Grigutis, Laurynas

Investigation into Aspects of Feedback and Gamification in Computer-Assisted Musical Instrument Tutoring

Investigation into aspects of feedback and gamification in computer-assisted musical instrument tutoring

Laurynas Grigutis

A thesis submitted to the University of Huddersfield
in partial fulfilment of the requirements for
the degree of Masters of Science

The University of Huddersfield

September 2017

Abstract

This thesis investigates how different variations of presentational performance feedback influences the motivation of a learner in a computer-assisted musical instrument tutoring context. Further, the thesis overviews methods to increase students' engagement with the learning process.

Positive and negative content balance in performance feedback is proposed through conducting user testing and evaluation. User testing was carried out using an online browser-based guitar tuition app developed for the purpose of this thesis. A custom-built application allowed full control over varying the balance of positive and negative content within performance feedback for different testing groups. Last experiment was aimed to find aspects of gamification to promote and encourage users to better themselves and to keep learning.

A brief description of up-to-date technology used to design and develop the guitar tuition software product is described.

Despite the results being inconclusive they appear to indicate that presenting negative feedback is detrimental to the learners' motivation. Furthermore, gamifying the learning experience and introducing competitive nature of games into musical instrument tuition appear to increase learners' engagement, and prevent the learner drifting into boredom.

Table of contents

Abstract 1

List of figures 4

1. Introduction 7

  1.1. Research questions 7

  1.2. Project aims 8

  1.3. Research focus and methodology 8

  1.4. Structure of the thesis 9

2. Literature review 10

  2.1. A brief history of web-based teaching platforms 10

  2.2. Pedagogic educational principles 11

    2.2.1. Learning styles 11

    2.2.2. Formative and summative learner feedback 12

    2.2.3. Motivation and self-efficacy 13

  2.3. Constructive feedback 13

  2.4. Gamification of computer-assisted education 14

  2.5. Computer-assisted learning in musical context 15

  2.6. Commercial CAMIT applications 17

    2.6.1. eMedia Masters Of Rock Guitar 17

    2.6.2. Synthesia 18

    2.6.3. Rocksmith 19

    2.6.4. Piano Maestro (JoyTunes) 19

    2.6.5. Yousician 20

    2.6.6. Overview 21

  2.7. Sound analysis considerations 19

    2.7.1. Technology / programming environment 19

    2.7.2. Capturing the sound 20

    2.7.3. Audio analysis 20

2.8. Summary 22

3. Core guitar tuition software (guitMaster) architecture 24

3.1. Implementation 24

3.1.1. Audio analysis algorithm implementation 25

3.1.2. Information representation 29

3.1.3. Timing 31

3.1.4. Software usage flow 32

3.1.5. Usability evaluation 34

4. Experiment 1 – Positive vs Negative feedback 37

4.1. Implementation 37

4.2. Results – Data 41

4.3. Result analysis – discussion 45

4.4. Limitations – discussion 47

4.5. Conclusions 47

5. Experiment 2 – Difficulty level and scoring system 49

5.1. Context 49

5.2. Implementation 51

5.3. Results 54

5.4. Conclusions 55

6. Experiment 3 – Deployment of gamification to increase learner's engagement 57

6.1. Implementation 57

6.2. Results – Data 61

6.3. Result analysis – discussion 57

6.4. Conclusions 59

7. Conclusions and Further Work 61

References 63

List of figures

Figure 1. Guitar Hero controller. 15
Figure 2. A balance of anxiety and boredom 16
Figure 3. IMUTUS screen view example (Schoonderwaldt et.al, 2004). 17
Figure 4. eMedia Masters Of Rock Guitar screen view example. 18
Figure 5. Synthesia screen view example. 18
Figure 6. Rocksmith screen view example. 19
Figure 7. Piano Maestro screen view example. 20
Figure 8. Yousician screen view example. 21
Figure 9. Rocksmith screen view example. 19
Figure 10. Guitar Hero screen view example 19
Figure 11. FFT of the fundamental and harmonic frequencies of the upper e string represented in frequency domain 21
Figure 12. Simplified view of an audio signal in the time and frequency domain 22
Figure 13. Audio input - Pitch detection - Visualization flow chart of "guitMaster" guitar tuition software 25
Figure 14. ACF function. Searching for optimal offset. 26
Figure 15. guitMaster early user interface screen capture example. 30
Figure 16. guitar tuition app GUI. (L. Grigutis, BEng final year project) 30
Figure 17. The app at the final stage - playback visualization in transparent yellow 31
Figure 18. guitMaster guitar tuition software flow diagram (from learner's perspective) 33
Figure 19. guitMaster - Positive Feedback 38
Figure 20. guitMaster - Negative Feedback 38
Figure 21. guitMaster - Mixed Feedback (positive first) 39
Figure 22. guitMaster - Mixed Feedback (negative first) 40
Figure 23. Average top score. 41
Figure 24. Average usage time. 42
Figure 25. Average improvement. 42
Figure 26. Average user satisfaction. 43
Figure 27. Positive vs. negative word index. 43
Figure 28. Assessment scores over I-P-A rounds - Positive feedback. 44
Figure 29. Assessment scores over I-P-A rounds - Negative feedback. 44
Figure 30. Assessment scores over I-P-A rounds - Mixed feedback (positive-first). 44
Figure 31. Assessment scores over I-P-A rounds - Mixed feedback (negative-first). 45

Figure 32. Dips in the learner's result due to drastic difficulty increase ... 48
Figure 33. Difficulty scoring X & Y values and meaning ... 50
Figure 34. Difficulty level and scoring system screen capture. ... 51
Figure 35. Linear optimization 1 - Given scores (plain dataset). ... 52
Figure 36. Linear optimization 2 - Given scores + normalized note "X distance" values. ... 52
Figure 37. Linear optimization 3 - Given scores + normalized note "Y distance" values. ... 52
Figure 38. Linear optimization 4 - Given scores + normalized tempo values. ... 52
Figure 39. Linear optimization 5 - Given scores + normalized tempo, "X-Distance" and "Y-Distance" values passed through a linear equation. ... 53
Figure 40. Linear optimization 6 - Manual variable weight adjustment. ... 53
Figure 41. Linear optimization 7 - The brute force optimization algorithm. ... 54
Figure 42. Linear optimization 8 - Final curve. ... 55
Figure 43. guitMaster homepage screen capture. ... 58
Figure 44. guitMaster screen capture. Experiment - Variation 1 ... 59
Figure 45. guitMaster screen capture. Experiment - Variation 2 ... 59
Figure 46. guitMaster screen capture. Experiment - Variation 3 ... 60
Figure 47. guitMaster screen capture. Experiment - Feedback interface ... 61
Figure 48. Average top score. ... 51
Figure 49. Average usage time. ... 51
Figure 50. Average summative score. ... 51
Figure 51. Average practice-assessment rounds. ... 51
Figure 52. Average user satisfaction. ... 52
Figure 53. Average pos. vs. neg. words. ... 52
Figure 54. Participants' scores for all practice-assessment rounds – Control ... 53
Figure 55. Participants' scores for all practice-assessment rounds – Personal Competition ... 53
Figure 56. Participants' scores for all practice-assessment rounds – Group Competition ... 54
Figure 57. Word cloud of the free text feedback ("Comments & Feedback" see Figure 47) given by participants of group 1 ... 54
Figure 58. Word cloud of the free text feedback ("Comments & Feedback" see Figure 47) given by participants of group 2 ... 55
Figure 59. Word cloud of the free text feedback ("Comments & Feedback" see Figure 47) given by participants of group 3 ... 55
Figure 60. Word cloud of the words choice feedback ("Please select 5 words…" see Figure 47) given by participants of group 1 ... 56

Figure 61. Word cloud of the words choice feedback (“Please select 5 words…” see Figure 47) given by participants of group 2 56
Figure 62. Word cloud of the words choice feedback (“Please select 5 words…” see Figure 47) given by participants of group 3 57

1. Introduction

Computer systems enrich our day to day lives e.g. easing our long-distance communications, providing predictive text and automated grammar checking. E-learning systems have been seen in state educational establishments for well over a decade (R. Slack, 1999). The ease of access that the online learning platforms provide is advantageous to many students who may lack the time required by conventional forms of studies in universities, music schools etc…

Music education benefits from computer systems that provide help writing and editing music notation as well as offer music playback with visual note tracking. A popular example of such a system used in music education is Sibelius (Sibelius, 2017). This system offers editing of full notation scores as well as splitting the score into smaller pieces. Sibelius provides several different styles of music notation including real-time music notation from an audio track, with the use of a microphone or Musical Instrument Digital Interface (MIDI). Even though music teachers have Sibelius and similar tools at their disposal Sedlacek (2010) described the work of Crha et al. (2010) (only available in Check language) where Crha et al. provide survey findings which show that less than one fifth (19.5%) out of a total of 620 primary music education teachers who participated in the survey had any direct experience with any type of music software. The survey was conducted seven years ago which in computer science is a large amount of time. If the same survey was repeated in 2017 results may differ.

At the current state of increasing use and popularity of social media it may prove to be difficult for a tutor to engage a young student and hold their attention. Countless number of ever changing games, seemingly purposely design to waste one's time, is making its way through the web (e.g. Flappy Bird, Candy Crush etc.). Though games like these may seem wasteful to some they are extremely popular and there are aspects of it that could be taken and used to capture one's attention and sway students towards a more constructive use of today's technology. How can computers be used to help people learn to play a music instrument like a piano or a guitar? Is there a superior way of presenting feedback on their progress along the way? What is the best method to increase learner's engagement? These and similar questions are addressed in this thesis.

1.1. Research questions

This research seeks to address the following research questions for computer assisted instrument learning:

1. What impact does the type and presentation of feedback have on the performance and engagement of tutees?

2. What preference do tutees have for the type and presentation of feedback?
3. What impact does gamification have on the performance and engagement of tutees?
4. What preference do tutees have in terms of gamification?

1.2. Project aims

The main project aims are to:

- Develop a guitar tuition software product with adaptive, user-oriented feedback
- Review to what extent pedagogic principles are incorporated into interactive educational software products
- Investigate aspects of feedback presentation methods preferred by learners
- Investigate methods to increase tutees' engagement

1.3. Research focus and methodology

Capturing and keeping a young person's attention in the physical world is no easy task considering that today's youth spend a large portion of their time immersed in their "cyber" life (social media, online discussion groups, online games etc.). Learning to play a musical instrument however requires investment of time and effort. To master an instrument (e.g. violin) one must spend a large amount of time in repetitive practice sessions. These sessions are designed to teach the violinist how to move their finger muscles, how to correctly hold the bow and so on. Where one's experiences in the "cyber" reality is filled with exciting flashing and ever-changing content, music lessons may get dull and boring just because of their repetitive nature. This research focuses on how capturing and utilising "cyber" experiences can be brought into musical instrument practice to enhance engagement and interaction longevity.

To answer the research questions and complete the project aims a guitar tuition software product with performance feedback was developed. Common computer-generated performance feedback presentation methods were reviewed. The most common and promising methods to increase a tutee's engagement were reviewed and later chosen to be tested with the help of user testing. Testing was carried out in three stages:

1. <u>Feedback presentation tests</u>

Research (discussed more in-depth in the following chapter) indicated that an important aspect of any performance feedback is positive vs. negative content ratio. The first testing stage was aimed at answering what is the balance between positive and negative feedback. Positive / negative ratio was controlled using both visual (colour) as well as textual (written language) feedback presentation.

2. Performance scoring and melody difficulty test / survey

The app auto increments the melody difficulty when the tutee successfully completes a practice assessment round. After Experiment 1 was carried out it became apparent that a more gradual difficulty increase is needed as well as a more accurate melody scoring algorithm. Experiment 2 was used to find both the optimal melody difficulty assessment as well as a more precise melody scoring method.

3. Engagement tests

To capture tutee's attention and increase their engagement in the practice sessions a method called gamification was chosen. Research discussed further in Chapter Two indicates an increase in engagement when the competitive nature of games is brought into the learning process. Stage three of the tests aimed to answer whether personal competition (competing against your own achievements) and group competition (competing against the achievements of others) increase one's engagement in the overall learning process.

1.4. Structure of the thesis

Chapter Two overviews how musical instrument tuition fits into the rise of the computer age, presents and evaluates computer-assisted education literature and reviews early, as well as current examples of computer-assisted teaching platforms.

Chapter Three presents an in-depth technical specification of the interactive guitar tuition software product ("GuitMaster").

Chapter Four contains description, results and analysis of experiment 1.

Chapter Five contains description, results and analysis of experiment 2.

Chapter Six contains description, results and analysis of experiment 3.

Chapter Seven presents conclusions and outlines further work.

As part of the data gathering method, testing stages One and Three contained a free text feedback taken from each participant after the test was finished. Analysis of this feedback / comments raised several intriguing points. Different information presentation methods can have strong positive as well as strong negative implications in learner's motivation. The following chapter will look at how technology made its way into education.

2. Literature review

There are a variety of reasons why someone may choose not to pursue musical training via the conventional methods. Hiring a personal tutor or going to a music school may be too expensive. Schools and group sessions usually have strict inflexible time schedules. Having a musical instrument tuition software tool that would decrease the workload of a teacher and in some instances, work fully independently as a stand-alone application (i.e. not require a teacher at all) may help bring music education to more people.

The conception, development and inherited issues relating to computer-assisted education and music training are discussed in this chapter, as well as a detailed overview of currently available commercial musical instrument tuition software, feedback presentation methods and different gamification techniques.

2.1. A brief history of web-based teaching platforms

Since the introduction of the World Wide Web (WWW or W3) in 1990s many online courses have been made available to the public (e.g. ELM-PE (Weber & Möllenberg, 1994) followed by ELM-ART (Brusilovsky et al., 1995). Before the rollout of higher complexity web content in the beginning of the 21st century, these courses were mostly non-interactive and static. There was a need for developing an adaptive more advanced platform that could offer some type of intuitive user-oriented feedback. A recent popular example of such a course is a language learning application "Duolingo". Creating such a platform was not an easy task to realise on the primitive web of the 20th century, as it needed to adapt to a variety of situations – a large number of students differing in their expectations, abilities and needs (Brusilovsky, 1999, p.19-25).

As computer technology and web content evolved, the field of computer-assisted education got more attention as the technology required for it to be successful became more commercially viable. By the end of the 20th century virtual learning environments (VLEs) were being introduced as an alternative to conventional education. VLE is defined as an open, computer-based environment that allows interactions with other participants and provides access to a wide range of resources (Wilson 1996). By this definition VLEs can be seen as a step towards today's learning platforms because the introduction of VLEs also introduced the concept of a virtual learning environment where students interact with one another. This student interaction allows for the development of an environment where the learners are able to support as well as compete with each other. Both of these concepts (support and competition) are often used in today's learning applications. The competitive aspect is further explored and used when introducing a game approach to learning applications.

At the beginning of the 21st century, face-to-face teaching was believed to be the best mode of teaching. “Intelligent” computer-assisted leaning platforms were therefore developed with the aim to try to imitate face-to-face teaching (Graesser, VanLehn, Rose, Jordan, & Harter, 2001).

The word “intelligent” in computer-assisted teaching was used to describe a type of platform that would be able to interpret the student’s actions at every step of their learning process as an actual teacher would. The platform would have to be able to evaluate student’s answer (e.g. to a given mathematical task) as well as every intermediate step taken along the way. Upon getting the final answer, the software would be able to discuss any of the possible paths the student could have taken to arrive at the correct answer.

An “intelligent” educational platform would be able to present tips and suggestions for performance improvements and give feedback on the student’s intermediate steps as well as the final answer. This type of computer-based tutoring system was referred to as Intelligent Tutoring System (ITS) (VanLehn, 2011, p. 197–198).

By VanLehn’s description of “intelligence” in context of web-based tutoring systems presented in 2011, signs of “intelligent” teaching technologies on the web can be found as early as late 1990s. A technology known as “Interactive problem solving support” was introduced in late 1990s. It offered the user help on each step of problem solving. The help provided by the platform could vary from signalling about a wrong step, to giving hints or automatically completing a step the student is struggling with (Brusilovsky, 1999, p.19-25).

A large problem facing the web-based learning in 1990s and early 2000s was the World Wide Web itself. Content of the web was end-user oriented and not machine-readable (Devedzic, 2004). This meant that even though by early 2000s there were numerous online courses and learning material on the web, there was no way for the end user to gain access to it without precisely knowing where to look. This issue spanned a wide range of topics and web content and gained large attention within the community. The idea of a machine-readable Semantic Web was born. For example, when a person goes online for a tutoring session, a computer accessing a machine-readable web server could search the servers for information from separate courses. Information that suits the end-user would be presented, information that is irrelevant – disregarded (Devedzic, 2004).

2.2. Pedagogic educational principles

2.2.1. Learning styles

Different students have different preferred ways to acquire and retain information and gain new skills.

Some students find it easier learning purely from theory e.g. reading books or articles about a particular topic. Others feel more comfortable experimenting and going through the experience themselves in order to learn and understand a new theory (Truong, 2016).

The potential to enhance learning by gaining insights into ways students prefer to learn has been researched for decades and is still being researched now. In late 1980s Felder and Silverman (Felder & Silverman, 1988) wrote that learning styles can be differentiated between the way students process information: active experimentation or reflective observations. A 2014 paper (Thalmann, 2014) surveying e-learning system developers suggested that even now learning styles were upon the most useful frameworks for adaptive system development. The learning style models were among such other sources such as previous knowledge and student background.

The most prominent differentiation for learning styles that arise from the theory is learning by theory vs learning by experiment or practice. In the context of musical education and in particular musical instrument tutoring there is the need for both theoretical knowledge and practical skill. An adaptive guitar tuition software would therefore benefit from employing both beforementioned learning styles. Allowing the student to learn by theory and learn by practice as well as giving the tutee the control over the ratio of the two learning methods might give the software a leading edge as compared to a stricter tutoring application which forces the student to use one way of learning instead of the other.

2.2.2. Formative and summative learner feedback

Feedback is an essential aspect of the learning process. It can have both positive and negative implications depending on the way it is presented (Hattie & Timperley, 2007). Several factors may influence the effectiveness of feedback on a student’s performance. The use and prioritization of summative vs. formative feedback (or assessment in general) has been in open debate since the 1960s (Tyler et. al., 1967). Taras’ (2008) survey of 50 university lecturers outlined the consensus of the differences in formative and summative assessment:

- Summative is generally considered to be scored assessments of tasks given as a final exam at the end of a period of the studies.
- Formative assessment as portrayed by the survey participants is oriented towards the learning process itself. Formative assessment is considered as intermediate, often non-graded tasks. This is achieved by providing work which raises a discussion and allows the lecturer to formulate constructive feedback aimed at building the learner’s knowledge.

Both summative and formative forms of feedback could potentially be used in a CAMIT system. Summative feedback would present a score or a grade to the learner upon a learning stage completion.

Formative feedback could be deployed during the practice stage by giving the tutee immediate feedback on their performance.

2.2.3. Motivation and self-efficacy

In early 1990s when computerised systems started playing an important role in education modernization teachers faced a perception problem. Students were not very familiar with computers and those who have used a computer perceived it as a tool or a source of entertainment, but not as a learning resource (Hoska, 1993). Overcoming this obstacle was essential because as Hoska points out, the educational advancements that can be achieved by deploying computer-based instruction (CBI) and computer-generated feedback was noticed early on. In her work Hoska also identifies several psychological advantages that could be achieved with the help of user-oriented feedback and CBI. A feeling of low self-sufficiency the student may have is severely detrimental to the process of learning. Hoska identified that even though a single lesson may not overcome years of experiences that resulted in a low self-sufficiency, motivational and assuring instruction and feedback can temporarily increase the student's confidence in their abilities and possibly encourage the learner to keep trying.

Hattie & Timperley (2007) conducted several meta-analyses of previous research on feedback and came to a conclusion that backs Hoska's, stating the importance of motivation within the instruction – learning – feedback cycle. They also found that the studies showing highest effect sizes ("magnitude of the difference between groups" (Sullivan & Feinn, 2012)) involved feedback containing task and improvement focussed information (e.g. cues, goal-related feedback and corrective feedback). Praise, reward and punishment were found to be less effective. Hattie & Timperley argue that praise and rewards (badges, stickers, etc…) should not be considered as feedback but rather as contingencies to activities because they contain so little task related information.

2.3. Constructive feedback

Timing, phrasing and presentation of feedback are all important aspects that can determine how effective the information presented is. The ratio of positive versus negative content within performance feedback can influence the learner's perspective, motivation and the overall outcome of the learning process (Hoska, 1993; Hattie & Timperley, 2007; Rathel, et al., 2014). These papers conclude that as the ratio of positive content in student evaluation increased the student engagement improved as well. Rathel et al. looked at groups of students with mild behaviour disorders, intellectual and learning disabilities. How this translates to a wider range of students of varying age, gender, intellectual abilities and skill level is yet to be definitively answered.

Positive and negative feedback can be differentiated to the user through the use of graphical (visual) and textual changes within the interface. Graphical differentiation is often made using different colours. Colours create emotions based on the elements those colours relate to in life (e.g. light blue and green – positive – relates to clear sky, clean water and calming summer weather, greenish yellow – negative – relates to dirty water, bile, rotten food) (Taylor et al., 2013). Colour associations can also be ingrained in the learner's subconscious by the prominent environment where many things and actions are colour-coded (e.g. traffic light green – go, red – stop). The colour association provides a useful tool to graphically differentiate positive and negative feedback.

2.4. Gamification of computer-assisted education

An important side of the conventional way of learning is practice. Schatt (2011) describes practice as one of the most important and fundamental behaviours needed to achieve musical instrument skill proficiency. Although practice is an important aspect of the learning process it is also repetitive and often discouraging by its nature. Schatt conducted a short survey of young music students (aged 12 - 13) on their perception of practicing to achieve higher skill level with any chosen instrument. A common theme among those who answered the question was that repetitive practice may not be very difficult, but its unchanging nature often drives students to find a "better" thing to do. "Better" in the context of 13-year-old music students' answer is most likely "something more interesting".

To increase students' engagement in practice sessions during this computer, social media and information dominated age, the learning process itself needs to be changed or amended. The entire conventional learning process is unlikely to be fully replaced in its entirety, but some repetitive and mundane aspects of it can be improved upon by deploying the same tool that modernized multiple aspects of the 21st century life – the computer. The learning process may produce the same level of results but with increased student engagement and lower drop-off rates if a method known as gamification was introduced.

Gamification in most cases is described as the use of game design elements, game-thinking and mechanics in a non-game context (Deterding et al., 2011; Zichermann, & Cunningham, 2011; Werbach & Hunter, 2012). "The concept of gamification is receiving increasing amounts of attention, particularly for its potential to motivate students" (Gooch et al, 2016). Gamification has been widely accepted as an interdisciplinary method for increasing student participation in education. Gamifying the learning process promotes entertainment alongside conventional conservative way of learning and teaching (Politis et al., 2017).

An example of a platform that offers a game experience yet with very little correlation to actual instrument tutoring is Guitar Hero. Guitar Hero offers a game console (e.g. Play Station) interface with constantly flashing content which is reminiscent of arcade gaming. Notes that should be played are displayed on the screen on an animated fret board. The software is controlled not with an actual guitar but rather a guitar-shaped joystick with five buttons on the fret board and a lever to imitate plucking a string (see Figure 1).

Figure 1. Guitar Hero controller.

Deploying game-like concepts in educational applications has the potential to introduce a specific positive aspect of games into education. Games keep the players motivated with specific and clear goals in a rule-bound system which provides clear objectives and immediate feedback. This gives the player, and in the case of a music tuition application the tutee, a smooth flow and a feeling of optimal experience (de-Marcos et al, 2017).

de-Marcos et al (2017) conducted an experiment to test the social gamification of learning by designing a technology-enabled learning experience that specifically addresses the motivation of learners. An undergraduate course was used to assess the impact of the gamification approach on learning performance and on learners' attitude.

The results suggested a positive impact on performance in practical assignments, but the control group performed better on written tasks. The overall attitude of the students towards the gamified system was positive. The authors argue that the rewarding scheme must be carefully designed considering timing and duration of such rewards to adequately impact the learners' motivation and create a compelling learning environment.

2.5. Computer-assisted learning in musical context

Teaching a person to play a musical instrument is somewhat different from academic tutoring. In exact science, there is a clear and definite solution to a given problem. If a student is asked to solve a mathematical equation a computer program can easily determine if the student's solution is correct or not. When teaching to play a musical instrument there is no clear line between right and wrong "solutions". By estimating pitch, a program may determine that a specific note is out of tune (i.e. had an error in its fundamental frequency) by an unacceptable margin. However, this note could still seem correct to some listeners – especially those with little musical training. To overcome this music tutoring software must provide an output that clearly indicates by how much the note was out of tune (Percival et. al, 2007).

Furthermore, it is a difficult task for a novice musical student to make sense of the conventional symbolic music notation and map it to analogous actions (Schnotz & Kurschner, 2008). Therefore, many computer-assisted musical instrumental tutoring (CAMIT) projects attempt to use a variety of pictorial analogies of music notation to reduce the cognitive load for the user (Sweller et. al, 1998).

The stress level in the overall practice process must be balanced as well. A subtle balance between anxiety and boredom (e.g. increasing the difficulty too quickly or too slowly) is debated and the point of balance is referred to as the “Flow Channel” (Csikszentmihalyi, 2014). This balance is visually depicted in Figure 2. Csikszentmihalyi suggests that to achieve the optimal flow the user of any system must be challenged as not to make the system too dull and boring. But on the other hand, challenge the learner beyond their capabilities will lead to anxiety and high levels of stress.

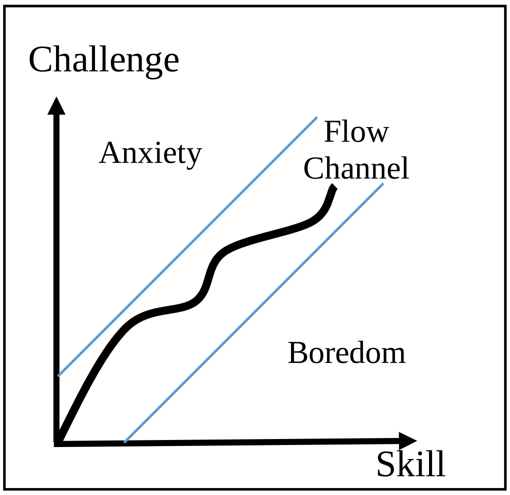

Figure 2. A balance of anxiety and boredom

Many attempts have been made to create CAMIT in the past thirty-five years (Percival et. al, 2007). Computers have been introduced into the music learning process as early as 1985 (Schoonderwaldt et.al, 2004). The early examples of CAMIT systems beyond presenting music notation would be Piano Tutor (Dannenberg et.al, 1990), Pianoforte (Smoliar et. al, 1995), IMUTUS (Schoonderwaldt et.al, 2004) (see Figure 3), Virtual Music Teacher (Mathieu, 2003).

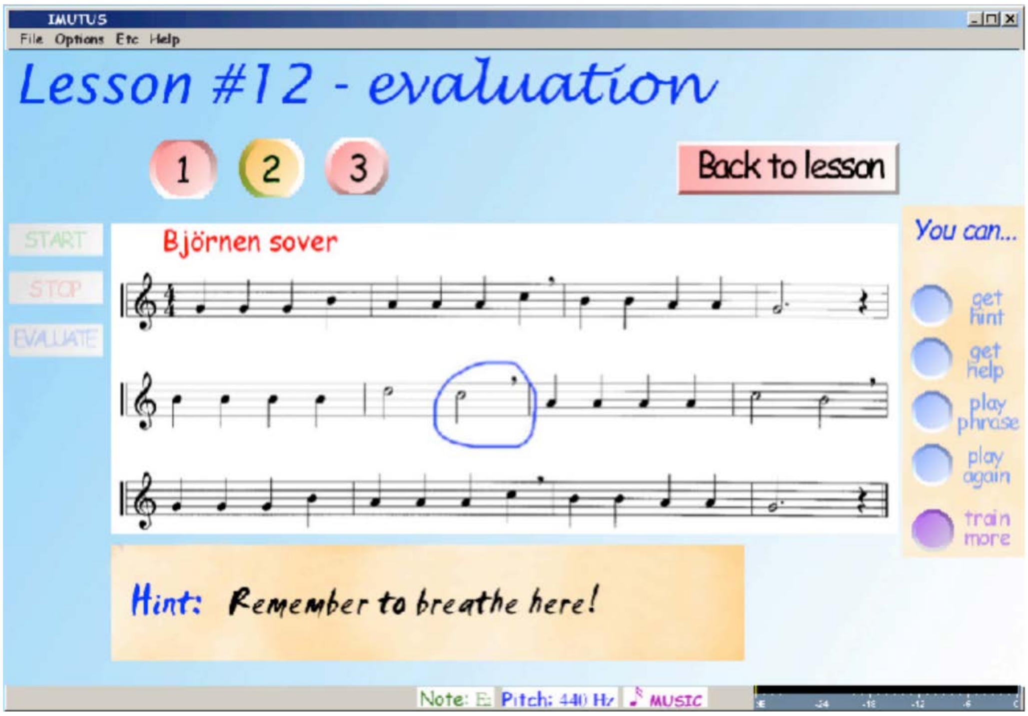

Figure 3. IMUTUS screen view example (Schoonderwaldt et.al, 2004).

These early CAMIT systems were designed to be stand-alone, autonomous tutors without the need for a teacher. Some form of visual and written feedback was provided by all of the products above. A common feature found in these systems is adaptive difficulty – the software decides if a student needs to practice an easier piece of music if the current piece is proving to be too difficult. The main setback of these systems as pointed out by Schoonderwaldt et.al. (2004) is the error in pitch detection accuracy.

Moore's law (Moore, 1975) predicted doubling of computer processing power every two years for the foreseeable future. The increase in computing power is utilised by some of the more recent examples of CAMIT systems - eMedia Masters Of Rock Guitar, Synthesia, Rochsmith and Yousician. These platforms fully employ the power of today's computers in creating user-oriented, advanced yet simple to understand graphical user interfaces (GUIs) and introducing gamification aspects to computer-aided musical education.

2.6. Commercial CAMIT applications

2.6.1. eMedia Masters Of Rock Guitar

eMedia Masters Of Rock Guitar (see Figure 4) offers a simple interface for notation playback. The interface includes an animated fret board with dots on the notes that are supposed to be played. Real time correct/incorrect note feedback is provided on the notation. Most notable limitations are:

- Has no scoring mechanism
- Has No progress monitoring.

- Provides no automated difficulty control.

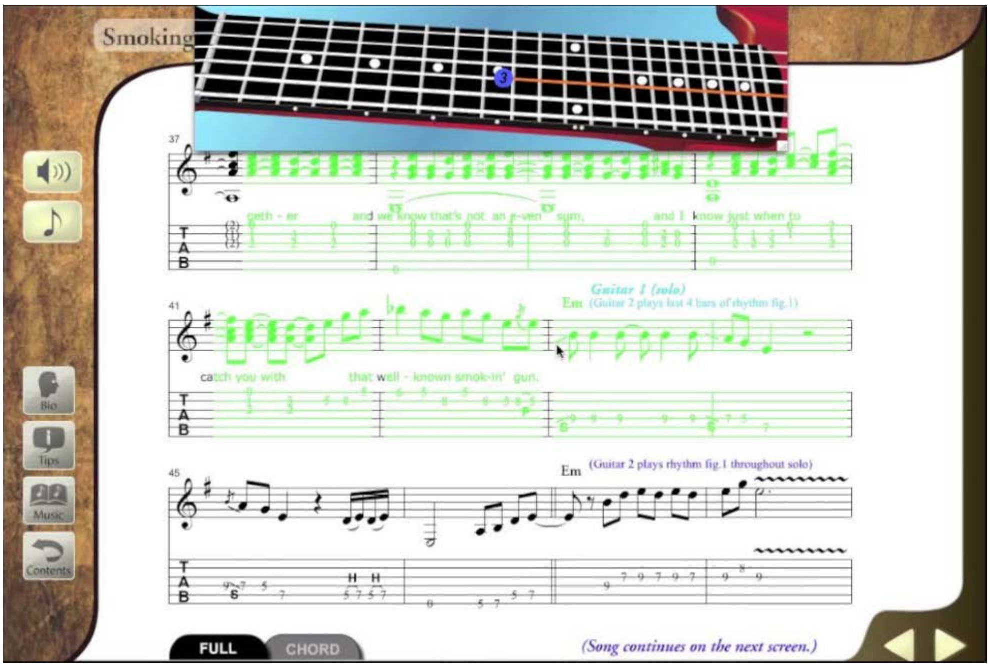

Figure 4. eMedia Masters Of Rock Guitar screen view example.

2.6.2. Synthesia

Synthesia (see Figure 5) – piano tuition software. Offers a visual interface indicating upcoming notes. Allows connecting a MIDI keyboard for a more realistic experience. Provides immediate note / timing feedback, as well as overall performance feedback. Provides progression monitoring and summative score. Has an aspect of competition (a leader board with other users' scores). Main limitations are:

- The sound is computer generated
- Has no microphone input to practice on an actual piano with real sound.

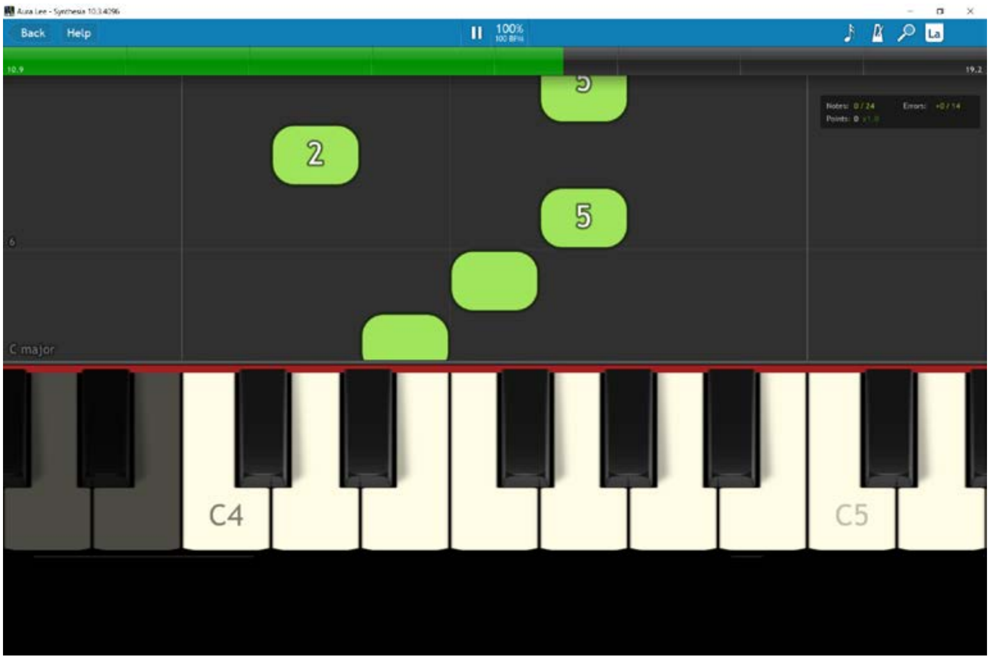

Figure 5. Synthesia screen view example.

2.6.3. Rocksmith

Rocksmith (see Figure 6) – similar interface to Guitar hero – has an arcade-like design with upcoming notescoming towards the user. The main advantage when comparing to Guitar Hero is that Rocksmith is controlled with an actual guitar. A noticeable positive feature is the automated difficulty control. As the student is starting to practice the melody, most of the notes are intentionally left out. As the learner progresses and hits more notes correctly more notes are added to the melody until all of the notes in the original version is displayed on the screen. Most notable limitations include:

- No visual incorrect note feedback information in real time. All of the feedback is presented in written form after the performance.
- The "Tab highway" reads from front to back rather then left to right (the conventional reading format for guitar tablature layout). This may present a difficulty when moving on from the game to reading music presented in guitar tablature format.
- The arcade-like design may be slightly distracting

Figure 6. Rocksmith screen view example.

2.6.4. Piano Maestro (JoyTunes)

Piano Maestro (see Figure 7) is a commercial application that teach the learner to play the piano. The app has both a limited free version as well as a subscription based full version. Piano Maestro gas a playful user interface (see Figure 7) which presents on-screen virtual piano keys where the immediate feedback is presented. When the playback starts there's a metronome to help keep the correct timing of notes. There's also the song notation flowing through the screen which is arguably the most prominent element on the entire interface.

The application first allows the learner to practice the melody slowly and without any backup music. The tutee first goes through the melody trying to hit the correct notes at the right time. If a note is missed the melody stops and the app displays a que which note should be played next. The practice mode allows the learner to familiarise themselves with the given melody and get instant feedback on their performance seeing on the virtual keyboard which key was hit in real time.

The following stage is the “Play” mode. The learner now hears music in the background to which he/she plays along. Just like in the previous stage there is immediate feedback on the virtual keyboard showing which key was hit in real time. At this stage the app does not stop if the tutee misses a note.

After each of the above described rounds the software gives an evaluation score in points. The app also splits the score into notes (number of correct notes / total number of notes) and timing (in percentage how accurate was the timing). This gives users a more detailed view of their own performance and allows them to make appropriate actions to rectify the aspect which had a worse score.

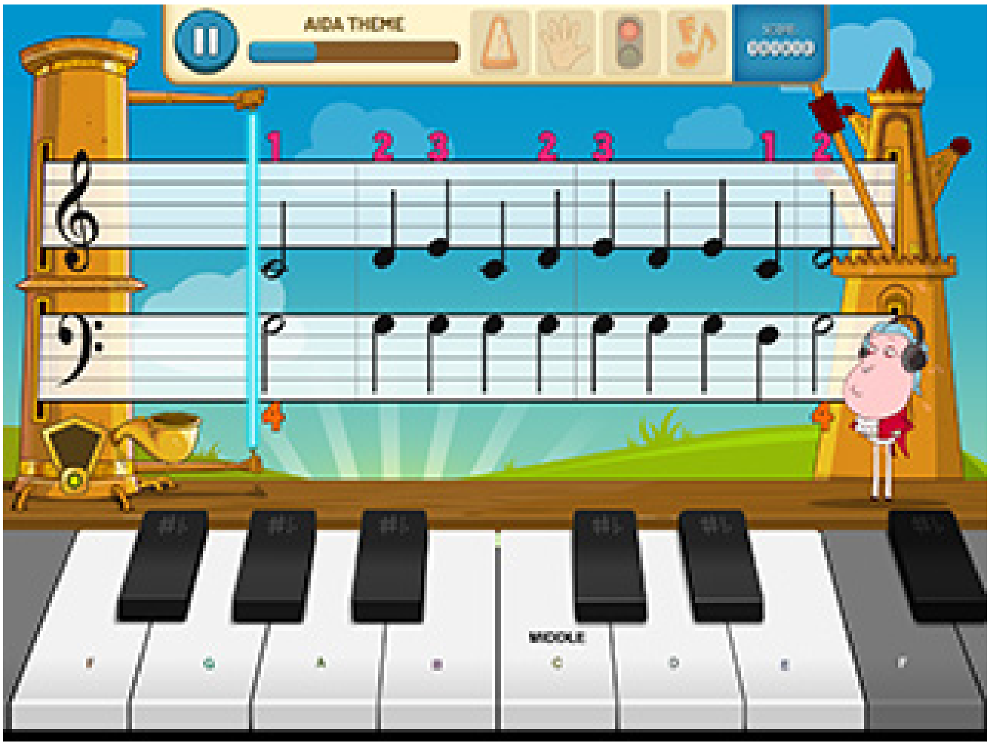

Figure 7. Piano Maestro screen view example.

2.6.5. Yousician

Yousician (see Figure 8) – Provides a left-to-right virtual fretboard with upcoming note and chord indication and more closely resembles an actual guitar tablature layout. Figure 8 provides a screen example of the Yousician user interface. Green “A” and blue “D” represents chords that should be played, the “bouncing” white ball provides visual representation of the rhythm whereas the smaller rounded rectangles with numbers on them represent a single note to be played on fret corresponding to the number on the rectangle. The app offers tutorial lessons to guide through the learning experience. Fairly simple and clear user interface. Provides immediate correct note feedback. When the learner plays an incorrect note the software pauses and waits for the user to get it right – then proceeds. The platform has cumulative score. It is available as a desktop application as well as a mobile app. The main limitations:

- Does not allow skipping easy steps for more advanced players (at least in the free version that the author was able to test)
- When an incorrect note is played the software does not indicate what that note was – it simply pauses and waits.

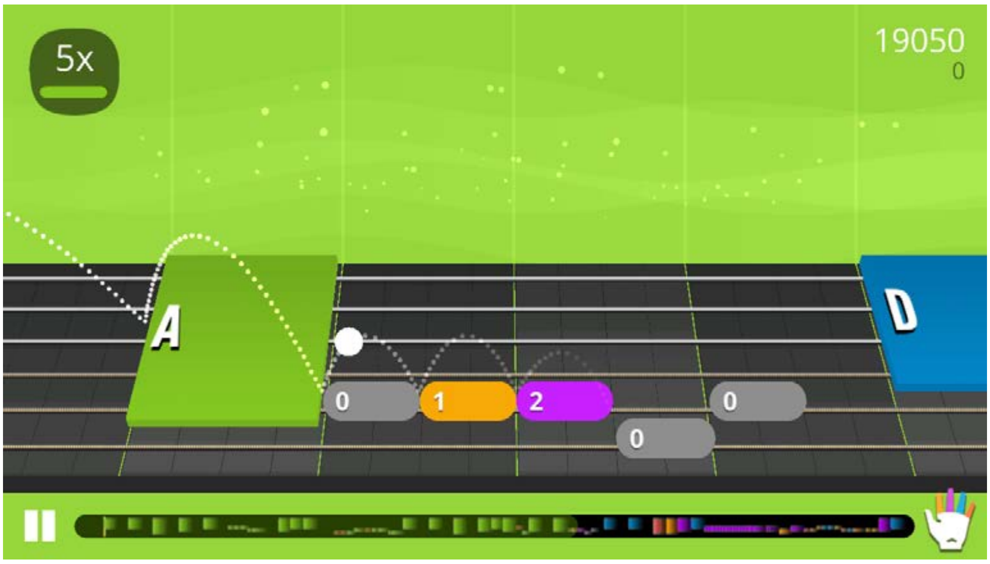

Figure 8. Yousician screen view example.

2.6.6. Overview

All currently available software products discussed in detail in sections above have positive aspects as well as some limitations. Yousician does seem like a good place to start for a beginner. There is a subtle balance between a game-like quality and a solely guitar tuition platform. Whereas eMedia Masters Of Rock Guitar does have an interactive user interface – the experience after using this product resembles that of a regular guitar practice session.

An example of a gamified learning platform is the Rocksmith guitar tuition software discussed in Section 2.6.3. This system deploys game-like qualities, not unlike those of the Guitar Hero game to capture one's attention and keep the learner interested throughout the learning process. Rocksmith however takes the game-like experience and introduces an actual musical instrument (guitar in this example) tutoring aspect by allowing the user to control the game via an actual guitar. There is a clear resemblance of a game-like experience in the Rocksmith CAMIT system (Figure 9) and Guitar Hero - an actual game (Figure 10).

Figure 9. Rocksmith screen view example.

Figure 10. Guitar Hero screen view example

Rocksmith provides enhanced entertainment, but the interface and the experience may be slightly too far away from actual guitar practice as the learner is constantly distracted from the musical aspects of the process by the use of extensive visuals.

A major limitation concerning all platforms discussed above, with the exception of Yousician and Piano Master, is that none have a free version available. The prices range from £20 to £60 as a one-off payment. Yousician and Piano Master are subscription-based products. Yearly subscription costs £100, though as mentioned, they do have free versions available with limitations in place.

Another limitation of most of the products is platform dependence. Most of them come in a CD format for a personal computer (PC) like eMedia, some are designated for gaming consoles (Rocksmith, Guitar Hero), Synthesia and Yousician can be downloaded from the web. Yousician has an advantage over the other products by having a mobile version of the application. Nonetheless both Yousician and Synthesia are PC / mobile applications requiring installation and lack the accessibility of a web-app. Therefore, a musical instrument tuition software built as a web application would have an advantage over the currently available ones by providing easy access to the resource.

2.7. Sound analysis considerations

2.7.1. Technology / programming environment

Currently, mobile platforms have a very measurable share of users in the overall scope of technology usage. Meeker (2015) in Kleiner Perkins Caufield Byers conference states that in 2014 73% of the planet's population (5.2 billion) used a mobile phone. 40% of these devices was a smart phone or a tablet. Meeker presents her findings which indicate that by the year 2014 there were 2.8 billion internet users (just under 40% of the planet's population). The capability to easily and seamlessly adapt to mobile platforms is therefore to be considered.

The software product at a later stage of development had to also be available for public access for more extensive experimentation. To increase the participant sample size and achieve significant data gathered by any experiments the software product had to be available for public access without any assistance from its developers.

The technology that would allow easy scalability, and seamless migration between mobile and desktop platforms as well as easy access to the general public is the browser-based programming language JavaScript (Flanagan, 2006).

2.7.2. Capturing the sound

An important aspect of the software is its standalone implementation and user independence (i.e. the ability for the users to operate the system on their own). The user must be able to set up and use the software with ease and without complication. To achieve such an implementation a sound is captured by the computers built-in microphone or an external recording device. Sound captured by a microphone is an analogue audio signal – a voltage changing in time. This time function must be converted to a sequence of numbers for the computer to be able to manipulate and analyse the signal. The conversion of a continuous-time function to sequence(s) of numbers is called analogue-to-digital conversion or ADC (Zolzer, 2008).

2.7.3. Audio analysis

Until 2010, there was little technological development on a web-based platform to allow audio playback and analysis and very poor standardisation. Technology for audio playback, manipulation and analysis was ad hoc, with different developers using different approaches which led to major browser compatibility issues. These issues were addressed by an introduction of a standardised native html <*audio*> element followed by introduction of W*eb audio API* (Web Audio API, 2017).

One of the major tasks the software must complete is audio analysis and pitch detection that determines the note being played at any given moment. W*eb audio API* is a high-level native JavaScript API for manipulating, processing, synthesizing and analysing audio in web applications (HTML5 Rocks, Mozilla Developer Network (MDN)). It allows the software to use native JavaScript for audio analysis without the need to deploy any third-party libraries or plugins.

Even with advanced *Web audio API* tools there is a challenge facing any frequency estimation and pitch detection – the same aspect that makes the music so pleasant to the ear – harmonics.

In acoustics, the basic vibration of a string is referred to as the fundamental frequency or the first harmonic. The second, third etc. harmonics refer to resonant frequencies above the fundamental frequency (Sengpiel, 2015; Fiedler, 2012).

Each string when plucked produces a specific fundamental frequency and a range of harmonic frequencies. Each harmonic starting from the first one, i.e. the fundamental frequency, appears with a lower amplitude and at a further distance from the fundamental in frequency domain. (Buckholz & Dantzig, 1976) (see Figure 11).

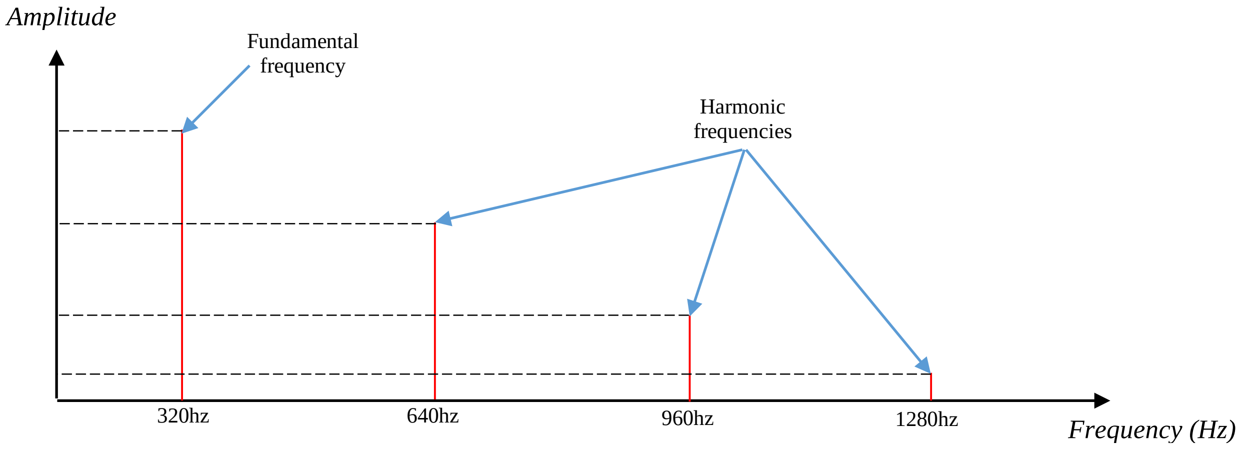

Figure 11. FFT of the fundamental and harmonic frequencies of the upper e string represented in frequency domain

The main issue of detecting the amplitude changes on relative frequencies is that a fundamental frequency of a thinner string often corresponds with a harmonic frequency of one of the thicker strings (e.g. the thicker E string has a fundamental frequency of about 80Hz and resonates at 320Hz – the fundamental frequency of the upper e string). This harmonic correspondence is what makes two strings with different fundamental frequencies sound good to the human ear.

It is this same correspondence that presents a challenge to pitch estimation algorithms as the algorithm must determine whether the frequency is a fundamental of a thinner string or a resonant harmonic frequency of a thicker one.

This problem is most prominent when Fast Fourier Transform (FFT) (see Figure 11) is used for pitch estimation. FFT is a fast, low complexity algorithm, developed by J.W. Cooley and John Tukey. FFT is used for computation of the Discrete Fourier transform (DFT) – an algorithm commonly used in engineering, computer science and signal processing (Oberst, 2007). The basic idea of Cooley-Turkey FFT algorithm is that it is a computational shortcut which is essential for minimizing the computational costs. The FFT takes calculations of the order of $Nlog_2N$, where direct DFT calculation is of the order of N^2 (Narasimhan & Veena, 2008).

Another algorithm called autocorrelation function (ACF) can be deployed alongside the FFT to achieve greater pitch detection accuracy (Kraft & Zolzer, 2015). Kraft & Zolzer describe pitch as a periodic wave in time domain. The location of the next peak presents an estimate period of the signal. For accuracy, more than one period (more than one repetitive peak in the signal) should be detected and some averaging applied to estimate the signal period (T) and produce the estimate frequency and pitch (f) with $f = \frac{1}{T}$.

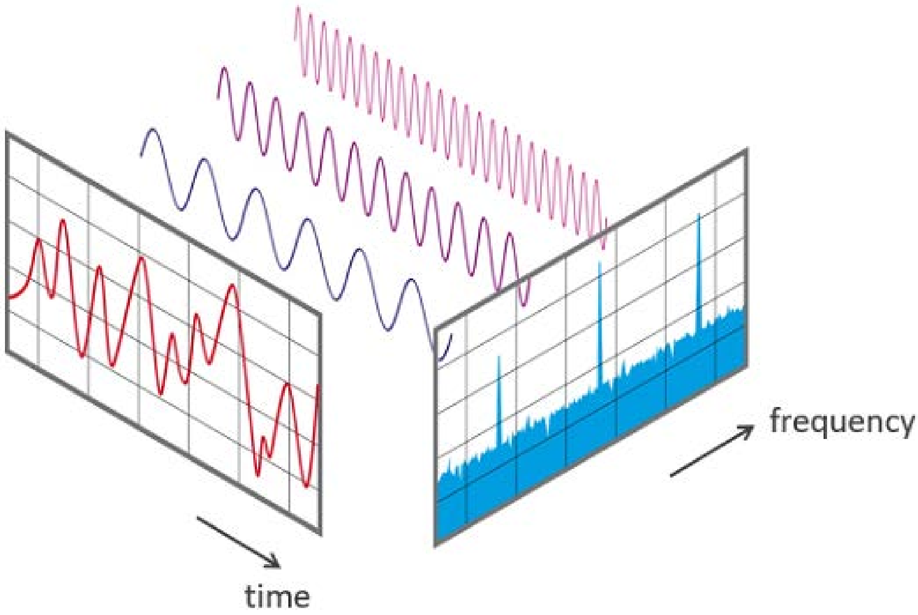

Figure 12. Simplified view of an audio signal in the time and frequency domain

Figure 12 shows a signal in time domain (left side represented in red) split into its frequency components (right side represented in blue). The autocorrelation function uses the original audio signal in time domain to detect periodicity of the signal and estimate the pitch from the period. Fast Fourier Transform algorithm is used to split this signal in time domain to its individual frequency components. The output can then be analysed in frequency domain.

2.8. Summary

The potential for computer systems to enhance education has been spotted early on yet, as is also the case in many other fields – education has been slow to adapt. Nowadays however computers are extensively used in classrooms for educational purposes across the world.

Early on computers were perceived as purely calculating machines so first to start using computers in education were technical fields e.g. mathematics and science. Music education has been slower to adapt yet still benefits from computerised systems that provide automated music notation and it's been gaining increasing popularity in the past decade.

Computer assisted musical instrument tutoring (CAMIT) systems can provide summative and formative performance feedback at varying steps along the learning path as well as of varying content. Research indicates that the use of colours associated with positive emotions and use positive content within feedback can increase student's performance and motivation in some situations.

With the uprising of social media platforms students have been daily subjected to increasing level of entertainment. The multimedia-oriented younger generation requires the implementation of gamification to increase one's engagement with the learning/practicing process. Gamification coupled with the use of user-oriented, task-focussed motivating feedback could potentially help produce a successful computer-assisted music instrument tutoring software product.

3. Core guitar tuition software (guitMaster) architecture

Section 2.7 discussed the most suitable programming language as well as the approach that will be taken for analysing the audio signals.

3.1. Implementation

Both ACF and FFT pitch detection approaches are utilized in the guitar tuition software "guitMaster". A high-level representation of this approach can be seen in Figure 13. The audio signal is sent to both the ACF and the FFT for pitch estimation. Determine string and fret from pitch. If the string and fret separately identified by both algorithms match then a "confident" match is found and the determined string and fret information is sent to the visualisation algorithm. Otherwise a new audio sample is taken.

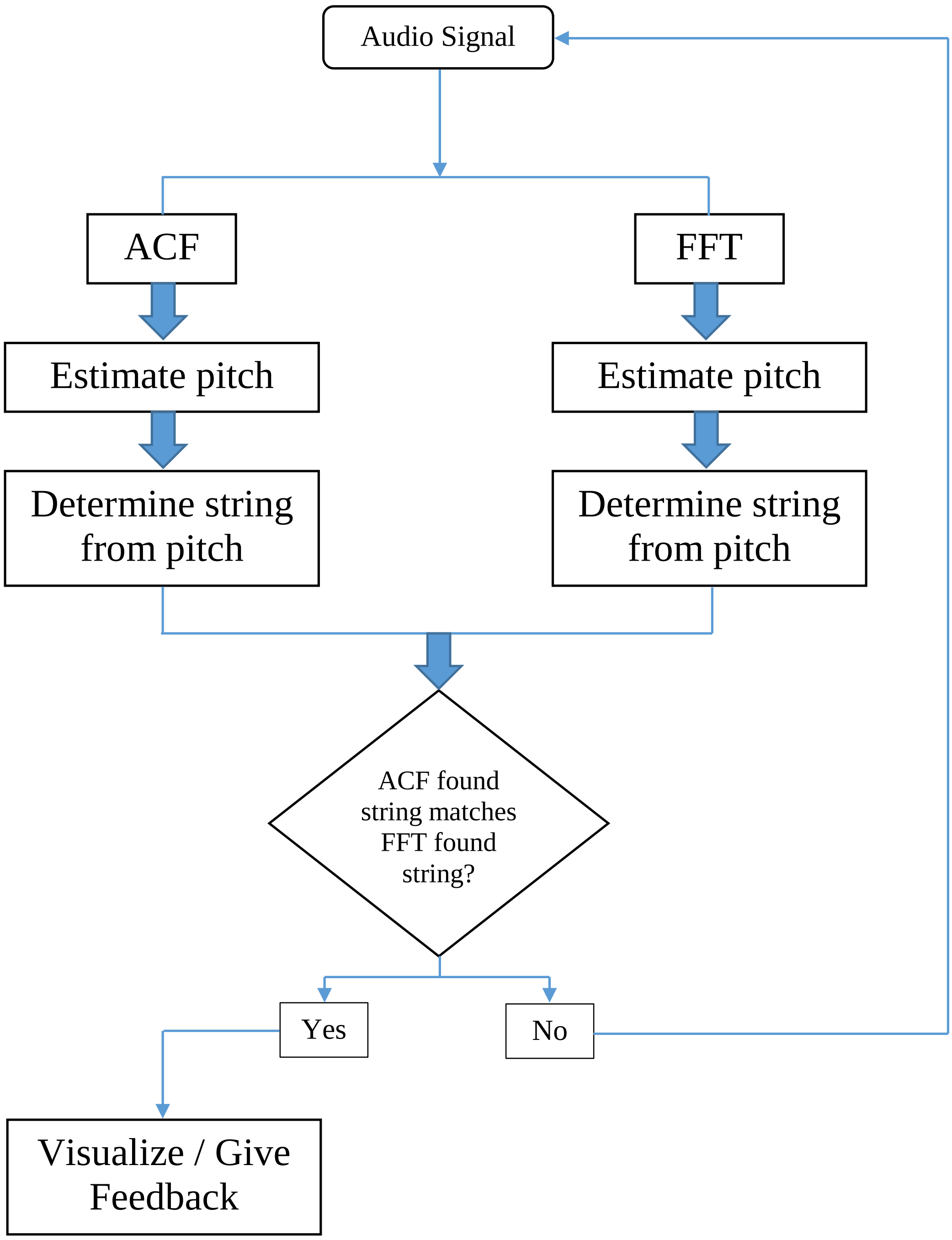

Figure 13. Audio input - Pitch detection - Visualization flow chart of "guitMaster" guitar tuition software

3.1.1. Audio analysis algorithm implementation

This subsection expands on the ACF and FFT implementation (the second step of the flow chart above (see Figure 13)). Both autocorrelation and Fast Fourier transform methods use the native JavaScript audio analysis engine *Web Audio API* to get the microphone input and fill the buffer with desired data samples.

ACF

To estimate the pitch of an audio signal using autocorrelation function the application must find the offset at which the signal repeats itself the closest – find the signals period (see Figure 14).

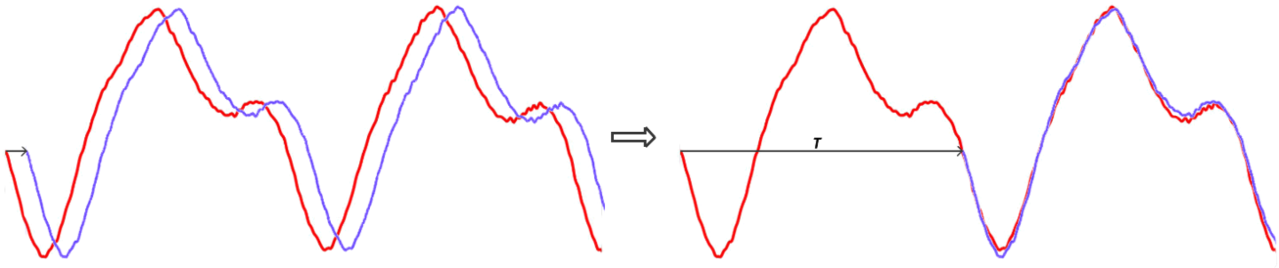

Figure 14. ACF function. Searching for optimal offset.

To detect the optimal offset the app uses a nested loop (for each element in the first half of the buffer iterates through the buffer again).

```
best_correlation = infinity
best_offset = 10

for offset from 10 to buffer / 2 {
  correlation_at_this_bin = 0

  for index from 0 to buffer / 2 {
    correlation_at_this_bin += absolute_value(bin_value[at index] - bin_value[at index +
offset])
  }

  if correlation_at_this_bin < best_correlation {
    best_correlation = correlation_at_this_bin
    best_offset = offset
  }
}
```

The buffer length used for the ACF calculations is 2048. We iterate only through the first half of the buffer because within the inner ***for each*** loop in the pseudo-code above we are checking the value ***at*** *index + offset*. This means that as we approach the end of the first half of the buffer on the outer ***for each*** loop the inner ***for each*** loop checks the values at the end of the entire buffer.

We start the *best_offset* and therefore the outer loop at index 10 (the 10^{th} element of the buffer) because otherwise we would be checking the buffer against itself (at index 0) and values close to it's original position. This would only be required if we needed to detect very high frequency signals (e.g. to detect a *12kHz* signal with sapling rate of *48k samples/second* the offset where the signal repeats itself in the buffer would be 4). To detect frequencies with such precision (at an offset of 3 or 4 steps ahead) the signal would have to be very clear and contain little to no background noise. The application is receiving the signal from the built-in microphone therefore in most cases there will be some ambient noise captured and this basic algorithm would detect false matches. The chosen starting offset value of 10 allows the app to detect a maximum frequency value of *4.8kHz* (with a sampling rate of *48k samples/second*). The software will only be required to detect frequencies up to *~400Hz* therefore starting at offset 10 is far enough from its original position to account for the ambient noise yet not so fat that the maximum detectable frequency becomes below the required value. Theoretically the offset could be initialised with 100 (which would allow detection of frequencies up to *480Hz* but all offset values above 10 produced similar accuracy results therefore the initial offset was kept at 10)

The closes match is found at the offset where the absolute *correlation_at_this_bin* value is the smallest. This is because the values at the current and the offset positions in the buffer are very similar (i.e. the signal is repeating itself) therefore taking the absolute value of *bin_value[**at** index] - bin_value[**at** index + offset]* gives us a value close to zero.

Having found the optimal offset the frequency of the audio signal can be calculated using the detected offset and the sampling rate (F_S). The sampling rate varies between different hardware and software setups but in most home use cases is 44100 or 48000. The app gets the sampling rate using the *Web Audio API*. The frequency of the signal can then be found using the following formula $f = \frac{F_S}{offset}$.

(https://developer.microsoft.com/en-us/microsoft-edge/testdrive/demos/webaudiotuner)

FFT

Using a Fast Fourier Transform an audio signal can be split into its individual frequency components as visualised in figure 12. The JavaScript *Web Audio API* natively provides a buffer of a desired size (the buffer size must be a power of two) containing amplitude values corresponding to separate frequency bins. Each of the frequency bins represent a distinct frequency range. The maximum measurable frequency also knows as the Nyquist frequency is half of the sampling rate (Condon & Ransom, 2016). In the simpler home setups, the sampling rate is usually 44100 so the Nyquist frequency and therefore the maximum measurable frequency would be around $22kHz$. As discussed in the ACF chapter above the maximum frequency the software needs to measure is *400Hz* therefore a limitation of *22kHz* will not present an issue in this case.

Two challenges that must be considered when using the FFT for pitch detection are measurement duration (the time it takes to fill up the buffer plus the time for performing the FFT and doing the pitch detection algorithm) and frequency resolution (the frequency range represented by each frequency bin in the buffer). These two challenges can be viewed as a single problem – the more precise the frequency measurement must be the larger sized buffer must be used which takes longer to fill. For example, if a relatively small 1024 buffer was chosen the measurement duration would be $\frac{1024}{44100Hz}$ or $23.22ms$ (plus the time for calculations) which is reasonably small as one fortieth of a second delay in timing would not be noticeable by a person using the app (playing a guitar). The frequency resolution is therefore $\frac{44100\ Hz}{1024}$ or $43.07Hz$. This means that the frequency difference between two consecutive bins in the buffer would differ by $43.07Hz$ whereas half of a tone on the thickest E string on a guitar (fret 0 or open string going to fret one) is approximately $4.5Hz$. The minimum buffer size needed to distinguish different notes on a guitar can therefore be estimated to be $\frac{44100\ Hz}{4.5Hz}$ or 9800. The closest power of two going up is 16384 which resolves to a frequency resolution of $\frac{44100\ Hz}{16384}$ or about $2.7Hz$.

Having chosen the FFT size of 16384 the sampling duration is $\frac{16384}{44100Hz}$ or approximately 370ms. This is over a third of a second which if not addressed will become noticeable by the user. The measurement duration is addressed by introducing a predefined delay in the timing algorithm as well as the visual feedback. Therefore, the algorithm which determines whether a note was played at the correct time takes into account the measurement duration and compensates for it. The optimal time delay for the scoring algorithm was found by trial and error starting at the before mentioned 370ms mark and landing on roughly 360ms. Differing sampling rate (44100 / 48000 when different hardware was used) led to the actual delay introduced to the software being slightly smaller than the delay estimated in the beginning.

```
max_aplitude = 0;
index_at_max_aplitude = -1;

for index from 0 to buffer {
    if bin_value[at index] > max_aplitude {
        max_aplitude = bin_value[at index];
        index_at_max_aplitude = index;
    }
}
frequency = sampling_rate * (index_at_max_aplitude / buffer)
```

The FFT buffer provided by *Web Audio API* is used to find the most prominent frequency (i.e. a peak picking approach). The algorithm finds the frequency with the greatest amplitude. This however does not necessarily result in the fundamental frequency. Due to the guitar being recorded with a simple microphone rather than a direct input there are several factors which may influence the most prominent frequency: angle of the microphone, strength of the pluck, background noise.

After having found the highest amplitude frequency the algorithm searches for resonant (or in some cases fundamental) frequencies in several other spots within the frequency spectrum dependant on the highest amplitude frequency it found. These several other spots are predefined in advance e.g. if the most prominent frequency is 320Hz this may be the fundamental of the thinnest *e* string or a “far” resonant frequency of the thickest “E” string (“E” strings’ fundamental frequency is about $82Hz$

therefore it might resonate at $320Hz$). The algorithm would check both $82Hz$ (which would indicate the thicker „E" string) and $640Hz$ (where the thinner „e" would have a resonant frequency and the thicker „E" would not (it would in theory but resonant frequencies this far away from the fundamental ($82Hz$) are usually very small and get lost in the background noise)). By performing these extra checks the algorithm is able to detect the frequency and therefore the note that is currently being played. Below is an example case written in pseudocode where the peak frequency detected by the FFT algorithm is *248Hz* (fundamental frequency of the open second string). The algorithm then checks the signal amplitude in the ranges of *80Hz - 85Hz and 160Hz - 168Hz* (indicating open string 6) and *120Hz-126Hz* (indicating string 5 at fret 2).

```
if frequency == 248 {
   if (max_amplitude between 80Hz and 85Hz) * 2 > amplitude at 248Hz or (max_amplitude
between 160Hz and 168Hz) * 2 > amplitude at 248Hz {
       string = 6;
       fret = 0;
   } else if (max_amplitude between 120Hz and 126Hz) * 2 > amplitude at 248Hz {
       string = 5;
       fret = 2;
   } else {
       string = 2;
       fret = 0;
   }
}
```

3.1.2. Information representation

(I will add the design ethos, the visual choices and flow of development in this subsection)

Chapter Two outlined the importance of both summative and formative assessment. Assistance and immediate feedback can help the learner remain task-focused and retain motivation (Hoska, 1993). Therefore, the software was developed to present immediate feedback to the learner visualizing the guitar notes currently playing by projecting the notes onto a guitar representation in the browser window (see Figure 15). The most recent technologies were used, and backwards browser compatibility disregarded.

A recent improvement to client-side visualization in JavaScript is a *requestAnimationFrame*() method (Understanding JavaScript's requestAnimationFrame(), 2016; window.requestAnimationFrame(), 2017). Using a predefined interval (e.g. 60 times per second) by using the *setInterval*() method was the most common way of animation before *requestAnimationFrame*() was introduced. The new method repeatedly calls the next animation frame whenever the browser is ready to perform a redraw. In most common browsers, this occurs about 60 frames per second (fps). Another visualization method that the software uses is the html canvas element. The canvas is where the visualizations at the rate of 60fps (browser redraw rate determined by the *requestAnimationFrame*() method) will be drawn.

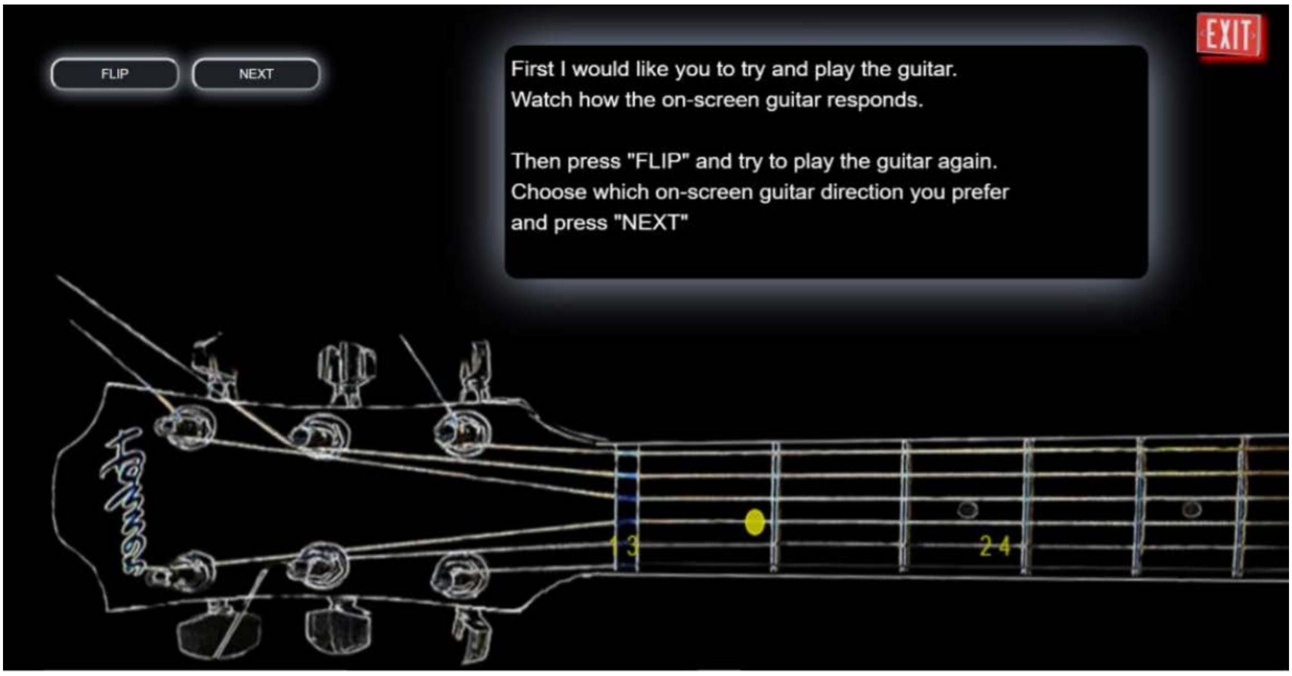

Figure 15. guitMaster early user interface screen capture example.

The design of the system was picked up from where the author left off in a Bachelor of Engineering final year project (Grigutis, 2015). Below is a figure of the system's user interface from the author's final year project (see Figure 16).

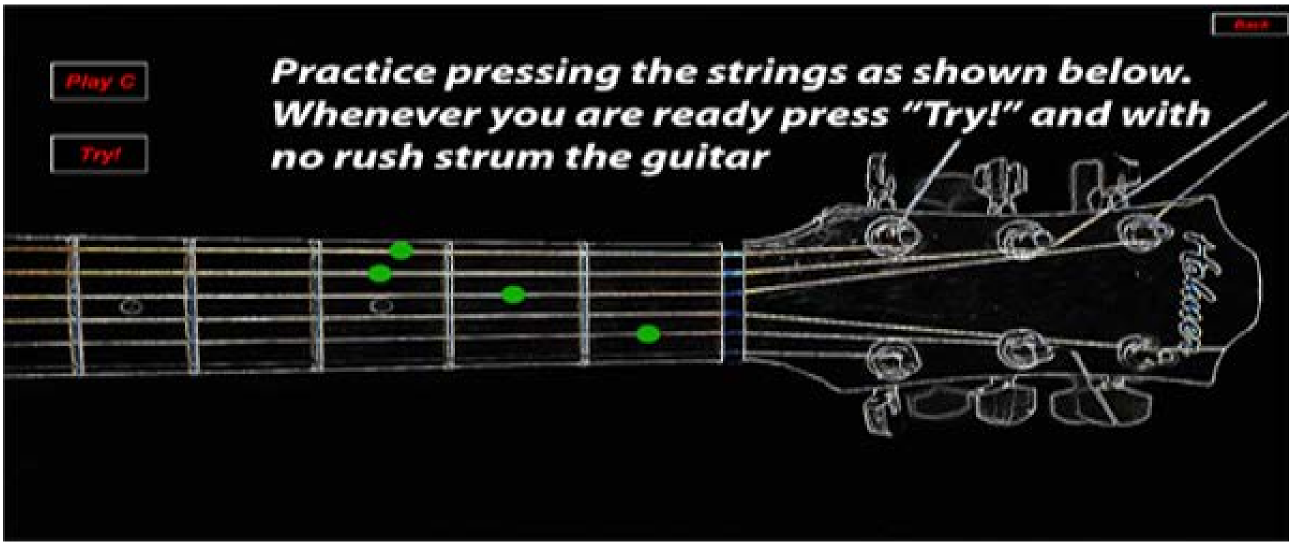

Figure 16. guitar tuition app GUI. (L. Grigutis, BEng final year project)

From the GUI seen in the figure above, a trial and error approach was taken for further app development. Every two weeks new functionality, visualization and audio playback (where the user hears playback of the melody he/she is practicing) approaches would be implemented to the system and trailed the author as well as by two other users – an advanced guitar player and a beginner. Features that would enhance the learning flow for the beginner at the same time not making it too dull for the pro would be kept and others disregarded.

Chapter Two discussed that humans have a tendency to associate colours with emotions. Colours that bring up positive emotions are perceived as positive (e.g. green representing spring or summer, warm and calm weather) and colour that are associated with negative emotions tend to be perceived as negative (e.g. red – traffic lights signalling to stop or even blood and pain). These two colours were chosen to represent correct and incorrect notes when giving the immediate feedback to the learner discussed in the first paragraph of this chapter. Yellow was chosen to visualise the playback (the app playing the melody the tutee is currently learning). The colour yellow was chosen as it also brings more positive associations (e.g. the Sun) rather than negative ones.

For the same reason yellow was chosen for the note sequence number as well. The note sequence number was introduced to help the learner follow the melody. It represents the point within the melody of each note. Number 3 for example states that this note is the third note within the melody.

The yellow dots at a later development stage were made slightly transparent (see Figure 17) for two reasons:

1. Firstly, to not draw attention from the main aspect of the system at the learning stage – immediate feedback.
2. Secondly, to not fully block the note sequence number

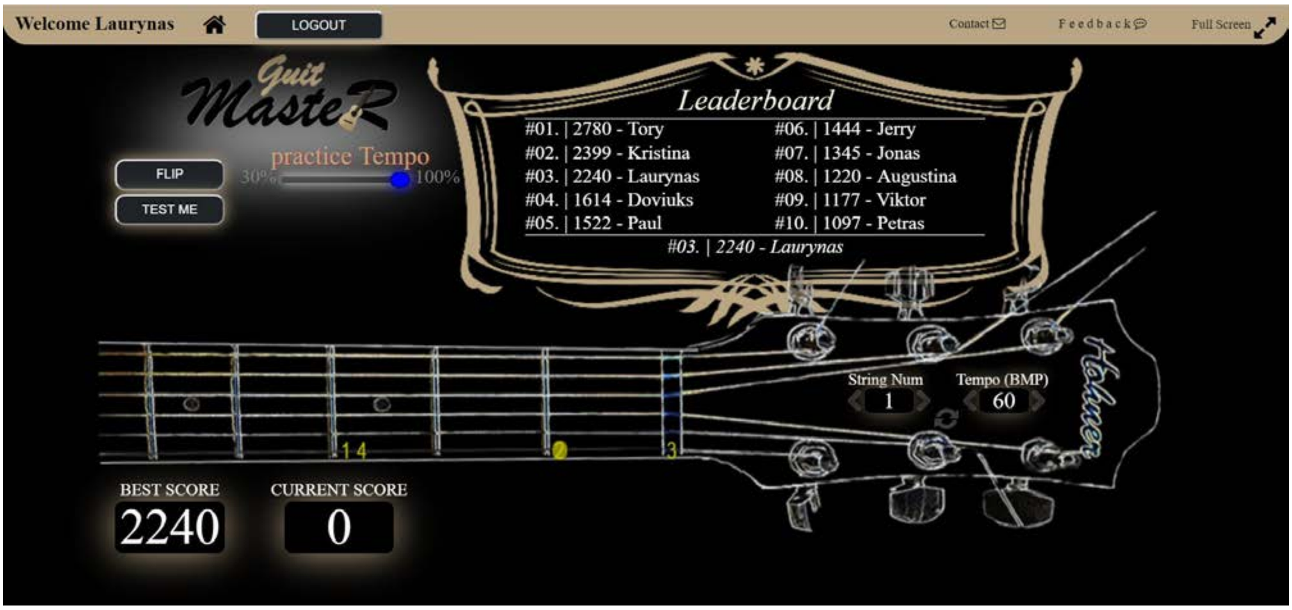

Figure 17. The app at the final stage - playback visualization in transparent yellow

3.1.3. Timing

Accurate timing is essential to a successful working software prototype. Though JavaScript has many benefits outlined in Section 3.1 it also has a large drawback, namely, performance. Whereas Java or C# running on the computer are quite efficient, the “loose type” nature (no requirement for strict variable type declarations) of JavaScript coupled with the fact the code is executed in a browser window and not directly by the computer processing utilities makes JavaScript comparably slower. The main thread of the browser window will in most instances be busy performing the FFT and ACF calculations for pitch estimation as well as animating and visualizing the immediate performance feedback onto the canvas at roughly 60 frames per second.

To increase performance and timing accuracy for the musical metronome, another recent web technology called *Web worker* (Using Web Workers, 2017) was used to utilise browser's capability to perform tasks in the background as well as on the main thread at the same time. This way even if some expensive computational task may overload the main thread for several milliseconds, the web worker will keep track of absolute time as the main thread does not affect the backend thread's performance.

3.1.4. Software usage flow

From the perspective of the learner the usage of the software is split into four distinct steps:

1. The learner is presented with a random eight note melody. The tutee is asked to familiarize with the given melody and to proceed to the next stage
2. On the second stage the learner is asked to play along to the tune. The playback starts (the user is wearing headphone to hear the playback) and the app gives immediate feedback by indicating (with a coloured circle) the note the user is currently playing on the on-screen guitar. If the learner hits the correct note at the correct time the feedback circle turns green otherwise the circle is red.
3. The learner proceeds to the third step when he/she feels confident enough to take the performance test. To complete the test the user must twice play the melody they practiced in previous step.
4. Step four is where the learner is presented with the assessment score and advised of the increase or decrease of the difficulty. If the user performed intermittently he/she is advised to try a new random 8 note melody at the same difficulty level.

The above four steps and the difficulty increase/decrease is is show in the flow diagram below.

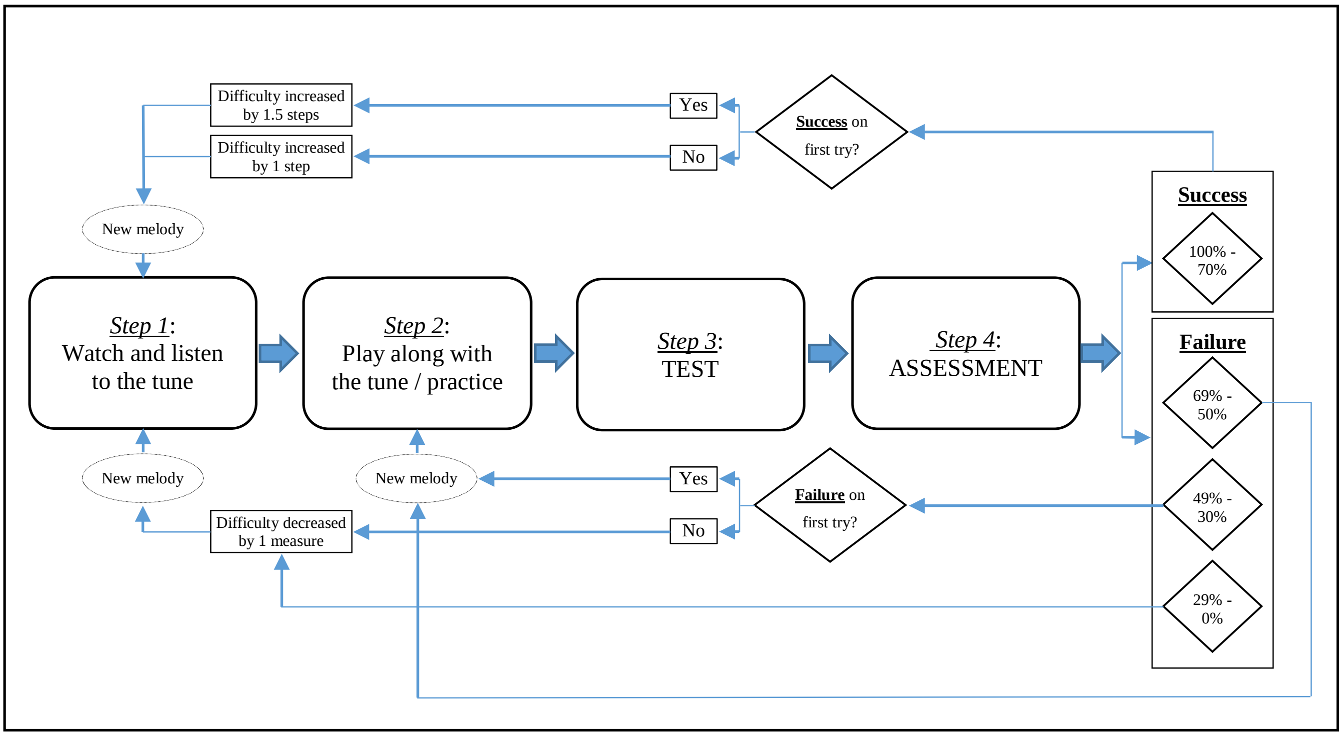

Figure 18. guitMaster guitar tuition software flow diagram (from learner's perspective)

Main aspects of the software:

- Difficulty – the number of strings used to construct a melody and the tempo in beats per minute (BMP).
- Melody – a random sequence of 4 notes constrained by the difficulty level repeated twice in a row (a total of 8 notes).
- Timing – a time keeping mechanism to ensure tempo is constant and the audio is synced with the visuals.

These are also the dimensions of assessment / complexity used to calculate the score for any given random four note melody. Values for these variables were initialized with a melody consisting of one string at 60BPM. The software however was set up to react to user's performance level and adjust the level of difficulty accordingly.

A concept of a practice-assessment iteration counter was introduced to determine if any given melody at a specific difficulty level is being performed for the first time. When a tutee is assessed the melody and its difficulty is adjusted; increasing or decreasing the number of strings the melody can consist of and the playback tempo. Table 1 presents the difficulty control in accordance with which level the student is at as well as the student's final assessment score (0 – 100%). The software usage flow incorporating the automated difficulty control is presented in a flow diagram (see Figure 18) at the top of this chapter.

<table>
<tr><th colspan="2" rowspan="3">Assessment score</th><th rowspan="3">Difficulty step to next stage</th><th colspan="4">Difficulty change</th></tr>
<tr><th colspan="2">First try</th><th colspan="2">Subsequent tries</th></tr>
<tr><th>Tempo</th><th>No. of strings</th><th>Tempo</th><th>No. of strings</th></tr>
<tr><td colspan="2">0%-29%</td><td>decreased</td><td>-10BPM</td><td>-1</td><td>-10BPM</td><td>-1</td></tr>
<tr><td>First try</td><td>Subsequent tries</td><td rowspan="2">unchanged</td><td colspan="4" rowspan="2">unchanged</td></tr>
<tr><td>30%-69%</td><td>50%-69%</td></tr>
<tr><td colspan="2">70%-100%</td><td>increased</td><td>+20BPM</td><td>+1</td><td>+10BPM</td><td>+1</td></tr>
</table>

Table 1. guitMaster difficulty change characteristics

A melody is the context of this application refers to the sequence of notes that is to be practiced and performed for assessment. A predefined set of melodies / segments extracted from well-known songs was considered at first. After considering the human factor in choice of melody set, as well as taking into account that some tutees may perform better or be irritated by a melody they know, a conclusion was drawn that in order to eliminate external factors, the melody generation would be randomised.

The melody consists of four random notes determined by the difficulty level the tutee is at the moment of generation. The melody is played in a continuous loop for the practice part of the round. Then the same melody is played twice for assessment. After each round, a new set of four random notes are generated and a new melody is presented to the user. The random note generation is always constrained by the number of strings available at the current difficulty level.

The concept of the iteration counter provides the software with a deeper knowledge of the user's learning journey. It allows for a more adaptive user-oriented approach to automated difficulty level control (see Figure 18).

By adopting the best currently available JavaScript web technologies such as *web workers* for timing, *requestAnimationFrame* and *html canvas element* for visualizations and *Web audio API* for audio analysis a comprehensive, accessible guitar tuition software with reasonably accurate pitch detection algorithm was developed. The software was then deployed in the following testing stages.

3.1.5. Usability evaluation

The main aim of the project is to develop a guitar tuition software product with an adaptive, user-oriented feedback system that deploys user-centred design in combination with usability evaluation. Usability evaluation involves measuring effectiveness, efficiency and user satisfaction (Grammons & Ippoliti, 2016; Gossen et al, 2017). Several successfully deployed usability evaluation techniques (Dewey & Wakefield 2016; Dewey & Wakefield 2017) were adopted where participants were asked to score their experience at a range of 1 to 10. Participants were then asked to choose 5 words that best describe their experience from a list of 30 words (15 positive and 15 negative) (Meyer, 2016). All 30 words were presented in a randomised order to eliminate any preference towards words at the beginning or the end of the list. The words presented to the participants were:

Negative:

- Boring
- Annoying
- Intimidating
- Old
- Dull
- Confusing
- Frustrating
- Hard to Use
- Ineffective
- Overwhelming
- Stressful
- Complex
- Difficult
- Impersonal
- Poor quality

Positive:

- Fun
- Engaging
- Effective
- Easy to use
- Sophisticated
- Comfortable
- Clear
- Appealing
- Inspiring
- Professional
- Innovative
- Intuitive
- Exciting
- Helpful
- Creative

An equal number of positive and negative keywords were provided to the learner to assure an equal distribution of positive and negative choices and empower the learner to use negative words to describe their experiences. The participants may feel that a positive response is more beneficial to the research. Providing the users with an equal distribution of both positive and negative choices may assure some of the participants that a negative evaluation of the software is acceptable as much as a positive one.

Furthermore, a free text input was given to each participant for a free text format evaluative test subject feedback.

Besides taking participant feedback via the above-mentioned methods, additional data was gathered by the software:

1. Overall time spent using the software,
2. Time spent on each of the practicing rounds/cycles (practicing a melody – performing – receiving feedback – receiving a new melody – practicing etc.),
3. Total number of practice rounds completed,
4. Achievement for each round in a form of a score given.

The score was determined by calculating how many strings the melody consisted of, the current tempo and how many notes the tutee played correctly (assessing the timing and pitch accuracy).

$$score = number\ of\ strings * tempo * \ assessment\ score(correct\ note\&timing)$$

In the equation above: number of stings (1 – 6); tempo (30 – 160); assessment score was presented to the user as a percentage (0 – 100%) but in the equation above was used as a fraction (i.e. 65% would become 0.65) solely to constrain the score and keep it as a smaller number.

4. Experiment 1 – Positive vs Negative feedback

An experiment was carried out to accurately weigh the impact and the outcome of differently formulated and presented feedback in computer-based musical instrument tutoring.

4.1. Implementation

A major implementation limitation of the software at this stage was its hardware dependency. No optimization, cross-browser and cross-device testing or debugging was carried out so the application was limited to a single computer using specific hardware. Because of these restrictions, the test sample was limited to 25 subjects.

The first variable chosen was the graphical representation of information by varying the dominant colour of the visual feedback presentation. As outlined in Section 2.2, colours are associated with emotions and the use of certain colours can potentially influence the effectiveness of feedback. Red for example, is in general associated with pain and blood. In the Lithuanian anthem colours of the national flag are mentioned. The anthem has a line reading “Colour red is for the blood spilled”. For a negative signal of failure, the colour red was chosen for the experiments.

As stated by Palmer et al. (2010) and backed by Taylor et al. (2013) individuals often associate light-blue and green with clear sky and calm comforting summer weather and positive emotions. Green is also commonly found to represent “go” or “clear” signalling colour so green was chosen to signal success. Taylor et al. also state that darker shades are perceived more negative whereas lighter shades are associated with more positive emotions. Therefore, dark blue – a combination of a more positive colour and a more negative shade was chosen to represent a neutral colour.

The second variable of the feedback given to the learner is textual content. The software can calculate correct/incorrect notes and correct/incorrect note onsets (timing) as well as the overall score. The phrasing therefore could be articulated in several different ways – all positive, all negative or a mixture of both. The following four feedback presentation / visualization variations were constructed:

1. All positive – feedback fixates on the correct aspects of learner’s performance (e.g. 50% of the notes were correct and 25% of the notes were played at the right time. Overall 37.5% CORRECT!). The dominant colour was green (see Figure 19).

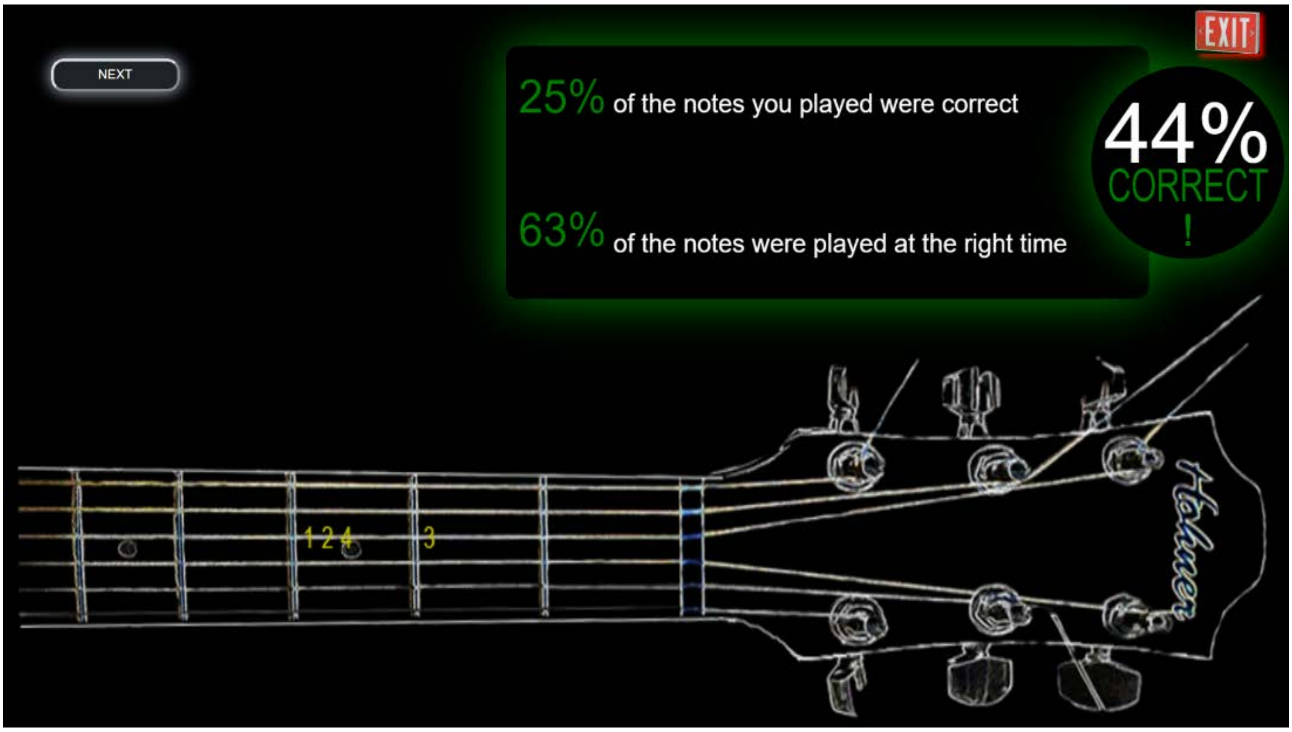

Figure 19. guitMaster - Positive Feedback

2. All negative – feedback fixated on the negative aspects of the learner's performance (e.g. 50% of the notes were incorrect and 75% of the notes were played at the wrong time. Overall 62.5% incorrect). The dominant colour for highlighting the information was red (Figure 20).

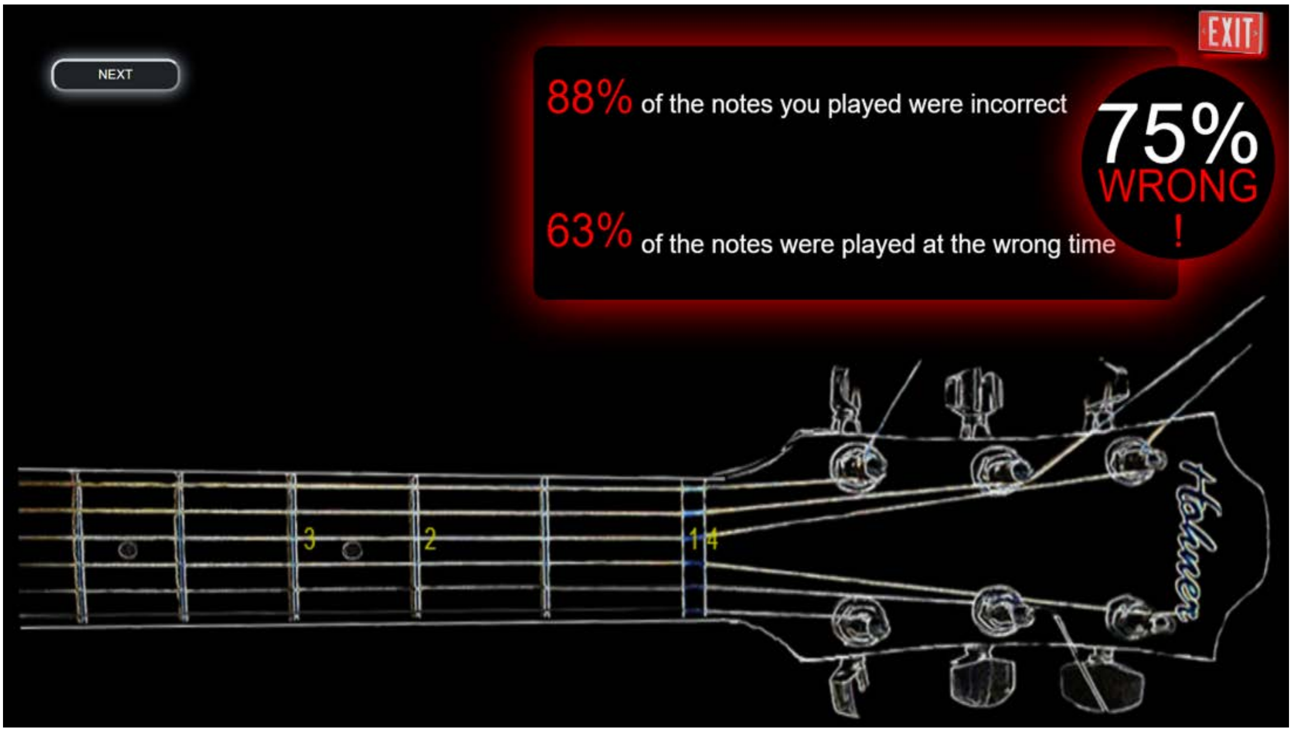

Figure 20. guitMaster - Negative Feedback

3. Mixed feedback (positive aspect first) – first part of the feedback highlighted the positive aspects of the performance, second part – the negative (e.g. 50% of the notes were correct but

75% of the notes were played at the wrong time. Overall 37.5%). The information was coloured according to the overall percentage: 0%-50% in red, 50%-70% in dark blue, 70%-100% in green. The correct and incorrect percentages were coloured in green and red accordingly (see Figure 21).

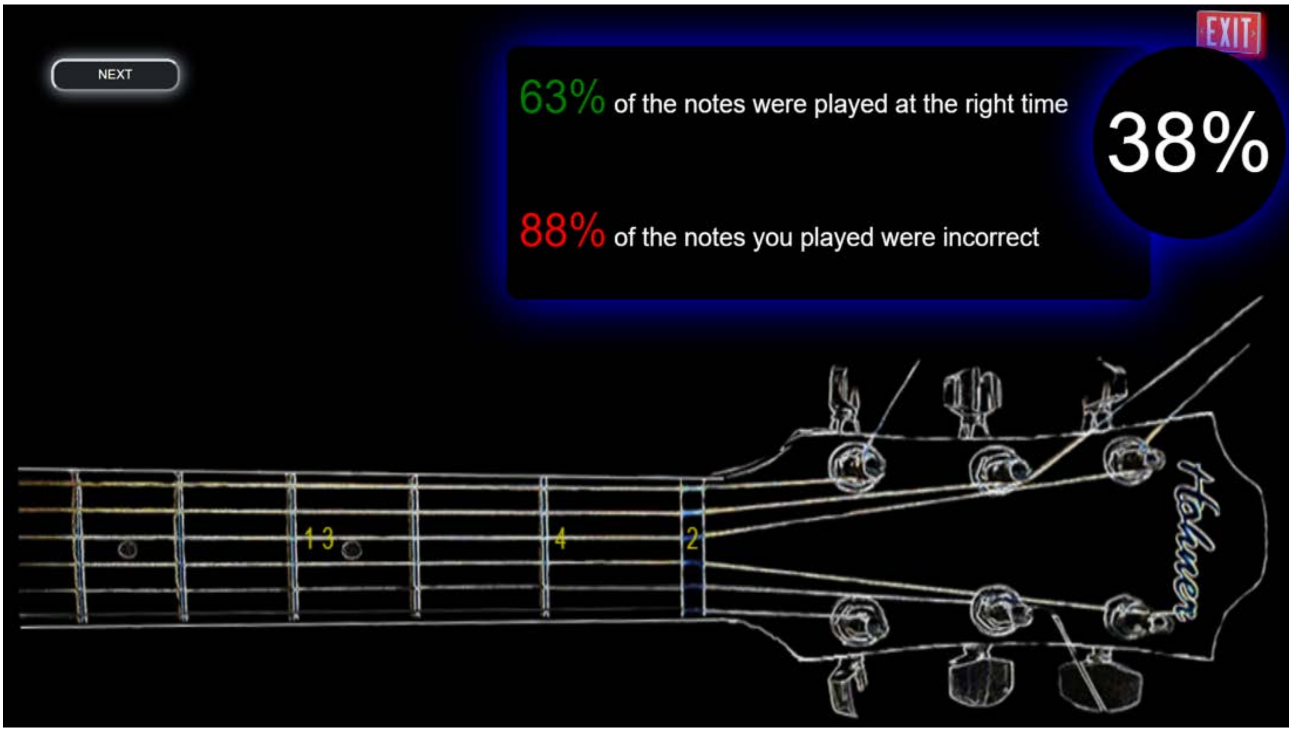

Figure 21. guitMaster - Mixed Feedback (positive first)

4. Mixed feedback (negative aspect first) – first part of the feedback highlighted the negative aspects of the performance, second part the positive (e.g. 75% of the notes were played at the wrong time but 50% of the notes you played were correct. Overall 37.5%). The information was coloured according to the overall percentage (0%-50% in red, 50%-70% in dark blue, 70%-100% in green). The correct and incorrect percentages were coloured in green and red accordingly (see Figure 22).

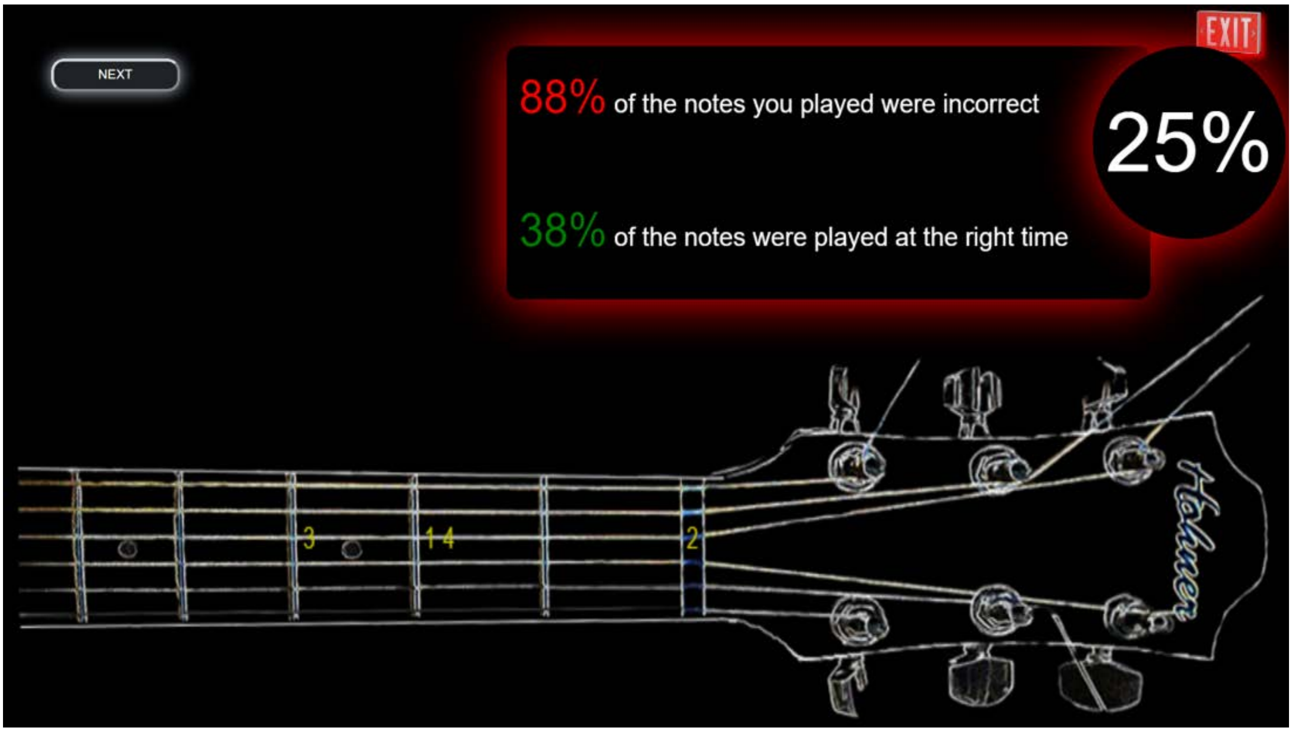

Figure 22. guitMaster - Mixed Feedback (negative first)

The testing was conducted in three sessions ranging between 4 and 8 hours. There was a total of 25 participants aged 20 to 35. The average test time was 45 minutes. Participants were split into groups using two distinct measures – by feedback and by proficiency level. Before starting the test, each test subject was asked to rate their ability to play guitar. Their answers determined which feedback presentation method group the participant would be assigned to. Removing the random learner assignation into the feedback presentation groups assured that all of the feedback presentation variations would be assigned an equal split of beginner, novice and proficient users. Subjects by feedback presentation type:

- 7 Positive feedback
- 6 Negative feedback
- 6 Mixed feedback (positive first)
- 6 Mixed feedback (negative first)

Subjects by skill level:

- 8 Proficient
- 5 Intermediate / novice
- 12 Beginner

Data was saved in a .txt file format (a separate .txt file for each experiment participant). Information was pulled from the text files into excel spreadsheets and analysed after all 25 tests were finished.

4.2. Results – Data

The first data measure was average top score. This shows the maximum score a user achieved over all their instruction-practice-assessment (I-P-A) cycles (see Figure 23). “Average top score by Feedback Type” in Figure 23 contains five fields: *“POSITIVE”*, *“NEGATIVE”*, *“MIXED POS”*, “*MIXED NEG” and “MIX MIX”*. The latter one (“MIX MIX”) is data combined from both mixed feedback variations.

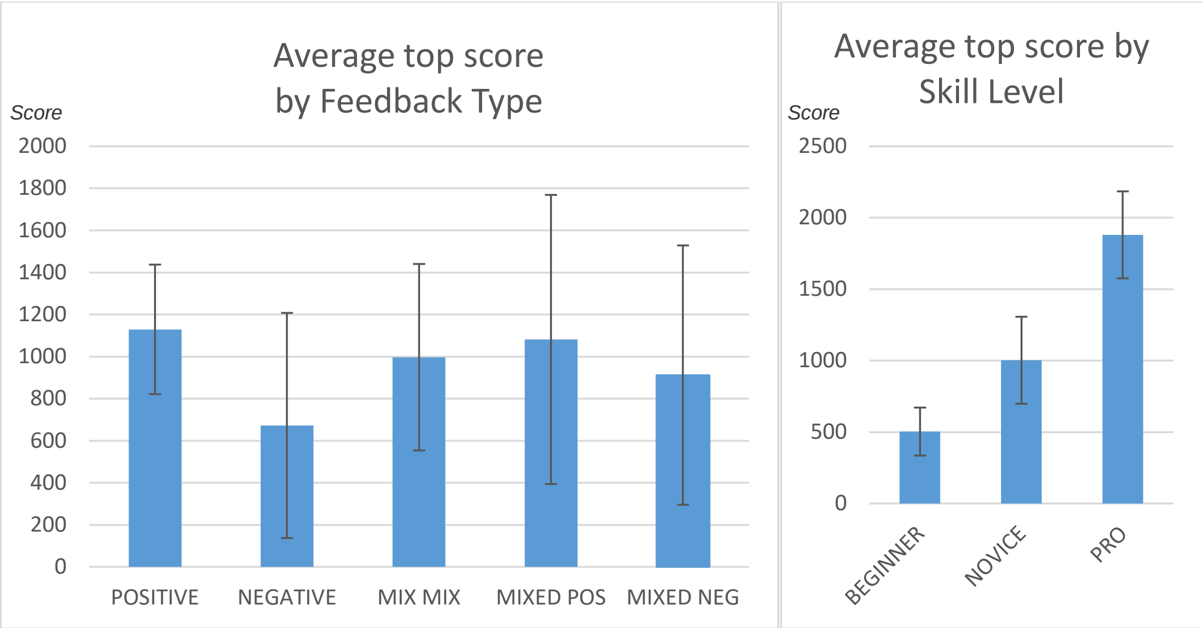

Figure 23. Average top score.

Following is average measures of the time the learner spent in the part of the experiment they were required to use the software. That is, the time the user spent in the I-P-A cycles (see Figure 24). This data excludes the time taken to familiarise with the software product as well as the time taken to complete a survey at the end of the experiment.

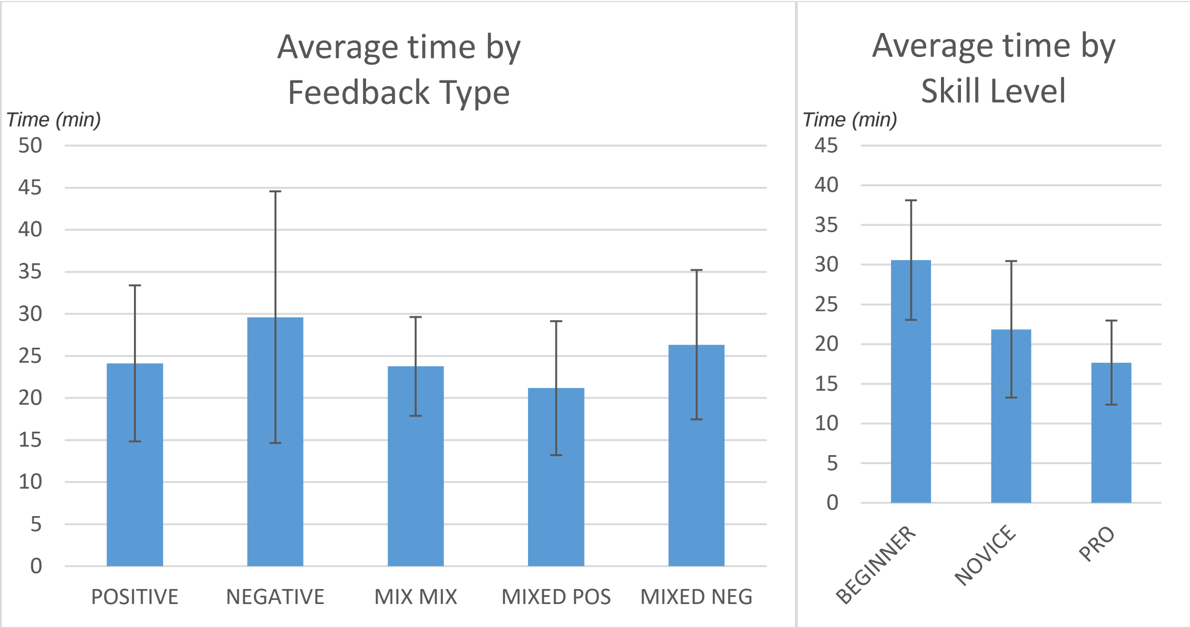

Figure 24. Average usage time.

The final data set gathered by the software is the average improvement. The average improvement was calculated by first averaging the results of the 2nd, 3rd and 4th I-P-A rounds. 1st assessment result was disregarded and the first assessment was left for the learners to get familiar with the assessment structure. Then the results of the 5th, 6th and 7th I-P-A cycles were averaged and an increase in percentages was calculated (see Figure 25).

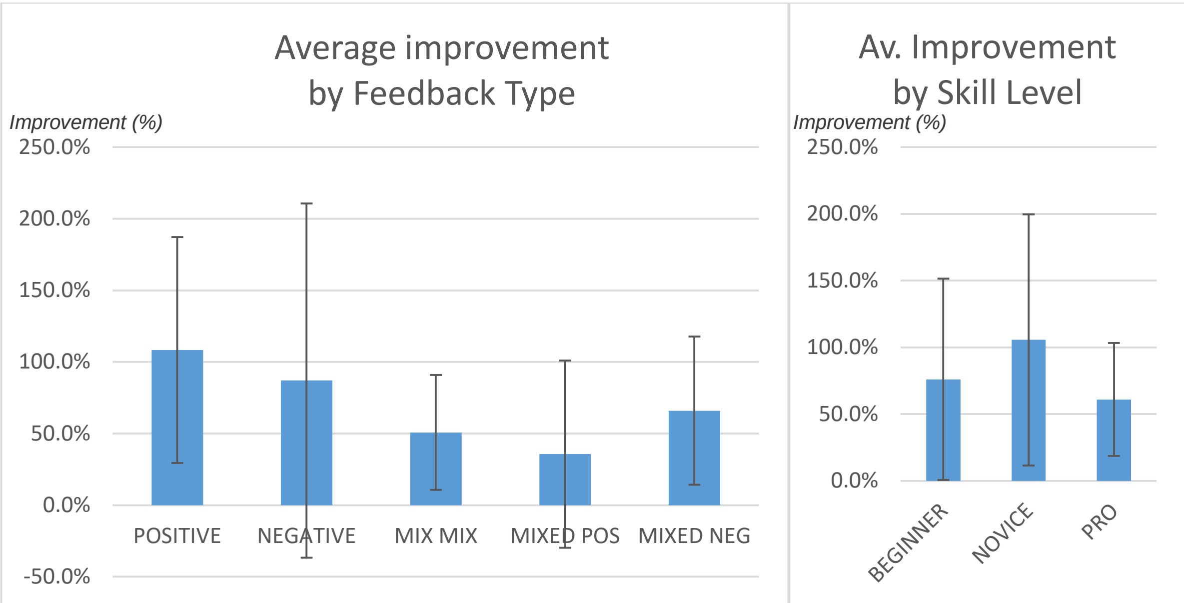

Figure 25. Average improvement.

The data for the following measures were acquired by asking the participants to complete a short survey at the end of the experiment. First – average user satisfaction score (see Figure 26)

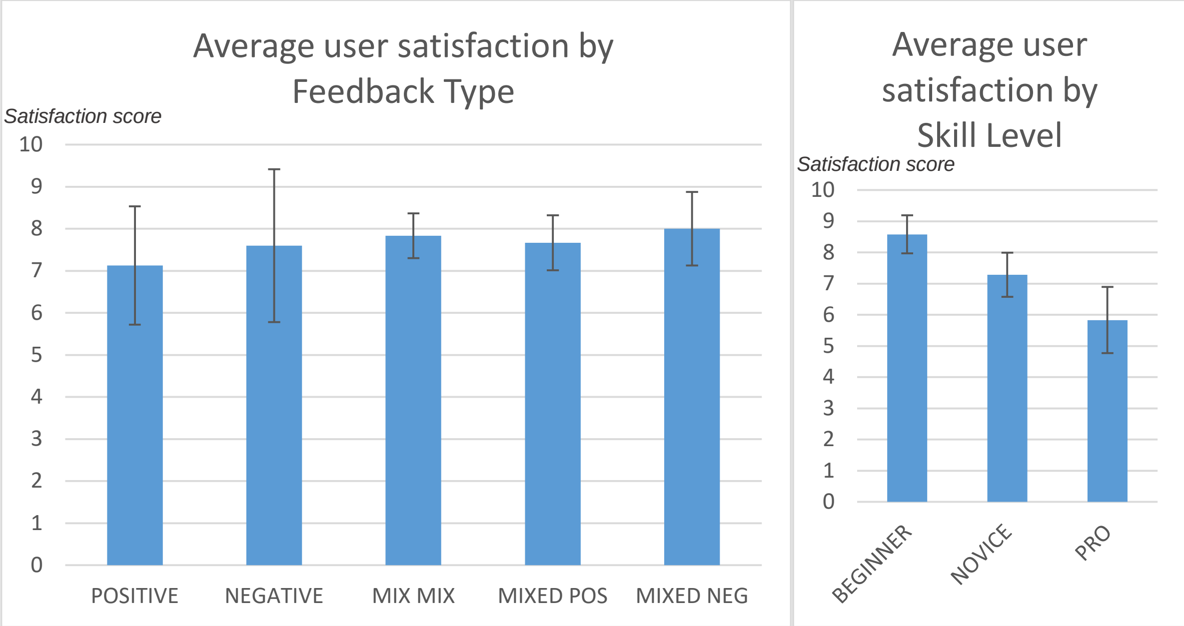

Figure 26. Average user satisfaction.

Then the participants had to pick five words out of thirty that “best describes their experience using the app”. They had 30 words (15 positive and 15 negative) to choose from. After all 25 tests had concluded each positive word chosen by the learner has a “weight” of +1 and each negative word a “weight” of -1. After summing the five words chosen and averaging the result for each feedback type group a positive versus negative word index was calculated (see Figure 27).

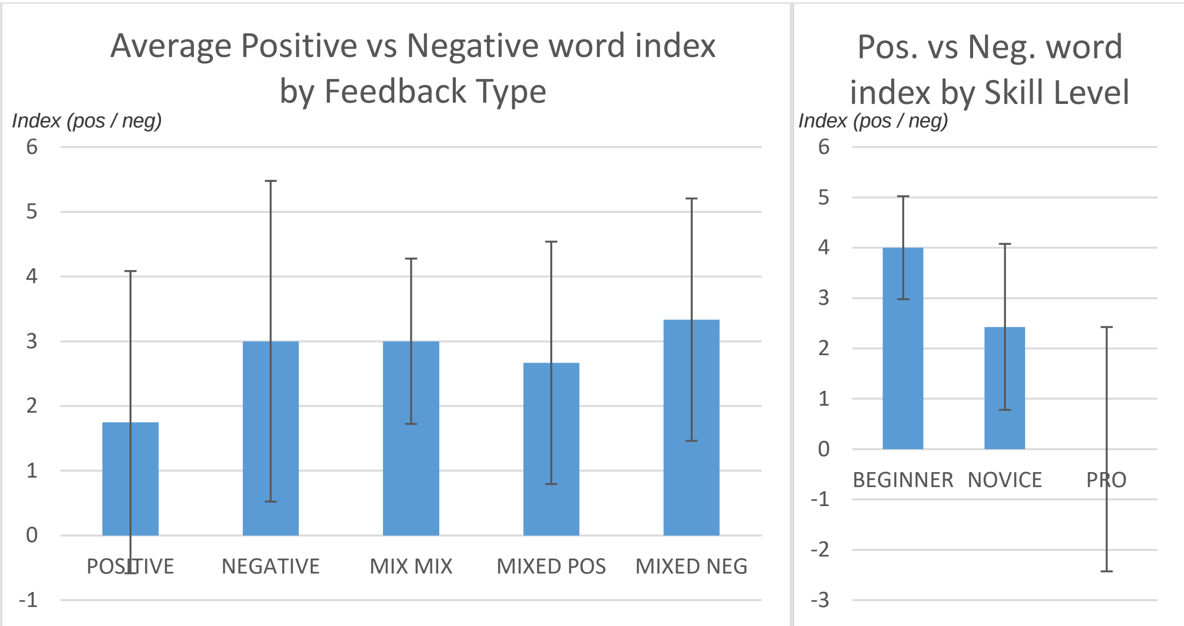

Figure 27. Positive vs. negative word index.

The following four figures (Figures 28 - 31) show the assessment scores over I-P-A rounds. The y-axis is the assessment score, the x-axis the I-P-A cycles. The four figures depict information as split by the different type feedback groups. Each line in the graphs below represents assessment score for each practice-assessment round for each individual test subject.

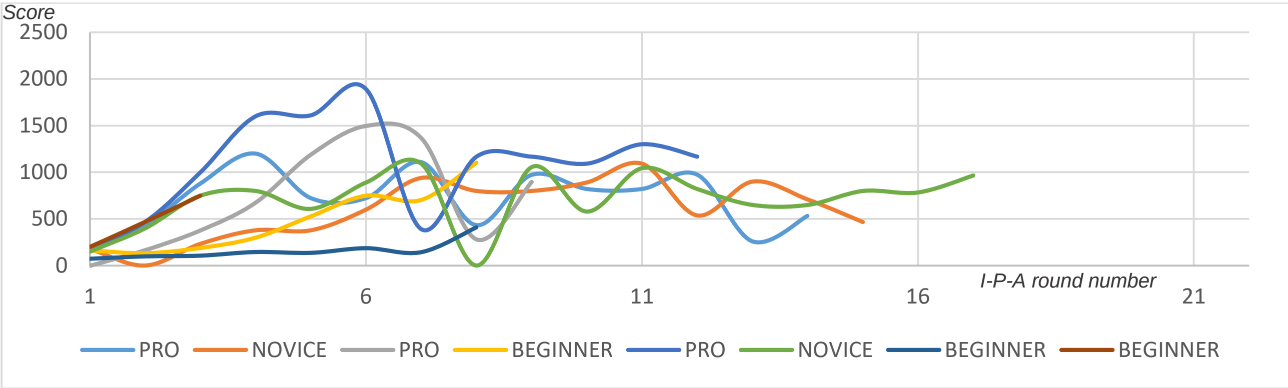

Figure 28. Assessment scores over I-P-A rounds - Positive feedback.

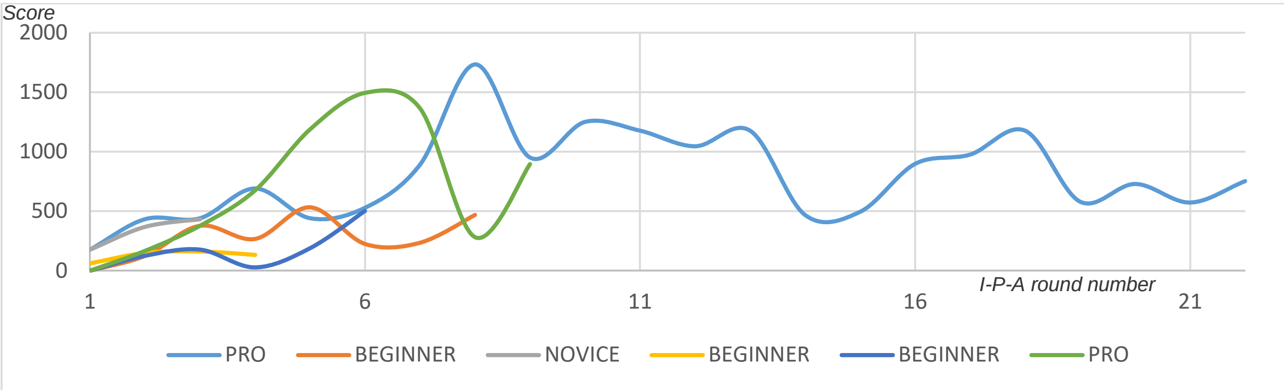

Figure 29. Assessment scores over I-P-A rounds - Negative feedback.

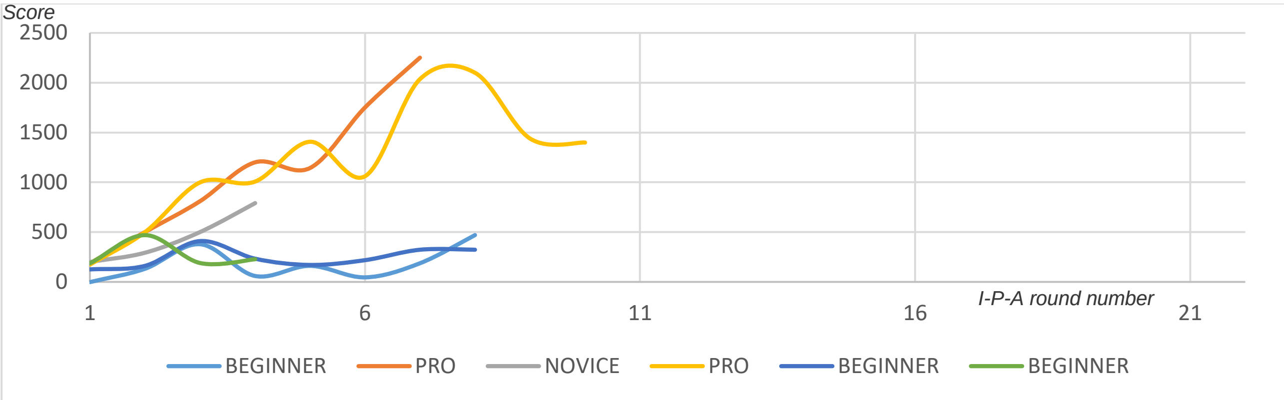

Figure 30. Assessment scores over I-P-A rounds - Mixed feedback (positive-first).

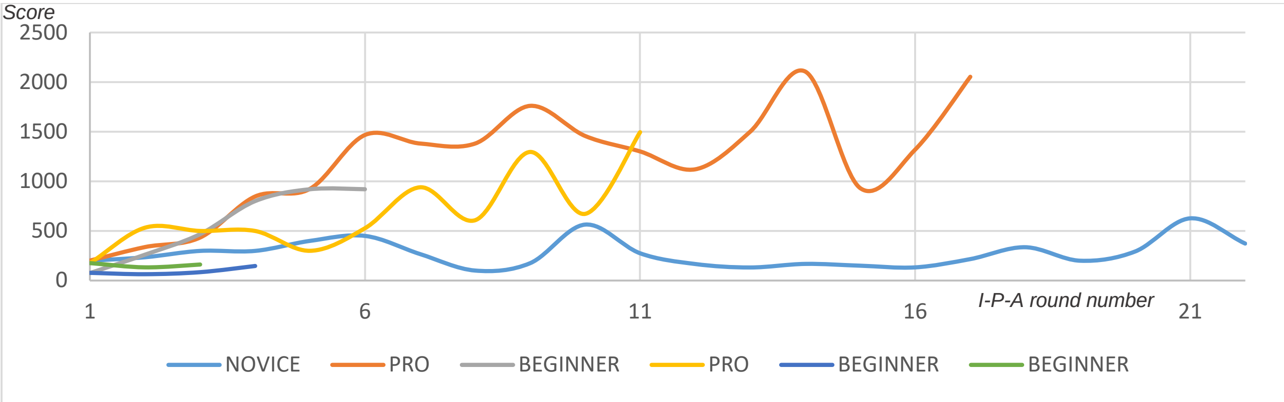

Figure 31. Assessment scores over I-P-A rounds - Mixed feedback (negative-first).

4.3. Result analysis – discussion

Looking at the results presented in the section above it is clear that a sample size of 25 participants with subjective data and results influenced by both skill and opinion is too small to draw any significant conclusions.

Confidence intervals in the Figures 23-27 were constructed using 95% confidence level. There is significant overlap between the groups when ordered by feedback type. With each feedback group containing participants varying by skill level, both skill and opinion related data varies significantly.

On the other hand, when results are analysed with the data split by skill level there's a clear difference in the groups (Proficient, Novice and Beginner users) in nearly all measured data aspects.

Beginner users spent more time using the software, gave on average higher satisfaction scores, chose more positive words as compared to novice and proficient users.

There are subtle differences in the measured data between the test subjects as grouped by feedback, though the confidence intervals show that the data is not extensive enough to be conclusive. To find the optimal type of feedback the following two steps were taken:

1. Normalize the data by applying standard score calculation (also known as z-score) where the average of a measured data set becomes 0 and the standard deviation 1.
2. Use the standardised dataset to add values and their relative weights for each measure (e.g. positive vs. negative word index, score given, average improvement etc.) grouped by feedback type
3. Derive the feedback type with the largest sum (achieving best results across all measures)

Weighted values are used to take into account any types of feedback that may perform better by a large margin in some of the measures and perform similar to others on the rest.

Performing the above three steps on the following measures: top result, time spent using the software, score given, Positive vs. Negative word index, Improvement and time spent practicing the melodies the following is derived:

	Top Result	Time	Score	Pos/Neg words	Improvement	Learning time	Σ
POS	0.2368557	-0.073288	-0.279436	-0.313883	0.4931679	-0.057044	0.01
NEG	-0.461698	0.3652405	0.0256952	0.1477097	-0.269191	0.2573613	0.07
MIX | P	0.1643842	-0.310240	0.0685207	0.0246182	-0.484237	-0.245404	-0.78
MIX | N	-0.095443	0.1035916	0.2826481	0.2708012	-0.17691326	0.1069955	**0.49**

Table 2. Accumulated data set across all measured aspects split by feedback type

The table above shows that if every measure from the experiment is considered, mixed feedback presenting the negative aspect first performed better than others.

Another measure is the free typed comments / suggestions each user can submit after the experiment. For some examples see (APPENDIX B.1).

The most commonly found comment is concerning the user interface. Participants across all four feedback variations were in agreement that to make the overall process more interesting the user interface had to be improved.

The overall consensus observed by reading the comments as well as communicating with the experiment participants after the tests was that the interface is too basic, too simplistic and in general – too boring. Some examples would be: *"(colour coding the frets or strings might help) More attractive interface."*, *"GUI should be way more interesting."*, *"the interface is really too poor. I'd have more colours more stuff there"*.

A feedback-specific common comment which also relates to the citations above is one about the textual negative feedback displayed in the colour red. Four out of six participants who were assigned to the negative feedback variation verbally communicated that they did not enjoy seeing the word "WRONG!" in front of them every single time they took an assessment. Three left specific comments concerning the usage of negative colours and language: *"When you get your score at the end of the test, would it be possible to state how much you got correct in green"*, *"Use percentage correct as opposed to incorrect after tests for feedback"*, *"i don't like that all's in red at the end there. If it was me I'd have more colours to make it more fun and less depressing. all it seemed to tell me that I managed to always be wrong to some extent"*.

4.4. Limitations – discussion

When looking at assessment score plots by feedback type (see Figures 28 - 31) an important trend can be observed – often across all feedback types (most prominent in Figure 28) the users reach a certain maximum score and then dramatically plummet down. When looking at the possible causes two can be found:

1. The automated software difficulty level increase is too steep increasing the difficulty level too rapidly
2. The scoring system is flawed

The software increases the difficulty by 1 string and 10 BPM every time the student completes a task with 70% or higher overall score (1 string and 20 BPM if the tutee completes the task on their first try). The difficulty level may be increasing too drastically. A redesign of the difficulty increase algorithm is needed.

The scoring system is set up in such a way that the student may get a high score with a relatively easy melody. The score is calculated taking the current tempo and multiplying it by the possible number of strings the melody can consist of. There is a chance that the learner may get a melody possibly consisting of six strings yet actually consisting of one (the notes are chosen randomly from the available six strings). In such case the current score may be unrealistically high whereas the following melody may actually consist of six strings. In such a case, there is a high likelihood the tutee might fail which could discourage him/her from practicing further.

4.5. Conclusions

Very specific comments addressing the user interface and the over usage of negative language and negatively perceived colours prove that the presentation of informational feedback does in fact matter. The fact that these feedback specific comments were received only for the negative feedback type test participants shows, that receiving only negative feedback is detrimental to the learning process. It is likely discouraging learners rather than motivating them. This may also impact the flow of the learner's journey. The flow and the balance between anxiety and boredom discussed in Section 2.5 must be revisited. Adjusting the difficulty and avoiding discouragement by negative content may help decrease users opting out shortly after starting the practice session.

Another aspect worth mentioning is discouragement by failure. Due to drastic increases in difficulty the learner is unexpectedly presented with a very difficult melody (as compared to the previous melody) and therefore is likely to fail the following assessment. The difficulty increase logic needs to be revaluated. In order to correct the anxiety – boredom balance the relative importance of tempo and note distance must be considered. Note distance - the distance between two concurrent notes on the x-axis (along the fret board) and on the y-axis (different strings) – must also be reviewed (see Figure 33). The relative weights of these variables when calculating and assigning the overall assessment score must be found prior to conducting any subsequent experiments. Drastic rise of the difficulty can increase anxiety and push the learner out of the correct flow balance. Figure 32 below outlines several cases across different user groups (Positive, Negative and Mixed feedback) where the user assessment score drops drastically due to a large unexpected increase in difficulty. The most drastic drop in user scores can be seen in most of the user scores in the left figure (Positive feedback), green and light blue lines in the middle figure (Negative feedback) and yellow and orange lines in the right figure (Mixed feedback (negative-first)).

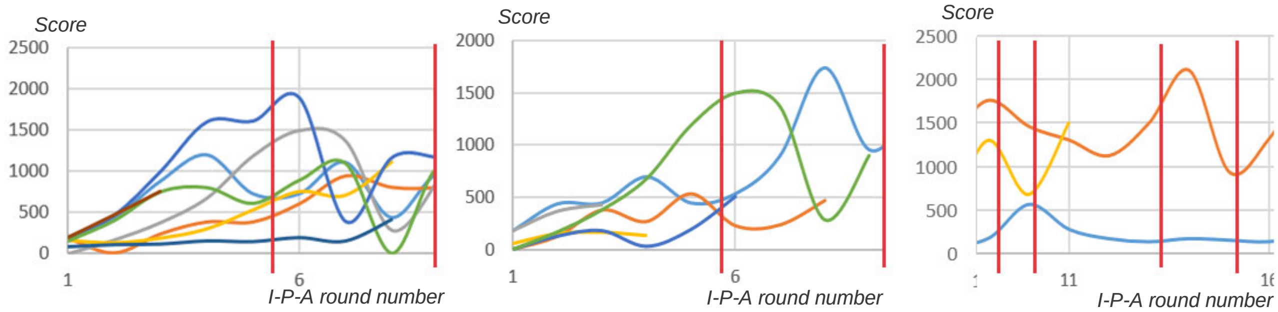

Figure 32. Dips in the learner's result due to drastic difficulty increase

5. Experiment 2 – Difficulty level and scoring system

5.1. Context

A conclusion found in Experiment 1 indicated that to acquire more accurate and reliable results and prevent drastic difficulty jumps a new scoring system and difficulty control had to be designed. During the first experiment if a learner successfully completes a round, the number of possible strings for the melody is increased by one and the tempo is increased by *10BPM*.

As discussed in the previous chapters a random melody generation approach was chosen to eliminate any possible bias against or for known melodies. Well known melodies may introduce a bias by providing certain participants with melodies they may enjoy whereas other participants may not. Therefore, the software picks four random notes on as many strings as it is allowed by the current level of difficulty. This introduces a chance that four notes can be randomly picked on the same string making the melody much easier than the given melody score may indicate. The melody score is calculated by multiplying the current tempo (BPM) and the <u>possible</u> number of strings i.e. if a melody can contain 4 strings and is currently played at *120BPM* the melody score would be *4 * 120.*

The learner would likely pass this assessment and get a high score (due to the melody being easy) and then get four more random notes picked. This time the software has an additional string to pick from as the user successfully completed the previous assessment. Therefore, the number of strings for the melody was increased (until it reaches the maximum number 6). If the app randomly picks all four notes on four different strings the calculated melody score would not be much higher that the score of the previous melody because the <u>possible</u> number of strings only increased by one. However, the actual difficulty of the melody increased dramatically and therefore the actual melody score should have increased with the difficulty.

To solve this problem two main approaches were taken:

1. First the number of strings should no longer be increased upon every successful completion of an assessment. The string number should rather be increased every third or fourth successful completion.
2. The melody score and therefore the user result score should be directly related to the actual difficulty of the melody.

In order to get the actual difficulty of the melody the app should calculate note distances - the distances between two concurrent notes on the x-axis (along the fret board) and on the y-axis (notes on different strings) (see Figure 33).

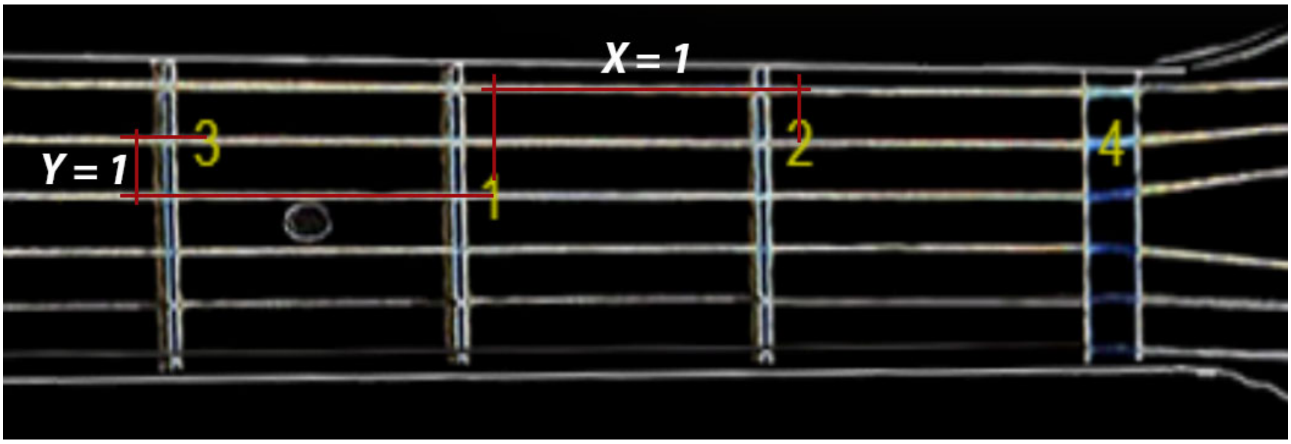

Figure 33. Difficulty scoring X & Y values and meaning

The figure above provides two variables *X & Y* to be used for difficulty estimation. The third variable readily available to the software is tempo.

In the specific context of computer-based instruction (CBI) in guitar skill training application the most reliable way of deriving the difficulty / score formula was to conduct a separate test / survey. The main aim of the test was to find the optimal solution to a linear equation:

$$D = \alpha \times X + \beta \times Y + \gamma \times T, \qquad (1).$$

where D is difficulty, X is distance of consecutive notes in the X axis (along the fret board), Y is distance of consecutive notes in the Y axis (between different strings), T is the current tempo and α, β, γ are the relative coefficients (or relative weights) for each of those variables. The aim of the experiment is to determine α, β, γ so that D can then be estimated based on X, Y, T.

Ten test subjects with advanced guitar skillset were asked to score 24 melodies presented in a randomized order. The participants were presented with 6 different melodies each containing one, two, three etc... strings played in four different tempos: 40 beats per minute (BMP), *80BPM*, *120BPM* and *160BPM*. These tempos were chosen to cover the range present in the software in equal steps. The scores given by the users were averaged to produce the difficulty score, D, for that melody.

5.2. Implementation

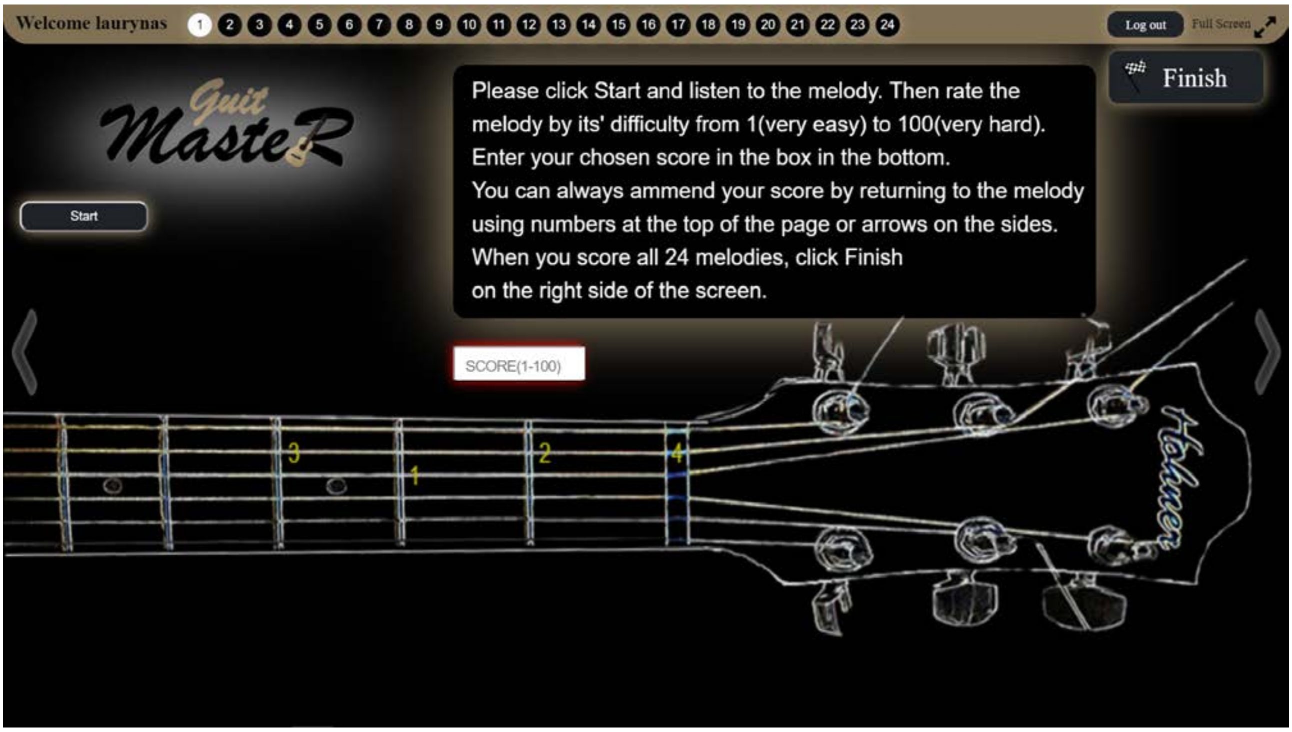

Figure 34. Difficulty level and scoring system screen capture.

Each user was asked to score all of the 24 melodies by its difficulty in a 1 – 100 range. When a sufficient number of test subjects was reached, the data was analysed, and a linear optimization algorithm deployed to determine the values of α, β, γ coefficients. A visual representation of the linear optimisation (see Figures 35 - 39) was created alongside a brute force approach (checking every possible solution to the equation) with the use of the programming language JavaScript. The visual representation will help the reader to clearly see the how the scoring / difficulty formula is being optimized for a given data sample. The user interface screen capture can be found in Figure 34 above.

Performing linear optimization allows finding the optimal solution to a linear equation. That means that given a function with a number of unknown variables and a dataset this function should fit, optimization will find optimal variable values where the function fits the dataset closest. Figures 36 – 38 show all three variables (note X distance (see Figure 36), note Y distance (see Figure 37) and tempo (see Figure 38)). The plain averaged dataset is presented below in Figure 35. The three sliders present in each figure were developed to visualise the optimisation. The slider values range from 0 to 1. These sliders allowed tweaking the relative weights with instant preview of the optimised curve.

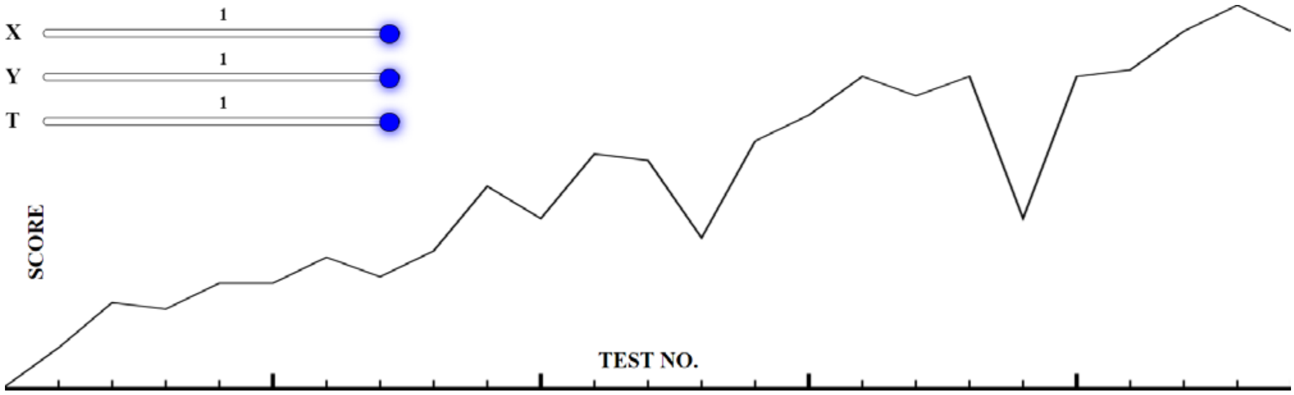

Figure 35. Linear optimization 1 - Given scores (plain dataset).

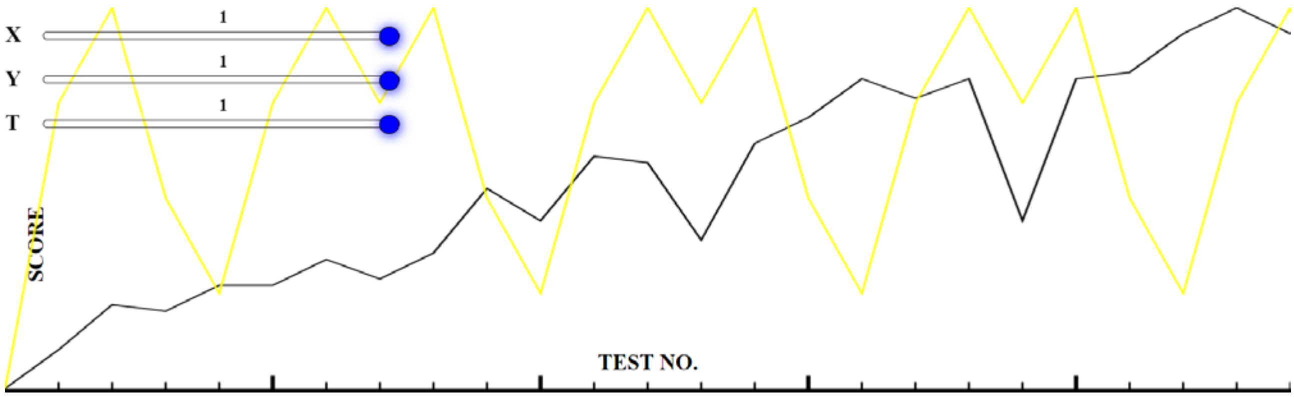

Figure 36. Linear optimization 2 - Given scores + normalized note "X distance" values.

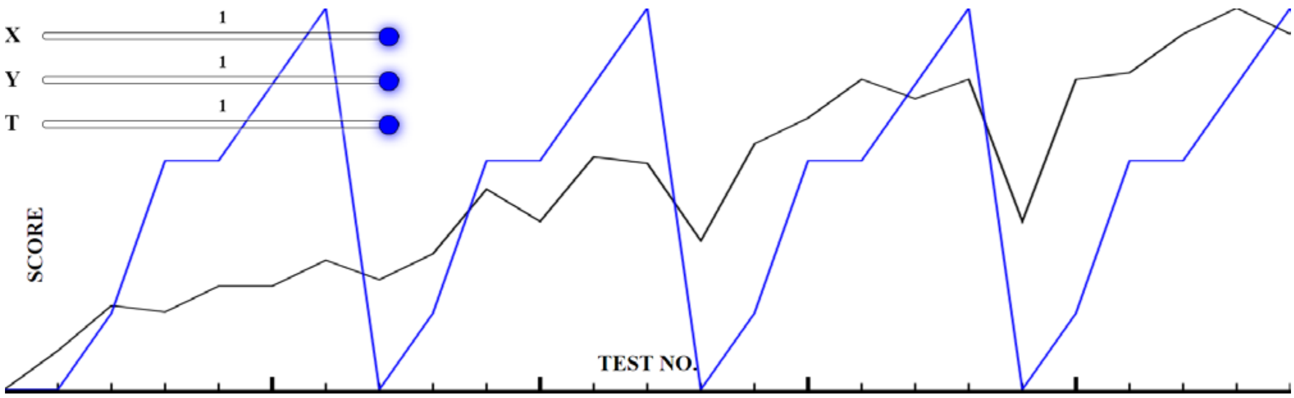

Figure 37. Linear optimization 3 - Given scores + normalized note "Y distance" values.

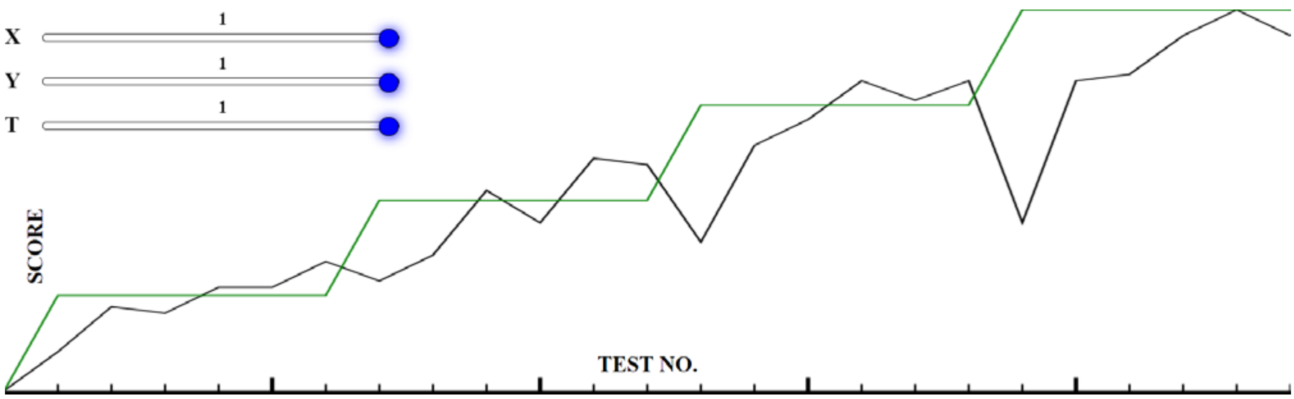

Figure 38. Linear optimization 4 - Given scores + normalized tempo values.

Placing the "X & Y Distance" and tempo values in the linear formula (Equation 1) with the relative α, β and γ weights set to 1 the following path is found (see Figure 39):

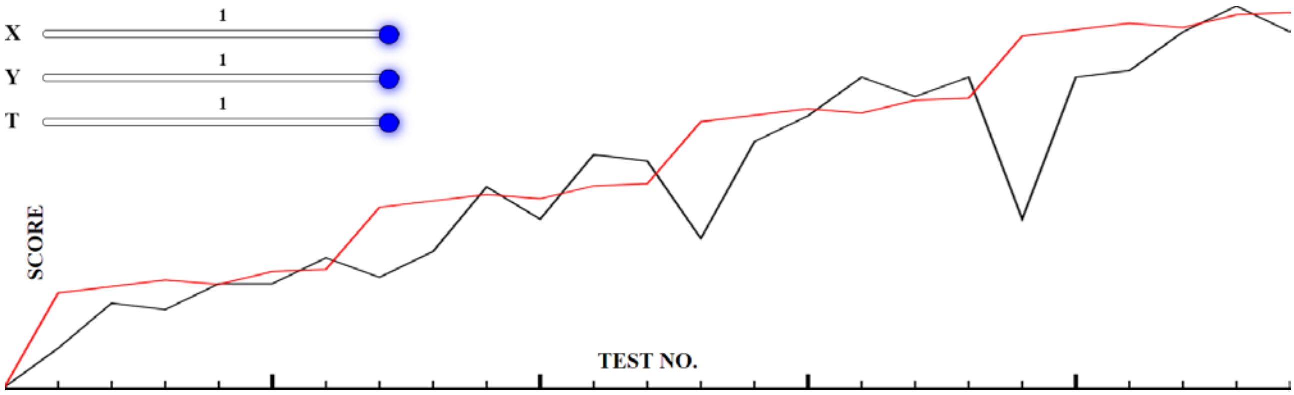

Figure 39. Linear optimization 5 - Given scores + normalized tempo, "X-Distance" and "Y-Distance" values passed through a linear equation.

The three sliders on the top left-hand side (see Figure 40) provide a manual way to adjust the relative weights of each of the three variables (tempo, X & Y note distance).

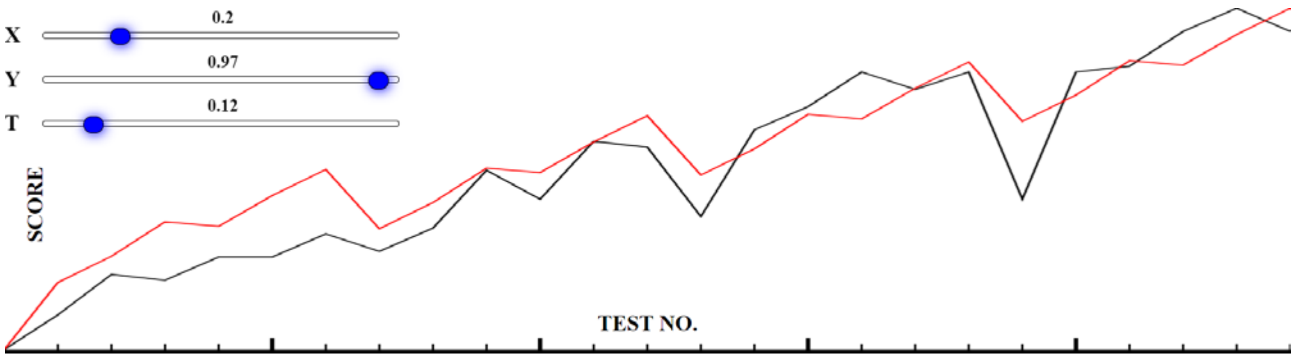

Figure 40. Linear optimization 6 - Manual variable weight adjustment.

The optimization algorithm goes through every possible value of α, β & γ (that's the three-level *for* loop seen in Figure 41. α, β & γ represented by variables a, b & c in the figure below) and for each melody score (the participant given score) looks at the melody difficulty this result was scored at ("X & Y Distances" and tempo), multiplies these three variables by the current iteration values of α, β & γ then subtracts the difficulty score D and adds the remainder to the total tally (*minFunc* see Figure 41) for the current iteration values of α, β & γ. After all scores have been exhausted the total tally should be as close to 0 as possible for a perfectly optimized function. After iterating through all possible α, β & γ their values that produced the smallest total tally are the values that optimize the given function best.

```
var minFunc = 0;
var finalFunc = 1000000000;
var minA = 1;
var minB = 1;
var minC = 1;
for(a=0;a<=1;a+=0.01){
  for(b=0;b<=1;b+=0.01){
    for(c=0;c<=1;c+=0.01){
      minFunc = 0;
      var i = 1;
      $('.individual_score').each(function(){
        i = i == 25?1:i;
          minFunc += Math.abs(a*(difficulty[(i)].X) + b*(difficulty[(i)].Y) + c*(difficulty[(i)].tempo) - difficulty[(i)].Score);
        i++
      })
      if(minFunc < finalFunc){
        finalFunc = minFunc;
        minA = a;
        minB = b;
        minC = c;
      }
    }
  }
}
```

Figure 41. Linear optimization 7 - The brute force optimization algorithm.

5.3. Results

The brute force approach found the optimal solution but it dismissed the X distance all together (this may be specific to the data set scored by the guitarists). The algorithm (see Figure 41) was producing a sum smaller by about 1% by dismissing the X distance (setting a (α) to 0), but the X distance is a factor that should not be reduced to zero. Deploying the visual optimization tool, the brute force approach was constrained for the X distance coefficient value to not go below 0.2, so as not to dismiss it completely.

Another factor that was found during the data analysis was that at the lowest tempo of 40 BPM the data curve was behaving in a non-linear way – not in accordance to the rest of the tempo values (Figure 40), so specific low tempo scalar coupled with an "if" statement was added to account for this. The low tempo scalar (set to a value of 0.54 by trial and error method) and the if statement are written in the pseudocode below.

```
α = 0.2;
β = 0.97;
γ = 0.12;
low_tempo_scalar = 0.54;

if tempo <= 40 {
   α = α * low_tempo_scalar;
   β = β * low_tempo_scalar;
   γ = γ * low_tempo_scalar;
}
```

The final optimized curve with the low tempo factor (described above) is presented in Figure 42.

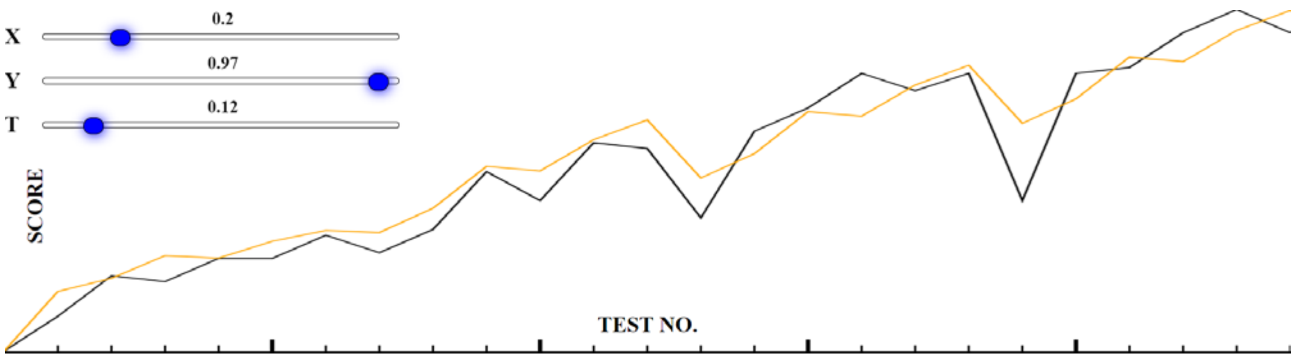

Figure 42. Linear optimization 8 - Final curve.

5.4. Conclusions

The new scoring algorithm found in this chapter confirmed the hypothesis outlined in the conclusions section of Experiment 1. The consecutive note X and Y distances and the melody tempo is not all equally important when determining the difficulty of any given melody. Results of this experiment indicate that the main factors determining the melody difficulty are the tempo and the note Y distance (the melody being played on different strings). The relative weight of the melody tempo if the smallest but the tempo range (40-160) is much greater than the range of note X distance (0-3) or note Y distance (0-6).

The consecutive note X distance presents the smallest effect when determining the melody difficulty. The X distance outcome is not unexpected. When changing the fret (X distance) only the fret board holding hand requires change in position as opposed to when changing strings both hands need to be repositioned. This requires a much higher level of coordination and therefore has a bigger impact when determining the difficulty of a melody played on a guitar.

Most notable limitations of the experiment were the number of participants and the number of melodies scored by each participant. The limitations were brought upon by time constrains and the need to have skilled guitarist take the test. 10 participants in total completed the test. 24 melodies were evaluated to not fatigue the subjects and ensure they all completed the experiment.

A further improvement to the algorithm could be made by repeating the experiment with a bigger group scoring a bigger sample of melodies.

Alongside the improved scoring system a more gradual difficulty increase was introduced to the system. The difficulty increase presented in Table 3 (page 34) was adjusted to more gradually increase/decrease the difficulty as the learner progresses. The main adjustments were:

1. Increase decrease the tempo by *5BPM (10BPM* in the case of first time success*)*
2. As presented by the findings of Experiment 2 the consecutive note Y distance (i.e. the number of strings the melody consists of) is the leading factor for increasing the apparent difficulty of

a melody. Therefore the system was adjusted to increase the number of strings by 1 only when the user has successfully completed a level (passed the assessment round with a score of 70% or higher) three times in a row. The decrease of the number of strings was kept to the original behaviour decreasing the number of strings by one every time the learner fails the assessment (fails the assessment round with a score of 29% or lower))

With the improved scoring system developed during the course of the Experient 2 and the adjusted difficulty increase the final experiment can be carried out.

6. Experiment 3 – Deployment of gamification to increase learner's engagement

A good user experience is a factor of great importance when it comes to engaging with any system. Experiment 1 proved that a very basic design with little to no interesting details will not pass the criterion by which we judge what is interesting and engaging. A 21st century user opens an interface with the expectation to be entertained and the user interface design provided to the participants when conducting Experiment 1proved to do just the opposite. The most common improvement suggestion was "a much more interesting interface".

Referring back to Csikszentmihalyi's, "Flow and the Foundations of Positive Psychology" another way to keep the optimal flow and to prevent the learner drifting into boredom is to ensure the user experience is engaging. A well-known method of increasing interest and engagement and getting users "hooked" is introducing some level of competitive principles into the learning experience. This method is used in most of the games that attract the biggest audiences even though some of these games appear to be very simple. The psychological competitiveness of human nature pushes us to not quit until the goal is met (e.g. finding ourselves amongst the top players or exceeding our own previous achievement). We can compete with ourselves and we can compete with others. Both of these competition types will be tested during the course of this experiment.

6.1. Implementation

The user interface from Experiment 1 was improved upon by introducing more colours and adding two more application layers available directly through the web-app interface: basic guitar lessons – online videos of a professional guitar teacher and a guitar tuner pointing to an external source (see Figure 43). Moreover, a playful app logo was created and a more interesting instruction and feedback information container was added to the top right corner of the page (see Figure 44). The core layout of the user interface was not changed. Additional control over the difficulty level was given to the user. The system would still automatically adjust the difficulty (though in this experiment more gradually) but the learner had the ability to jump to any difficulty level of their choice. This allowed more advanced players to skip the first slow tempo rounds.

The two most meaningful changes and the basis for the experiment was the introduction of a personal best score (see Figure 45) and a leader board (see Figure 46). These addressed the two layers of competition – personal (competition with ourselves) and group – (competition with other individuals). The data gathering method was similar to the one used in Experiment 1. The software would gather information on the learner's score after each assessment. Total time spent using the app was also calculated automatically. An additional measure in Experiment 3 is summative score information – adding up all the assessment scores together.

Figure 43. guitMaster homepage screen capture.

Finally, on completion the participant was asked to complete a short survey: score the app from 1 to 10, leave free text feedback and choose five keywords out of a total of ten. The five most popular positive and five most popular negative words from Experiment 1 were chosen and presented to the user in a randomised order. The user satisfaction feedback interface is shown in Figure 47.

Each participant was presented with each of the three variations of the test. The test order was decided to be such. A randomised order of the variations/interfaces was first considered. A random order would ensure none of the variations would suffer decrease in any of the measures due to learner's fatigue or familiarisation of the software. However, in order to introduce new aspects to the learning process as the participant transitions from one variation to the other rather than removing functionality the following variation order was used for all participants: Variation 1 (Control), Variation 2 (Personal competition), Variation 3 (Group competition).

Variation 1 (see Figure 44) – the control with no added competitive aspects. This was the comparative benchmark. The data from this test could also be used as a comparison with the collected data from the Experiment 1. A decrease in "boring" user interface mentioned in the comments would clearly indicate that the introduction of colours, logo and a slightly more intricate design improves user experience.

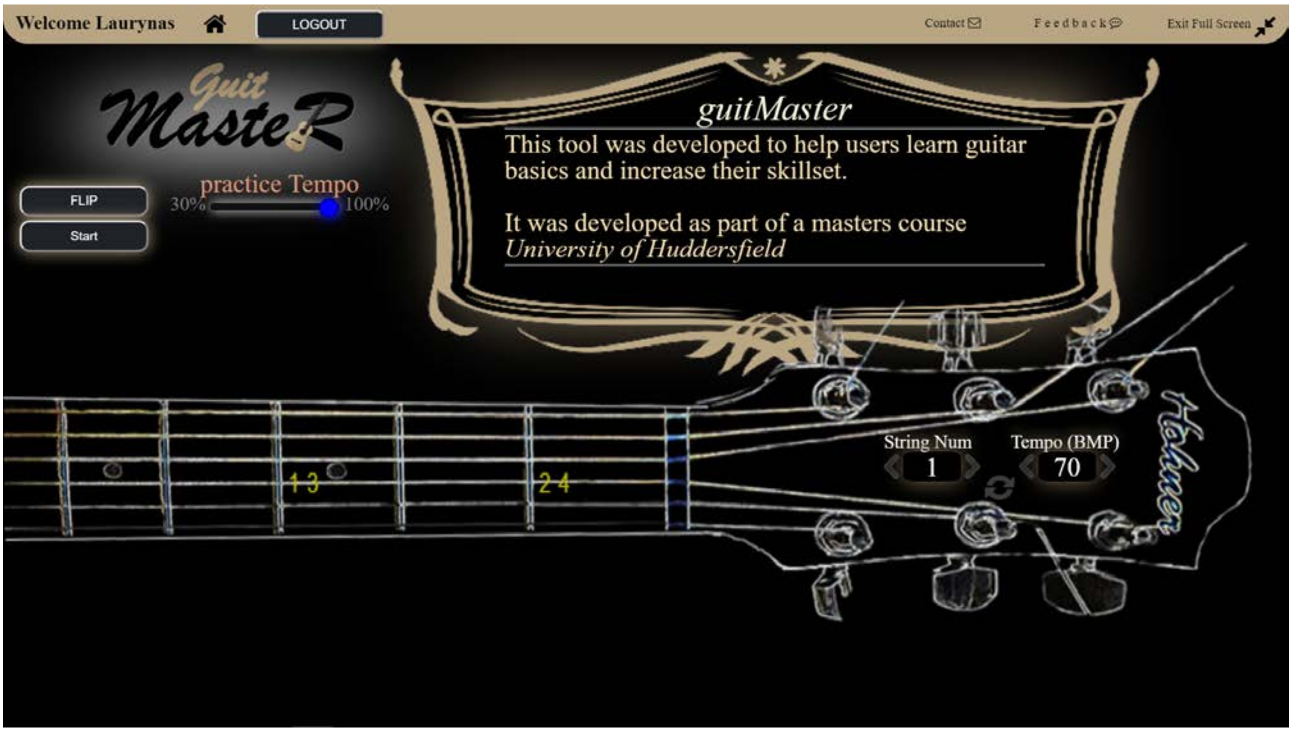

Figure 44. guitMaster screen capture. Experiment - Variation 1

Variation 2 (see Figure 45) – the personal competition test. Here the participants were provided with information on their personal best score (held in a database and retained over time) and their session best score (reset after restarting the app or navigating to the homepage).

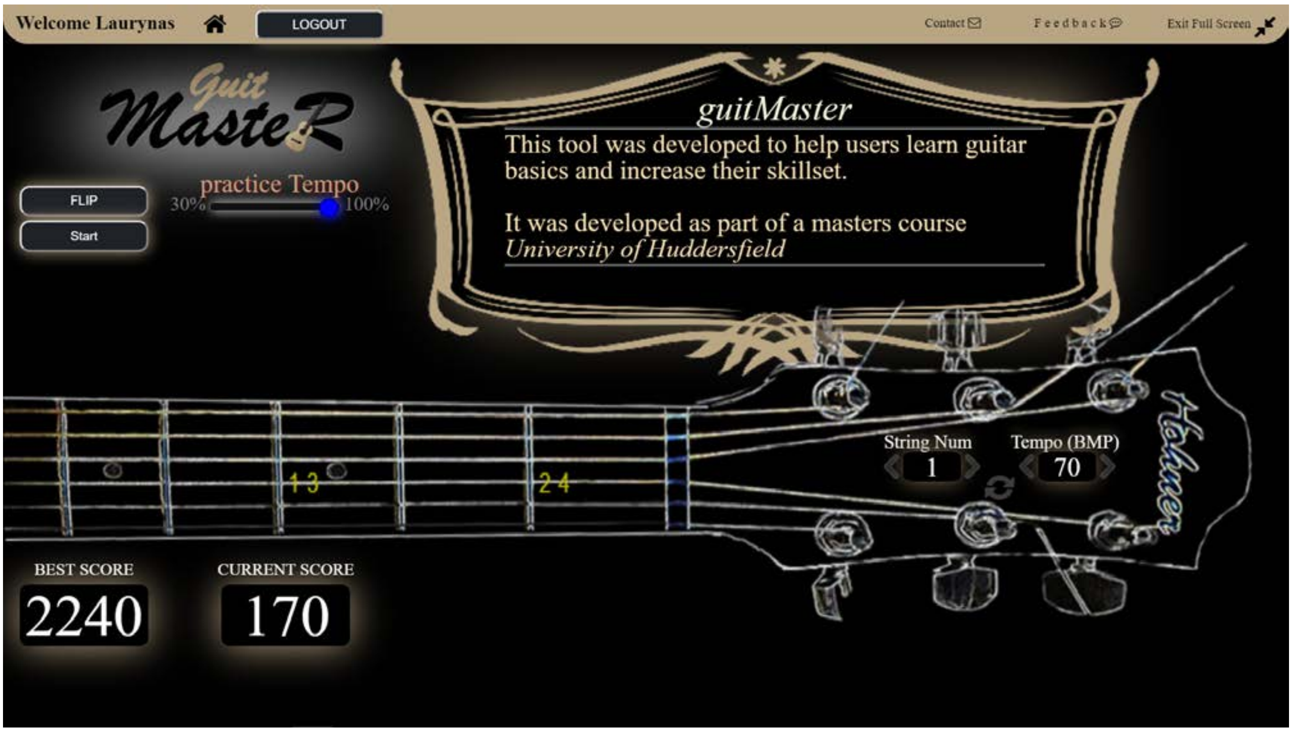

Figure 45. guitMaster screen capture. Experiment - Variation 2

Variation 3 (see Figure 46) – the group competition test. Same as in variation 2 the participants were presented with their overall best score and their current session best score. Variation 3 also provided the user with a leader board holding information on other system users' best scores. The leader board was initially filled with dummy data. A pre-test was carried out with a beginner and a professional guitar player. Top scores of these two pre-tests were used as a top score (professional player) and a bottom score (the beginner player). The remaining middle eight spots of the leader board were then filled with randomised data. Measured data used in the later analysis was collected and ordered by the software and then stored in a remote SQL database provided by a commercial third party. The learner was also presented with their number in the full leader board even if they were not within the top ten players (see Figure 46). Additional 20 spots were filled with dummy data progressively decreasing to 0. This ensured that when the tutee exceeded their previous best score their spot in the leader board would likely increase. The idea was to increase engagement and motivation by showcasing that every instruction-practise-assessment round pushes the learner higher up in the leader board.

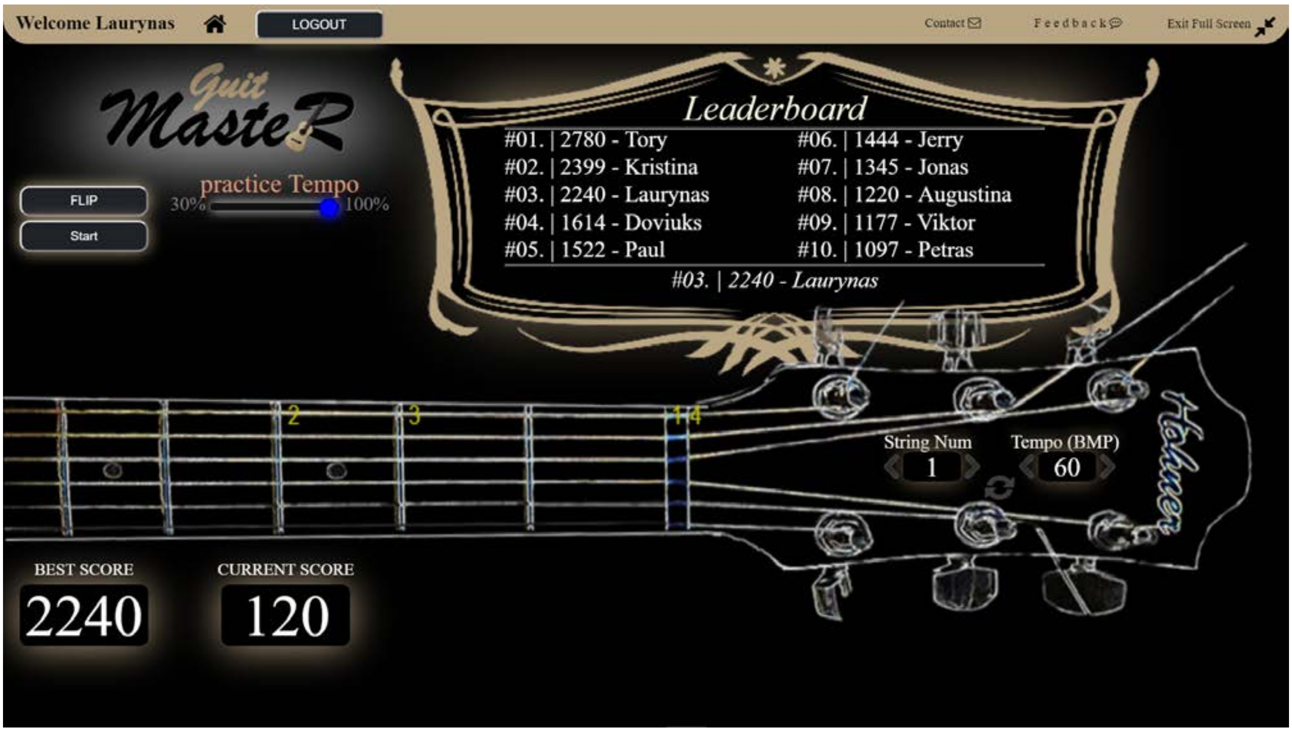

Figure 46. guitMaster screen capture. Experiment - Variation 3

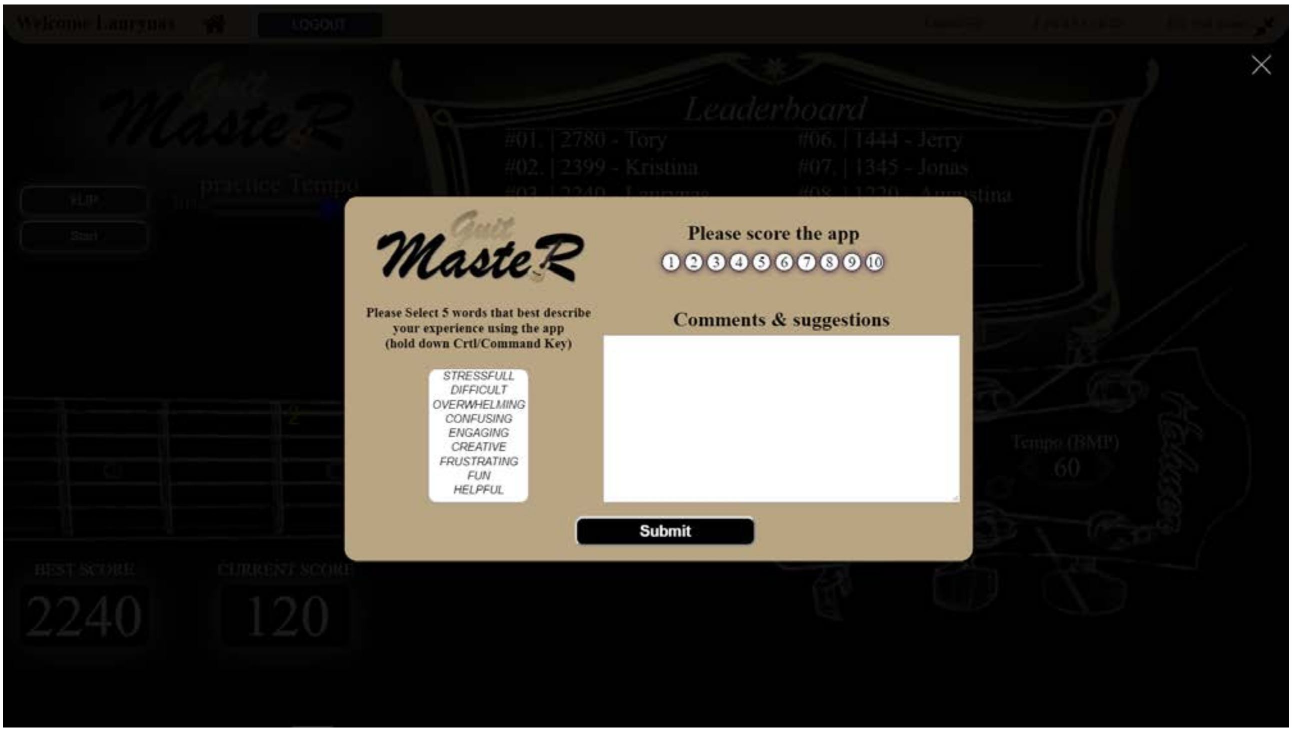

Figure 47. guitMaster screen capture. Experiment - Feedback interface

6.2. Results – Data

The test was carried out by 12 participants. Each of the participants used the app for as long as they wanted and were asked to leave a short qualitative feedback at the end of each step (after deciding to quit using the current interface). 10 of the 12 participants were beginners, 2 participants were novice guitar players.

The first data measure was average top assessment score. This shows the maximum score a user achieved over all their instruction-practice-assessment (I-P-A) cycles (see Figure 48). Next is an average measure of the time the learner spent in the part of the experiment they were required to use the software (see Figure 49). The final two data sets gathered by the software are the average overall cumulative score (see Figure 50) and the average number of rounds (see figure 51).

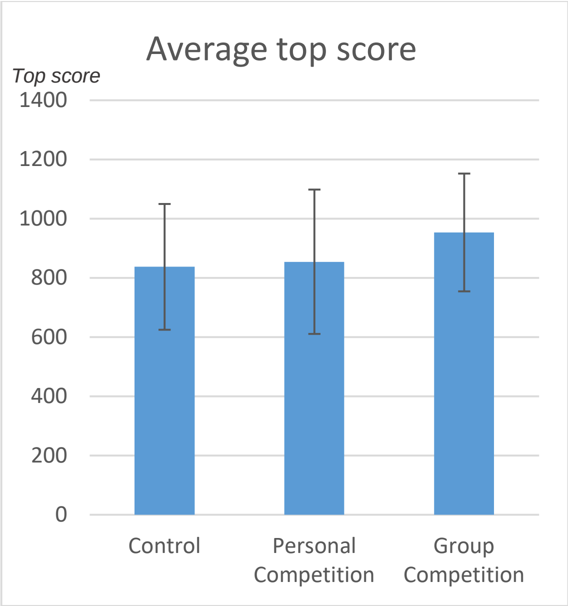

Figure 48. Average top score.

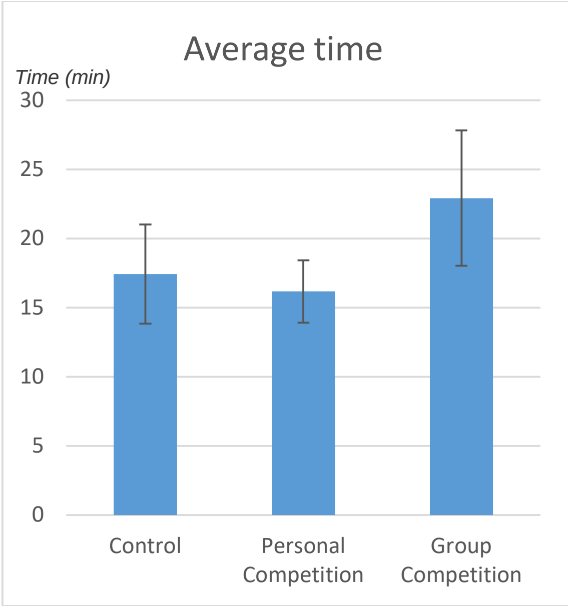

Figure 49. Average usage time.

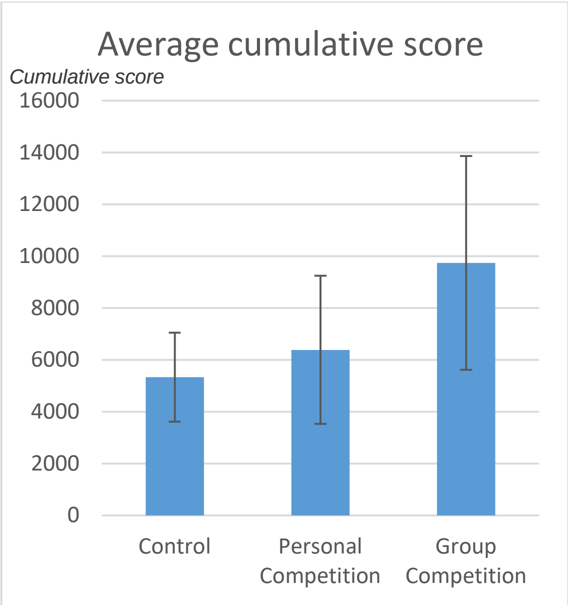

Figure 50. Average summative score.

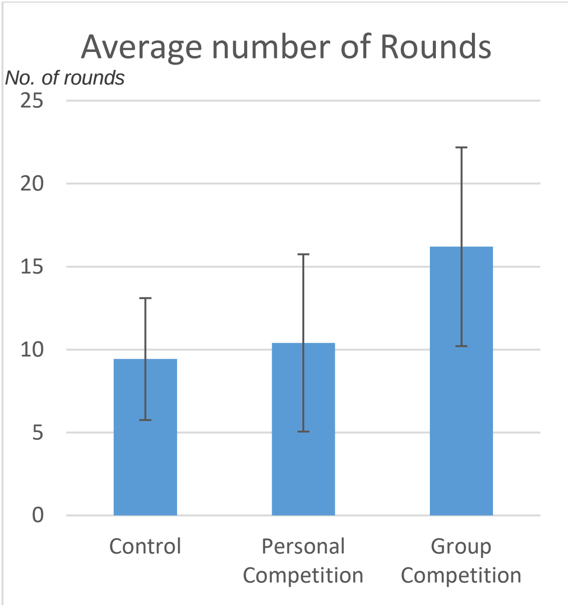

Figure 51. Average practice-assessment rounds.

The data for the following measures were acquired by asking the participants to complete a short survey at the end of the experiment. First – average participant satisfaction score (see Figure 52). Then the participants had to pick five words out of 10 that "best describes their experience using the app". They had 10 words (5 positive and 5 negative) to choose from. After the testing stage had concluded each positive word chosen by the learner has a "weight" of +1 and each negative word a "weight" of -1. After summing the five words chosen and averaging the result for each group a positive versus negative word index was calculated (see Figure 53).

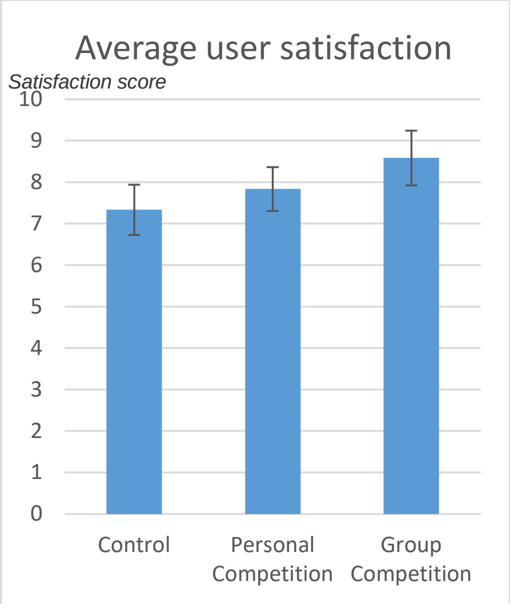

Figure 52. Average user satisfaction.

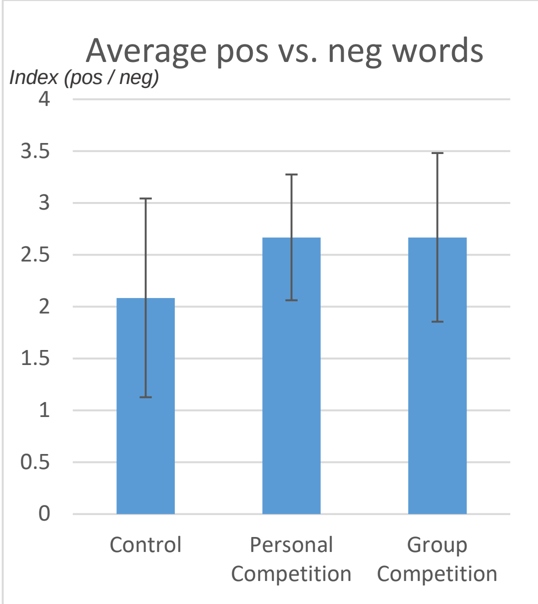

Figure 53. Average pos. vs. neg. words.

User Group	User Number											
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	#11	#12
Control	20	12	17	31	10	20	26	12	12	13	16	20
Personal Competition	17	10	18	24	13	16	20	14	18	11	19	14
Group Competition	33	16	16	36	17	28	30	10	24	11	26	28

Table 4. Average usage time

The participants' scores for all practice-assessment rounds they went through are shown in the figures 54 – 56 below. Keeping consistency with the figures presented above the data is split by these following groups:

- Group 1 – Control
- Group 2 – Personal Competition
- Group 3 – Group Competition

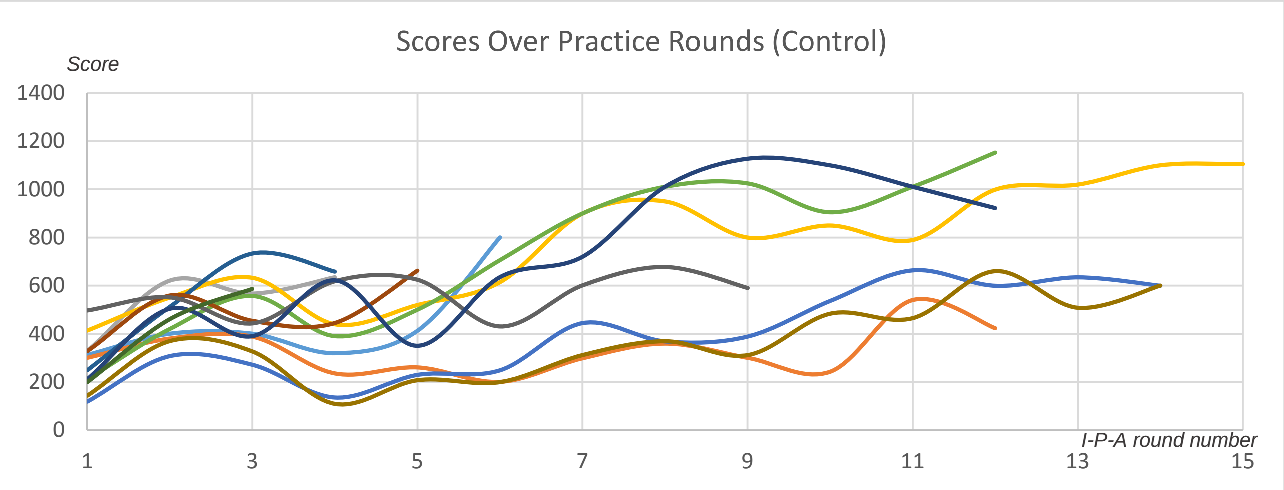

Figure 54. Participants' scores for all practice-assessment rounds – Control

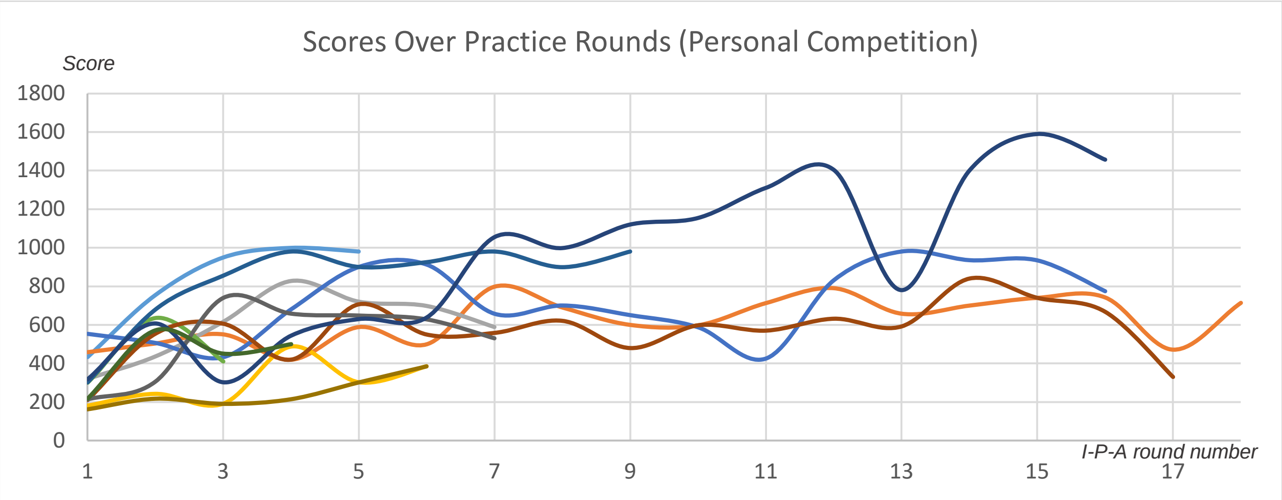

Figure 55. Participants' scores for all practice-assessment rounds – Personal Competition

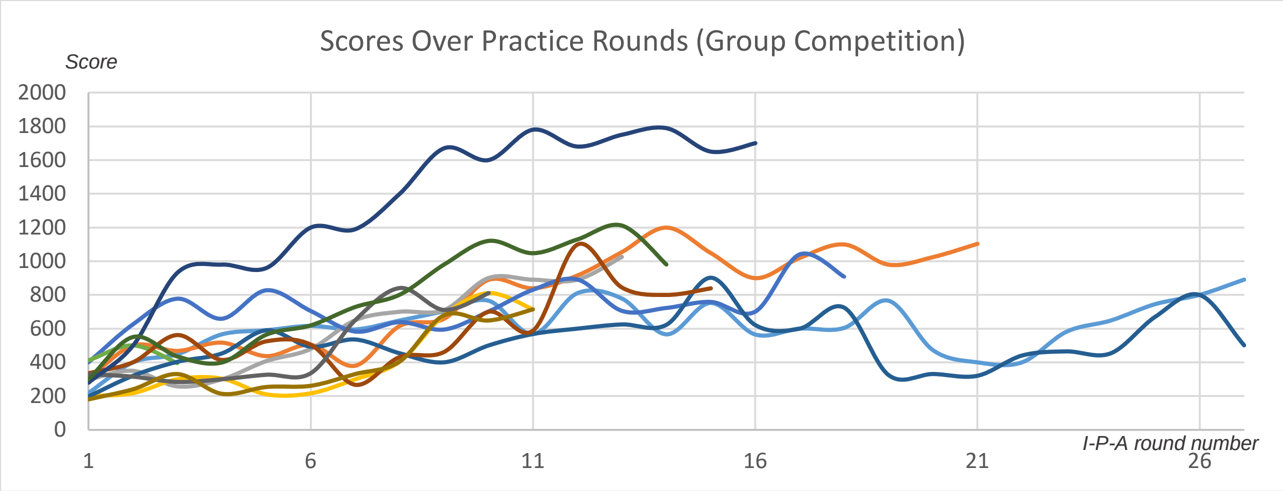

Figure 56. Participants' scores for all practice-assessment rounds – Group Competition

The qualitative data gathered in the post-test feedback survey can be represented using word clouds. The font size of each word in the six figures below is mapped to the frequency the word appeared in the free text feedback (see Figures 57 – 59) and the predefined word selection (see Figures 60 – 62).

Figure 57. Word cloud of the free text feedback ("Comments & Feedback" see Figure 47) given by participants of group 1

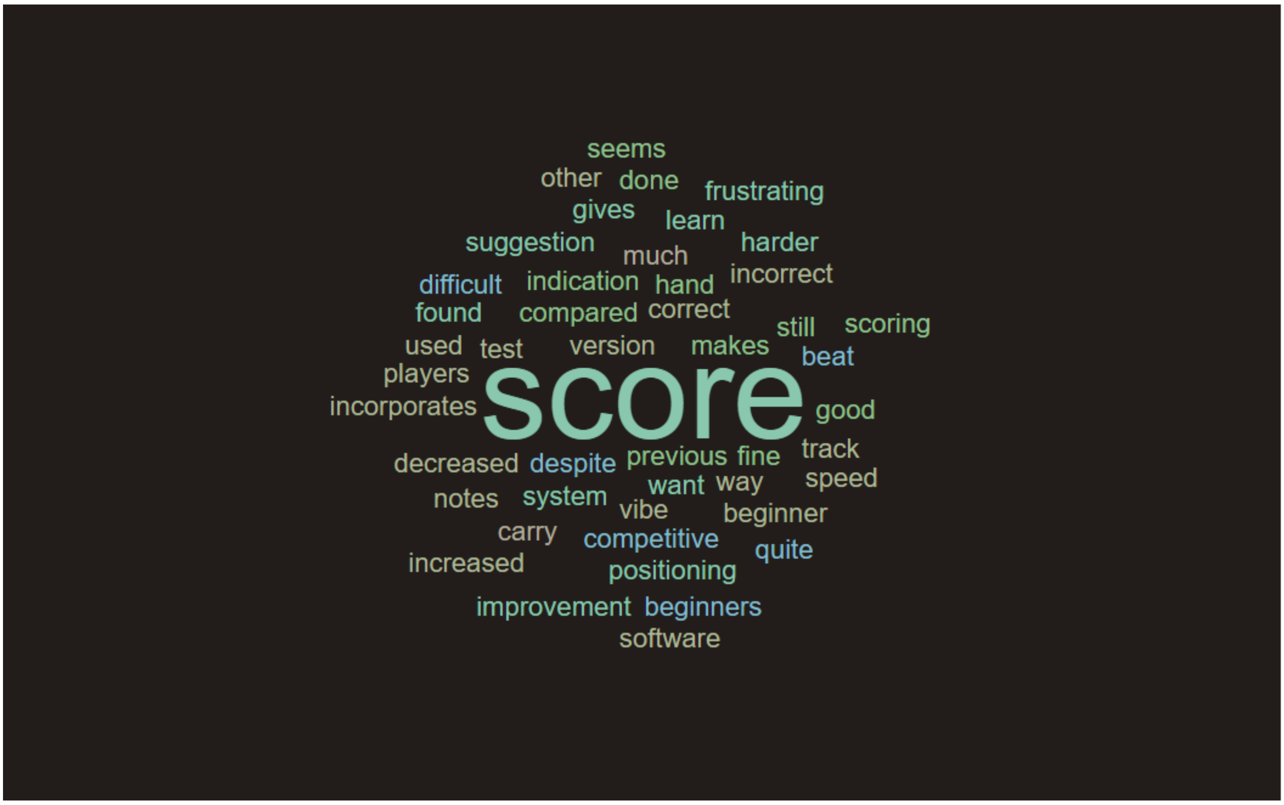

Figure 58. Word cloud of the free text feedback ("Comments & Feedback" see Figure 47) given by participants of group 2

Figure 59. Word cloud of the free text feedback ("Comments & Feedback" see Figure 47) given by participants of group 3

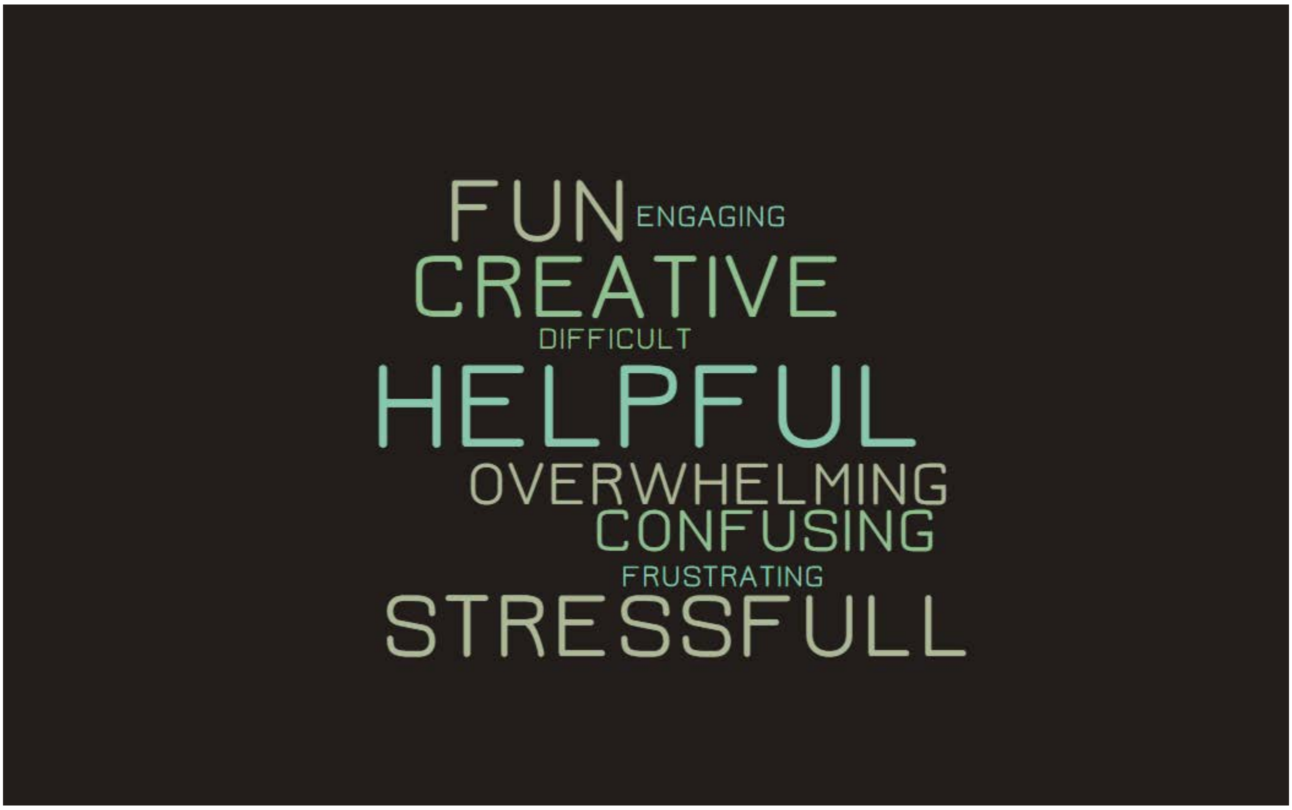

Figure 60. Word cloud of the words choice feedback ("Please select 5 words…" see Figure 47) given by participants of group 1

Figure 61. Word cloud of the words choice feedback ("Please select 5 words…" see Figure 47) given by participants of group 2

Figure 62. Word cloud of the words choice feedback ("Please select 5 words…" see Figure 47) given by participants of group 3

6.3. Result analysis – discussion

Taking a closer look at figures 48 – 53 a pattern can be observed. In all measures except for average positive vs. negative word index (see Figure 53) Variation 3 outperformed the other two. Worse performer of the three was the Control. It produced the worse results in all but one measure – average usage time (see Figure 49). Variation 2 has a marginal 7% decrease in usage time when compared against the Control. This result was not unexpected.

The choice to subject participants to the three variations in the same order every time for reasons outlined in section 6.1 had a preconceived trade off of subjecting the learners to more fatigue when transitioning to the following variation. This is clearly represented in Table 3. Participants who spent 20 minutes or more using Variation 1 (the Control) had a usage time decrease of 23% on average when comparing to Variation 2. Learners that spent less than 20 minutes on the Control, on average increased their time spent on Variation 2 by 11%. Comparing Control and Variation 3 nine out of 12 participants extended their usage time. This clearly shows that group competition increases engagement more than personal competition does.

The above result is backed up by the average number of practice-assessment rounds data (see figure 51). Variation 3 (group competition) shows an average 59% increase in the number of times the participants went through the practice-assessment cycles. There is only a marginal increase (7%) for this measure when comparing Variation 2 and Control.

The participants' scores for all practice-assessment rounds figures (see Figures 54 – 56) outline the improved app usage flow as compared to the same figures for Experiment 1 (Figures 28 – 31). Looking at figures 28 – 31 it is clearly visible how the flow is suddenly interrupted by a drastic increase in difficulty. This increase often causes the participant to get a low score and as a result negatively impact their motivation to carry on practicing.

The intermediate experiment (Experiment 2) which devised a new scoring and difficulty control system has clearly improved the overall system. A much more precise score calculation provides a more accurate estimate of how difficult the generated melody is and therefore in the end presents a more accurate score to the user. A more gradual difficulty increase (not increasing the number of strings to play on on every successful round, but rather on every third successful round) provides a much more stable flow control and avoids discouraging the learner by increasing the difficulty too much. Even at times where the difficulty does increase more than expected – and this can happen due to the nature of the randomly generated melodies – the new scoring system accounts for this increased difficulty and rewards the learner with more points (a larger score) even if only a few of the notes were played correctly. An accurate score presented to the user increased the overall quality of the system.

After looking at the qualitative data we see that the average positive vs. negative word index (see Figure 53) did not show any statistically significant results. The results did vary - both Variation 2 and Variation 3 performed better than Control, but there is a major overlap in confidence intervals.

Average top assessment score (see Figure 48) and average summative assessment score (Figure 50) both progressively increased with every variation, but this result could possibly be repeated even without making any changes to the software just by getting more familiar with the software.

A 20% increase in participants' satisfaction (Figure 52) (comparing Control and Variation 3) after the experiment, indicates that users were much more likely to be intrigued by the software when a group competition aspect was introduced to the learning process. Variation 2 also shows an increase in the user satisfaction when compared to the Control.

Finally, the free-text participant feedback fully supports the data. 7 out of 12 participants left specific feedback addressing the leader board and stating that it made them want to try again. All 12 stated the same when asked after the experiment. Most of the participants also addressed fatigue stating that the reason they weren't using Variation 3 more was because of finger pains.

Five out of the participants asked permission to use the software at a later point. Three of them logged back in within the same week and used the software again. All of those who went on to use the app again used Variation 3.

Further analysis of the qualitative data with the help of word cloud technique (where the word font-size and position on the image is mapped to its frequency of appearance within a given sample) provide some more insights (see Figures 57 – 62). Variation 2 (personal competition) received the least free text feedback (5 out of 12 participants filled in this field) therefore the words are spread out quite equally (most of the words appearing only once). The single exception is the word “score” which was mentioned by all four participants (a total of five times). This clearly indicates that the introduction of the scoring system (as the Control variation does not present a score at all) is noticed by the users.

A closer look at the variation 3 word cloud (see Figure 59) we can notice a few new words that are not seen in the responses for the first two variations – “leader board”, “try harder” and “included”. The feelings expressed by these phrases back the previously discussed theoretical gamified approach to learning principle – it makes the tutees more likely to try again and try to exceed the scores of their peers. This is again backed by the usage time (see Figure 49) and the practice-assessment rounds (see Figure 51) data.

Furthermore, this theory is backed by the following three word clouds (see Figures 60 – 62). These contain words chosen by participants from a pre-generated list. The list contained five positively and negatively perceived words. The four most common choices for the Control variation were “helpful”, “fun”, “stressful” and “creative”. After introducing the scoring system and the competitive principle in Variations 2 and 3 the most common word chosen became “engaging”. One of the main aims of gamifying the learning experience is to increase the learner’s motivation and engagement therefore “engaging” as a most commonly chosen word is a positive outcome.

The third most used word for Variation 2 however was “frustrating”, this word was also chosen by two users for Variation 3. This can indicate that the introduction of the scoring system and the competition aspect into the software does increase the stress level and may affect the learner in a negative way as well.

6.4. Conclusions

The overarching goal of this experiment was to show an increase in engagement with the software product. The design of the experiment where each participant performed tests with all three variations in a row likely influenced all of the measures especially those which we would expect to see an increase as the tutee gets more used to the software itself – top score and cumulative score. Even though a decrease in usage time was expected due to fatigue for the later tested Variation 2 and last tested Variation 3, the introduced competitive gamification principles had a greater impact than was anticipated. A 31% increase in software usage time (when compared to the Control) was observed which coupled with free-text form participant feedback discussed in the section above and a 20% increase in participant scoring of the software clearly showcases the positive effect of personal and group-based competition tested in Variation 3. The increase in both usage time and participant app scores were observed with minor confidence interval overlaps.

7. Conclusions and Further Work

Differing the type and presentation of feedback has a measurable impact on the performance and motivation of the learner. When an optimal balance of positive and negative content within feedback is achieved, anxiety is less likely to discourage learners and deter them from continuing the practice. A negative impact was observed when performance feedback was presented in a negatively perceived colour (red) and using negative language (outlining the negative aspects of the tutee's performance rather than the positive ones). The negative feedback type also received the worse overall qualitative user feedback. In the survey which each participant completed after the experiment there were direct mentions of the colour red being used too often and the text being too negative. Subjects who used positive and mixed feedback types did not make any direct references to the feedback presentation at all. This shows that the negative feedback stands out in the learners' experience (stands out negatively in the case of this study).

Mixed feedback that presented negative aspects of performance feedback first and positive ones afterwards appear to be most effective though the data was not sufficient as the confidence intervals do overlap. However, it appears that balanced and correctly formulated feedback can prevent anxiety. Csikszentmihalyi's flow, discussed in more detail in Chapter Two, was found to be an important factor influencing learners' engagement with the learning process. Figures 28 – 31 presented in chapter 4.2 show how an unbalanced software difficulty level can lead to the learner suddenly receiving a low score due to a drastically increased difficulty which leads to discouragement by failure. The frustration after the user receives an extra difficult melody that he/she is unable to successfully perform was noted by the author during the experiments.

Another factor influencing the flow balance is boredom. A common note from the participants of Experiment 1 was that the user interface is too dull and boring. The interface was designed in such a way that the main aspect which was being tested (i.e. the different feedback presentation techniques) would be the most prominent element on the screen drawing the most attention from the participants. This was supposed to increase the effect size of the different feedback types. As seen in the experiment participants' survey responses the simplistic user interface had a major drawback. It made the overall experience too dull and according to several users this was a major factor which made them stop using the app.

Introducing a more interesting and playful user interface and gamifying the learning experience by incorporating a competitive aspect such as introducing a leader board into the learning platform, demonstrated a strong positive impact on the learners' engagement in computer-based guitar tuition. Tutees were more motivated to keep practicing with a goal to reach and exceed their own achievements, as well as achievements of other individuals, using the same learning platform. Not unlike the Experiment 1 all participants were asked to leave feedback after using each of the three variations. None of the comments addressed the user interface directly. This indicates that introduction of a logo and several coloured elements to the interface made it more engaging. However Experiment 2 had fewer advanced participants as compared to the Experiment 1 which needs to be taken into account.The group competition variation outperformed the personal competition as well as the control variations in usage time. By design the variations were presented in the order Control, personal competition, group competition for the participants to gain access to new functionality (like a leader board) rather than lose functionality as they transfer from one variation to the other. Even though participants were more fatigued when using the third variation (group competition) the usage time increased as compared to both personal competition and the control variant. This appears to indicate that introducing of the leader board which gamifies the learning experience by targeting the competitive nature of the participants increased their engagement and motivation. This however cannot be stated with certainty as the confidence intervals on the usage time figure (see Figure 49) overlap. Further testing with a larger group size is needed to acquire more confident data.

More work needs to be done to further advance this research in both fields – positive / negative feedback balance and gamification to fully understand the potential benefits of both of these techniques to enhance learners experience and engagement levels. However the findings from this experiment show a positive indication for both being utilised successfully.

Improvements in the software would also enhance the learning experience. They have been identified as follows:

1. Better pitch and onset detection resulting in more accurate feedback for the learner
2. Decreasing hardware dependencies
3. Better graphical user interface that provides more a realistic virtualisation of the fret board

Further testing of textual feedback wording and visual display, manipulating more psychological effects could be carried out. Both feedback and gamification experiments could take the findings presented in this paper to further advance the hypothesis and test improved experiments on a larger test sample, which would provide more accurate results with a higher level of statistical significance.

References

1. Brusilovsky, P., Schwarz, E., & Weber, G. (1996). ELM-ART: An intelligent tutoring system on World Wide Web. Department of Psychology, University of Trier.
2. Brusilovsky, P. (1999). Adaptive and Intelligent Technologies for Web-based Education, Künstliche Intelligenz. *Special Issue on Intelligent Systems and Teleteaching.*
3. Buckholz, R. H. , Dantzig, J. A. (1976). Acoustic generation of harmonics. Ultrasonics Symposium Conference.
4. Condon, J. J & Ransom, S. M (2016). Essential Radio Astronomy. Princeton University Press. pp. 280–281
5. Crha, B., Jurcikova, T., Prudikova, M. (2010). Výzkum využití multimediálních technologií v hudební výchově. Zpráva z grantového specifi ckého výzkumu MUNI/ A/1025/2009. (Research into the Use of Multimedia Technologies in Music Education. Report from the Specific Research Project MUNI/A/1025/2009).
6. Csikszentmihalyi, M. (2014). Flow and the Foundations of Positive Psychology.
7. Dannenberg, R. B., Sanchez, M., Joseph, A., Joseph, R., Saul, R., and Capell, P. (1990). A computer-based multi-media tutor for beginning piano students. *Journal of New Music Research*, 19(2-3):155–173.
8. de-Marcos, L., García-Cabot, A., García-López, E. (2017). Towards the Social Gamification of e-Learning: a Practical Experiment. International Journal of Engineering Educadion, IJEE
9. Dempsey, J. V., Hoska, D. M. (2003). Interactive Instruction and Feedback, Chapter 5.
10. Devedzic, V. (2004). Education and the Semantic Web. *International Journal of Artificial Intelligence in Education*.
11. Dewey, C., Wakefield, J. P. (2016). Novel designs for the audio mixing interface based on data visualisation first principles. Audio Engineering Society, Convention Paper.
12. Dewey, C., Wakefield, J. P. (2017). Formal usability evaluation of audio track widget graphical representation for two-dimensional stage audio mixing interface. Audio Engineering Society, Convention Paper.
13. Dixon, D.S., Khaled, D., Nacke, L. (2011). From game design elements to gamefulness: Defining gamification. Paper presented at the Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments.
14. Felder, R. M., & Silverman, L. K. (1988). Learning and teaching styles in engineering education. Engineering Education, 78(7), 674–681.
15. Fiedler, B. (2012). Mathematical models of flageolet harmonics on stringed instruments. *Journal of Computational and Applied Mathematics*.

16. Flanagan, D. (2006). JavaScript– The Definitive Guide, Chapter 1, Fifth edition.
17. Gammons, R. W., Ippoliti, C. (2016). User-Centered Design for First-Year Library Instruction Programs.
18. Gooch, D., Vasalou, A., Benton, L., Khaled, R. (2016). Using Gamification to Motivate Students with Dyslexia. Proceedings of the 2016 CHI Conference on human factors in computing systems.
19. Gossen, T., Kotzyba, M., Nürnberger, A. (2017). Search Engine for Children: User-Centered Design. *Datenbank-Spektrum, Volume 17*, Issue 1
20. Graesser, A. C., VanLehn, K., Rosé, C. P., Jordan, P. W., Harter, D. (2001). Intelligent tutoring systems with conversational dialogue. *AI Magazine*.
21. Grigutis. L. guitMaster. (2017). Retrieved from: www.guitmaster.com
22. GuitarHero. (2017). Retrieved from: https://www.guitarhero.com/uk/en/
23. Hattie, J., Timperley, H. (2007). The Power of Feedback. *Review of Educational Research, Vol. 77*, No. 1, 81-112
24. Kraft, S., Zolzer, U. (2015). Polyphonic pitch detection by matching spectral and autocorrelation peaks. 23rd European Signal Processing Conference (EUSIPCO).
25. Mathieu, G. (2003). Virtual Music Teacher.
26. Meeker, M. (2015). Internet trends 2015 – code conference. Kleiner Perkins Caufield Byers (KPCB).
27. Meyer, K. (2016). Microsoft Desirability Toolkit Product Reaction Words. Retrieved from: https://www.nngroup.com/articles/desirability-reaction-words/
28. Moore, G. E. (1975). Progress In Digital Integrated Electronics, 5. International Electron Devices Meeting, *IEEE*, 11-13
29. Narasimhan, S.V., Veena, S. (2008). Signal Processing Principles and Implementation Revised Edition.
30. Oberst, U. (2007). The Fast Fourier Transform. *SIAM Journal on Control and Optimization, Volume 46*, Issue 2.
31. Palmer, S. E., Schloss, K. B. (2010). An ecological valence theory of human color preference. Proceedings of the National Academy of Sciences, 107, 8877–8882.
32. Percival, G., Wang, Y., Tzanetakis, G. (2007). Effective Use of Multimedia for Computer-Assisted Musical Instrument Tutoring.
33. Picco, G., Ahmad, R., Ives, B. (2001). Web-Based Virtual Learning Environments: A Research Framework and a Preliminary Assessment of Effectiveness in Basic IT Skills Training. *MIS Quarterly*. *Volume 25*, Issue 4.

34. Politis, D., Margounakis, D., Aleksić, V., Karanikas, N. (2017). From Music Gamification to the Musification of Games: A Synaesthetic Learning Pathway. *International Journal of New Technologies in Science and Engineering Vol. 4,* Issue. 1, ISSN 2349-0780
35. Rathel, J. M., Drasgow, E., Brown, W. H., Marshall, K. J. (2014). Increasing Induction-Level Teachers' Positive-to-Negative Communication Ratio and Use of Behavior-Specific Praise Through E-Mailed Performance Feedback and Its Effect on Students' Task Engagement. *Journal of Positive Behavior Interventions Vol 16*, Issue 4.
36. Rochsmith. (2017). Retrieved from: https://rocksmith.ubisoft.com/rocksmith/en-us/home/
37. Schatt, M. D. (2011). If I have time: Junior high school instrumentalists' attitudes regarding practice. *Visions of Research in Music Education.*
38. Schnotz, W., Kurschner, C. (2008). External and internal representations in the acquisition and use of knowledge: visualization effects on mental model construction. *Instructional Science 36,* 175–190.
39. Schoonderwaldt, E., Hansen, K., Askenfeld, A., (2004). IMUTUS – an interactive system for learning to play a musical instrument. *In Proceedings of the International Conference of Interactive Computer Aided Learning (ICL),* 143–150.
40. Sedlacek, M. (2010). The Multimedia Technologies Applications In Music Education. Masaryk University Faculty of Education. music teacher conference MUSICA VIVA IN SCHOLA
41. Sengpiel, E. (2005). Harmonics, Overtones, and the Fundamental Calculations of Harmonics from Fundamental Frequency. Retrieved from: http://www.sengpielaudio.com/calculator-harmonics.htm .
42. Sibelius (2017). Retrieved from: http://www.avid.com/sibelius
43. Slack, R. (1999). PEDACTICE - The Use of Multimedia in Schools. *Centre for Educational Sociology*. No. 17.
44. Sweller, J., van Merrienboer, J. J. G., Paas, F. G. W. C. (1998). Cognitive Architecture and Instructional Design. *Educational Psychology Review 10*, 251–296.
45. Smoliar, S. W. , Waterworth, J. A., Kellock, P. R. (1995). pianoforte: a system for piano education beyond notation literacy. In MULTIMEDIA '95: *Proceedings of the third ACM international conference on Multimedia*, 457–465.
46. Sullivan, G. M., Feinn, R. (2012). Using Effect Size—or Why the P Value Is Not Enough. *Journal of Graduate Medical Education*, 4(3): 279–282.
47. Synthesia. (2017). Retrieved from: http://www.synthesiagame.com/
48. Taras, M. (2008). Summative and formative assessment. *Active Learning in Higher Education, Volume 9,* Issue 2.

49. Taylor, C., Schloss, K., Palmer, S. E., Franklin, A. (2013). Color preferences in infants and adults are different. *Psychonomic Bulletin & Review, Volume 20,* Issue 5, pp 916–922
50. Tyler, R., Gagne, R., Scriven, M. (1967). Perspectives on Curriculum Evaluation,AERA Monograph Series – Curriculum Evaluation*. Scriven, 1967: 42.*
51. Thalmann, S. (2014). Adaptation criteria for the personalised delivery of learning materials: A multi-stage empirical investigation. Australasian Journal of Educational Technology, 30
52. Truong, H. M. (2016). Integrating learning styles and adaptive e-learning system: Current developments, problems and opportunities. Computers in Human Behavior 55 (2016) 1185–1193
53. Understanding JavaScript's requestAnimationFrame() method for smooth animations. (2016). Retrieved from: http://www.javascriptkit.com/javatutors/requestanimationframe.shtml
54. Using Web Workers. (2017). Retrieved from: https://developer.mozilla.org/en-US/docs/Web/API/Web_Workers_API/Using_web_workers
55. VanLehn, K. (2001). The Relative Effectiveness of Human Tutoring, Intelligent Tutoring Systems, and Other Tutoring Systems. *Educational Psychologist. Volume 46*, Issue 4.
56. Web Audio API. (2017). Retrieved from: https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API
57. Weber, G. and Möllenberg, A. (1994). ELM-PE: A knowledge-based programming environment for learning LISP. Proceedings of ED-MEDIA '94, Vancouver, Canada, pp. 557-562.
58. Werbach, K., & Hunter, D. (2012). For the win: How game thinking can revolutionize your business.
59. window.requestAnimationFrame(). (2017). Retrieved from: https://developer.mozilla.org/en-US/docs/Web/API/window/requestAnimationFrame
60. Wilson, B. G. (1996). Constructivist Learning Environments: Case Studies in Instructional Design. *Educational Technology Publications.*
61. Yousician. (2017). Retrieved from: https://yousician.com/
62. Zichermann, G., Cunningham, C. (2011). Gamification by design: Implementing game mechanics in web and mobile apps: “O’Reilly Media, Inc.”.
63. Zolzer, U. (2008). Digital Audio Signal Processing, 2nd edition.