FREETARGET

Theory of Operation

SUMMARY

This document describes the circuit and algorithm used to create a low cost electronic target for general use.

APPLICABLE DOCUMENTS

This section contains documents that are relevant to the understanding of this project.

Web Sites

Law of Cosines

https://www.mathsisfun.com/algebra/trig-cosine-law.html

2.2. Technical References

2.3. Software

2.4. Glossary

Term	Description
D	Distance between opposite sensors. Must be at least sqrt(2) x Target diameter, typically 230 mm for 155mm pistol target
Report Time	Time in microseconds between the pellet hitting the paper and received by the sensors
Sensor	One of four sensors located at the cardinal points.
V	Speed of sound in air, 0.34mm / us

INTRODUCTION

Electronic targets that capture a shot and display the results have been around for a number of years. There are no shortage of suppliers, but

unfortunately, the cost is outside of the price range of most people.

Requirements

The requirements of this project are:

1. Uses a conventional lap top or PC for the display device

2. Signal processing is handled off-board on a low cost processor

3. Material cost to be less than $100

4. Open source to allow the project to evolve in time.

OPERATION

The fundamental problem to be solved is to detect the location of a pellet or bullet as it passes through a prescribed area. Once that is done, the

presentation of the information is a relatively straight forward if not detailed matter.

There are a number of ways of doing this:

High speed video recording Measurement through a light gate Acoustic location

Acoustic location has been used for many years and the technology is readily accessible to most developers. For this reason, acoustic location is used in this project.

Overview

The eTarget uses four acoustic sensors (microphones) arranged around the target as shown in Figure 1. The pellet moving over the sensors will emit a sound wave in a circular pattern that is picked up by the sensors. The sound radiates outwards uniformly at the speed of sound, So the time values of n, e, s, and w are proportional to the distance, and if the values are known, then we can compute the position of the pellet.

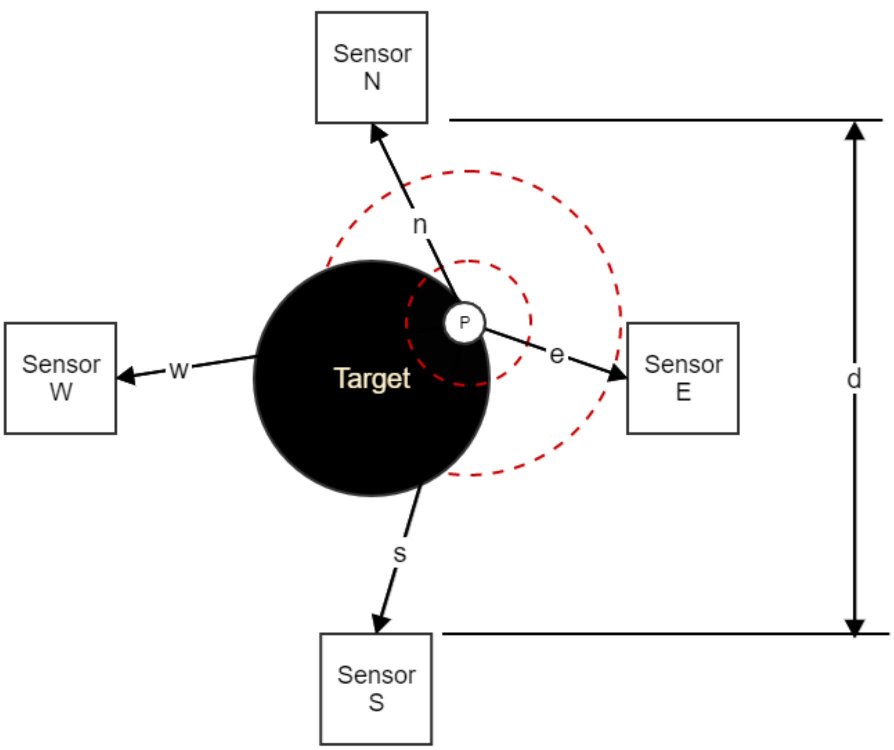

Figure 1: Acoustic Sensor Arrangement

Calculations

Using Figure 1 as a starting point, the relevant physical values are applicable.

For an air pistol or rifle target, the distance between sensors is 230mm

Position resolution to identify a hit to within 0.1 point (ex 10.7 Vs 10.8) is Resolution = space between rings / 10

= 2.5mm / 10

= 0.25mm or better
Longest report time occurs if the pellet passes immediately in front of a sensor

Time = distance between sensors / speed of sound = 100 mm / 343 mm/ms

= 291 us
Time resolution to locate the pellet is:

Time resolution = distance resolution / speed of sound = 0.25mm / 323mm/m

= 0.00072 ms
= 0.7 us per 0.1 point

Circuit

Many implementations found on the internet use the analog conversion circuit of an Arduino or Raspberry Pi to measure the time between sensors. From the calculation above, this circuit is nowhere near fast enough or precise enough for the task at hand. The solution has to be to do the acquisition in hardware and the calculations in software.

Requirements

The time acquisition circuit for this project needs to meet the following requirements: Measure the sound from four sensors

Measure the sound to better than 0.7 us. or better
For the purposes of our design, an 8MHz or 0.125us sample will be assumed

Measure the sound for up to 291 us.
For the purposes of our design, 350us will be assumed

Number of timer bits = log2(Maximum Time / Time Resolution). = log2(350 us / 0.125 us)

= 12 bits or better

Block Diagram

The block diagram for the data acquisition circuit is shown in Figure 2

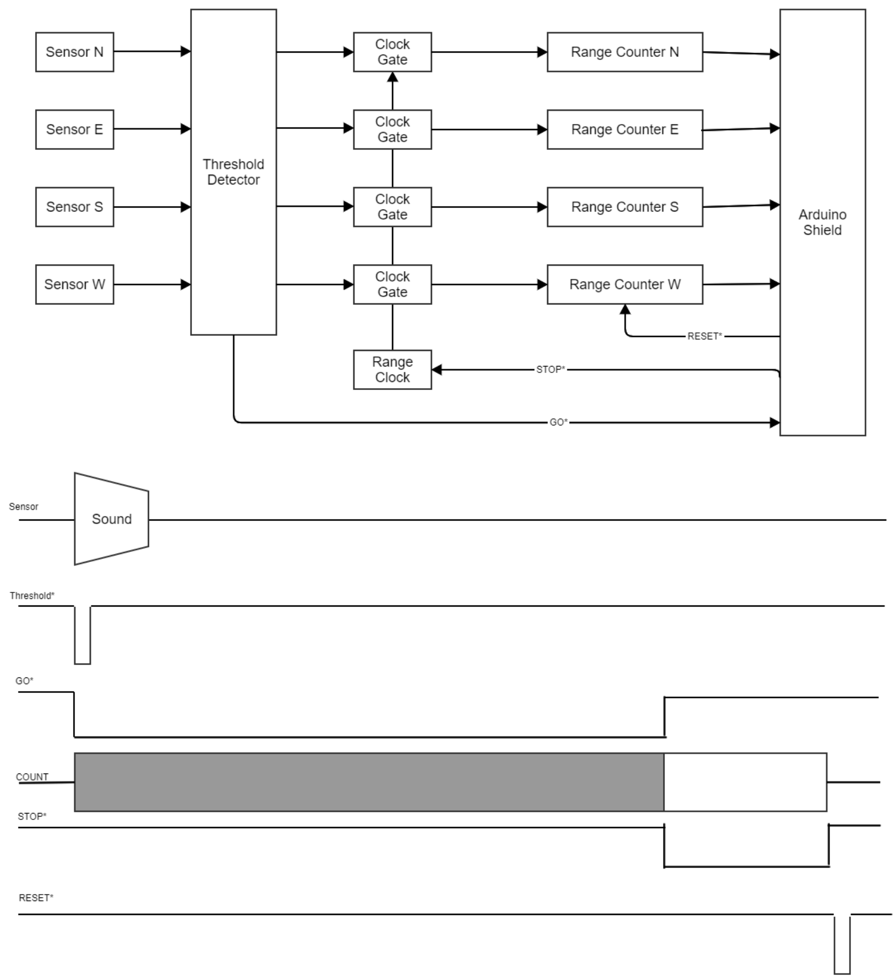

Figure 2: Acquisition Circuit Block Diagram

As the projectile passes over the sensor, the sound wave will generate a wave from the sensor that when it exceeds a threshold will generate a pulse which is the signal used to start GO* to begin driving the counters. A short time later, the Arduino generates STOP* to hold the count at their current values. The software then reads the values out, does it's calculations, and pulses RESET* to clear the counters and repeat the process.

SHOT LOCATION

Starting with Figure 1, the geometry of the shot detection is shown in Figures 3 and 4. The distances N, E, S, and W are the distances between the projectile and the sensors. The distance K is the shortest distance between the sides of the sensors.

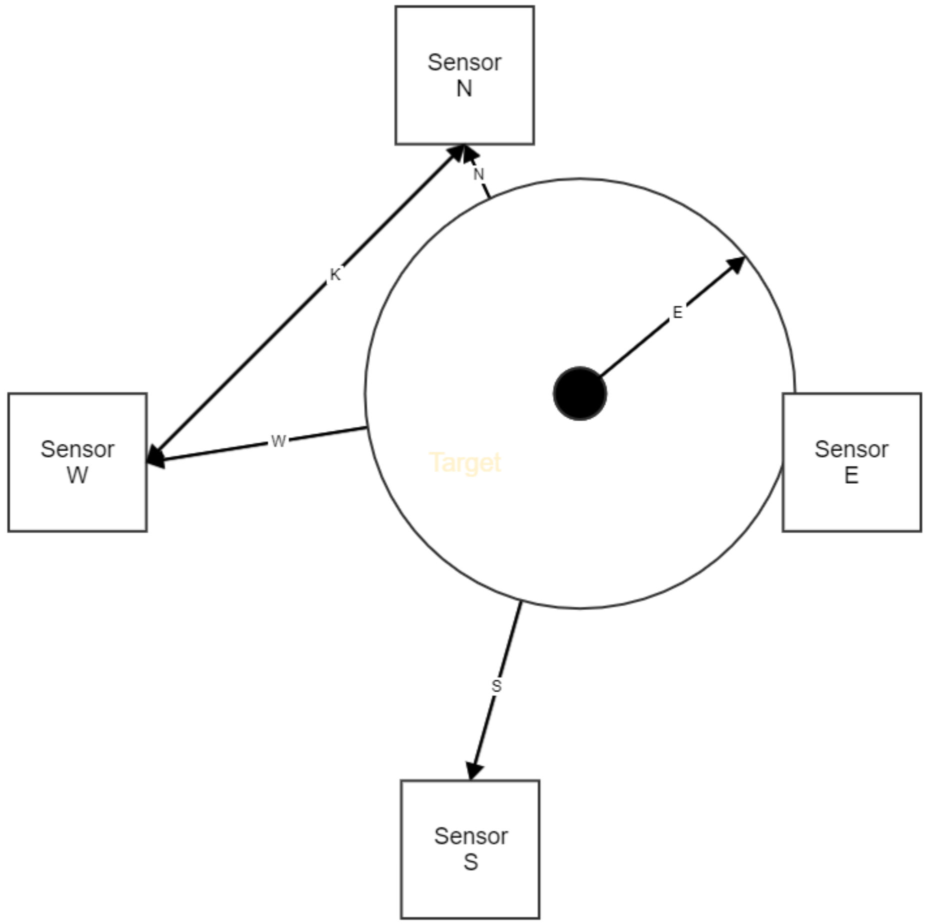

Figure 3: Shot Geometry, Sound Propagation

Sound travels out uni-formally from the projectile (distance e) until it reaches Sensor E and begins the Sensor e Timer. Subsequently, Sensors N, S, and see the sound wave and begin timing. When the timers are stopped, and the largest counter (Sensor E) is subtracted from the other counters, we are left with distances n, s, and w. Since the only unknown in this system is the distance e, our problem simplifies to finding the value of e that when added to n, s, and w gives us the distances N to W and the location of the impact.

n = E -N

s = E -S

w = E- W

If we guess the value of e and insert it into the geometry of Figure 4 we get the four triangles shown in Figure 5. The base of each triangle is the known distance K, and the sides are the known counter values summed with the estimate for E

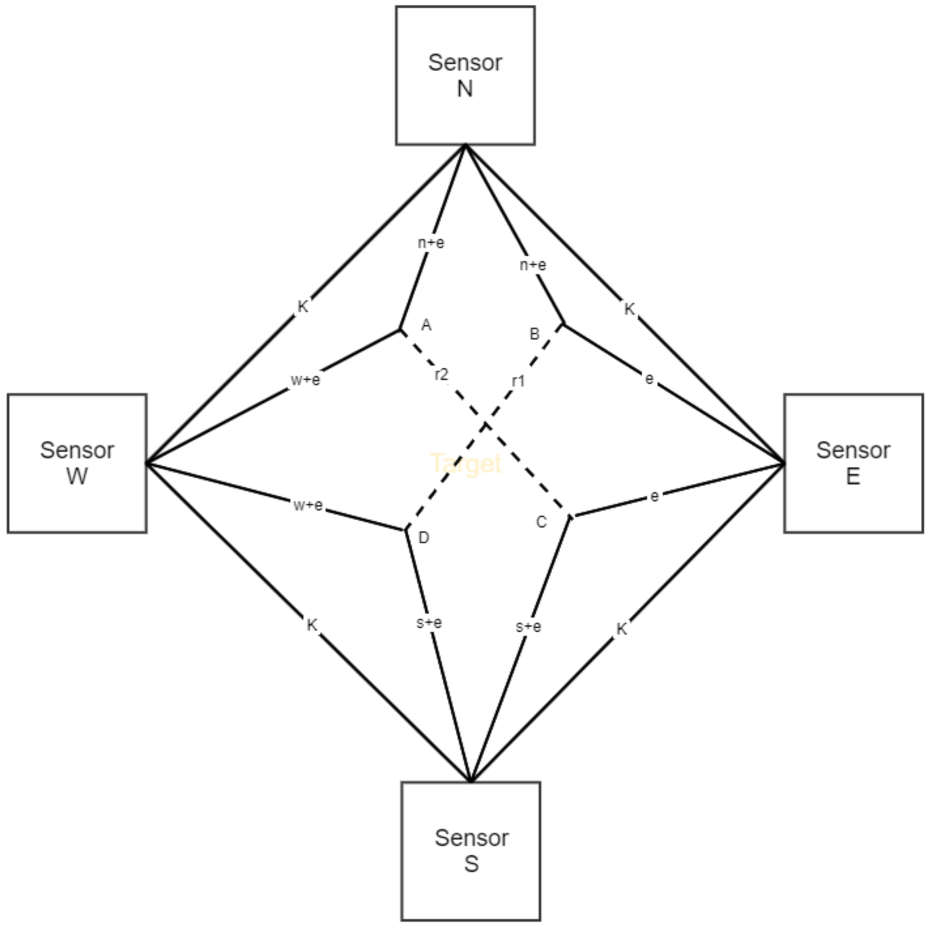

Figure 4: Shot Geometry, Triangle Intersection.

We can find the locations A, B, C, and D using trigonometry. For the purposes of illustration, triangle K-(s+e)-e is rotated 45 degrees and shown in Figure 5.

Figure 5-4: Intersection Triangle, Law of Cosines

From the Law of Cosines,

$$\mathbf{e^2 = (s+e)^2 + K^2 - 2(s+e)K \cos (P)}$$

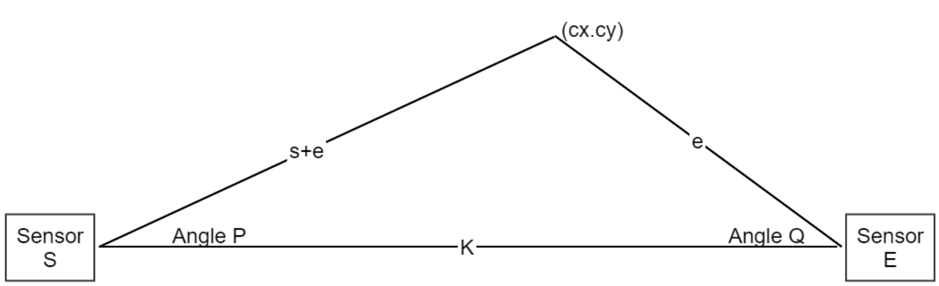

Figure 5: Rotated Triangle

Re arranging terms

$$P = \frac{\arccos((e^2 - (s+e)^2 - K^2)}{-2(s+e)K}$$

By trigonometry

cx = (s+e) cos(P)
cy = (s+e) sin(P)

Rotating theta back into the original circle.

Cx = (s+e) cos(P + 45')
Cy = (s+e) sin(P + 45')

This process is repeated three more times for the pairs N-E N-W and S-W, yielding locations: (Nx, Ny) (Ex, Ey) (Sx, Sy) (Wx, Wy)

The new estimated location for the shot becomes

$X_{estimate}$ = (Nx + Ex + Sx + Wx) / 4
$Y_{estimate}$ = (Ny + Ey + Sy + Wy) / 4

This will provide a new estimate for the value of e (e_n) which is compared to the previous e_{n-1} Once the difference between the estimates is sufficiently small, the location is solved.

Simplfied

```
while ( |(en – en-1)| > threshold )
 {
   Compute locations (xn,yn) (xe, ye), (xs, ys), (xw, yw) based on the timers and the estimate (e)
   Compute the new location Xestimate + Yestimate
   Use the computed location to improve the value of e
   Compute the new estimate for this cycle
}
```

Z_OFFSET

The algorithm shown above was revised in July 2021 to include the effect of the distance between the paper and sensor planes. Figure 3 assumes that the paper and sensors lie in the same plane so that the sound travels the same distance as the pellet when viewed from the front. In reality, the paper and sensors are displaced by a distance Z_OFFSET as shown in a side view of the target in Figure 6.

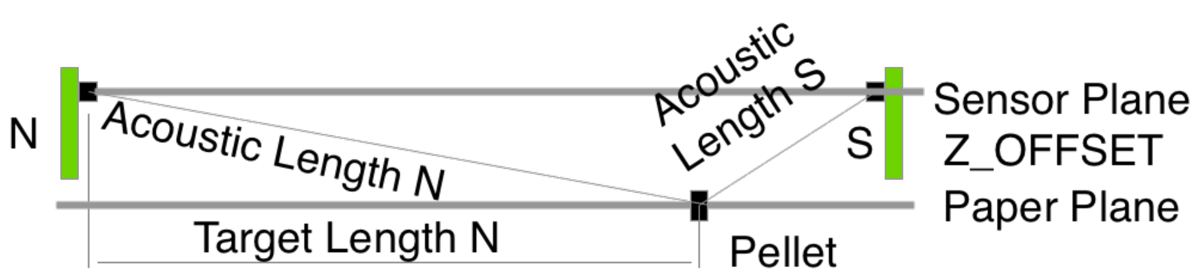

Figure 6: Z_OFFSET

The acoustic length is the hypotenuse of the Target Length and the Z_OFFSET, and is what is measured by the timers.

If the pellet strikes the target in the centre (10.9) then the acoustic length on all of the paths is the same and the effect of the Z_OFFSET is cancelled out and the computed position is correct.

If on the other hand the pellet strikes in the 1 ring, as shown int Figure 6, the difference between the Target Length N and Acoustic Length N is fairly small. On the other hand, Acoustic Length N and Target Length S can be fairly substantial. This will skew the computed shot towards the North sensor.

The solution is to subtract out the Z offset from the measured time.

From Geometry 101

Acoustic Length = sqrt(Target Length **2 + Z_OFFSET ** 2)

Rearranging terms the target length used in the calculations becomes

Target Length = sqrt(Acoustic Length ** 2 – Z_OFFSET ** 2)

Thus on each iteration of the successive approximation, the new target length is recomputed.