



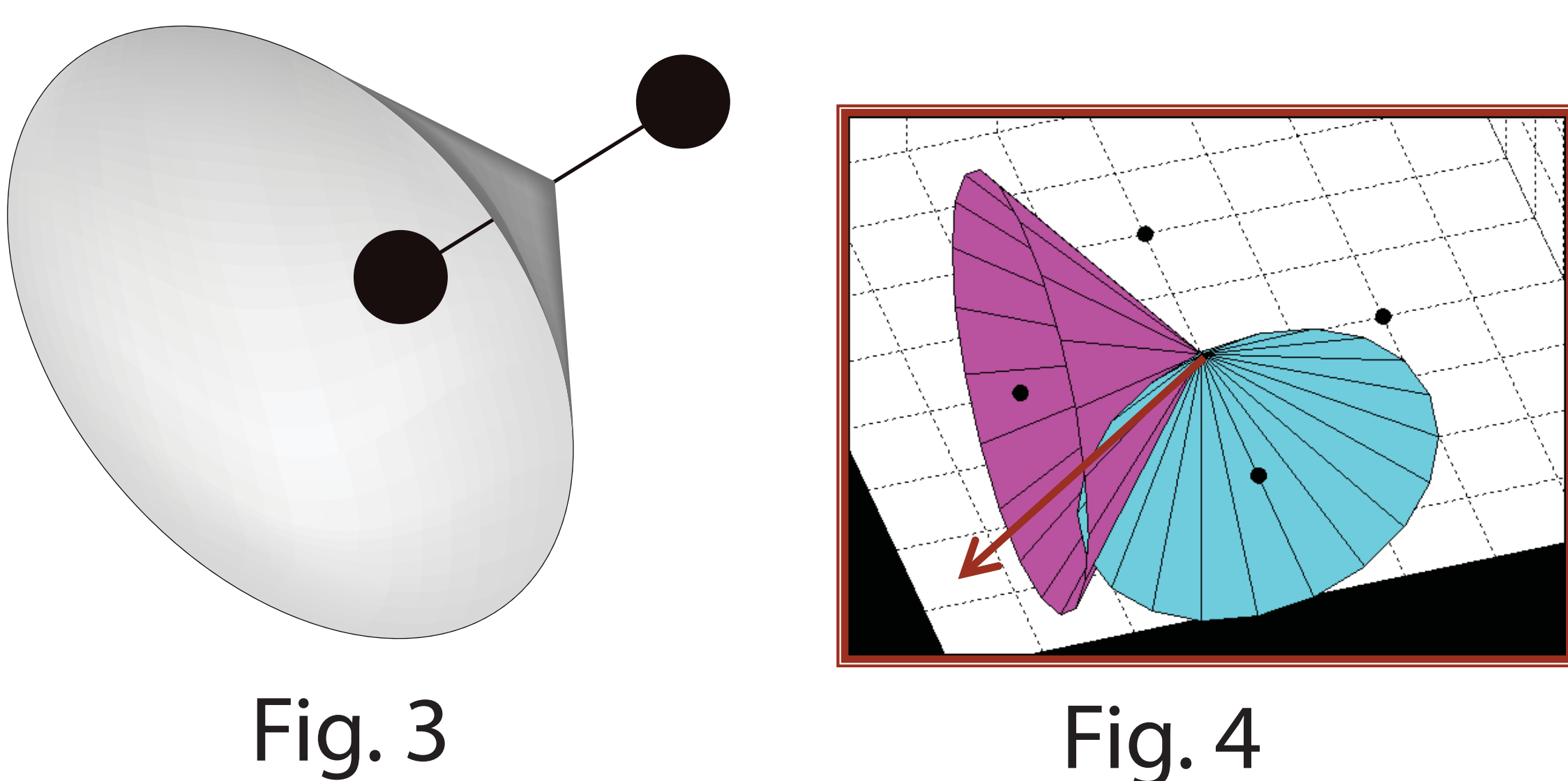
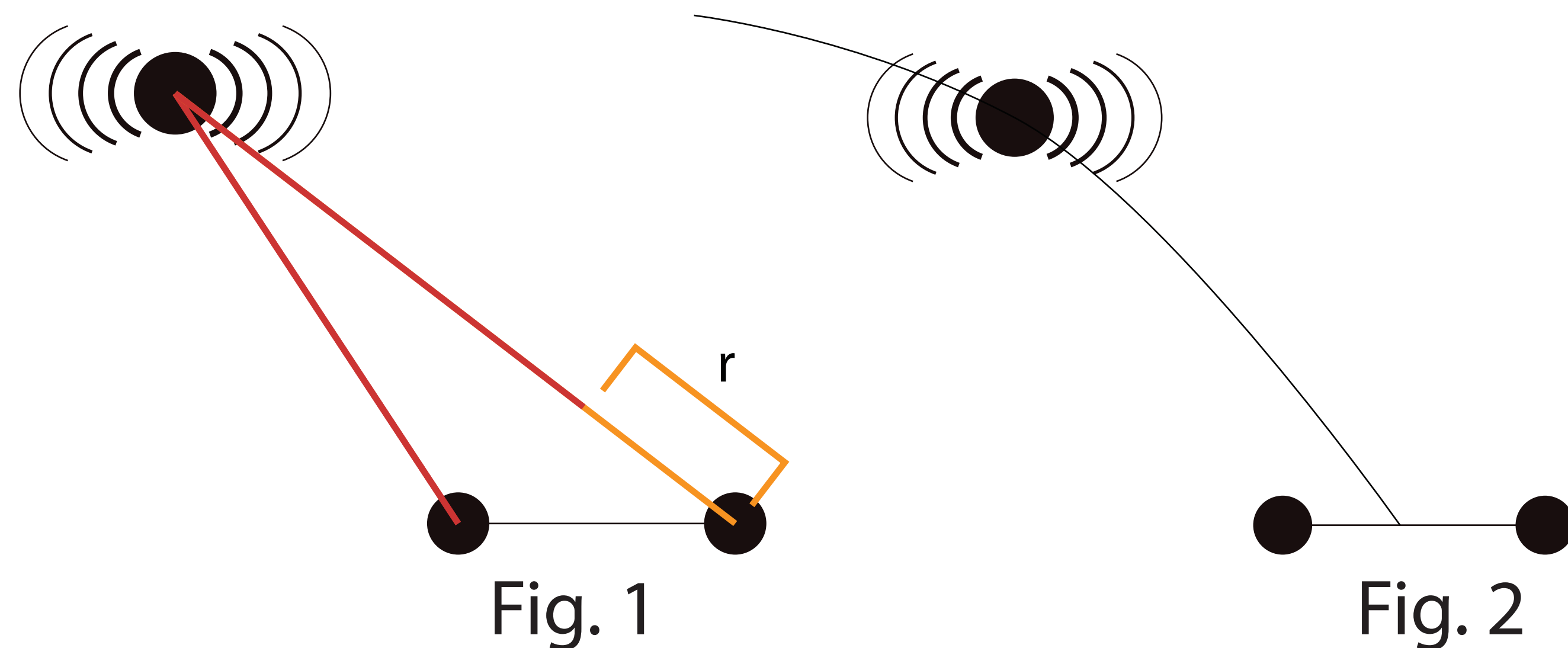
Abstract

We propose a distributed algorithm for single acoustic source localization. The algorithm is based on the local estimate of time-difference-of-arrival at pairs of sensors, and the distributed implementation is based on a method of successive projections upon bearing lines. Convergence of the algorithm is discussed and an approximate closed form expression is given in the projection problem. The computational complexity is suitable for use in real-time application, where data collection in a central node may be infeasible.

Time Difference of Arrival (TDOA)

For a given sound source and pair of microphones, there is an additional distance r that the sound wave must travel to reach the second microphone (Fig 1). For a known speed of propagation c , this can be found from the time difference of arrival (TDOA), denoted by τ . Because r and τ are proportional to each other, we will use τ in our calculations.

The TDOA corresponds to a hyperboloid of possible source locations (Fig 2). However, when the spacing between the microphones is relatively small compared to the distance to the source, we can use a cone to approximate the hyperboloid (Fig 3). This simplifies our calculations. When we place 2 pairs perpendicular to each other, we get a bearing line (Fig 4). We then use successive projections across multiple bearing lines to find the source.



Methodology

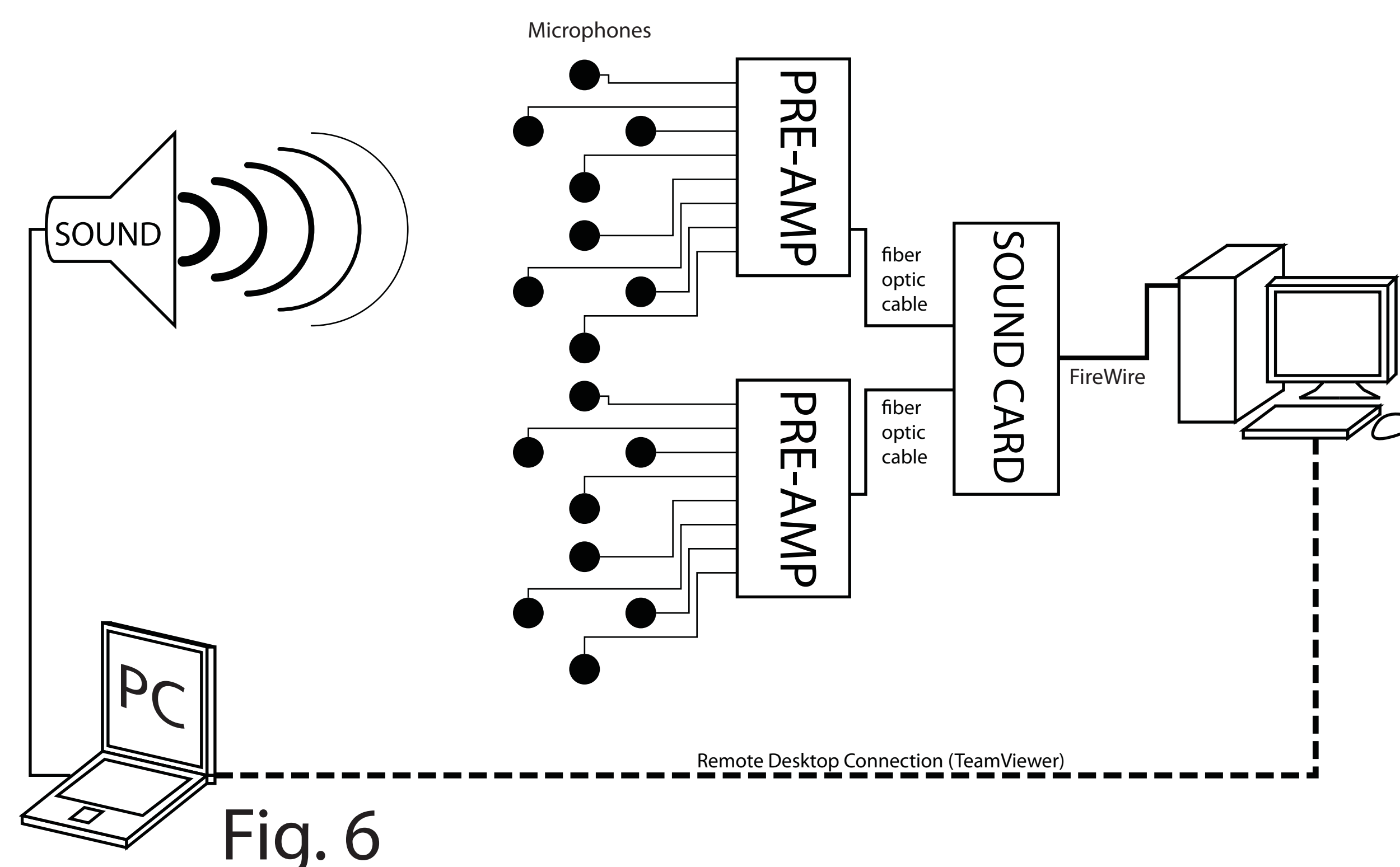
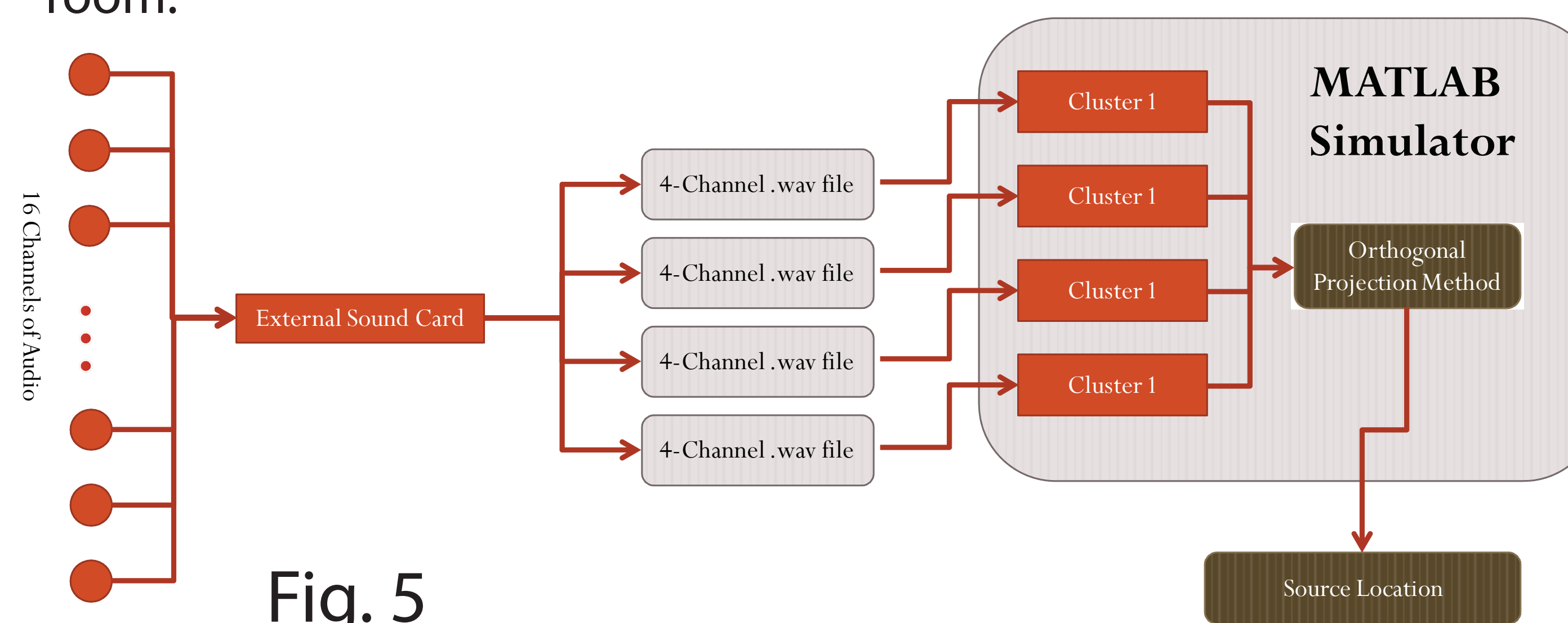
We will use a capture-calculate testbed to perform localization experiments. This testbed cannot localize in real-time, meaning that we cannot assess what impact factors such as synchronization and wireless communication would have on the system. What it does provide, however, is a “best-case” test of both systems. Figure 5 shows a conceptual flowchart, and Figure 6 shows the hardware schematic. The system has 2 major components: Capture and Calculate.

Capture

I have 16 digital microphones arranged in 4 clusters. The 16 microphones connect to two 8-port pre-amplifiers, which connect to an external sound card. The sound card then connects to a PC via firewire. In this way, we can record 16 channels of audio (one for each microphone) simultaneously.

Calculate

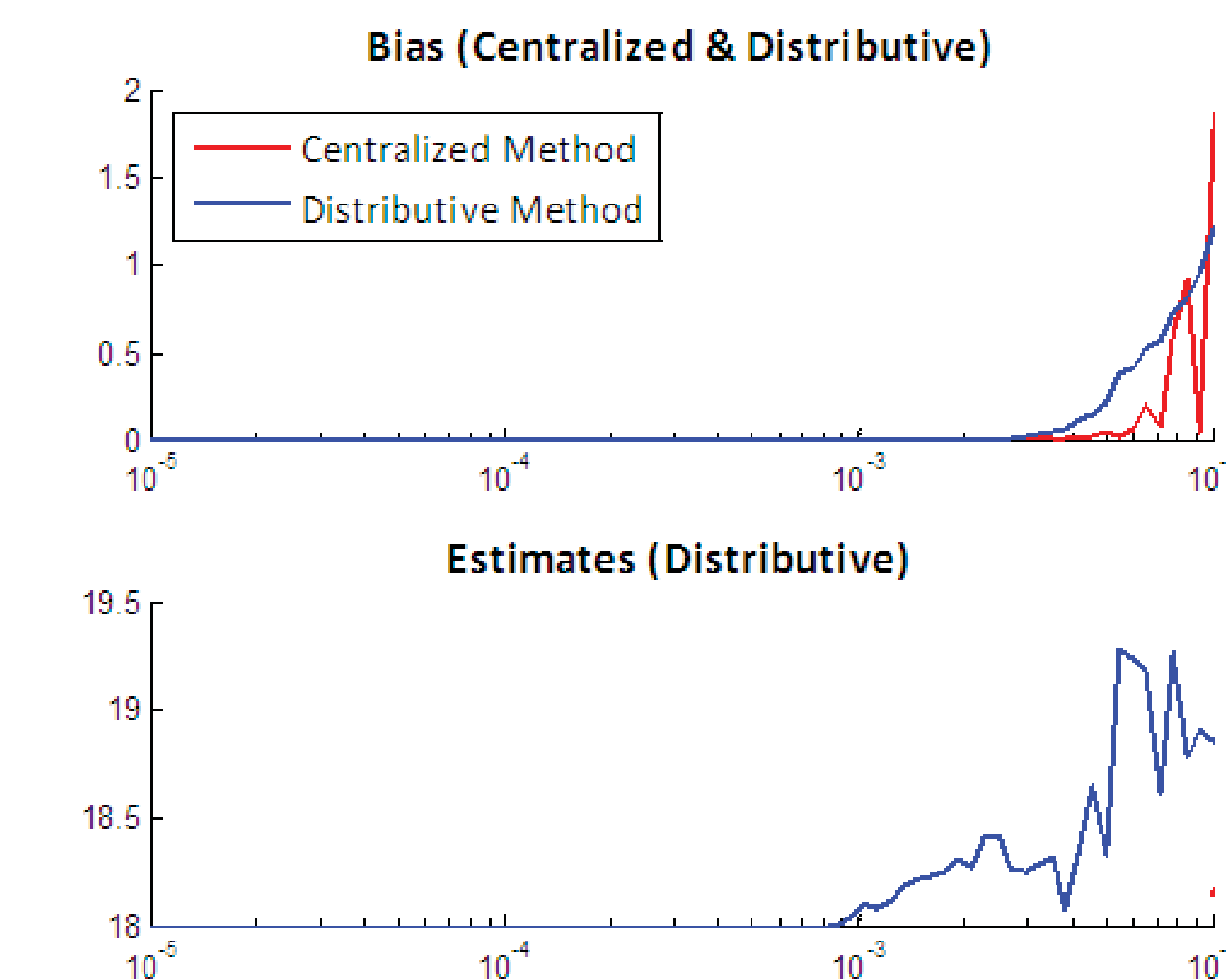
I organize each file such that it contains all the audio captured from a given cluster. MATLAB then opens the files, calculates the time delay using a cross-correlation, generates the bearing lines, and estimates the location of the source. I calculate the bias—the straight-line distance between my estimate of the source location and my measurement of the source location with respect to the room.



Results

The initial results are very promising. The bulk of my work lies in MATLAB, where we do theoretical simulations with noise. Figure 7 shows one such simulation. Each datapoint of the graph represents the average bias from 100 trials with the given level of noise. As the level of noise increases, so does the bias. However, the simulations show that the distributive method I am using is comparable to a centralized method where one computer would calculate everything—in theory, at least.

I also managed to build a working testbed and I have attached the results of my 2 tests, each made up of 3 trials. I resulted in an average bias of 4.3 cm. Figure 8 shows a table with my results.



Bias (m)	Projections	
0.0287	28	Test 1
0.0207	28	
0.0207	28	
0.0626	48	Test 2
0.0626	48	
0.0626	48	
0.0430	38	Averages

Fig. 8

Fig. 7

Further Work

In my first round of tests, I was limited to 4 clusters of microphones. In my further work, I want to know how the number of microphone clusters will affect the performance of the algorithm, and if at any point., additional microphones would slow down the system. Pure acoustic localization is not as robust as a real-world system would demand. While this is a good exercise in theory, further work would attempt to pair this system with a beamforming or video tracking system--depending on the application.

