The Iby and Aladar Fleischman Faculty of Engineering
Tel Aviv University

Music Designed for Your Running Pace

Project Number: 2943
Names: Noa Zisbrod, Yevgeniy Frenkel
Advisor: Dr. Lior Arbel

Introduction

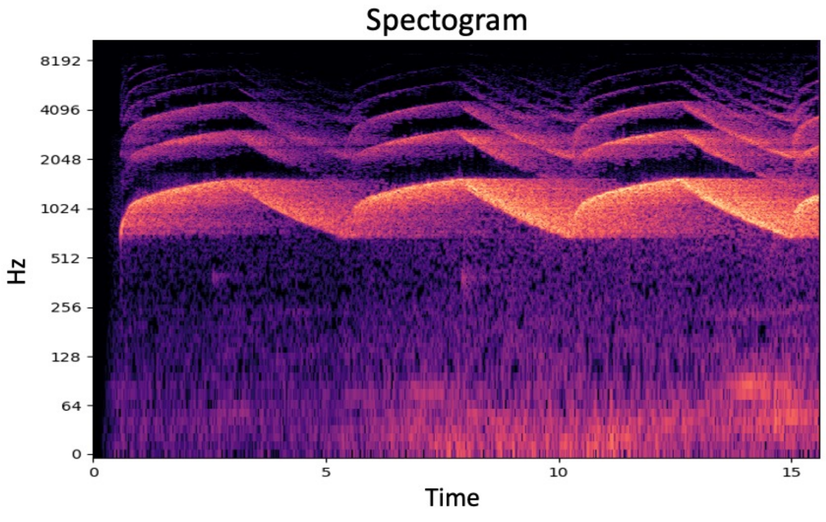
- The program creates music playlists tailored to the user's activity tempo, like running, yoga, basketball..
- Features: Playlist generation algorithm, GUI.
- Environment: Python.
- Theoretical basis: Deep understanding of data structures and algorithms.

Motivation

- Musical Beats for Healthier Feet!
- One touch: Immediately start your training activity with your favorite music tailored to match the tempo.
- Unique: The feature cannot be found in today's Music platforms.

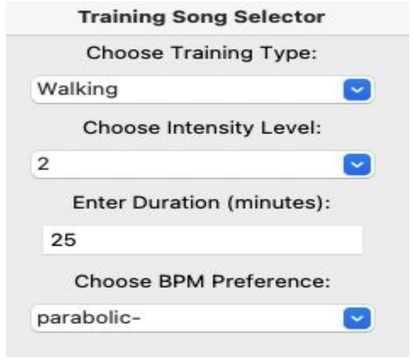
Implementation

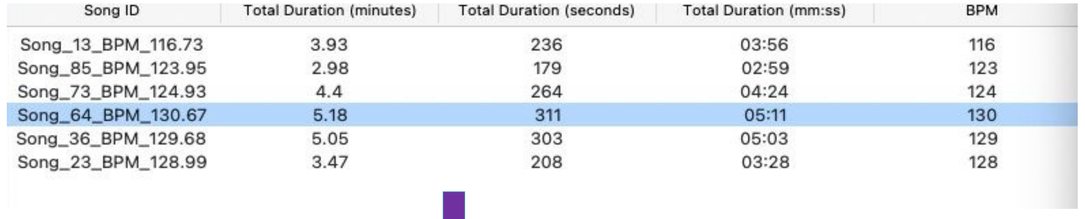
- What's behind the music?
  To start writing the algorithm we started reading about Beat Tracking – the process of moving from audio sound to signals, leading us to Spectogram - a visual representation of the spectrum of frequencies in a signal as it varies with time.
- Using Matlab, we created various Spectograms. For example, one of an alarm:



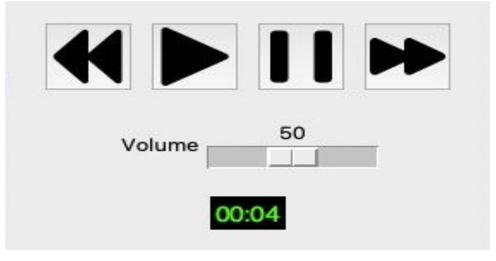
- We looked for different types of methods that the algorithm will be based on.
- Simulated DB: We created a simulated database of songs, with: Name, time duration, BPM (Beats Per Minute) range etc. It plays as the Music library of the user later.
- Using BPM, we identify how intense the song is, during training activity.
- Dynamic Programming: the algorithm customizes the playlist based on the user's training type, intensity level, and duration.
- Customize Your training : Custom training allows users to design a personalized workout with defined intervals and specific BPM values for each interval.
- Iterating: The algorithm checks different combinations of songs from the given database to create the desired playlist. Using the knapsack problem method the final playlist is created and ready to play within the desired time duration and BPM range.
- In real life: The DB represents the user's Music library on the phone or PC.
- DBs and Edge Cases: empty DB, big DB, small DB and different kinds were generated to the algorithm, checking the results each time.
- Player: Using a Python algorithm, a player runs the final playlist, ensuring user comfort.

GUI





Song ID	Total Duration (minutes)	Total Duration (seconds)	Total Duration (mm:ss)	BPM
Song_13_BPM_116.73	3.93	236	03:56	116
Song_85_BPM_123.95	2.98	179	02:59	123
Song_73_BPM_124.93	4.4	264	04:24	124
Song_64_BPM_130.67	5.18	311	05:11	130
Song_36_BPM_129.68	5.05	303	05:03	129
Song_23_BPM_128.99	3.47	208	03:28	128



Target Duration: 25 minutes
Actual Duration: 25.00 minutes
Error: 0.00 minutes

Results & Conclusions

- The algorithm handled edge cases of DBs, asking the user to pick his preferences again (e.g longer time duration).
- Complexity – **O(n·k)** ; n is max number of songs in user's DB and k is the desired time duration from the user.
- Accuracy: The algorithm includes a feature that calculates the percentage accuracy by comparing the final playlist duration to the desired length.
- The algorithm demonstrated almost 100% accuracy on normal operating conditions. When dealing with edge cases, user should re-enter his answer.