Computer Vision-Based Interface for the Control of Meta-Instruments

Frederic Jean[1], Alexandra Branzan Albu[2], Wolfgang A. Schloss[3], and Peter Driessen[2]

[1] Computer Vision and Systems Laboratory, Dept. of Electrical and Computer Engineering, Laval University, Quebec (QC), Canada
fjean@gel.ulaval.ca
[2] Dept. of Electrical and Computer Engineering, University of Victoria, Victoria (BC), Canada
aalbu@ece.uvic.ca, peter@ece.uvic.ca
[3] School of Music, University of Victoria, Victoria (BC), Canada
aschloss@finearts.uvic.ca

Abstract. This paper describes a "virtual keyboard" for the control of meta-instruments. The proposed approach uses video input data and computer vision algorithms for tracking feet motion and their interaction with a planar keyboard with no force feedback. The design of the "virtual keyboard" is directly inspired from the traditional, organ-style bank of foot pedals. The proposed approach accurately detects in real-time the hit of a keyboard with either one or both feet, as well as the location(s) of hit(s) (i.e. what keys have been "pressed").

Keywords: computer vision, tracking, real-time event detection.

1 Introduction

Inserting a computer in the musician-instrument loop leads to a dramatic change in the paradigm of interaction. As shown in [1], the new paradigm eliminates the one-on-one correspondence between the performer's actions and the sonic result, which is characteristic for acoustical instruments. A computer-mediated interaction results in meta-instruments, which use algorithms for music generation so that a particular gesture of the performer can have practically any musical result. This fundamental change in paradigm challenges the perception of cause-and-effect relationships in live performance using meta-instruments. A reasonable trade-off between the one-on-one acoustical correspondence and the unlimited number of mappings possible with meta-instruments is to enable the performer to select among a limited number of mappings by interacting with the software for music generation.

For instance, the Mathews/Boie Radio Drum is a sensor able to track the position of two mallets in 3D and in real-time. It generates no sound; the effect of a performed gesture is entirely determined by software. The musician interacts with the software via a bank of organ-style foot pedals. The pedals are used for various kinds of control information rather than playing pitches. The proposed prototype mimics the real key-

board currently in use for the control of meta-instruments, but brings the advantage of portability and flexibility of use. Its underlying interaction paradigm falls in the category of perceptual interfaces.

Computer Vision techniques play a central role in the design of perceptual interfaces. Such interfaces are suitable for a variety of applications, ranging from health care [2] to video games [3], and to music generation and control. The application context is essential for the process of selecting the set of relevant gestures, as well as for the required level of accuracy and latency in gesture recognition. Thus, the remainder of this section will focus on vision-based interfaces related to musical applications.

Recently proposed computer vision approaches for musical interfaces define their set of gestures among various facial motions and/or expressions. The Mouthesizer described in [4] extracts basic shape parameters of the mouth cavity such as height, width and aspect ratio and maps them to musical control parameters. The system proposed in [5] processes infrared data for recognizing head nods, tilts, and shakes. However, controlling musical interfaces via facial expressions imposes extra cognitive load on the performer. Moreover, in concerts, facial expressions are integral part of the artistic performance. Therefore, controlling an interface with facial gestures is not an optimal solution for live concerts or rehearsals.

Our approach addresses the constraints imposed by the use of meta-instruments during live performance or rehearsal by tracking feet motion relatively to a virtual keyboard. The design of the virtual keyboard is directly inspired from the traditional, organ-style bank of foot pedals. In the proposed prototype, colour cues and elevated height for the keys corresponding to black organ keys (see Fig. 1a) are used to help performers locate the desired key. Moreover, a similar set of foot motions are used for controlling the interaction with the virtual keyboard as in the traditional set-up. This similarity has a positive impact on the learnability of the interface. The following section provides a detailed description of the proposed approach.

2 Proposed Approach

The proposed approach uses input data from a top-view web camera for the real-time detection of two main events:

A. The hit of a keyboard with either one or both feet, as well as the location(s) of hit(s) (i.e. what keys have been "pressed"). Apostrophes differentiate between a real key press, which gives force feedback, and the virtual key press detected by our approach, which means physical foot-key contact only.
B. The idle status of a foot after a keyboard hit. A foot has idle status if the user maintains "pressing" the key with the foot, thus signifying that the interface response associated with the key "press" must be continued.

For event detection, the relative positions of both feet with respect to the keyboard are continuously tracked. Tracking is the core of our approach, as its output is used for event detection. Prior to tracking, preprocessing is needed for initialization and background subtraction. All steps of the proposed approach are detailed below.

Manual initialization. This process assumes that both camera and keyboard have fixed positions during the musical performance, and it takes place before the perform-

ance begins. A typical result is shown in Fig. 1. During initialization, the user specifies via a simple graphical interface the spatial location of the keyboard on the image plane as well as a 'workspace' W(x,y) surrounding this keyboard.

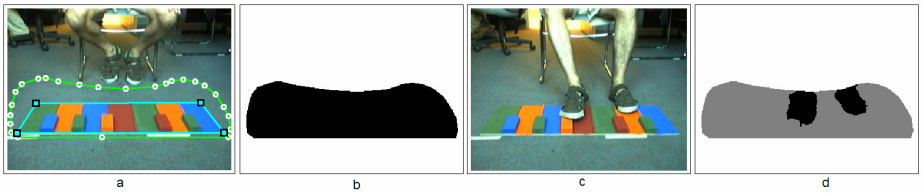

Fig.1. a) workspace boundary specified as a discrete set of contour points (circles); corners of the keyboard (squares) are also user-specified. b) mask W of the workspace; c) original frame with feet present in the workspace; d) result of feet detection via background subtraction.

Background subtraction. The algorithm herein computes statistics of static background, with feet not present in the workspace; for robust results, 2 seconds of video with static background are sufficient. Foreground-background segmentation (see Fig. 1c,d) uses the difference in their corresponding first- and second-order statistics.

Feet Tracking. The proposed tracking algorithm relies upon colour cues. Therefore, the video sequences in our database were acquired with shoes having their forward extremities outlined in white. These white regions will be thereafter called markers.

Tracking begins with finding one or two feet regions using background subtraction. Once feet regions found, feet correspondence is achieved by computing the maximum-overlap with feet locations in the previous frame. In some cases, a simple inter-foot distinction is not possible, since region merging occurs.

Region merging is an artifact due to imperfect background subtraction; therefore, its detection is important for the search for markers. Moreover, if both markers are located in one region, it is essential to keep a consistent correspondence for markers located on the left and right foot respectively. This correspondence is based on a maximum overlap criterion with the markers positions in the previous frame.

Marker detection is based on gray-scale information. Since markers are white, bright pixel in the foot regions are likely to belong to a marker. For two disjoint foot regions, one marker region is detected inside each of these using connected component labeling. Specifically, in each foot region the largest connected component composed of bright pixels is associated to a marker. When region merging occurs, the two largest connected components composed of bright pixels are associated with markers.

Detection of keyboard hits (event A). As in the case of physical foot pedals, the velocity of the foot descending upon the key is maximal just before the hit occurs. Therefore, the hit detection uses the trajectory of the y-coordinate of the marker's centre of mass. Let $p_{mc}(t)$ denote the projection onto the image plane of the trajectory of the center of mass of one marker tracked throughout the video sequence. The foot *f* containing that marker is considered to have hit the keyboard at frame t_h if the following conditions are simultaneously met:

$$\begin{aligned} &1.\ v_y^f(t_h-1) = p_{mc,y}^f(t_h-1) - p_{mc,y}^f(t_h-2) > \tau_v \\ &2.\ a_y^f(t_h) = v_y^f(t_h) - v_y^f(t_h-1) = p_{mc,y}^f(t_h) - 2p_{mc,y}^f(t_h-1) + p_{mc,y}^f(t_h-2) < 0 \\ &3.\ a_y^f(t_h-1) = v_y^f(t_h-1) - v_y^f(t_h-2) = p_{mc,y}^f(t_h-1) - 2p_{mc,y}^f(t_h-2) + p_{mc,y}^f(t_h-3) > 0 \end{aligned} \quad (1)$$

The above equations associate a keyboard hit with the occurrence of a local maxima in the vertical velocity of the marker. In order to preserve only 'significant' local maxima as opposed to those induced by noise, the y-velocity in the previous frame (t_h-1) must be above a certain threshold τ_v. The temporal location of the keyboard hit is indicated by a zero-crossing (positive towards negative) in the vertical acceleration.

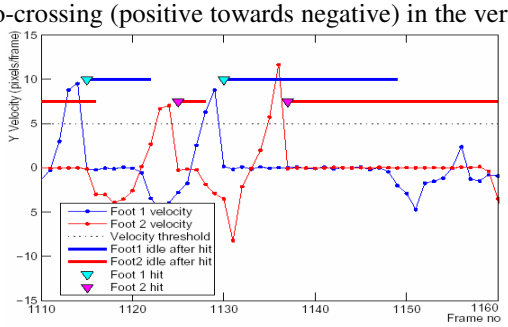

Fig. 2. Keyboard hit and idle state detections; τ_v=7 pixels/sec and τ_i=5 pixels.

Detection of a foot idle state (event B). The foot idle state may occur after a keyboard hit; it means that the user has intentionally left his foot on a key of the keyboard in order to continue the action associated with that specific key. Once a keyboard hit for foot *f* detected at frame t_h, foot *f* is in an idle state on a key at current frame t_c if the spatial location of its marker's center of mass stays within τ_i pixels during [t_h, t_c]. Fig. 2 shows an example of keyboard hit and idle state detections.

Key Identification. Once a keyboard hit event detected, we must determine which key has been "pressed". The key identification locates the contact point between the foot and the keyboard, which belongs to the "pressed" key. This contact point is approximated by the center of mass $p_{mc}(t_h)$ of the marker on the foot "pressing" the key. To identify the "pressed" key, a homography between the real keyboard and the image plane is performed. The homography matrix $\boldsymbol{H}$ is computed using the correspondence between the keyboard corners b_s in the image and the keyboard model corners b_s^M. It maps any point inside the keyboard model into a point in the image, in particular the corners of each key in the keyboard. The correspondence between the key corners in the real keyboard $k_{l,s}^M$ and the key corners in the image plane $k_{l,s}$ is computed with $\hat{k}_{l,s} = H\hat{k}_{l,s}^M$, where $\hat{k}_{l,s}^M = \left[k_{l,s}^M, 1\right]^T$, $\hat{k}_{l,s} = \left[e \cdot k_{l,s}, e\right]^T$ are expressed in homogeneous coordinates with *e* being the scale factor. Key corner coordinates determine labeled polygonal regions for every key in the image plane. Thus, the key identification process finds the label l_h corresponding to the polygonal region containing $p_{mc}(t_h)$.

3 Experimental Results

The keyboard prototype designed for this work represents one octave of a piano keyboard and it is made of wood, glue, and paint. Video data was acquired with a Logitech Quickcam Pro 4000 with a 320x400 frame resolution and a frame rate of 30 fps.

The proposed approach was validated on 8307 frames of video footage acquired with a music performer simulating foot actions which are typically used for the control of a meta-instrument with foot pedals. The video footage was parsed in two sequences, acquired at different times. Based on the values shown in Table 1, the mean ratio of missed key-"press" detections is 11.5 %, while the mean ratio of false key-"press" detections is 1%. The mean error in key identification is 9.4%. The proposed approach detects all key-"presses" one frame (i.e. 1/30 seconds) after their occurrence. All events are detected in real-time, thus our approach fulfils a critical requirement for a no-latency interaction with the meta-instrument.

Table 1. Experimental validation

Statistical performance measures	Seq. 1	Seq. 2
Total no. of frames	8307	1225
Total no. of key-press events	391	157
No. of key-press events detected correctly	328	145
No. of missed key-press detections	63	12
No. of false key-press detections	4	0
No. of errors in key identification	37	0

4 Conclusion

This paper describes a novel approach for the design of a lightweight and portable keyboard to be used for the control of a meta-instrument. The proposed approach uses computer vision algorithms for tracking feet and their interactions (key-"presses" and idle states) with the virtual keyboard in real-time.

Experimental results have proven that the performances of our approach are excellent in ambient daylight. Future work will concentrate on testing the stability and robustness of our algorithms in poor lighting conditions, which are often present in concert environments.

References

1. Schloss, W.A., and Jaffe, D. Intelligent Musical Instruments. INTERFACE Journal for New Music Research, 22,3 (Aug. 1993).
2. Graetzel, C., Fong, T., Grange, S., and Baur, C. A non-contact mouse for surgeon-computer interaction. Technology and Health Care, 12, 3, 2004
3. Sony EyeToy; www.eyetoy.com.
4. Lyons, M.J., Haehnel, M., and, Tetsutani, N. Designing, Playing, and Performing with a Vision-Based Mouth Interface. In Proc. of Conf. on New Interfaces for Musical Expression (NIME'03), 2003, 116-121.
5. Merril, D. Head-tracking for gestural and continuous control of parameterized audio effects. In Proc. of Conf. on New Interfaces for Musical Expression (NIME'03), 2003, 218-219.