Auto-mashup accompaniment

**Goal** - to create an automatic accompaniment music mashup system for live musical input

**Approach** - to develop a hybrid system which uses an existing performance following method and combines it with a mashup engine adapted for real-time applications

**Scope** - an experimental prototype to explore the creative possibilities in music interaction by searching and synchronising matching musical content in real-time

---

Predicting and matching harmonic content

**Performance follower**
- Make beat-synchronous chromagram from input audio
- Use dynamic programming to search over long and short term history
- Find best match and use next beat as best prediction of future content

http://www.eecs.qmul.ac.uk/~adams/pf/

**Mashup engine**
- Adapt offline AutoMashUpper system (ISMIR, 2013) for real-time use
- Measure compatibility between predicted chromagram and database using cosine similarity between beat-synchronous chromagram frames
- Find the best matching beat segment across the database and retrieve the corresponding audio for synchronous playback

---

Implementation

**Improvasher system architecture**
- Performance Follower: MAX patch
- Mashup engine: standalone C++ Juce application
- OSC connects the two modules

**Database of pre-analysed songs**
- Sonic Annotator with QM-VAMP plugin to extract beats and NNLS-Chroma for the chromagram
- Rubberband for time-stretching and pitch-shifting the set of candidate songs

Constraints and usage scenarios

**Real-time constraints**
- Fix input tempo to prevent the need for real-time beat-tracking and time-stretching
- Used pre-computed database of candidate songs to reduce computational burden
- Pre-compute pitch-shifted versions of candidate songs to expand scope of matches

**Modes of operation**
- Live musician input: mashup accompaniment is driven by user selected songs
- Re-use musician input: use musician’s performance as mashup data to accompany songs
- DJ mode: create mashup in real-time between existing songs