

Research Proposal

Wen, Xue

Title

Object-oriented analysis of pitched musical audio signals

Background

A major difficulty in musical signal analysis arises when we look at a multi-voiced (polyphonic) signal rather than an instrument solo. While enriching the contents of music, the use of multiple voices raises difficulty in MIR tasks such as tempo detection, pitch calculating and instrument identification. The key problem is that the basic musical properties, s. a. tempo, pitch, intensity and instrument class, usually apply to each voice in a multi-part piece. When trying to measure some property of one voice, other voices exist as noises. This tends to make the measurement inaccurate or even impossible. Thus a front-end that is able to provide some focusing on a single voice out of a musical mixture is helpful for following stages. Hypothesizing that each musical voice corresponds to a sound source (object), I call such a front-end “object-oriented” analysis.

Objective

While a full separation of sound sources does not seem feasible at the moment, I aim my work towards a system that focuses on only a few objects in the signal so that further processing on these objects are made convenient. For each object, the system outputs its time-dependent spectral characteristics. In particular, for each pitched object, the time-dependent frequency and amplitude (and phase angle?) of each of its partials is estimated. A time-variant filter may then be constructed to give a “focused” result that is minimizes the effects of other sources on the focused object.

Methodology

The method used will largely follow the idea of tracking and grouping of “sinusoid trajectories”, which has already been shown by McAulay & Quatieri as well as Brown & Cooke. While transplanting the ideas to (pitched) musical signals, some high-level knowledges can be introduced for a better result of this tracking and grouping. Such knowledges may include physical level cues such as common onset / offset / modulation, harmonic structures, as well as object level cues, i.e. parametric modelling of instruments. The physical cues form a “general” object model, while the object level cues form a target model.

A multi-resolution Fourier transformer is used for front end analysis. The spectrogram(s) is then searched for objects based on general or target models. The existence of an object is supported by its partials, indicating (1) there is enough energy at its partial frequencies, and (2) a significant portion of the partials are not masked. The search starts from the McAulay-Quateri style peak-picking. Grouping and tracking of peaks are performed at the same time to get a result that satisfies both horizontal (time-continuity) and vertical (spectral distribution) hypotheses of the object model. Pitch detection is not carried out explicitly, but rather implied within each trajectory group. A time-variant filter can then be constructed for each trajectory group to provide the desired focusing.

Suggested applications

1. Improved note annotation: conventional annotations of musical materials have always been in time-domain, e.g. what happens when. Yet it's also possible to annotate what happened when AND where in spectrum.
2. Instrument tracking: this is made possible by setting up an instrument model on which the trajectory groups are matched in the manner of an instrument recognizer.
3. Instrument enhancement / suppression: this is made possible by setting the time-variant filter to have a gain larger/smaller than unity on the target trajectory group while keeping unity gain elsewhere.
4. Front-end for music transcription: instrument recognition could be made more robust with a focused input; pitch detection is implicitly involved in the trajectories; melody detection is more meaningful when it is focused on one object, etc.

Problems to solve: first stage

1. Time-frequency trade off: what to do to detect all significant peaks regardless of frequency?
2. Grouping criteria: what is, and how to get, a satisfactory result of tracking and grouping?
3. Masked partials: what to do if some partials are buried in noise (masked)?
4. Overlapping partials: what to do if partials of groups meet?