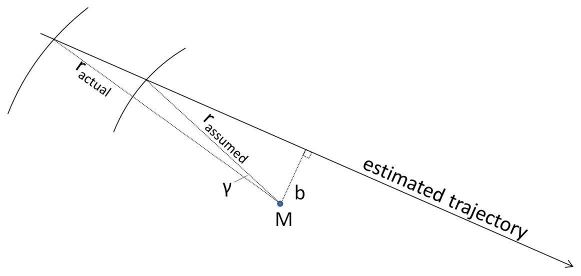


Figure 8: Bearing estimation. Although the actual range and the assumed range are quite different, the bearing estimation error γ stays low if the miss distance is much smaller than the true and the assumed ranges.

trajectory. However, there is a degenerate geometry where not just the two correct trajectory solutions line up with each other, but also the two image solutions. Consider Figure 7, where the shockwave front is perpendicular to the line defined by the two sensor locations and both sensors detect the same shockwave AOA. In this case, the miss distance estimates are not sufficient to differentiate between the correct and the image trajectory: the geometry is symmetric to the M_1M_2 axis. To avoid such situations, at least three non-collinear sensors are required for unambiguous trajectory estimation.

The bearing estimates are computed in Step 6 for all sensors that reported shockwave detections. For two-sided shots, we compute the approximate range of the shooter using the previously estimated bullet speed and the weapon-specific parameters (speed at muzzle and deceleration). Using this approximate range, we find the estimated position of the shooter on the trajectory.

For one-sided shots, where the range cannot be approximated, we use a predefined value instead. Using the trajectory estimate and this *assumed* range, we find the estimated position S of the shooter on the trajectory. The angle of the $\vec{MS}$ vector is reported as the bearing estimate for sensor M.

It is important to note that, for one-sided shots, we do not have enough information to compute where exactly the shooter is on the line of trajectory. The assumed shooter-sensor distance is just an assumption that corresponds to a typical case identified by the user requirements. How can the bearing estimation still be accurate then? Clearly, if the true range is close to the assumed one, the estimated bearing will be close to the true one. More importantly, the bearing estimation error will still be low if the miss distance is much smaller than the true range and the assumed range (see Figure 8 as an example). Considering that the maximum shockwave detection range of our sensors are about 30 m (but the important shots are much closer) and the typical range that the users are interested in is 100-200 m, this assumption holds.

This leaves us with only one bad case: when the shooter is very close to the sensor, the reported bearing may have a large error. From a practical point of view, this is not a critical issue. When the shooter is that close to the sensors, the users can identify the source location without any technology and have no time to use their smartphones in any case.

6. IMPLEMENTATION

The software that enables networked trajectory and bearing estimation consists of three distinct components: a) the microcontroller code running on the ZigBit module; b) the Android code that collects detections from the network, dispatches the sensor fusion algorithm, and displays the result; and c) the sensor fusion algorithm that computes the bearing to the shooter from a set of shockwave detections.

6.1 MCU implementation

The ZigBit microcontroller is programmed in TinyOS 2.1. It handles mesh networking, location awareness and communication with the FPGA subsystem. For positioning, we rely on the u-blox GPS module, as well as on the digital compass. When the FPGA subsystem signals through an interrupt line that a shockwave detection is available, the shockwave time of arrival (TOA) and shockwave duration measurements for each microphone, as well as the compass heading, are placed in a radio packet and broadcast to the network.

For mesh networking, we rely on the Directed Flood Routing Framework (DFRF) [10], a policy-driven, configurable routing middleware, to achieve restricted network-wide broadcast. The routing framework is configured as follows. Each packet has a unique sequence number and a hop count field. The hop count field specifies how many times a particular packet can be forwarded, and is initially set to 10. On reception of a new packet, it is placed in the node's routing buffer, its hop count value is decremented and is rebroadcast. After forwarding the packet, it is kept in the routing buffer for a while. This way, the node can remember the recently forwarded packets, so that subsequent receptions of the same detection packet can be ignored. This mesh networking configuration ensures that all detections are available at every node of a fairly small mobile sensor network, and saves bandwidth by avoiding unnecessary rebroadcast of previously forwarded detections.

6.2 Android implementation

The Android code running on the Dell Streak is responsible for collecting shockwave detections from the sensor network, dispatching the sensor fusion algorithm, and displaying the result. Since Android is built on top of a Linux system and is programmed in Java, it was straightforward to port the TinyOS Java SDK to Android. This way, we can

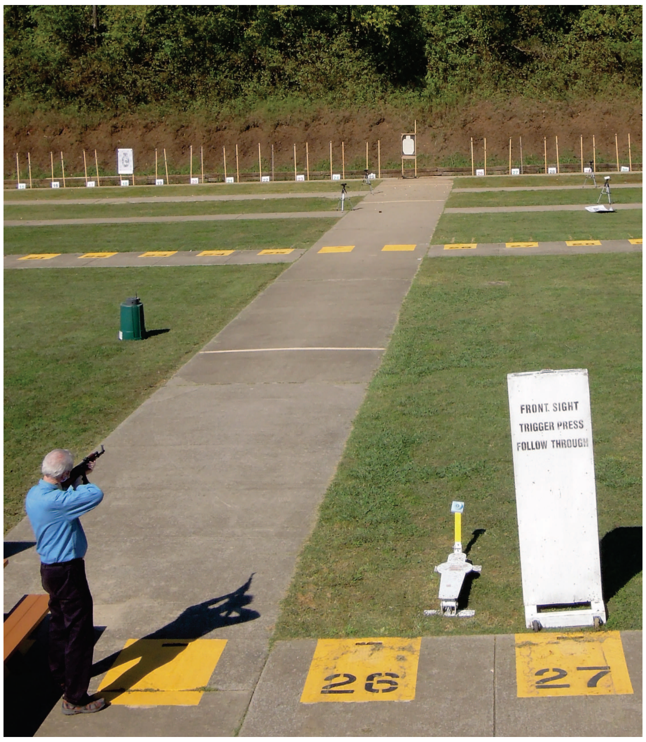


Figure 9: Shooting range with the four sensors on tripods in the distance. The green bucket in the foreground holds a repeater, as the TelosB we used for our remote user interface had trouble picking up the messages from the distant sensors.

use the TelosB mote, connected to the Streak directly via USB, as a TinyOS base station that collects the shockwave detections coming from the sensor network.

When at least two detections of the same shot are available (and a timeout comparable to the routing delay has passed since receiving the last detection), the sensor fusion algorithm is invoked with the detection set as its input data. The computed trajectory estimate, as well as the bearing estimate to the shooter is displayed on-screen.

6.3 Sensor fusion implementation

The sensor fusion is implemented in MATLAB. We rely on MATLAB Coder to generate C code from the MATLAB source files. SWIG is used to generate JNI wrappers around the generated C code so that it can be compiled to an ARM Linux shared object file that can be loaded and used by the Dalvik virtual machine running the Android code. The sensor fusion procedure takes less than $100msec$ on the smartphone.

7. EVALUATION

We built four prototype sensor nodes and evaluated the system on a shooting range. The four sensors were placed approximately at the corners of a $6x8m$ rectangle as shown in Figure 9. Originally, the four nodes had the same orientation, but they were changed randomly during the test multiple times. We took 44 shots with an AK-47. Most of the shots were approximately $35-40m$ from the closest sensor. 12 shots were taken from farther away at about $80m$

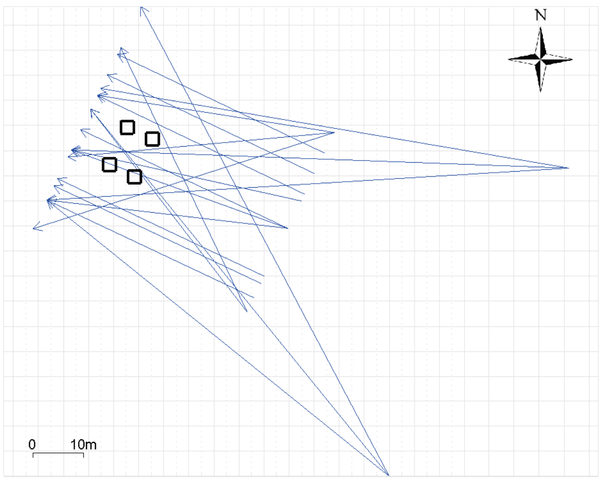


Figure 10: Map of sensor positions (squares) and shot trajectories (arrows).

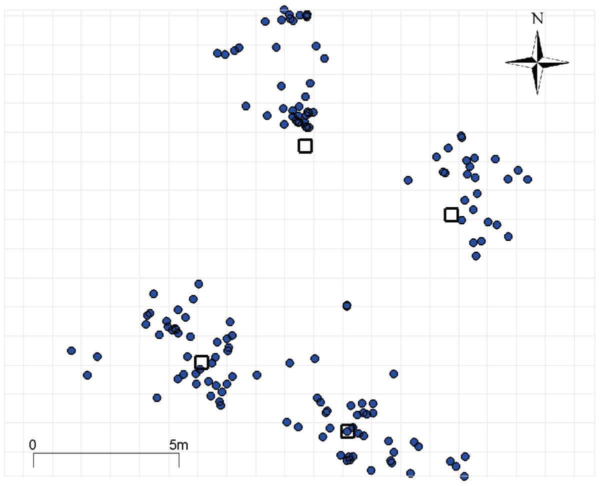
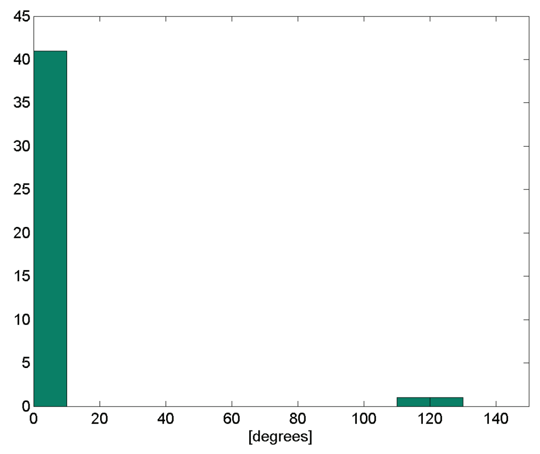


Figure 11: Map of reported GPS positions (dots) vs. the groundtruth (squares).

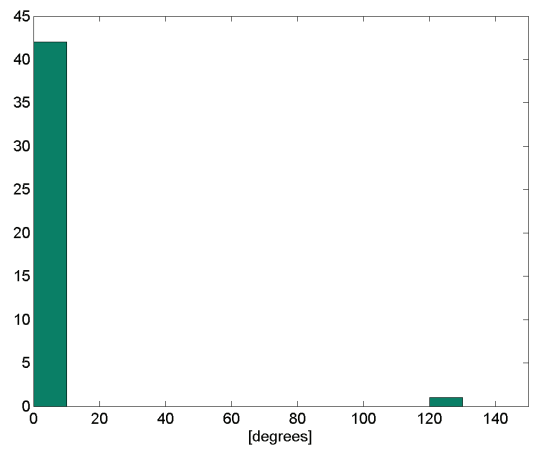
from the sensors. While the system is designed for longer range shots, we were constrained by the limitations of this particular shooting range.

We supplied the sensor fusion algorithm with the following parameters. We assumed that the weapon is an AK-47 and that the range is $150m$. Accordingly, the bullet speed at the muzzle was set to $720m/s$, and the projectile deceleration was assumed to be $-555m/s^2$. The weapon-specific coefficient k was set to 0.00625. The speed of sound was set to $335m/s$.

Figure 10 shows all 44 shots relative to the four sensor positions (note that we typically took 2 shots from the same position at the same target). Each arrow represents at least one shot starting at the shooter location and indicating the target with an arrowhead. As you can see, less than one third of the shots pass in-between any sensors. This is intentional; we wanted to push the limits of the technology since estimating the trajectory is much more difficult when only one side of the shockwave is detected.

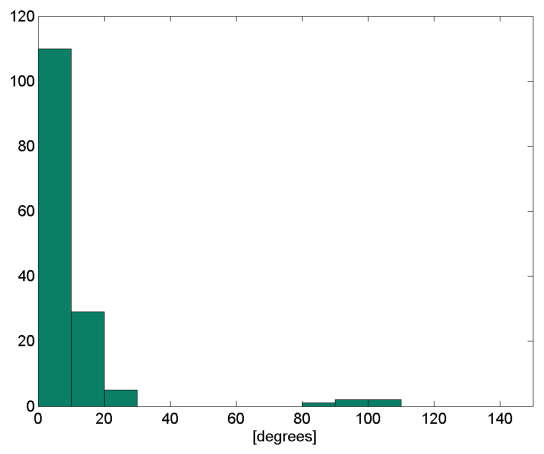


(a)

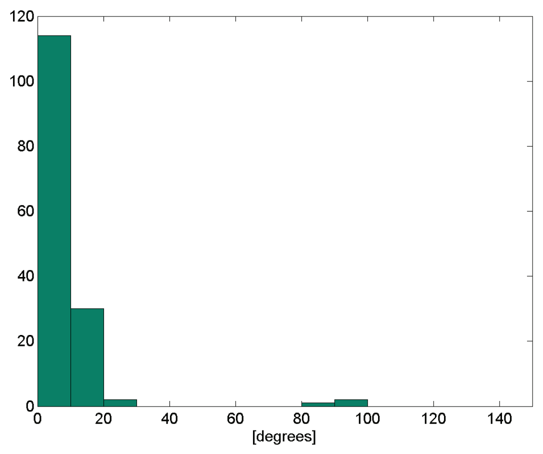


(b)

Figure 12: Histogram of trajectory angle errors in degrees derived from measurements from all sensor detections when a) the GPS-reported locations and b) the manually surveyed locations are used.



(a)



(b)

Figure 13: Histogram of bearing errors in degrees derived from measurements from all sensor detections when a) the GPS-reported locations and b) the manually surveyed locations are used.

7.1 Sensor positioning errors

Unfortunately, our sensor positioning approach displayed much larger errors than anticipated. We had used the given GPS receiver on a couple of different designs before and experienced only $1-2m$ errors in stationary setups under relatively benign environments. This is in line with the datasheet; however, during our field test, the GPS on this design behaved somewhat erratically showing up to $5m$ errors. Figure 11 shows the ground truth sensor positions (small squares) and the GPS reported locations (dots) corresponding to the times of the 44 shots. The average GPS error was $2.5m$ which is significantly worse than expected. We suspect that the new sensor node has some kind of noise source that proved to be too close to the GPS antenna.

Similarly, the compass was affected by the sensor node hardware as described in Section 4. We manually calibrated each compass according to the procedure described in the datasheet. This improved the accuracy significantly. Still, turning the sensor nodes $360°$ and comparing the reported orientation with the ground truth at multiple angles resulted in up to $8°$ of error. Again, this is much worse than the expected $1°$ error.

Previous WSN-based shooter localization systems used many sensors [14, 17]. Such systems are less sensitive to sensor location errors because the many random positioning errors have a tendency to average out. (Also, these systems were evaluated using surveyed positions.) Here we only have two or three sensors, yet sensor location errors have a small impact as we'll see.

7.2 All sensors

One of the sensors had a very low detection rate compared to the others. We suspect some kind of hardware issue with the acoustic channels on that particular node. Thus, about half of the shots were detected by three or fewer sensors. One shot was picked up by a single sensor only, so the following evaluation is based on 43 shots.

First, we use all available sensors to estimate the trajectory. The trajectory angle errors are shown in Figure 12(a). Two shots were incorrectly localized: the disambiguation algorithm picked the wrong trajectory from the possible two. All of the remaining 41 shots had trajectory angle errors of less than 10° with the average error of 3.1°. To evaluate the effect of the relatively large sensor position errors, we reran the localization algorithm for all the shots using the ground truth sensor positions. Not surprisingly, only a single trajectory was incorrectly localized in this case (see Figure 12(b)). This was caused by multiple factors. The AOA angle estimates had a larger than usual error on one node (either due to detection error or the compass). The geometry of the other two nodes were close to the degenerate case. That means that effectively, we only had the minimally required two measurements and they did not agree on the shockwave angle. Interestingly, the extra shot whose trajectory was selected correctly using the precise sensor locations increased the average trajectory angle error to 3.4°.

When you plot the trajectory using a map as the user interface, it is the trajectory accuracy that matters. However, the primary user interface for soldiers is a simple visual and/or audio indication of the bearing to the shooter. This could be as simple as a recorded audio played in the headset: "Shooter at 5 o'clock." In this case, it is the bearing to the shooter from the sensor node that matters. With this kind of feedback, the resolution of the bearing effectively becomes 30°. That is, a bearing error of ±15° is still considered perfect.

Fixing the shooter range at 150m (even though the real range varied between 35 and 85m only) and computing the individual bearing estimates from each sensor's point of view for the 41 shots that were correctly localized results in an average bearing error of 7°. The histogram of bearing estimation errors is shown in Figure 13(a). This result is much better than we anticipated considering that it includes all error sources: sensor position and orientation errors and the trajectory estimation errors (angle and displacement). If we remove just one source and use the ground truth sensor positions, the average error becomes 6.5° and the histogram of bearing estimation errors (Figure 13(b).) shows only a slight improvement. This means that the positioning errors of a consumer-grade GPS receiver does not contribute heavily to the overall bearing estimation error, which is affected predominantly by the fixed range assumption and the errors already present in the trajectory estimation. Also, this finding supports our hardware choice of the u-blox LEA-6T, a consumer-grade GPS receiver for sensor node localization.

These results indicate that the system is surprisingly insensitive to sensor position errors. Their main effect is potentially throwing off the disambiguation algorithm, but the trajectory angle and bearing errors if the correct trajectory option is picked remain very low. Why is that? The individual trajectory angle estimates do not depend on the sensor position at all. The wrong position shifts the trajectory, but it does not rotate it. If the correct trajectory option is selected by the disambiguation algorithm, it averages the two or more individual estimates. Its angle might have a few degrees of error and it may be shifted by a few meters. But the shooter is far away compared to the potential trajectory displacement, so its effect is minor. This is a huge advantage of not relying on shockwave TDOAs across sensors where sensor location errors have a significant impact!

7.3 Sensor pairs

As we have shown before, the minimum number of sensors required for trajectory estimation in the general case is 2. Therefore, we took the 43 shots that were detected by at least 2 sensors and selected all possible sensor pairs for each shot and ran the trajectory and bearing estimation algorithm on the resulting 193 cases. There were 5 cases where the measurements proved inconsistent and the algorithm did not return a solution. The system picked the right trajectory for 164 cases (87% of remaining 188 shots). The average trajectory estimation error for these were 3.3°. Of the resulting 376 individual bearing estimates 307 had under 15° error. The mean error of the bearing estimates when the right trajectory was picked came to 5.8°. Figure 14 shows the histogram of bearing errors including the cases when the wrong trajectory was selected.

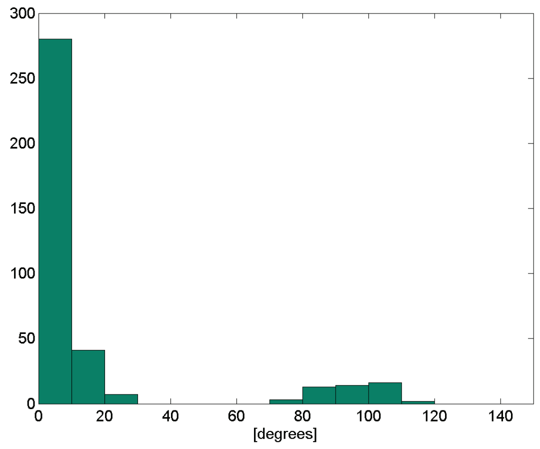


Figure 14: Histogram of bearing errors in degrees derived from measurements from pairs of sensors. The bars above 70° correspond to cases when the wrong trajectory was selected. The figure illustrates the worst possible case for the system since it only uses the absolute minimum number of sensors, the majority of the cases are one-sided shots, it includes all cases with degenerate geometry, and suffers from relatively large sensor localization and orientation errors.

If we use the correct sensor positions, the number of correctly selected trajectories improves significantly: while the same 188 out of 193 cases were localized, 175 (11 more) were correctly estimated. Most of the remaining 13 incorrect ones were close to the degenerate geometry. The trajectory angle and bearing accuracies were basically the same as the case using the GPS supplied locations.

7.4 Shot library

We utilized the same shot library from Aberdeen Proving Grounds as in [17]. It contains 33 AK-47 shots measured using ten sensors. The sensor field was approximately 30x30m and the shots were taken from ranges between 50 and 300m.

The sensor nodes were placed on surveyed locations and had known orientation. The sensor data includes the shockwave length and AOA that our system needs. Since the setup used many sensors and our objective is to use as few as possible, we first looked at all possible pairs of sensors per shot and compared it to all possible sensor triplets per shot. The former resulted in over 1300 data points, while the latter came to well over 3000 cases.

For sensor pairs, only 87% of the cases were localized. Out of these, the system identified the trajectories correctly 94% of the time. These are somewhat lower rates than we expected considering that the sensor positions and orientation were precise. There are two reasons for this. First, the shooter range and consequently, the speed of the projectile near the sensors varied significantly, hence, the fixed projectile speed assumption for one-sided shots had a more significant impact. Second, there were many cases when the bullet passed directly over the sensors. Our current 2D solution is unable to handle such cases well (however, it will not be difficult to generalize the solution to 3D).

The mean bearing error of the correctly localized shots from each individual sensor's perspective was 4.6°. As expected, using three sensors instead of two improves the situation. About 99% of the 3348 cases were detected, that is, the system reported a trajectory and bearing estimate. The correct trajectory was selected 91% of the time. In other words, 3002 cases were correctly localized. The mean trajectory angle error for these cases came to 3.3°, while the corresponding bearing error was 4.1°. Figure 15 shows the histogram of bearing errors for sensor pairs and triplets including those corresponding to image trajectories.

7.5 Summary

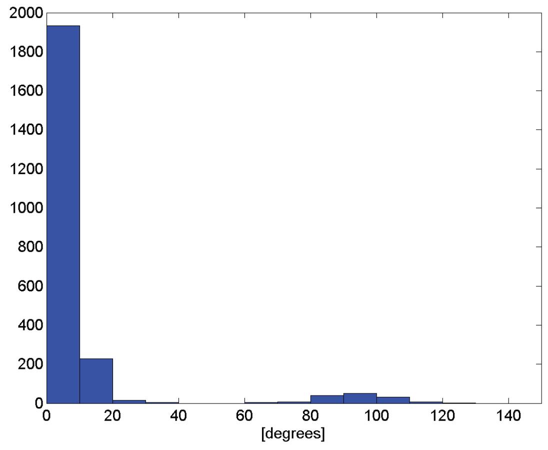
The evaluation of the system allows us to draw some clear conclusions. The most important lesson is that we need to improve the system's ability to resolve the trajectory ambiguity problem for one-sided shots. When the bullet passed in-between two sensors, not a single trajectory was missed. It is the easy case that most other systems have focused on. Sensor location and orientation errors have an impact, so improvement in those areas will help. But even with no measurement errors of any kind, in the case of a degenerate geometry as described in Section 5, no system can find the correct trajectory. To lessen the probability of this bad case, at least three sensors should be used.

If and when the system identifies the correct trajectory, the bearing estimation is remarkably accurate. The errors remained under 7° in all the setups we tested. This is better than what is required in most deployment scenarios. Another distinguishing characteristic of the system as opposed to TDOA-based methods is that this error is very insensitive to sensor location errors.

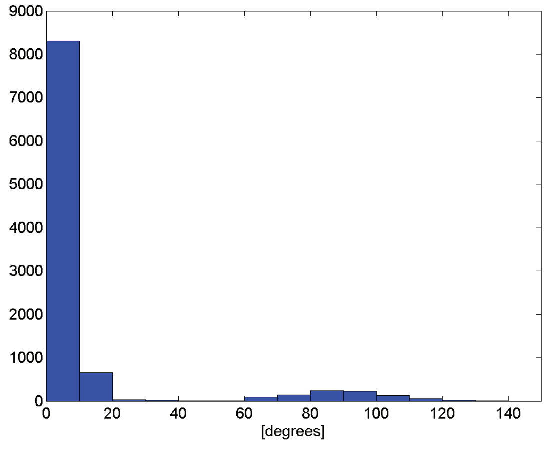
8. CONCLUSIONS

The presented paper has multiple novel contributions. To the best of our knowledge, it is the first shooter localization system that relies on shockwave length and shockwave AOA only. In contrast to the state-of-the-art, it does not use the TDOA of the shockwaves across multiple sensors. As such, it does not require time synchronization of the sensors. Instead, it utilizes the shockwave length and AOA at each sensor to identify the two possible trajectories in the horizontal plane of the nodes and use one or more nearby sensors to disambiguate the two. Relying on shockwaves only means that the sensors do not even need to have line of sight to the shooter as the muzzle blast is not used. An additional attribute is that the system is more responsive as it does not have to wait for the muzzle blast which may arrive a full second after the shockwave for longer range shots. Moreover, it is the first reported networked system that is truly distributed. It does not require a central high-performance computer to run the sensor fusion and it still has lower latency. Each and every node calculates the solution from the information available from the network. Finally, this is the first reported implementation and field evaluation of a smartphone-based shooter localization system.

There are three tradeoffs for not utilizing muzzle blasts. First, the system assumes a known weapon type. We have focused on AK-47s, the most widely used rifle in the world. Second, the system is unable to estimate the range to the shooter for one-sided shots. However, bearing information is far more valuable than range as the latter can be usually



(a)



(b)

Figure 15: Histogram of bearing errors in degrees derived from measurements from all sensor pairs (a) and triplets (b) in the shot library.

determined by the trained eye based on the terrain given an accurate bearing estimate. In contrast, humans are not very good at determining the direction to the source of a shot especially in urban or mountainous terrain. Finally, the system becomes susceptible to the trajectory ambiguity problem for one-sided shots when only two sensors are available. However, our system is the first that we are aware of that specifically addresses one-sided shots using shockwaves only and these initial results are highly encouraging.

Our future work will include testing the system with various different weapons to try to relax the known weapon assumption. We will also generalize the trajectory estimation to 3D. Finally, we will investigate using the shockwave TDOAs for eliminating the errors in the disambiguation of the candidate trajectories.

9. REFERENCES

[1] J. Ash, G. Whipps, and R. Kozick. Performance of shockwave-based shooter localization under model misspecification. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 2694 –2697, march 2010.

[2] U.S. Patent 7,292,501 B2: Compact Shooter Localization System and Method, Nov. 2007.

[3] Raytheon BBN technologies, boomerang website. http://www.bbn.com/products_and_services/boomerang/.

[4] Cypress Semiconductor. Cypress Programmable System-on-Chip. http://www.cypress.com/?id=1353.

[5] T. Damarla, L. Kaplan, and G. Whipps. Sniper localization using acoustic asynchronous sensors. *Sensors Journal, IEEE*, 10(9):1469 –1478, Sept. 2010.

[6] U.S. Patent 6,178,141 B1: Acoustic counter-sniper system, Jan. 2001.

[7] A. Lédeczi, T. Hay, P. Völgyesi, R. Hay, A. Nádas, and S. Jayaraman. Wireless Acoustic Emission Sensor Network for Structural Monitoring. *IEEE Sensors Journal*, 2009.

[8] A. Lédeczi, A. Nádas, P. Völgyesi, G. Balogh, B. Kusý, J. Sallai, G. Pap, S. Dóra, K. Molnár, M. Maróti, and G. Simon. Countersniper system for urban warfare. *ACM Transactions on Sensor Networks*, 1(1):153–177, Nov. 2005.

[9] D. Lindgren, O. Wilsson, F. Gustafsson, and H. Habberstad. Shooter localization in wireless microphone networks. *EURASIP Journal on Advances in Signal Processing*, 2010, 2010.

[10] M. Maróti. Directed flood-routing framework for wireless sensor networks. In *In Proc. of the 5th ACM/IFIP/USENIX International Conference on Middleware*, pages 99–114, New York, NY, USA, 2004. Springer-Verlag New York, Inc.

[11] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 364 – 369, april 2005.

[12] J. Sallai, A. Lédeczi, and P. Völgyesi. Acoustic shooter localization with a minimal number of single-channel wireless sensor nodes. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, SenSys '11, pages 96–107, New York, NY, USA, 2011. ACM.

[13] ShotSpotter website. http://www.shotspotter.com/products/military.html.

[14] G. Simon, M. Maróti, A. Lédeczi, G. Balogh, B. Kusý, A. Nádas, G. Pap, J. Sallai, and K. Frampton. Sensor network-based countersniper system. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, SenSys, New York, NY, USA, 2004. ACM.

[15] S. Szilvási, B. Babják, A. Lédeczi, and P. Völgyesi. Towards a versatile wireless platform for low-power applications. *International Journal of Digital Information and Wireless Communications*, 1(2), 2012.

[16] P. Volgyesi. Octopus: Wireless sensor for multichannel acoustic sensing - hardware description. *Technical Report (ISIS-10-103), http://www.isis.vanderbilt.edu*, Dec. 2010.

[17] P. Volgyesi, G. Balogh, A. Nadas, C. B. Nash, and A. Ledeczi. Shooter localization and weapon classification with soldier-wearable networked sensors. In *Proceedings of the 5th international conference on Mobile systems, applications and services*, MobiSys '07, pages 113–126, New York, NY, USA, 2007. ACM.

[18] G. Whitham. Flow pattern of a supersonic projectile. *Communications on pure and applied mathematics*, 5(3):301, 1952.

[19] J. Williams. A data distribution service for mobile devices. Master's thesis, Vanderbilt University, 2011.