4TB3 Final Project Proposal

Translating your pixels into a melody.

Created Using Simply Defined Languages and the Web

Prepared for: Dr. Emil Sekerinski
Group 11:
Benjamin Miller - 001416516
Prakhar Jalan - 001450321
Tony Wang - 001418056

Table of Contents:

Project Overview (Description) **2**
- What is BitTune? 2
- How does it work? 2
- Future Considerations? 3

Resources for the Project (Textbooks, manuals, software, articles, etc.) **3**
- Converting Pictures into Music Resources 3
- Backend and Web Resources 3
- Language Creation and Grammar Definition Resources 4

Division of Work **4**
- Student 1 4
- Student 2 4
- Student 3 4

Weekly Schedule **4**
- Week 1 (March 17th - 23rd): 4
- Week 2 (March 24th - March 30th): 5
- Week 3 (March 31st - April 6th): 5
- Week 4 (April 7th - April 13th): 5

Project Overview (Description)

What is BitTune?

The project that we would like to propose is a music interpretation application that translates a user drawn picture into a seamlessly integrated musical melody.

How does it work?

We provide the user with a blank grid-like canvas to draw any amount of notes (selected from a panel of note-color pairs), along with a few other (necessary) customizable functions. This includes features such as the instrument used, the specific tempo, a fixed octave (refer to Figure 1.0 below). The interface will allow the user to playback the melody derived from the drawn (pixelated) image in the browser. For the time being, a set squared region of pixels (i.e. 3 by 3) will be mapped to an eighth note (for the sake of displaying an appealing picture). The music will be transposed from the picture using a clearly defined grammar and rules which will be implemented by us and compiled into web assembly in order to provide a fast and scalable implementation.

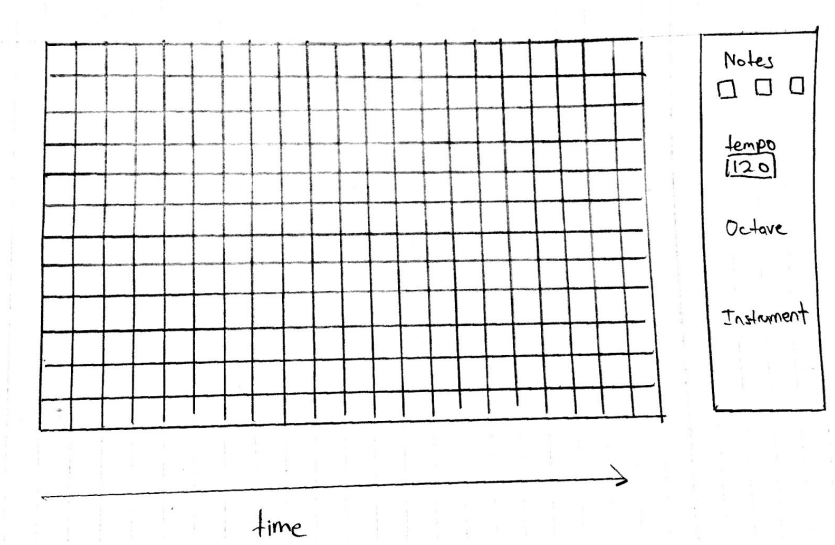

Figure 1. Sample UI - Grid Canvas with musical element(s) selection on right (Colored Notes, Tempo, Octave, Instrument)

In terms of implementation, we will be defining a drawing language which will convert simple notes into the picture scheme as defined above. Essentially, this will be an interesting way to transcribe existing music into art, and this art will be compatible with our existing music engine which transforms pictures into music. The drawing language will also have the ability to toggle

different patterns when converting the music into a picture. E.g. the user will have the ability to pick a wave pattern, or checkered pattern, which will be outputted in the resulting picture.

This project will essentially provide an intuitive and ‘creative’ way to use interface to make music without having any knowledge of playing music. Children will be able to draw simple pictures and have these pictures transformed into songs that they can then improve on and enjoy.

Future Considerations?

The music app could be expanded in many ways, and could have a variety of features added to it. For example, on top of the existing melody creating functionality, to improve what is produced in terms of sound quality, we could add in bass notes; which would create more of a progression, therefore introducing the idea of chords, and a more complex output. Going off the trail of an ‘appealing production’, different instruments can be used, or a drum pattern as a backing track could be inserted (which could have also been created using the canvas - introducing the idea of merging pictures).

Another useful feature that could be added would be allowing users to import and export music files or pictures, allowing them to share their creations with others.

Resources for the Project (Textbooks, manuals, software, articles, etc.)

The resources that we will be using to complete the project are listed below. Also, here are some useful points (made by our TA) that we will be considering right off the bat:

- There are midi related web libraries as well as a javascript (and typescript!) target for antlr. We could move the entire thing into the browser. See https://github.com/mudcube/MIDI.js/ which has a fairly similar api to the one we used in the lab.
- Another option for keeping the amount of work down on things not directly related to the compiler aspects of the project is to create a jupyter widget which would let you write the backend in python and facilitate the websocket connection to a frontend component along with simple state synchronization between a python object in the kernel (server) and a connected js object. Check out https://github.com/jupyter-widgets/widget-cookiecutter to create a simple template for that.

- Lastly check out https://github.com/antlr/antlr4/blob/master/doc/parsing-binary-files.md for defining a grammar that parses binary data (like png).

Converting Pictures into Music Resources

Working with pixels in Python (PIL): https://pythonprogramming.net/python-pixel-arrays/
Converting python into music using the midi library: https://github.com/vishnubob/python-midi

Backend and Web Resources

Python into javascript: https://github.com/pypyjs/pypyjs
Web Assembly: https://webassembly.org/getting-started/developers-guide/
Flask documentation for backend: http://flask.pocoo.org/

Language Creation and Grammar Definition Resources

Generating a parser in Python: https://tomassetti.me/parsing-in-python/
ANTLR: https://www.antlr.org/
Possible code generation in Python:
https://doc.zeroc.com/ice/3.6/language-mappings/python-mapping/client-side-slice-to-python-mapping/code-generation-in-python

Division of Work

There are several major areas in this project which are defined below:

Student 1

Converting pixels from images into usable data and creating the web user interface. This would include the drawable canvas, buttons, and melodic notes and scale template options.

Converting the usable data which is received from an input image into music by transcribing it through our defined grammar and rules while sending usable and correct data into the midi API.

Student 2

Creating the grammar for the language which converts RGB data into sound using the midi library. This student will be responsible for creating rules and definitions for what constitutes a legal and parasable picture.

Student 3

Creating a drawing language which will convert musical note inputs into a picture which is compliant with our existing rules.

Weekly Schedule

Week 1 (March 17th - 23rd):

Week one will be spent defining a grammar and language for the music. As well as beginning to build the interface which the end user will use.

Week 2 (March 24th - March 30th):

Further defining the grammar and rules for the drawing language as well as converting simple images into notes that will play music. There is no user interface at this point but the implementation on a textual level will be complete.

Week 3 (March 31st - April 6th):

Testing out our first implementation and having the ability to play and perform simple melodies in the browser with limited user interaction. We will be defining unit tests and blackbox testing the project implementation at this stage.

Week 4 (April 7th - April 13th):

Fine tuning the demo implementation, adding templates for users to build off of, creating colour palettes for users to more easily create music that sounds good, and finalizing our poster. As well as adding any last minute features to allow the user more customization options over their drawings and music.