CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

Synchronized Audio Capture with an Array of Mobile Devices

A thesis submitted in partial fulfillment of the requirements

For the degree of Master of Science in Software Engineering

By

Nick Schaafsma

May 2016

The thesis of Nick Schaafsma is approved:

Michael Barnes, Ph.D.	Date
Adam Kaplan, Ph.D.	Date
Ani Nahapetian, Ph.D., Chair	Date

California State University, Northridge

Preface

Loop Multitrack Recorder is a project that I first started working on in a summer course offered at California State University Northridge called Introduction to Software Engineering. When the course ended, I found that I really enjoyed working on the project so I continued working on the project for fun. A year and a half later, the project now allows you to enjoy most features that you would expect from a mobile recording app (graphical sound editor, apply effects, undo/redo, import/export to Google Drive or the device, etc.). The newest feature, for which this thesis primarily concerns, involves creating a "Jam Session" where multiple users can get together to record sounds on their own devices in a synchronized and collaborative manner. For screenshots and more information about the app please visit Loop Multitrack Recorder [1] or search "Loop Multitrack Recorder" on an Android device in Google Play.

Acknowledgements

I want to thank my parents Susan and Michael Schaafsma for all their love and support. I want to thank my advisor Ani Nahapetian for guiding me throughout the process, giving me ideas & support, and helping me stay focused. I want to thank my boss Mike Boctor for letting me take off work to work on my thesis. I want to thank my teachers for all of their hard work and creative lesson plans. I also want to thank my brother, Ryan, for inspiring me to become an engineer.

Table of Contents

List of Figures

List of Tables

ABSTRACT

Synchronized Audio Capture with an Array of Mobile Devices

By

Nick Schaafsma

Master of Science in Software Engineering

The objective of this project is to measure the accuracy of using a high decibel sound as a reference for synchronizing audio recordings in a system composed of mobile devices and cloud technologies. The implementation is compared with one that only relies on a cloud provider's global clock.

Chapter 1: Introduction

When recording sound, it is often desirable to record and combine sounds from multiple locations to produce a sound of higher quality than if they were produced on their own. The process of combining multiple sounds into one output is called beamforming and has been applied in various fields ranging from business conference calling systems to scientific radio astronomy and sonar mapping.

Synchronization plays an important role in beamforming. If you do not know when a sound was captured, it is difficult to know at what point the sounds should be combined. Standard solutions typically use hardware to stream audio data into a single computer to be processed. These solutions, while very accurate and reliable, lack the mobility and accessibility you would typically find in a mobile device.

The motivation for this project was to see whether you can take advantage of common devices like smart-phones for use in a beamforming application. I had imagined getting together with some musicians and having them all record their instruments together with their own devices. Instead of purchasing microphones and special equipment, it would be less expensive to use what we already have to set up an ad-hoc recording studio.

Since the mobile devices would be communicating through a wireless medium and most likely be running at different processing rates, the idea of achieving perfect synchronization is desirable but very difficult to obtain.

Recent work by Sur, Wei, Zhang [2] and Smeding, Bosma, Castañeda [3] showed that it is definitely possible to implement beamforming solutions in a mobile system. Sur, Wei, and Zhang took a clever approach for handling the synchronizing problem. Using local timestamps from each of the device's network adapter and creating Linear Regression models around these timestamps, they found an Algebraic solution for synchronizing the devices together. One problem they encountered, however, was that the network adapter's timestamp was not always available in some devices [2]. They also mentioned that you could use an external audio beacon to periodically act as a reference for alignment, but they felt that the beacon would interfere with the recording. This idea inspired me to use an initial loud noise at the start of the recording to act as an alignment reference for all the devices. When compared to another approach that only used a cloud provider's global clock, the Peak Alignment Implementation, as I call it, produced better results in terms of synchronization accuracy.

Past studies by Silipo [4] and Stone, Moore [5] have shown that relatively small across-channel delays (less than 20 milliseconds) can result in significant loss of speech intelligibility [6][7]. In my experiment, the Peak Alignment Implementation achieved an average delay of roughly 17 milliseconds while my other approach, called the Server Clock Implementation, achieved an average delay of roughly 42 milliseconds.

With regards to sensor networks, Deligeorges, Cakiades, Wang, and Doyle [8] described a strategy using a Fusion algorithm to synchronize sensors together for detecting/locating sounds. It achieved very accurate results (sub-millisecond) due to being written in way that directly access the hardware.

In regards to Audio Forensics and Multimedia Synchronization, Rodriguez, Apolinario, Biscainho [9] and Su, Hajj-Ahmad, Wu, Oard [10] described an idea in which you can exploit electric network frequency (ENF) signals to synchronize audio and video recordings together. An

ENF signal is a unique signal that gets in embedded in audio recordings captured from devices directly connected to a power grid. The ENF signal acts as a fingerprint for which multiple recordings can use for alignment/synchronization. However, since the devices need to be connected to a power grid, the strategy did not seem suitable for my mobile application.

Chapter 2: Approach and Implementation

Two approaches for synchronizing audio together were implemented and are currently available in the Loop Multitrack Recorder app on Google Play. The first approach, called the Server Clock Implementation, uses a cloud technology called Photon Unity Networking along with their clock API as a basis for synchronization. By knowing when each device started recording based on a common server clock, you can align all the recordings to the device that started last as depicted in Figures 2-i and 2-ii.

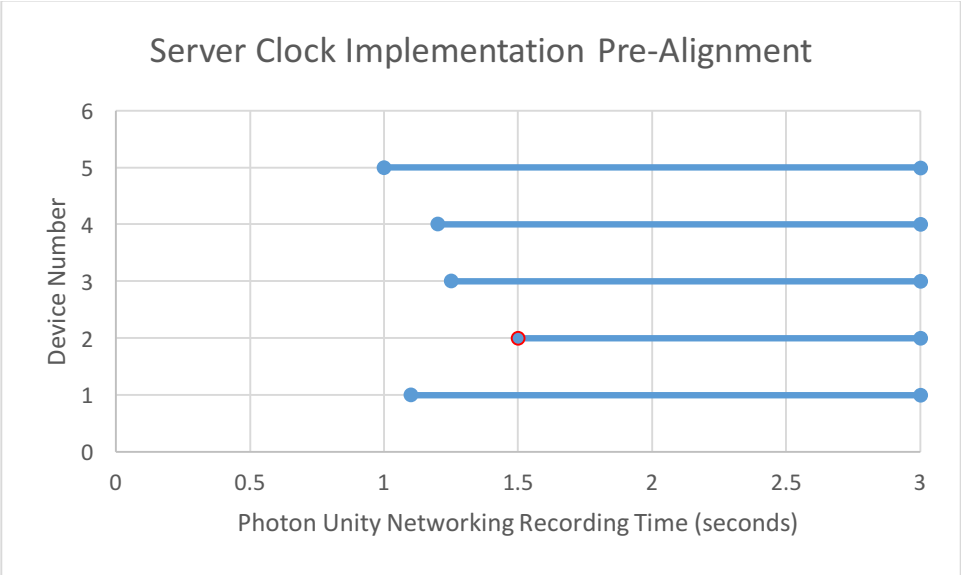

Figure 2-i: A hypothetical example for demonstration purposes. Each line represents a device's recording time. The red marker indicates that device 2 started recording the latest relative to the other devices.

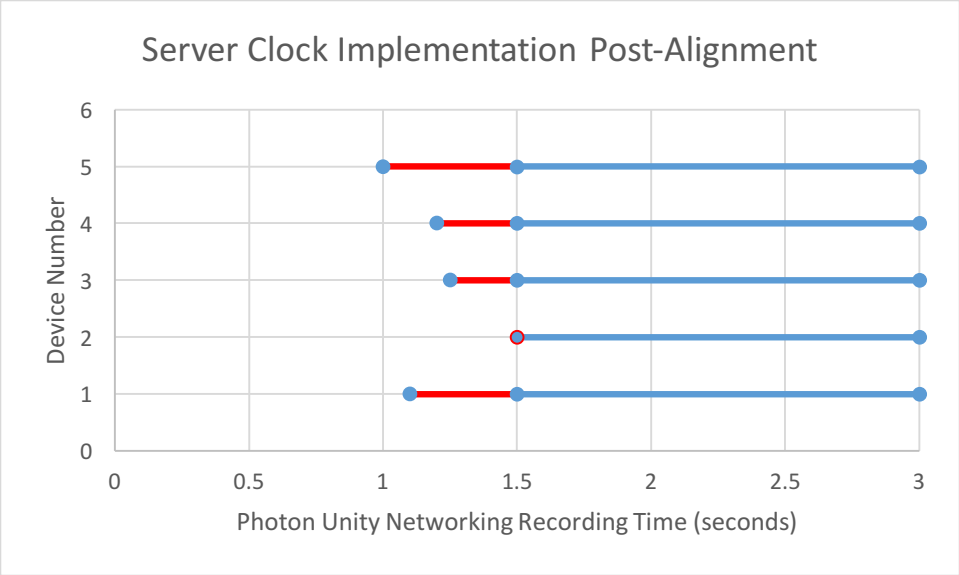

Figure 2-ii: Once you know the time at which the last device started recording, you can trim off the beginning of each of the other devices' recordings up until that point. In this way, the recordings are aligned and ready to be combined.

The problem with this approach, however, is that the common clock being used must already be in synch with all the devices. It turns out that this was not always the case. On initial runs of the approach and shown in Figure 2-iii's example, a very noticeable and unwanted reverberation effect was produced in the combined recordings. Upon further research, it was described in a blog by a Photon Unity Networking administrator that the global server clock contains an inaccuracy of roughly 15 milliseconds [11]. This inaccuracy, compounded with the inaccuracies associated with operating system process schedules and varying hardware [3], led me to search for a different approach.

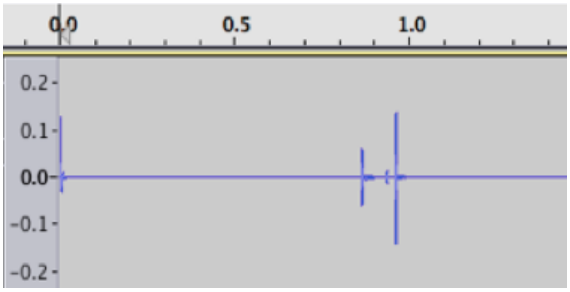

Figure 2-iii: One sound combined from three devices using the Server Clock Implementation. If the implementation were perfect, you would expect to only see one peak from the sound. However, you will notice that there are 3 peaks in this picture that produced an unwanted reverberation sound effect. The distance between these peaks is a measure of error in the synchronization process.

The second approach, called the Peak Alignment Implementation, requires the user to create a loud acute noise at the start of the recording to use as an alignment reference. You can use just about anything to create the noise. In my experiment, two drum sticks were clacked together to produce a noise at approximately the same distance from all the devices. When all the devices stop recording, the algorithm locates the first peak in decibel reading from the initial noise and trims off up to a second beyond that peak from all of the recordings as depicted in Figures 2-iv and 2-v.

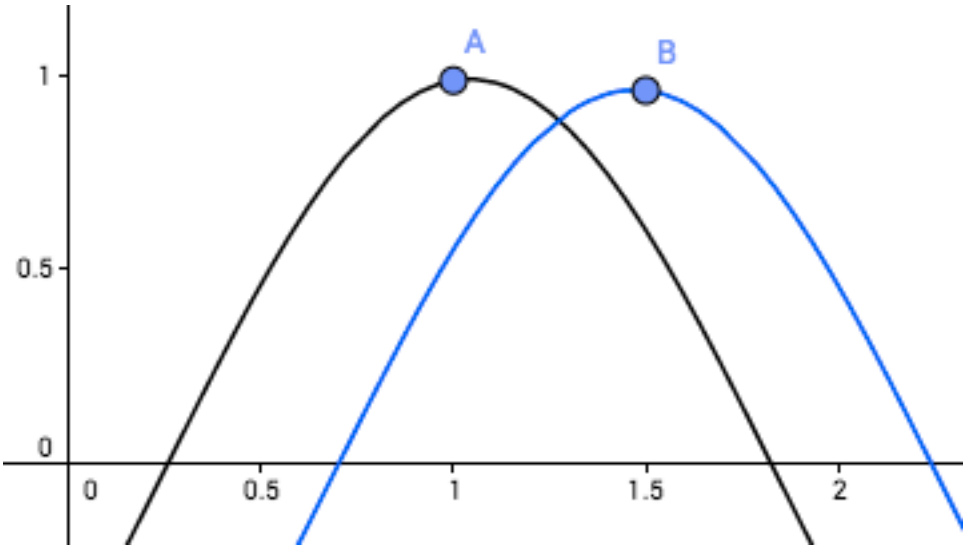

Figure 2-iv: A hypothetical example of one loud sound captured by two devices. Point A is the peak of the sound captured by device one and point B is the peak of the sound captured by device two. The y-axis represents the decibel reading of the sound and the x-axis represents when that decibel reading was captured in seconds.

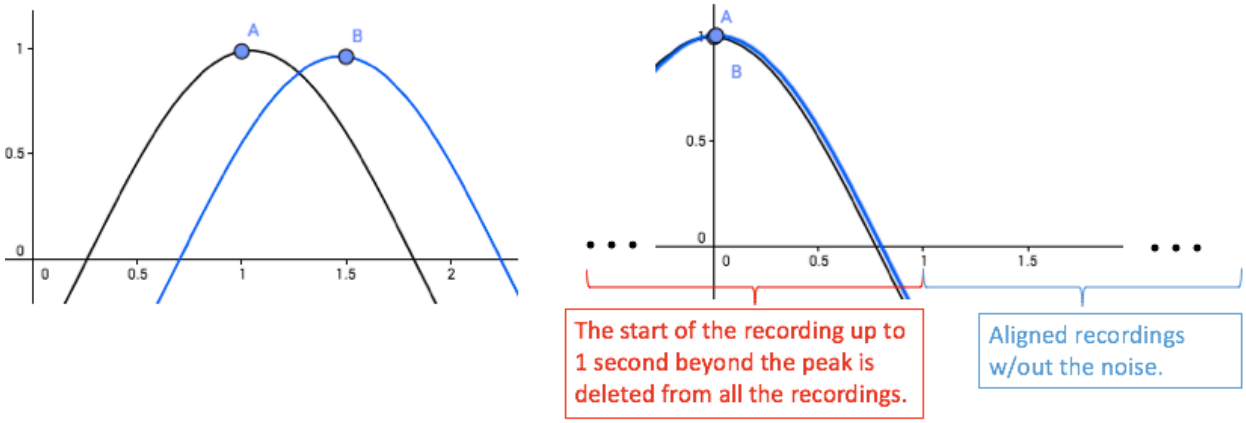

Figure 2-v: The Peak Alignment Implementation locates all the peaks in the devices for the initial sound and trims off 1 second beyond that point of reference. In this way, the recordings are aligned and ready to be combined without the initial noise.

Both approaches incorporated the use of Unity 3D in C# along with Photon Unity Network's API and Google Drive's API for uploading/downloading files to and from a shared Google Drive folder. Midworld's open source project, Google Drive for Unity 3D [12], provided a solid foundation for implementing the file transfer and account authorization aspects of the solutions. Photon Unity Network helped facilitate the communication between the devices using Remote Procedure Calls (RPCs) and provided the common clock. All of these technologies mentioned are displayed in Figure 3-vi's technology stack.

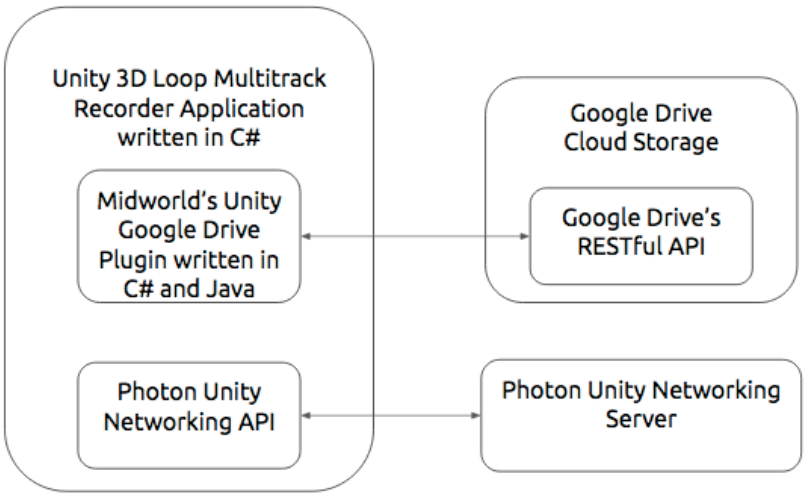

Figure 2-vi: Technology Stack

Assuming that a Google Drive folder has been shared prior to a session's creation, the device that created the session will upload a file containing a unique Photon Unity Network Room Identifier. With this id, the devices connect to the same room and communicate with each other using RPC's as depicted in Figure 2-vii. The RPCs are primarily invoked in the background, so all the user has to do is create or join a session, start the recording session, create an initial noise (if using the Peak Alignment Implementation), and press the stop recording button. Whenever someone stops recording, their audio file is uploaded to the shared folder. When all the devices have finished uploading their audio files, the device that created the session will download, combine, and upload the final combined audio to the shared folder.

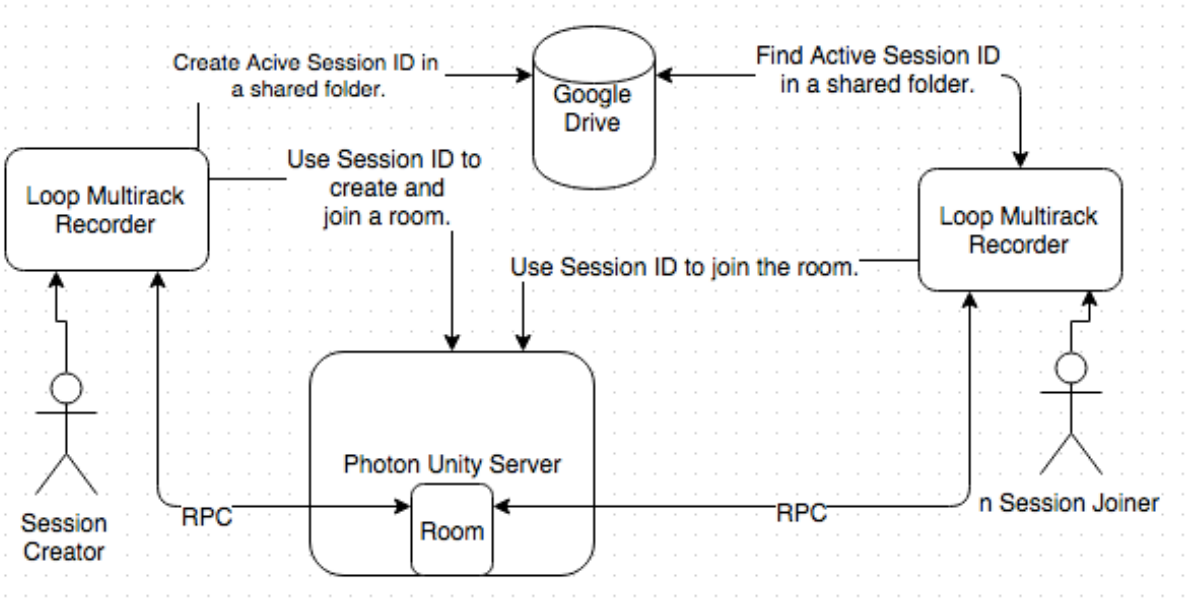

Figure 2-vii: Session Creation

Chapter 3: Experimental Setup

To measure the accuracy of the implementations, a testing ground was set up such that each device could be placed at equidistant locations and record metronome beats coming from the origin as shown in Figure 3-i. It is expected that the beat would reach each device at the same moment in time. So if you observe a set of peaks for one beat that align at the same time, then you can say the implementation is accurate. Furthermore, if the peaks are spread out for one beat, then you can say the implementation is inaccurate. The average of these differences provided a key metric for comparing the two implementations with each other.

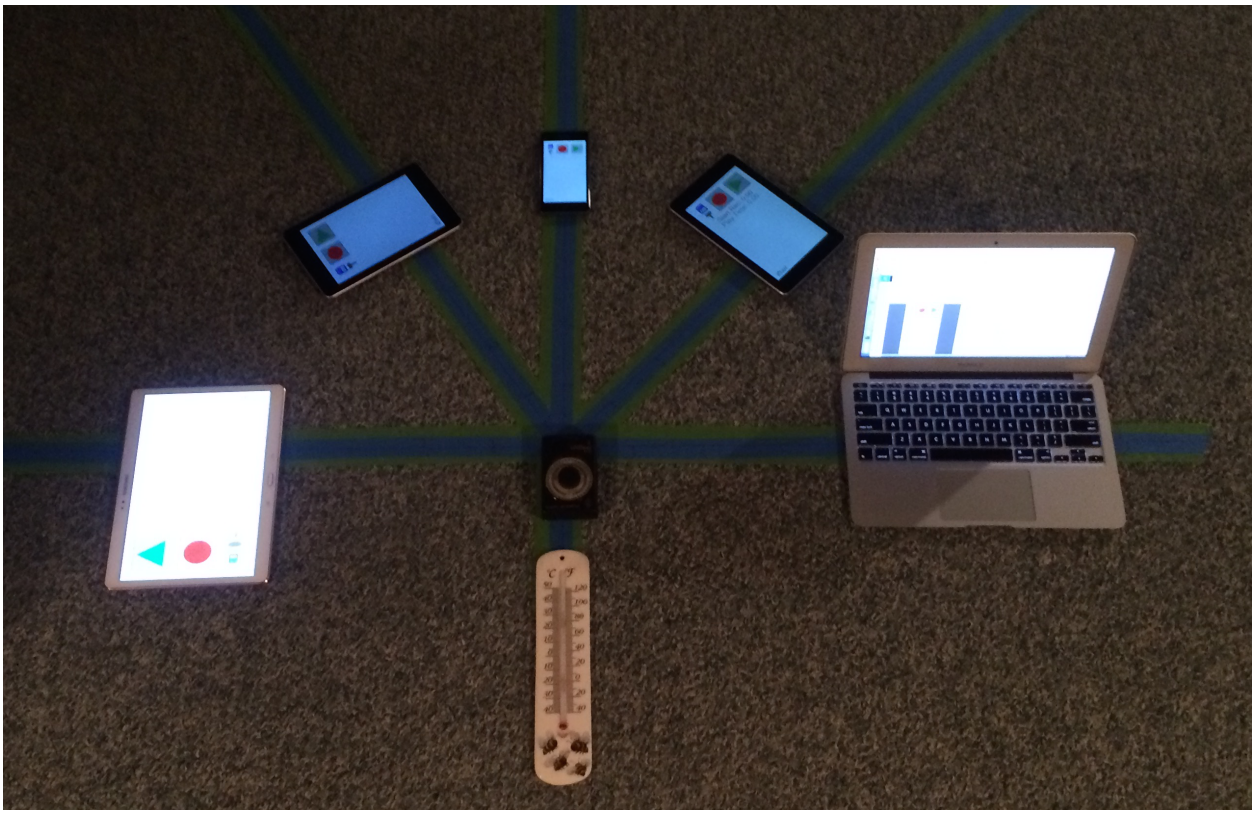

Figure 3-i: All the devices are spaced at 32 cm from the metronome origin. Also included is a thermometer at the bottom of the picture that measured the temperature of the experimental environment.

Prior to gathering data pertaining to the accuracy of the implementations, error due to the experimental process/environment was taken into account. Some of the devices contained a tiny pinhole that represented the device's microphone location. To get an idea of what we can expect if any of these locations were off, we can assume that the error in the true location was not greater than 10 cm and use the speed of sound formula [13] to derive an expected error. The following is the derivation of the potential error introduced:

(1) $$v \approx \left(331\frac{m}{s}\right)\sqrt{1+\frac{T_C}{273^\circ}}$$

$v\ is\ the\ velocity\ of\ sound\ and\ T_C\ is\ the\ temperature\ in\ Celcius.$

(2) $$d = v * t$$

$d\ is\ distance\ and\ t\ is\ time.$

$Assuming\ that\ the$

(3) $$d_{error} < 10\ cm$$

implies that

(4) $$v * t_{error} < 10\ cm$$

(5) $$t_{error} < \frac{10\ cm}{v}$$

(6) $$t_{error} < \frac{10\ cm}{\left(331\ \left(\frac{m}{s}\right) * 100\left(\frac{cm}{m}\right)\right)\sqrt{1+\frac{T_C}{273^\circ}}}$$

$During\ the\ experiment\ the\ temperature\ fluctuated\ between\ 19^\circ\ and\ 22^\circ$

(7) $$19^\circ \leq T_C \leq 22^\circ$$

(8) $$t_{error\ @\ 19^\circ} < \frac{10\ cm}{(331\ \left(\frac{m}{s}\right) * 100\left(\frac{cm}{m}\right))\sqrt{1+\frac{19^\circ}{273^\circ}}}$$

(9) $$t_{error\ @\ 19^\circ} < 0.00029212\ s$$

(10) $$t_{error\ @\ 22^\circ} < \frac{10\ cm}{(331\ \left(\frac{m}{s}\right) * 100\left(\frac{cm}{m}\right))\sqrt{1 + \frac{22^\circ}{273^\circ}}}$$

(11) $$t_{error\ @\ 22^\circ} < 0.000290631\ s$$

From the previous derivation, it is safe to say that approximating the devices' microphone locations potentially introduced up to 0.0003 seconds of error.

The audio was recorded in sets of 5 at each equidistant location: 8 cm, 16 cm, 32 cm, and 64 cm as shown in Table 3-i. The metronome was set at a rate of 60 beats per minute and each device recorded roughly a minute of recording. The data range used in analysis were beats 2 through 57, as the first beat produced was often corrupted by the sound of turning the metronome's power switch on.

Table 3-i: The devices' names, the distances they were spaced from the metronome origin during the experiment, and their operating systems.

The mobile devices used in the experiment.	The set of distances from the metronome origin used in the experiment.	The device's operating system.
MacBook Air	{8, 16, 32, 64}	OS X Yosemite 10.10.5
Nexus 7	{8, 16, 32, 64}	Android Lollipop 5.1.1
Sony Xperia Z1 Compact	{8, 16, 32, 64}	Android Lollipop 5.1.1
Nexus 7	{32, 64}	Android Lollipop 5.1.1
Samsung Galaxy Tab S	{32, 64}	Android Lollipop 5.0.2
Wittner MT-50 Quartz Metronome	{0}	None (Strictly Hardware)
Photon Unity Network Server	Unknown	Most likely Windows (Cloud Provider encapsulates the version of Windows)

The files generated from the experiment were WAV files. WAV files primarily consist of an array of decibel readings captured by the device's sound card. The readings, called samples, are of a byte format which are typically converted into a real number ranging from -1 to 1. The sample indices of this array are used in conjunction with the known sampling rate to map the sound over time in a digital format.

The points of interest in the WAV files were the peak times corresponding to the metronome beats. To locate and consolidate these peaks from the WAV files, an algorithm loosely based on Originlab's Window Search Peak Finding [14] concept of partitioning large data into windows of a fixed size was created to locate each peak in every window. To avoid echoes and external noises from being misinterpreted as metronome peaks, every time a peak was found, the algorithm would jump a fixed amount and parse the next window. When the algorithm finishes its

traversal, the peak times are then placed inside a CSV file for easier data manipulation and analysis using Microsoft Excel.

Chapter 4: Analysis

If you look at only one beat, there should be only one initial peak on each of the device's recordings. If all the devices have the same time corresponding to this beat, then there wouldn't be any difference in the times and all the devices would be considered in synch. If there is a difference between any two devices, then you can average all of these differences to produce a measurement corresponding to the average error in synchronization. Table 4-i shows a glimpse of how this average error was produced in Excel, using the average of the absolute differences between device peak times at a beat.

Table 4-i: Fields D1 through D5 are the peak times at a beat for devices 1 through 5 respectively. This screenshot is of an Excel spreadsheet corresponding to Run 1 for the 64 cm spacing using the Peak Alignment Implementation.

Beat_n	D1	D2	D3	D4	D5	abs(D1 – D2)	abs(D1-D3)	abs(D1-D4)	abs(D1-D5)	abs(D2-D3)	abs(D2-D4)	abs(D2-D5)	abs(D3-D4)	abs(D3-D5)	abs(D4-D5)	Average Error
2	5.858185941	5.873514739	5.865578231	5.857823129	5.863900227	0.015328798	0.00739229	0.000362812	0.005714286	0.007936508	0.01569161	0.009614512	0.007755102	0.001678005	0.006077098	0.007755102
3	6.892471655	6.907845805	6.899909297	6.892154195	6.898185941	0.01537415	0.007437642	0.00031746	0.005714286	0.007936508	0.01569161	0.009659864	0.007755102	0.001723356	0.006031746	0.007764172
4	7.926712018	7.942131519	7.934195011	7.926394558	7.932517007	0.015419501	0.007482993	0.00031746	0.005804989	0.007936508	0.015736961	0.009614512	0.007800454	0.001678005	0.006122449	0.007791383
5	8.961043084	8.976462585	8.968526077	8.960725624	8.966802721	0.015419501	0.007482993	0.00031746	0.005759637	0.007936508	0.015736961	0.009659864	0.007800454	0.001723356	0.006077098	0.007791383
6	9.995283447	10.0107483	10.00281179	9.995011338	10.00113379	0.015464853	0.007528345	0.000272109	0.00585034	0.007936508	0.015736961	0.009614512	0.007800454	0.001678005	0.006122449	0.007800454
7	11.02956916	11.04507937	11.03714286	11.0293424	11.03546485	0.015510204	0.007573696	0.000226757	0.005895692	0.007936508	0.015736961	0.009614512	0.007800454	0.001678005	0.006122449	0.007809524
8	12.06380952	12.07931973	12.07142857	12.06362812	12.06975057	0.015510204	0.007619048	0.000181406	0.005941043	0.007891156	0.01569161	0.009569161	0.007800454	0.001678005	0.006122449	0.007800454
9	13.09814059	13.11365079	13.10575964	13.09795918	13.10408163	0.015510204	0.007619048	0.000181406	0.005941043	0.007891156	0.01569161	0.009569161	0.007800454	0.001678005	0.006122449	0.007800454
10	14.13238095	14.14793651	14.14004535	14.1322449	14.13836735	0.015555556	0.007664399	0.000136054	0.005986395	0.007891156	0.01569161	0.009569161	0.007800454	0.001678005	0.006122449	0.007809524
11	15.16666667	15.18226757	15.17437642	15.16657596	15.17269841	0.015600907	0.007709751	9.07029E-05	0.006031746	0.007891156	0.01569161	0.009569161	0.007800454	0.001678005	0.006122449	0.007818594
12	16.20090703	16.21655329	16.20866213	16.20086168	16.20698413	0.015646259	0.007755102	4.53515E-05	0.006077098	0.007891156	0.01569161	0.009569161	0.007800454	0.001678005	0.006122449	0.007827664
13	17.2352381	17.25088435	17.2429932	17.23514739	17.24131519	0.015646259	0.007755102	9.07029E-05	0.006077098	0.007891156	0.015736961	0.009569161	0.007845805	0.001678005	0.0061678	0.007845805
14	18.26947846	18.28517007	18.27727891	18.26943311	18.27560091	0.01569161	0.007800454	4.53515E-05	0.006122449	0.007891156	0.015736961	0.009569161	0.007845805	0.001678005	0.0061678	0.007854875
15	19.30376417	19.31950113	19.31160998	19.30376417	19.30993197	0.015736961	0.007845805	0	0.0061678	0.007891156	0.015736961	0.009569161	0.007845805	0.001678005	0.0061678	0.007863946
16	20.33800454	20.35378685	20.34589569	20.33804989	20.34421769	0.015782313	0.007891156	4.53515E-05	0.006213152	0.007891156	0.015736961	0.009569161	0.007845805	0.001678005	0.0061678	0.007882086
17	21.3723356	21.38811791	21.38022676	21.37238095	21.37854875	0.015782313	0.007891156	4.53515E-05	0.006213152	0.007891156	0.015736961	0.009569161	0.007845805	0.001678005	0.0061678	0.007882086
18	22.40657596	22.42240363	22.41451247	22.40666667	22.41283447	0.015827664	0.007936508	9.07029E-05	0.006258503	0.007891156	0.015736961	0.009569161	0.007845805	0.001678005	0.0061678	0.007900227
19	23.44086168	23.45673469	23.44884354	23.44099773	23.44721088	0.015873016	0.007981859	0.000136054	0.006349206	0.007891156	0.015736961	0.00952381	0.007845805	0.001632653	0.006213152	0.007918367
20	24.47510204	24.49097506	24.48312925	24.47528345	24.48145125	0.015873016	0.008027211	0.000181406	0.006349206	0.007845805	0.01569161	0.00952381	0.007845805	0.001678005	0.0061678	0.007918367
21	25.50943311	25.52530612	25.51746032	25.50956916	25.51582766	0.015873016	0.008027211	0.000136054	0.006394558	0.007845805	0.015736961	0.009478458	0.007891156	0.001632653	0.006258503	0.007927438
22	26.54367347	26.55959184	26.55174603	26.54385488	26.55011338	0.015918367	0.008072562	0.000181406	0.006439909	0.007845805	0.015736961	0.009478458	0.007891156	0.001632653	0.006258503	0.007945578
23	27.57795918	27.5939229	27.5860771	27.57818594	27.58444444	0.015963719	0.008117914	0.000226757	0.006485261	0.007845805	0.015736961	0.009478458	0.007891156	0.001632653	0.006258503	0.007963719
24	28.61219955	28.62820862	28.62036281	28.61247166	28.61873016	0.01600907	0.008163265	0.000272109	0.006530612	0.007845805	0.015736961	0.009478458	0.007891156	0.001632653	0.006258503	0.007981859
25	29.64648526	29.66253968	29.65469388	29.64680272	29.65306122	0.016054422	0.008208617	0.00031746	0.006575964	0.007845805	0.015736961	0.009478458	0.007891156	0.001632653	0.006258503	0.008
26	30.71505669	30.73115646	30.72331066	30.7154195	30.721678	0.016099773	0.008253968	0.000362812	0.006621315	0.007845805	0.015736961	0.009478458	0.007891156	0.001632653	0.006258503	0.008018141
27	31.71505669	31.73115646	31.72331066	31.7154195	31.721678	0.016099773	0.008253968	0.000362812	0.006621315	0.007845805	0.015736961	0.009478458	0.007891156	0.001632653	0.006258503	0.008018141
28	32.74929705	32.76544218	32.75759637	32.74970522	32.75596372	0.016145125	0.00829932	0.000408163	0.006666667	0.007845805	0.015736961	0.009478458	0.007891156	0.001632653	0.006258503	0.008036281
29	33.78358277	33.79977324	33.79192744	33.78399093	33.79029478	0.016190476	0.008344671	0.000408163	0.006712018	0.007845805	0.015782313	0.009478458	0.007936508	0.001632653	0.006303855	0.008063492
30	34.81786848	34.83405896	34.82621315	34.81827664	34.8245805	0.016190476	0.008344671	0.000408163	0.006712018	0.007845805	0.015782313	0.009478458	0.007936508	0.001632653	0.006303855	0.008063492

Spreadsheets like Table 4-i were used to create graphical visualizations of the data for all the different configurations used in the experiment. For example, the graphs displayed in Figures 4-i and 4-ii show the average error in accuracy when the devices were spaced 64 centimeters from the metronome for both implementations. You will notice that at this particular configuration, the Peak Alignment Implementation performed better than the Server Clock Implementation in terms of accuracy.

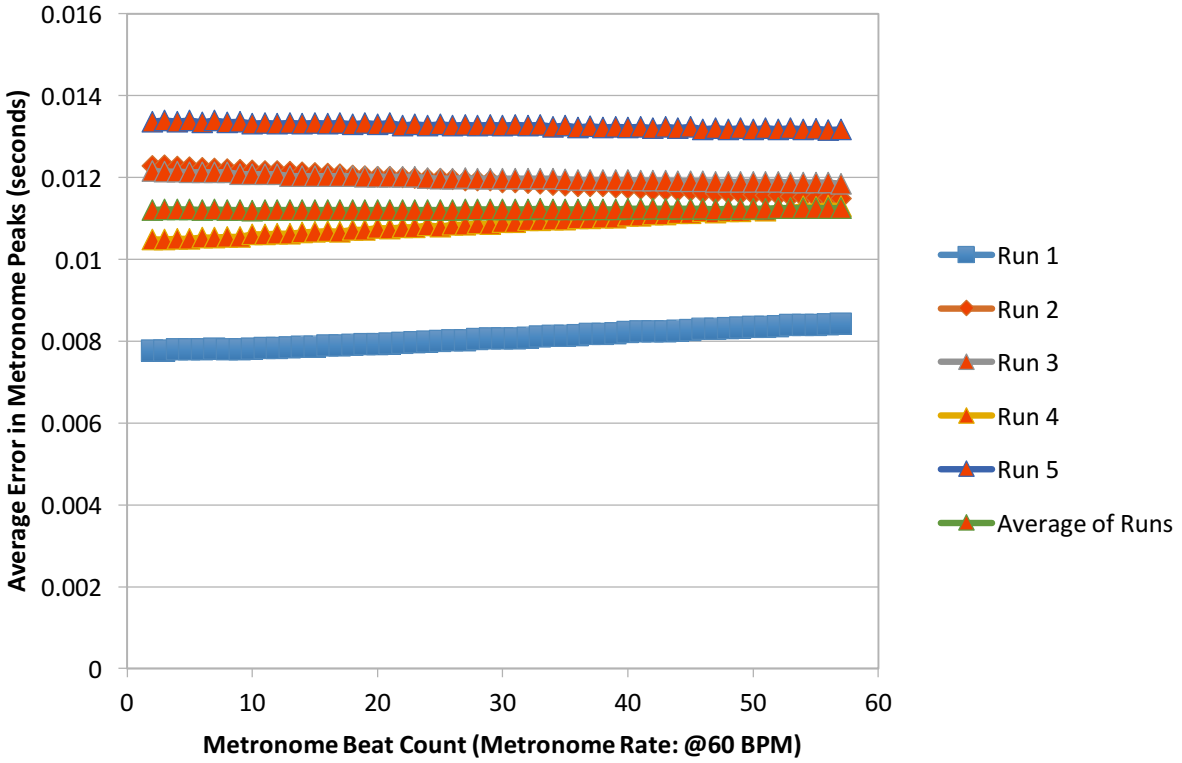

Figure 4-i: Average errors (seconds) in the Peak Alignment Implementation for the 64 cm spacing from the metronome origin.

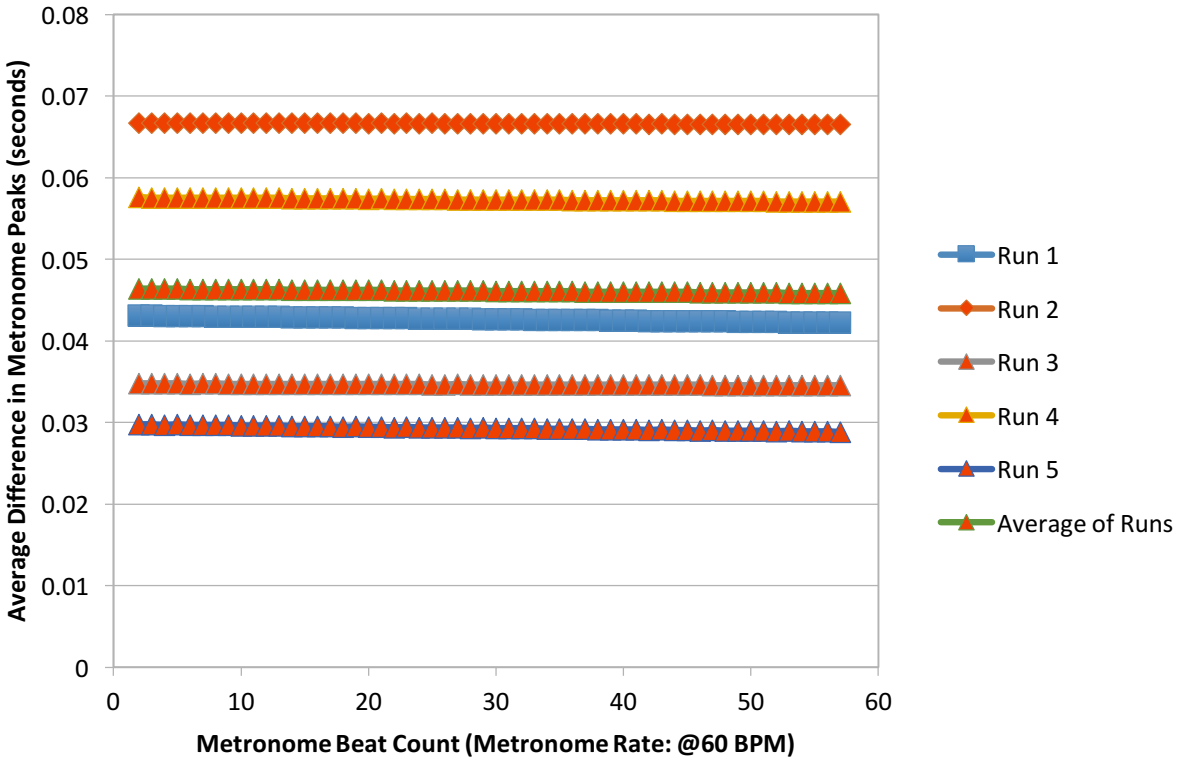

Figure 4-ii: Average errors (seconds) in the Server Clock Implementation for the 64 cm spacing.

Figures 4-iii and 4-iv show that even when all the devices were spaced at different distances from metronome origin, the Peak Alignment Implementation produced better results. This also aligns with our previous derivation of expected error due to approximating the location of the devices' microphones. In the derivation we found that if the devices were off by 10 centimeters, you can expect a relatively small value of 0.0003 seconds in error. The data from Figures 4-iii and 4-iv did not show any large patterns of error that would suggest otherwise.

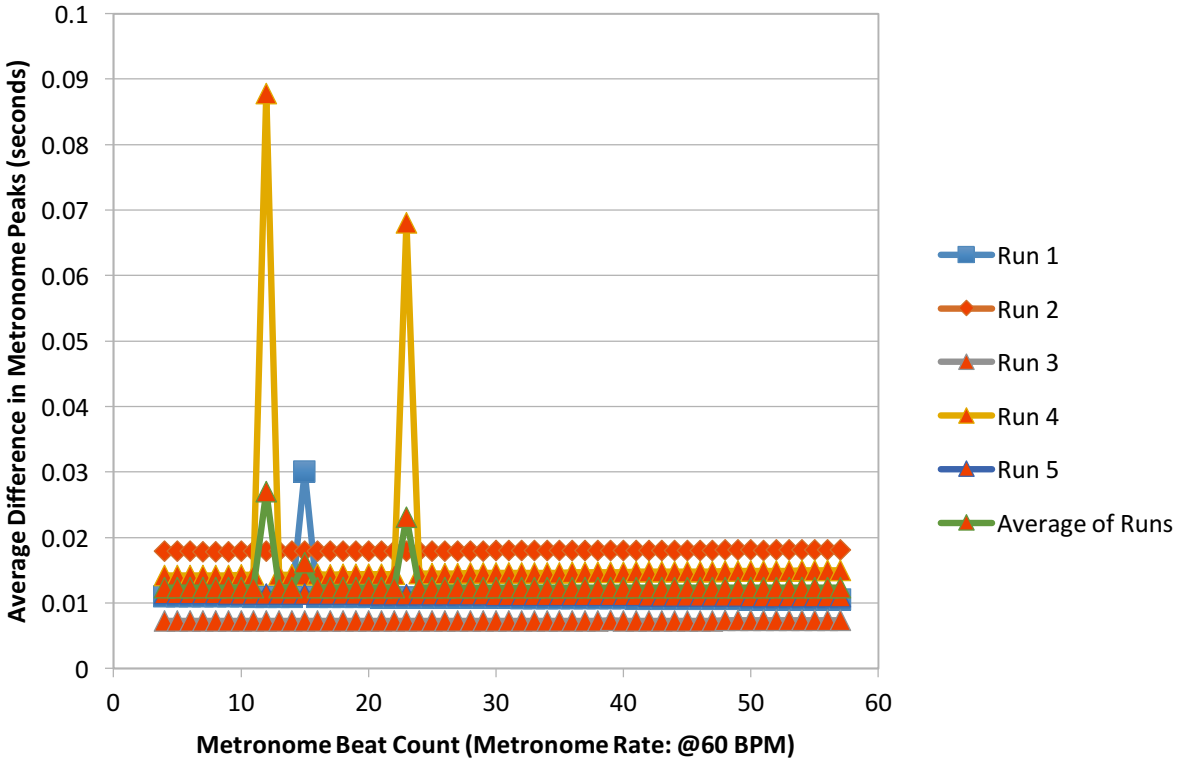

Figure 4-iii: Average errors (seconds) in the Peak Alignment Implementation for a scenario where each device was spaced at different locations from the metronome origin (4, 8, 16, 32, 64 cm).

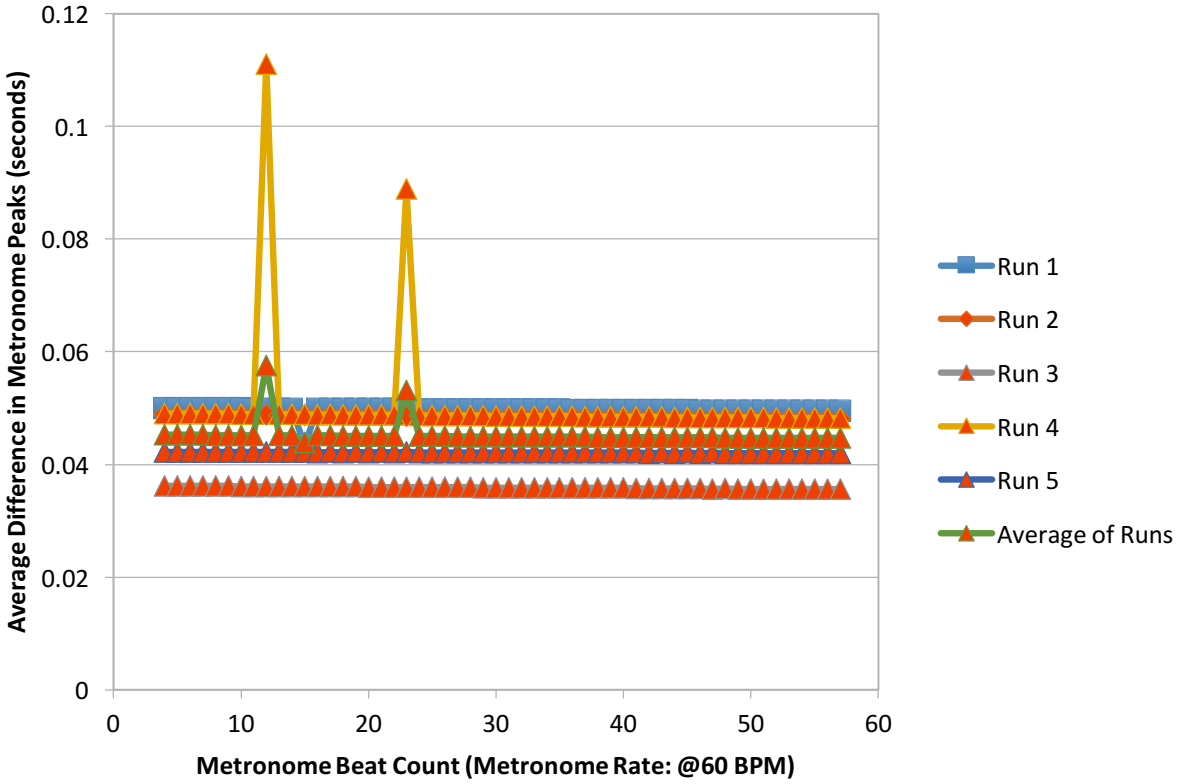

Figure 4-iv: Average errors (seconds) in the Server Clock Implementation for a scenario where each device was spaced at different locations from the metronome origin (4, 8, 16, 32, 64 cm).

Figures 4-v and 4-vi show all the averages in error for all the locations and runs or the entire experiment in two graphs. At first glance you will notice that the Peak Alignment Implementation, on average, had less error compared with the Server Clock Implementation. You might also notice the large spikes in the graphs. The large spikes, or errors, can likely be attributed to external noises leaking into the experimental environment and or issues relating to the window search algorithm. Since it is difficult to pinpoint the exact cause of these spikes, we can look at Figures 4-vii & 4-viii and locate an outlier-free subset that follows a linear progression you would expect from a normal metronome. With this subset, we can formally conduct a statistical analysis on whether the results from the Peak Alignment Implementation is significantly different than the Server Clock Implementation.

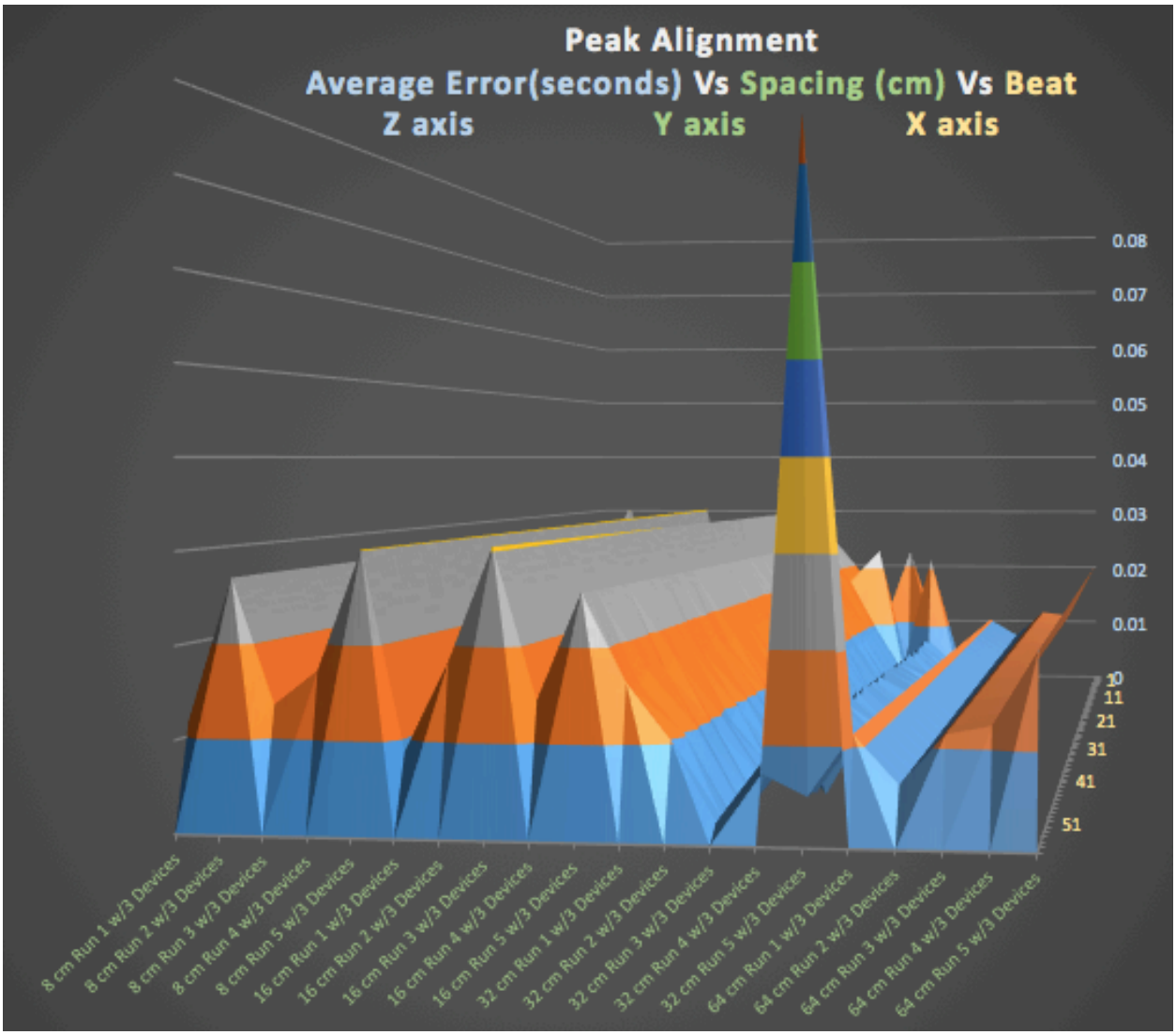

Figure 4-v: All the runs corresponding to the Peak Alignment Implementation.

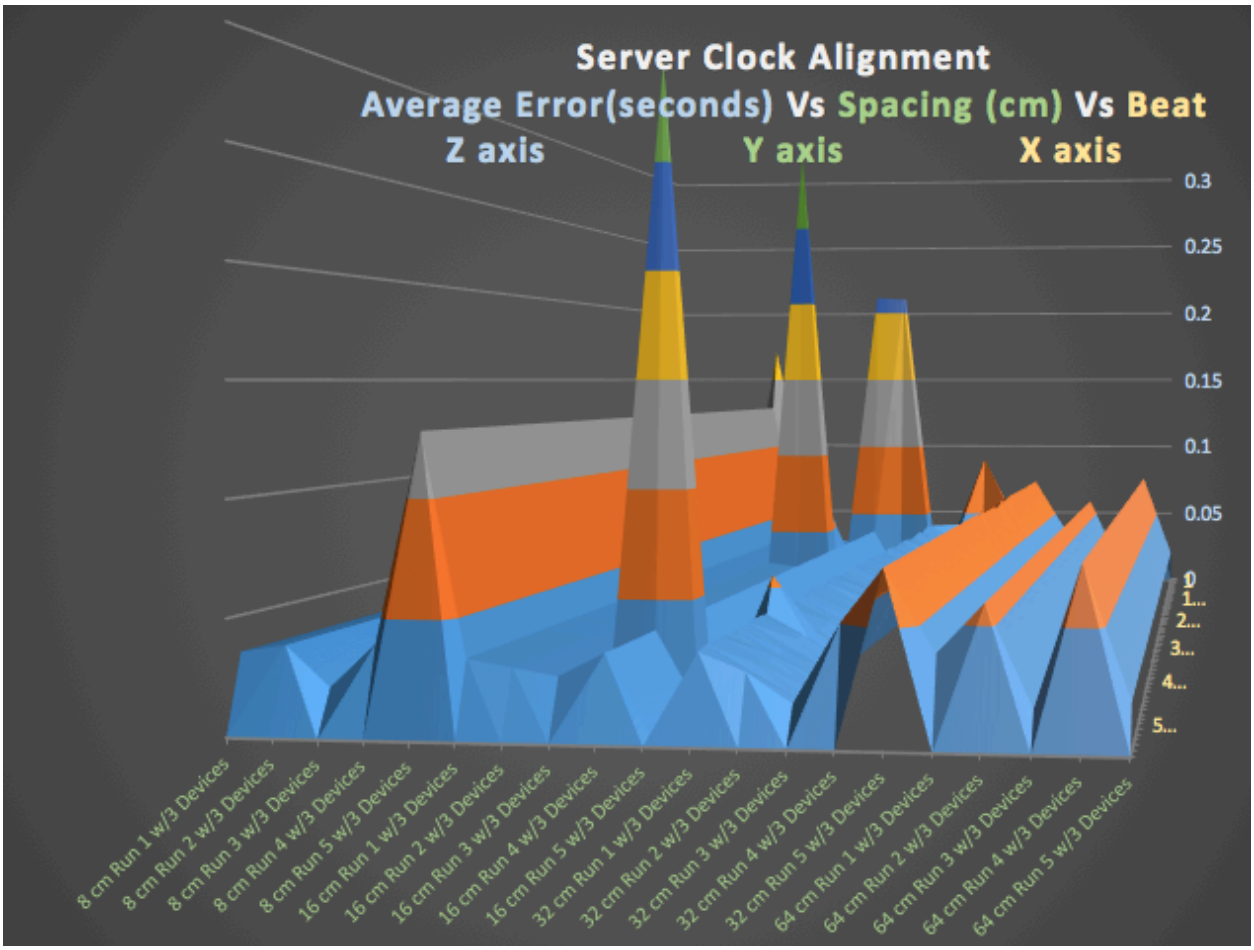

Figure 4-vi: All the runs corresponding to the Server Clock Implementation.

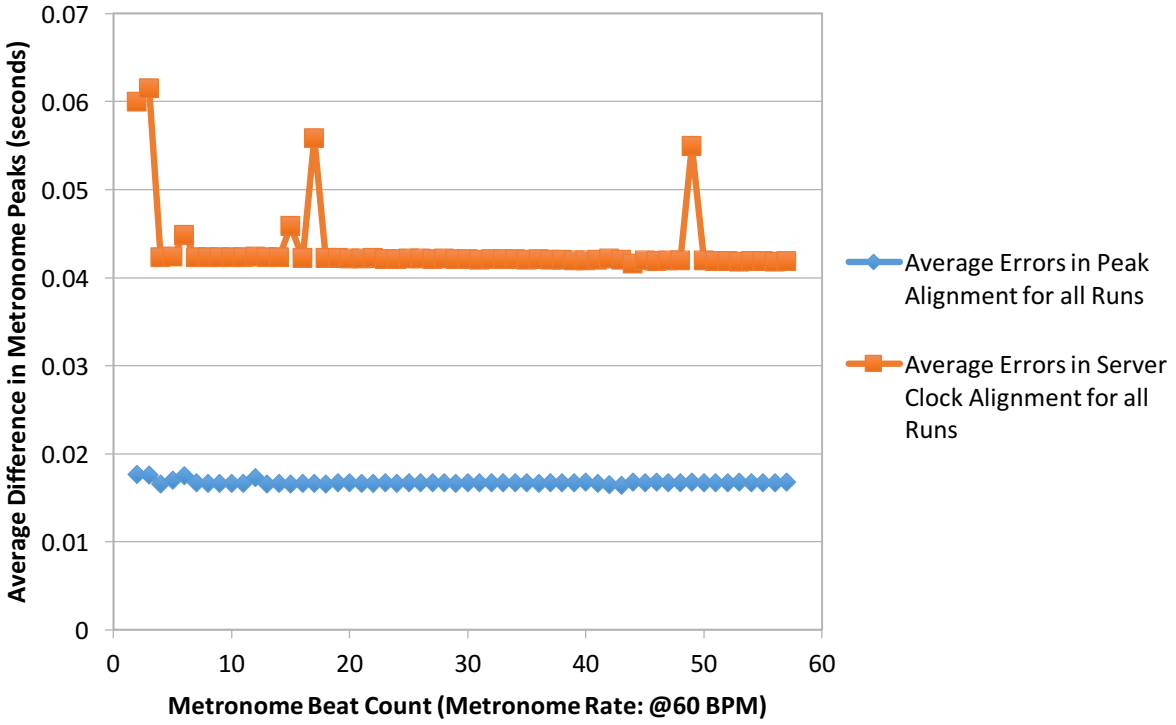

Figure 4-vii: Average errors (seconds) in Peak Alignment vs Server Clock Implementations.

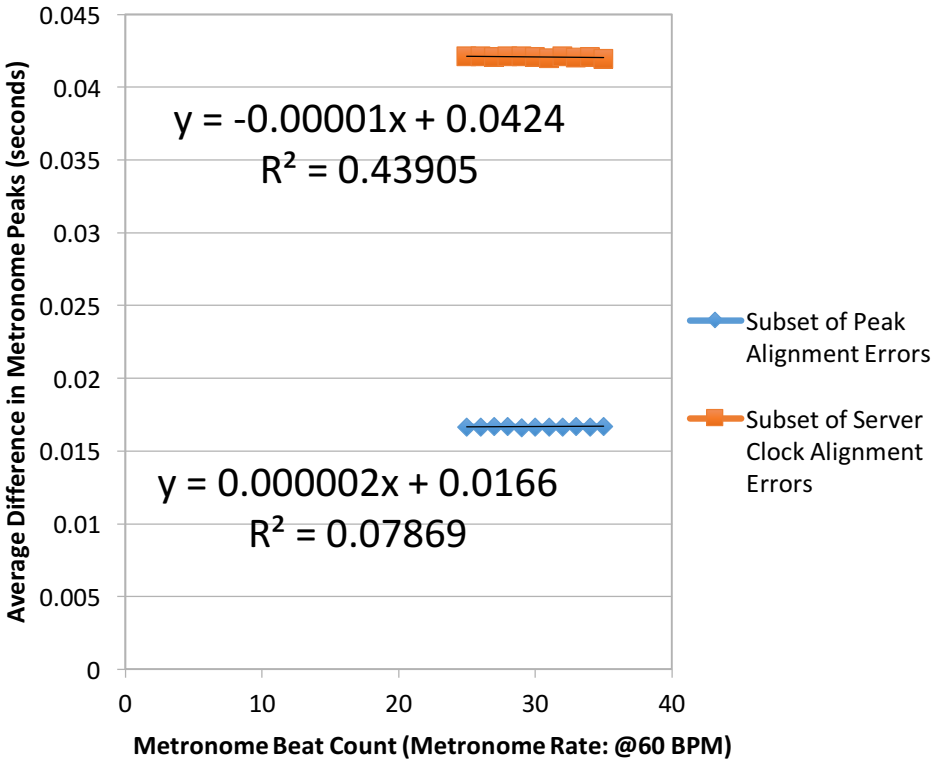

Figure 4-viii: Outlier-Free Subset

At this point you can see that in ideal conditions during the experiment, the Server Clock Implementation experienced an average error of approximately 42 milliseconds and the Peak Alignment Implementation experienced an average error of approximately 17 milliseconds.
If you considered the observed range of values that produced theses averages, it might be the case that there may be no difference between the averages since the data sets overlap too heavily. To account for this scenario, a Paired One-Sided T-Test was performed within the sub-set data (beats 25 through 35) as follows:

(1)
$$H_0: \mu_p = \mu_s$$
$$H_0: \mu_p - \mu_s = 0$$

The null hypothesis, H_0, is making the assumption that there is no difference between the population mean error for the Peak Alignment Implementation, μ_p, and the Server Clock Implementation, μ_s.

(2)
$$H_a: \mu_p < \mu_s$$
$$H_a: \mu_p - \mu_s < 0$$

The alternate hypothesis, H_a, is that the population mean error for the Peak Alignment Implementation is less than the Server Clock Implementation.

(3) *The sample data from beat 25 was chosen at random from within the subset data range of beat 25 through beat 35. The difference in error has been assumed to come from a Normal Distribution as shown in Figure 4-ix.*

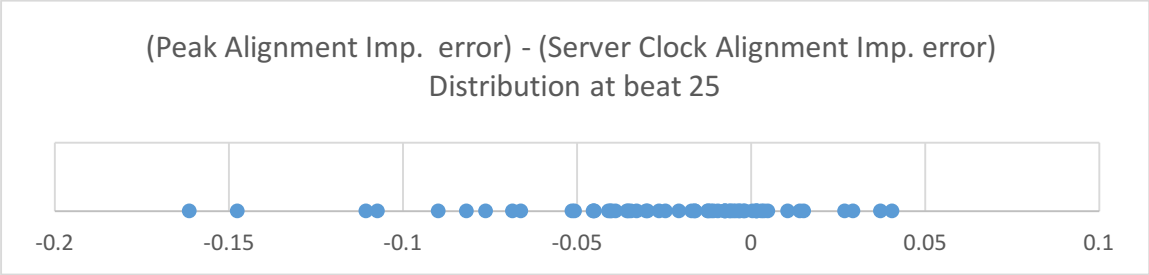

Figure 4-ix: The difference between the Peak Alignment Implementation error and the Server Clock Implementation error closely resembles a Normal Distribution.

(4) *Using Excel's Paired One Sided TTest function gives a*

$$P\ Value = 2.68234E\text{-}06$$

The P Value is the probability that the averages are the same when considering the data sets' observed variation. Since the probability is very small, we can reject that the averages are the same and we can accept our alternative hypothesis that the mean error for the Peak Alignment Implementation is smaller than the mean error for the Server Clock Implementation [15].

To see whether or not the factor of distance influenced the mean error, a One-Way Analysis of Variance (ANOVA) test was performed on the sub-set data for the two implementations at beat number 33. ANOVA is an extension of the T-Test. Instead of considering two population means, we are now considering multiple population means. In the following, ANOVA is used to see whether or not spacing the devices at 8cm or 16cm or 32cm or 64cm from the metronome origin made any difference with respect to the average error observed in the Peak Alignment Implementation. The first three points are the conditions that need to be satisfied prior to conducting ANOVA.

(1) Beat 33 was chosen at random from the subset beat range of 25 through 35.

(2) The F-Test for homogeneity in variance was accepted to be equal for all the distances [16].

(3) The distributions have been assumed to come from Normal Distributions as shown in Figure 4-x.

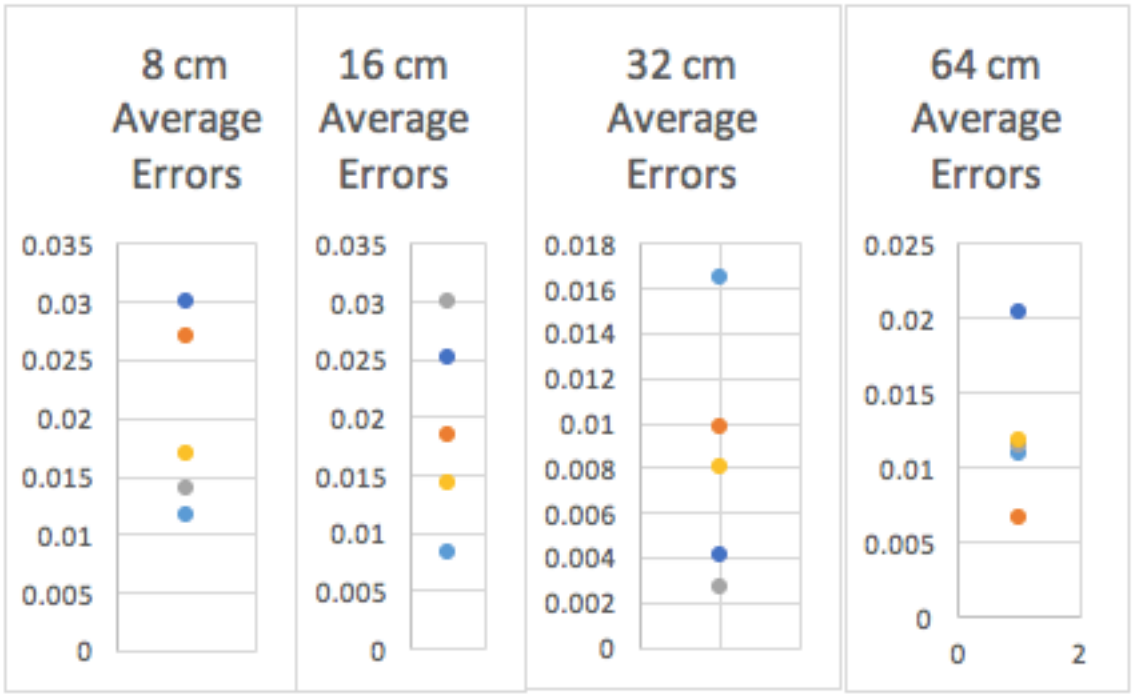

Figure 4-x: Average errors (seconds) at each spacing. Light Blue indicates Run 1, Orange indicates Run 2, Grey indicates Run 3, Yellow indicates Run 4, and Dark Blue indicates Run 5.

(4) $$H_0: \mu_8 = \mu_{16} = \mu_{32} = \mu_{64}$$

The null hypothesis, H_0, is making the assumption that there is no difference between the population mean error at each spacing.

(5) $$H_a\colon \mu_8 \neq \mu_{16} \neq \mu_{32} \neq \mu_{64}$$

The alternate hypothesis, H_a, is that the population mean error at each spacing is different.

(6) *Table 4-ii: ANOVA Summary when considering the devices' distance from the metronome origin as a factor.*

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
8 cm Averages	5	0.099803477	0.019960695	6.72726E-05
16 cm Averages	5	0.096356765	0.019271353	7.44571E-05
32 cm Averages	5	0.041239607	0.008247921	2.9596E-05
64 cm Averages	5	0.061284958	0.012256992	2.47161E-05

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F cri*
Between Groups	0.000479752	3	0.000159917	3.262920842	0.048977697	3.23887
Within Groups	0.000784167	16	4.90104E-05			
Total	0.001263919	19				

The small P-value in Table 4-ii indicates that we can't accept that the average errors at each spacing for the Peak Alignment Implementation are the same. This implies that we can accept the alternative hypothesis that there is a difference in the average errors when considering the device distance from the metronome origin. Upon further investigation, a T-Test was performed for the 8/16 cm spacing group and the 32/64 cm spacing group. Contrary to my expectation, the test concluded that it is plausible that the mean error for the 32/64 cm group is less than the 8/16 cm group for the Peak Alignment Implementation.

For the Server Clock Implementation, a similar ANOVA test was conducted and concluded that there was no significant difference between the average errors at each spacing. Since its usage doesn't require you to be relatively in the same proximity, this result was more aligned with my expectation.

To determine whether or not the amount of devices used in the experiment impacted the results, we can look at the runs corresponding to the 32/64 cm group and conduct a One-Way ANOVA test comparing the average errors experienced between any 2 of the 5 devices, 3 of the 5 devices, 4 of the 5 devices, and 5 of the 5 devices. The 32/64 cm group was chosen for the ANOVA calculation because all 5 devices were used in the 10 runs corresponding to the 32 and 64 cm group. Beat number 27 was chosen at random from within the subset range (beat 25 through beat 35) to satisfy the first condition of the ANOVA test as follows:

(1) The data set pertaining to beat 27 was chosen at random. The devices that produced each average were chosen at random.

(2) The F-Test for homogeneity in variance was accepted to be equal for all the distances [16].

(3) The distributions have been assumed to come from Normal Distributions as shown in Figure 4-xi.

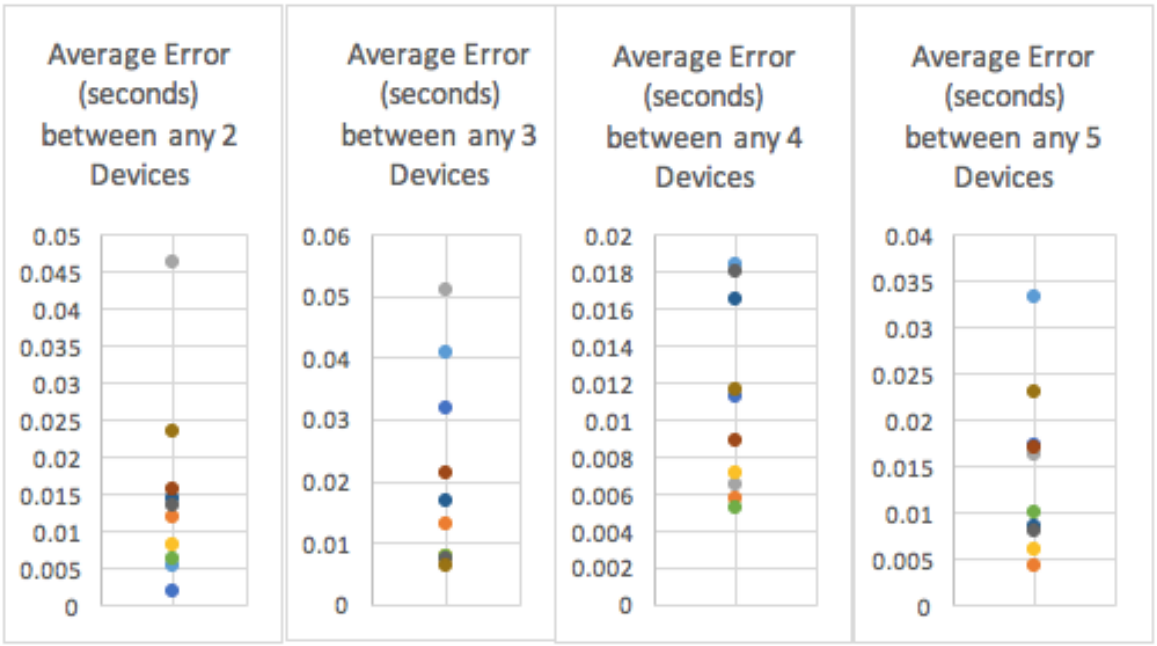

Figure 4-xi: Average error (seconds) for randomly selected devices at beat 27. Each point represents an averaged run in the 32/64 cm group. Each color corresponds to one of the ten runs.

(4) $$H_0\colon\ \mu_2 = \mu_3 = \mu_4 = \mu_5$$

The null hypothesis, H_0, is making the assumption that there is no difference between the population mean average error for any additional devices considered.

(5) $$H_a\colon\ \mu_2 \neq \mu_3 \neq \mu_4 \neq \mu_5$$

The alternate hypothesis, H_a, is that the population mean average error is different when more or less devices are considered.

(6) *Table 4-iii: ANOVA Summary when considering the number of devices in use as a factor.*

SUMMARY

Groups	*Count*	*Sum*	*Average*	*Variance*
2	10	0.147165533	0.014716553	0.000160095
3	10	0.203854875	0.020385488	0.000247913
4	10	0.109463341	0.010946334	2.60423E-05
5	10	0.144365079	0.014436508	8.00348E-05

ANOVA

Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F cri*
Between Groups	0.000457748	3	0.000152583	1.18721681	0.328290265	2.86626
Within Groups	0.004626764	36	0.000128521			
Total	0.005084512	39				

The large P-value in Table 4-iii indicates that it is plausible that the average error when considering 2 devices could be the same as the average error when considering up to 5 devices. This means that considering more or less devices did not significantly impact the Peak Alignment Implementation's synchronization error. Similarly, the Server Clock Implementation's ANOVA test concluded that there was also no significant difference when considering additional devices in use. Theoretically both implementations can handle up to 8 devices, however, this scenario was not tested in this experiment.

Chapter 5: Conclusion

Using a loud noise at the beginning of a recording session improved the process of aligning multiple recordings together. The Peak Alignment Implementation that used this technique, on average, produced an error of approximately 17 milliseconds during the experiment. This is 26 milliseconds better than the Server Clock Implementation which primarily used a cloud provider's global clock for synchronization.

Any error in synchronization produced an explicit reverberation sound effect when combining multiple recordings that focused on a single source of audio. The Peak Alignment Implementation significantly reduced this effect, but required an initial noise to be created at the start of the recording and within the devices' proximity. The Server Clock Implementation, on the other hand, did not require the devices to be in the same proximity or an initial noise to be generated by the user. In this way the Server Clock Implementation could be used for a wider range of applications, if you can tolerate average errors of around 42 milliseconds.

Both solutions did not compromise accuracy when the number of devices varied. There was little difference in the amount of error observed when considering the use of 2, 3, 4, or 5 devices in a session.

References

[1] Nick Schaafsma. “Loop Multitrack Recorder”. https://sites.google.com/site/loopmultitrackmusicrecorder/ Retrieved April 2016.

[2] Sanjib Sur, Teng Wei, and Xinyu Zhang. “Autodirective audio capturing through a synchronized smartphone array,” in Proceedings of the 12th annual international conference on Mobile systems, applications, and services (MobiSys '14). ACM, New York, NY, USA, 2014. pp. 28 - 41.

[3] Roy Smeding, Sjoerd Bosma, Supervisor: Dr. Jorge Martínez Castañeda. “Smartphone Audio Acquisition and Synchronization Using an Acoustic Beacon With Application to Beamforming,” Delf University of Technology, Netherlands. 2015.

[4] Silipo, R., Greenberg, S., and Arai, T. “Temporal constraints on speech intelligibility as deduced from exceedingly sparse spectral representations,” in Proceedings of Eurospeech, Budapest, 1999. pp. 2687 - 2690.

[5] Stone, M.A., and Moore, B.C.J. “Tolerable hearing aid delays. III. Effects of speech production and perception of across-frequency variation in delay,” Ear Hear, 2003. pp. 24, 175 - 183.

[6] J. V. Lichtenauer, J. Shen, and M. Pantic, “Cost-effective solution to synchronized audio-visual capture using multiple sensors,” in IEEE Conference on Advanced Video and Signal Based Surveillance (AVSS 2009), 2009. pp. 324 - 329.

[7] K. W. Grant, V. van Wassenhove, and D. Poeppel. “Discrimination of auditory-visual synchrony,” in International Conference on Audio-Visual Speech Processing, 2003. pp. 31 - 35

[8] S. Deligeorges, G.Cakiades, J. George, Y. Wang, and F. Doyle. “A Mobile Self Synchronizing Smart Sensor Array for Detection and Localization of Impulsive Threat Sources,” 2015 IEEE International Conference on Multisensor Fusion and lntegration for Intelligent Systems (MFI) Sept 14 - 16, 2015. San Diego, CA, USA

[9] D. Rodriguez, J. Apolinario, and L. Biscainho, “Audio authenticity: Detecting ENF discontinuity with high precision phase analysis,” IEEE Trans. Inf. Forensics Security, Sep. 2010. vol. 5, no. 3, pp. 534 - 543.

[10] H. Su, A. Hajj-Ahmad, M. Wu, and D. Oard, “Exploring the use of ENF for multimedia synchronization,” in IEEE Int’l Conf. Acoustics, Speech, and Signal Processing (ICASSP), 2014.

[11] Tobias (Admin). “Significant Problems with PhotonNetwork.time and timestamps,” http://forum.photonengine.com/discussion/1397/significant-problems-with-photonnetwork-time-and-timestamps February 2012. Retrieved January 2016.

[12] Midworld. “Google Drive for Unity 3D,” https://github.com/midworld/unity-googledrive Retrieved April 2016.

[13] Raymond A. Serway and John W. Jewett, Junior. Physics for Scientists and Engineers with Modern Physics, Seventh Edition. Thomson Higher Education, Belmont, CA, USA, 2008. pp. 475.

[14] Window Search Peak Finding Algorithm. http://www.originlab.com/doc/Origin-Help/PA-Algorithm Retrieved December 2015.

[15] Ann E. Watkins, Richard L. Scheaffer, and George W. Cobb. Statistics in Action, Understanding a World of Data 2nd Edition. Key Curriculum Press. 2008. pp. 625-626.

[16] Charles Caiontz. “Homogenity of Variances”. http://www.real-statistics.com/one-way-analysis-of-variance-anova/homogeneity-variances/ Retrieved February 2016.