

# Robot Musical Accompaniment: Real-time Synchronization using Visual Cue Recognition

Angelica Lim    Takeshi Mizumoto    Takuma Otsuka    Toru Takahashi  
 Kazunori Komatani    Tetsuya Ogata    Hiroshi G. Okuno

Graduate School of Informatics, Kyoto University

## 1 Introduction

In a typical musical ensemble performance, each musician must play at the same speed, at the same location in the piece. When this happens, we say they are synchronized. Currently, musical accompaniment systems such as [1] [2] achieve good synchronization with human soloists. However, in larger ensembles with many players playing different parts, synchronization becomes difficult.

Several existing approaches in audio processing could be applied for large ensemble synchronization. Score followers such as [3] are capable of dealing with polyphonic input, though have the constraint of requiring prior knowledge of all players' scores. This is not the case in real musical ensembles; each player typically only knows his own part. Beat trackers such as [4] can also deal with polyphonic input, but a percussive beat is required in order to reliably extract a tempo. At the same time, studies in music [5] suggest that visual cues, such as a conductor's baton movements or musicians' body movements, are necessary for temporal coordination. Indeed, certain musical situations, such as a simultaneous first note of a piece, are virtually impossible to synchronize using audio alone.

In light of these issues, we propose a vision-based accompanist system which a) starts and stops in synchrony with the lead player b) adapts to tempo changes indicated by the leader c) can be used for polyphonic music or noisy environments and d) by nature, does not require a full score a priori nor use of percussion. As a starting point for future research, we have implemented a filtered Hough line detector and state machine to recognize *visual cues* (or *gestures*) of a human flute player. We install the system onto a robot thereminist [6] and synthetic vocalist, which use this visual information to play a piece in synchrony with one human lead player.

## 2 A Vision-based Ensemble Synchronization System

### 2.1 Visual Cue Recognition

Based on empirical observation, we have identified three types of visual cues (see Figure 1) used by flutists to indicate timing events for synchronization:

- A *Start Cue* indicates the start of a piece or section of music. It is a down-up-down movement of the flute.
- An *End Cue* indicates the end of a long, held note or fermata. It is a down-up movement.
- A *Beat Cue* indicates a beat in music. Similar to the End Cue, it is a down-up movement. A *tempo change* can be deduced from a series of beat cues.

Since each cue is linked to the instrument's movement, we locate and track the flute to recognize these gestures. Flutes are not easily tracked using standard methods because they are shiny, producing specular reflection. In particular, tracking by optical flow techniques is difficult because features must remain consistent over time, which is rarely the case with reflective objects. Thus, we use a simple Hough Transform algorithm, which can robustly locate

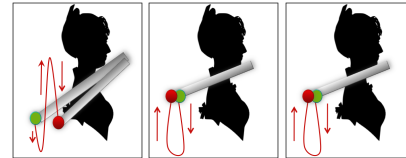


Figure 1: Trajectories of flute visual cues

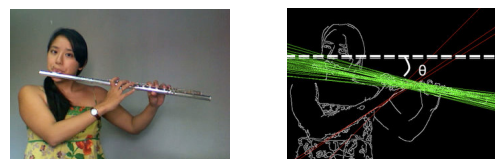


Figure 2: (a) Original input image and (b) processed image with detected Hough lines in green; outliers marked in red

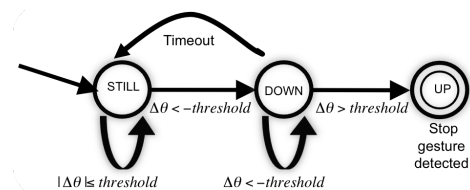


Figure 3: State machine to detect End Cue

the straight flute throughout a stream of input images.

#### 2.1.1 Hough Line Detection

We first perform Canny edge detection [7] and the Hough Transform [8] on each image. This outputs multiple lines with the same orientation, corresponding to the flute, as shown in Figure 2(b).

#### 2.1.2 Outlier Line Pruning

To increase robustness against background clutter, we use the RANSAC algorithm [9], a popular outlier detector, to prune the set of detected lines. Since a flute produces many lines, we improve the localization of the flute by removing outlier lines with dissimilar location and angles. Once pruning is complete, we extract the mean angle  $\theta$  of the remaining lines for input into our recognizers.

#### 2.1.3 Finite State Machine

Three finite state machines (FSM) serve as gesture recognizers, defined from the trajectories in Figure 1. Figure 3 shows the End Cue FSM as an example.

At each time step, we determine the instantaneous change in  $\theta$  (derived from the image processing stage) between two subsequent video frames  $F$  at time  $t - 1$  and  $t$ .

$$\Delta\theta = \theta(F_t) - \theta(F_{t-1}) \quad (1)$$

Using this  $\Delta\theta$ , we determine the current state of the flutist's movement with respect to finite state machines (FSM) defined from the trajectories in Figure 1.

$$STATE(\Delta\theta) = \begin{cases} DOWN & \text{if } \Delta\theta < -threshold \\ UP & \text{if } \Delta\theta > threshold \\ STILL & \text{otherwise} \end{cases} \quad (2)$$

A *DOWN* state indicates that the flutist is currently moving the end of her flute downwards, and so on. The *threshold* acts as a rudimentary smoother, to prevent small angle fluctuations from being detected as intentional movement. Finally, we define a timeout (Fig. 3) to ensure that detected states are temporally close. If too much time passes between changes in state, we assume the two movements do not belong to the same cue, and transition back to the initial state.

### 2.1.4 Calculation of Intended Tempo

If multiple beat cues are detected, we attempt to determine the tempo or beat interval intended by the flutist. We require that detected beat cues are equally spaced apart; this ensures we are truly seeing a cue pattern. Given a 3-beat gesture sequence  $b_0, b_1, b_2$ , where  $t(b)$  is the time when beat  $b$  is detected,  $t(b_0) < t(b_1) < t(b_2)$ , we detect a *tempo change* if

$$t(b_2) - t(b_1) - (t(b_1) - t(b_0)) < 100ms. \quad (3)$$

We then extract the beat interval according to the following formula, and from this infer the tempo.

$$beatinterval = (t(b_2) - t(b_0))/2 \quad (4)$$

### 2.1.5 Gesture Filtering by Score Location

Finally, we add an extra layer of filtering (see Fig. 4) which decides whether or not to modify the accompanist's performance, based on the current score location. This is similar to humans paying attention to other players during critical passages in the score, as described in [5]. For example, Start Cues are ignored if the player is already playing, and End Cues are ignored if the current note is not being held. Tempo changes are valid to establish the initial tempo as well as change tempo mid-song.

## 2.2 Ensemble Player System

An overview of our gesture recognition module and ensemble system is shown in Figure 4. Our robot's Point Grey Fly camera is used to take greyscale images at 1024x728 resolution. These input frames are then processed to extract the flute's angle, which is fed into three FSMs which run simultaneously. Every time an FSM recognizes a gesture, a message is sent via TCP/IP to the ensemble players. Gesture filtering by score location is then performed on the player's side, with respect to their individual scores. In this way, we control the theremin robot's arm and Vocaloid synthetic voice running on Windows.

All modules run as fast as possible without any special inter-module synchronization scheme. However, in order to have perfect synchronization, we should account for delays such as network lag and image processing time, and adjust player's performances accordingly. Though not implemented here, it is planned as future work.

## 3 Evaluation

We evaluate the accuracy of our gesture recognizer by recording its output given 30 samples of each gesture (performed by the same flutist). The same experiment is performed at 3 different brightness levels, shown in Figure 5. The results are shown in Table 1.

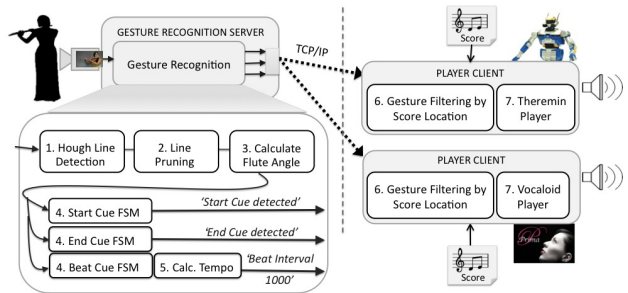


Figure 4: Overview of the Visual Cue Ensemble System

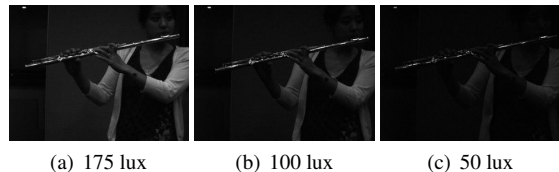


Figure 5: Actual input images from robot's camera for our three experimental conditions.

Visual Cue to Detect	175 lux	100 lux	50 lux
Start (%)	97	100	83
End (%)	100	97	100
Tempo Change (%)	87	63	73
↔ Average Beat Interval (ms)	935	863	882

Table 1: Recognition rates for each type of gesture. The average beat intervals were detected with the user's intended beat interval at 870 ms (69 bpm).

Note that our recognizer's performance decreases at lower light levels, but exhibits robustness even in very dark lighting conditions. The Tempo Change cue appears harder to recognize, since it is a more complicated gesture comprised of a sequence of 3 beat cues. However, when successful, the detected beat interval is fairly accurate.

## 4 Conclusion and Future Work

We have presented a promising new paradigm for using visual cues to synchronize musical accompaniment, and have implemented it on a robot system. Anthropomorphic music robots typically have multiple sources of input, and in the future we hope to combine audio with visual information to improve interaction. Other research paths include generalizing gesture recognition to other instruments, and using eye contact as another mode of human-robot communication.

This research was supported in part by KAKENHI and GCOE.

## References

- [1] R. Dannenberg, *An On-line Algorithm for Real-time Accompaniment*, Proceedings of ICMC, Paris, France, 1984.
- [2] C. Raphael, *Synthesizing Musical Accompaniments with Bayesian Belief Networks* Journal of New Music Research, vol. 30, 2001, pp. 59-67.
- [3] A. Cont, *A coupled duration-focused architecture for realtime music to score alignment*. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2009.
- [4] M. Goto et al. *Music Understanding At The Beat Level Real-time Beat Tracking For Audio Signals*, Computational Auditory Scene Analysis, 1998.
- [5] A. Williamson and J.W. Davidson, *Exploring co-performer communication* Musicae Scientiae, vol. 6, 2002, pp. 53-72.
- [6] T.Mizumoto et al. *Thereminist Robot: Development of a Robot Theremin Player with Feedforward and Feedback Arm Control based on a Theremin's Pitch Model*, Proc. of IROS 2009.
- [7] J. Canny, *A Computational Approach to Edge Detection* IEEE Trans. on Pattern Analysis and Machine Intell., vol. 8, 1986, pp. 679-698.
- [8] R.O. Duda and P.E. Hart, *Use of the Hough transformation to detect lines and curves in pictures* Commun. ACM, vol. 15, 1972, pp. 11-15.
- [9] R.C. Bolles et al., *A RANSAC-based approach to model fitting and its application to finding cylinders in range data* Proc. of IJCAI, 1981, pp. 637-643.