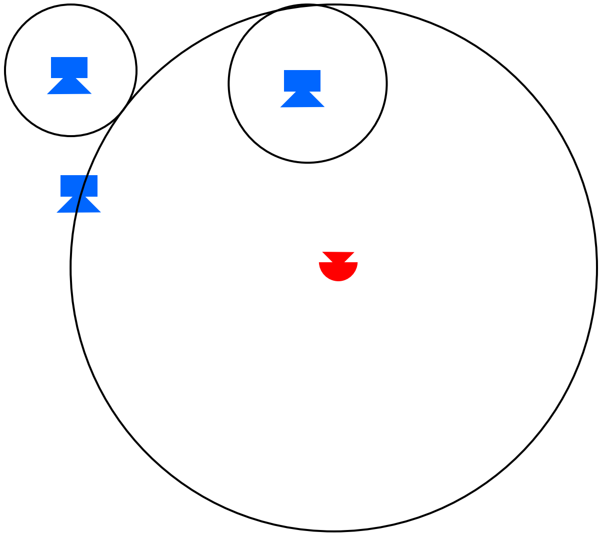
Robot Location
Using Three Omnidirectional
Sound Beacons



John Swindle

swindle@compuserve.com

September 2010

rev e

Three Beacons – Who Cares?

- Mini-DARPA challenge in back yard. (See the DPRG email archive.)
- Beacons like GPS, but with far better precision and stability for use in smaller spaces

Three Beacons – Proposed Solutions

From the DPRG listserver, several things were proposed:

- Infrared
- Lasers
- Radio
- Many hobbyist sonars working together
- Rotating any of the above stuff with servos
- Three omnidirectional audio sources and a microphone
- Two omnidirectional audio sources and a microphone

(Guess who came up with the last two proposals.)

The solution shown here uses three cheap speakers
and one cheap microphone with ***no sync***.

No motors, no servos, nothing that points anything at anything.
(It’s just wrong, isn’t it? Robots gotta move!)

Started five years ago, but revenue-generating work got in the way:
fun projects went back on the shelf.

Three Beacons – Design Criteria

- #1: Educational to me
- Educational to others in DPRG
- Educational and interesting to children
- Audible
- Works in negative Signal-to-Noise environments
- Omnidirectional. No moving parts! (other than diaphragms and electrons)
- No sync between beacons and robot
- Not a replacement for odometry or obstacle avoidance (robot motionless)
- Cheap *and* inexpensive! Cheap off-the-shelf consumer stuff.
- No specialized parts. No changes, or only trivial changes to consumer stuff.
- Minimalist: Use fewer even if more parts would make it easier
- Parameterized
- Auto-calibrated
- No “flutzing”

Fishing without bait.

Three Beacons – Design Criteria

That was a list of *my* design criteria. You can get different, maybe better results with your *own* criteria. But let me elaborate on a few of my criteria:

- Audible

I saw members wondering why their robots plowed into obstacles, why the IR and/or sonar had not "seen" the obstacles. Also, I saw children asking, "How does sonar work?" and "How does IR work?" The children couldn't see or hear the sonar or IR, and obviously, since the robots ran into things, it didn't seem to work. So, I wanted something that children could hear, so they'd have some idea about what was going on.

- Works in negative Signal-to-Noise environments

I heard members complain about sunlight at The Science Place and other things that interfered with their sensors. IMHO, a sensor that is affected by ambient sound and ambient light is practically useless. "Practically." That is, it might be fun, but useless in the real world. I wanted something that was robust while being incredibly cheap. My criteria is -30dB SNR.

- No sync between beacons and robot

This is what prompted me to do this at all. I didn't care about audio beacons, but I boldly claimed that they could be built cheaply with no sync. I felt that I had to prove it.

Three Beacons – Is It Sonar? Is It Triangulation?

No, it’s *not sonar*.

It is not echolocation (active sonar) and it is not passive sonar.

The beacons are pinging, the robot is ***not*** pinging.

Direct-path sound used, *not* the echoes.

And, No, it’s *not triangulation*.

Triangulation requires ***directional*** sensors (antennas or microphones)
that are *pointed* at two beacons, forming a triangle. Alternately,
for triangulation, the beacons could be directional and rotating.
But my design criteria require the solution to be omnidirectional without moving.

Three Beacons – Apollonius, Pothenot, LORAN, GPS

The three beacons are a solution to one of Apollonius' problems.
His problems had to do with finding circles that were
tangent to any combination of three points, lines, or circles.

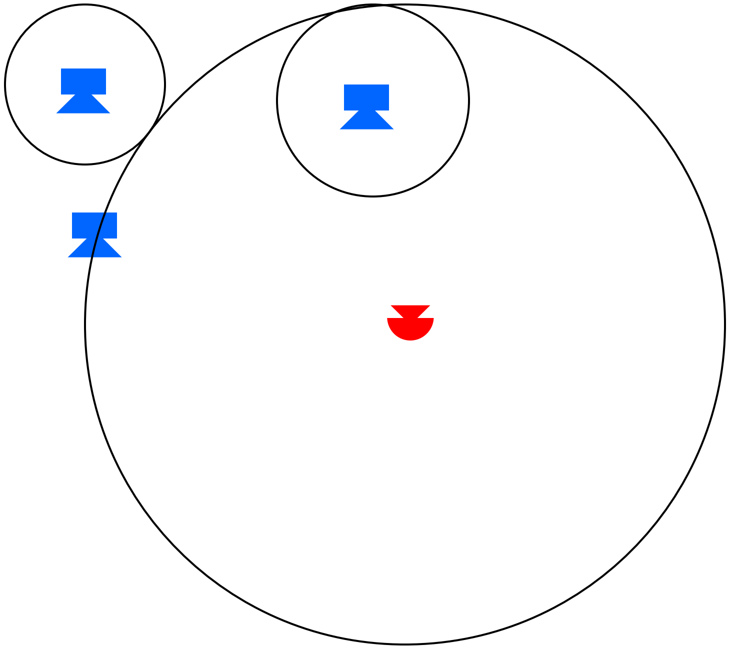
In my case, I needed to find the radius of the big circle
after measuring the radiuses of the smaller circles.

Nothing new.
Apollonius did it
2,200 years ago!

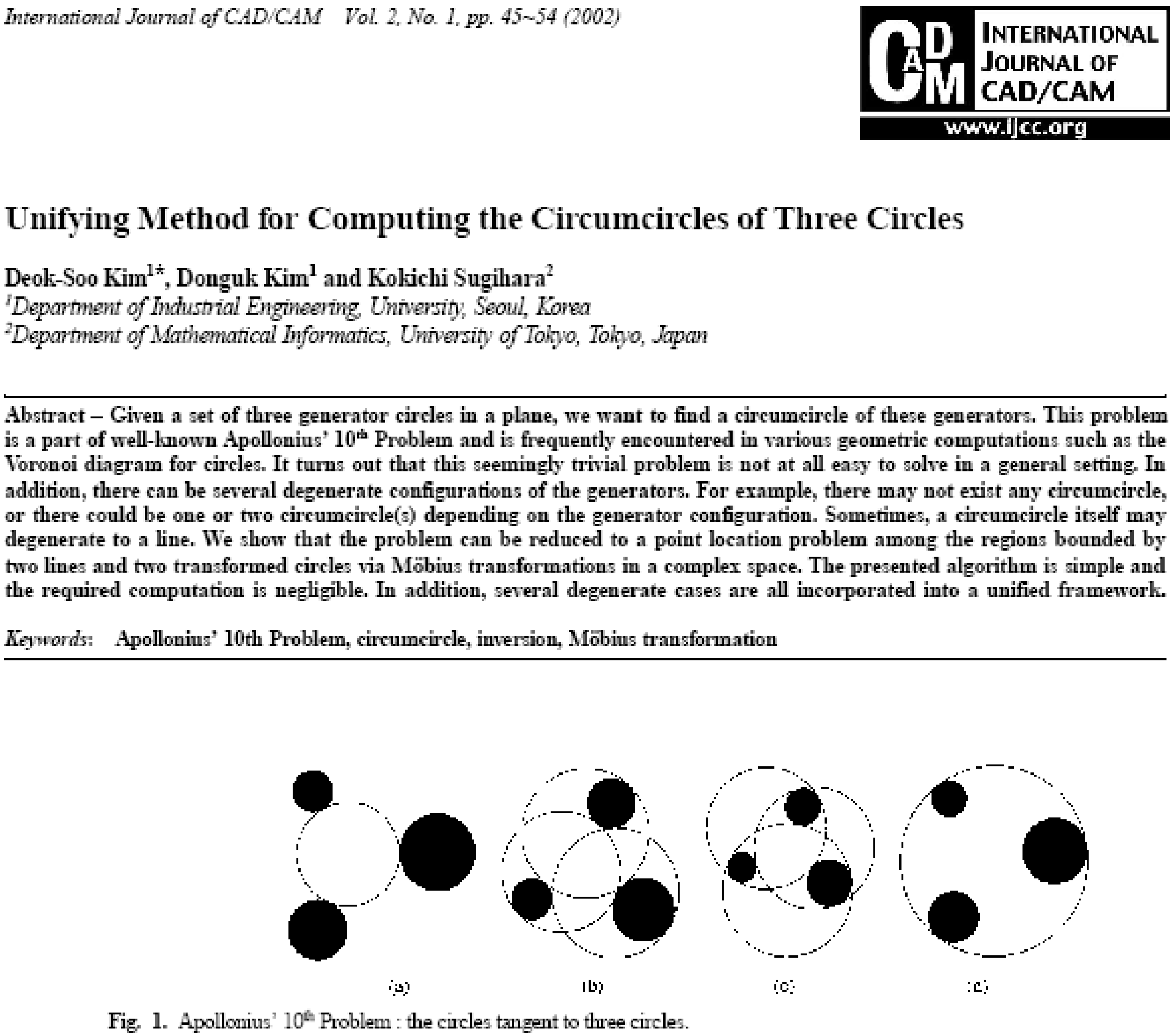
This is *not* Pothenot's problem.

Three-beacon solution also known as:

- Trilateration
- Multilateration (New World Order?)
- Hyperbolic Localization
- LORAN (in use for *70 years*)
- GPS



Three Beacons – Apollonius, Pothenot, LORAN, GPS



International Journal of CAD/CAM Vol. 2, No. 1, pp. 45~54 (2002)

INTERNATIONAL JOURNAL OF CAD/CAM
www.ijcc.org

Unifying Method for Computing the Circumcircles of Three Circles

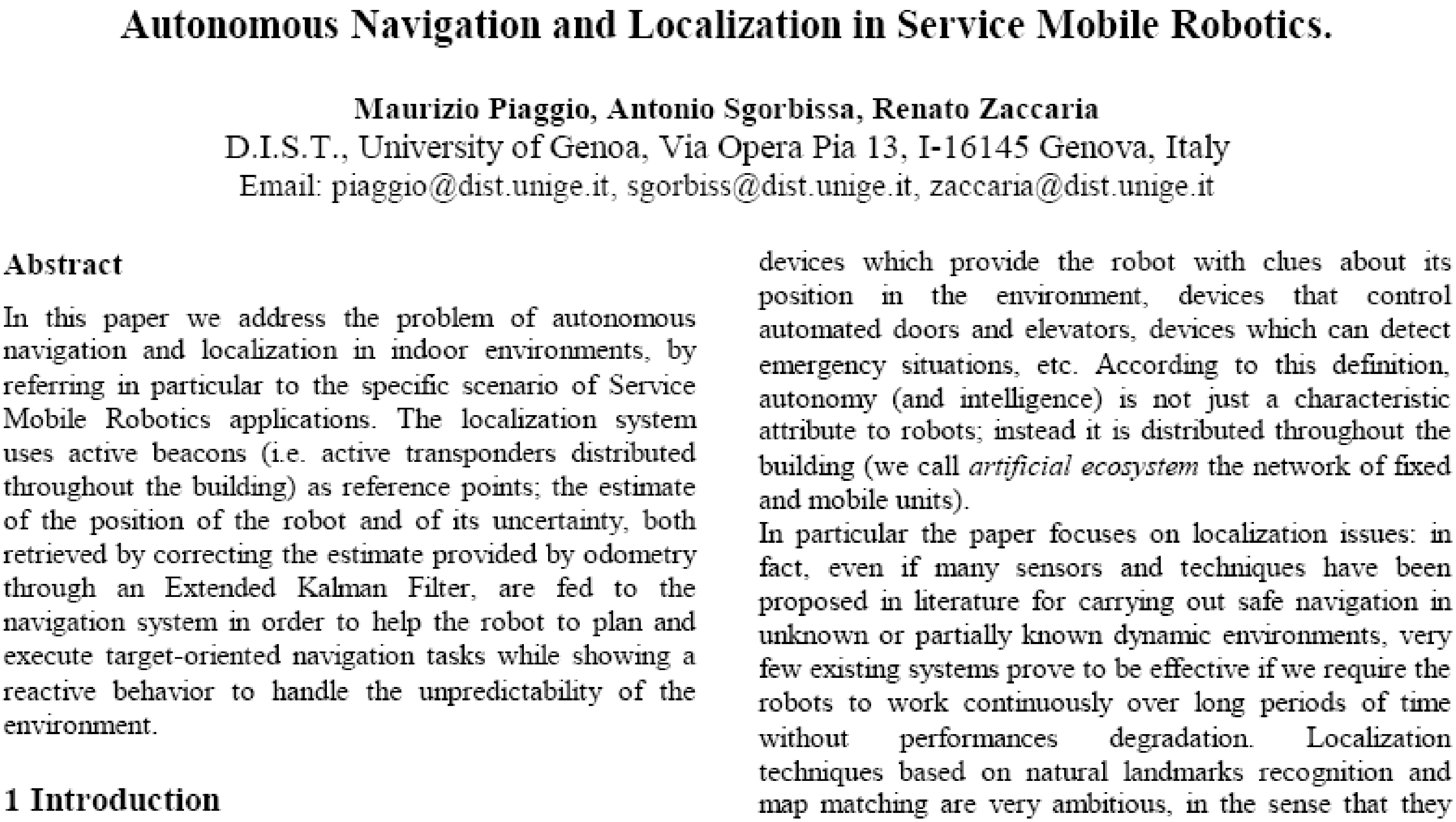
Deok-Soo Kim[1]*, Donguk Kim[1] and Kokichi Sugihara[2]
[1] *Department of Industrial Engineering, University, Seoul, Korea*
[2] *Department of Mathematical Informatics, University of Tokyo, Tokyo, Japan*

Abstract – Given a set of three generator circles in a plane, we want to find a circumcircle of these generators. This problem is a part of well-known Apollonius' 10th Problem and is frequently encountered in various geometric computations such as the Voronoi diagram for circles. It turns out that this seemingly trivial problem is not at all easy to solve in a general setting. In addition, there can be several degenerate configurations of the generators. For example, there may not exist any circumcircle, or there could be one or two circumcircle(s) depending on the generator configuration. Sometimes, a circumcircle itself may degenerate to a line. We show that the problem can be reduced to a point location problem among the regions bounded by two lines and two transformed circles via Möbius transformations in a complex space. The presented algorithm is simple and the required computation is negligible. In addition, several degenerate cases are all incorporated into a unified framework.

Keywords: Apollonius' 10th Problem, circumcircle, inversion, Möbius transformation

Fig. 1. Apollonius' 10th Problem : the circles tangent to three circles.

Three Beacons – Apollonius, Pothenot, LORAN, GPS



Autonomous Navigation and Localization in Service Mobile Robotics.

Maurizio Piaggio, Antonio Sgorbissa, Renato Zaccaria
D.I.S.T., University of Genoa, Via Opera Pia 13, I-16145 Genova, Italy
Email: piaggio@dist.unige.it, sgorbiss@dist.unige.it, zaccaria@dist.unige.it

Abstract

In this paper we address the problem of autonomous navigation and localization in indoor environments, by referring in particular to the specific scenario of Service Mobile Robotics applications. The localization system uses active beacons (i.e. active transponders distributed throughout the building) as reference points; the estimate of the position of the robot and of its uncertainty, both retrieved by correcting the estimate provided by odometry through an Extended Kalman Filter, are fed to the navigation system in order to help the robot to plan and execute target-oriented navigation tasks while showing a reactive behavior to handle the unpredictability of the environment.

1 Introduction

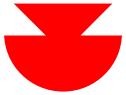
devices which provide the robot with clues about its position in the environment, devices that control automated doors and elevators, devices which can detect emergency situations, etc. According to this definition, autonomy (and intelligence) is not just a characteristic attribute to robots; instead it is distributed throughout the building (we call *artificial ecosystem* the network of fixed and mobile units).

In particular the paper focuses on localization issues: in fact, even if many sensors and techniques have been proposed in literature for carrying out safe navigation in unknown or partially known dynamic environments, very few existing systems prove to be effective if we require the robots to work continuously over long periods of time without performances degradation. Localization techniques based on natural landmarks recognition and map matching are very ambitious, in the sense that they

Two Beacons vs. Three Beacons – Cheap but Didn’t Work

Two cheap speakers would be cheaper and simpler than three speakers because all cheap PCs have stereo sound systems, but only the nice ones have more than two channels.

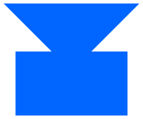
You can always make better decisions with more measurements,
so more beacons are better than fewer,
but going beyond two beacons makes everything much more complicated in the PC world.

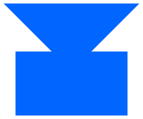


Using just two beacons, I tried coupling **relative loudness** with the traditional **time-of-arrival**, each of which provides independent equations that can be solved together to show one position for the robot.

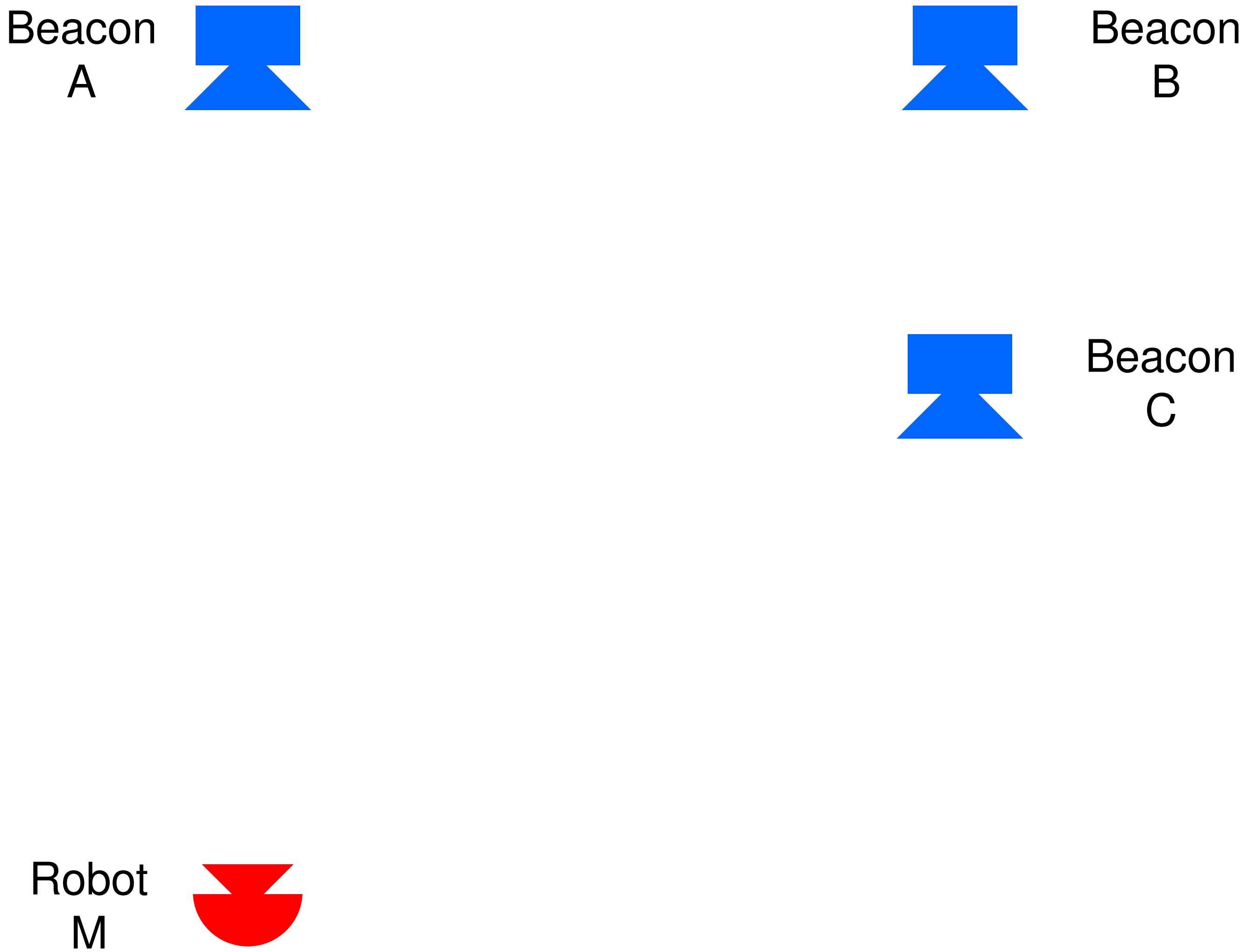
Using cheap consumer hardware, my relative loudness measurements were never repeatable, so I abandoned the two-beacon solution.

Nevertheless, I like the conceptual simplicity of the two-beacon method.

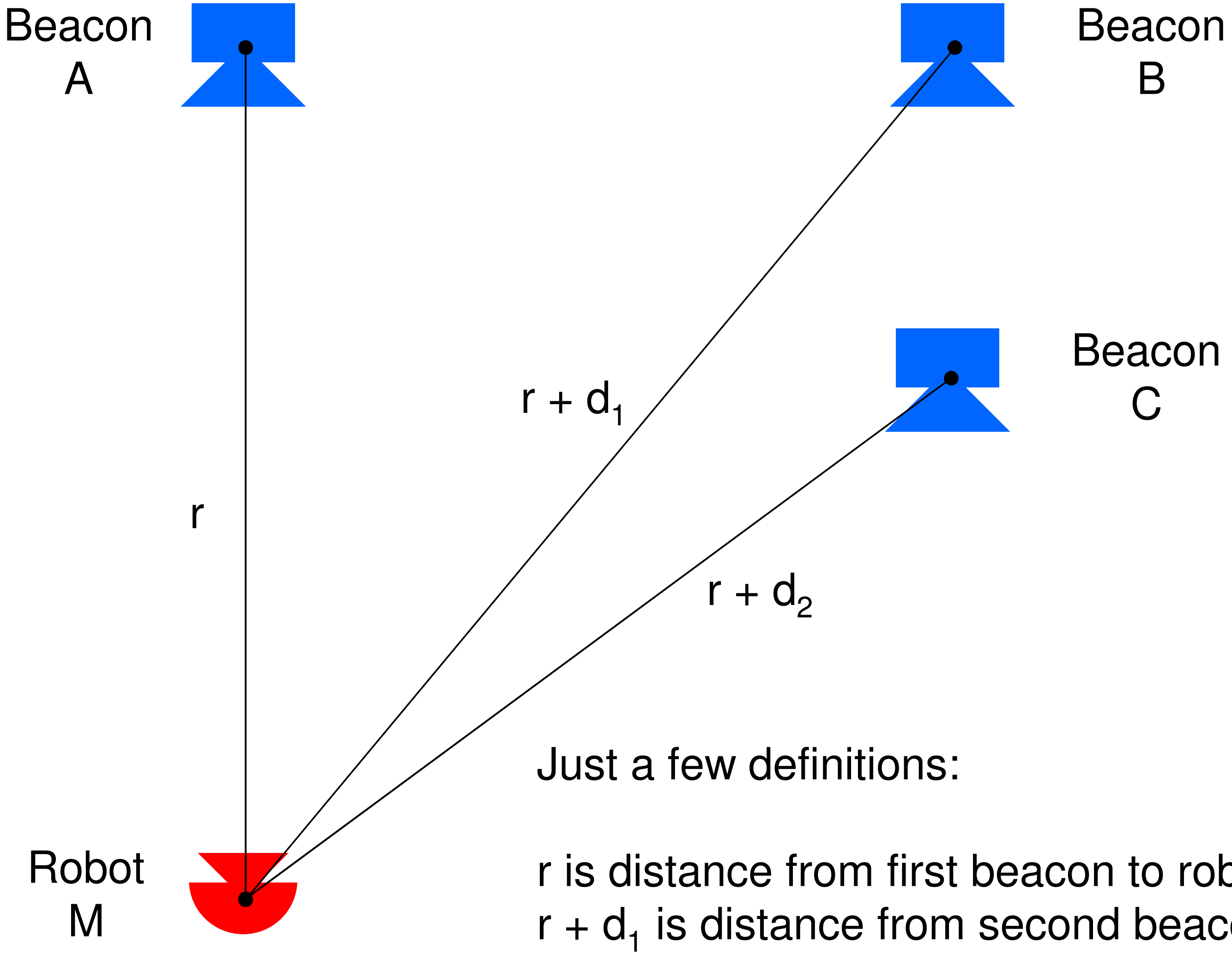




Three Beacons – Definitions



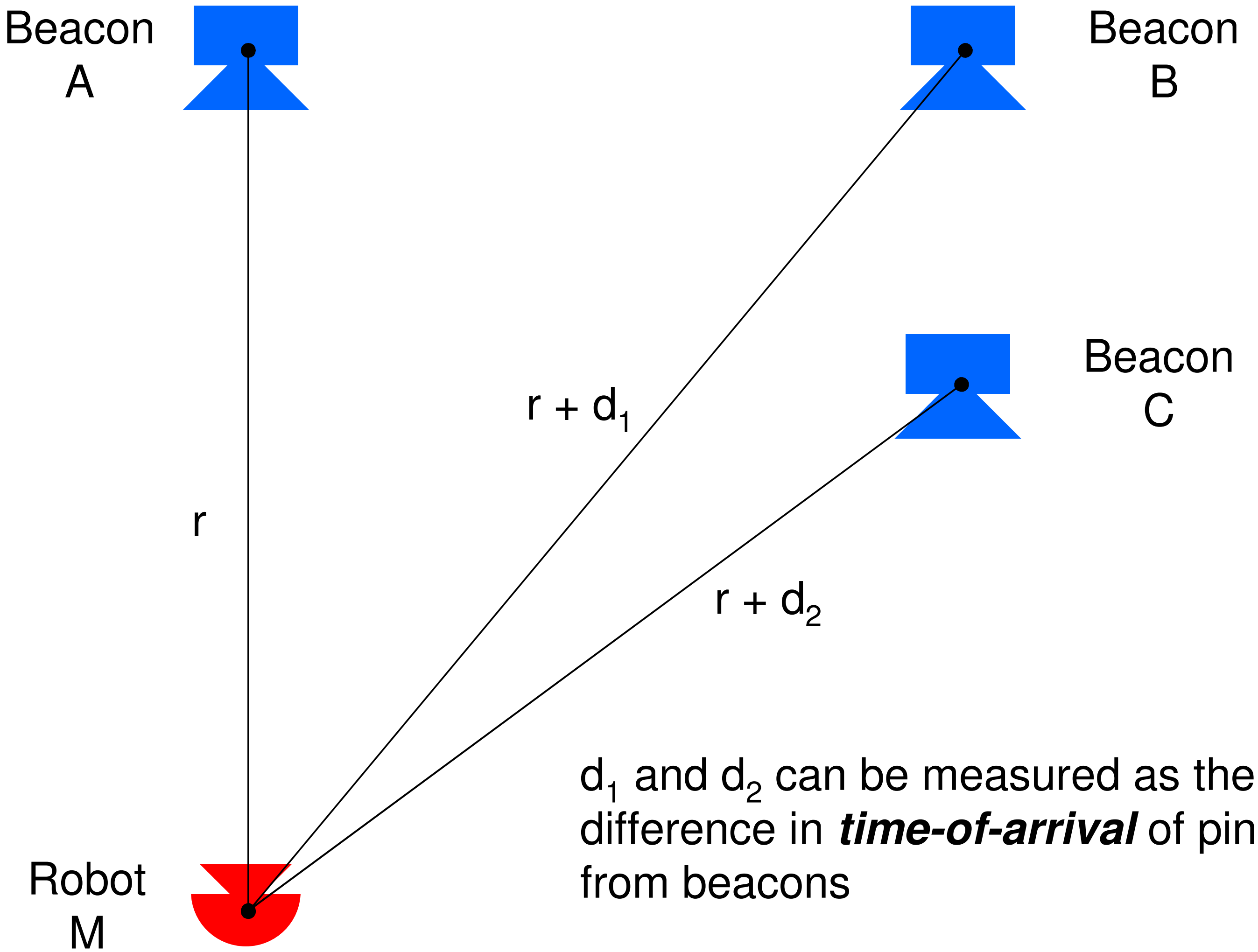
Three Beacons – Definitions



Just a few definitions:

r is distance from first beacon to robot
r + d_1 is distance from second beacon to robot
r + d_2 is distance from third beacon to robot

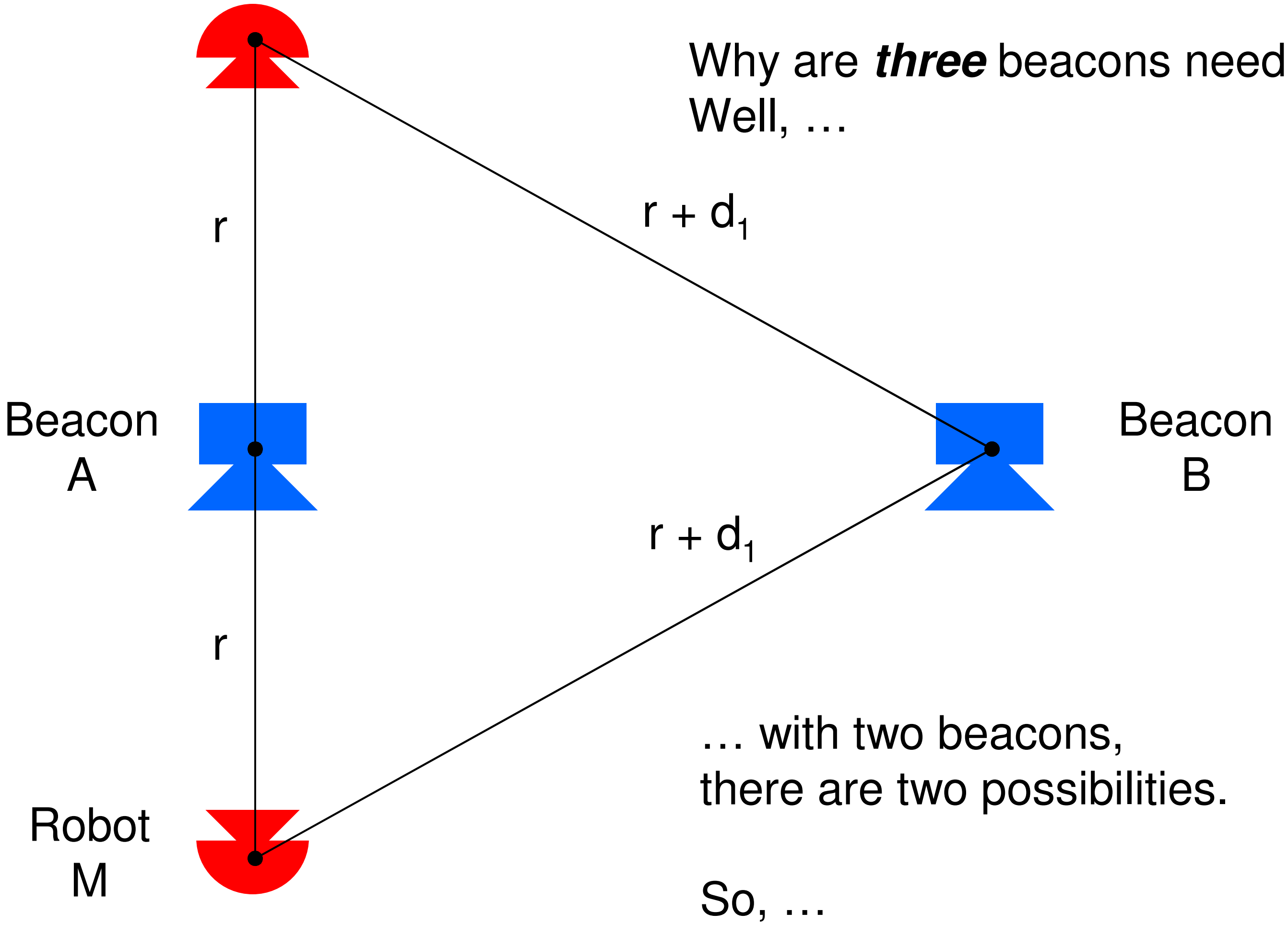
Three Beacons – Time-of-Arrival Definitions



d_1 and d_2 can be measured as the difference in ***time-of-arrival*** of pings from beacons

If we calculate r from d_1 and d_2, we'll know where the robot is.

Two Beacons vs. Three Beacons – Two Solutions (Not Good)

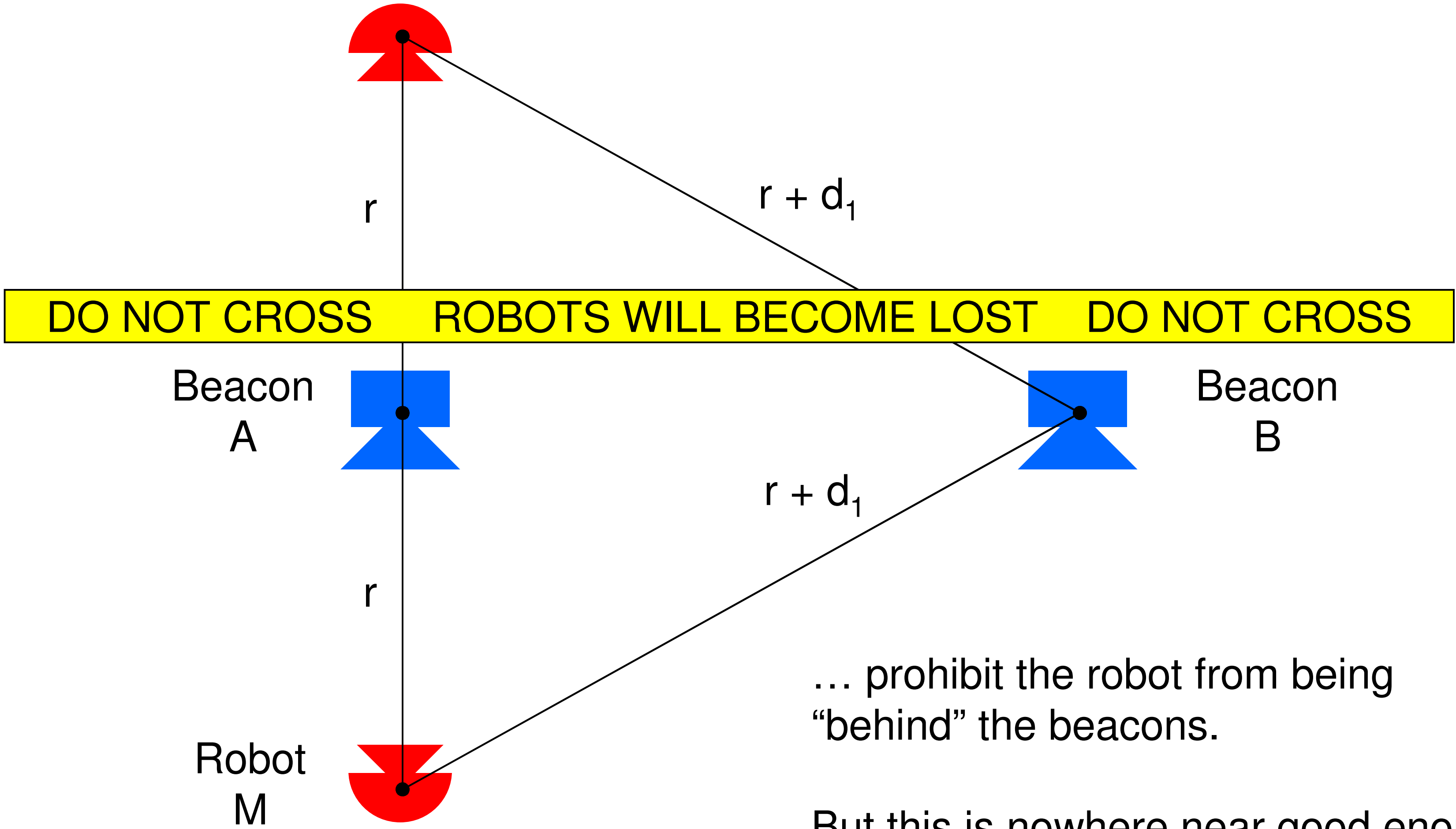


Why are ***three*** beacons needed?
Well, …

… with two beacons,
there are two possibilities.

So, …

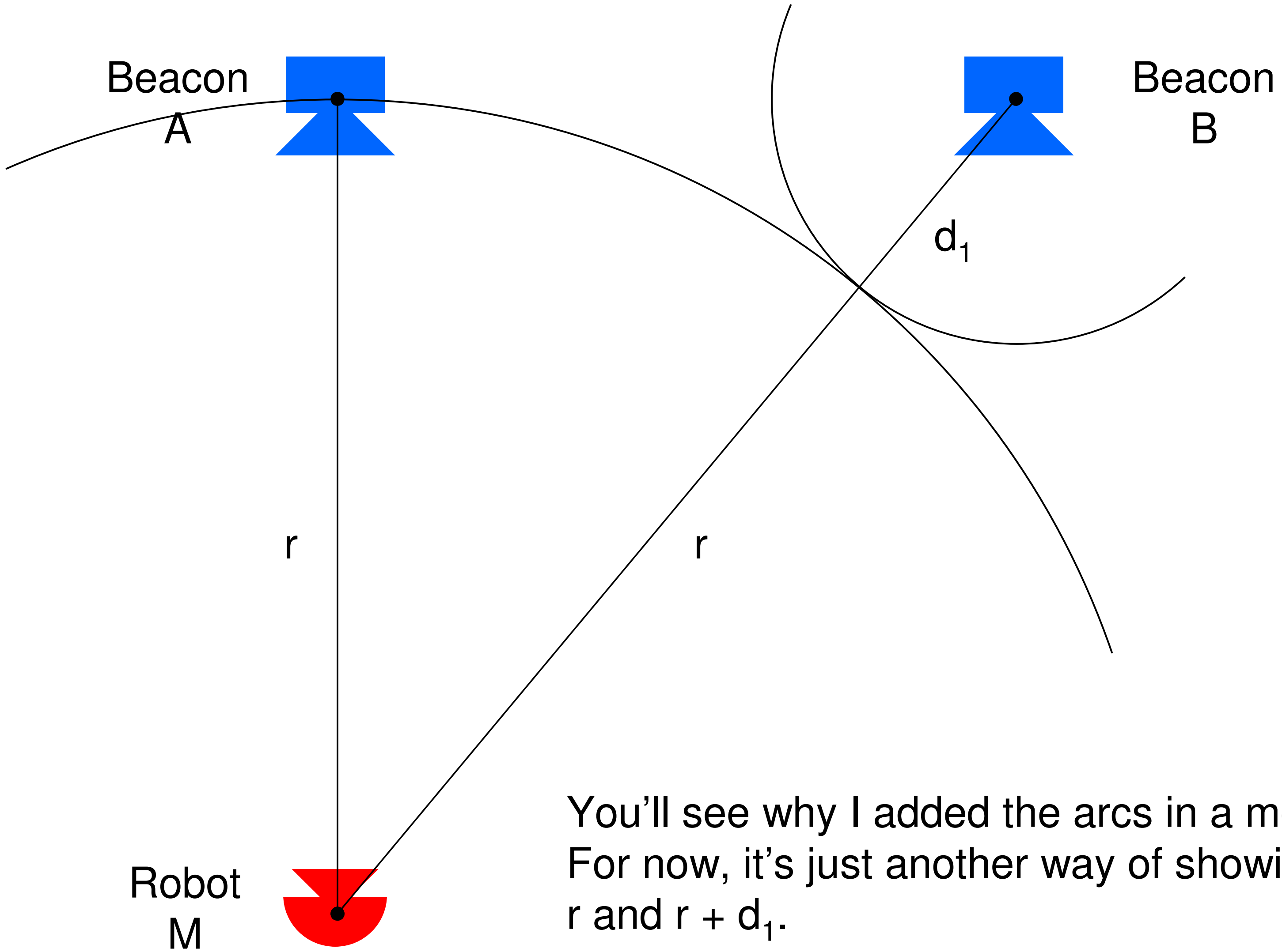
Two Beacons vs. Three Beacons – Two Solutions (Not Good)



… prohibit the robot from being "behind" the beacons.

But this is nowhere near good enough. The problem is ***much worse***.

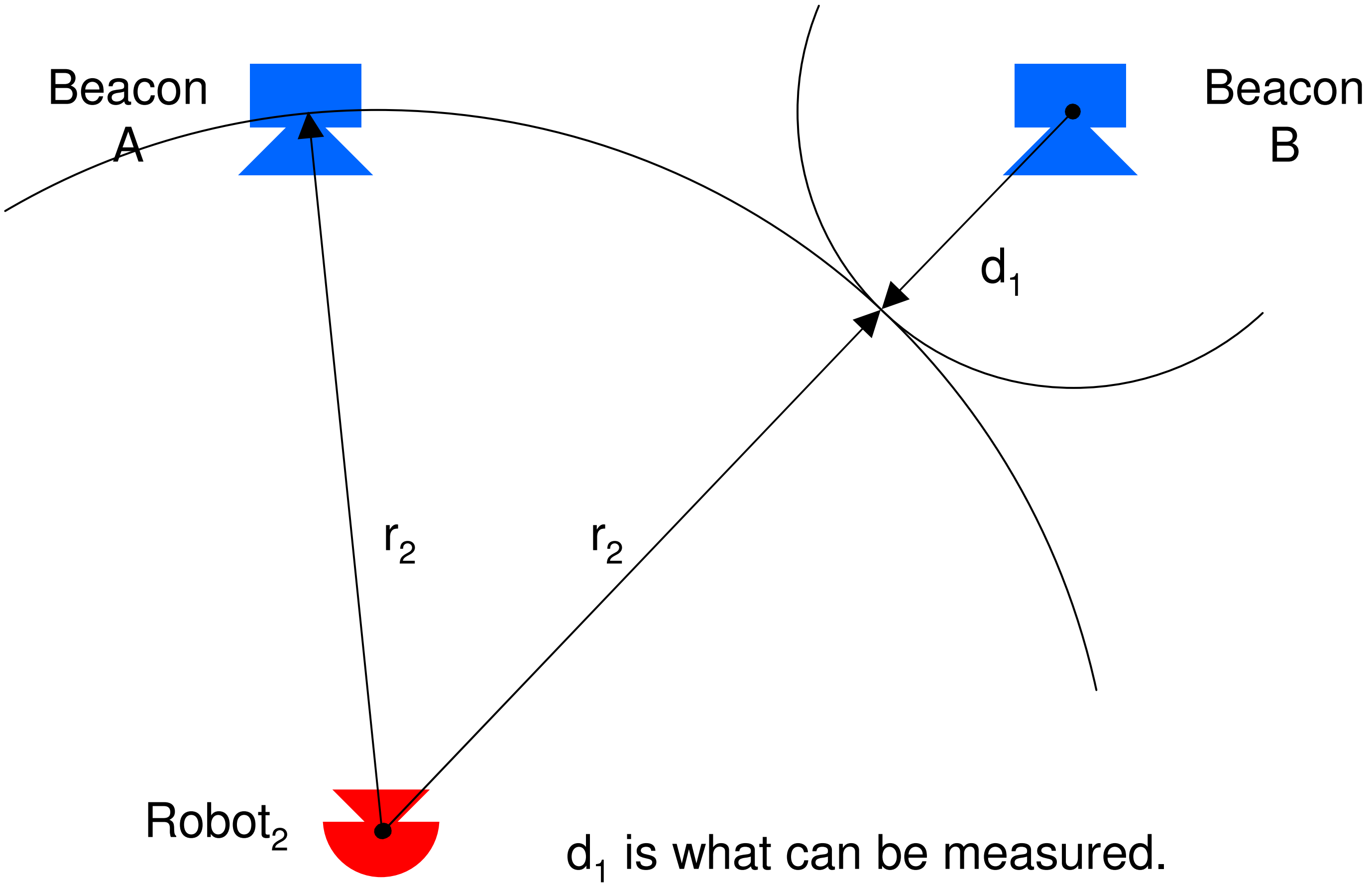
Two Beacons vs. Three Beacons – Infinite Solutions



You’ll see why I added the arcs in a moment.
For now, it’s just another way of showing
r and r + d_1.

Compare with the next slide.

Two Beacons vs. Three Beacons – Infinite Solutions

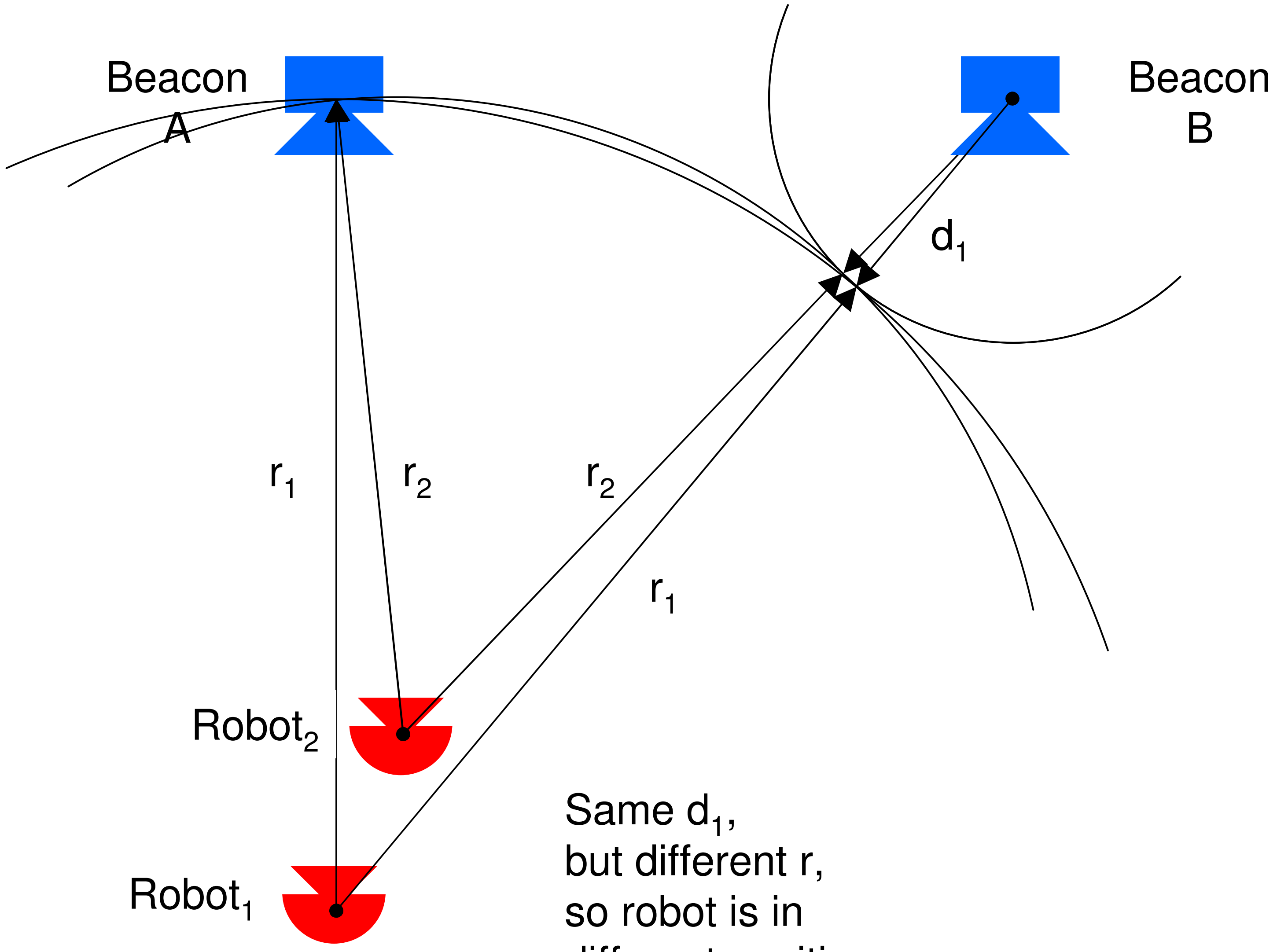


d_1 is what can be measured.

d_1 didn’t change, so robot can be in many different places with same measured d_1.

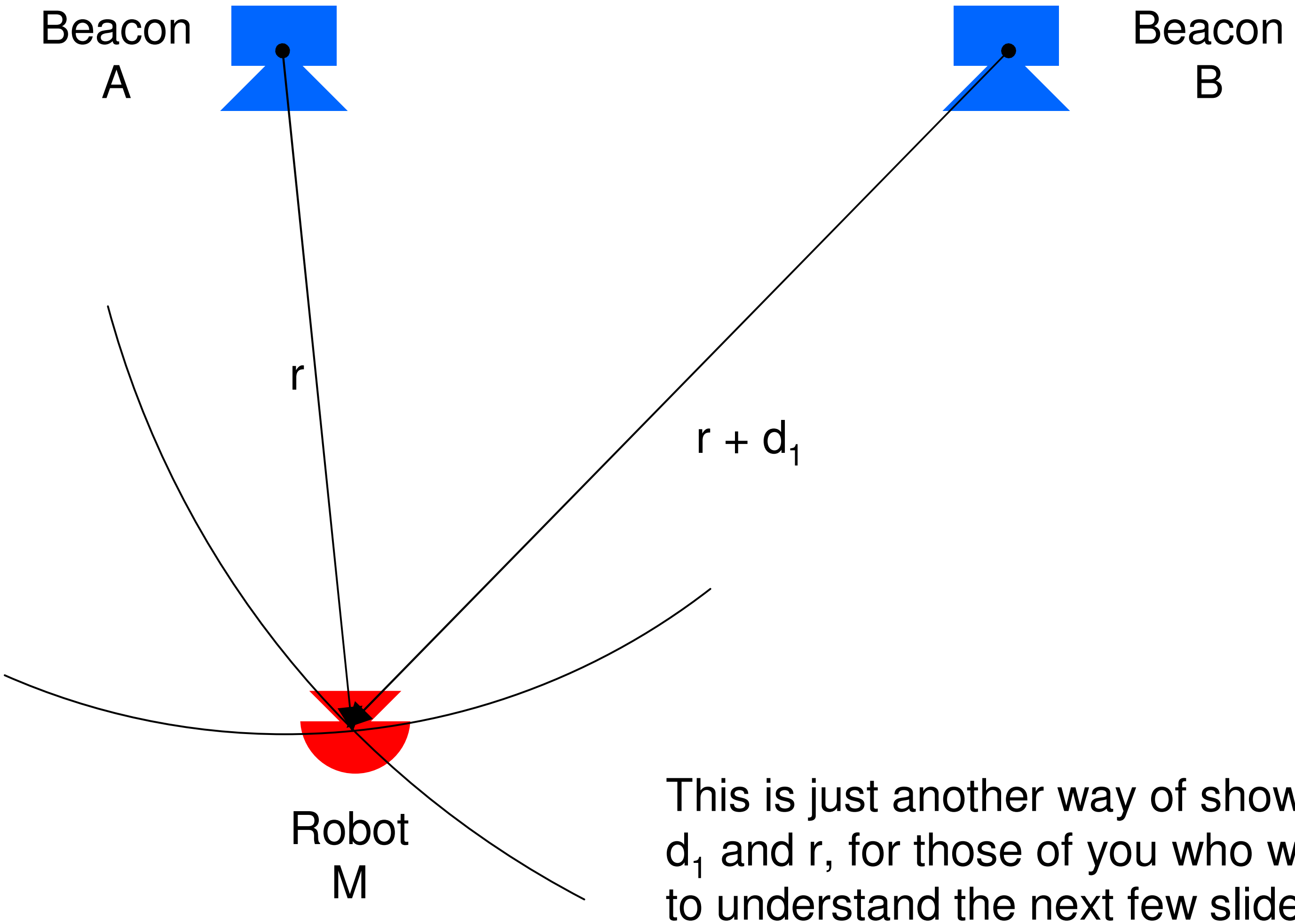
Compare with previous slide

Two Beacons vs. Three Beacons – Infinite Solutions



Same d_1,
but different r,
so robot is in
different positions.

Two Beacons vs. Three Beacons – Infinite Solutions

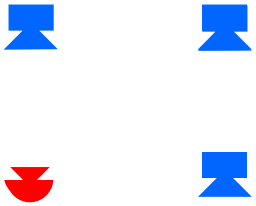


This is just another way of showing d_1 and r, for those of you who want to understand the next few slides.

Otherwise, ignore this slide.

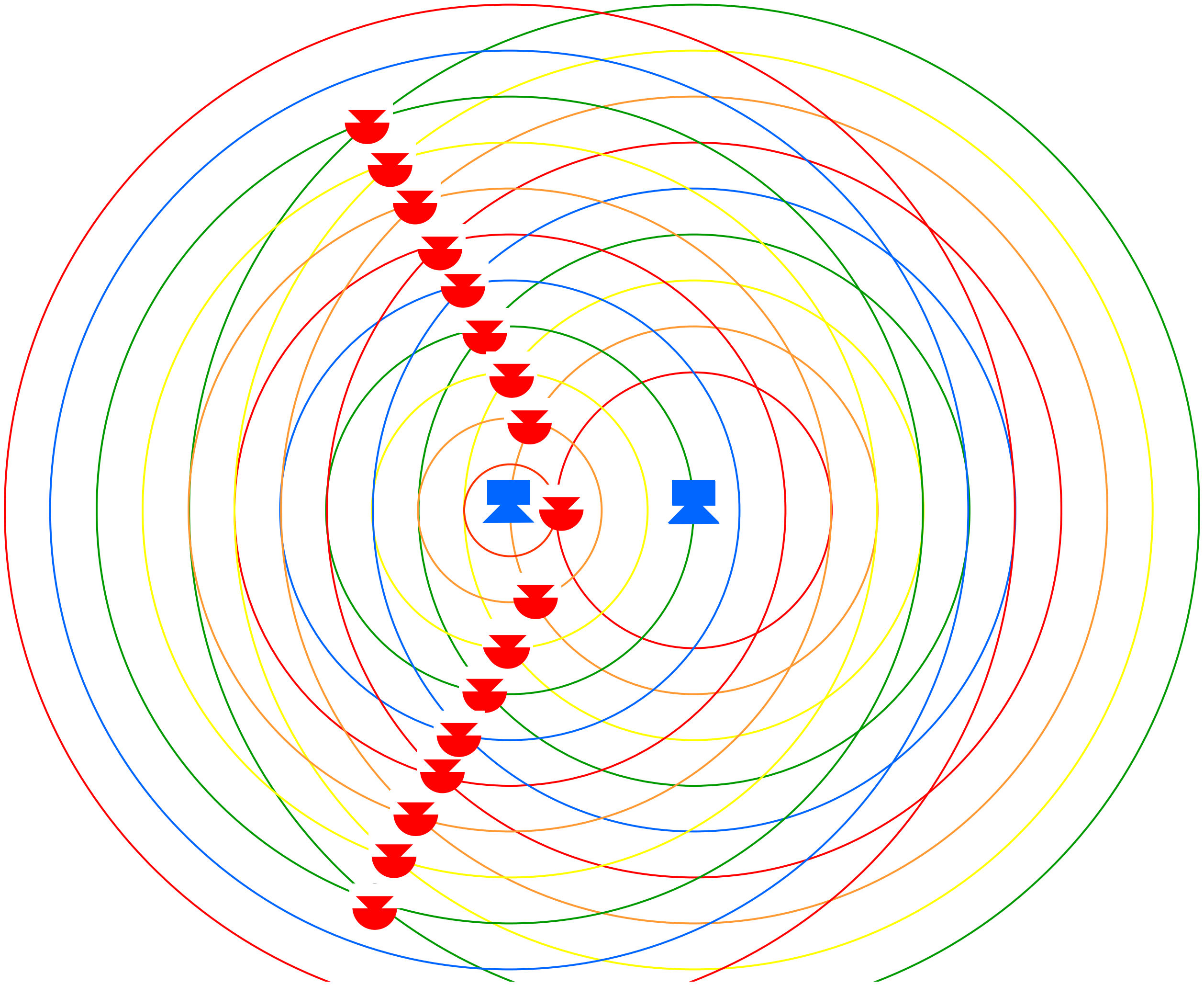
Three Beacons – Time Difference of Arrival (TDOA) Hyperbola

An example.
Note the location of the robot



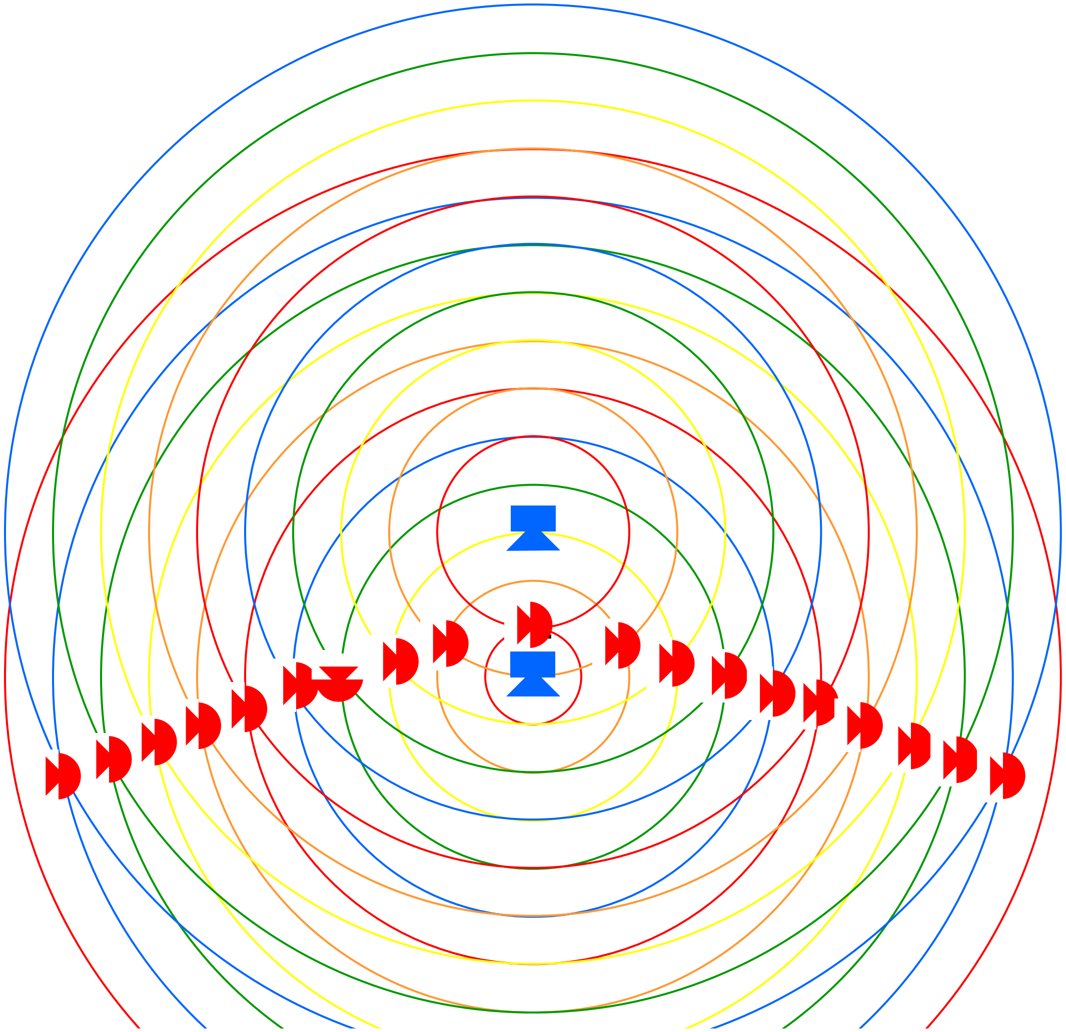
Three Beacons – Time Difference of Arrival (TDOA) Hyperbola

Each pair of beacons yields a curve of constant TDOA that the robot is on. Here we see the contribution from two of the beacons. I call this d_1.



Three Beacons – Time Difference of Arrival (TDOA) Hyperbola

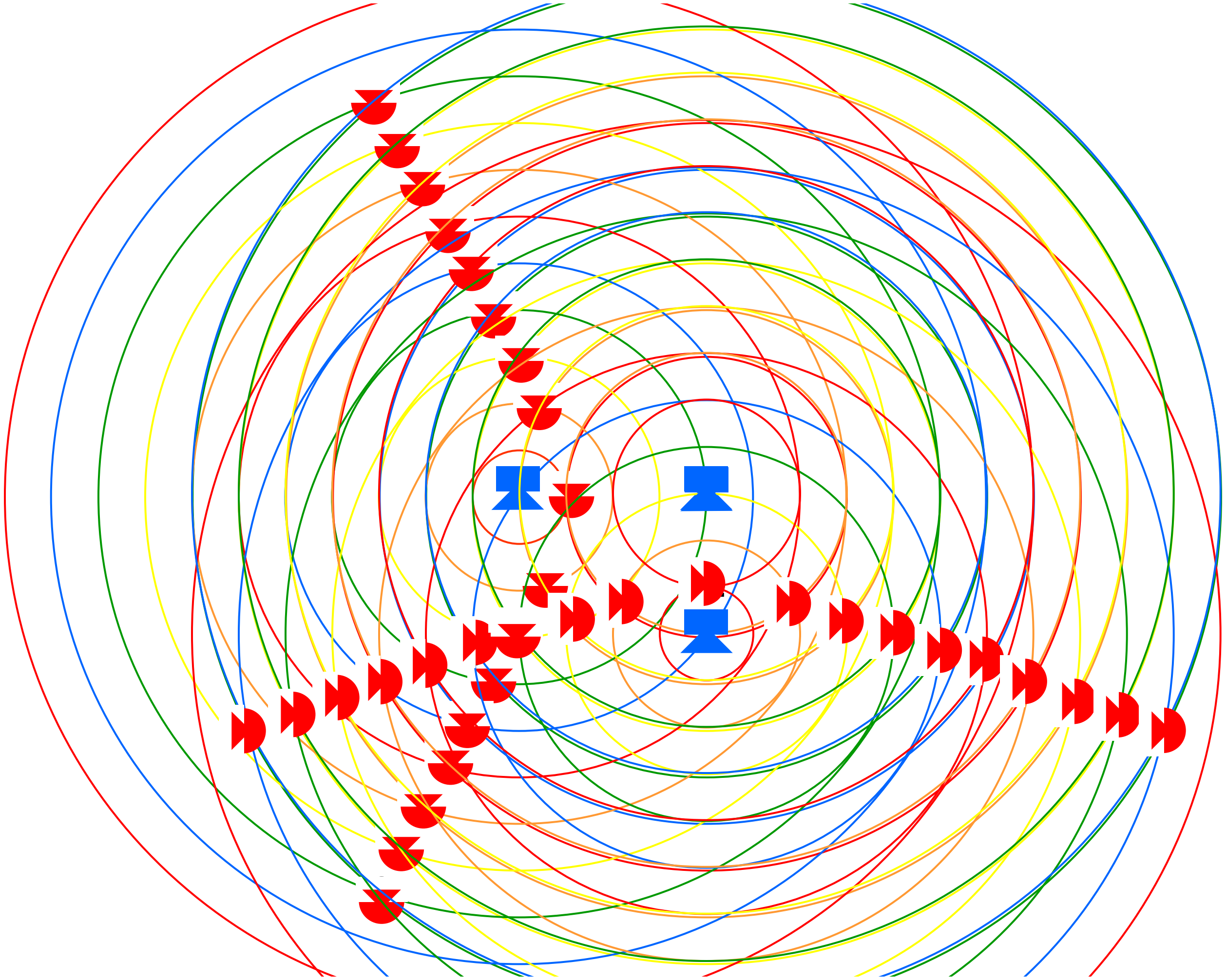
Here we see the constant TDOA contribution from another pair of beacons.
I call this one d_2.



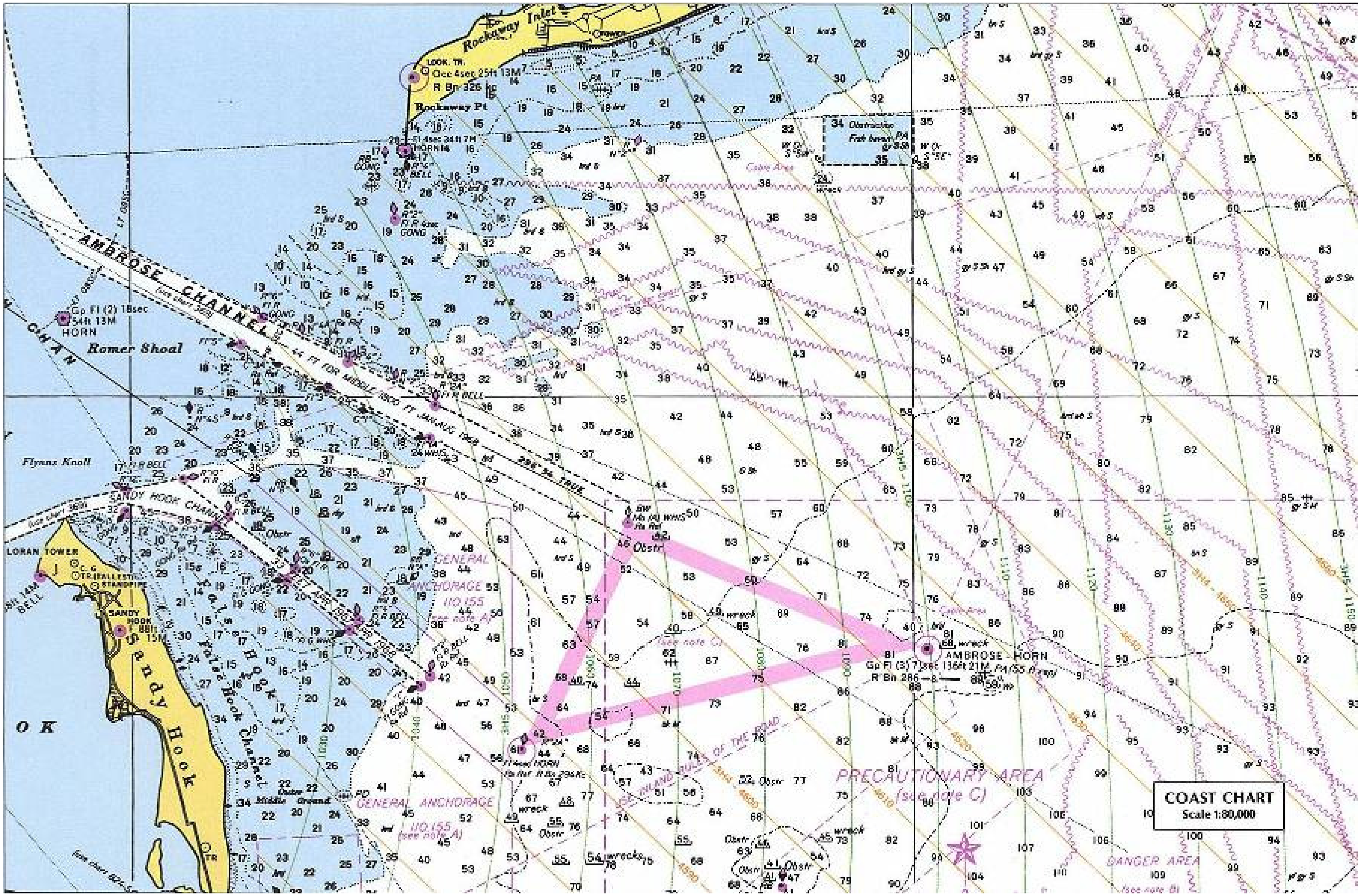
Three Beacons – Time Difference of Arrival (TDOA) Hyperbola

Combining the contributions, the two curves intersect at just one point.

Wondering if there’s a third curve (since there can be three pairings of three beacons)?
You’d need to already know r, and that would require a sync, which is a Catch-22 or chicken-and-egg problem if the beacons are radios. Essentially, one of the beacons is the sync.



Three Beacons – LORAN chart TDOA Hyperbolas

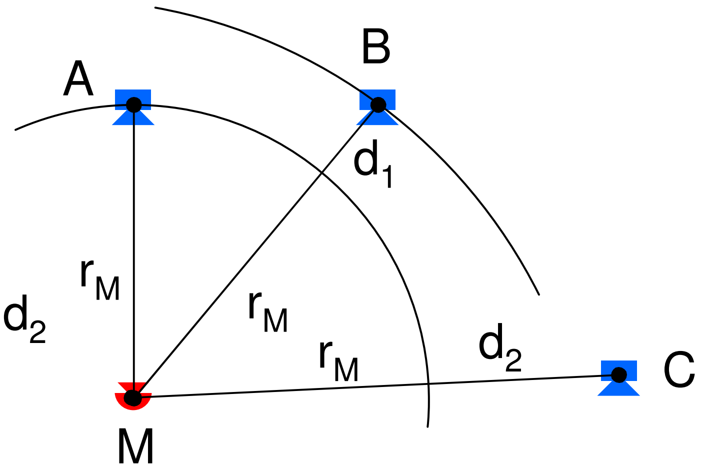


Three Beacons – Math that Decimates the Audience

Absolute knowns (given):
Location of beacon A: (x_A, y_A)
Location of beacon B: (x_B, y_B)
Location of beacon C: (x_C, y_C)

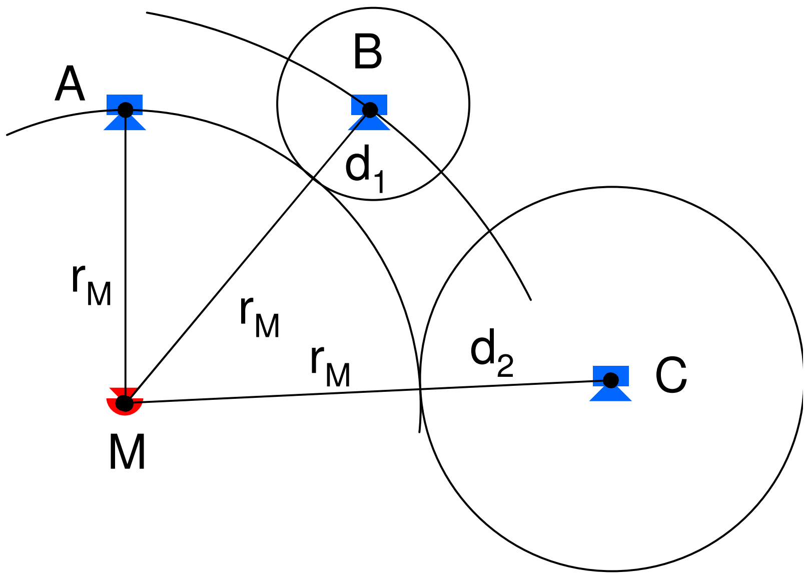
Measured knowns:
Distance differences in time of arrival: d_1, d_2

Desired unknowns:
Location of robot: (x_M, y_M) and/or
Distance from one beacon to the robot: r_M
(M is for “microphone” in my script)



Three Beacons – Solve the Equations

What Dale says will cut you in half, cuts again!
(More math.)



Equations for the circles:

$$(x_M{}^2 - x_A{}^2)^2 + (y_M{}^2 - y_A{}^2) = r_M{}^2$$
$$(x_M{}^2 - x_B{}^2)^2 + (y_M{}^2 - y_B{}^2) = (r_M + d_1)^2$$
$$(x_M{}^2 - x_C{}^2)^2 + (y_M{}^2 - y_C{}^2) = (r_M + d_2)^2$$

or this could be written as the equations
for two hyperbolas.

Since this is posed as a high-school algebra problem these days,
I suspect that I chose the more complex equations to solve.
Nevertheless, it works, and I’m not going to go back and simplify it now.

Three Beacons – Solve the Equations

Octave/Maple/Matlab code to solve for the robot's position:

The beacons are at (xA, yA), (xB, yB) and (xC, yC).
The robot is at (xM, yM).   (M is for "microphone".)

Refer to the figure on the previous slide and also slide 7.
rB and rC are the radii of the circles around the beacons that intersect with the "big" circle rM. rA is set to zero, turning that circle into a point.
rB is d_1, rC is d_2.

```
syms xB xC yC xM yM rM rB rC
S = solve((  0 - xM )^2 + (  0 - yM )^2 - ( rM +  0 )^2,
          ( xB - xM )^2 + (  0 - yM )^2 - ( rM + rB )^2,
          ( xC - xM )^2 + ( yC - yM )^2 - ( rM + rC )^2,
           xM, yM, rM )
```

The script rotates the coordinates of the beacons so that Beacon A is at the origin and Beacon B is on one of the axes. That eliminates three variables.

The script un-rotates the results before displaying them.

Three Beacons – Solve the Equations: High School Math?

```
xM = 1/2*(-yC*rB/(4*rC^2*xB^2-8*rB*xC*rC*xB-4*yC^2*xB^2+4*xC^2*rB^2+4*yC^2*rB^2)*(4*rB^2*yC^3-4*rB^2*rC^2
*yC+4*rC^2*yC*xB^2-4*rB*rC*yC*xB^2-4*rB^2*yC*xB*xC+4*rB^3*rC*yC+4*rB^2*xC^2*yC-4*yC^3*xB^2-4*xC^2*yC*
xB^2+4*yC*xB^3*xC+4*(2*xC^4*rB^4*rC^2+rB^6*xC^4+2*rB^3*xC*rC*xB*yC^4-4*rB*xC^3*rC*xB^3*yC^2+4*rB*xC*
rC^3*xB^3*yC^2-2*rB*xC*rC*xB^3*yC^4-4*rB^3*xC*rC^3*xB*yC^2+4*rB^3*xC^3*rC*xB*yC^2-4*rB^3*xC^3*rC^3*xB
+2*rB^3*xC*rC^5*xB-4*rB^4*xC*rC^4*xB-4*rB^3*xC*rC^3*xB^3+4*rB^3*xC^3*rC*xB^3-2*rB^5*xC^3*rC*xB-2*rB*
xC*rC^5*xB^3+rB^6*xC^2*yC^2-rB^6*xC^2*rC^2-2*rB^2*yC^2*xB^3*xC^3+rB^2*yC^2*xB^4*xC^2-2*rB^5*rC*yC^2
*xC^2+2*rB^2*xC^4*yC^2*xB^2-3*rC^2*xB^2*rB^2*xC^4-2*rB^4*xC^4*yC^2-rC^4*xB^2*rB^4-2*rC^4*xB^4*xC^2-
2*rC^2*xB^5*xC^3+rC^2*xB^6*xC^2+2*rC^4*xB^5*xC-2*rC^5*xB^4*rB-rC^6*xB^2*rB^2+2*rC^5*xB^2*rB^3+2*rC^4*
xB^4*rB^2+yC^2*xB^6*rC^2+xC^6*rB^2*xB^2-2*xC^5*rB^2*xB^3+xC^4*rB^2*xB^4+2*xC^5*rB^4*xB-
2*xC^4*rB^5*rC+3*rC^4*xB^2*rB^2*xC^2-3*rC^2*xB^4*xC^2*rB^2+2*rC^4*xB^3*rB^2*xC-2*rC^3*xB^4*rB*xC^2
+3*rC^2*xB^2*xC^2*rB^4-2*rB*xC^5*rC*xB^3+2*rB*xC*rC^3*xB^5+2*rB^3*xC^5*rC*xB+2*rB^5*xC*rC^3*xB
+4*rB*xC^3*rC^3*xB^3+4*rB*xC^4*rC*xB^4-2*rB*xC^3*rC*xB^5-2*rB^3*xC^4*rC*xB^2+2*rB^4*xC^3*rC^2*xB-
xC^2*rB^4*rC^4+2*xC^2*rB^5*rC^3-2*xC^4*rB^4*xB^2-rB^4*yC^4*xC^2-rB^2*yC^4*rC^2*xB^2+rB^2*yC^4*xC^2
*xB^2+2*rB^2*rC^4*yC^2*xB^2-2*rB^3*rC^3*yC^2*xB^2+2*rB^4*rC^2*yC^2*xC^2+2*rC^3*yC^2*xB^4*rB
+2*rC^2*yC^2*xB^4*xC^2-2*rC^2*yC^2*xB^5*xC-2*rB^2*rC^2*yC^2*xB^4+rB^4*rC^2*yC^2*xB^2-2*rB^4*yC^2
*xB^2*xC^2+2*rB^4*yC^2*xB*xC^3-rC^4*xB^6+rC^6*xB^4-xC^6*rB^4+rC^2*xB^4*xC^4-4*rB^2*rC^2*yC^2*xC^2
*xB^2-2*rB^2*rC^2*yC^2*xB^3*xC+4*rB^4*rC^2*yC^2*xB*xC-2*rC^4*yC^2*xB^4+rC^2*yC^4*xB^4+4*rB^3*rC
*yC^2*xB^3*xC-2*rB^3*rC*yC^2*xB^2*xC^2+4*rB*rC*yC^2*xB^4*xC^2-2*rB*rC*yC^2*xB^5*xC-2*rB^5*yC^2
*xB*xC*rC)^(1/2))+xC^2*rB+yC^2*rB-rC*xB^2+rB^2*rC-rB*rC^2)/(-rC*xB+xC*rB)
```

```
yM = 1/2/(4*rC^2*xB^2-8*rB*xC*rC*xB-4*yC^2*xB^2+4*xC^2*rB^2+4*yC^2*rB^2)*(4*rB^2*yC^3-
4*rB^2*rC^2*yC+4*rC^2*yC*xB^2-4*rB*rC*yC*xB^2-4*rB^2*yC*xB*xC+4*rB^3*rC*yC+4*rB^2*xC^2*yC-4*yC^3*xB^2-
4*xC^2*yC*xB^2+4*yC*xB^3*xC+4*(2*xC^4*rB^4*rC^2+rB^6*xC^4+2*rB^3*xC*rC*xB*yC^4-
4*rB*xC^3*rC*xB^3*yC^2+4*rB*xC*rC^3*xB^3*yC^2-2*rB*xC*rC*xB^3*yC^4-
4*rB^3*xC*rC^3*xB*yC^2+4*rB^3*xC^3*rC*xB*yC^2-4*rB^3*xC^3*rC^3*xB+2*rB^3*xC*rC^5*xB-4*rB^4*xC*rC^4*xB-
4*rB^3*xC*rC^3*xB^3+4*rB^3*xC^3*rC*xB^3-2*rB^5*xC^3*rC*xB-2*rB*xC*rC^5*xB^3+rB^6*xC^2*yC^2-
rB^6*xC^2*rC^2-2*rB^2*yC^2*xB^3*xC^3+rB^2*yC^2*xB^4*xC^2-2*rB^5*rC*yC^2*xC^2+2*rB^2*xC^4*yC^2*xB^2-
3*rC^2*xB^2*rB^2*xC^4-2*rB^4*xC^4*yC^2-rC^4*xB^2*rB^4-2*rC^4*xB^4*xC^2-
2*rC^2*xB^5*xC^3+rC^2*xB^6*xC^2+2*rC^4*xB^5*xC-2*rC^5*xB^4*rB-
rC^6*xB^2*rB^2+2*rC^5*xB^2*rB^3+2*rC^4*xB^4*rB^2+yC^2*xB^6*rC^2+xC^6*rB^2*xB^2-
2*xC^5*rB^2*xB^3+xC^4*rB^2*xB^4+2*xC^5*rB^4*xB-2*xC^4*rB^5*rC+3*rC^4*xB^2*rB^2*xC^2-
3*rC^2*xB^4*xC^2*rB^2+2*rC^4*xB^3*rB^2*xC-2*rC^3*xB^4*rB*xC^2+3*rC^2*xB^2*xC^2*rB^4-
2*rB*xC^5*rC*xB^3+2*rB*xC*rC^3*xB^5+2*rB^3*xC^5*rC*xB+2*rB^5*xC*rC^3*xB+4*rB*xC^3*rC^3*xB^3+4*rB*xC^4
```

Three Beacons – What's Next?

THE MATH WAS THE **EASY** PART!

(but, we're half way through the slide show, so the rest must not be too bad)

So…

What's needed to **implement** the three beacons?

and

Does it work?

and

Is it any good???

Three Beacons – Cheap Implementation

• **Computer** that runs Octave and Playrec or Matlab	Already have it
• **Amplified speakers** (three* stereo sets for 3 speakers)	$10 x 3
• **5.1 or 7.1 audio adaptor**	$50
• **Wireless mic and receiver, or 59 cent mic and $10 cable**	$11 or $14
• **AC power strip** (too many wall warts)	$5
• **Speaker extention cables**	$10 x 3
TOTAL COST	About $130

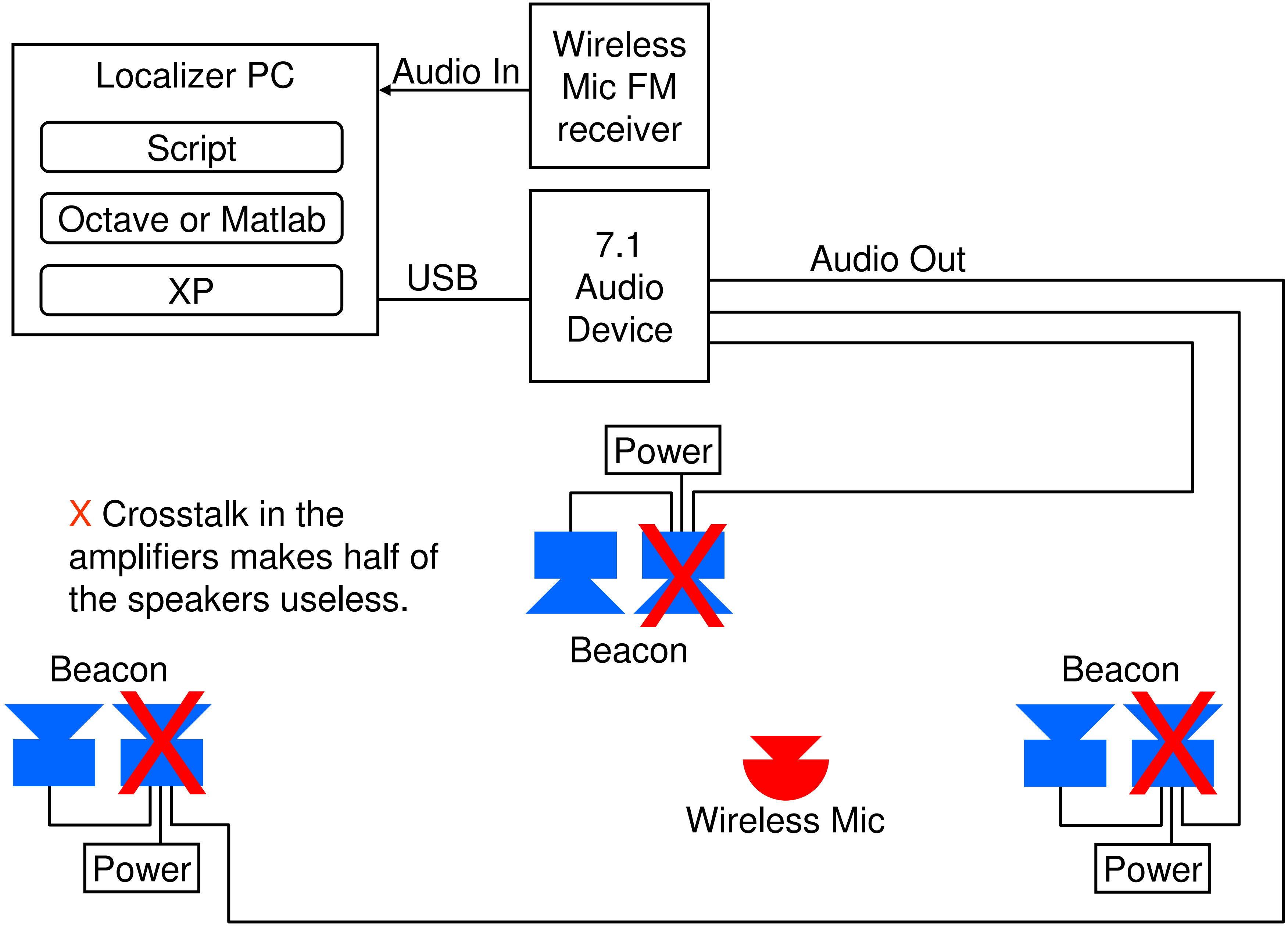
The only changes I made to off-the-shelf cheap consumer products are:

1) I **added fiberglass insulation** inside the cheap speakers to reduce the resonances of their plastic boxes, and

2) I **replaced the wire** between two of the speakers so that they could be 30 feet apart.

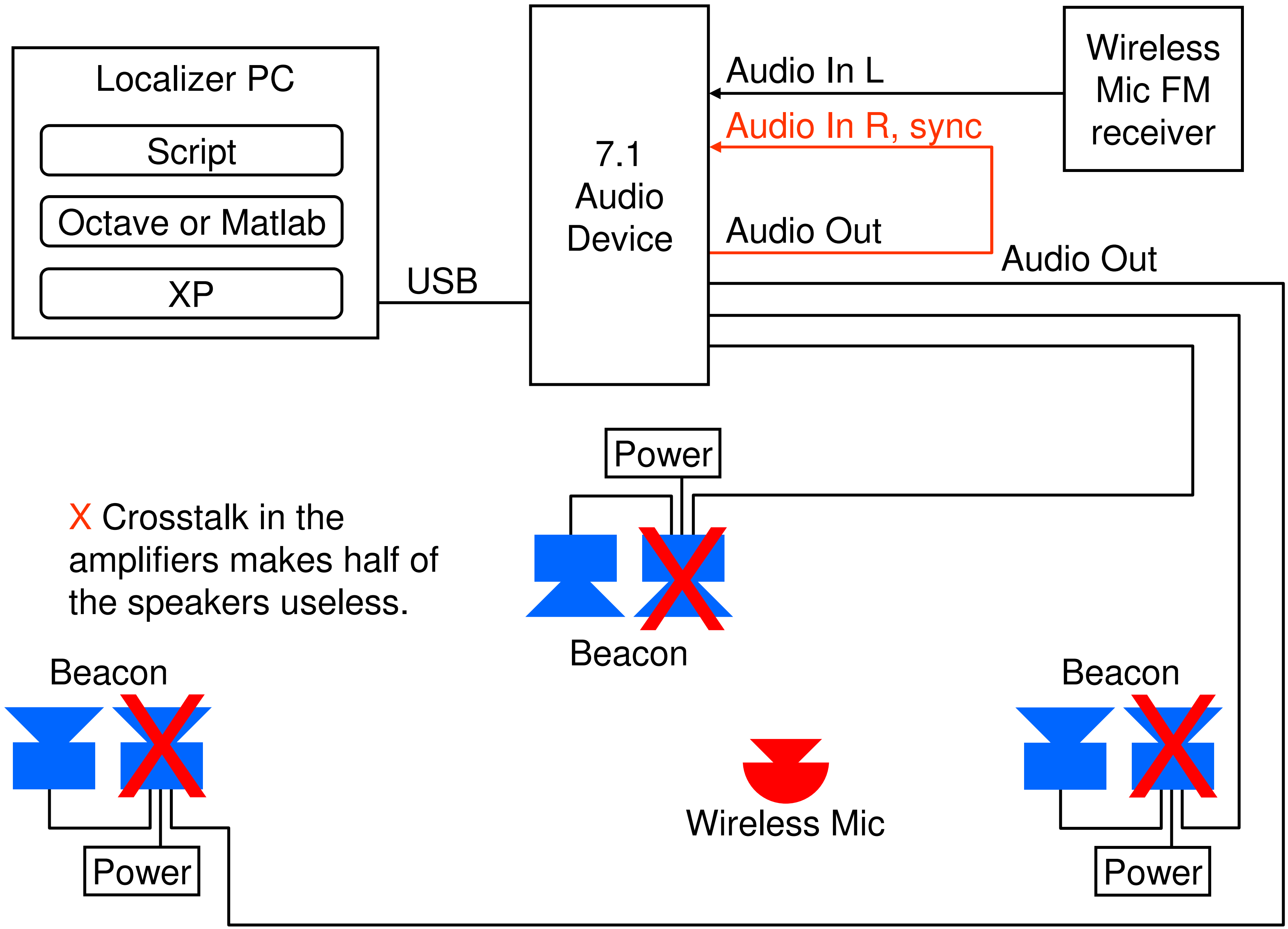
* If crosstalk were not a problem, two stereo sets would be enough.

Three Beacons – Cheap Implementation without Sync



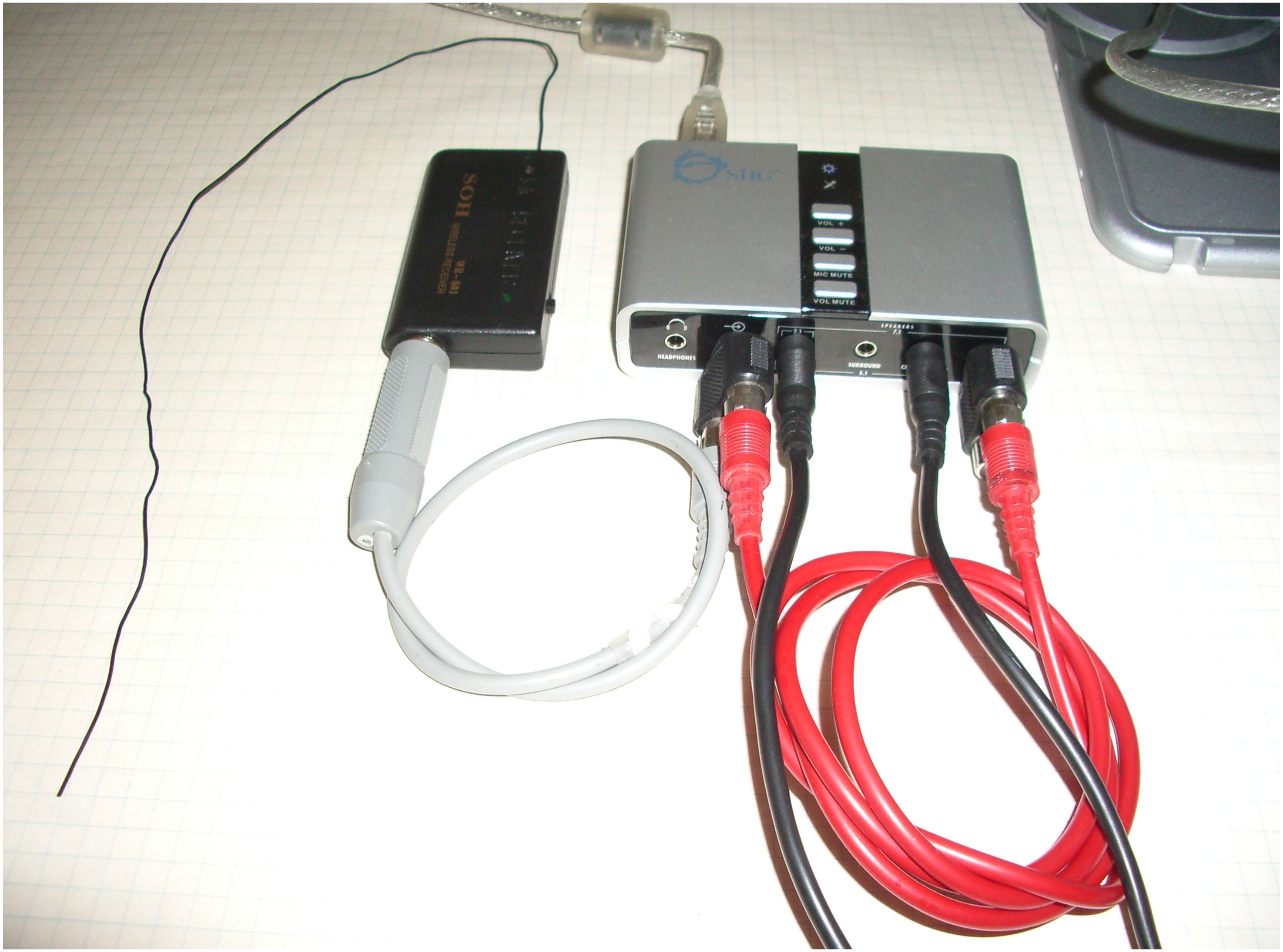
X Crosstalk in the amplifiers makes half of the speakers useless.

Three Beacons – Cheap Implementation with Sync

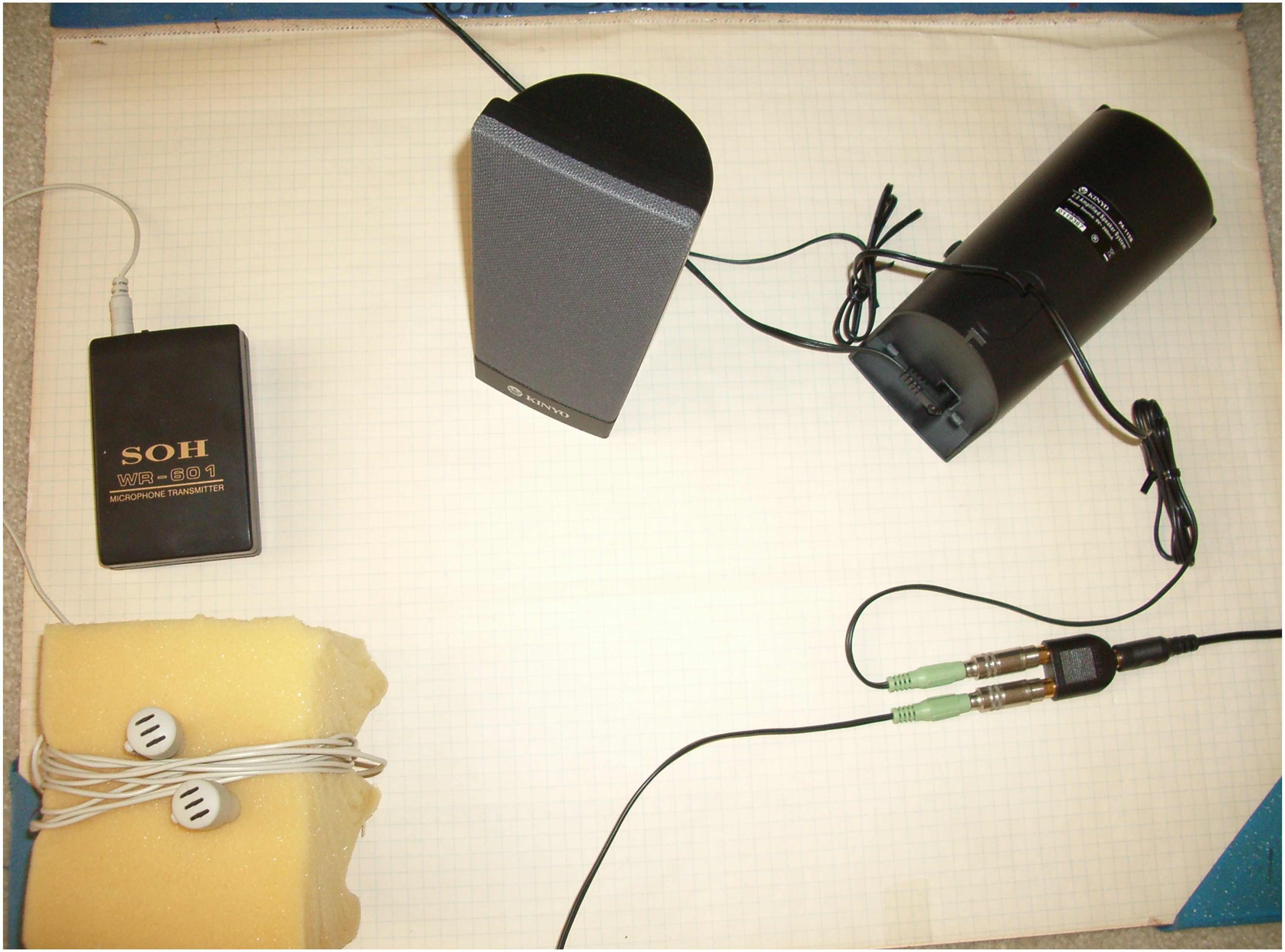


X Crosstalk in the amplifiers makes half of the speakers useless.

Three Beacons – Cheap Implementation with Sync



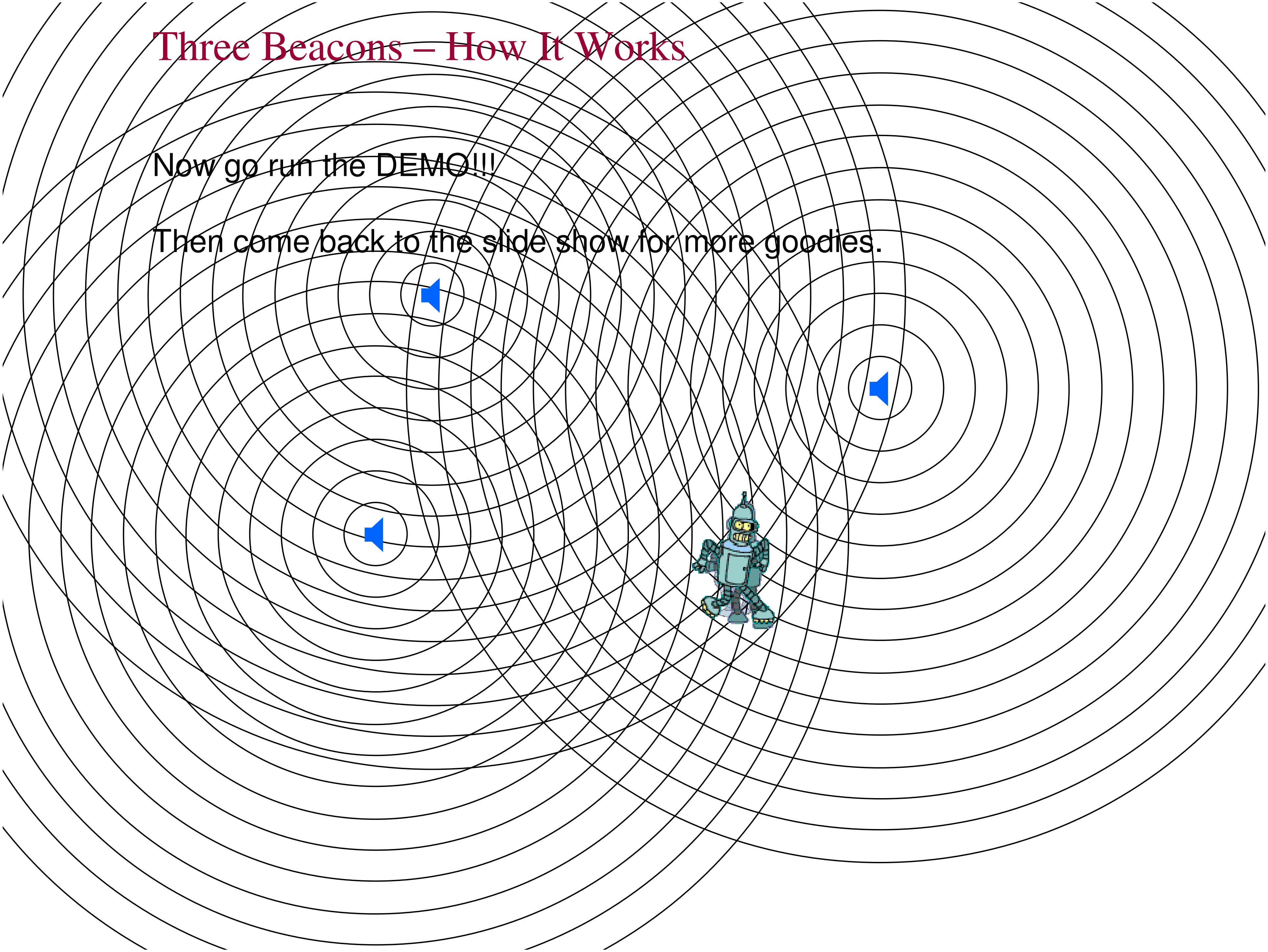
Three Beacons – Cheap Implementation



Three Beacons – How It Works

Now go run the DEMO!!!

Then come back to the slide show for more goodies.



Three Beacons – How It Works: Ping Shape

- **Haversine** instead of **Impulse**
  - Impulse itself is short, but its **wide bandwidth causes ringing** in the amplifiers and in the transducers so the ping is very very long.
  - Wide pings require the beacons to be farther from the walls.
  - **Impulse severely lacks power** because it is only one sample wide. In my experiments, the impulse had only about one-fourth the power of a haversine. You can’t make the impulse louder because everything is already at full volume, and you must never allow the ping to be distorted.
  - Impulse depends on inverse system function (deconvolution) which is totally impractical because of speaker polar response.

Three Beacons – How It Works: Ping Shape

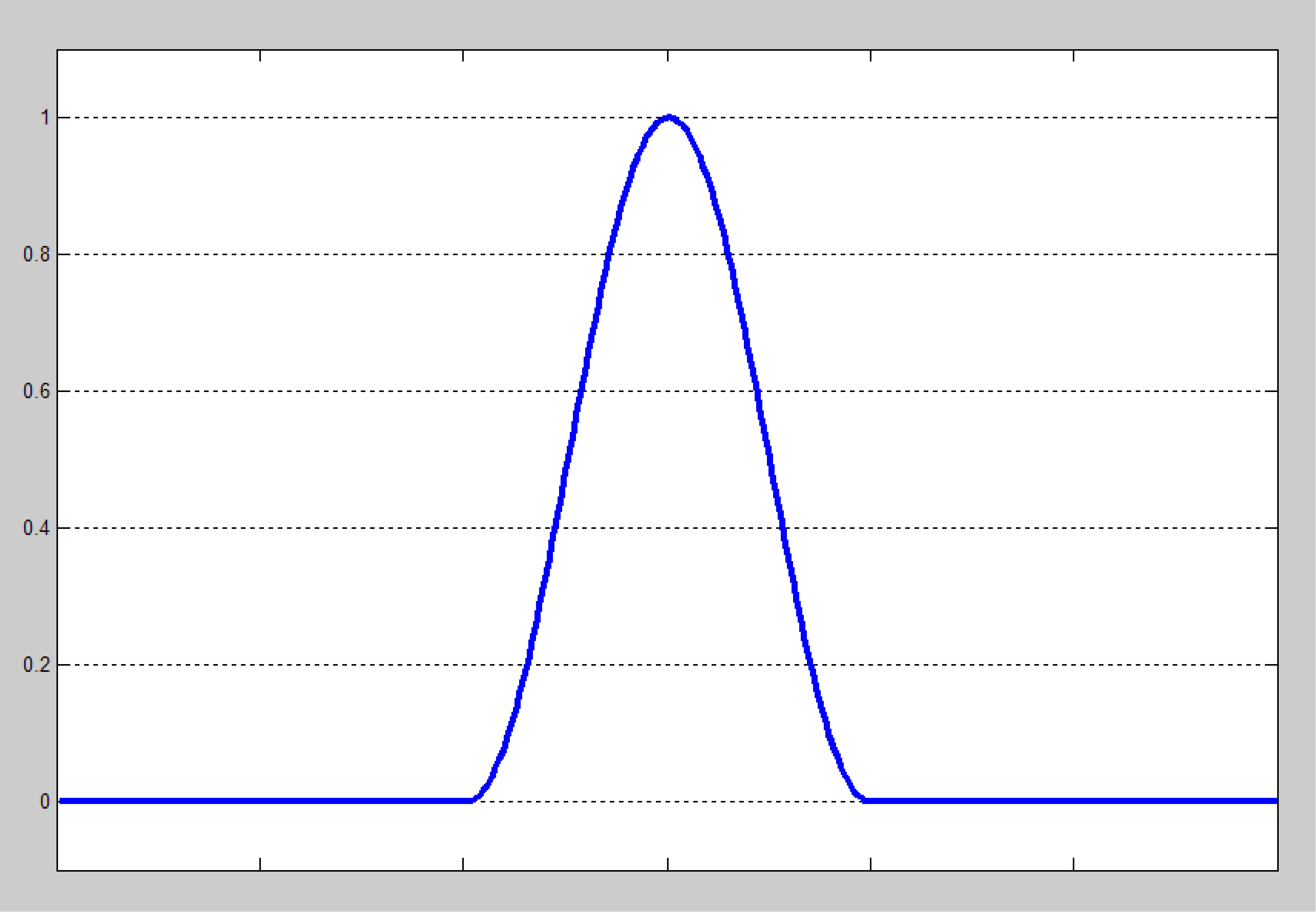
- **Haversine** instead of **Chirp**
  - Chirp is far too long (wide). Echoes from the early part of a chirp interfere with the latter parts of the same chirp.
  - Chirp depends on correlation, which cannot work because of self-similar crosstalk, precursors, and echoes (and remember, this is *not sonar*, so *echoes are bad!*).
    See slide "Nuances: Correlation Cannot Work"
  - Recorded chirp does not have the same frequency or phase as the emitted chirp, so you can't correlate them. You'd have to know the system function, which is impractical for uneven polar response.
    Look closely at the waveforms in MIT's LOUD project, and you'll realize that correlations of those chirps are bogus.

The LOUD chirp is 75 ms long. If it were used for beacons, then any object within 30 feet of the beacons would distort the recorded chirp, as shown in the LOUD datasets. If any part of the recorded chirp is correlated with the emitted chirp, then *all other* parts of the recording will be *uncorrelated*. Correlation is not appropriate for those kinds of sounds in a live (echoic) environment.

Three Beacons – How It Works: Ping Shape

- **Haversine (versine)**
  - Haversine is compact and yet it is also bandwidth limited, so the ping is short. Haversine is *gentle* to the system.
  - Haversine is several samples wide so it carries significant power.

Three Beacons – How It Works: Transmitted Haversine Ping



Three Beacons – How It Works: Ping Separation and Rep Rate

Three-beacon solutions don’t care about the ***loudness*** of the pings. Only the ***arrival time*** of the pings matters.

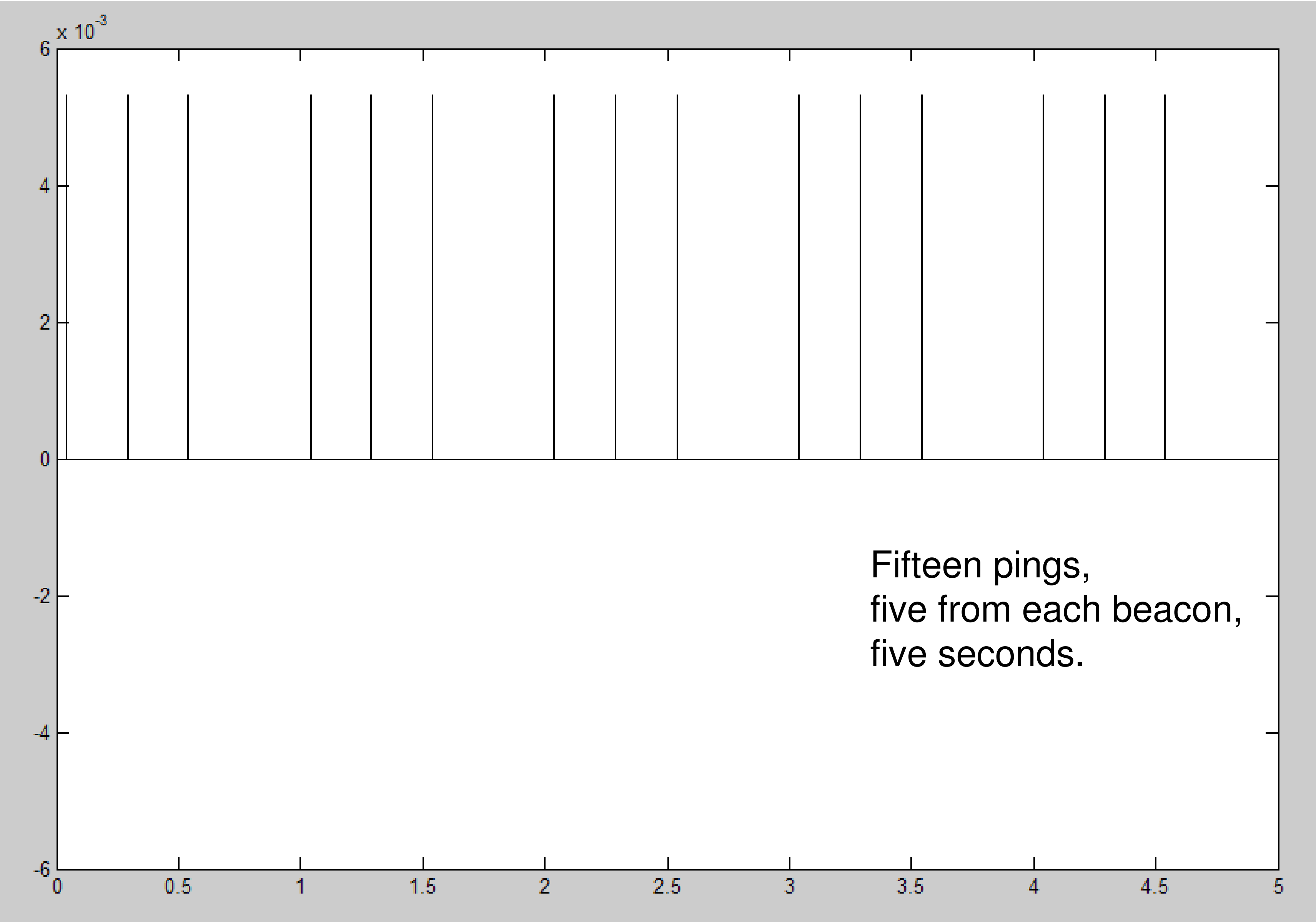
The solution depends on the ***direct-path*** (echo-free, anechoic) sound from the beacons to the mic on the robot. Echoes are assumed to always be *after* the direct-path sounds. To ensure that *old echoes* do not affect *new direct-path* sound, echoes must fade before the new pings. Two parameters are required:

➢ **Ping separation**: The ping has width. The robot can be equidistant from two or three beacons. The script cannot calculate timings when the pings are coincident (the two or three pings just sound like one ping). The pings from each beacon must be separated in time. They must be separated enough to prevent overlap in the arena, and they must be separated enough to allow echoes to die out.

➢ **Rep rate**: Echoes from previous pings could corrupt current pings. The rate that a beacon emits pings must be no faster than the ping separation.

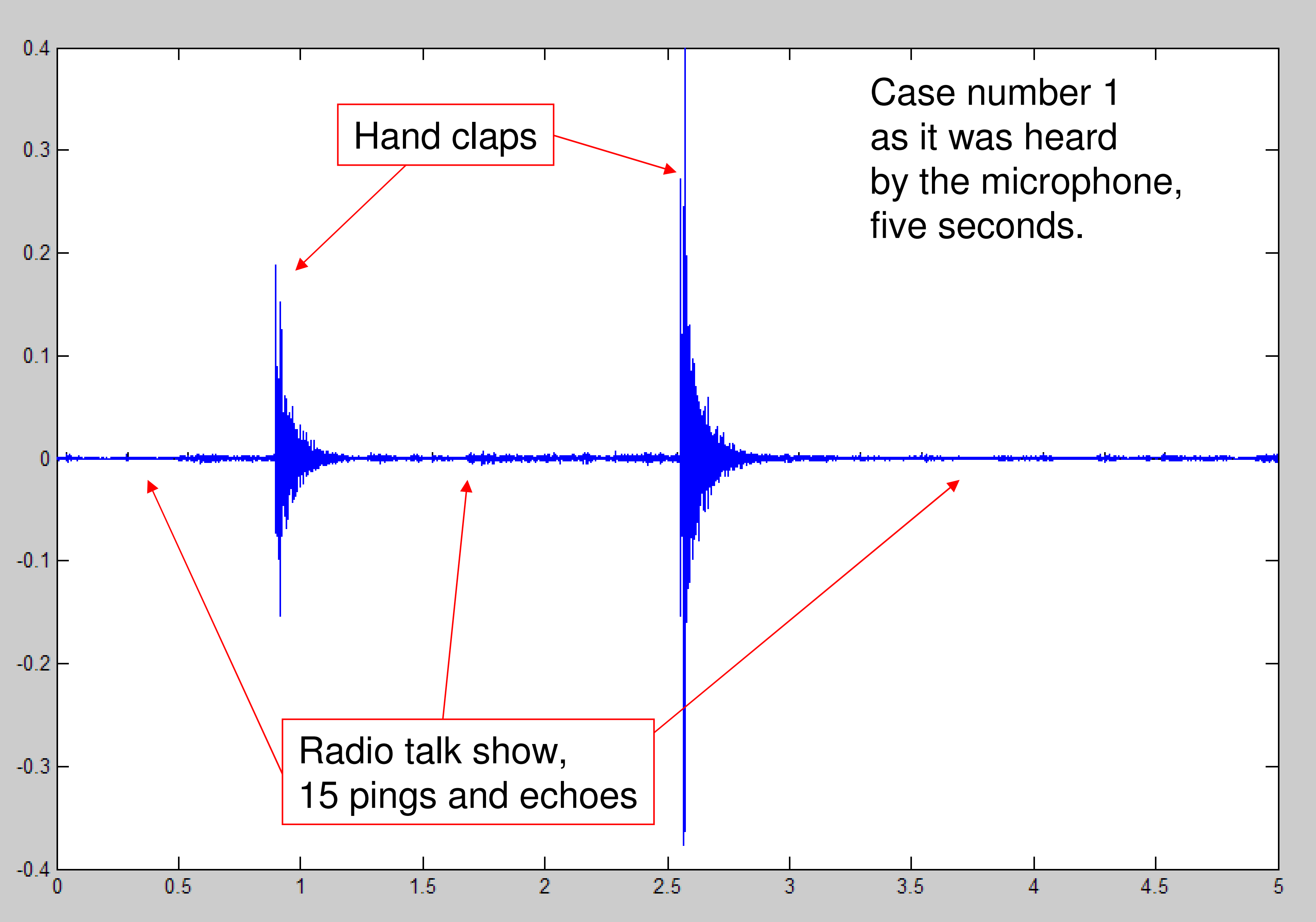
Which beacon is which? To aid in identifying the beacons, I chose to emit pings **once per second per beacon**, and to stagger the beacon’s emissions by **one-quarter second**, with **one-quarter second of silence**.

Three Beacons – How It Works: Emit Pings



Fifteen pings,
five from each beacon,
five seconds.

Three Beacons – How It Works: Record Pings, Dynamic Range

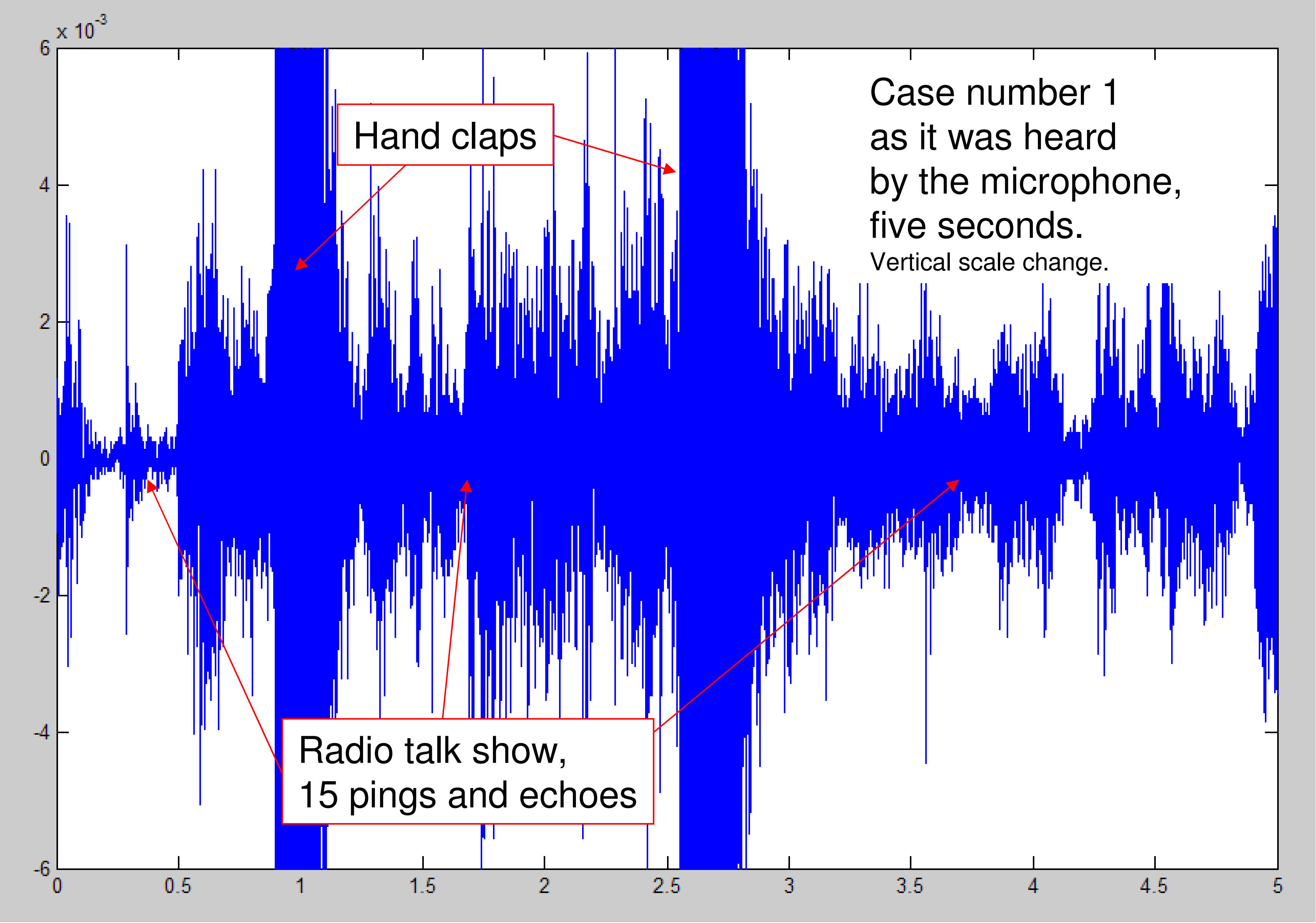


Hand claps

Case number 1
as it was heard
by the microphone,
five seconds.

Radio talk show,
15 pings and echoes

Three Beacons – How It Works: Record Pings



Case number 1
as it was heard
by the microphone,
five seconds.
Vertical scale change.

Three Beacons – How It Works: Filter

High-pass filter the recorded sound to eliminate the room rumble.
Less rumble means fewer pings are needed for good results.

Low-pass filter the recorded sound to eliminate transients that might mess up the threshold and cause a false trigger on a beacon.

I seemed to get the best results by high-pass filtering with a cutoff that was about an octave below the putative frequency of the haversine (that is, the frequency of a sine wave made up of many of the haversines strung together).

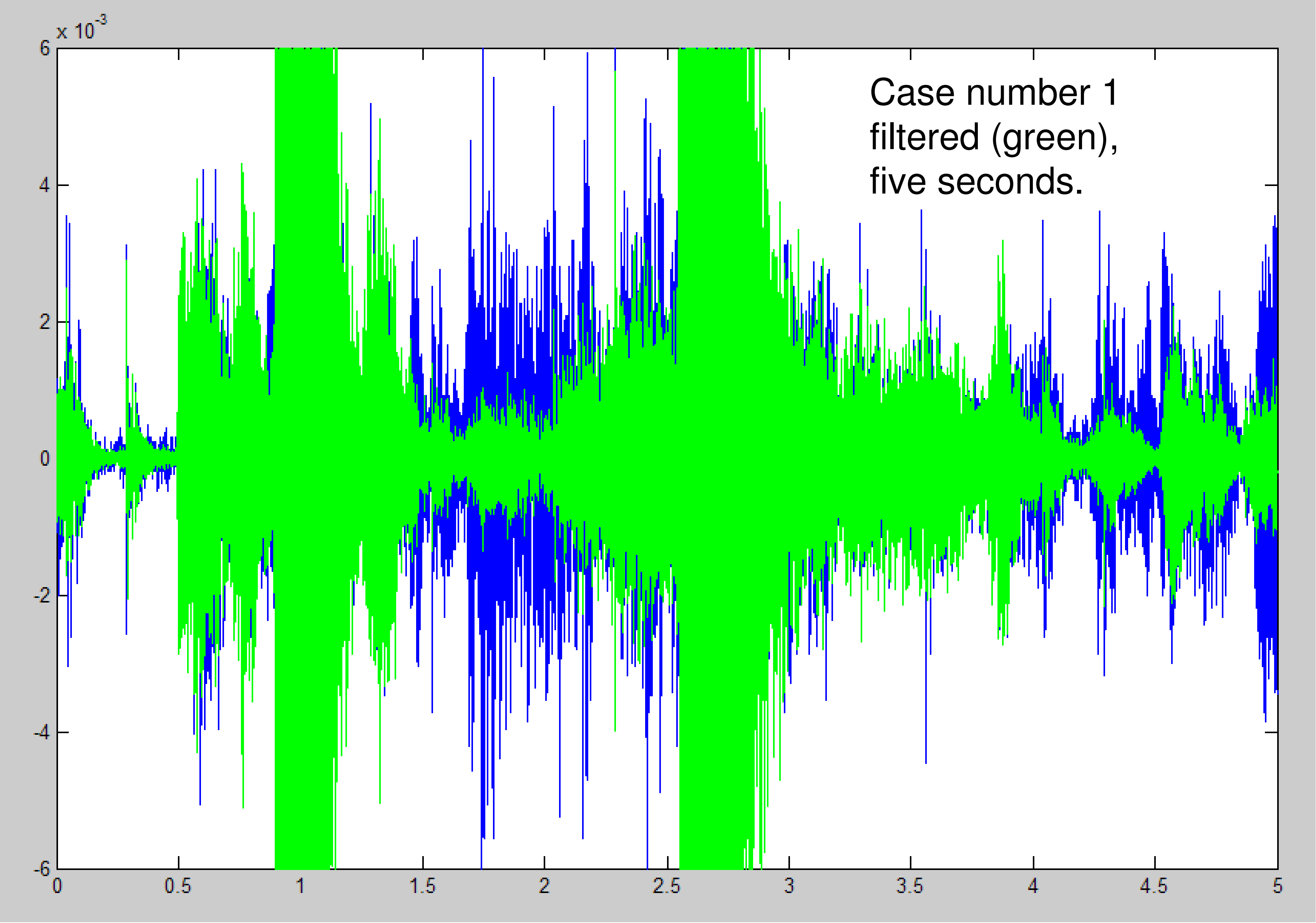
I also noticed that the high-pass filter needs to cut off at least at 300 Hz, which seems to be the higher frequency of air being blown about by air handlers.

I used a matched filter as the low-pass filter, that is, I just convolved the recorded sound with the emitted ping. That squashes the ping somewhat. It’s not really right: the recorded ping doesn’t match the emitted ping.

Be careful: Filters distort the recording. You have to balance the filter’s ability to remove undesired features against its propensity to create artifacts.

See “Nuances: Noise Filters Won’t Work” for more fun facts.

Three Beacons – How It Works: Filter



Case number 1 filtered (green), five seconds.

Three Beacons – How It Works: Waveform Averaging

When the desired signal is repetitive, it can be emphasized by chopping the waveform into sections that are as long as one iteration of the desired signal, and then summing those sections together (thus the two names: *waveform averaging* or *summation averaging*).

It's very old. Seems largely ignored. It is used in some ECG monitors (PQRST). Whenever you "try it again" you do a form of averaging.

It emphasizes the desired signal almost magically.
It doesn't require knowledge of the signal *(as a matched filter requires)*
nor knowledge of the noise *(as a Wiener filter requires)!*

It emphasizes the repetitive part by the square root of the number of sections that are averaged together, so, averaging 25 pings together improves the signal-to-noise ratio (SNR) by 5x or 14 dB.

High-speed scopes make eye diagrams, but not by doing waveform averaging. It's not the same. One builds samples to fill gaps in time, the other averages samples that are already in all the gaps. Both require precise timing, but they are different. Waveform averaging is not a scatter plot.

Cepstrum (a form of autocorrelation) extracts repetitive signals from speech, but it is not waveform averaging and it will not work here. Also, although the waveform averaging could be done by correlation, it is terribly inefficient. See slide "Nuances: Correlation Cannot Work"

Three Beacons – How It Works: Compression

The claps and shouts easily overwhelm the sound of the pings, so the claps and shouts must be dealt with. Two possible approaches are:

- Waveform averaging, that is, *more* waveform averaging. Sum more sections together. It always works, but it takes a while.

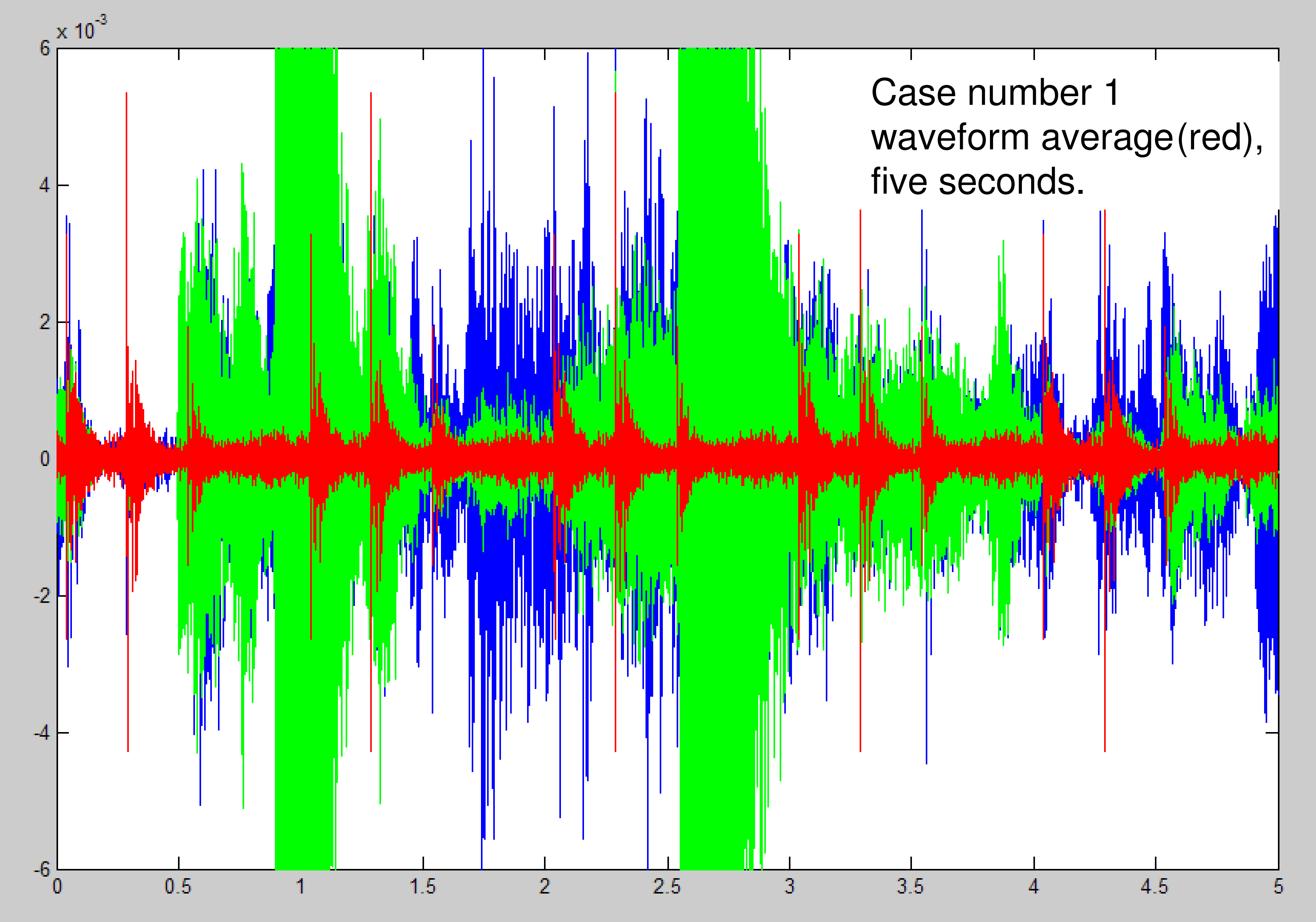
- Compression, that is, nonlinear dynamic range compression, not data compression.

I chose to use compression to handle the 100:1 dynamic range of the loud claps vs. the quiet pings.

A **median filter** (a filter that discards outliers) is a simple and effective way of eliminating impulsive noise such as claps. Its effect is similar to that of a compressor.

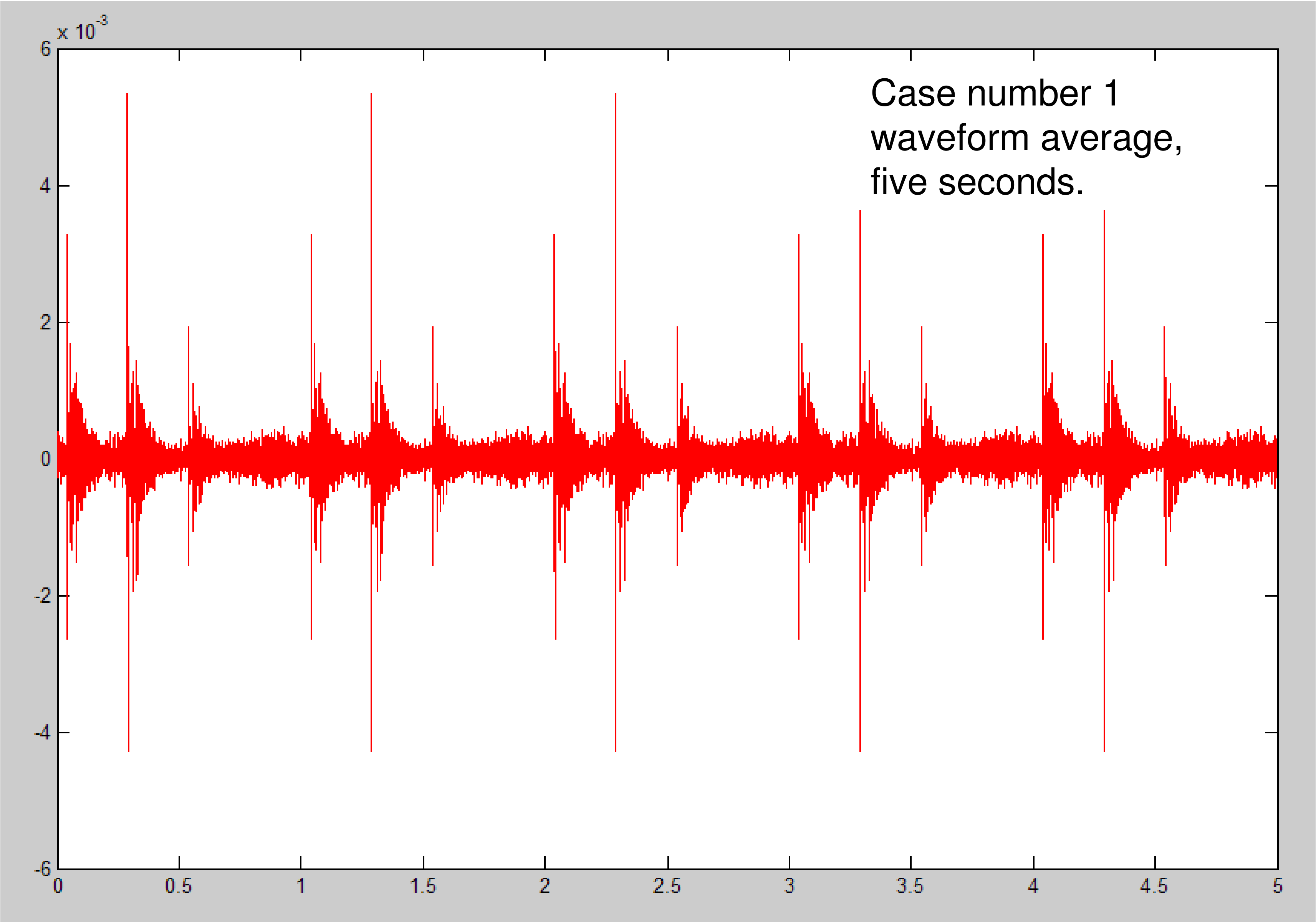
Correlation, autocorrelation, and cepstrum have no hope of working correctly unless the dynamic range is compressed first.
I don't use any correlation, but if you use it, you must do compression.

Three Beacons – How It Works: Waveform Averaging



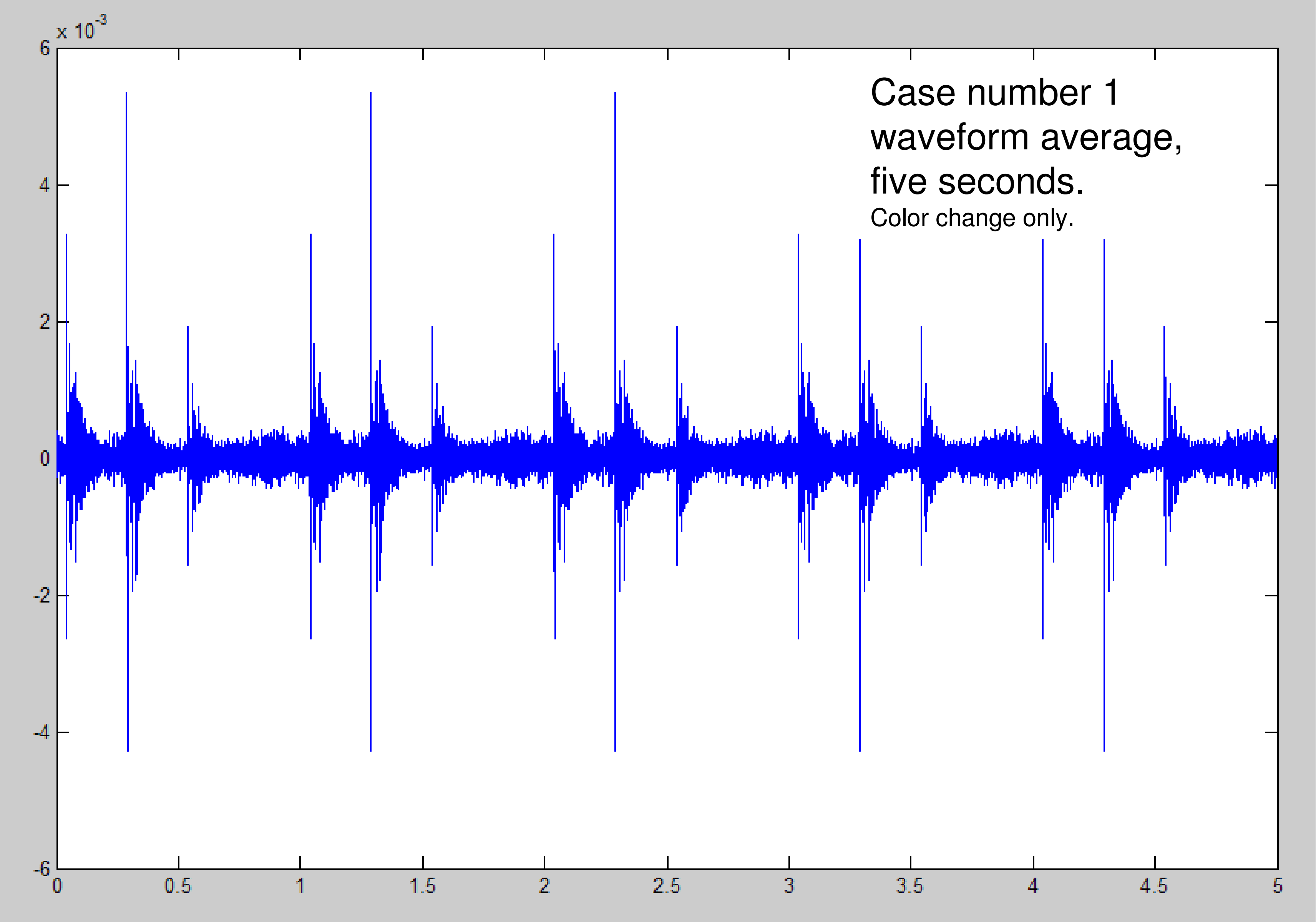
Case number 1 waveform average(red), five seconds.

Three Beacons – How It Works: Waveform Averaging



Case number 1 waveform average, five seconds.

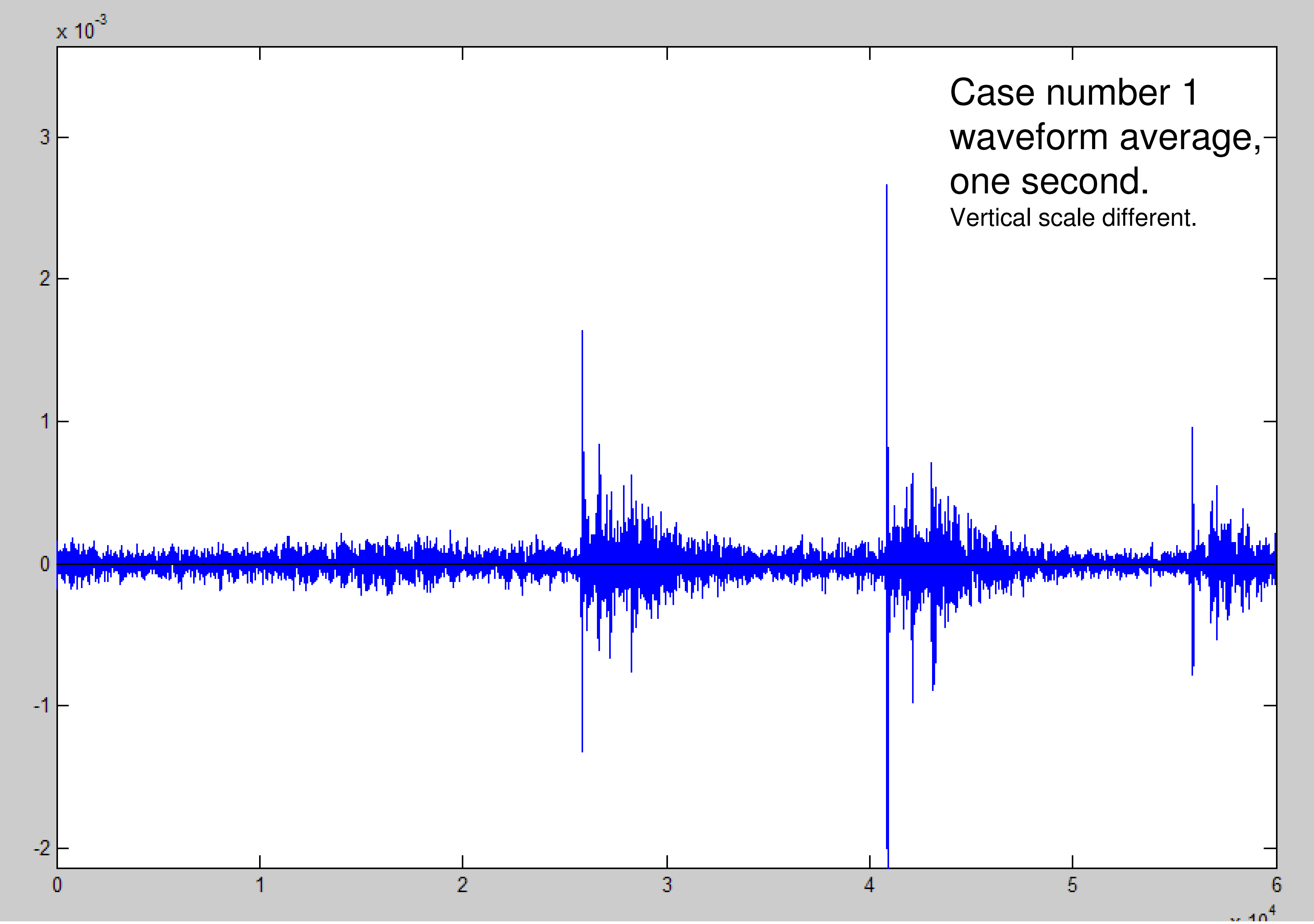
Three Beacons – How It Works: Waveform Averaging



Case number 1 waveform average, five seconds.

Color change only.

Three Beacons – How It Works: Waveform Averaging



Case number 1 waveform average, one second.

Vertical scale different.

Three Beacons – How It Works: Set Thresholds

By far, the hardest thing to do is to properly set the thresholds automatically. It is *extremely easy* to trigger on the *wrong* part of the waveform.

The waveform consists of:

- ambient noise,
- crosstalk,
- precursors,
- pings, and
- echoes.

Three Beacons – How It Works: Set Thresholds

I chose a threshold of at least two times the noise floor of the recorded sound (*not* the noise floor of the recording chain). Typically, one chooses *three* times, but I found that the results could be gathered faster with a lower threshold.

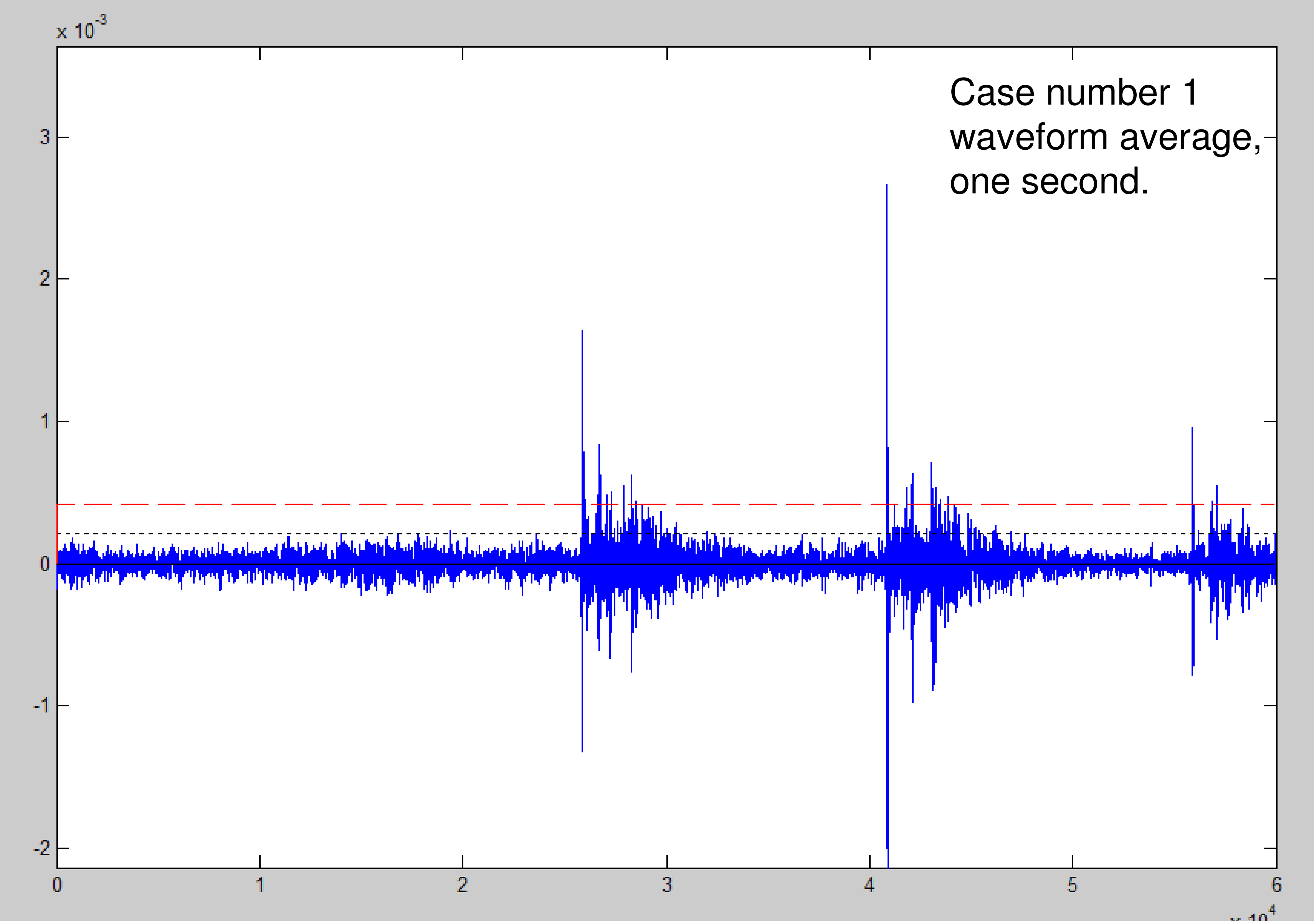
The huge dynamic range and the filter artifacts require special attention.
I had the threshold look ahead so that the threshold was always at least 10% of the peaks that were ahead of the threshold.

Sometimes, the thresholds are wrong, causing one of two things to happen:

- ➢ The beacons are not detected, or
- ➢ The calculated position is nonsense (e.g., negative radius for the big circle)

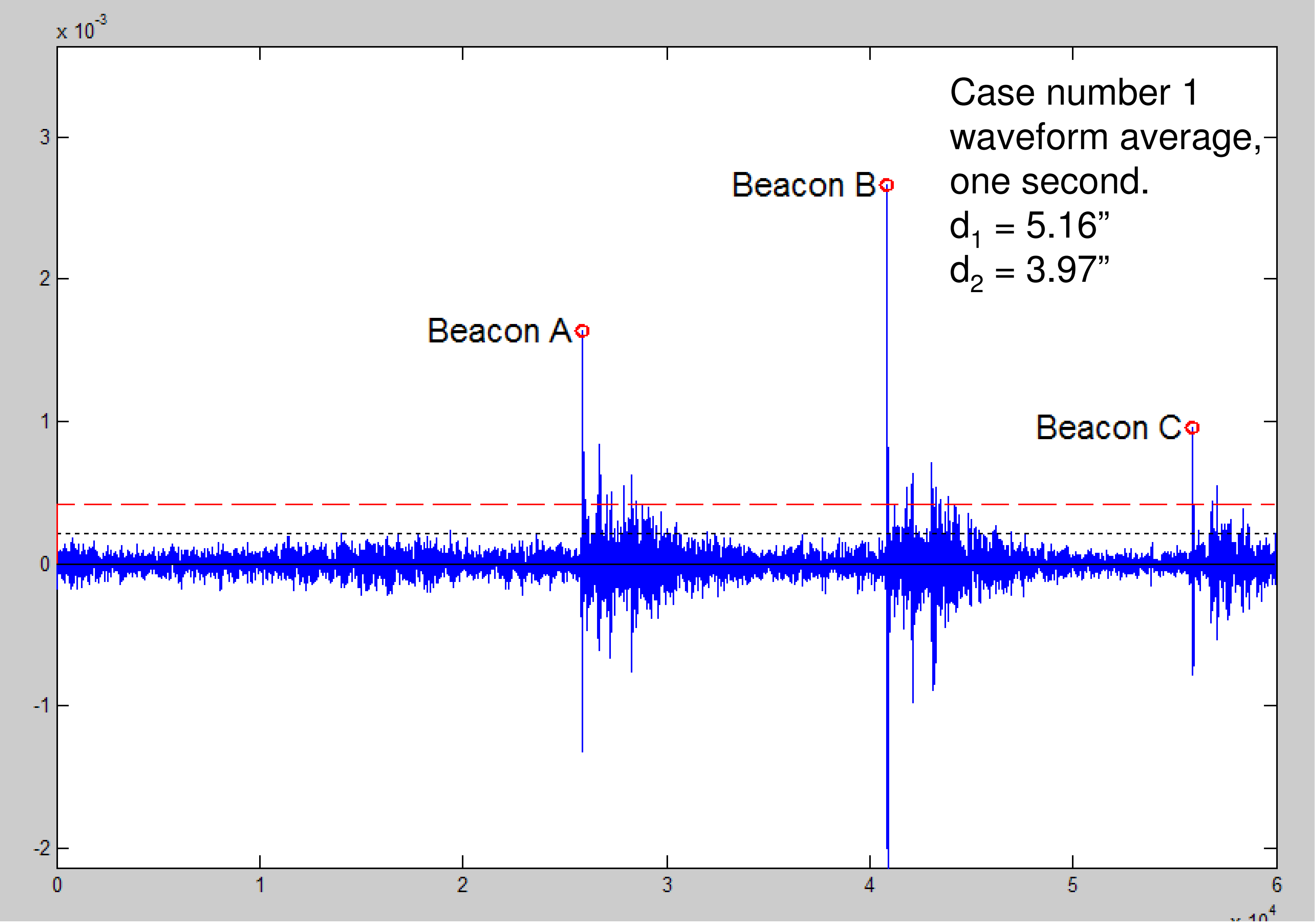
The solution to both of the problems is to increase the number of pings.

Three Beacons – How It Works: Set Thresholds



Case number 1 waveform average, one second.

Three Beacons – How It Works: Find Peaks

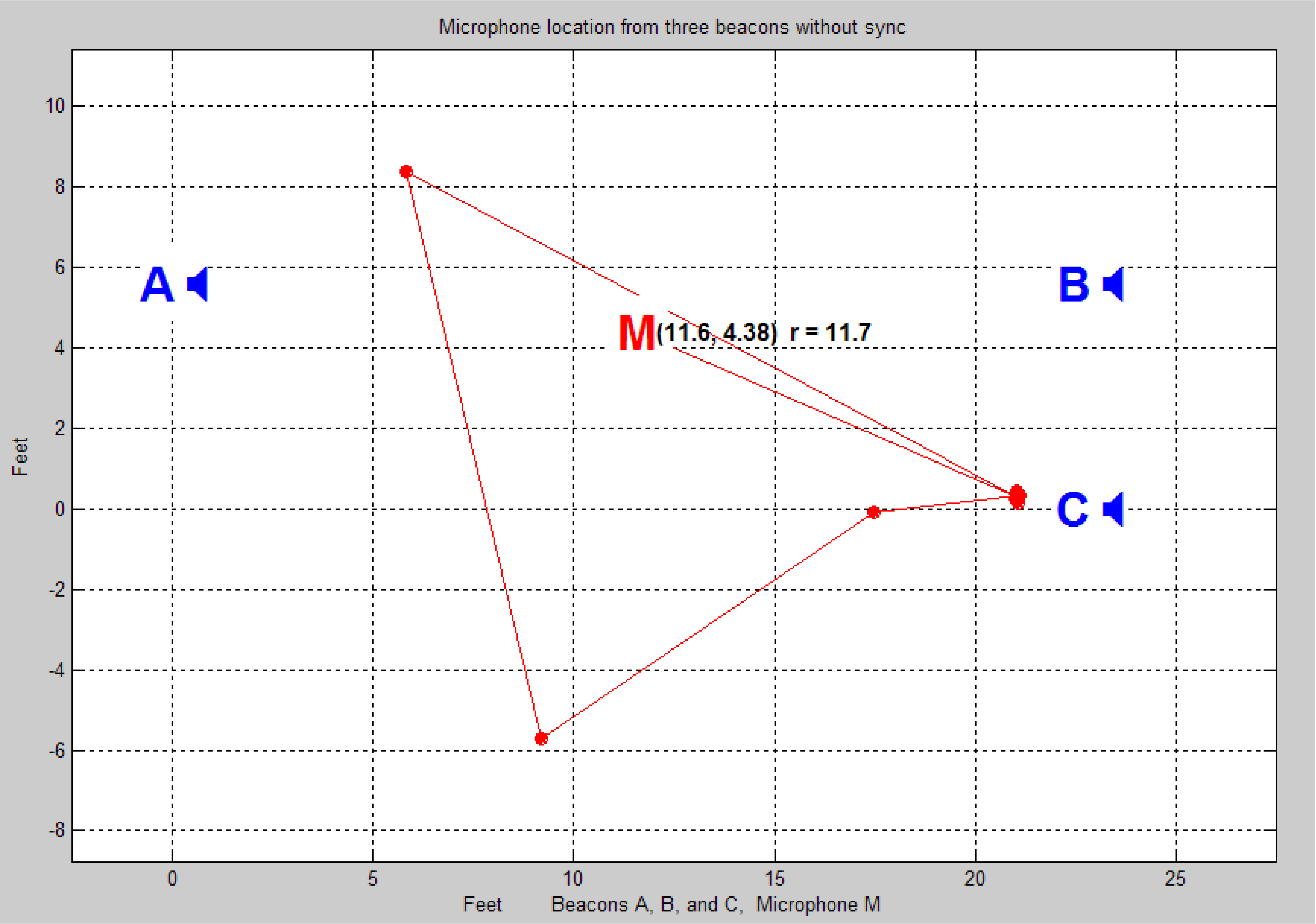


Case number 1 waveform average, one second.

d_1 = 5.16”

d_2 = 3.97”

Three Beacons – How It Works: Calculate and Display Location



Three Beacons – Results: Is It Any Good?

Pros:

- **Resolution** is superb, about a quarter of an inch (the sample rate).
- **Accuracy** is dependent on exact coordinates of the beacons, so accuracy can be very poor. I’ve seen errors of 4” to 15%. But, *repeatability* is far more important than *accuracy* for Square Dance and Figure 8. Also, accuracy can be corrected with 2D interpolation, or by correcting the beacon’s coordinates from known robot locations. (Written but not debugged yet.)
- **Repeatability** is empirically measured as about ***one inch.*** In a small arena (living room), the repeatability is the sample rate, 0.28”.
- Script rejects bogus results, so robot rarely goes beserk.
- Script tracks sample rate fluctuations automatically so the beacons run asynchronously, without any regard to the robot.
- Script automatically handles a **huge dynamic range** of noises and distances.
- It is cheap! (about $130)

Three Beacons – Results: Is It Any Good?

Cons:

- It is **SLOW** partly because of waveform averaging (lots of pings) and partly due to scripting language (most time wasted setting thresholds).
- **Robot has to be still**. This ***corrects,*** but does not ***replace*** odometry.
- Design criteria affected the quality of the result. (But it's CHEAP!!!)
- Measurement errors have a huge impact.
- I believe that the negative SNR (and thus the long measurement times) makes it impractical to use a Kalman filter to do on-the-fly localization, but that would be cool.

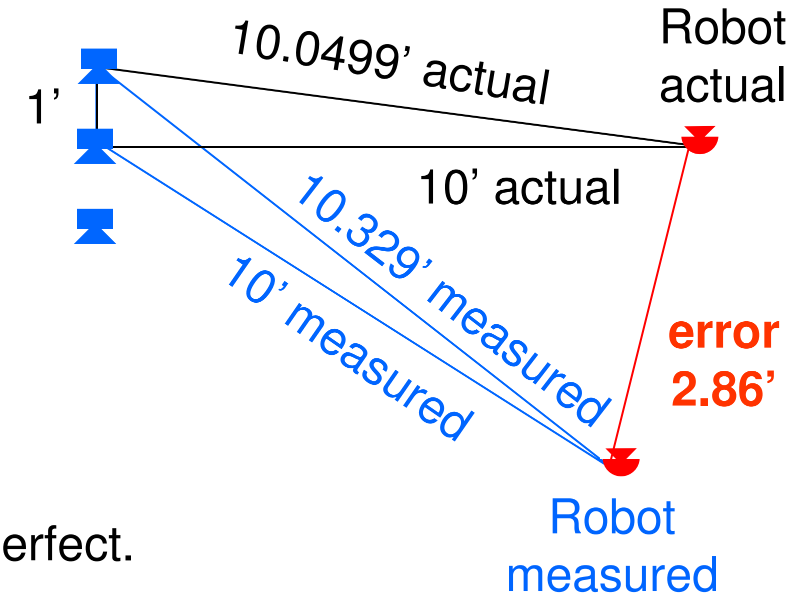
Three Beacons – Measurement Errors

MEASUREMENT ERRORS HAVE HUGE IMPACT

At certain angles:
a sample that is merely ONE sample too soon or late in time
can cause an error of SEVERAL FEET in the robot's reported position.

One audio sample at 48,000 samples per second represents
a distance of only about one quarter of an inch (0.28").

For example,
if the beacons are one foot apart,
and the robot is ten feet away,
forming a right triangle,
then a ***one-sample error*** in d_1 or d_2
results in a ***THREE FOOT error***
in the position of the robot!



The measurements have to be almost perfect.

Three Beacons – Measurement Errors

MEASUREMENT ERRORS HAVE HUGE IMPACT

On the other hand,
a **one-sample error** causes only a **one-half-inch error**
when the beacons and the robot are collinear (lined up).

(BTW, if this were triangulation, you couldn’t allow the robot
to be collinear with the beacons. You wouldn’t get *any*
distance information. But this is trilateration, not triangulation.)

Three Beacons – Measurement Errors

- ***Quantization*** (sampling)
    - Voltage errors
    - Timing errors

      See the slide “Nuances: Matching Sample Rates”

- ***Noise***
    - Electronic (see example below)
    - Ambient

All of these sources of errors are handled by doing

- filtering,
- waveform averaging, and
- compression.

The *electronic noise* in a mic circuit can be surprisingly terrible on consumer gadgets. Using an external 59 cent mic connected to the microphone input jack of one external USB gadget, I saw about -24dB noise (terrible), even when the mic was disconnected. With the same mic, I saw -60dB noise (good) in my laptop’s built-in mic jack. It would take hundreds of pings to overcome that 48dB difference. (It’s not a 5v FET bias problem. The mic worked on both inputs. The noise was still there when the mic was disconnected.)

Three Beacons – Measurement Errors

- ***System response***

This is handled by carefully choosing the ping,
and keeping the beacons more than the ping's width from
reflective surfaces.

I don't believe it is practical to do anything more than that.

It is not practical to deconvolve the system response
out of the results. To do so would require knowledge of the
responses at *every* angle that the beacon can be with respect to the robot.
Even then, all the possible angles would have to be checked on every pass
to see which gave the most likely response.

I considered this to be completely impractical, and it turned out that
it was not necessary.

Three Beacons – Measurement Errors

- ***Distortion*** due to overdriven electronics
  - The output amplifiers can be overdriven.
  - The speakers can be overdriven (buzzing, cone whapping).
  - The microphone can be overdriven.

Distortion causes a sample or two of error in triggering on the beacons.

It’s easy to prevent the output chain from being overdriven.

The huge dynamic range makes it somewhat more difficult to guarantee that the microphone is not overdriven by sounds that are too loud.
Keep the microphone several feet from beacons if the beacons are really loud.

It does not matter if noises or echoes are distorted. But a ping recording can be clipped because of noise, and that will damage the waveform average.

Distortion can be measured directly by emitting a sine wave long enough to ensure that the environment is completely filled with echoes, and then comparing the peak to the RMS of the recorded sound. But this overconstrains you. Since the echoes are 10x to 20x as loud as the direct-path sound, the echoes will distort even when the direct-path sound is not distorted, and all you care about is direct-path. I don’t have a fix for this.

Three Beacons – Measurement Errors

- ***No two beacons are the same***, so don’t depend on the exact shape of the waveform coming from any beacons.

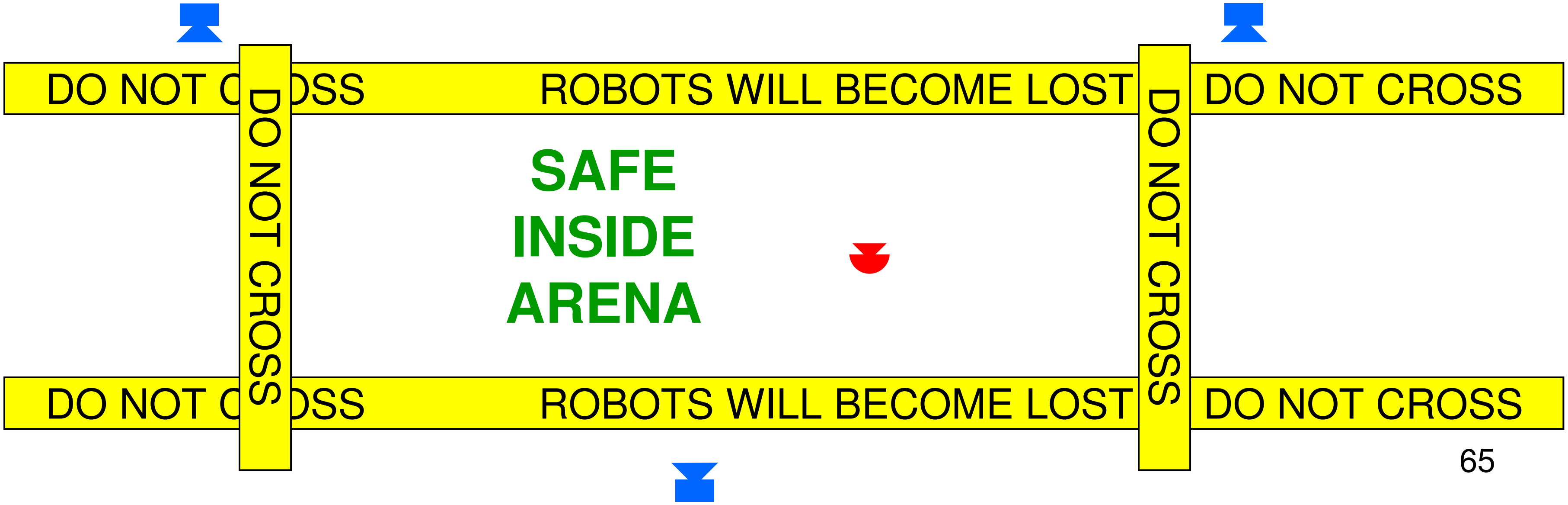
- ***Nothing is truly omnidirectional***. The only cure I can think of is to build special omni speakers, and that violates one of the design criteria.

- ***Obstacles*** between the beacons and the robot might affect the arrival time and the ping waveshape. The bottoms of tables certainly distort the waveshape.

Three Beacons – Recommended Usage

Recommendations to keep errors from being such a problem:

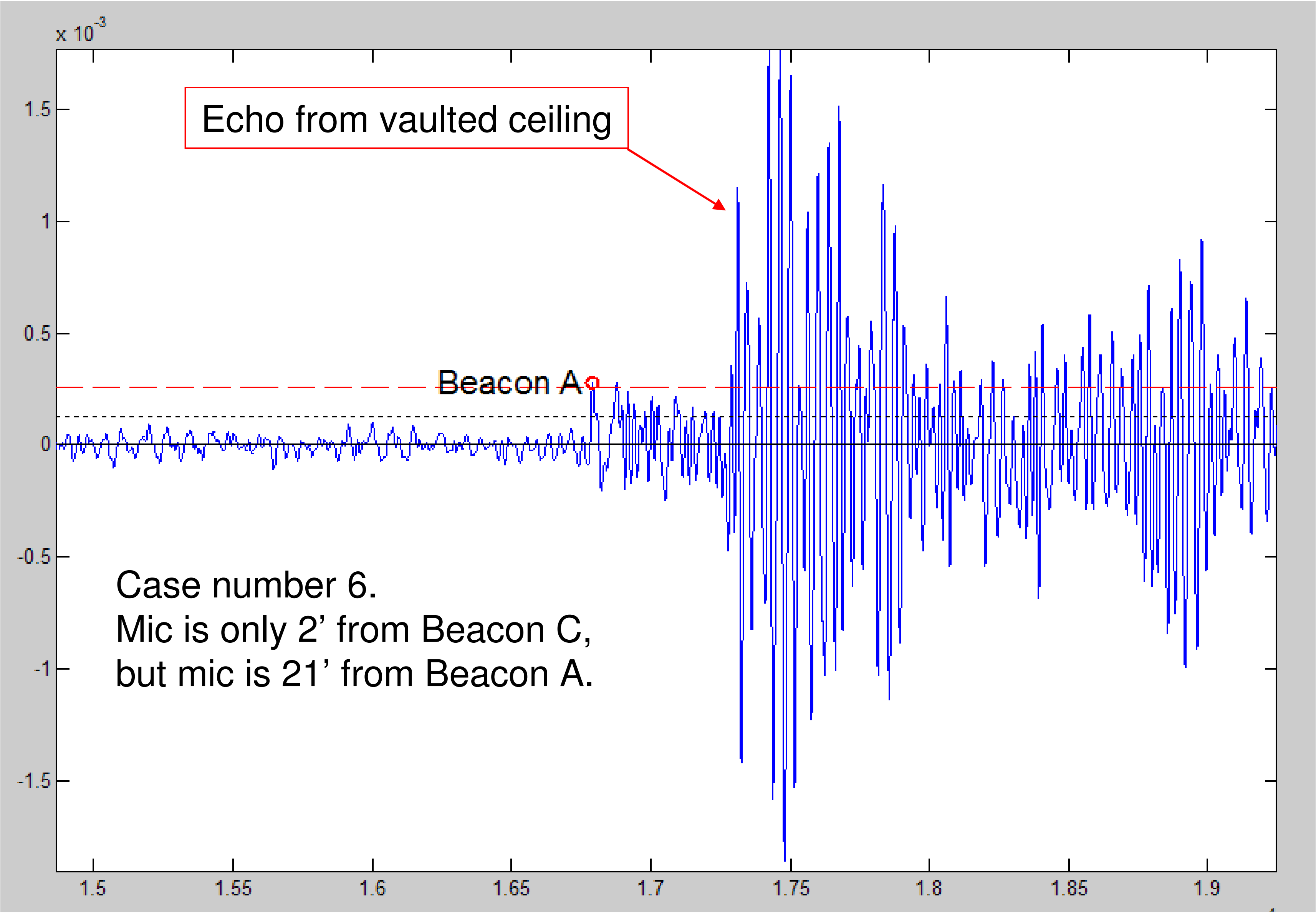
- Only allow the robot to operate in the safe area (the arena).
- Keep the beacons at least *one ping's width* from vertical reflective objects.
- Keep the *robot* at least one ping's width from the beacons.
- Define the arena to be a box that the beacons are *outside of.*
- Calibrate the reported robot locations to known locations, or depend on the *repeatability* of the script instead of depending on *accuracy.*
- Arrange the beacons in an equilateral or an isosceles triangle, not collinear.
- If the beacons are collinear, keep the robot on one side of the beacons.



Three Beacons – Nuances: Echoes are Louder than Direct Sound

- **Echoes (multipath sounds) are louder than direct-path sounds**
  - Nobody believes me, so look at the waveforms.
  - There's only one pathway for the direct-path sound, but there are multiple pathways for the echoes.
  - The echoes can reinforce one another.
  - A corner yields an echo that is 8x as loud as the direct-path sound.
  - With another corner, the reinforced echoes can be even louder.
  - I often see echoes that are 10x as loud as the pings.
  - This multipath anomaly is being exploited by some cell phone signalling protocols that depend heavily on the RF echoes off multiple buildings in order to achieve a strong enough signal. So, believe me or not, your cell phone is probably listening to echoes of the base station signal, ***not the direct-path signal!***

Three Beacons – Nuances: Echoes are Louder than Direct Sound



Echo from vaulted ceiling

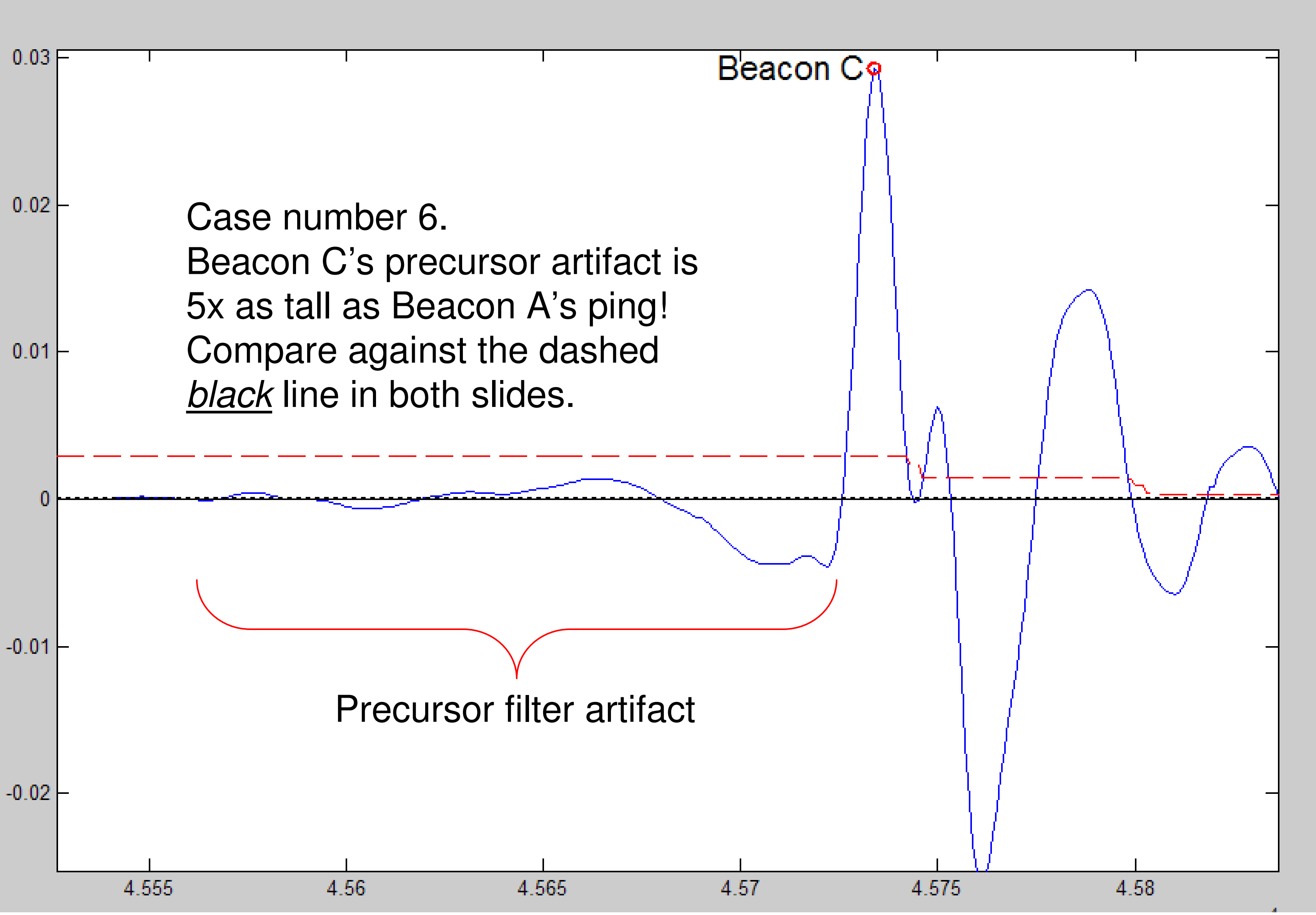
Beacon A

Case number 6.
Mic is only 2’ from Beacon C,
but mic is 21’ from Beacon A.

Three Beacons – Nuances: Filter Artifacts

- Filter artifacts
  - Filters are useful to allow the script to trigger correctly on pings and not on ambient noise or drift.
  - The tighter the filter, the more it distorts the signal.
  - High-pass filters create **precursors**.
    (I *made up* that name because I don't know what the correct name is. In multi-tap equalizers, "precursor" has a different meaning.)
  - Precursors can be one-tenth as tall as the desired ping.
  - Low-pass filters squash the ping, making it wider and thus requiring that the beacons be farther from the walls.
  - The real world (the speakers, the microphone, the room, the air) filters the sound whether you intend it to or not, so the artifacts are there no matter what you do, but you can make the situation worse with a bad filter.

Three Beacons – Nuances: Filter Artifacts, Precursor



Case number 6.
Beacon C’s precursor artifact is 5x as tall as Beacon A’s ping!
Compare against the dashed *black* line in both slides.

Precursor filter artifact

Three Beacons – Nuances: Noise Filters Won’t Work

Noise filters, noise gates, multi-band noise gates, and adaptive noise filters depend on positive signal-to-noise ratio in the frequency bands that they filter. Due to the design criteria, the beacons are often quieter than ambient noise. Thus the “noise profile” is louder than the beacons, and the noise gate squelches the pings. Indeed, for Case 1, Audacity’s noise removal tool (a multi-band noise gate) cannot distinguish the pings from the radio show. That means that either nothing is removed, or that only the claps remain. If the pings were louder than the radio show, the noise gate would help some, but it would still pass the claps. A pop filter might help with the claps.

Noise gate artifacts sound spooky. Good for Halloween.

Why not choose ping frequencies where there is no ambient noise?
There are no such frequencies. It is a conceit for us to think that there is no noise at 40kHz (the frequency most often used in hobbyist sonars).
We just can’t hear the noise. There’s plenty of noise around 40kHz.
(At 40kHz, the sonar can’t “see” soft things (like couches) because they readily absorb high frequencies. Beacons are not sonar, so this isn’t the problem.)
The problem is: High frequency speakers are directional (which is OK for sonar), but the beacons require omnidirectional sound.

Three Beacons – Nuances: Matching Sample Rates

- Sample rates never match
    - Even if the software sets the sample rates to be *exactly the same*, **there's no way to force the sample rates to be the same unless all the A-to-D converters are slaved to the same clock**. This is exactly what's done with professional audio interfaces, but consumer interfaces do not have the capability to guarantee the same sample rates between gadgets.
    - **The USB spec allows all isochronous transactions to gain or lose one sample per millisecond**. Poorly-designed gadgets allow this gain or loss to accumulate whereas well-designed gadgets correct the error every millisecond.
    - Windows messes up the rate. Windows is not a RTOS; don't expect it to guarantee the rate. For example, power management, such as display dimming, causes thousands of samples of error.
    - Waveform averaging is completely worthless unless the sample rates match exactly.
    - The script has a sort of software PLL that detects the actual rate and corrects for it. This is absolutely crucial for correct results.

Three Beacons – Nuances: Correlation Cannot Work

The common wisdom is that correlation (and autocorrelation, next slide) is the best way to process anything related to sonar or radar. Well, I’ve tried over and over to use correlation and I’ve concluded that **correlation is worthless whenever any of the noise is similar to the signal**.

- **Correlation cannot work when the signal waveshape is corrupted by echoes of itself**. This happens when the beacons are close to walls. Correlation cannot even detect that any part of the signal is present.
- **Correlation cannot work when the echoes are louder than the pings**. Correlation selects the echoes instead of selecting the pings.
- **Correlation cannot work when the signal is filtered harshly** such that the filter creates self-similar artifacts, precursors. Correlation selects the precursors instead of selecting the pings.
- **Correlation cannot work when the cheap electronics suffer from crosstalk**. The crosstalk is a perfect, noiseless copy of the signal. Correlation selects the crosstalk instead of selecting the pings.
- Although correlation could be used to perform the waveform averaging, it is a horribly inefficient way of doing it, wasting lots of time and memory.

Correlation is completely worthless for this application.

Three Beacons – Nuances: Autocorrelation

Autocorrelation is correlation where the two inputs to the correlator are the same signal. Autocorrelation can detect repetitive signals. It might seem that since the pings are repetitive, that the pings could easily be detected with autocorrelation.

The recording of the sound from any one of the three beacons is *not* similar to the sound from either of the other two beacons, due to polar response patterns of the three speakers and the microphone, and due to 1/r loudness and the 10x louder echoes.

So, autocorrelation *cannot* give *any* information about the ping separation (which is used to determine DTOA, that is, d_1, and d_2). Autocorrelation (or cepstrum) only provides the rep rate, which is already known and is not something that needs to be measured. It is one second in these examples. (Autocorrelation or cepstrum can help with calibration of the system, but they are useless during runtime.)

Three Beacons – Nuances: Crosstalk

Crosstalk is a *huge* problem with cheap consumer electronics.

Crosstalk can be louder than the ping from a distant beacon!!

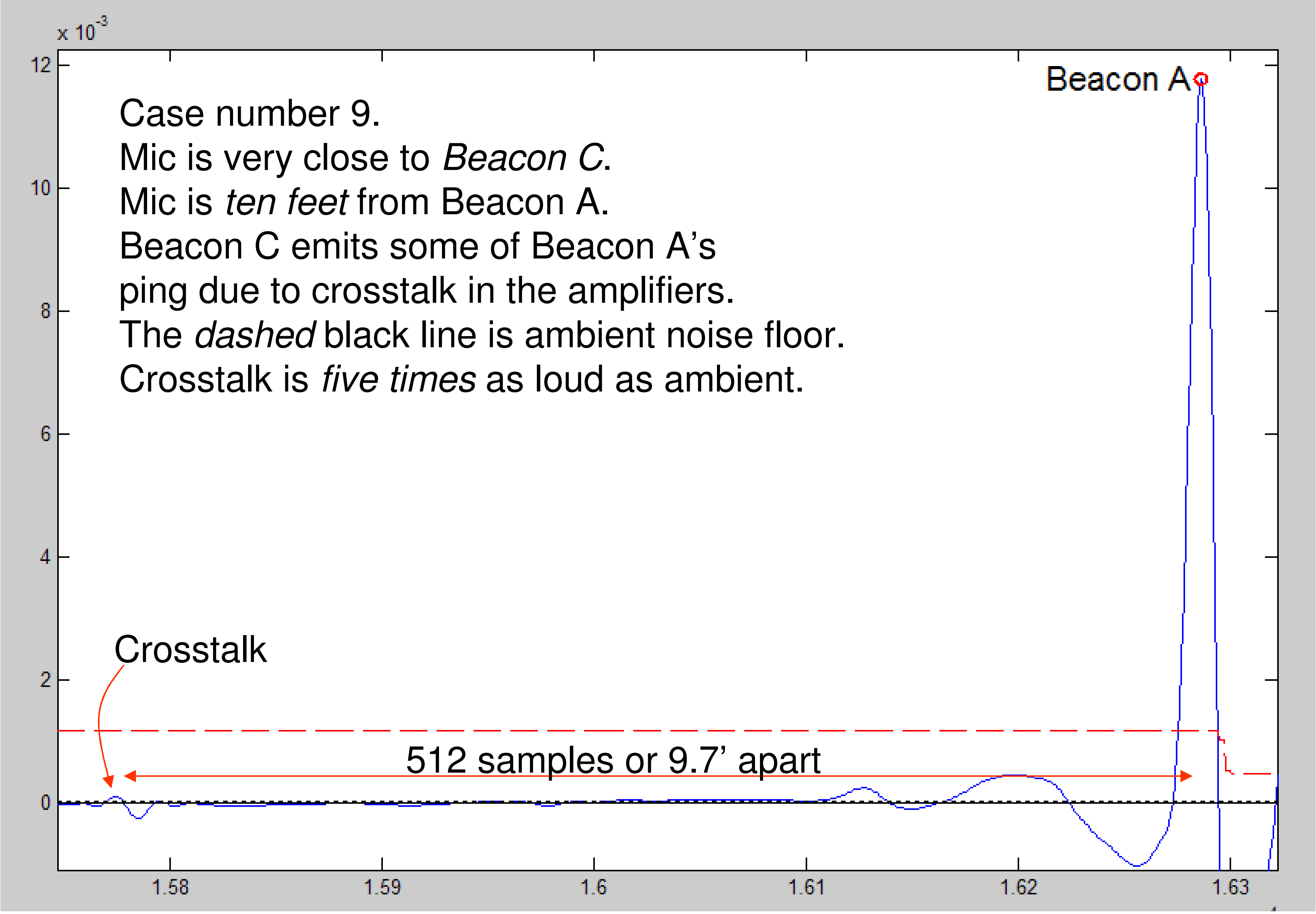
In the cases where I used the same audio device for both input and for output, I saw crosstalk *even when the speakers or microphone were* ***disconnected.*** This shows that the crosstalk is entirely electronic.

To be realistic (that is, robot independent of the beacons), all scenarios shown here use separate audio input and output devices, so all of the crosstalk is where one beacon's *output* channel affects another beacon's *output* channel, and then the crosstalk is carried as sound through the air.

Crosstalk makes thresholding difficult, especially when the arena is quiet. Crosstalk is often four times as loud as ambient, so 3x (three sigma) thresholds don't work. This is one reason why I say that thresholding is the hardest part of it all. (It's embarrassing to trigger on a signal that came from the *wrong beacon*!)

Correlation is fooled into selecting the crosstalk instead of the ping.

Three Beacons – Nuances: Crosstalk



Case number 9.
Mic is very close to *Beacon C*.
Mic is *ten feet* from Beacon A.
Beacon C emits some of Beacon A’s
ping due to crosstalk in the amplifiers.
The *dashed* black line is ambient noise floor.
Crosstalk is *five times* as loud as ambient.

Crosstalk

512 samples or 9.7’ apart

Three Beacons – Nuances: Handling Large Dynamic Range

- Dynamic range is a real pain
  - See the “How It Works” slides for an example of extremes.
  - When the microphone is close to one beacon and far from another, the difference in the measured amplitudes of the pings can easily be 20:1. The **theoretical ratio is the inverse of the distances**. If the mic were 3 feet from one beacon and 30 feet from another, the ratio of amplitudes should be 10, but I measure much greater differences (about 20:1), probably because the polar response pattern of the speakers is uneven.
  - The walls and corners concentrate echoes, making echoes far louder than direct-path sound. Often I see echoes 20x as loud as pings.
  - I recommend that the mic be kept several feet from beacons, but, in most cases, the script correctly triggers even when the mic is next to a beacon. In those cases, the ratio is about 100:1.
  - ***A hand clap or a shout is easily more than 100x as loud as pings***.
  - It was a real challenge to reliably trigger with such large changes in amplitude within a few milliseconds, but the script does it.

Three Beacons – Lessons Learned

Education was my #1 goal. I want this to be educational to you, so I'll try to help with some of the concepts (the lessons learned). I enjoy teaching.

The most important lessons are:

- *Sample rates* must match. They must be *exactly* the same, not just close.
- *Dynamic range compression* is extremely useful.
- *Filtering* is much *less* important than you might think.
- *Correlation* is useless.
- *Cheap stuff* does work. You can afford superb localization at home.

Other lessons:

- *Use cheap speakers*, speakers with single drivers in small cabinets. Large drivers, more drivers, and large cabinets smear the beacon's location.
- *Use cheap microphones*. They are omni. PA mics are cardioid. Won't work. Nice recording mics have big diaphragms and they won't work either.
- *Acoustically isolate the mic*. Noise is not a problem, but *coupling* is. Coupling smears the apparent location of the robot.
- *Don't point up*. If you make a speaker omni by pointing it up, you'll make indoor echoes worse. Outdoors, you'll lose all your sound in the sky.
- And all of the "Nuances" described in previous slides.

Three Beacons – What's Next?

- Socket communication (Wifi, internet) between localizer and robot. Collaborate to win Square Dance and the outdoor contest.
- Speed it up *a lot*. Compile it. Allow time for lots more pings, to handle lots more noise.
- Weatherproof it. Put it on a lawnmower.
- Productize it?

Can you get a copy of the script? Probably not.

My script is for me. Write your own. Most scripts that I get from elsewhere are junk. Seriously. You waste your life while debugging other people's code (so why should you debug my code?). Learn by writing your own script so that you understand what it does. Many programmers seem to write elegant-looking code that meets the source code control system requirements, but which fails to handle corner conditions. In other words, code that looks good but doesn't work. If you want to see my script, we'll need a NDA. (Script has 1,450 executable lines in it.)

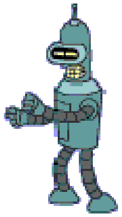
I hope you give me credit if you patent something from what you learned here.

Have fun, and learn something while you're having fun!

Robot Location
Using Three Omnidirectional
Sound Beacons

John Swindle

swindle@compuserve.com



September 2010