CP2024-II HW0105

Hsin-Jui Lin

March 13, 2024

1 Taiko Music Generator (20 pts)

Have you ever heard of or played the game Taiko no Tatsujin(太鼓の達人)? Here is how to game playlink. It's a rhythm game where players hit taiko-notes by striking the center or rim of a taiko drum. You might have seen taiko arcade outside, such as the ones near our school in Gongguan. But did you know that this game is also available on various platforms like Nintendo, PS, mobile apps (RHYTHM CONNECTlink), and even on computers through simulators (Taiko-san Daijiro)? Notably, simulators allow users to play their own taiko music by creating custom chart(譜面) and music.

Figure 1: Taiko no Tatsujin game

In the simulator, the game is run by reading in .tja files, which are considered charts controlling the taiko-notes. However, the contents of tja files are not very intuitive. Therefore, this time, I want you to develop program which can convert tja files into a json file that represents the music and taiko-notes's timeline using timestamp notation. In other words, the goal is to map the taiko-notes to specific seconds on the music timeline.

1.1 tja file

(See this websitelink and download samplelink tja file) To make it easier for you to analyze tja files, I also provide a simple explanation. This is the typical structure of a tja file:

```
// ===================================================
//                  Header Area
//  Header area set the tja global and inital setting
// ===================================================
TITLE
.
.
.
// ===================================================

// ===================================================
//                  Body Area 1
//          Body area set the music chart value
// ===================================================
COURSE
// Music and Chart Setting in this course
.
#START
// Chart Contents
.
.
#END
// ===================================================

// ===================================================
//                  Body Area n
//        Body amount is be decide according to
//              how many course does it have
//                Range from 1 ~ 5 totally
// ===================================================
COURSE
// Music and Chart Setting in this course
.
#START
// Chart Contents
.
.
#END
// ===================================================
```

You only need to get the values corresponding to the following key (ignore the others):

- BPM:[float]
- OFFSET:[float]
- COURSE:[string|uint] // Easy:0, Normal:1, Hard:2, Oni:3, Edit:4
- #START

- #MEASURE beat[uint]/note[uint]
- #BPMCHANGE [float]
- #END
- xxxxxxxxx, // chart content

1.2 Output Timeline Chart json

To output as a chart json, here is the following is the json format structure:

```
{
  "data": [
        {
            "course": "taiko difficulty number [uint]",
            "chart": [
                ["taiko-note number [uint]", "timestamp [float]"],
                                ...
            ]
        },
        {
            "course": ...
            "chart": [
                ...
            ]
        },
        ...
    ]
}
```

You just need to detect these taiko-notes (Figure 3):

- DON(Red): 1
- KA(Blue): 2
- BIGDON(Big Red): 3
- BIGKA(Big Blue): 4
- Blank(Nothing): 0 (5, 6, 7, 8, 9)

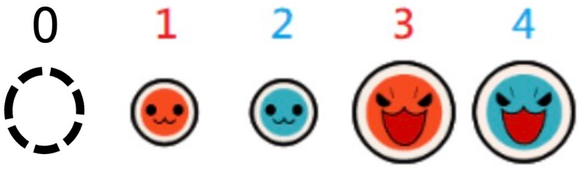


Figure 2: Taiko Notes

1.3 Convert tja file into Timeline Chart

After understanding which values to extract from the tja files and knowing their principles, in order to map taiko-notes to the music timeline, you need to understand a formula algorithm:

$$duration = \frac{60}{bpm} * \frac{beat}{Length(chart)} * \frac{4}{note}$$

You must track the BPM, beat, and note value in the chart to determine the duration. There's tja file example:

```
// 怪物.tja
BPM:170
OFFSET:-1.646
COURSE:Normal

#START
#MEASURE 4/4
1011, // duration = (60/170)*(4/4)*(4/4) ~= 0.352
```

Then it looks like Figure 4link.

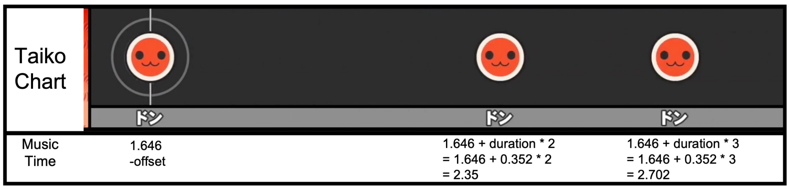

Taiko Chart
ドン
ドン
ドン
Music Time
1.646 -offset
1.646 + duration * 2 = 1.646 + 0.352 * 2 = 2.35
1.646 + duration * 3 = 1.646 + 0.352 * 3 = 2.702

Figure 3: Timeline Chart Example

And chart.json looks like:

```
{
  "data": [
        {
            "course": 1,
            "chart": [
                [1, 1.646],
                [1, 2.35],
                [1, 2.702]
            ]
        }
    ]
}
```

Before calculating duration, beware that if $Length(chart) < beat$, then you have to extend the length of chart content to $lcm(Length(chart), beat)$. But if $Length(chart) = 0$, extend 0's number of beat:

```
#MEASURE 6/4
1, -> 100000, duration = 0.4
, -> 000000, duration = 0.4
02, -> 000200, duration = 0.4
102, -> 100020, duration = 0.4
```

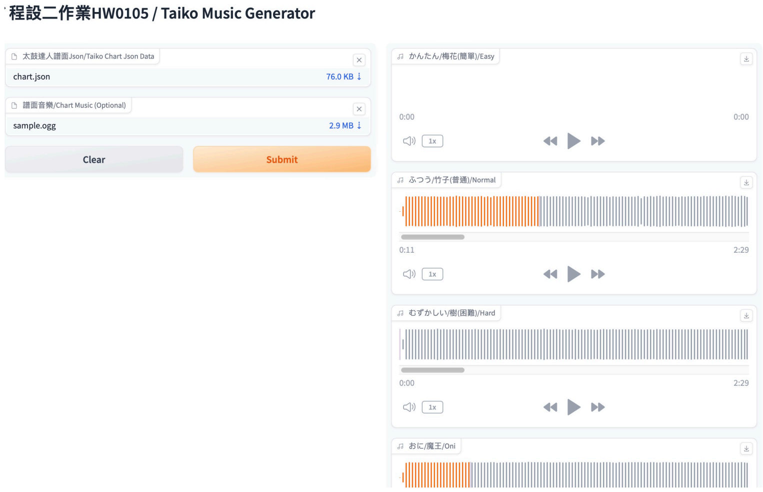
```
2401, -> 200400000100, duration = 0.2
```

So if bpm = 150 and offset = 0 currently. And that is to be:

```
"chart": [
    [1, 0],
    [2, 6],
    [1, 7.2],
    [2, 8.8],
    [2, 9.6],
    [4, 10.2],
    [1, 11.4]
]
```

1.4 Generate Taiko Music

You can go in this websitelink website and generate taiko music by inputting the chart json file and the music ogg file. You can see the Figure 5 example.



程設二作業HW0105 / Taiko Music Generator

太鼓達人譜面Json/Taiko Chart Json Data: chart.json 76.0 KB

譜面音樂/Chart Music (Optional): sample.ogg 2.9 MB

Figure 4: Taiko Music Generator Website

1.5 Example by Running Program

Again, the purpose of your program is to convert the contents of tja files into output the contents of chart json files. There's the input and output example following below:

1.5.1 Input

```
TITLE:
SUBTITLE:
BPM:170
WAVE:
OFFSET:-1.646
```

```
DEMOSTART:35.15
SONGVOL:100
SEVOL:100


COURSE:Oni
LEVEL:7
BALLOON:2
SCOREINIT:
SCOREDIFF:
#START
11211122,
,
#END


COURSE:Hard
LEVEL:6
BALLOON:2,24
SCOREINIT:
SCOREDIFF:
#START
10101212,
#END

COURSE:Normal
LEVEL:5
BALLOON:14,7,18,7
SCOREINIT:930,4870
SCOREDIFF:305

#START
1011,
10010101,
,
#END


COURSE:Easy
LEVEL:3
BALLOON:11,5,14,5
SCOREINIT:
SCOREDIFF:
#START
1,
1,
1,
12,
,
#END
```

1.5.2 Output

```
{
  "data": [
    {
      "course": 3,
      "chart": [
        [1, 1.646000],
        [1, 1.822471],
        [2, 1.998941],
        [1, 2.175412],
        [1, 2.351882],
        [1, 2.528353],
        [2, 2.704823],
        [2, 2.881294]
      ]
    },
    {
      "course": 2,
      "chart": [
        [1, 1.646000],
        [1, 1.998941],
        [1, 2.351882],
        [2, 2.528353],
        [1, 2.704823],
        [2, 2.881294]
      ]
    },
    {
      "course": 1,
      "chart": [
        [1, 1.646000],
        [1, 2.351882],
        [1, 2.704824],
        [1, 3.057765],
        [1, 3.587177],
        [1, 3.940118],
        [1, 4.293059]
      ]
    },
    {
      "course": 0,
      "chart": [
        [1, 1.646000],
        [1, 3.057765],
        [1, 4.469530],
        [1, 5.881294],
        [2, 6.587176]
      ]
    }
  ]
}
```

If there is an error issue occurred, you have to output nothing and exit the program.

For your convenience, I'll provide some fileslink for your development and testing:

- taiko.h (optional to use)

- sample tja and ogg files
- sample chart json files
- urls of people playing Taiko no Tatsujin on YouTube (for validating the taiko music that you generate)