RICE UNIVERSITY

Violin Virtuoso: A Game for Violin Education

by

Linda Hill

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Science in Computer Science

APPROVED, THESIS COMMITTEE:

Joe Warren, Chair & Professor
Computer Science

Ronald N. Goldman, Professor
Computer Science

Kenneth Goldsmith,
Professor of Violin

HOUSTON, TEXAS
DECEMBER 2011

ABSTRACT

Violin Virtuoso: A Live Analyzed Music Game

by

Linda Hill

This study develops a way to use the technology of video games to help teach people the violin. The unique characteristics of this research are to create a visual representation for music for violinists, to utilize pedagogy for violin education within a video game, and to use the Fast Fourier Transform to process input from an actual acoustic instrument and use the data to track user progress. Music games like *JamGuru* [9], *Rocksmith* [19], *LittleBigStar* [13], and *Rock Band* [18] have made strides in these areas of research; however, most of these technologies have not been applied to the violin. Expected results are that students' progress rate will increase as a result of playing the game. The input processing used in this research could be applied to areas where sound recognition is important. Furthermore, the concept of an adaptable game can be applied to other academic subjects.

ACKNOWLEDGMENTS

I would like to thank several people for their help in completing this thesis. Violin Virtuoso began as a project in Rice University's computer science class, Comp 460. The motivation for developing Violin Virtuoso was to create a fun video game in the style of *Guitar Hero*. The expected result was that users would be able to learn the basics of playing the violin and would be motivated to play the violin because of the game. Thank you to Sanjay Chatterjee and Shams Imam for their contributions to this project.

I would also like to thank my committee. I would like to thank Dr. Joe Warren for advising me during my thesis and for his insight and constructive help related to video games. I would like to thank Dr. Ron Goldman for his advising and instruction while I have attended Rice. Thank you to Mr. Kenneth Goldsmith for being a wonderful violin teacher for me for several years. In addition I would like to thank my Suzuki Book 1 teacher trainers, Rhonda Cole and Lisa Lederer, whose instruction helped me develop the pedagogy for Violin Virtuoso. I would also like to thank the rest of the computer science department at Rice University for their knowledge and guidance. Thank you to my mom for her never-ending love and support during my masters. Thank you also to my fiancée Jeff Bridge for his encouragement.

TABLE OF CONTENTS

Page

1. INTRODUCTION 1
   1.1 Motivation for Creating Violin Virtuoso 1
   1.2 Motivation for Creating Serious Games 3
   1.3 Approaches to Serious Games 5
   1.4 Approaches to Music Games 8
2. MUSICAL REPRESENTATION 15
   2.1 Traditional Notation 15
   2.2 *Lily Pond* Notation 18
   2.3 Onscreen Representation 21
3. GAMEPLAY 26
   3.1 User Interface 26
   3.2 User Feedback 29
4. INPUT PROCESSING 35
   4.1 Fast Fourier Transform and Harmonic Product Spectrum .. 35
   4.2 Violin Input Processing 37
5. PEDAGOGY 41
   5.1 Level One 41
   5.2 Level Two 42
   5.3 Level Three 44
   5.4 Level Four 46
   5.5 Testing 49
6. CONCLUSIONS 49
7. REFERENCES 53

Violin Virtuoso: A Game for Violin Education

Chapter 1

Introduction

1.1 Motivation for Creating Violin Virtuoso

Guitar Hero is an extremely popular video game to come on the market recently. This game uses a controller shaped like a guitar with five buttons to simulate playing a guitar in time to popular music scrolling down a screen. The game is extremely addictive, and users play for hours to master difficult songs.

Guitar Hero actually succeeds as a simulation game in many ways; however, aspects of playing the guitar are missing. The game covers more breadth in learning music and the guitar than the finer details of guitar technique. A real guitar has 20 or more frets as opposed to the mere five buttons on the guitar controller. These buttons are aligned vertically, which mimics a single string, but an actual guitar has six strings. In addition, pressing a button does not always yield the same note whereas pressing a string down at a specific fret would (Arsenault, pg. 2 [1]).

Each button can be said to represent one fret which covers six strings, a defense for having so few buttons. Therefore, going back and forth as well as in a linear order on the buttons for notes that appear to be in sequence makes sense. On a real guitar, one goes back and forth with the fingers even when pitches are progressing in one direction if

the person playing is not shifting up the neck of the instrument. However, no real shifting or moving between strings will actually ever occur with this controller.

In addition, harmony is somewhat simulated. The typical rock guitar "power chord," a perfect fifth is often used. This chord is played on the guitar controller by holding down two buttons, which is similar to what occurs when playing a chord on the guitar. However, the missing piece is extensive chord simulation, which involves a rapid change of position of many fingers between frets instead of changing only two buttons (Arsenault, pg. 3 [1]).

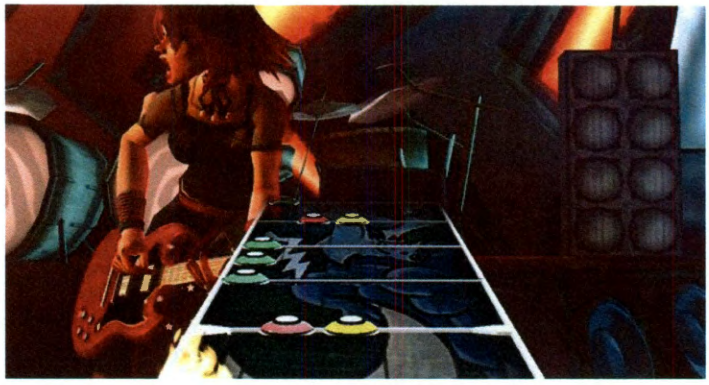

Figure 1: Guitar Hero [3]. The green, red, and yellow notes move toward the bottom of the picture along the fingerboard.

Last, *Guitar Hero* is described as a rhythm game. When a note is displayed at a specific time on the screen, the user must match the correct button on the controller. The notes scroll down what looks like a fingerboard of a guitar, and the user must play the note on the guitar when the note reaches the front of the fingerboard [Figure 1]. However, if the user misses the timing for this note, then the sound of the guitar cuts out completely until the user gets back in rhythm. This is something that never occurs when someone is actually playing an instrument. In *Guitar Hero* hearing the incorrect playing

and having some visual feedback would be a more realistic and useful approach (Arsenault, pg. 6 [1]).

Generally the user playing *Guitar Hero* gets a broad sense of melody, rhythm, and harmony with a relatively simple controller compared to an actual guitar. However, the game is not structured as an actual way to learn an instrument. By creating Violin Virtuoso I take advantage of the fun aspects of *Guitar Hero* and use them with an actual violin, not a simulation.

1.2 Motivation for Creating Serious Games

Most of the students in schools and employees receiving training have grown up in a digital age. The way students perceive and process information has changed dramatically, and education needs to adapt to remain effective. Employees perceive corporate training as being incredibly boring. Escaping the office is one of the primary motivations for employees to attend training, not for any intrinsic value in the training. Although new technologies have been attempted such as using CD-ROMs and internet training, the lack of engaging material has remained the same. CD-ROMs end up sitting unused on shelves, and internet training completion rates are often less than 50% (Prensky, pg.3 [19]). In schools, many children are resistant to education not because school is hard, but because it is so boring (Prensky, pg.7 [19]). For these reasons training games are important.

Training becomes something that people are happy to engage in, and the training becomes more effective as well when games and fun are incorporated into learning. The idea is to bring together serious training and the ever growing industry of entertainment.

Anecdotal evidence is given for the importance of developing serious games in a paper by Michael Zyda [27]. One of the reasons given is the degree to which youth spend large amounts of time absorbed in playing video games. The author frequently encounters mothers whose children spend hours playing such games as America's Army (MOVES Institute and the US Army). Consequently, the children have gained a tremendous amount of knowledge about the US Army. There would obviously be tremendous benefits to creating similarly engrossing, yet educational, games that could exploit what the author calls collateral learning, the ability to learn things outside of traditional classroom methods (Zyda, pg. 27 [27]).

There are several effects of game-based learning. Even the driest and most boring subjects will be fun to teach. Games created by small groups of people will teach entire populations, businesses, grade levels, and industries. Currently, only students who have access to "star" teachers have the advantage of a great educational experience. Creating fun educational games will allow a much wider audience to have quality instruction. Fun training online will become more prevalent as talent and creativity on the web win out over boring training (Prensky, pp. 17-18 [19]).

Violin Virtuoso has the ability to be an educational experience that brings the violin to a previously untapped audience. Although its primary use will be as a teaching aid for students already receiving private instruction, Violin Virtuoso will be accessible to complete novices. In addition, in future work, the game will be available open source on the web.

1.3 Approaches to Serious Games

Research in the area of video games, specifically serious video games, can have a positive impact on other areas including virtual reality, corporate and government training, and education. Zyda also discusses important aspects to consider in serious game research.

One of these is the importance of story. The gaming industry has experimented in the past with a genre called edutainment, which consisted of educational software combined with game-like interfaces as an afterthought. Unfortunately, this combination was unsuccessful, which indicates the need for developing the story first and inserting the pedagogy into an already well-crafted game (Zyda, pg. 29 [24]). This is the approach used in Violin Virtuoso where the play of the game is heavily based on the mechanics of *Guitar Hero*. The aspect of violin pedagogy is inserted into the game as a means to advance game play.

The Levee Patroller game was created with the purpose of developing a more standardized approach to serious games. The main tensions identified in serious games are that a game needs to be fun, instruct the user with the necessary material, and be realistic (Harteveld, pg. 1 [5]). Levee Patroller resolves these issues by mirroring reality as much as possible. The landscape is in 3-D and very closely resembles the regions visited by a patroller. Another realistic aspect is that elements such as weather are taken into account. For example, if there is rain in the scene, the visibility in the game lessens just as visibility would during an actual inspection (Harteveld, pg. 9 [5]). To make the game more fun, the user can decide how many levees will fail or allow the computer to choose, making the game more unpredictable and thus more enjoyable. Pedagogy is

implemented by incrementing the levels of the game as the user masters the skills and responsibilities encountered in the current level.

Another training game concerned with developing the overall genre of serious games is the Flooding Control Trainer. One of the most challenging aspects of developing this game was resolving the conflicts created by working with professionals from many diverse backgrounds. While all members of the team agreed on the importance of achieving learning objectives, implementation strategies varied widely (Hussain et al., pg. 29 [7]). For example, the instructional designers and assessment experts wanted the gaming platform to be chosen based on pedagogical needs. However, the product release timeline and customer preferences dictated what type of platform should be used (Hussain et al., pg. 21 [7]). Determining the different levels of the game was another consideration. The team members agreed that the game should be layered; however, discrepancies existed between the instructional designers, assessment experts, and the game developers as to which elements should occur in each level (Hussain et al., pg. 22 [7]). A similar tension arose between these groups regarding instructional strategy versus gaming strategy. Initially the game developers were able to easily incorporate instructional strategy into the game design. As the game progressed, however, the instructional designers and assessment experts requested more feedback and increased scaffolding, which conflicted with the game designers' effort to create an immersive game (Hussain et al., pg. 24 [7]).

One of the options not explored by the above two serious games is the ability for the user or an instructor to modify the game for a specific student. In my research, I will be focusing on the creation of modular units which allow teachers to add their own

training materials to my game. An example of a method that has had success in this area is the "Digital and Modular Design Scheme based on education theory in RPG Learning Game (DMS-e RPGLearning)". Using this design, teachers can create independent RPG games for specific learning components. In addition several RPG units can be combined into a larger game platform. The teacher or designer creates a drama for the game which can be broken down into several scenes. The standardization of the overall design ensures that the game is created to incorporate five parts: to establish the ability of the student, choose level-appropriate content, teach the content through an action or activity, evaluate the student, and provide a reward or feedback based on the student's progress (Shu Hui Hsu and Ming-Shen Jian, pg. 1734 [22]).

One creative idea the designers of this method have developed is to allow the student to choose a character for the game play based on the current level. The student feedback after a lesson occurs in the form of trigger which rewards the student and moves the action forward (Shu Hui Hsu and Ming-Shen Jian, pp. 1734 [22]). A game created using the modular RPG system was tested on only 58 5^{th} grade students studying geometry. Although the students showed a 20% increase in skill level after playing the game, the results are taken from an extremely small sample of students, and no control group is indicated (Shu Hui Hsu and Ming-Shen Jian, pg. 1737 [22]).

In the future, Violin Virtuoso will be tested on a large pool of violin students at approximately the same level. Measurements that will be tested are the amount students have increased practicing since playing the game, improvement in intonation and rhythm, and greater retention of previous repertoire.

1.4 Approaches to Music Games

The use of immersive or embodied controllers and interfaces that use gesture as input is a large part of the popularity of the music and rhythm games. In a paper on *Rock Band* (Games and Electronic Arts [20]), Tanenbaum and Bizzocchi [23] explore possible reasons for this partiality. All games can be characterized by their ludic interaction, which is the quality of the game that relates to flow, the balance between a game being boring because it is too easy and frustrating because it is too challenging. Ludic efficiency relates to how the interface (or controller) helps or hinders the user in playing the game. Ludic optimization is the balance between a controller being complicated enough that the game is interesting but does not create a problem for the user to succeed. The guitar used in *Rock Band* (like the one used in *Guitar Hero*) requires a certain set of skills to play the game. The guitar is uniquely designed to be held a certain way, and the user must press the correct buttons on the neck to succeed at the game. The same is true for the violin used in Violin Virtuoso. Enjoyment of the game is increased by the challenge of learning the unique structure of the controller (Tanenbaum and Bizzocchi, pp. 128-129 [23]).

In addition, unlike traditional video games, having a musical controller gives the user a direct connection between his body and the game. There is a kinesthetic pleasure to playing the game in addition to the challenge of getting a good score. The fact that the controller resembles a guitar adds to the embodiment aspect of the interface. The interface causes the user to be in the role of a rock star, even with a minimum amount of movement (Tanenbaum and Bizzocchi, pp. 129 – 130 [23]). Likewise, the controller described in this thesis allows the user a certain amount of kinesthetic pleasure as well as

the ability to immerse oneself in the role of a premier solo violinist. The controller is a violin which can be classified as a 16-button controller for the beginning user. The set of gestures used to play the game with the violin are inherent in the instrument. One must move the bow and place the left hand in the correct position at the correct time. This adds to the fun of the game which means students will play the game more, practice more, and improve more quickly.

Professional guitarists find the controller in *Guitar Hero*, which is similar to the controller in *Rock Band*, difficult to operate. The controller actually hinders the ability of the guitarist to be successful at the game. Conversely non-guitarists cannot transfer their knowledge of the controller to actually playing a guitar (Tanenbaum and Bizzocchi, pg. 133 [23]). The lack of learning an actual instrument in musical games like *Rock Band* and *Guitar Hero* provides additional motivation for this thesis. A student who has mastered a specific level of Violin Virtuoso will have mastered an actual technique or musical concept on an actual violin.

Several other benefits to creating a musical game or multi-media music education system are described in a paper by Percival et al [17]. In addition to motivating the student to practice, the music game or tool can improve the quality of private lessons and provide additional ways to practice (Percival et al., pg. 68 [17]). The modular aspect of Violin Virtuoso will allow teachers to add specific tutorials and technical exercises. In this way, a student will have the benefit of a teacher-guided practice session built into the game.

Using sensors and other input to allow students to see their position while playing is another idea of how to use a multi-media tool to enhance lessons. Percival et al.

elaborate that this is a way the student can achieve a greater understanding of teacher instruction. Similarly, body sensors can be used to provide feedback to the user via a computer animation interface. As a practice aid, a multi-media tool can most obviously be used for pitch matching and correcting rhythm. Students don't always have sufficient ear training to allow for accurate pitch correction and are often reluctant to use tuners and metronomes (Percival et al., pg. 69 [17]). This is an excellent example of how Violin Virtuoso can be so valuable. Its main criteria for scoring are correct rhythm and accurate pitch.

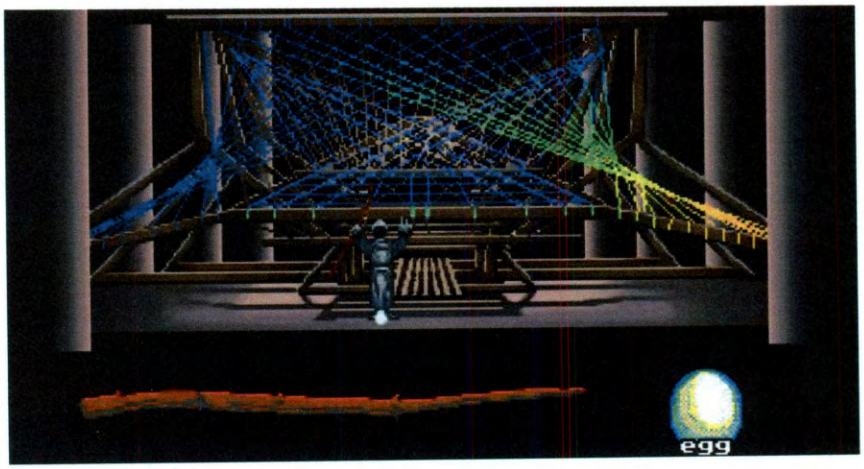

Figure 2: Loom [15]. The protagonist in the game uses music to weave spells.

Several examples of how games can be used to motivate students to practice are included in the paper by Percival et al. One type of game is a narrative, a fun story which immerses the player in the activity. An example of this game is *Loom* [15] where the user casts spells by playing melodies or replicating melodies by ear [Figure 2]. Another way to use a narrative to improve practicing is by having a skill move the action forward. For example, the player may need a ladder to get over a wall. If an exercise on an instrument is attempted but only 70% correct, the player only gets 70% of the ladder. To

get the rest of the ladder, the user may have to repeat the exercise several times at a much higher percentage (Percival et al., pp. 71 – 72 [17]).

The Digital Violin Tutor is an interesting project by Percival et al. that further develops a few of the above ideas. This multi-media tool includes video, scrolling graphical displays, 2-D animations of the fingerboard, and 3-D avatar animations. The tutor transcribes user audio and compares the transcriptions to the pitches and rhythms in the teacher's audio. Corrections of mistakes are demonstrated by fingerboard animations, the avatar, or by video. The lack of a gaming element is a possible drawback of the tool. Although the student receives feedback related to performance, there is not a sense of play or fun. This might render the tool less effective than a game which has similar feedback but also allows the user to feel a sense of challenge and accomplishment.

MEAWS (Musician Evaluation and Audition for Winds and Strings; pronounced as either "Muse" or "Miaos") is another tool produced by Percival et al. This tool focuses primarily on motivating the student to practice exercises. The exercises are presented as tests, and a grade is given based on the student's performance. This tool is intended for use with an instructor and a larger curriculum. MEAWS is specifically concerned with providing tools for practice rather than specifically motivating the student (Percival et al., pp. 72- 73 [17]).

Lately, several games in addition to the Digital Violin Tuner and MEAWS have been in production that attempt to incorporate the technology of using an actual instrument to create a more realistic experience. Power Gig is a game in the tradition of *Guitar Hero* and *Rock Band.* This game supplies a scaled down version of a guitar that can be played in a color-coded fashion, mimicking the keys from *Guitar Hero.* In a more advanced

mode, the colors on the game are replaced by the string number. This allows for a slightly more realistic experience of playing the guitar; however, many of the reviews state that the game is still too unrealistic to be a large improvement over the lack of guitar pedagogy seen in *Guitar Hero* and *Rock Band.*

Figure 3: Jam Guru [9]. The notes in this game stream down the screen to indicate specifically where the left hand should be.

Jam Origin has released electric guitar software called *JamGuru* [9] using their OFFBEAT engine. This software was preceded by *LittleBigStar* [14], designed by one of the cofounders of *Jam Origin*, and is available online. *JamGuru* lets the user learn songs by seeing them scroll across the screen. The game then processes the user's playing and provides feedback. This game has a different layout from some of the games that preceded it such as *Guitar Hero* and *Rock Band.* Instead of using the fingerboard as a means for the notes to travel and indicate the string, the notes simply stream down the screen and land on the fingerboard to indicate left hand position [Figure 3]. This seems somewhat complicated, and may not be easy for the user to decipher. The main benefit of *JamGuru* is that the user can play any guitar with the game. The songs are special versions for use with the game and are purchased on an individual basis. In addition, the game comes with a tuner and has several tutorials to help you learn the songs more

easily. The documentation does not state if the game is designed with a specific pedagogy to teach beginner guitarists.

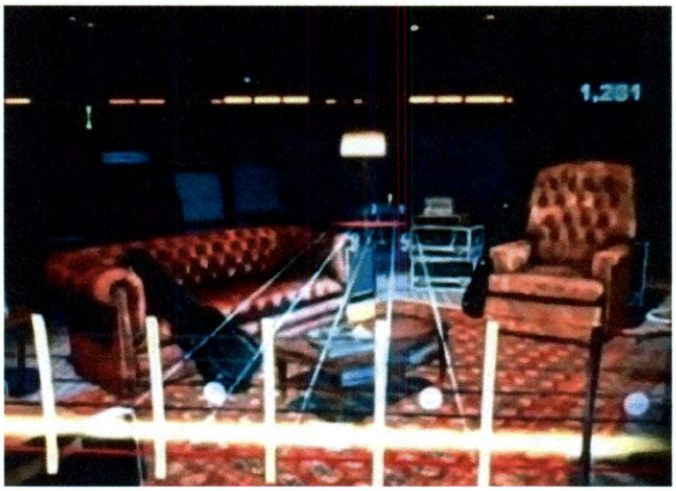

Figure 4: Rocksmith [21]. The notes stream down specific frets which are numbered. The string for the next note is highlighted on the screen.

Rocksmith [21], created by Ubisoft for PC, XBOX 360 and PS3 is another realistic guitar game that has just been released this October. *Rocksmith* was originally called *Guitar Rising* and was developed by Gametank. This game is getting positive reviews from several sources including *Rolling Stone Magazine* and *Bits N Bytes Gaming*. The user can play the game with any real guitar, similarly to *JamGuru*. *Rocksmith* also includes an extensive library; however, the user is unable to add additional songs. Like *JamGuru*, the notes stream down the screen and land in place on the guitar fingerboard. However, instead of streaming unguided, the notes travel along the fret where they will be played, and the appropriate string is highlighted [Figure 4]. In addition, *Rocksmith* seems to have incorporated a logical way for the player to advance. The game senses how many the notes the user is able to play and adjusts the difficulty of the song

accordingly. Also included to help the player progress are mini games and chord charts. This game has not yet been released; however, the early reviews seem promising.

My Violin is a software tool designed to teach children how to play the violin. This tool includes 160 violin lessons and mini games as well as tutorials to help the student advance. Also included are a metronome and a violin tuner. The drawback to this game is that the curriculum is completely predesigned. Individual teachers and students are unable to adapt the game to their own needs. Its value seems to be primarily in teaching students who are not already taking private violin lessons.

Even with all the exciting musical games and tools available my thesis will provide a unique contribution. Many of the musical games are fun but provide no real training; however, the current tools being used to teach violin do not combine a large gaming element. My thesis aims to combine fun as well as a structured pedagogy to provide real training of the violin. The controller used in the game is an actual acoustic violin, and the progression of the game instructs the player from the very basics of violin technique through more advanced pieces. In future work, Violin Virtuoso will be extensible, allowing teachers to expand the curriculum and create tutorials and exercises to play in the game.

Chapter 2

Musical Representation

Transcribing traditional western musical notation into something that could be represented in a video game was a major challenge in developing Violin Virtuoso. I used a musical notation language created for *Lily Pond* [13], an open-source music notation software, as a means to transcribe traditional musical scores into a file that could be read into the computer. Several aspects of traditional Western music were considered in order to represent the pieces used in Violin Virtuoso accurately including the time signature, the key signature, the range of pitches of a set of notes, the specific pitch of one note, the duration of a single note, how to indicate rests, and violin notation such as fingerings and bowings.

2.1 Traditional Notation

Western music is written on what is typically called a *staff*, which consists of five lines allowing pitches to be placed in relation to their frequency and duration. Figure 5 is an example of one measure of music on a staff. The staff reads from left to right, and the pitches, also called notes, are placed on the staff in the order in which they occur in the piece. The higher a note is on the staff, the higher its frequency and vice versa. The pitches are printed in varying shapes with items such as dots and stems used to denote the duration of the note. At the beginning of the staff is a symbol called a *clef*. This indicates the range of frequencies that will be displayed on the staff. Violin Virtuoso is

primarily concerned with the pitches that fall in the range of the treble clef. In order to accommodate the entire range of the violin, ledger lines are used above and below the staff.

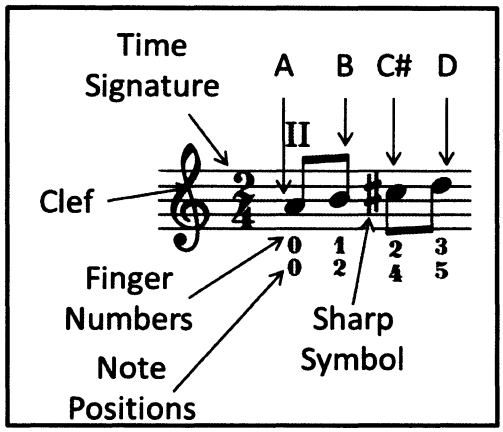

Figure 5: The staff. An example of one measure of music showing the treble clef, the $\frac{2}{4}$ time signature, the notes A, B, C#, and D, fingerings, and note positions.

Immediately following the clef is the time signature, which provides the basic rhythmic value in the music. The most basic note value in music is the *whole note*, which can be subdivided into many different values. One of the most common values is the *quarter note*, or ¼ the value of a whole note. In Figure 5 the $\frac{2}{4}$ is the *time signature* of this particular piece of music. This means that each *measure* of music contains the equivalent of 2 quarter notes. The example in figure 4 is one measure of eighth notes, which is the mathematical equivalent of two quarter notes. The distinctive feature of the eighth notes is the single beam connecting two notes together. The number of notes connected by a beam is inconsequential; however, each beam divides the time value of the note in half. For example, two notes connected by two horizontal beams would be sixteenth notes. A dot placed next to a note has the opposite effect of a beam. The dot

adds half the value of the note to itself, i.e. a dotted quarter note would be the equivalent of a quarter note plus an eighth note.

The pitches of notes are arranged alphabetically using the letters A through G. Going forwards in the alphabet means going higher in frequency while going backwards has the opposite effect. Before each note there may be sharp, #, or flat, *b*, symbols. Many notes are simply different spellings of the same pitch. For example, a c-sharp (C#), which raises a C by a half-step is the same as a d-flat (D*b*), which lowers a D by a half-step. These symbols indicate that a pitch is to be played either slightly higher or lower.

Bowings are an important characteristic specific to violin (and other string instruments) sheet music that Violin Virtuoso takes into account. The bowings tell the violinist how many notes are to be played in a single bow stroke and the appropriate direction of the bow. Usually a violinist will play notes "as they come" where the bow moves in a different direction for each note. However, there are several ways to indicate different types of bowings. The symbols, V , and ⊓, indicate that a note should be an upbow or downbow respectively, regardless of the direction of the previous or following note [Figure 6].

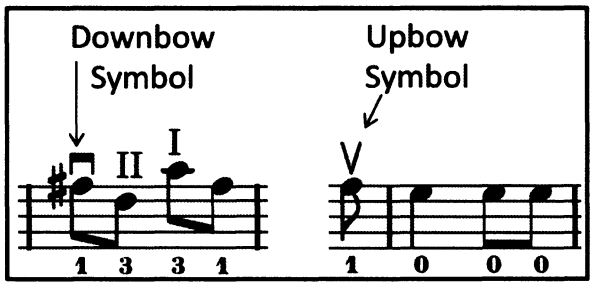

Figure 6: Bowings. The first measure begins with a down bow. The second measure is preceded by an upbow.

A slur is an arc, ⌒ , which connects all of the notes under the slur to be in the same bow. Likewise, a tie, which looks the same as a slur, can connect notes of the same pitch in one bow, effectively creating a note with a long duration.

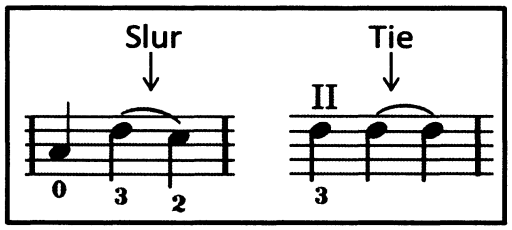

Figure 7: Ties and Slurs. A slur connects notes of different pitch while a tie connects notes with the same pitch.

A few additional properties are included throughout the music such as specific tempos, dynamics, and musical expression. Violin Virtuoso is not currently concerned with these characteristic; however, they may be included in future work.

2.2 *Lily Pond* Notation

As an intermediate representation between traditional Western notation and the onscreen musical notation, the songs used in Violin Virtuoso are notated using the language developed by the creators of *Lily Pond* [13], a free, open-source musical notation software. The parser used by *Lily Pond* to typeset music encompasses all the variations of notation encountered in Western musical notation. Rather than importing this entire parser I chose to write my own, which allowed me to focus specifically on the issues found most often in violin sheet music.

The parser I developed is designed to create an entire song that can be displayed during game play. The *Lily Pond* file is processed in a loop until all the characters have been accounted for. Due to the consistency in the files used in Violin Virtuoso, much of

the parser is hard-coded. In the first switch statement, the parser determines if the incoming character is going to be followed by a descriptive word, an actual note, or a specific metronome marking. After the required action is handled, this loop is repeated until the end of the file is reached.

The parser searches for specific words found at the beginning of the song. These descriptors include references to the relative pitch range of the song and whether the song begins with an incomplete measure, otherwise known as a "pickup note". The relative pitch is a field that remains constant while parsing the song to aid in determining the pitches of individual notes. The current assumption is that sharps and flats are identified individually for each note in the *Lily Pond* song file; however, future iterations of Violin Virtuoso will also handle key signatures. Occasionally a descriptive word occurs after the processing of notes has begun for a song file. The parser is designed to identify when this occurs and return to parsing text. At this time, the only occurrences of this are related to bowings and grace notes.

Determining individual notes requires attention to many details. The first important criteria are finding out the name of the pitch and whether or not a modifier such as a sharp or flat has been attached to the name string. From here the parser must go through successive characters to ascertain the correct range for the note, or if, in fact, the note is a rest. The range of the note is saved to be used when computing the pitch of the next note. Next the parser identifies the duration of the note and computes any additional dotted value. The *Lily Pond* language assumes that a note's duration is the same as the previous note unless indicated otherwise. This convention holds for string assignment and bow direction as well. In computing bowings, the note parser is able to handle slurs and ties,

which are used to connect two or more notes in the same bow, in addition to the explicit upbow and downbow text. Fingering and note position on the violin fingerboard are indicated for each note and are parsed successively. In addition, the parser uses the information about the note name and register to tell Violin Virtuoso the pitch frequency of the note.

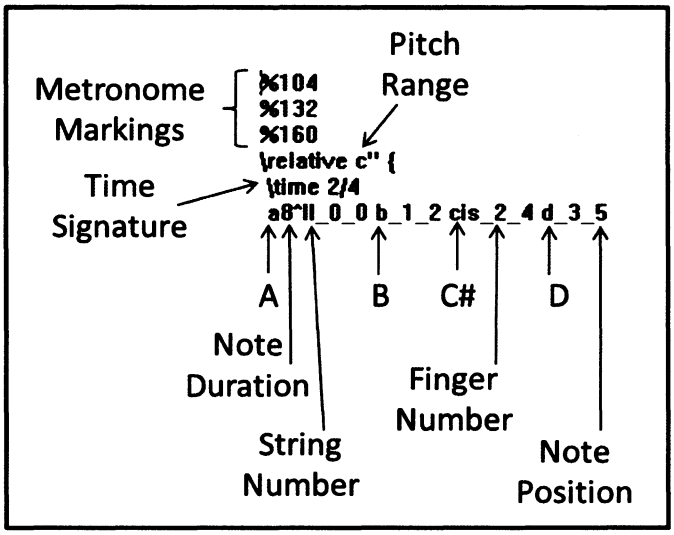

Figure 8: Lily Pond. A short example showing metronome markings at the top followed by the pitch range, time signature, notes A, B, and C# as well as their durations, finger numbers, and positions.

As soon as all the characteristics of a note have been determined, a note object specified by the note duration is created. Several lists are maintained related to the note. The first is a list of all the notes in the song. The second list subdivides the song into beats, currently an eighth note, and stores notes in segments the length of one beat. An offset field is maintained for notes whose durations are either less than or greater than one beat. This field plus a note's duration dictate its place in the beat list. As soon as an entire beat has been filled this list of notes is added to the list of beats. These lists are used by the game to determine when to display each note.

2.3 Onscreen Representation

The objective of the onscreen representation of the sheet music is to make the music easily accessible to the gamer. The overall visual of the game is very similar to *Guitar Hero* or *Rock Band.* The violin can be thought of as the 16-button controller for the game [Figure 5]. The black portion of the violin, located on its neck, is the fingerboard and is where the violinist places the fingers of the left hand to create different pitches. The four white lines across the fingerboard are actually tapes and are used so the gamer knows where to put each finger. The colored circles on the fingerboard simply demonstrate the colors that will be used on the screen to let the user know which string to play and are not present on the user's violin. The user's four fingers can be placed on any of the four tapes, on any of the four strings, thus creating a 16-button device.

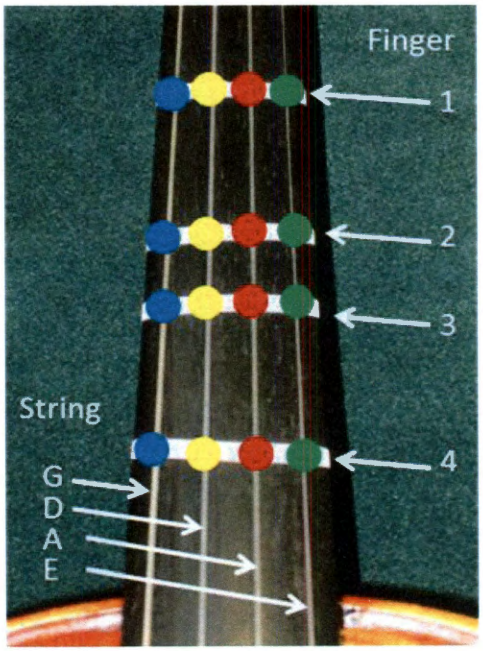

Figure 9: Violin as 16 Button Controller. The fingerboard with white tapes spanning all four strings. The colored dots demonstrate the 16 buttons, but are not visible on the violin.

In later levels, the number of buttons on the controller increases because the fingers navigate between the tapes and further up the fingerboard as more notes are introduced in the songs. The user "activates" the buttons by drawing the violin bow perpendicularly across the string being played.

There were several iterations of the gameplay screen before deciding on the current design. Initially the screen was created so the notes would scroll vertically down the screen. The notes were to travel from the middle of the screen along one of four lines which represent the four strings on the violin. The notes were color-coded, blue, yellow, red, green, based on the string they traversed. At the bottom of the screen were large circles the same color as the notes on the string. The user was expected to play a note when the note reached the large circle. Both violin teachers and game developers who were presented with this idea decided that it was too complicated for the user. A new design was needed that would allow the user to look in only one area of the screen during gameplay.

The next iteration saw several changes. The fingerboard was removed and the button name was placed directly in the center of the note. A discussion arose as to whether note names or finger numbers in the middle of the note would be easier for the user. Although many violin teachers preferred only having note names, this was seen as impractical since users who have never played the violin may have no prior knowledge of note names. To make the game more accessible at the beginning level, finger numbers were placed in the notes. Computer keyboard input was introduced as a way to debug the game in the beginning stages. The following keys: 1, 2, 3, 4; Q, W, E, R; A, S, D, F; and Z, X, C, V were used for the different notes on the four strings. In addition, the keys \, Enter, Shift,

and Ctrl were used, one for each string, as "strum bars" to mimic bowing on the appropriate string. A horizontal layout was intuitive due to the horizontal nature of the computer keyboard. The notes scrolled horizontally across the strings, and the user played the note when it reached the other side.

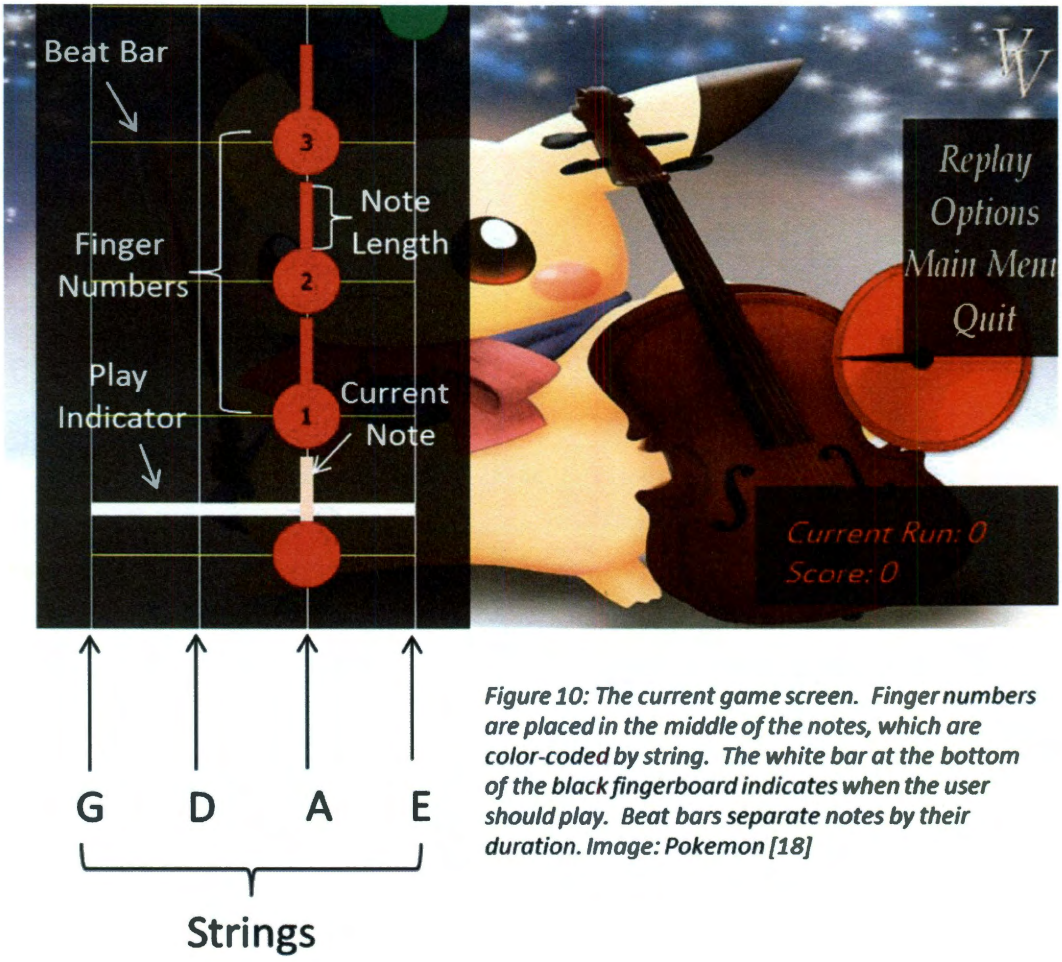

Figure 10: The current game screen. Finger numbers are placed in the middle of the notes, which are color-coded by string. The white bar at the bottom of the black fingerboard indicates when the user should play. Beat bars separate notes by their duration. Image: Pokemon [18]

After the sound processing of the violin was implemented into the game, I was able to test the game with a violin and reevaluate the screen design. The decision was made that vertical scrolling was actually more effective than horizontal when holding a violin. The user can point the violin at the screen and look at the violin fingerboard and bow placement as well as the notes scrolling down the screen. Initially the strings were drawn

against the picture on the game-screen. This made the notes and strings difficult to see while playing the game, so a black background was chosen for the area directly behind the strings. This black background also resembles the violin fingerboard. Near the bottom of the fingerboard is a wide line. When the note reaches this line, the user plays the corresponding note on the violin [Figure 10].

I experimented with creating different shapes for the notes to indicate the appropriate fingering; however, this, combined with actual finger numbers and string colors seemed overcomplicated for the user. Because there is a one to one correspondence between the note names and the finger number, the notes are all circles and the finger numbers are placed directly in the middle of each note. A zero represents playing an open string without fingers.

There has been some discussion about changes to the note design as the game begins to incorporate more difficult levels. As the songs become more difficult, the user has to play notes that are not on the original fingerboard tapes. One way to indicate these additional notes will be to add a + or – to the finger number in the middle of the note. In much more advanced levels, the user will need to move the hand further up the fingerboard. To allow this to happen naturally, the finger numbers will gradually be removed and replaced by the note names. The user will be able, at this point, to learn which note names correspond to specific places on the fingerboard and will become less dependent on the tapes.

The parser includes the tempo from the Lily Pond file in the song. The tempo determines how closely the notes follow each other while moving down the string; however, the rate at which the notes move stays constant regardless of the tempo. Along

with the notes, beat bars also scroll down the fingerboard. The beat bars are horizontal lines separated by eighth note intervals. The interval between beat bars may be something the user can modify in the future. The timing of the note is indicated by how many beat bars are between the current note and the previous note [Figure 10].

The parser is responsible for collecting a list of beats. The beats are made up of a certain number of notes depending on their length. The notes included in a beat are subdivided according to their duration and placed on the beat bars accordingly. For example, an eighth note followed by a quarter note (the equivalent length of two eighth notes) would have no beat bar between them since the beat bars are an eighth note apart; however, the note following the quarter note would be placed with one beat bar between itself and the quarter note. The user is expected to play the note with the violin bow in one continuous stroke for the entire duration of the note. The length of the note is shown by a tail that extends from the note, in the style of *Guitar Hero* and *Rock Band.* Usually the tail of the note will end directly before the next note begins; however, if the user is expected to pause because a rest follows a note, the tail will end before the user must play the next note, and the next note will begin at the end of the duration of the rest [Figure 10].

Currently the game does not handle bowings; however, future versions will include points for correctly playing specific downbow and upbow markings as well as slurs and ties. Because the game is currently for beginners, most of the songs do not have any bowings marked, so the primary emphasis is on playing the correct pitch and rhythm, which is a considerable achievement.

Chapter 3

Gameplay

There were two main technical considerations in creating the gameplay for Violin Virtuoso. The first was ease of the player's ability to understand the game and quickly begin playing. The second was the user's ability to process feedback and progress through the game. These issues are related to, but not dependent on the pedagogical aspect of the game. A natural interface and sufficient feedback are necessary components for a successful game and allow the pedagogy to be successfully implemented.

3.1 User Interface

One of the main concerns for the user is the ability to understand the notes on the screen and when to play them. Therefore, the first thing to determine was the overall layout of the screen. As stated previously, the two options were to have the notes scroll horizontally across the screen or vertically with user feedback items such as accuracy meters and the score below or next to the fingerboard respectively. A vertical orientation with notes scrolling down the strings on the fingerboard was determined to be more intuitive for a gamer playing the violin.

Another crucial element was how and when the notes should appear on the screen. The notes first appear at the top of the screen, and the user is expected to play a note when the note lines up with the highlighting beat bar close to the bottom of the screen. Several speeds were tested for the notes to travel down the screen, and the decision was

made that a slow speed with notes grouped closely together was the most accessible format for the user. In addition, the notes are colored based on the string on which they appear, and the finger numbers in the center of the notes are written as large as possible. Future iterations of the game will include a fingerboard with a 3-D perspective, which will also allow for more notes to appear on the screen at one time.

Figure 12: Hard Menu. Includes four songs, navigation buttons (for use if more songs are added), and a back to main menu button.

http://www.vladstudio.com/wallpaper/?violin

Tests with students playing the game showed that the ability to intuitively understand the game play varied depending on the student's level of gaming experience. Very young and/or inexperienced students need either a demo or video showing the basic mechanics of the game. For future work, several videos of the game will be posted on a game website showing songs from different levels. In addition a "screen shot" video of the game being played will be included in the intro level of the game. The game is designed to be fun and accessible for both violin students and gamers with a casual interest in the violin. In order for gamers with no experience playing the violin to succeed at the game,

several pictures will be included in the tutorial levels. These will show the correct placement of the violin and left hand and the bow and bow hand.

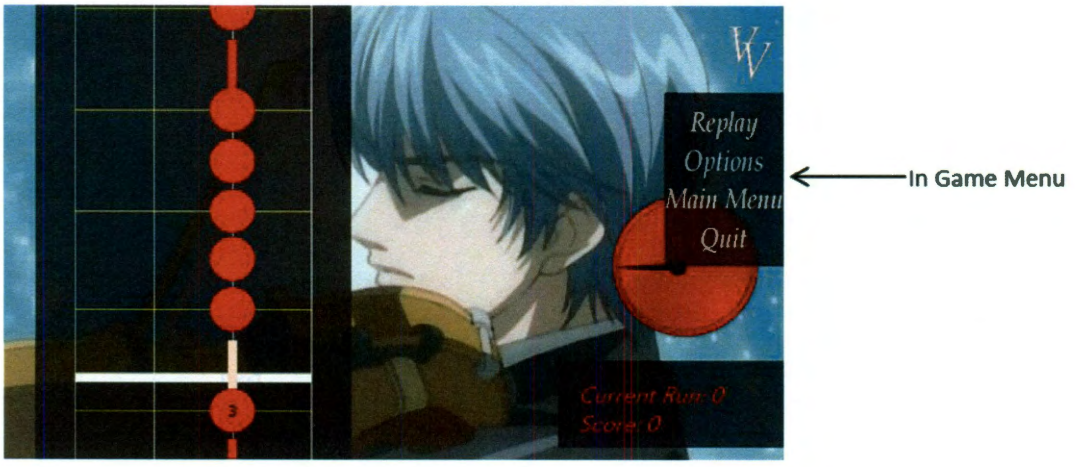

Figure 13: In Game Menu. User can replay the song, access options to turn metronome/song on or off, go back to main menu, or quit game. Image: La Corda d'Oro [12]

Another design consideration of the game was the accessibility of the menus. The game includes a splash screen which transitions to the first menu. This menu allows the user to choose which level they would like to play. There are four levels: beginner, easy, medium, and hard. After the user chooses a level, the next menu allows the user to choose one of several songs. Each menu contains arrows which allow the user to scroll through the songs with the mouse and a back button to return to the previous menu [Figure 12]. In the future, each level will also include the option of accessing a tutorial menu specific to the songs in that level. These tutorials will contain exercises to help users improve their playing ability and maximize their score in subsequent songs.

On the game screen is a menu which is accessed by hovering over the "Menu" button. From this menu, the user can choose to replay the game, return to the main menu, quit the game, or access more options [Figure 13]. The additional options are to turn the

metronome or the song audio on or off; however, the song audio is currently not implemented. Another option to be included is the ability to modify the metronome speed. Future testing by teachers and students will help to determine additional features that need to be modified or added.

3.2 User Feedback

User Feedback, or the ability for the user to know if the correct action has been performed, is one of the key aspects of a video game. This feedback is especially crucial for Violin Virtuoso since success indicates learning specific skills. Currently, the main criterion for scoring is playing the correct note at the correct time. This is indicated to the user in several ways. When the note hits the stationary bar at the bottom of the fingerboard, this bar changes color; green for playing correctly and red for playing incorrectly. In addition, the note itself is changed from its original color (based on the string) to a gold star when played correctly [Figure 14].

Getting the note to change from its original shape to a gold star without a significant amount of lag was a major challenge. Several things were attempted before an immediate change was possible. The process requiring the most time is comparing the note played by the user with the note expected by the game.

This comparison was initially accomplished by keeping track of the game time and the timing of each note. When a note from the song being played reached the bottom bar line, the game would request an analysis of the note played by the user at the same time in terms of total game time. The way data from the XNA microphone class was being requested created a lag of up to at least 100ms between when a note is played and a gold

star can be displayed on the screen. Unfortunately, switching to the NAudio audio class, which allows requests to be made at intervals of 50ms, did not shorten the lag. In addition, the time required to keep track of the user audio was system time instead of game time. Because the game time is contingent upon certain details in the programming code, these two times were not always the same. In non-trivial songs this inconsistency between the two songs created considerable lag as the song progressed. Often by the end of the song, the note being analyzed was completely off the screen before a gold star could even be processed.

One attempt at solving this was to redesign the bottom beat bar in such a way that the user played the note early enough to allow analysis before the note was off the screen. However, having both a game and system clock still created an unacceptable amount of lag by the end of the song. To fix this problem, keeping track of time was completely removed from the note analysis procedure. Every time a note reached the top of the beat bar, the game would tell the computer to record the violin, and when the note was through the beat bar, recording would be stopped. The lag problem appeared to be solved by both moving the beat bar to allow for the 100ms delay and removing the time component. Unfortunately, tests with songs containing very short notes failed using this method. Calling start and stop recording for each note when the notes are extremely close together is very expensive and actually created segmentation faults.

The final result involved several changes. The XNA microphone class was discovered to allow requests to be made at extremely small intervals, so the code was adjusted to use the XNA microphone class instead of the NAudio audio class. When a new songs starts, a call is made to the game to start recording the user input. The

recording is never asked to stop until the end of the song. Because starting and stopping the mic from recording was so time consuming, the game relies on simply calling for an analysis of the note whenever the note hits the top of the bottom beat bar. This improved method has eliminated any noticeable lag during gameplay.

An internal timer is maintained which tracks when a call for note analysis is made. Because the mic sampling frequency is at a rate of 44.1Kbps, the game can ask for the data received by the mic at a specific time in the song. For example, if a note is played between 1.1s and 1.5s according to the internal timer, the byte(1.1 * 44.1K) to byte(1.5 * 44.1K) is the data the game needs to analyze for that note. In addition, because XNA allows for data to be requested at very small intervals, the game can ask for the most recent batch of data at virtually every update. The game can process and store the data as soon as the data becomes available. When a note analysis request is made, the data has already been processed which reduces most of the lag. In addition, only one timer is necessary. At the beginning of a song, the system time is stored, and the difference between the current system time and the time recorded at the beginning of the song is used to calculate the elapsed time of the song.

In addition to changing the shape of notes to gold stars, other visual elements are used to help the player realize that he has played correctly. Tails are used to help the user play the correct duration of the note. The tail of the note extends vertically out of the note past the beat bars. If there is no break between one note and the note following it, the tail of the first note almost connects to the second note. If the user begins the note on time and continues to hold the note through the tail, then the tail will become highlighted the same color as the gold star as the user reaches that part of the note. If the user begins late, the

tail will never highlight for that note even if the gamer bows the note for the rest of the duration. In addition to feedback from the tail turning gold, the user's score is also affected by how well he sustains the note during the tail.

The tail is implemented in the shapes class. Whenever a note is created, a shape for the note is designed based on the type of note. In the case of the tail, the shape is based on the note belonging to the tail. Initially the tail is a single rectangle which is divided into several segments. The segments are universally the same size throughout the game; however, the number of segments varies based on the length of the tail's note. Each tail segment receives a tail shape creating an array of rectangles within the original rectangle shape. The last rectangle shape in the array is adjusted to the length of the note so the shape does not exceed the total tail length. The tail also has three colors which are updated at various times. The base color is the color that comes with the note. For example, if the note is played on the A string, then both the note and the tail will be red. The tail turns a lighter shade of the base color when it is analyzed because the user should be playing that segment of the tail. Finally, if the tail is analyzed and the user has played this part of the note correctly, then the tail will turn gold.

Other feedback devices relate directly to the score. To the right of the screen is a dial which changes with how well the user is doing. The song starts with the dial in a neutral position, pointing straight up, and the background color of the dial is a shade of orange. As the user begins to improve his score the dial rotates to the right, and the background starts to become green. If the gamer begins to lose points then the dial moves in the other direction and the color of the background turns red [Figure 14].

Below the dial is a score box containing the score and the current run value which is the number of notes in a row the user has played correctly. The score is calculated by the beat class which keeps track of which notes are being displayed on the screen. The beat class updates the current note and its shape and color. To update the note's shape and color an analysis is begun of the note. This analysis loops for the duration of the note. If the note is determined to match the expected frequency, then the score is incremented and the current run count is incremented. In addition, the shape and color of the current note is changed to a gold star. If the analysis discovers that the gamer missed the beginning of the note, then a forced missed occurs which increments the number of notes the gamer has missed. During and at the end of the song a text box appears at the top of the screen corresponding to the percentage of correct notes. The comments range from suggesting more practice to giving praise, and the color of the text changes to match the dial [Figure 14].

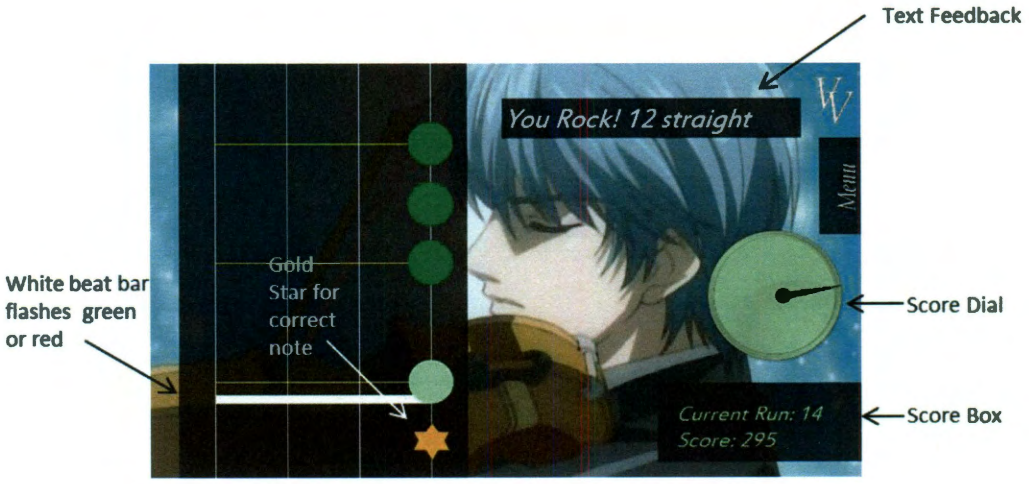

Figure 14: During Game Feedback. Correct notes become a gold star. Note textual feedback, score box and score dial. Image: La Corda d'Oro [12]

At the end of the game is a text box of statistics for the game such as the current run, longest run, the percentage of correct notes, the number of notes played out of time, the number of correct notes out of the total notes, and the overall score [Figure 15].

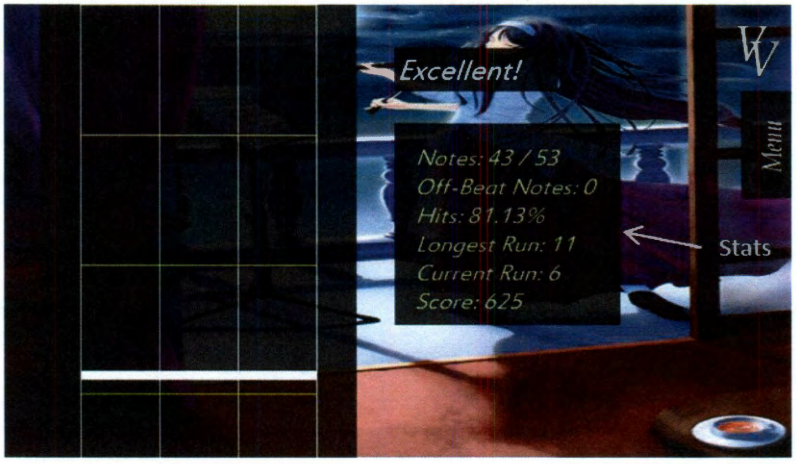

Figure 15: End of game stats. Included are number of correct notes, number of notes played late, percentage of accurate notes, longest run, current run, and overall score. Image: Tsukihime [26]

Chapter 4

Input Processing

The Fast Fourier Transform was used to process the data from the violin. This combined with the Harmonic Product Spectrum has resulted in an extremely accurate prediction of the notes played by the user. The processed signals are stored in easily accessible batches for use during game play. This results in online processing and almost instantaneous feedback for the user.

4.1 Fast Fourier Transform and Harmonic Product Spectrum

Fourier Transforms can be used to analyze the signals of systems including sound produced by an instrument. The Discrete Fourier Transform formula is a sum which allows a signal to be processed in discrete sections despite the difficulty caused by a finite signal and continuous frequency. The first complication, that a signal must be finite to compute its spectrum, is solved by assuming that the signal extends over [0, N – 1]. The second issue, that the frequency variable is continuous, can be solved by computing the spectrum at only a few frequencies, such as the equally spaced ones $f = k/K, k \in \{0, \ldots, K-1\}$.

The resulting Discrete Fourier Transform is:

$$S(k) = \sum_{n=0}^{N-1} s(n) e^{-\frac{s\pi nk}{K}}, \mathrm{k} \in \{0, \ldots, K-1\}$$

Where $S(k)$ is short for $S\left(e^{j2\pi\frac{k}{K}}\right)$. (Johnson [10]).

The Fast Fourier Transform is an algorithm for computing the Discrete Fourier Transform, which is of $O(N2)$ complexity, with a complexity of $O(N\log N)$ for certain length outputs (Johnson [11]). The FFT produces a spectrum of all the frequencies of the signal in a

given time. The spectrum of a given sound wave is the frequencies of the sound wave and their respective strength in a given time block. The accuracy of the spectrum can vary depending on the sample rate and the number of measurements per time segment. If the sample rate is 48,000 samples (of sound pressure) per second these can be divided into different size bins. As the number of samples per bins increases, the likelihood of accurate pitch detection also increases. However, determining the precise time a frequency was produced becomes more difficult as the bins become larger. In Violin Virtuoso a combination of these two approaches is used. Each bin is relatively large; however, a sliding window is used so the frequencies in the bin are calculated several times. This effectively creates several smaller bins within the larger leading to more precise time measurements.

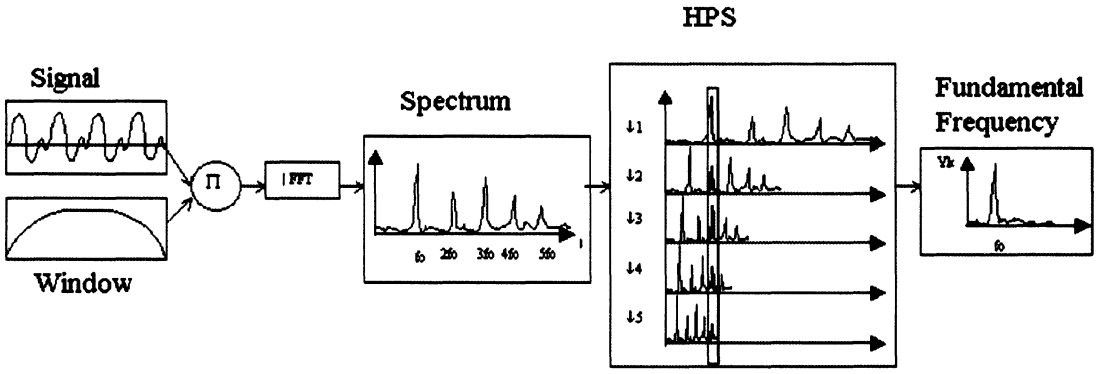

Figure 16: Harmonic Product Spectrum. The signal is processed by the FFT into a spectrum of frequencies. The fundamental frequency is the result of reducing taking the average of integer reductions of the spectrum, the harmonic product spectrum.

Any output signal from an instrument has harmonics, including the voice or the violin, because any physical system is usually non-linear. Only completely isolated systems might produce a pure sine wave as an output signal. Because the sound wave of an instrument is not just a pure sine wave, there will be many peaks in this spectrum. When analyzing a pitch from an instrument, one needs to find the fundamental frequency. One way to accomplish this is with the Harmonic Product Spectrum [Figure 16]. The Harmonic Product Spectrum finds the correct pitch by compressing the spectrum of the sound wave by a preset number of integer factors. The

harmonics line up one by one at the fundamental pitch. When the compressions are average, there is a strong peak at the fundamental frequency, but nowhere else (de la Cuadra, [2]).

4.2 Violin Input Processing

The violin input is collected as an array of data, and the game produces a Fast Fourier Transform from this data. A common FFT algorithm, #, the UIUC FFT algorithm from the University of Illinois Urbana-Champaign [9], has been ported into C# to produce the FFT used in this program.

A separate portion of the program computes the Harmonic Product Spectrum processing which handles the harmonics produced by the violin frequencies. The harmonics are caused by the unique way the violin body resonates when a string is played. The violin is made up of several pieces including a bridge, sound post, bass bar, f-holes, and front and back plates. The front and back plates of the violin form a sound box and have resonant modes. The bridge is placed on top of the front plate under the strings. When the bow moves across the string, the vibrations cause the bridge to rock back and forth, sending an entire set of harmonics through the front plate (Hsieh, pg. 31 [6]). The sound post is positioned under one of the feet of the bridge, stabilizing this side of the bridge, which allows the rocking of the bridge from the other foot to send vibrations of large amplitude up and down the front plate. The bass bar is glued to under the front plate near the other foot of the bridge [Figure 17]. This design keeps vibrations from being sent to inefficient higher resonant modes (Gough, pg. 28 [4]).

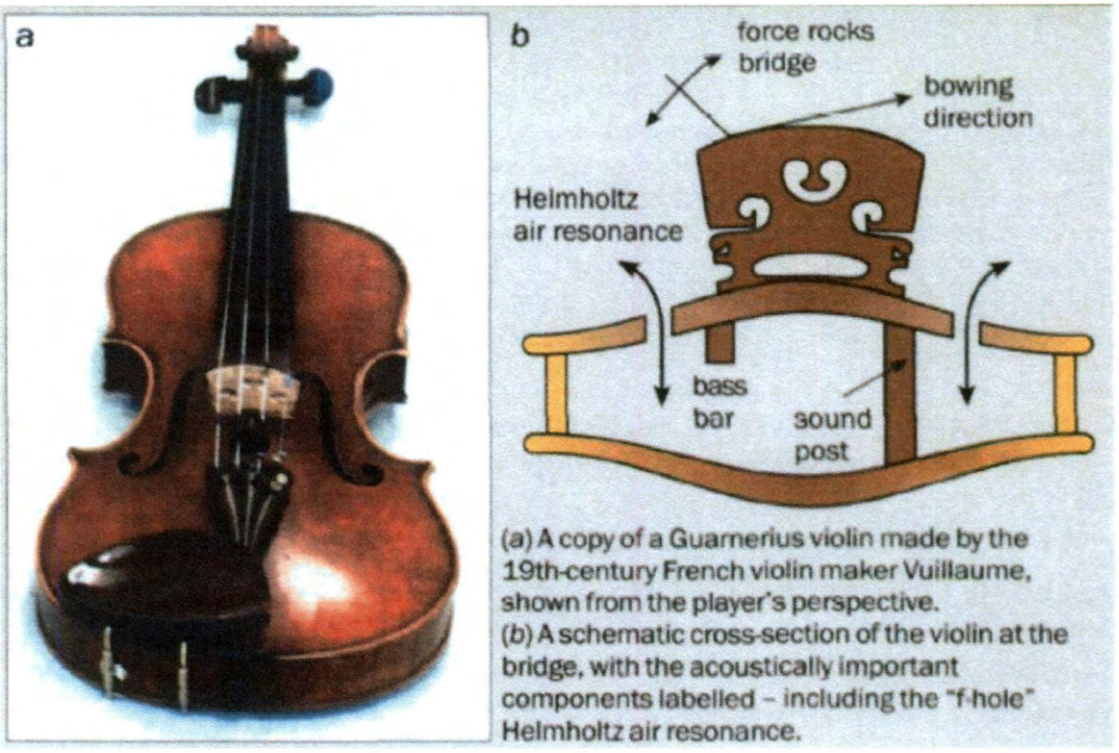

Figure 17: Violin Cross-Section (Gough pg. 27 [4]).

The harmonics sent by the bridge to the front plate are amplified differently depending on the resonant modes of the plate. F-holes are carved on either side of the front plate and the air moves backwards and forwards through the f-holes which is called the Helmholtz air resonance. This resonance affects the vibrational modes at higher frequencies and results in greater sound output in the lower frequencies. The area of the f-holes, the volume of the violin body, and the shape of the front and back plates determine the resonant frequencies of the instrument (Gough, pg. 28 [4]). Violin makers can tune both the front and back plates to enhance their vibrations. The most prominent resonant modes are the second and fifth harmonics of the plate. Although the FFT of a vibrating string could be measured and the correct pitch consistently analyzed, the harmonics amplified by the violin's body corresponding to its resonant modes produce very different results. The result is that the spectrum of the sound from the violin varies as much

between different notes played on one violin as the same note played on different violins (Hsieh, pg 32. [6]). The below diagram [Figure 18] shows the FFT output from a string in isolation, the violin body itself, and from the combination of these two factors. Thus the sound produced by the violin differs dramatically from either of the two sets of vibrations produced individually.

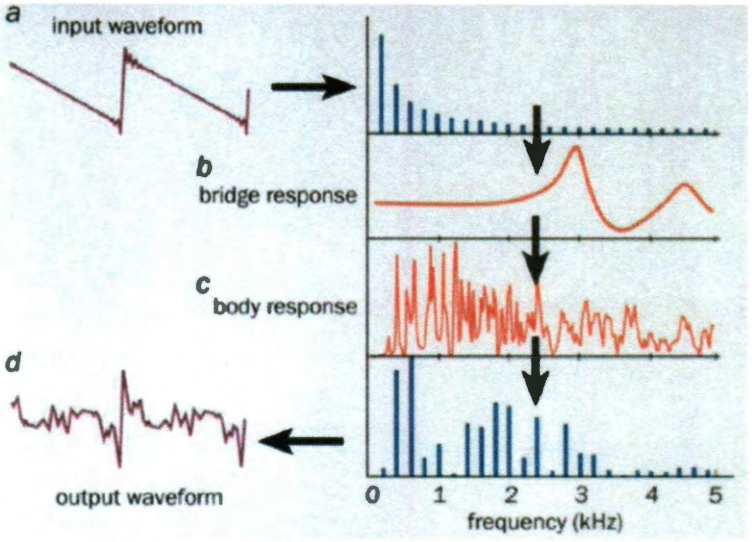

Figure 18: Violin Signal Spectrum (Gough pg. 28 [4]).
a. FFT of vibrating string
b. Response from bridge
c. Harmonics of pitch amplified by violin's body.
d. The FFT of the multi-resonance acoustic output.

To handle all of these harmonics, the program computes the geometric mean of the input data from the violin after the data has been processed with the FFT. Next the FFT is run and the harmonics are averaged. The averaging and FFT processing is only run a set number of times based on the sampling rate and the expected number of harmonics that would be captured. XNA allows a sampling rate of 44.1 kHz, which is used in Violin Virtuoso. With each geometric mean, some of the harmonics will be close to 0 as opposed to some harmonic of the actual note's frequency. Only the actual note will never be 0 allowing the harmonics to be eliminated because this note will have the highest amplitude in the FFT at the end of the Harmonic Spectrum processing.

Another portion of the program is devoted to holding data. Specifically, the data held is the note the user has played, the start and end time of the note in the game, and the weight. The weight is given to the note based on the average of the note's amplitude and the amplitudes of all the other detected frequencies in one measurement of a bin of frequencies. A probability for the note is calculated based on this average together with a smaller percentage of the note's value from several previous measurements of the bin.

The program must connect the user interface to what is happening on the violin by wiring the part of the program that manages the data with the microphone. This part of the program interprets the bit pattern received from the microphone and turns the bits into a double value. The double is then turned into values that represent the amplitude of the frequencies from the violin. After "m" amount of data has been built up, the data is sent to be analyzed. This is the part of the program that also keeps track of the score by using the data after the data is processed to see if the user matched the note or not and then either adding or subtracting a point from the score.

The program also keeps track of data in regard to time by storing "m" amount of data, which is then analyzed by sending the data to the harmonic spectrum and using those results to create a prediction of the frequency that was received from the violin. The blocks of "m-size" data overlap by an offset, "d". Each block is processed sequentially, and the analyzer keeps track of the timing contained within each block of data. The GUI can then ask the analyzer portion of the program if a note has been processed at a specific time. This occurs when a note hits the bottom bar in the GUI. The GUI then sends a call to the analyzer to start processing the note. When the note passes the bar, the GUI then calls the "stop analyze" method. Sometimes the notes are very close together, such as at the end of the tail of the note. In this case, stop analyze is not called; however, when start is called on a new note, the stop analyze is automatically called on the previous note.

Chapter 5

Pedagogy

The pedagogy for Violin Virtuoso has been carefully structured so a user can begin playing the game immediately with no prior knowledge of the violin. The basic goal of the pedagogy is to take a beginner from not being able to play the violin at all to progressing through the main teaching points of the first few songs of Book One of the Suzuki Violin Repertoire (Suzuki [24]). To enable this, many of the levels have tutorial sections. In future work the entire first book in the Suzuki method will be incorporated, and several previews to songs as well as additional tutorials will be included in the game. I will seek permission from the Suzuki Association of the Americas before publishing any portion of this thesis or Violin Virtuoso.

5.1 Level One

The first level introduces basic bow mechanics and left-hand placement as well as a sense of rhythm. Currently level one is divided into Beginner and Easy. The Beginner level games are clapping games which let the user experience the game without having to think about how to play a violin. These games are based on the five rhythms used in the "Twinkle Twinkle Little Star Variations" as well as the "Twinkle Theme". The game mode is set to recognize any sort of audible sound at the correct rhythm instead of correct pitches. On the game screen, notes travel down the E String even though the user is only clapping his hands.

The first tutorial games included in this level one are currently placed in the Easy level and involve navigating the A and E strings without the left hand. Individual games allow the user to play one specific "Twinkle" rhythm several times first on either the A or E string and then alternating between the A and E strings. For this, the game only needs to detect the correct string

and rhythm, and the UI displays the notes as open string notes travelling down the appropriate string, A or E.

The second group of exercises introduce the left hand and are located in the Medium level. For these games, the same "Twinkle" rhythms are used that were introduced in the Easy and Beginner levels. The first games involve playing only one finger alternating with open string on either the A or E string. In the future additional games will be added to include more than one finger at a time until finally all four fingers are being played with the "Twinkle" rhythms. The final tutorial in this level is the "Monkey Song".

In the future, additional games will be included that incorporate all the "Twinkle" rhythms from the previous exercises and add the notes from the "Twinkle Theme". These "Twinkle Variations" are the first songs in Suzuki Book 1 [24]. The last game in the series will be the "Twinkle Theme," which is currently in the Hard level. Achieving an excellent score with "Twinkle Theme" indicates the user has learned more than 30 skills associated with playing the violin (Murillo [16]).

5.2 Level Two

Level two will includes several songs that build on the techniques learned in level one. Three songs that will comprise this level, "Lightly Row", "May Song", and "Song of the Wind" are currently grouped with the "Twinkle Theme" in the Hard Level. A very young violin student can take as much as a year to learn all the songs included in the game up to this point. These are the last songs currently implemented in Violin Virtuoso. The rest of the songs and exercises described in the chapter represent future work.

The previews that will be included with each song usually involve some small section of the song played with one of the "Twinkle" rhythms to make the passage in the actual song easier to

play. However, several of the songs in Level Two, "Go Tell Aunt Rhody", "Long, Long Ago", "Perpetual Motion", and "Perpetual Motion" with doubles, will not actually have previews because the skills used in these songs simply solidify previous techniques.

The first song that will be in level two is "Lightly Row". This song will have three preview games. The first one is a game where the user plays the open E string followed by second finger on the A string. This is to counteract the fact that the "Twinkle" games all have the gamer playing the open E string followed by the third finger on the A string. The next two previews are partial examples of the A scale and A arpeggio because "Lightly Row" can be divided into sections depending on whether or not they include either of these exercises.

The next song, "Song of the Wind", will have three previews. The first two previews teach the user how to hop between two strings with the same finger. Because this technique can be difficult the first time it is encountered, the previews isolate the technique. The second preview uses a small section of the song where this technique is needed. The last preview is part of future work because this preview involves bowings. The user must learn how to play two successive down-bows. Bowings can easily be parsed into the game; however detecting bowings for scoring would require an accelerometer and possibly a gyrometer to be placed on the bow much like a "wii" controller. Detecting bowings is something that will be included in future iterations of the game.

The previews that will be included in the next two songs, "O Come, Little Children" and "May Song", are also part of future work related to bowings. The difficult aspect of "O Come, Little Children" is that this song begins up-bow which is the opposite direction of most of the other songs. In addition, there are several places where the user plays an up-bow and must immediately play another up-bow. This is similar to the technique in "Song of the Wind"; however, the bow stays on the string in "O Come, Little Children".

"May Song" is the first song in which the dotted quarter note is introduced. For this reason, the user is encouraged to use most of the bow. The preview that will be included involves learning to subdivide the bow into parts which match the length of the note being played. For instance, the first note in "May Song" and its preview is a dotted quarter note which is three times longer than an eighth note. The user is asked to play nearly half the bow for the dotted quarter note and slightly more than an inch for the eighth note. Detecting subdivision of the bow is not possible without some sort of motion detection on the bow.

"Allegro", the next song, will have only three previews designed to teach the student quick finger action with the left hand. "Allegro" includes the four notes, f#, g#, a, and f# played with one of the "Twinkle" rhythms as a preview. This finger pattern is played in two different tempos in the song so each of these places is given its own preview at the tempo the piece will be played.

5.3 Level Three

The difference between Level Three and Level Two is quite large. For this section, the game must be adapted to include four more buttons on the violin. In addition, the user will be adding two more strings. This level only includes a few songs; however, many exercises and adaptations of previous songs are included to help the user learn these more advanced techniques.

The first important thing the user needs to learn to be successful in Level 3 of songs is the ability to play on the D and G strings. To make this easier, the "Monkey Song" for both the D and G strings is included, and the "Twinkle" variations from Level One

beginning on the G string instead of the A string are repeated. Additionally, both the D and G scales and arpeggios are introduced which help reinforce the use of buttons on the G and D strings.

The first new song in this level is "Allegretto". One of the most important teaching points in this song is that the song spans three strings for the first time, G, D, and A. The first two previews that will be included in this song are to help the user hop from the D string to the G string with the first finger. These previews are structured similarly to the preview in Song of the Wind involving changing strings with the same finger. Because playing the D and G strings is still new in this level, the technique is reintroduced. The other teaching point involves a bowing technique called an accent. Detecting this type of bowing articulation would involve measuring volume and bow direction which are currently not available.

The next song, "Andantino", is very similar to "Allegretto" in terms of techniques used. The D and G strings are played extensively and the E string is added briefly to encompass the entire violin. Again, one of the important techniques is creating accents with the bow which is beyond the current scope of the game. The next exercises will introduce the first big adjustment in actual gameplay. This involves creating two more buttons, one each on the A and E strings between the first and second finger buttons. This button is typically called "low 2" so the gameplay screen draws a "2-" in the center of any note that is a low 2 on the A or E strings. In musical terms, this note is called a C natural on the A string and a G natural on the E string. William Starr's The Suzuki Violinist: A Guide for Teachers and Parents has an excellent exercise that introduces low

2 using a fragment of the G major scale (Starr, pg. 101 [23]). This exercise will be used as one of the tutorials for Level 3.

The G scale and arpeggio are unique because they can both be played on all four strings to create a two-octave scale (twice as long as the preceding A and D scales). The tutorials introducing the complete two-octave G major scale and arpeggio will appear directly after the low 2 tutorial. Both exercises are played with one of the "Twinkle" rhythms.

The first song in Level 3 that uses low 2 is "Etude". No previews are necessary for this song due to the extensive tutorials that precede it. Also included are the doubles for "Etude" which, like "Perpetual Motion", asks the user to play "Etude" doubling each note so that the bow moves twice as fast and plays each note twice.

5.4 Level Four

The final level of this game is level four. This level is the culmination of all the techniques learned by the user in the previous games with the addition of a few more skills. The next skill introduced is the "high 3" followed by "hooked" bowings, changing keys and grace notes. In this level is also an introduction of the musical form the minuet, with three minuets written by Bach. A small explanation about the minuet form will be written in the preview section of "Minuet 1". Each of the above techniques will be introduced by a tutorial or a song preview in this level.

"Minuet 1" by J.S. Bach, the first song in this level, will have two previews. The first is to master the hooked bowing. This is when two up-bows or down-bows occur consecutively, often with a dotted rhythm. Again, although the bowing will be easy to

display on the screen, detecting hooked bowing is part of future work. This piece changes key from G Major to D major, which is the first time this happens in one song. This involves switching from high 2, C#, to low 2, C natural. Therefore, the second preview is an exercise involving switching between the high 2 and low 2 finger buttons frequently.

The next new skill is the high 3 which, like the low 2, requires two new buttons on the game controller. The new buttons are between the third and fourth fingers on the D and G strings and is drawn with a "3+" in the center of each note. A tutorial will be added to teach the high third finger.

"Minuet 2," the second piece in this level, will have three previews. The first is a G Major arpeggio preview using one of the "Twinkle" rhythms to break down the first measure of the piece. This is a difficult measure because it involves many string crossings with quick bow strokes. The second preview is related to the high 3 exercises which occur earlier in the level. The preview is taken directly from the second half of the piece and involves using the high three, a triplet rhythmic figure, and tricky string crossings.

The last preview for this song is to teach the user how to bow slurs. Previous songs have had a note continue in the same direction as the previous note; however, in previous songs, the bow was not required to move in a smooth motion for these connected notes. The preview for these slurs is useful for playing the song correctly; however, detecting slurs is part of future work.

"Minuet 3" is the last of the three minuets in Book 1. This piece has some left hand button patterns which are difficult, so those patterns will be introduced in a preview. In

addition, there is a complicated slur and separated notes combination that is used throughout the piece, which is also included in a preview. The last difficult technique in "Minuet 3" is that the piece changes keys in the second half of the song much like "Minuet 1".

The technique "hooked bowing" in introduced in "The Happy Farmer". To add to the difficulty of connecting two notes in the same bow, the rhythm in this particular hooked bowing dotted, as mentioned previously. The skill of playing hooked bowings with dotted rhythms will be broken down into two previews. The first focuses on the actual bowing which is a smooth bow connected to a staccato bow. After this is mastered, the user is ready for the second preview, which combines the bowing with the notes used from the first measure of "The Happy Farmer". The third preview is the same as the first preview; however, the bowing is completely smooth instead of hooked. Then the following preview adds the notes from the middle of the song where the dotted rhythm is played with a slur. The last preview for this song is from the third whole measure of the piece. Coordinating the left hand and bow with the string crossings in this measure is difficult, so the measure is broken down into slower segments.

The last piece in Suzuki Book 1 and in Level 4 is "Gavotte" by F. J. Gossec. This piece is by far the most difficult piece in the book and is designed to compel the user to solidify all previously learned techniques. One new technique that will be added in the future is the grace note. This is a very short decorative note played just before the beat leading into another note. The grace note is new in this song and will have the first preview in this level. Another difficult passage is a series of 16^{th} notes roughly half-way through "Gavotte". The difficulty of these notes is that the user must move the second

finger quickly between the high 2 and low 2 positions. The preview for this section repeats the same notes while gradually increasing the tempo. Another intricate section involves quick notes and large string crossings. This passage is broken down into approximately eight previews so the passage will be easy to play in the song. In this last preview section is the final new technique in the game, which is pizzicato. Pizzicato is the act of plucking the string with one of the fingers of the right hand. This is not terribly problematic; however, the complicated aspect is that the user must quickly transfer from using the bow to play the note to simply plucking.

5.5 Testing

The game was tested on several people ranging from children as young as 3 to adults. All participants were eager to play the game; however, certain groups experienced more success than others. Children over the age of 7 with beginning to advanced skills on the violin had no trouble understanding the user interface and completing the first level. Children younger than this, who were also unexposed to video games found Violin Virtuoso more challenging. Experiments will be made with larger and slower scrolling notes in the beginning levels. Adolescents and college students seemed to enjoy the game the most and wanted to play repeatedly. One participant was able to go from no knowledge of the violin to a recognizable rendition of the theme from Tchaikovsky's violin concerto. Many of the adults who tried the game had some trepidation due to lack of skills on the violin; however, they often continued to play the game until they could successful complete several songs.

Chapter 6

Conclusions

Violin Virtuoso creates a visual representation of music for violinists. Translating Western classical music from the written page onto a computer screen was uniquely challenging in that the notes to be played have to mirror onscreen what is supposed to occur on the violin. Several attempts were made such as horizontal scrolling and color coded notes before a successful implementation was developed. In addition, Violin Virtuoso has a systematic pedagogy to ensure the user actually learns to play the violin by playing the game. The user's playing is measured for correctness, and feedback is provided so the user can improve. This makes Violin Virtuoso a great education tool for teaching the violin. Furthermore, great care was taken to incorporate the fun gameplay elements of games like *Guitar Hero* so students enjoy playing the Violin Virtuoso, which makes the game a fun way to get students to practice.

In future work, I intend to detect bow direction by using an accelerometer so pieces with complicated bowings will have more accurate scoring. Other possible work may include sensors on the user's bow arm and left hand as a way to measure violin posture, video analysis of the user's playing, and additional sensors on the bow to detect pressure and bow position. In addition, the game will be modular allowing teachers to add their own songs and tutorials to fit their violin curriculum.

Chapter 7

References

[1] Arsenault, D. *Guitar Hero*: "Not like playing guitar at all"? *Loading...*, 2 (2).

[2] de la Cuadra, P., Pitch Detection Methods Review; [cited 2011, November]. Available from: https://ccrma.stanford.edu/~pdelac/154/m154paper.htm

[3] *Guitar Hero*. [PlayStation2]. Sunnyvale, CA: *RedOctane*, Nov. 8, 2005.

[4] Gough, C. 2000, Science and the Stradivarius. *Physics World*, 13 (2000), pp. 27–33.

[5] Harteveld, C., Guimarães, R., Mayer, I., Bidarra, R.: Balancing pedagogy, game and reality components within a unique serious game for training levee inspection. In: Hui, K., et al. (eds.) Technologies for E-Learning and Digital Entertainment. Proceedings of the Second International Conference, Edutainment 2007, pp. 128–139. Springer, Heidelberg (2007)

[6] Hsieh, A., 2004, Cremona Revisited: The Science of Violin Making, *Engineering & Science*, 4 pp. 29-33

[7] Hussain, T., Feurzeig, W., Cannon-Bowers, J., Coleman, S., Koenig, A., Lee, J., Menaker, E., Moffitt, K., Murphy, C., Pounds, K., Roberts, B., Seip, J., Souders, V. and Wainess, R. (2010) "Development of game-based training systems: Lessons learned in an inter-disciplinary field in the making," in J. Cannon-Bowers and C. Bowers (Eds.) *Serious Game Design and Development: Technologies for Training and Learning*, Hershey, PA: IGI Global, pp. 47-80.

[8] Jones, D. 2006. Decimation-in-time (DIT) Radix-2 FFT. Connexions, September 15, 2006. http://cnx.org/content/m12016/1.7/.

[9] *JamGuru.* [PC]. Kaliningrad, Russia: Ultimate Guitar, 2010. Available from: http://plus.ultimate-guitar.com/jamguru/

[10] Johnson, D. 2009. Discrete-Time Fourier Transform (DTFT). Connexions, July 6, 2009. http://cnx.org/content/m10247/2.31/.

[11] Johnson, D. 2010. Fast Fourier Transform (FFT). Connexions, June 20, 2010. http://cnx.org/content/m10250/2.19/.

[12] *La Corda d'Oro.* [Windows XP]. Japan: Koei, September 19, 2003.

[13] *Lily Pond.* [Internet]. c2003-2011 [modified 2011, July 4; cited 2011, February]. Available from: www.lilypond.org.

[14] *LittleBigStar.* [Windows XP, Vista]. Aarhus, Denmark: Ole. http://littlebigstar.net/main/

[15] *Loom.* [DOS floppy disk]. San Francisco, CA: Lucasfilm Games, 1990.

[16] Murillo, R., 2000. Suzuki Violin Pre-Twinkle Skills. Talent Education.org Suzuki Method Central, http://www.talenteducation.org/index.php?option=com_content&view=article&id=84&Itemid=91

[17] Percival ,G., Wang, Y., and Tzanetakis, G. Effective Use of Multimedia for Computer-Assisted Musical Instrument Tutoring. EMME '07: Proceedings of the international workshop on Educational Multimedia and Multimedia Education, pp. 67–76, Augsburg, Bavaria, Germany, 2007

[18] *Pokemon.* [Game Boy]. Tokyo, Japan: Satoshi Tajiri, 1996.

[19] Prensky, M. 2000. Digital Game-Based Learning. New York: McGraw Hill.

[20] *Rock Band.* [Playstation 3, Xbox 360]. Redwood City, CA: MTV Games and Electronic Arts, Nov. 20, 2007.

[21] *Rocksmith.* [Playstation 3, Xbox 360, Xbox Live]. Montreuil, France: Ubisoft, 2011. Available from: http://rocksmith.ubi.com/rocksmith/en-US/home/index.aspx

[22] Shu Hui Hsu and Ming-Shen Jian, "Digital and Modular Design Scheme based on education theory in RPG learning game," Advanced Communication Technology,

2009. ICACT 2009. 11th International Conference on, vol.03, no., pp.1733-1737, 15-18 Feb. 2009

[23] Starr, W. The Suzuki Violinist. Knoxville, TN: Kingston Ellis Press; 1976.

[24] Suzuki, S. Suzuki Violin School Vol. 1. Miami, FL: Summy-Birchard, Inc.; 2008.

[25] Tanenbaum, J. and Bizzocchi, J. (2009). *Rock Band*: a case study in the design of embodied interface experience. In Proceedings of the 2009 ACM SIGGRAPH Symposium on Video Games (New Orleans, Louisiana, August 04 - 06, 2009). S. N. Spencer, Ed. Sandbox '09. ACM, New York, NY, 127-134.

[26] *Tsukihime*. [PC-NScripter]. Japan: Type-Moon, December 2000.

[27] Zyda, M., "From visual simulation to virtual reality to games," *Computer*, vol.38, no.9, pp. 25- 32, Sept. 2005.