HARMONICALLY INFORMED MULTI-PITCH TRACKING

Zhiyao Duan
Northwestern University
Evanston, IL, USA
zhiyaoduan00@gmail.com

Jinyu Han
Northwestern University
Evanston, IL, USA
jinyuhan@gmail.com

Bryan Pardo
Northwestern University
Evanston, IL USA
pardo@northwestern.edu

ABSTRACT

This paper presents a novel system for multi-pitch tracking, i.e. estimate the pitch trajectory of each monophonic source in a mixture of harmonic sounds. The system consists of two stages: multi-pitch estimation and pitch trajectory formation. In the first stage, we propose a new approach based on modeling spectral peaks and non-peak regions to estimate pitches and polyphony in each single frame. In the second stage, we view the pitch trajectory formation problem as a constrained clustering problem of pitch estimates in all the frames. Constraints are imposed on some pairs of pitch estimates, according to time and frequency proximity. In clustering, harmonic structure is employed as the feature. The proposed system is tested on 10 recorded four-part J. S. Bach chorales. Both multi-pitch estimation and tracking results are very promising. In addition, for multi-pitch estimation, the proposed system is shown to outperform a state-of-the-art multi-pitch estimation approach.

1. INTRODUCTION

Multi-pitch analysis is a fundamental research problems in music information retrieval (MIR). Pitch analysis results can provide helpful information for many other applications, such as automatic music transcription, audio source separation, content-based music search, etc. This task, however, remains challenging and existing methods do not match human ability in either accuracy or robustness.

Due to the complexity of multi-pitch analysis, researchers try to break it into different subproblems. *Multi-pitch Estimation (MPE)* [1, 2] usually refers to estimating pitches and the number of concurrent pitches (polyphony) in each single time frame. Based on MPE, *Note Formation* [3, 4] forms notes using pitch estimates in adjacent frames.

These two subproblems are important, however, they do not constitute the whole multi-pitch analysis problem. In a piece of polyphonic music consisting of several sound sources (usually different instruments), estimating pitches and finding the pitch trajectory for each underlying source, is a more advanced problem, which we call *Multi-pitch Tracking (MPT)*.

Multi-pitch tracking is similar to stream organization in Auditory Scene Analysis (ASA) [5], which is described as a grouping process of distinct acoustic events into a single perceptual entity, i.e. a stream. In [5], Bregman describes two main grouping processes in stream organization. *Simultaneous grouping* is to "integrate components that occur at the same time in different parts of the spectrum". *Sequential grouping* is to "put together events that follow one another in time". It is noted that MPE involves only simultaneous grouping, where harmonics of a sound source are integrated into a single pitch. Based on MPE, note formation also involves sequential grouping, where proximate pitches are integrated in a small time scale into notes. MPT also involves sequential grouping, but in a much larger time scale, i.e. the whole music piece.

Multi-pitch tracking is closely related to monaural source separation, which in fact estimates more information, i.e. the timbre of each source. A number of source separation systems [6–8] are built on multi-pitch estimation results. Other methods [9] usually train prior source models from solo excerpts. Those methods [10, 11] not relying on pitch estimation results nor prior source models utilize the same amount of information as MPT, but they are only tested on mixtures of two or three sounds.

Multi-pitch tracking is a difficult problem. Even for humans, only highly trained musicians have the ability to track concurrent pitch trajectories in listening. Therefore, some researchers proposed to only detect the main melody line and bass line of a polyphonic music [12, 13].

To our knowledge, few systems address the multi-pitch tracking problem. Kameoka et al. [14] proposed a multi-pitch analyzer based on harmonic temporal structured clustering that jointly estimates pitch, intensity, onset and duration of each underlying source. Chang et al. [15] presented a multi-pitch tracking system that and tracks pitches into note contours using Hidden Markov Models. Although these methods track concurrent pitch trajectories of multiple sound sources, they are only evaluated in the multi-pitch estimation and note formation level [14, 15]. Lagrange and Tzanetakis [16] proposed a sound source separation method that streams spectral peaks. It was not, however, evaluated on pitch tracking performance [16].

In this paper, we propose a system to address the MPT problem. Our system differs from previous systems in several ways. We start with multi-pitch estimation, where we

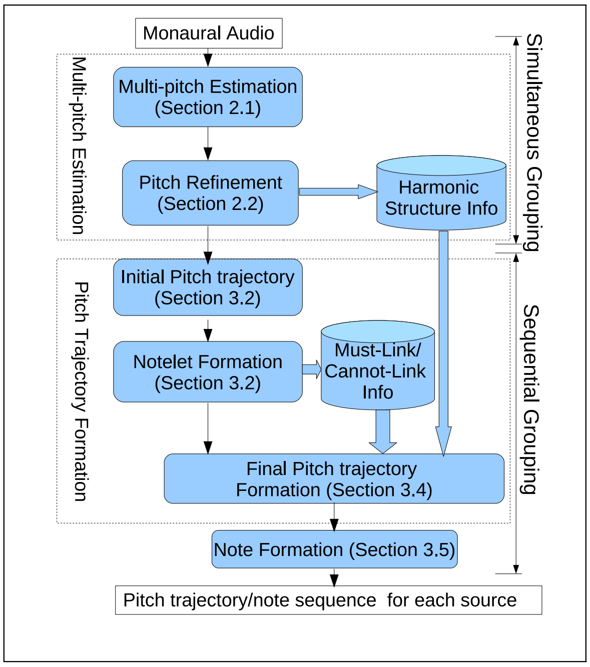

Figure 1. System overview.

propose a new method that models spectral peaks and non-peak regions. Given pitch estimates for individual frames, we cast pitch trajectory formation as a constrained clustering problem, where each cluster corresponds to a trajectory. Harmonic structure is used as the feature of each pitch in clustering. Finally, note formation happens after pitch trajectories are formed, instead of forming notes and then placing them in streams, as previous systems have done.

Figure 1 shows the system structure. We address the MPT problem in two stages. The first stage is *multi-pitch estimation* (Section 2), where pitches and polyphony in each frame are estimated, and then refined using estimates in neighboring frames. The second stage is *pitch trajectory formation* (Section 3). Initial pitch trajectories are formed by grouping pitch estimates across frames according to pitch height. Within each initial trajectory, pitch estimates that are close in frequency and contiguous in time are grouped to form *notelets*. Final pitch trajectories are obtained through constrained clustering of pitch estimates, where must-link constraints are imposed on pitch pairs in each notelet and cannot-link constraints are imposed on pitch pairs of concurrent notelets. From the view of Auditory Scene Analysis, the first stage is simultaneous grouping and the second stage is sequential grouping.

2. MULTI-PITCH ESTIMATION

2.1 Multi-pitch Estimation In A Single Frame

In multi-pitch estimation, we break the music audio into 46 ms frames with a 10 ms hop size. We estimate polyphony and pitches in each time frame, but do not group estimates across adjacent frames into notes or trajectories. We view this problem (given polyphony N) as a Maximum Likelihood parameter estimation problem in the frequency domain. The parameters to be estimated are the set of pitches $\boldsymbol{\theta} = \{F_0^1, \cdots, F_0^N\}$. The observation is the spectrum, which is represented as spectral *peaks* and *non-peak regions*. Peaks are detected using the peak detection algorithm in [10]. The non-peak region is defined as the set of frequencies falling more than a quarter-tone from any observed peak.

For harmonic sounds, peaks typically appear only near integer multiples of F0s. We try to find the set of F0s with harmonics that maximize the probability of the occurrence of observed peaks, and minimize the probability that they have harmonics in non-peak regions.

Thus, the likelihood function can be defined as follows:

$$\mathcal{L}(\boldsymbol{\theta}) = \mathcal{L}_{\text{peak}}(\boldsymbol{\theta}) \cdot \mathcal{L}_{\text{non-peak region}}(\boldsymbol{\theta}) \quad (1)$$

To model $\mathcal{L}_{\text{peak}}(\boldsymbol{\theta})$, each detected peak k is represented by its frequency f_k and amplitude a_k. We assume conditional independence between peaks, given a set of F0s. In addition, we consider the probability that a peak is *normal* (caused by some harmonic) or *spurious* (caused by other reasons). Then the peak likelihood is defined as

$$\mathcal{L}_{\text{peak}}(\boldsymbol{\theta}) = \prod_{k=1}^{K} p\left(f_k, a_k|\boldsymbol{\theta}\right) \quad (2)$$

$$= \prod_{k=1}^{K} \sum_{s_k} p\left(f_k, a_k|s_k, \boldsymbol{\theta}\right) P\left(s_k|\boldsymbol{\theta}\right) \quad (3)$$

where K is the number of detected peaks; s_k is the binary variable to indicate whether the k-th peak is normal ($s_k = 0$) or spurious ($s_k = 1$).

In modeling $p\left(f_k, a_k|s_k = 0, \boldsymbol{\theta}\right)$, we notice that a normal peak may be generated by several F0s when their harmonics overlap at the peak position. In this case, however, the probability is conditioned on multiple F0s, which leads to a combinatorial problem in training we wish to avoid. Instead, we adopt the binary masking assumption [17], i.e. each peak is generated by only one F0, the one having the largest likelihood to generate the peak. Thus $p\left(f_k, a_k|s_k = 0, \boldsymbol{\theta}\right)$ is approximated by $\max_{F0 \in \boldsymbol{\theta}} p(f_k, a_k|F_0)$. For each $F_0 \in \boldsymbol{\theta}$, a harmonic number h_k is calculated as the nearest harmonic position of F_0 from f_k. Then,

$$p\left(f_k, a_k|F_0\right) = p\left(f_k|F_0\right) p\left(a_k|f_k, F_0\right) \quad (4)$$

$$= p\left(d_k|F_0\right) p\left(a_k|f_k, h_k\right) \quad (5)$$

where $d_k = f_k - F_0 - 12\log_2 h_k$, is the frequency deviation of the k-th peak from its corresponding harmonic position in the logarithmic frequency domain.

In modeling $p\left(f_k, a_k|s_k = 1, \boldsymbol{\theta}\right)$, $\boldsymbol{\theta}$ can be ignored, since spurious peaks are assumed to be unrelated to F0s:

$$p\left(f_k, a_k|s_k = 1, \boldsymbol{\theta}\right) = p\left(f_k, a_k|s_k = 1\right) \quad (6)$$

To model $\mathcal{L}_{\text{non-peak region}}(\boldsymbol{\theta})$, it is noted that instead of telling us where F0s or their ideal harmonics should be, the non-peak regions tell us where they should not be. A good set of F0s would predict as few harmonics as possible in the non-peak regions. Therefore, we define the non-peak region likelihood in terms of the probability of

From polyphonic (random chord) data	
$P(s_k\|\boldsymbol{\theta})$	approximated with $P(s_k)$, then learned as the proportion of spurious peaks
$p(d_k\|F_0)$	approximated with $p(d_k)$, then learned as a GMM from normal peaks
$p(a_k\|f_k, h_k)$	learned non-parametrically from normal peaks
$p(f_k, a_k\|s_k = 1)$	learned as a 2-D single Gaussian from spurious peaks
From monophonic (individual note) data	
$P(e_h = 1\|F_0)$	learned non-parametrically from normal peaks

Table 1. Model parameters that are learned from the training data.

not detecting any harmonic in the non-peak regions, given an assumed set of F0s. Again, with the conditional independence assumption of non-peak regions, the likelihood can be defined as:

$$\mathcal{L}_{\text{non-peak region}}(\boldsymbol{\theta}) = \prod_{F_0 \in \boldsymbol{\theta}} \prod_{\substack{hF_0 \in \mathcal{F}_{\text{np}} \\ h \in 1..H}} 1 - P\left(e_h = 1|F_0\right) \quad (7)$$

where N is the polyphony; e_h is a binary variable that indicates whether the h-th harmonic of F_0 is detected; $\mathcal{F}_{\text{np}}$ is the set of frequencies in the non-peak regions; H is the largest harmonic number we consider.

A harmonic not being detected in the non-peak region is because the corresponding peak in the source signal was weak or not present (e.g. even harmonics of clarinet). Therefore, this probability can be learned from monophonic training data, i.e. the monophonic notes used to generate the polyphonic training data.

The above probabilities are learned in monophonic or polyphonic training data, as described in Table 1. The monophonic training data are monophonic notes selected from the University of Iowa data set [1]. The polyphonic training data are randomly mixed chords of polyphony from 2 to 6, generated by mixing the above mentioned monophonic notes. Spectral peaks and non-peak regions are detected and collected in all the frames of the training data. Ground-truth F0s are obtained by YIN [18], a single pitch detection algorithm, on each note prior to mixing. The frequency deviation of each peak from the nearest harmonic of any ground-truth F0 is calculated. If the deviation is less than a quarter-tone, the peak is labeled normal, otherwise spurious. The "a quarter-tone" threshold is used here was selected in accordance with the standard tolerance used in measuring correctness of F0 estimation.

So far, given a set of F0s $\boldsymbol{\theta}$, its likelihood $\mathcal{L}(\boldsymbol{\theta})$ can be calculated in Eq. (1). The search space of the maximum likelihood solution $\hat{\boldsymbol{\theta}}$, however, is very large. We constrain this problem with a greedy search strategy. We start from an empty set $\hat{\boldsymbol{\theta}}^0$. In each iteration, we add an F0 estimate to $\hat{\boldsymbol{\theta}}^n$ to get a new set $\hat{\boldsymbol{\theta}}^{n+1}$ that gets maximum likelihood among all the possible values of the newly added F0. This iteration terminates when $n = N$, the given polyphony.

If the polyphony is not given, we need to decide when to terminate. Since $\mathcal{L}(\hat{\boldsymbol{\theta}}^n)$ increases with n, we propose a simple threshold-based method. The minimum number of F0s that achieves a likelihood greater than the threshold is returned as the polyphony estimate:

$$\begin{aligned} N = &\min_{1 \le n \le M} n, \\ s.t. \quad &\Delta(n) \ge T \cdot \Delta(M) \end{aligned} \quad (8)$$

where $\Delta(n) = \mathcal{L}(\hat{\boldsymbol{\theta}}^n) - \mathcal{L}(\hat{\boldsymbol{\theta}}^1)$ is the maximum increase of likelihood that could be achieved when the polyphony is set to be n. M is the maximum allowed polyphony which is set to 9 in all experiments; T is a learned threshold which is set to 0.88 (a value chosen by a machine learner using the monophonic and polyphonic training data). This polyphony estimation method works well on a large data set containing both music pieces and block musical chords with polyphony from 1 to 6.

2.2 Refine Pitch Estimates Using Neighboring Frames

There are often insertion, deletion and substitution errors in the multi-pitch estimation of a single frame. We propose a refinement method using estimates in neighboring frames: For each frame t, we build a weighted histogram in the frequency domain, where each bin corresponds to a semitone in the pitch range. Then, a triangular weighting function centered at t is imposed on a neighborhood of t, whose radius is r frames. The refined polyphony estimate N is calculated as the weighted average of polyphony estimates in all the frames in this neighborhood. The N bins with the highest histogram values are selected to reconstruct refined pitch estimates. For each of these bins, if there is an original pitch estimate in frame t that falls inside this bin, the original pitch estimate is used as the refined pitch estimate. Otherwise, the refined pitch estimate is calculated as the weighted average frequency of all the pitch estimates in this neighborhood that fall inside this bin. In our system, the radius r is set to 9 frames. This method removed a number of inconsistent estimation errors.

3. PITCH TRAJECTORY FORMATION

Given pitch estimates in all frames, we view pitch trajectory formation as a constrained clustering problem, where each *pitch trajectory* corresponds to a cluster.

3.1 Constrained Clustering

Constrained clustering [19,20] is a class of semi-supervised learning algorithms that make use of domain knowledge during clustering. Constraints can be imposed on different levels, where the instance level is the simplest and most common one. In instance-level constraints, there are two

[1] http://theremin.music.uiowa.edu/

basic forms: *must-link* and *cannot-link*. A must-link constraint specifies that two instances should be assigned to the same cluster. A cannot-link constraint specifies that two instances should not be assigned to the same cluster.

For our pitch trajectory formation problem, we adopt these two constraints. A must-link is imposed between two similar pitches in adjacent frames, since they probably belong to the same trajectory. A cannot-link is imposed between two pitches within a frame, due to our assumption that sources are monophonic. We then formulate the clustering problem to minimize the intra-class distance J, as the K-means algorithm does:

$$J = \sum_{k=1}^{K} \sum_{x_i \in T_k} \|\mathbf{x}_i - \mathbf{c}_k\|^2 \quad (9)$$

where K is the number of pitch trajectories; $\mathbf{x}_i$ is a feature vector in trajectory T_k and $\mathbf{c}_k$ is the mean feature vector in trajectory T_k; $\|\cdot\|$ denotes the Euclidean distance.

Wagstaff et al. [19] proposed a constrained K-means clustering algorithm, which iteratively changes the cluster labels of all points, without violating any constraint. For our multi-pitch tracking problem, however, this algorithm does not work. The reason is that almost every pitch estimate has must-links or cannot-links to other points, so it is almost impossible to change a pitch's label without violating any constraint. In addition, Davidson and Ravi [20] proved that finding a feasible solution, i.e. a label assignment without violating any constraint, of a clustering problem containing cannot-link constraints is NP-complete.

We now propose an iterative greedy algorithm that finds a low-cost (in terms of Eq. (9)) assignment of pitches to trajectories within a reasonable time.

3.2 Initial Pitch Trajectory Formation

Although the general problem is NP-complete, we can remove some constraints to make it tractable. A trivial example is to remove all constraints, where random label assignment can be a solution.

Instead of random assignment, we utilize the intrinsic structure of polyphonic music to assign initial labels. Note that pitch trajectories (e.g. melody and baseline) are roughly ordered in pitch, although sometimes they interweave. Since there are at most K pitches in each frame, we sort them from high to low and assign labels from 1 to K.

Then must-links are imposed on similar pitches that are in adjacent frames *and* have the same initial trajectory label. The maximal must-link difference between pitches in adjacent frames is set to 0.3 semitones (30 cents). Pitches connected by must-links form a short trajectory, which we call a *notelet*, since it is supposed to be some part of a note. Once notelets are formed, cannot-links are imposed between all pitches in two notelets that overlap more than 30ms. We say that two such notelets are in a *cannot-link relation*. We allow the 30ms overlap within a melodic line as it may be reverberation or ringing of a string. We chose conservative values for these parameters to ensure that they are reasonable for common real-world scenarios.

This results in an initial set of track assignments that is reasonably correct. By assigning trajectory labels on a frame-by-frame basis, assuming a fixed small number of trajectories, and only forming notelets from within sets of adjacent pitch estimates of the same trajectory, we greatly reduce the space searched. This provides a reasonable starting point and bypasses the NP-completeness problem.

Before showing how we improve on this initial solution by minimizing Eq. (9), we first explain the feature we use.

3.3 Harmonic Structure

Feature vectors in Eq. (9) should have the property that they are close in the same trajectory and far in different trajectories. Pitch height is not suitable because the underlying pitch estimates may show octave errors and the melodic lines overlap in range. Harmonic structure has been proven to be a good choice [10, 11] for music played by harmonic instruments, which is defined as the vector of relative amplitudes of harmonics of a pitch. Harmonic structures of the same instrument are similar, even if their pitches and loudness are different. On the other hand, different instruments usually have very different harmonic structures [10].

We calculate harmonic structure as follows: First, harmonics of each pitch are found from spectral peaks. For overlapping harmonics of different pitches, the peak likelihood in Eq. (4) is used to distribute energy to each pitch. Harmonic structures are then normalized to have the same total energy. The first fifty harmonics are used here.

3.4 Final Pitch Trajectory Formation

We start from the initial set of pitch trajectories created in Section 3.2, where pitches are assigned trajectory labels based on pitch height, and then placed into notelets based on time and pitch proximity. All pitch estimates comprising a notelet share the same trajectory label. All notelets that share a label form a *trajectory*. We now consider reassigning notelets to different trajectories to minimize the cost function in Eq. (9).

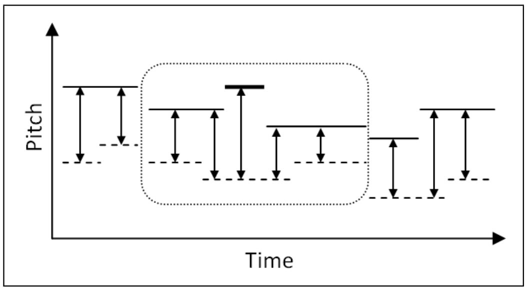

Figure 2. Illustration of a swap-set (the rounded rectangle) for a notelet (the bold solid line). Solid and dashed lines represent notelets in trajectory T_k and T_l, respectively. Cannot-link relations are indicated by arrows.

Suppose we want to change the trajectory of a notelet $\mathfrak{n}$ from T_k to T_l. We cannot do this in isolation, since there may be a notelet in T_l that overlaps $\mathfrak{n}$ and we assume monophonic pitch trajectories. We could simply swap the trajectories for two overlapping notelets. This, however may

For each notelet $\mathfrak{n}$, with trajectory T_k J_0 = cost of current trajectory assignments (Eq. (9)) J_{best} = J_0 **For** each trajectory T_l, such that $T_l \neq T_k$ Find the swap-set between T_l and T_k containing $\mathfrak{n}$ J = (Eq. (9)) if we swap every notelet in the swap-set **If** $J < J_{best}$, $J_{best} = J$, **End** **End** **If** $J_{best} < J_0$, Perform swap that produces J_{best}, **End** **End**

Table 2. The pitch trajectory formation algorithm.

cause a chain reaction, since the swap may cause new overlaps within a trajectory. Instead, we select two trajectories and pick a notelet $\mathfrak{n}$ from one of the trajectories. We then find all notelets in these two trajectories, T_k and T_l that connect to $\mathfrak{n}$ via a path of cannot-link relations (defined in Section 3.2). We call this the *swap-set*, as illustrated in Figure 2. These are the notelets affected by a potential trajectory swap between two notelets. All notelets in a swap set are swapped together, rather than individually, and the new trajectory are evaluated with the cost function in Eq. (9). Table 2 describes the process we use to swap trajectories for notelets until the trajectories of all notelets reach fixed points. In our experiment, this usually takes 3 to 4 rounds (where a round is a traversal of all notelets).

3.5 Note Formation

After pitch trajectories are formed, we form notes in each trajectory from the notelets. Two notelets are considered to be in the same note if the time gap between them is less than 100ms and their frequency difference is less than 0.3 semitone. Then the pitches in the gap are reconstructed using the average frequency of the note. Notes of length less than 100ms are considered spurious and removed. The 0.3 semitone parameter is the same as the one in imposing must-links. The 100ms parameter is set without tuning to adapt to the tempo and note lengths of the test music.

4. EXPERIMENT

The proposed system was tested on 10 real music performances, totaling 330 seconds of audio. Each performance was of a four-part Bach chorale, performed by a quartet of instruments: violin (Track 1), clarinet (Track 2), tenor saxophone (Track 3) and bassoon (Track 4). Each musician's part was recorded in isolation at 44.1 kHz, while the musician listened to the others through headphones. Audio mixtures were created by summing the four tracks. In testing, each piece was broken into 46 ms frames with center times spaced every 10 ms.

The ground-truth pitch trajectories of each testing piece were estimated using YIN [18] with manual corrections on monophonic sound tracks prior to mixture. The ground-truth notes (frequency, onset and offset) for each source were obtained by segmenting its pitch trajectory manually.

We evaluate the proposed system at the frame-level. For each estimated pitch trajectory, a pitch estimate in a frame is called correct if it deviates less than a quarter-tone from the pitch in the ground-truth pitch trajectory. Then precision and recall are calculated for each pitch trajectory by

%	Initial		Final	
No.	Precision	Recall	Precision	Recall
1	66.9±3.2	66.9±3.2	**82.7±4.9**	**71.3±5.5**
2	52.3±6.5	52.1±6.5	**64.1±8.8**	**52.4±7.9**
3	61.0±8.5	59.3±8.7	**78.3±11.5**	**70.8±12.7**
4	81.8±5.0	65.3±7.4	**82.6±5.4**	**73.9±5.8**

Table 3. Frame-level evaluation results (Mean±Std) for each pitch trajectory. Track No. from 1 to 4 corresponds to the 4 parts of the quartets. "Initial" refers to the initial pitch trajectory formation (Section 3.2); "Final" refers to the final pitch trajectory formation (Section 3.4)

%	Precision	Recall
Klapuri06 [1]	87.2±2.0	66.2±3.4
Multi-pitch estimation (MPE)	84.9±1.7	**79.9±2.9**
Multi-pitch tracking (MPT)	**88.6±1.7**	77.0±3.5

Table 4. Frame-level evaluation results in the mixture instead of each trajectory.

$$\text{Precision} = \frac{\#\text{cor}}{\#\text{est}} \qquad \text{Recall} = \frac{\#\text{cor}}{\#\text{ref}} \tag{10}$$

where #cor, #est and #ref are the number of correctly estimated, estimated and reference pitches, respectively.

Table 3 presents the average frame-level evaluation results on the 10 pieces. The final tracking results are compared with the initial tracking results obtained from Section 3.2, which serves as a baseline obtained from multi-pitch estimation. For all four tracks, the proposed pitch trajectory formation method significantly improves either precision or recall from the baseline method. For Track 3, this improvement is up to 17.3% in precision and 11.5% in recall. In addition, in both initial and final tracking, results of Track 1 and 4 are better than Track 2 and 3. This is in accordance with previous researchers' observations that melody and baseline are easier to transcribe than other source streams [12].

Table 4 presents the frame-level evaluation results. We compare our system with Klapuri06 [1], a state-of-the-art multi-pitch estimation approach. We used Klapuri's code and suggested parameter settings. We can see that the best precision is obtained by MPT (Section 2+3), while the best recall is obtained by MPE (Section 2). Klapuri06 gets a high precision, which is indistinguishable from MPT, but its recall is much lower than both MPE and MPT. From MPE to MPT, precision has a significant improvement of 3.7%, while the two recalls are indistinguishable, considering the variances. It indicates that pitch trajectory formation improves the multi-pitch estimation results.

We also evaluate our system at the note-level in the mixture, as other researchers do [3,4,15]. It is evaluated in two ways. In the first way (Onset), a note estimate is correct if its frequency (average over all pitch estimates in this note) deviates less than a quarter-tone from a ground-truth note, and its onset time differs less than 50ms/100ms from the

%	Precision	Recall	AOR
Onset(50ms)	49.1±4.8	58.0±4.8	76.7±2.8
Onset(50ms)+Off	34.9±5.3	41.2±5.8	87.8±1.8
Onset(100ms)	65.5±4.7	77.4±3.1	73.7±3.1
Onset(100ms)+Off	46.0±5.5	54.3±5.5	85.1±2.3

Table 5. Note-level evaluation results in the mixture instead of each trajectory.

ground-truth onset time. The second way (Onset+Off) has the same requirements for frequency and onset time, but also requires that each offset time estimate deviate from the true offset time by less than 20% of the true note duration. Precision and recall are calculated by Eq. (10), where #cor, #est and #ref are the number of correctly estimated, estimated, and reference notes, respectively. *Average Overlap Ratio (AOR)* between correctly estimated notes and their ground-truth notes is calculated as

$$\text{AOR} = \text{Average}\left(\frac{\min\{\text{offsets}\} - \max\{\text{onsets}\}}{\max\{\text{offsets}\} - \min\{\text{onsets}\}}\right) \quad (11)$$

Given a correctly estimate note and its corresponding ground truth note, "onsets" is the set of onset times for both the estimated and the and the true note and "offsets" is similarly defined. "Average" is over all correctly estimated notes.

Table 5 shows the note-level evaluation results. In Onset (100ms), both precision and recall are promising, and AOR value is high. This indicates that our system outputs good note estimates in both frequency and duration (offset subtracts onset). However, either reducing the onset threshold from 100ms to 50ms or adding the offset criterion decreases precision and recall significantly. This indicates that our system does not estimate the absolute onset/offset times precisely. This is not a surprise, since our current system does not have an onset/offset detection module.

5. CONCLUSION

Wee presented a novel system for multi-pitch tracking. Our system first estimates pitches and polyphony in each time frame. Then pitch trajectories are formed by constrained clustering pitch estimates across frames. Our system achieved promising results on ten real music pieces.

Currently, when clustering the pitches into trajectories, only harmonic structure is used. Future work includes incorporating musicological information into clustering.

This work was supported by National Science Foundation grant IIS-0643752.

6. REFERENCES

[1] A. Klapuri: "Multiple fundamental frequency estimation by summing harmonic amplitudes," *Proc. ISMIR*, pp. 216-221, 2006.

[2] M. Davy, S. J. Godsill, and J. Idier: "Bayesian analysis of western tonal music," *J. Acoust. Soc. Am.*, Vol. 119, No. 4, pp. 2498-2517, 2006.

[3] M. Ryynänen and A. Klapuri: "Polyphonic music transcription using note event modeling," *Proc. WASPAA*, pp. 319-322, 2005.

[4] G. E. Poliner and D. Ellis: "A Discriminative Model for Polyphonic Piano Transcription" *EURASIP Journal on Advances in Signal Processing*, Vol. 2007, 2007.

[5] A. S. Bregman: *Auditory Scene Analysis*, Cambridge, MA: MIT Press, 1990.

[6] T. Virtanen and A. Klapuri: "Separation of harmonic sounds using multipitch analysis and iterative parameter estimation," *Proc. WASPAA*, pp. 83-86, 2001.

[7] M. Bay and J. W. Beauchamp: "Harmonic source separation using prestored spectra," *Proc. ICA*, pp. 561-568, 2006.

[8] J. Woodruff, Y. Li and D. Wang: "Resolving overlapping harmonics for monaural musical sound separation using pitch and common amplitude modulation," *Proc. ISMIR*, 2008.

[9] E. Vincent: "Musical source separation using time-frequency source priors," *IEEE Trans. Audio Speech Language Process.*, Vol. 14, No. 1, pp. 91- 98, 2006.

[10] Z. Duan, Y. Zhang, C. Zhang and Z. Shi: "Unsupervised single-channel music source separation by average harmonic structure modeling," *IEEE Trans. Audio Speech Language Process.*, Vol. 16, No. 4, pp. 766-778, 2008.

[11] M. Kim, S. Choi: "Monaural music source separation: Nonnegativity, sparseness, and shift-invariance," *Proc. ICA*, pp. 617-624, 2006.

[12] M. Goto: "A real-time music scene description system: predominant-F0 estimation for detecting melody and bass lines in real-world audio signal," *Speech Communication*, Vol. 43, pp. 311-329, 2004.

[13] J. Eggink and G. J. Brown: "Extracting melody lines from comlex audio," *Proc. ISMIR*, 2004.

[14] H. Kameoka, T. Nishimoto, and S. Sagayama: "A multipitch analyzer based on harmonic temporal structured clustering," *IEEE Trans. on Audio, Speech and Language Processing*, Vol. 15, No. 3, pp. 982-994, 2007.

[15] W.-C. Chang, A. W.Y. Su, C. Yeh, A. Roebel and X. Rodet: "Multiple-F0 tracking based on a high-order HMM model," *Proc. DAFx*, 2008.

[16] M. Lagrange and G. Tzanetakis: "Sound source tracking and formation using normalized cuts," *Proc. ICASSP*, 2007.

[17] Ö. Yılmaz and S. Rickard: "Blind separation of speech mixtures via time-frequency masking," *IEEE Trans. Signal Process.*, Vol. 52, No. 7, pp. 1830-1847, 2004.

[18] A. de Cheveigné and H. Kawahara: "YIN, a fundamental frequency estimator for speech and music," *J. Acoust. Soc. Am.*, Vol. 111, pp. 1917-1930, 2002.

[19] K. Wagstaff, C. Cardie, S. Rogers, and S. Schroedl: "Constrained K-means clustering with background knowledge," *Proc. ICML*, 2001.

[20] I. Davidson and S. S. Ravi: "Clustering with constraints: feasibility issues and the k-means algorithm," *Proc. SDM*, 2005.