AutoDJ

Devon D'Apuzzo devondapuzzo2017@u.northwestern.edu | Victor Lalo victorlalo2017@u.northwestern.edu | Matthew Shipley matthewshiple2016@u.northwestern.edu

Goal

Automatically mix an imported batch of songs together so that each song seamlessly transitions into the next.

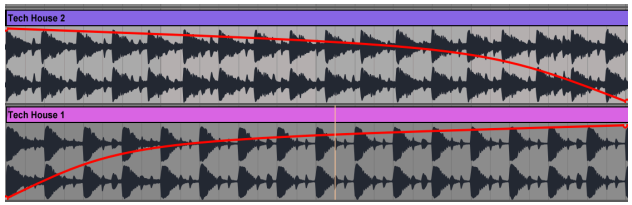
Purpose

- Allow user to playback a group of songs without breaks
- Arrange songs together in a natural order
- Seamlessly change tempo from song to song
- Match up strong beats of each song during the transitions
- Match up song entrances at strong phrase markers
- Avoid clashes in frequency during the song transitions
- Incorporate algorithm into an easy to use web-based interface

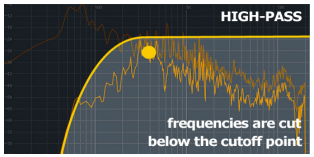
Current Alternatives

- **iTunes Crossfade:** User sets a time length, and software fades out from the previous song and into the next song during the specified time
- **Spotify Gapless Playback:** Zero gap between songs; immediately concatenates the two audio files, but no overlap between songs
- **VirtualDJ AutoMix:** Closest to our desired outcome. User adds song to playlist, and VirtualDJ beatmatches the songs and continuously changes tempo during song transitions. VirtualDJ only mixes in the user selected order and does not automatically determine the optimal order
- **Human DJ:** Dynamically selects and plays records, changing the speed and pitch manually and lining up phrases organically responding to the audience. Optimal sound, but requires a lot of equipment (turntables, mixer, amplifier).

Algorithm Implementation



Strong beats in each song are detected. Songs are assumed to be at similar BPM. The sample numbers of each strong beat are lined up over an identical 32-bar phrase. Exponential fades and high-pass filters are applied to ensure even output amplitude.



Step By Step:

1. Import playlist and read indvidual .mp3, .wav, .m4a, etc. files
2. Analyze beat onsets and index the beat sample points.
3. Begin crossfading the first two songs:
   1. Apply high-pass filter to second song to avoid clashing kick drums or bass.
   2. Apply fading function (seen above) over last 32-beat phrase of the first song and first 32-beat phrase of the second song.
4. Zero pad both song sample arrays to same length and add to create new sample array.
5. Repeat above steps using new mix as the first song until all songs become a single mix sample array.
6. Bounce back to .mp3 and encode for playback

Future Additions

1. Sorting algorithm for optimal transitions
   - Sort by increasing tempo
   - Sort by matching song keys
2. Dynamic, gradual tempo adjustment
   - Smoothens transitions and allows for wide range of BPMs
3. Advanced beat-spectrum analysis
   - Computes ideal time to begin transitioning between songs
   - Helpful for recognizing more complex phrase patterns
4. Advanced frequency analysis
   - Applies multiple band filters with adequate bandwidths for optimal frequency mixing
   - Minimizes frequency clashing
5. Real-time song addition
   - Add songs to playlist to mix in at the end of ongoing mix
6. Improve performance and memory capacity
   - Increase speed of mixing process
   - Allocate space more efficiently to increase playlist size

Conclusion

AutoDJ is adept enough to compete with crossfading functions such as those found in iTunes or Spotify, but in its current state, is not as advanced as some algorithms such as VirtualDJ.

It's frequency filtering sets it apart from other softwares as it allows the songs to mix without clashing elements such as kicks and bass.

With proper beat-spectrum analysis, AutoDJ can serve as a compositional teaching tool to highlight phrasing, dynamic tension, and frequency distribution in electronic music.

Works Cited

A Tutorial on Audio Compression – *Davis Pan*

Music Meter and Tempo Tracking from Raw Polyphonic Audio – *Aggelos Pikrakis, Iasonas Antonopoulos and Sergios Theodoridis*

Multi-Band, Digital Audio Noise Filtering – *Steven E. Boze*

Filterbank Structure and Method for Filtering and Separating an Information Signal into Different Bands – *Robert Brennan and Anthony Todd Schneider*

Northwestern University

EECS 352 – Machine Perception of Music and Audio

Bryan Pardo