HS 4114 - Project Report
Detecting Singer's Tonic using Background *Tanpura*

Anish Mukundlal Chaurasiya (180260007)
Anustup Bhattacharyya (22D0714)

May 2023

1 Goal

Detecting Singer's Tonic using Background *Tanpura*.

2 Introduction

What is a *tanpura*?

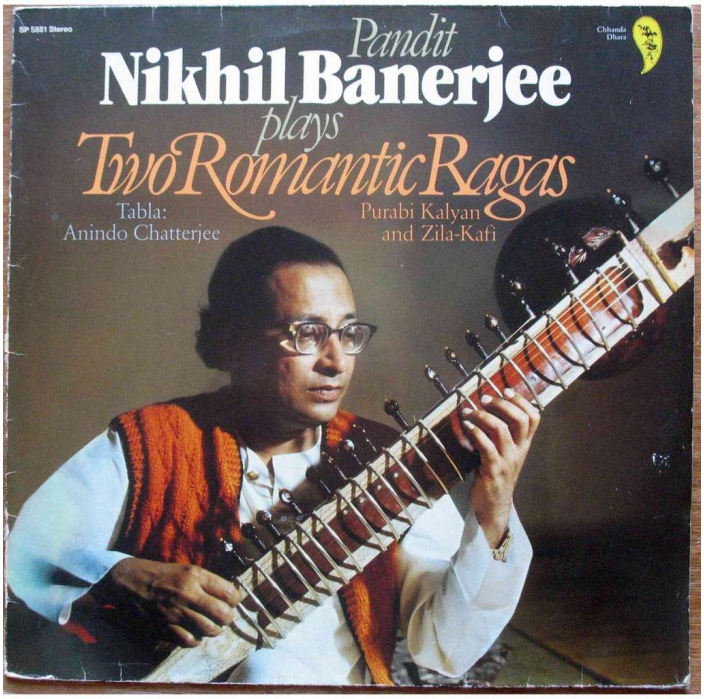
"The *tanpura* (Sanskrit: तंबूरा, romanized: Taṃbūrā), also referred to as *tambura* and *tanpuri*, is a long-necked plucked string instrument, originating in India, found in various forms in Indian music. It does not play the melody but rather supports and sustains the melody of another instrument or singer by providing a continuous harmonic bourdon or drone. A *tanpura* is not played in rhythm with the soloist or percussionist…"[1]

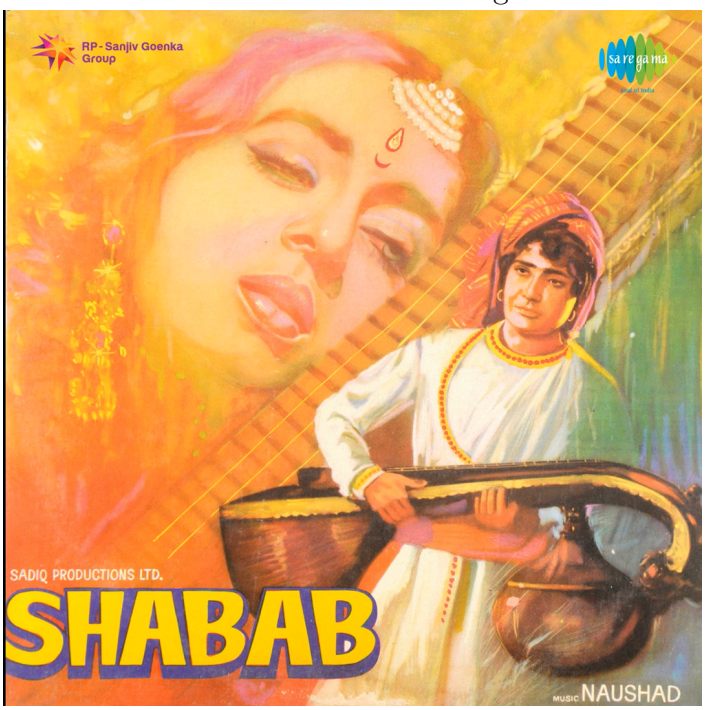
3 Dataset

We have taken samples from the standard electrical instruments and acoustic ones.

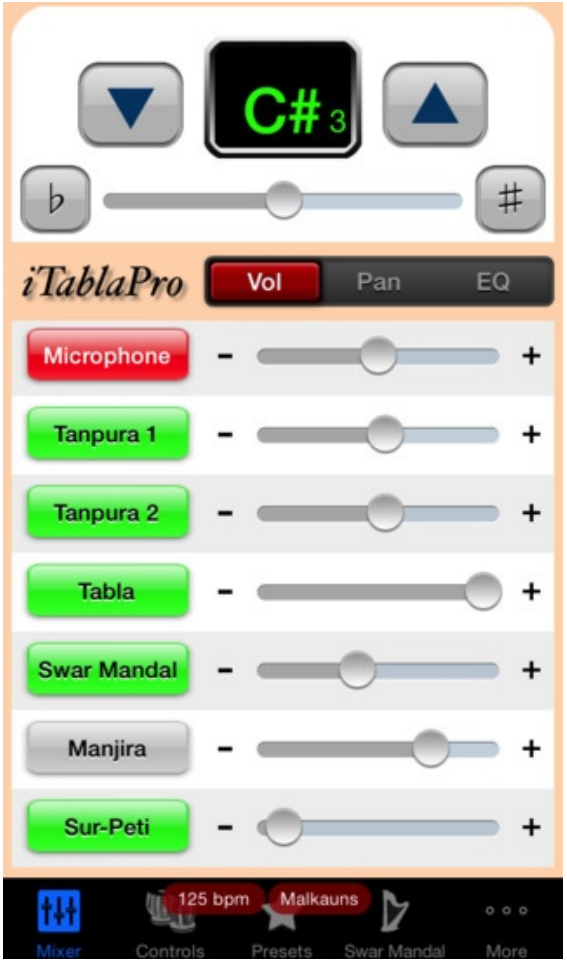
[1] https://en.wikipedia.org/wiki/tanpura

(i) Acoustic *tanpura* sampled from recordings on YouTube, one played on Sitar on Raag Zila Kafi by Nikhil Banerjee[2] and the other sung by Ustad Amir Khan in the film *Shabab* in Raag Multani[3].





(ii) iTablaPro is an electronic tabla and *tanpura* (drone) player for the iPhone and iPod Touch. But even more importantly, it is the first electronic tabla and *tanpura* that actually sounds real, ideal for Indian Classical musicians and students, making it the perfect companion for daily riyaz (practice).



[2]https://www.youtube.com/watch?v=mWygAwWkmGk
[3]https://www.youtube.com/watch?v=zLppZzk1vrQ

(iii) The Saarang Maestro Dx is a popular twin digital tambura (*tanpura*) with natural *tanpura* sound produced using modern sampler technology. It has a stereo output and can be used as single or dual *tanpura*, 4 or 5 strings, with automatic tuning of 4 notes and manual tuning of the 5th note.

4 Approach

Preprocessing First, we will preprocess the audio file to remove noise, filter out unwanted frequencies, and normalize the signal. Ideally, technically-sound modern recordings which do not need preprocessing are used.

Particulars of the *tanpura* The *tanpura* has three Sa and one other note (mostly Pa or Ni). We take the most frequently occurring note in the *tanpura* (the two Sa strings).

***tanpura*s in unison** Some of the recordings even have two *tanpura*s strung together, which makes it more complex.

Sampling To take different sample lengths of the *tanpura* recording and find the differences in the results.

Identify the drone pitch We will take the most frequently occurring note as the note pitch, which we will get through the pitch-detection algorithm.

5 Methods and Description of Code

5.1 Download the audio from the Youtube Link

```
with open('tanpura_yt_link.txt') as f:
    lines = [line.rstrip() for line in f]

for url in lines:
    yt = YouTube(url)

    video = yt.streams.filter(only_audio=True).first()

    out_file = video.download(output_path=".")

    base, ext = os.path.splitext(out_file)
    new_file = base + '.wav'
    print(new_file)
    os.rename(out_file, new_file)
```

5.2 Trim the audio into different lengths (20s, 10s, 5s, 3s, 1s, 0.5s)

```python
def trim_audio(input_path, output_path, start_sec, end_sec):
    """Trims an audio file from `start_sec` to `end_sec` and saves the result to `output_path`."""
    audio = AudioSegment.from_file(input_path)

    # Calculate the start and end positions in milliseconds
    start_pos = start_sec * 1000
    end_pos = end_sec * 1000

    # Trim the audio segment
    trimmed_audio = audio[start_pos:end_pos]

    # Save the trimmed audio segment to the output file
    trimmed_audio.export(output_path, format='wav')
```

```python
# To trim the video file and save it in trimmed_file folder
# T=20 sec, 10 sec, 5 sec, 3 sec, 1 sec, 0.5 sec
for x in os.listdir():
    if x.endswith(".wav"):
        # Prints only .wav file present in the folder
        print(x)
        trim_audio(x, './trimmed_file_20s/'+x, 5, 25)
        trim_audio(x, './trimmed_file_10s/'+x, 5, 15)
        trim_audio(x, './trimmed_file_5s/'+x, 5, 10)
        trim_audio(x, './trimmed_file_3s/'+x, 5, 8)
        trim_audio(x, './trimmed_file_1s/'+x, 5, 6)
        trim_audio(x, './trimmed_file_1s/'+x, 5, 5.5)
```

5.3 A function to read the trimmed sound clips and provide a PCD along with an F_0 using *librosa.pyin()*

```python
## reads all the sound clip from all the trimmed folder and provides a PCD along with tonic note
#saves in notes_output.csv file in respective folder
trimmed_folder_name=["./trimmed_file_20s/","./trimmed_file_10s/","./trimmed_file_5s/","./trimmed_f

for folder in trimmed_folder_name:
    Notes_Freq=[]
    Sound_Names=[]
    Mode=[]
    for x in os.listdir(folder):
        if x.endswith(".wav"):
            strip_x=x.split("-")[2].strip()
            # print(strip_x)
            Sound_Names.append(strip_x)
            tanpura, sr = librosa.load(folder+x)
            f0_singer, _, _ = librosa.pyin(tanpura,
                                           fmin=librosa.note_to_hz('C2'),
                                           fmax=librosa.note_to_hz('C6'))
            notes = librosa.hz_to_note(
                librosa.midi_to_hz(
                    librosa.hz_to_midi(f0_singer[f0_singer > 0])
                )
            )
            # print(freq(notes))
            Notes_Freq.append(freq(notes))
            Mode.append(st.mode(notes))
            # print("Mode: ",st.mode(notes))
    df=pd.DataFrame(
        {'Sound_Names': Sound_Names,
        'Notes_Freq': Notes_Freq,
        'Mode': Mode
        })
    df.to_csv(folder+"notes_output.csv", index=False)
```

5.4 To calculate the frequency of occurence of notes

```python
## To Calculate frequency of notes
def freq(lst):
        d = {}
        for i in lst:
            if d.get(i):
                d[i] += 1
            else:
                d[i] = 1
        return d
```

5.5 Convert *.mp3* to *.wav* files for ease of use

```python
## REcordings of Tanpura
from os import path
from pydub import AudioSegment
rec_folder="./recordings/"
# assign files
for x in os.listdir(rec_folder):
        if x.endswith(".mp3"):
            input_file = rec_folder+x
            output_file = rec_folder+x.split(".")[0]+".wav"
            # convert mp3 file to wav file
            sound = AudioSegment.from_mp3(input_file)
            sound.export(output_file, format="wav")
```

6 Results

6.1 20s

trimmed_file_20s_tanpura_output

	Sound_Names	Notes_Freq	Mode
0	B Scale or 7 Kattai	{'B2': 735, 'F#3': 21}	B2
1	A Scale or 6 Kattai	{'A2': 792}	A2
2	A Scale or 65 Kattai	{'A#2': 785}	A#2
3	C Scale or 1 Kattai	{'C2': 547, 'G2': 93, 'A2': 163, 'C3': 9}	C2
4	C Scale or 15 Kattai	{'C#2': 515, 'G#2': 96, 'A#2': 162, 'C#3': 60}	C#2
5	D Scale or 2 Kattai	{'D2': 486, 'A2': 122, 'B2': 136, 'D3': 87}	D2
6	D Scale or 25 Kattai	{'A#2': 102, 'D#2': 494, 'C3': 110, 'D#3': 128}	D#2
7	E Scale or 3 Kattai	{'B2': 116, 'E3': 169, 'E2': 462, 'C#3': 88}	E2
8	F Scale or 4 Kattai	{'F2': 777, 'C#3': 5, 'C3': 24}	F2
9	F Scale or 45 Kattai	{'F#2': 789, 'C#2': 3}	F#2
10	G Scale or 5 Kattai	{'G2': 792, 'D3': 21, 'D#3': 1}	G2
11	G Scale or 55 Kattai	{'G#2': 797}	G#2

6.2 10s

trimmed_file_10s_tanpura_output

	Sound_Names	Notes_Freq	Mode
0	B Scale or 7 Kattai	{'B2': 373, 'F#3': 15}	B2
1	A Scale or 6 Kattai	{'A2': 395}	A2
2	A Scale or 65 Kattai	{'A#2': 393}	A#2
3	C Scale or 1 Kattai	{'C2': 273, 'G2': 49, 'A2': 81}	C2
4	C Scale or 15 Kattai	{'C#2': 256, 'G#2': 50, 'A#2': 76, 'C#3': 34}	C#2
5	D Scale or 2 Kattai	{'D2': 250, 'A2': 64, 'B2': 71, 'D3': 30}	D2
6	D Scale or 25 Kattai	{'A#2': 50, 'D#2': 267, 'C3': 56, 'D#3': 45}	D#2
7	E Scale or 3 Kattai	{'B2': 52, 'E3': 85, 'E2': 230, 'C#3': 50}	E2
8	F Scale or 4 Kattai	{'F2': 383, 'C#3': 2, 'C3': 13}	F2
9	F Scale or 45 Kattai	{'F#2': 388, 'C#2': 3}	F#2
10	G Scale or 5 Kattai	{'G2': 395, 'D3': 11}	G2
11	G Scale or 55 Kattai	{'G#2': 399}	G#2

6.3 5s

trimmed_file_5s_tanpura_output

	Sound_Names	Notes_Freq	Mode
0	B Scale or 7 Kattai	{'B2': 191, 'F#3': 11}	B2
1	A Scale or 6 Kattai	{'A2': 193}	A2
2	A Scale or 65 Kattai	{'A#2': 202}	A#2
3	C Scale or 1 Kattai	{'C2': 131, 'G2': 27, 'A2': 47}	C2
4	C Scale or 15 Kattai	{'C#2': 157, 'G#2': 16, 'A#2': 26, 'C#3': 6}	C#2
5	D Scale or 2 Kattai	{'D2': 113, 'A2': 42, 'B2': 48}	D2
6	D Scale or 25 Kattai	{'A#2': 31, 'D#2': 132, 'C3': 21, 'D#3': 25}	D#2
7	E Scale or 3 Kattai	{'B2': 30, 'E3': 44, 'E2': 115, 'C#3': 19}	E2
8	F Scale or 4 Kattai	{'F2': 201}	F2
9	F Scale or 45 Kattai	{'F#2': 203, 'C#2': 3}	F#2
10	G Scale or 5 Kattai	{'G2': 202, 'D3': 11}	G2
11	G Scale or 55 Kattai	{'G#2': 199}	G#2

6.4 3s

trimmed_file_3s_tanpura_output

	Sound_Names	Notes_Freq	Mode
0	B Scale or 7 Kattai	{'B2': 107}	B2
1	A Scale or 6 Kattai	{'A2': 117}	A2
2	A Scale or 65 Kattai	{'A#2': 116}	A#2
3	C Scale or 1 Kattai	{'C2': 75, 'G2': 20, 'A2': 25}	C2
4	C Scale or 15 Kattai	{'C#2': 62, 'G#2': 16, 'A#2': 26, 'C#3': 23}	C#2
5	D Scale or 2 Kattai	{'D2': 78, 'A2': 21, 'B2': 24}	D2
6	D Scale or 25 Kattai	{'A#2': 12, 'D#2': 105, 'C3': 8}	D#2
7	E Scale or 3 Kattai	{'B2': 11, 'E3': 44, 'E2': 62, 'C#3': 8}	E2
8	F Scale or 4 Kattai	{'F2': 119}	F2
9	F Scale or 45 Kattai	{'F#2': 117, 'C#2': 3}	F#2
10	G Scale or 5 Kattai	{'G2': 116, 'D3': 11}	G2
11	G Scale or 55 Kattai	{'G#2': 118}	G#2

6.5 1s

trimmed_file_1s_tanpura_output

	Sound_Names	Notes_Freq	Mode
0	B Scale or 7 Kattai	{'B2': 43}	B2
1	A Scale or 6 Kattai	{'A2': 31}	A2
2	A Scale or 65 Kattai	{'A#2': 43}	A#2
3	C Scale or 1 Kattai	{'C2': 34, 'G2': 8}	C2
4	C Scale or 15 Kattai	{'C#2': 42}	C#2
5	D Scale or 2 Kattai	{'D2': 10, 'A2': 7, 'B2': 25}	B2
6	D Scale or 25 Kattai	{'A#2': 12, 'D#3': 30}	D#3
7	E Scale or 3 Kattai	{'B2': 11, 'E3': 32}	E3
8	F Scale or 4 Kattai	{'F2': 43}	F2
9	F Scale or 45 Kattai	{'F#2': 43}	F#2
10	G Scale or 5 Kattai	{'G2': 43}	G2
11	G Scale or 55 Kattai	{'G#2': 43}	G#2

6.6 0.5s

trimmed_file_0.5s_tanpura_output

	Sound_Names	Notes_Freq	Mode
0	B Scale or 7 Kattai	{'B2': 21}	B2
1	A Scale or 6 Kattai	{'A2': 10, 'E3': 6, 'F3': 4}	A2
2	A Scale or 65 Kattai	{'A#2': 21}	A#2
3	C Scale or 1 Kattai	{'C2': 21}	C2
4	C Scale or 15 Kattai	{'C#2': 20}	C#2
5	D Scale or 2 Kattai	{'D2': 10, 'A2': 7, 'B2': 3}	D2
6	D Scale or 25 Kattai	{'A#2': 12, 'D#2': 7}	A#2
7	E Scale or 3 Kattai	{'B2': 10, 'E2': 8}	B2
8	F Scale or 4 Kattai	{'F2': 21}	F2
9	F Scale or 45 Kattai	{'F#2': 21}	F#2
10	G Scale or 5 Kattai	{'G2': 21}	G2
11	G Scale or 55 Kattai	{'G#2': 21}	G#2

7 Relevance to the topics covered in class

- Pitch Detection Algorithm (*librosa.pyin()*)
- Frame-wise decomposition
- Pitch Class Distribution (PCD)

8 Summary

- The tonic of the background *tanpura* was successfully detected with 1s, 3s, 5s, 10s, and 20s sound clips.
- For 0.5 s, might gan error which was seen in one out of three different rec sample
- An optimized and scalable code has been composed, which can be further worked on to make a Python module for easy use