University of South Carolina
Scholar Commons

Senior Theses | Honors College

Spring 5-10-2014

Live Musical Steganography

Latia Hutchinson
University of South Carolina - Columbia

Follow this and additional works at: https://scholarcommons.sc.edu/senior_theses

Part of the Applied Mathematics Commons, and the Music Commons

Recommended Citation
Hutchinson, Latia, "Live Musical Steganography" (2014). *Senior Theses*. 20.
https://scholarcommons.sc.edu/senior_theses/20

This Thesis is brought to you by the Honors College at Scholar Commons. It has been accepted for inclusion in Senior Theses by an authorized administrator of Scholar Commons. For more information, please contact dillarda@mailbox.sc.edu.

LIVE MUSICAL STEGANOGRAPHY

By

Tia Hutchinson

Submitted in Partial Fulfillment
of the Requirements for
Graduation with Honors from the
South Carolina Honors College

May, 2014

Approved:

Stephen Fenner
Director of Thesis

Mandy Fang
Second Reader

Steve Lynn, Dean
For South Carolina Honors College

Thesis Table of Contents:

- Thesis Summary…….pp. 2-3
- Introduction: Steganography……..pp.4-6
- Algorithm……pp. 7-10
- Before and After of a chosen composition……..pp.11-13
- Further Development……..pp.14-15
- Works Cited……pg.16

Thesis Summary:

Live Musical Steganography is a project created as a way to combine the two typically unrelated fields of music and information security into a cohesive entity that will hopefully spark one's imagination and inspire further development that could one day be beneficial in the world of security. For those who are unfamiliar with the term steganography, it can be defined as the art and science of preserving the integrity and confidentiality of a message by hiding the existence of that message within some larger body of data. In the field of steganography, much research and development has gone into methods of obscuring the presence of information digitally in photos or audio files. However, this project is unique in that the information is hidden in a composition that can be performed without unsuspecting listeners being aware of the modification of the piece to include a secret message.

The process of transmission is as follows. There is a sender (call her Alice) and a recipient (call him Bob). Alice and Bob are in a place where they can be near each other without having to seemingly be together (like a park). Alice would play her modified composition, which contains the secret message within the work. While Bob is aware that a message is being transmitted and records the performance accordingly, everyone else would probably just assume that Alice is playing her piece because music is her hobby. Bob would then compare Alice's performance to the notes of the original composition and write down the notes that differ. From there, he would be able to obtain the secret message. The most interesting part of this project is the fact that the piece that Alice plays sounds just as musical as the original composition.

Now that the basic gist of how the message will be transmitted has been addressed, we can briefly discuss the algorithm for inserting the message into a composition. The first step is to choose a composition large enough to conceal a message. Next, the message should be encrypted into what is known as a ciphertext so that if security was somehow compromised and the message was retrieved, it would be a jumble of letters instead of a coherent message. After encrypting the message, each letter of the message should be turned into a corresponding note and inserted consistently (such as every third note of each measure) into the composition. The resulting composition is now ready for Alice to play for Bob.

While there is an added layer of security provided with the message being encrypted before being inserted into the composition, most of the security of the message lies in the unassuming nature of the set-up. If the transmission between Alice and Bob were to occur in a park, then the casual observer may see what appears to be a normal day with a couple jogging, a mother pushing a stroller, Alice playing around on her flute, Bob on his laptop, and some teens playing Frisbee. In the midst of this normality however, Alice could be transmitting information that is of national importance to Bob.

Introduction:

Brief History:

Steganography, which is Greek for "covered writing" can be defined as "the art and science of communicating in such a way that hides the communication[1]". In other words, steganography involves transmitting a message through a medium that obscures the presence of that message. The earliest known written recording of steganography dates back as far as 400 B.C.[1]. In one of the stories, a king shaved the head of one of his prisoners, wrote a message on the bald head, waited for the hair to grow back, and then sent the prisoner to the intended recipient who was able to shave the head and retrieve the message without it being detected by an outside party[1]. During WWII, soldiers used "null ciphers" where for example, the third letter of each word in a note could be assembled into a secret message[1]. In today's time, steganographic applications have become much more advanced than null ciphers or using secret ink to transmit a secret message; the field of steganography has now developed into providing a digital sense of security.

Approach:

Before delving into a discussion on how steganography has developed, let's first cover some useful definitions. The cover medium is the body of work used to hide the secret message, the hidden data is the secret message one wants to transmit, while the stego-text is the finished product of the hidden data inserted into the cover medium[2]. In simplest terms, Cover Medium + Hidden Data = Stego-Text. Two important procedures to keep in mind when beefing up the security of a stego-text is that the cover medium should have enough noise

that the presence of the hidden data is undetectable—in other words having every other letter be a letter in the secret message would greatly increase the chances of the message's presence being detected. The second procedure is that the hidden data should be encrypted instead of just inserted into the cover media as the naked message so that if the stego-text was compromised by an outside party, the resulting message would be a jumble of letters instead of the actual message.

The three major mediums used for modern steganographic transmission are text files, image files, and audio files. In text files, the approaches used to transmit messages are line shifting (moving lines of text vertically from a stationary point and using the value to encrypt the secret message[1]), word shifting (same as line shifting, but moving words horizontally from a stationary point[1]), and feature specific encoding (encoding the secret message into formatted text[1]). In image files, which tend to be the most thought-of application of digital steganography, the main way of transmitting messages is through changing the least significant bits of each byte that makes up the pixels of the picture. By changing one, or at most two, of the least significant bits of each byte, the resulting image still looks like the original and any changes that occurred are barely perceivable. In audio files, messages are transmitted through the encoding of the least significant bit (similar to image files), phase coding (the phase of a particular audio segment is replaced with a reference phase which represents the hidden data[1]), and spread spectrum (the hidden data is encoded over the entire spectrum of the audio file[1]).

As stated in the thesis summary, the uniqueness of this project lies in the fact that the medium used to transmit the secret message won't be any of the aforementioned media, but rather a composition. The challenge in modifying a composition is that the resulting stego-text should be just as melodic sounding as the original so as to not alert unsuspecting listeners to the presence of a message. In this project, I have developed a basic algorithmic procedure that is general enough to be used with any secret message and composition pair to produce a stego-text that can be melodically performed.

Algorithm:

The meat of this project is the algorithmic procedure used to insert the message which for this example will be "Joe Smith to bomb Central Station on Thursday" into the cover medium which will be the *Allemande* from Cello Suite No. 4 in E-flat major by Johann Sebastian Bach. The message has 38 characters and the piece can hold a 40-character message, so this message/cover medium pair is perfect. I took care to develop a procedure that could be used for any message length and for any composition. In the further development section, I discuss alternate steps that could be utilized for this process given more time and resources. Displayed beneath each step will be the execution of that step using the example message and composition mentioned above. In the before and after section, one can see the finished product and compare the original piece to the stego-text.

1. Turn the plaintext message into its numerical value based on the letters. A=0, B=1,…, Y=24, Z=25.
    a. Note: The spaces shown are for demonstrative purposes only; they are not encoded to anything as the English language is redundant enough that messages can be understood without spaces.

Message:	**J**	**O**	**E**		**S**	**M**	**I**	**T**	**H**		**T**	**O**
Letter #:	9	14	4		17	12	8	18	7		18	14
Message:	**B**	**O**	**M**	**B**		**C**	**E**	**N**	**T**	**R**	**A**	**L**
Letter #:	1	14	12	1		2	4	13	18	16	0	11
Message:	**S**	**T**	**A**	**T**	**I**	**O**	**N**		**O**	**N**		
Letter #:	17	18	0	18	8	14	13		14	13		
Message:	**T**	**H**	**U**	**R**	**S**	**D**	**A**	**Y**				
Letter #:	18	7	19	16	17	3	0	24				

Figure 1

2. Encrypt the message. For demonstrative purposes, I'm going to use a Caesar cipher (moving each letter a fixed number of spaces down the

alphabet). However, in real life application, it would be beneficial to use a more difficult encryption method to ensure security.

a. Fixed number (n) = 3
b. The equation: message $+_{\text{mod } 26}$ 3 = encrypted message

Message:	J	O	E		S	M	I	T	H		T	O
Letter #:	9	14	4		17	12	8	18	7		18	14
Encryption #:	12	17	7		20	15	11	21	10		21	17
Message:	B	O	M	B		C	E	N	T	R	A	L
Letter #:	1	14	12	1		2	4	13	18	16	0	11
Encryption #:	4	17	15	4		5	7	16	21	19	3	14
Message:	S	T	A	T	I	O	N		O	N		
Letter #:	17	18	0	18	8	14	13		14	13		
Encryption #:	20	21	3	21	11	17	16		17	16		
Message:	T	H	U	R	S	D	A	Y				
Letter #:	18	7	19	16	17	3	0	24				
Encryption #:	21	10	22	19	20	6	3	1				

Figure 2

3. Observe the key signature of the composition to be used (for example, E^b major). Then, count how many unique letters are in the message: that's the number of unique notes needed to encode the message into notes. Starting with the tonic note and going up the major scale, assign notes to the letters of the encrypted message based on the frequency of the letter. (For example: highest frequency = E^b, next highest = F, etc.) If more than one letter has the same frequency, assign the notes based on which letter appears first in the message. Once the major scale is exhausted, use notes that are in the key signature but either higher than the ending scale pitch, or lower than the beginning scale pitch. If these notes are exhausted, use accidentals closest to the scale. In the current example:
   a. Key signature: E^b (this is the scale we will use to assign notes)
   b. Number of unique letters: 17 letters
      i. Note: Because a two-level scale covers 15 letters, we will need two notes outside of the scale. Here, I chose to use two notes lower than the beginning pitch.
      ii. Note: In my subscript system using E_1^b for an example, the 1 refers to the lowest E^b that a regular flute can play, closest to middle C. E_2^b would be the next highest E^b, and so on. C_0 actually corresponds to the note middle C on the piano.
   c. Assign notes based on frequency:

Letter:	Encryption #:	Frequency:	Note Assigned	Letter:	Encryption #:	Frequency:	Note Assigned

			:				:
T	21	6 times	E_1^b	N	16	2 times	G_2
O	17	4 times	F_1	R	19	2 times	A_2^b
S	20	3 times	G_1	J	12	1 time	B_2^b
A	3	3 times	A_1^b	C	5	1 time	C_2
E	7	2 times	B_1^b	L	14	1 time	D_2
M	15	2 times	C_1	U	22	1 time	E_3^b
I	11	2 times	D	D	6	1 time	C_0
H	10	2 times	E_2^b	Y	1	1 time	D_0
B	2	2 times	F_2				

Figure 3

d. Now that all the letters have been assigned a note, the message can now be inserted into the composition. Choose a fixed beat of each measure to change into the "stego-note". Then insert the message one letter per measure by changing the first note (not including grace notes) of the fixed beat to the appropriate stego-note. Below, the tabular form of which letter goes to which measure is displayed. The finished product (the modified composition) can be seen in the before and after section.
   i. Tempo: (4/4) time
   ii. Fixed Beat: beat 3 of each measure
   iii. Insertion of each letter into a measure:

Message:	**J**	**O**	**E**		**S**	**M**	**I**	**T**	**H**		**T**	**O**
Letter #:	9	14	4		17	12	8	18	7		18	14
Encryption #:	12	17	7		20	15	11	21	10		21	17
Note, Measure #:	B_2^b 1	F_1 2	B_1^b 3		G_1 4	C_1 5	D_1 6	E_1^b 7	E_2^b 8		E_1^b 9	F_1 10
Message:	**B**	**O**	**M**	**B**		**C**	**E**	**N**	**T**	**R**	**A**	**L**
Letter #:	1	14	12	1		2	4	13	18	16	0	11
Encryption #:	4	17	15	4		5	7	16	21	19	3	14
Note, Measure #:	F_2 11	F_1 12	C_1 13	F_2 14		C_2 15	B_1^b 16	G_2 17	E_1^b 18	A_2^b 19	A_1^b 20	D_2 21
Message:	**S**	**T**	**A**	**T**	**I**	**O**	**N**		**O**	**N**		
Letter #:	17	18	0	18	8	14	13		14	13		
Encryption #:	20	21	3	21	11	17	16		17	16		
Note, Measure #:	G_1 22	E_1^b 23	A_1^b 24	E_1^b 25	D_1 26	F_1 27	G_2 28		F_1 29	G_2 30		
Message:	**T**	**H**	**U**	**R**	**S**	**D**	**A**	**Y**				
Letter #:	18	7	19	16	17	3	0	24				
Encryption #:	21	10	22	19	20	6	3	1				
Note, Measure #:	E_1^b 31	E_2^b 32	E_3^b 33	A_2^b 34	G_1 35	C_0 36	A_1^b 37	D_0 38				

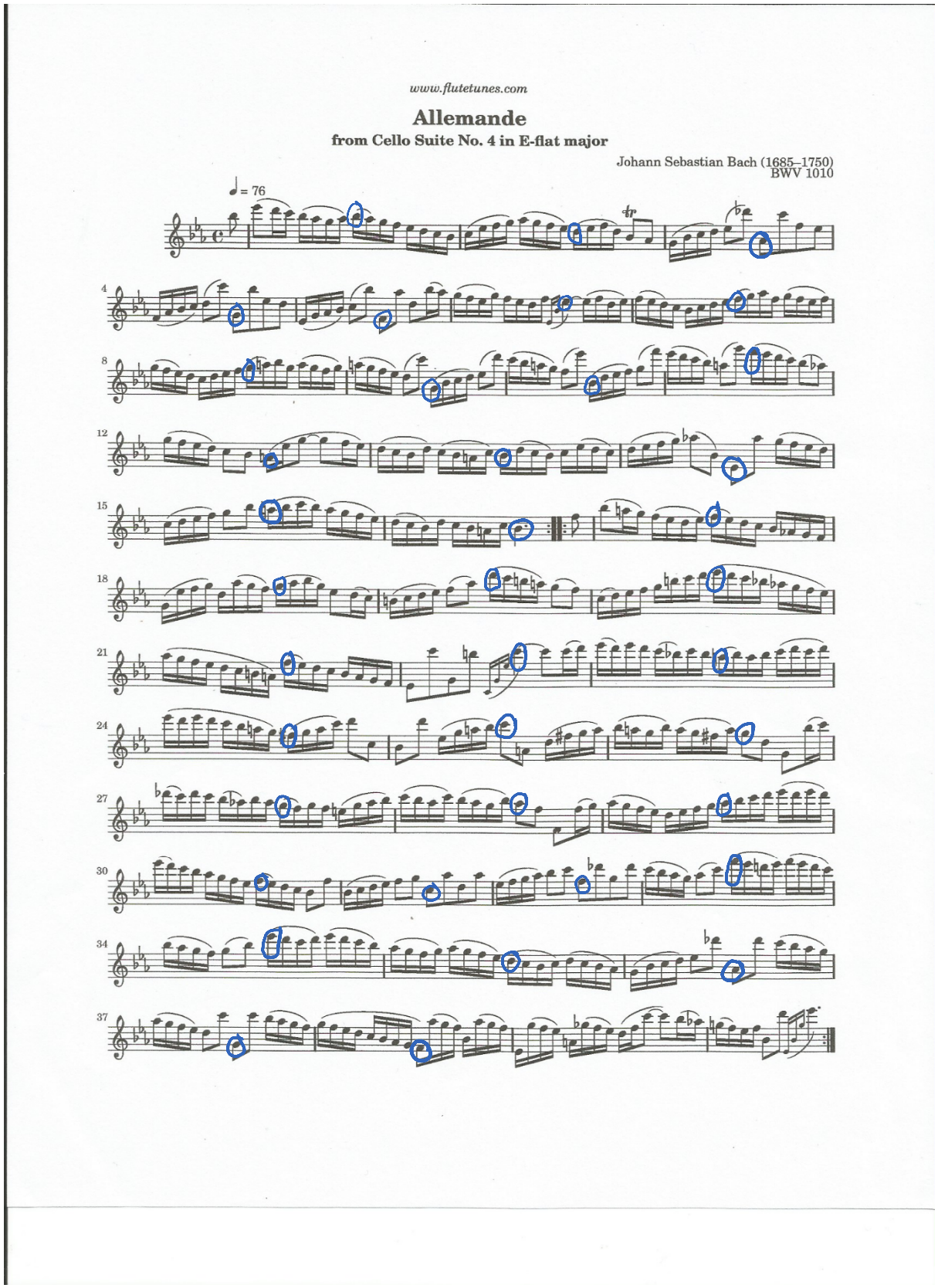
Figure 4

Further Comments:

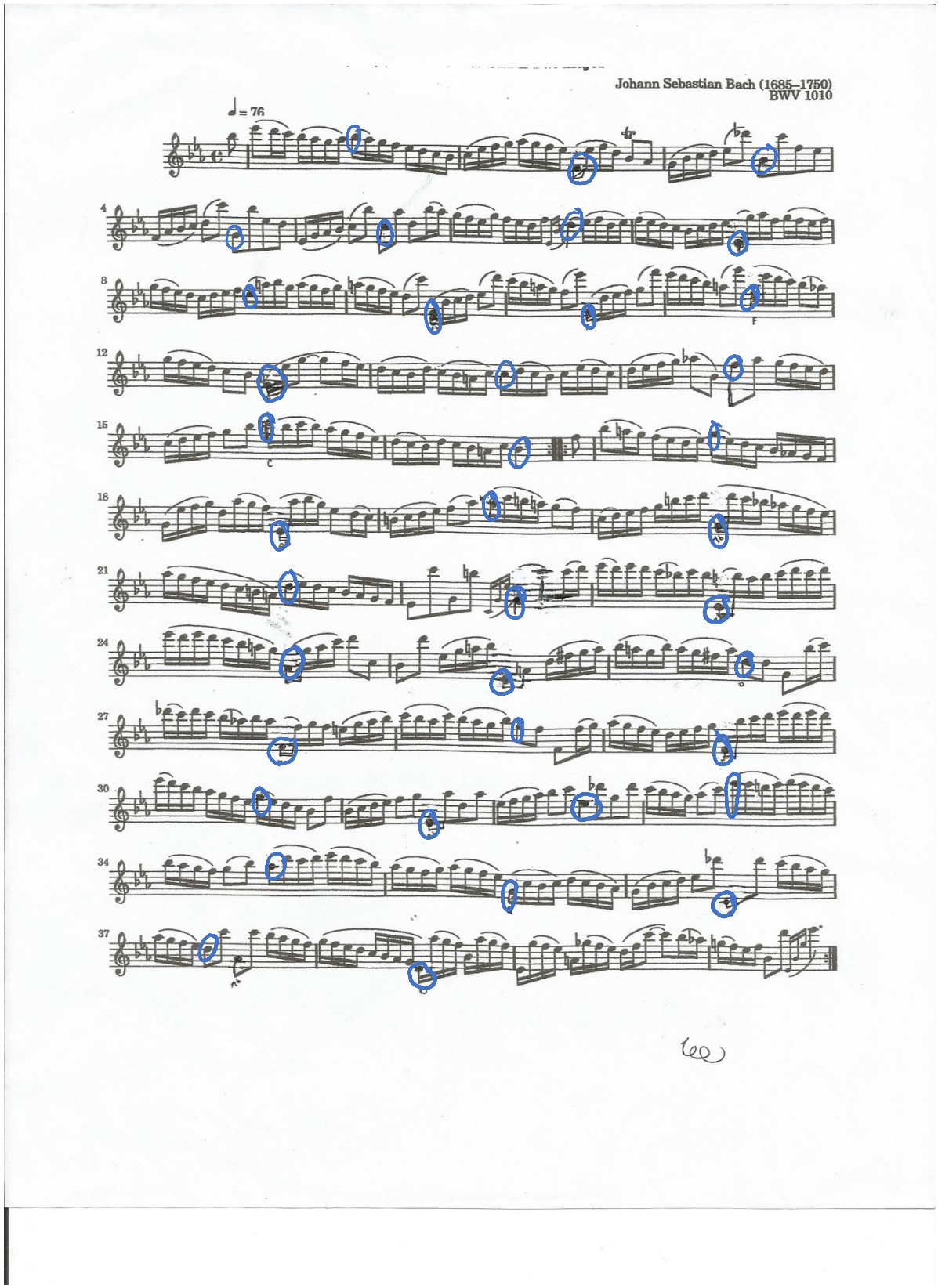
In order for Bob (the recipient) to be able to extract the message from the music, there are a couple of facts that he needs to know prior to the transmission of the message. He needs to have the original composition, know which beat of each measure is modified with the “stego-note”, know the encryption method used, and the three rightmost columns of ***Figure 3*** in order to be able to know which note goes to what encryption number.

When Alice (the sender) plays the modified piece in the park, Bob will record the performance. Using a tuner, a pitch finder, or simply a piano he can then identify the modified note per measure and record such. Using the chart, he can then transform the sequence of notes into the encrypted message. Next, he would decrypt the message: in this example, he would subtract (mod 26) three from each numerical value in order to get the numerical values for the original message. Finally, he would turn the values into their “corresponding letters” and end up with the resulting message.

Before Modifications (Original Composition):



After Modifications (Stego Composition):



Before and After (comments):

The circled blue notes represent the first note of beat three of every measure (not counting grace notes). Upon close examination of the piece, you will see that there are three instances where the original note is the same as the stego-note (measures 1, 4, and 16). The rest of the measures differ between the original note and the stego-note. The process used to modify the original composition involved me whiting-out the notes that were different and then replacing the note with the stego-note written in a fine-point pen. Because the message and the composition was relatively small (less than 40 characters), this process of whiting-out each note and hand-writing the replacement was feasible. For larger composition/message pairs however, it is recommended that one uses software such as Finale Notepad which is free and can be downloaded from the internet. It allows the user to manipulate scanned-in compositions saved in the correct format (.midi or .xml files).

Further Development:

In this section, I want to discuss some directions that could be explored with more time and resources.

There are numerous different ways to decide which note in a measure is modified into a “stego-note”. In the example, I simply picked a fixed beat and manipulated the same beat in every measure. However, an alternative solution would be to stagger the changed note in every measure in order to raise the level of security. An example of staggering would be changing the first note in the first measure, the second note in the second measure, the third note in the third measure, the fourth note in the fourth measure, cycling back to changing the first note in the fifth measure, and continuing on for the duration of the piece. A second alternative solution to picking a fixed beat every measure and inserting the message in order from start to finish would be to have a pseudo-random generator spit out a list of measures in random order and changing a fixed beat in the measure. In this case, Bob would then require either the same pseudo-random generator with the same parameters, or a list of the order of measures so that he could correctly recover the message.

The next topic worthy of discussion is the technology that Bob uses to retrieve the message. If Bob had music writing software such as Finale, he could get the midi form of the original composition and open it up in Finale to hear what the piece sounds like. He could then compare the recording of Alice’s playing to the original piece and mark the discrepancies in the two pieces. Going along with this train of thought, he would no longer have to be made aware of which

beat will be modified because he could compare the two pieces (unless there are instances in which the stego-note is the same as the original note). Bob could also use a piano to identify what the modified notes are.

Another development to consider would be creating music software that has the ability to record what's being played and write it out as sheet music (Finale does not yet have this capability). If this kind of software was available, Bob wouldn't need a copy of the original piece for analysis. He could simply sit in the park and listen to Alice play her instrument while his laptop records and writes out what's being played. From there, armed with the knowledge of what beat was modified and which note corresponds to what letter, he could extract the message. Since there does not yet exist any technology capable of accurately penning down sheet music solely off of a recording, another method to consider is Bob using musical software such as Melodyne (a pitch correction plug-in)[3]. This software allows users to freeze a recording on a particular pitch so that it can be modified at the discretion of the user. It also identifies what note the frozen pitch is. Such technology would definitely speed up the time it takes Bob to extract the message from the composition.

Works Cited:

1. Dunbar, Bret. “A detailed look at Steganographic Techniques and their use in an

   Open-Systems Environment”. *SANS Institute InfoSec Reading Room.* 18 Jan. 2002. http://www.sans.org/reading-room/whitepapers/covert/a-detailed-look-at-steganographic-techniques-and-their-use-in-an-opensystems-environment-677?show=a-detailed-look-at-steganographic-techniques-and-their-use-in-an-opensystems-environment-677&cat=covert . Feb. 2014

2. Kessler, Gary C. “Steganography: Hiding Data Within Data”. *Windows and .Net*

   Magazine. Sept. 2001. http://www.garykessler.net/library/steganography.html . Feb. 2014

3. “What is Melodyne”. *Celemony_.* http://www.celemony.com/en/melodyne/what-is-melodyne . 10 Apr. 2014