SOUND MY VISION: REAL-TIME VIDEO ANALYSIS ON MOBILE PLATFORMS FOR CONTROLLING MULTIMEDIA PERFORMANCES

Miranda Kreković
School of Computer and Communication Sciences (EPFL), Lausanne, Switzerland
miranda.krekovic@epfl.ch

Franco Grbac
Independent researcher
franco.grbac@gmail.com

Gordan Kreković
Faculty of Electrical Engineering and Computing, University of Zagreb, Croatia
gordan.krekovic@fer.hr

ABSTRACT

This paper presents Sound My Vision, an Android application for controlling music expression and multimedia projects. Unlike other similar applications which collect data only from sensors and input devices, Sound My Vision also analyses input video in real time and extracts low-level video features. Such a versatile controller can be used in various scenarios from entertainment and experimentation to live music performances, installations and multimedia projects. The application can replace complex setups that are usually required for capturing and analyzing a video signal in live performances. Additionally, mobility of smartphones allows perspective changes in sense that the performer can become either an object or a subject involved in controlling the expression. The most important contributions of this paper are selection of general and low-level video feature and the technical solution for seamless real-time video extraction on the Android platform.

INTRODUCTION

In the context of new interfaces for musical expression, mobile devices such as smartphones and tablets take an important place. They are multipurpose and omnipresent devices powerful enough for real-time digital signal processing. What makes them different from pocket computers are hardware prerequisites like touch screens, microphones, cameras, and various sensors which can serve as a foundation for building musical interfaces. For those reasons, mobile devices have become an interesting platform for computer music research [1-3] and for developing practical applications [4, 5].

The research conducted by Kell Thor and Marcelo M. Wanderley at the beginning of 2014 showed that there were more than 5000 iOS applications for making music available on the official app store [2]. The variety and versatility of those applications ensure their usage in different scenarios – from entertainment and experimentation to music production and live performances. While some applications are capable of producing sounds by themselves, others are designed to serve as controllers, so they only send parameter data to other devices.

Musical expression can be observed within several dimensions such as pitch range, pitch style and tuning, dynamic range, timbre style and process, articulation, ornamentation, number of parts, and spatial dimension. [6]. In order to achieve expressibility in those dimensions, most mobile applications primarily rely on inputs from the touch screen. Using the graphical representations, a variety of metaphors for controlling musical expression can be employed. The metaphors such as keyboards, dials, strings, pads, and sliders are intuitive to use because they inherently inform users how to interact with the application and what sonic results of their actions they may expect [7].

In addition to a touch screen, mobile devices are usually equipped with various peripherals and sensors including gyroscope, accelerometers, Global Positioning System (GPS) module, microphone, and cameras. Most of the mentioned input devices and sensors have been used in mobile applications for making music and controlling musical performances. However, this is not the case with cameras.

We assume that the first reason for not using the input image for controlling musical expression is the lack of the systematic research of video features which can be generally used for such purposes. The second reason is that just until a few years ago mobile devices were not capable of complex video analysis in real time.

Video analysis represents a challenging part of many interactive motion sensing systems for installations, enhanced dance choreographies, and other multimedia projects [8]. Real-time video feature extraction on mobile devices can make technology more accessible and convenient for musicians and performance artists. Using a smartphone or tablet, they can easily control an interactive system by dance movements or any other visual information. For such a purpose musicians and performing artists would otherwise need a complex hardware setup with cameras, computers, and lots of cables. Therefore, using widely present and relatively affordable devices they could be higher motivated to explore multimodalities of interaction between physical movements, musical expression, and multimedia.

Another important benefit of using live video input from mobile devices is related to the philosophical perspective. While in usual setups cameras are either fixed in one place or have limited trajectories, mobile devices can move in space without limitations and this allows artists to change a perspective. Instead of being an object observed by the camera, a performer can sometimes become a subject who carries a mobile device to capture the audience or a part of scenography which is not visible from other angles. Such a changed perspective is an interesting philosophical foundation of exploring interactivity between the performer and a multimedia system.

Use of mobile applications could open even more interesting opportunities. For instance, by applying the principles of crowdsourcing, the audience can capture the scene by their smartphones and thereby contribute to the interaction with the multimedia system. Their multiple perspectives can be combined when using mobile devices from different places in the audience.

Since there is a lot of practical benefits and new opportunities for performance artists, the goal of our research was to design and develop a mobile application for controlling interactive and multimedia system which also supports real-time video analysis. As the result, we created Sound My Vision, an Android application which captures data from sensors and the touch screen, but additionally extracts video features form the camera input. The mobile application sends collected data over the wireless network so that the other device with a server side application can use those parameters for controlling a multimedia project. The communication is based on the Open Sound Control (OSC) protocol for integrating multimedia equipment and software.

Two most important contributions of this research are (1) selection of video features convenient in general cases of controlling multimedia systems using the input video and (2) a technical solution which allows seamless real time video analysis on the Android operating system. Sound My Vision can serve as a versatile and generic controller intended for multimedia artists, musicians, lighting designers, contemporary dancers, and other performance artists.

SOUND MY VISION

As an OSC controller, Sound My Vision is comparable to existing applications such as andOSC, Kontrolleur, and OSCdroid. These applications are intended to send control parameters over the wireless network and they are not capable of producing sounds by themselves. The concept of mobile controllers usually implies a certain level of flexibility in sense that users can define parameter names and ranges.

Unlike the mentioned applications, Sound My Vision additionally supports real-time extraction of video features from the input camera. Video features are treated in the same ways as other parameters collected from sensors and from the touch screen. The general data flow is illustrated in Figure 1.

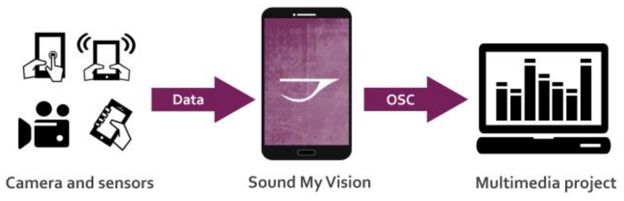
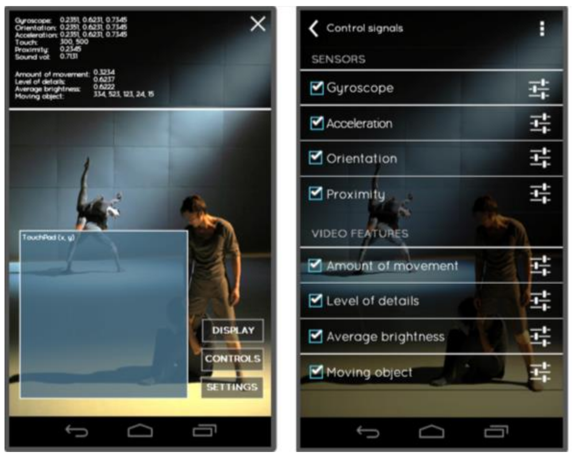


Figure 1. Sound My Vision as a versatile controller for interactive and multimedia works.

The application collects the following parameters from sensors, buttons, and the touch screen: (1) physical movements of the device (orientation, linear acceleration, and radial acceleration), (2) proximity of near objects, (3) geographic coordinates, (4) audio volume level controlled by the side buttons, and (5) coordinates of the point where the user touched the screen.

On the other hand, the video features extracted in real-time are selected to describe the movements in the scene and the general characteristic of the input image as described later in more details. For that reason, the following features are extracted: (1) amount of movement, coordinates, size and inclination of the moving object, (2) level of details in the current image, and (3) level of brightness of the current image.

The user can choose which of these parameters will be calculated and sent over the network as shown in Figure 2. Additionally, the application provides a possibility to define arbitrary OSC names and value ranges for each of the parameters. This feature allows users to adapt the output format in order to simplify the server-side implementation.


```
Gyroscope:    0.2351, 0.6231, 0.7345
Orientation:  0.2351, 0.6231, 0.7345
Acceleration: 0.2351, 0.6231, 0.7345
Touch:        300, 500
Proximity:    0.2345
Sound vol:    0.7131

Amount of movement:  0.3234
Level of details:    0.6237
Average brightness:  0.6222
Moving object:       334, 523, 123, 24, 15
```

Control signals

SENSORS
- ✓ Gyroscope
- ✓ Acceleration
- ✓ Orientation
- ✓ Proximity

VIDEO FEATURES
- ✓ Amount of movement
- ✓ Level of details
- ✓ Average brightness
- ✓ Moving object

Figure 2. Screenshots from the application: home screen (left) and configuration of controls (right).

The parameters calculated from raw sensor data and features from the input video were selected based on modalities of interaction which they can provide in certain use cases. Additionally, we have also taken into consideration availability of sensors in popular Android devices. The rest of this section explains design decisions regarding selection and calculation of sensor data and video features.

Sensor data

The Android operating system supports three general categories of sensors: (1) motion sensors which measure acceleration forces and rotational forces along three axes, (2) environmental sensors which measure air temperature and pressure, illumination, and humidity, and (3) position sensors which measure physical position of a device.

Mobile applications for controlling musical expression usually employ movement and orientation sensors in order to translate physical state and movement of the device to the musical content. One such example is a system for sound synthesis on microstructure level based on the movement [9]. It receives raw data captured from the movement sensors, extracts relevant statistical features, and maps them to parameters of a dynamic stochastic synthesizer using fuzzy logic.

In order to provide raw data which can be used either directly or for extracting higher-level features, Sound My Vision reads data from gyroscope, linear accelerometer, and the rotation sensor.

Location data can be successfully used in various musical applications such as interactive compositions based on the user's location [10] and for navigation purposes [11]. To obtain the geographic data, Sound My Vision employs both location mechanisms supported in the Android platform – location based on the GPS signal and the network location provider estimated from WiFi and cell tower signals. Such an approach ensures the balance between accuracy, frequency of updates, and efficiency of the battery usage.

The proximity sensor provides a measure of how far away an object is from the device. It can be used for controlling sound modulations and effects via hand movements interacting with an infrared beam of light. Besides being a part of some commercial sound synthesizer, proximity sensors are used in the do-it-yourself community, and discussed in the academic literature [12].

In order to exploit all hardware sensors and inputs available in most Android devices, Sound My Vision also reads events from the volume buttons which are usually located on the side of the device. These buttons serve as a metaphor for increasing and decreasing a value of any parameter defined by the mappings on the server side.

Touchpad

If it is enabled by the user, the touchpad is located on the home screen. When the user taps on the touchpad, a small circle appears indicating the position where the user tapped. Any change of the position updates the coordinates X and Y of the circle which are sent to the server side. This way, the user can simultaneously control two parameters using an intuitive touch gestures. Similar two-dimensional controllers can be found in some commercial MIDI controllers and other mobile applications.

Video features

The research on the connection between movement and music languages is continuously yielding interesting results, insights, and technical tools for almost three decades [13, 14]. Important and widely represented set of techniques in that field is computer vision based motion capture. While some computer vision systems were developed to recognize specific groups of physical gestures such as hand movements [15], other serve as generic frameworks for further development and experiments.

The purpose of Sound My Vision is to provide lower-level features of the visual data captured by the camera. Those features are intended to be directly mapped to parameters of an interactive system, or used for higher-level analysis such as recognizing more complex gestures or changes in the scene.

In various usage cases such as interactive installations and dance performances, a moving object is usually the most interesting part of the scene. For that reason, Sound My Vision calculates a group of video features which identifies the position, size, and orientation of the moving object.

General image characteristics such as brightness and level of details may also be useful in applications which rely on analyzing synthesized graphic or complex scenography. Therefore, besides the mentioned movement features, Sound My Vision also extracts general image features of the current video frame. On the server side such features can be either mapped directly to control parameters (e.g. darker image causes the lower sound volume) or they can be observed in longer time frames to extract higher-level parameters (e.g. frequency of the blinking light on the stage).

When selecting the convenient video features, the following aspects were taken into account: (1) relevance in possible use cases, (2) abstractness of the feature, (3) dimensionality, and (4) feasibility of feature extraction on mobile devices. Regarding abstractness, the goal was to find features which are obviously and intuitively related to the current image or sequence. A gradual change in the observed scene should cause a similar change in the appropriate feature. On the other hand, the desirable characteristic was generality so that the selected features can be used to calculate other higher-level features on the server side.

The feature dimensionality was also an important factor, since vectors of numbers and multidimensional data structures would be highly inappropriate for being used as control parameters. For that reason, only scalar features were preferred in the selection.

Moving Object

Separation of the moving object from the background is achieved with two different methods. The user can choose which one is more convenient for the intended usage. The first method is based on background mixture models as initially proposed by Stauffer and Grimson [16]. This is an adaptive technique which relies on the assumption that every pixel's intensity in the video can be modeled using a Gaussian mixture model. Following a simple heuristic it can be determined which intensities most probably belong to the background.

The second method is based on a simple frame differencing. The application allows user to choose the image which represents the background at any moment by tapping the button on the screen. The moving object is determined by calculating the difference between every frame and the reference background selected by the user.

The advantage of the first method is that it does not require any user actions, while the advantage of the second method is somewhat faster detection of moving objects. If there are multiple moving objects in the scene, the application tracks the largest of them. The following features are extracted: (1) amount of movement, (2) coordinates of the moving objects, (3) its orientation, and (4) dimensions (height and width). In order to calculate coordinates, orientation, and dimensions, the algorithm approximates the object's silhouette with an ellipse and calculates its center, angle, and length of axes.

The server-side system can use these parameters to recognize specific situations on the scene or calculate higher-level features of the movement. For instance, based on the coordinates of the moving objects, it would be possible to reconstruct its trajectory and thereby recognize a moving pattern or gesture. Similarly, based on the aspect ratio of the moving object, the system could identify the moment when a dancer spreads the arms.

Image Characteristics

In order to provide parameters which quantify some general characteristics of the image, we selected two simple scalar features. The first one is the average brightness calculated by averaging intensity values of all pixels in the image. It is larger if the image is brighter, so it is related to the amount and position of lights on the scene in relation to the camera. Therefore, the average brightness can be used to synchronize the scene lightning with the music or other multimedia modalities.

The second feature is the level of details. It indicates the textural quality of the image and can be used for differencing between scenes with various numbers of visual elements in it. The level of details is estimated by considering presence of edges in the image, so the number of pixels classified as edges is normalized with the image size. In order to detect edges, we used the Canny edge detection algorithm [17].

Visualization

The main screen of the mobile application by default shows the image from the camera. In addition, the user can turn on additional visual information which could help in understating how the feature extraction works. In particular, besides the original camera image, the application can display any combination of these visualizations: (1) detected moving objects in the image, (2) the ellipse which fits the largest moving object, and (3) detected edges in the image. These visualizations can be overlapped so that the user can see multiple visualizations at the same time together with the original image if it is enabled. Figure 3 shows several such examples.

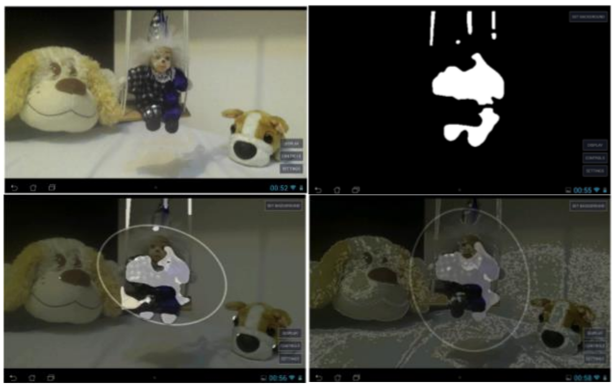


Figure 3. (1) The original image, (2) All moving objects, (3) All of the previous plus the ellipse fitting the largest object, (4) All of the previous plus edges.

TECHNICAL IMPLEMENTATION

The application Sound My Vision is developed both for smartphones and tablets with Android operating systems. It comprises of five modules: user interface, data processing module, OSC module, OpenCV Integrator, and the local database as shown in Figure 4.

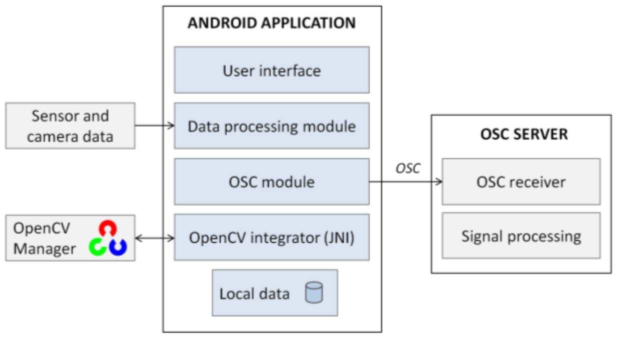


Figure 4. Software architecture.

The user interface adjusts adaptively to different device characteristics (orientation, resolution and dimension of the screen).

The local phone database stores the main properties of the application (OSC names and range values of the parameters, as well as the IP address and port of the OSC server) in order to provide their fast storage and retrieval.

Data processing module collects data from sensors, while OpenCV Integrator enables extracting features from the video signal captured by a camera. All functionalities related to image processing were implemented in C/C++ using the OpenCV library [18]. In order to load and integrate the library with an Android application, it is necessary to install OpenCV Manager on the device. It is an Android service targeted to manage OpenCV library binaries on end users devices. It allows sharing the OpenCV dynamic libraries between different applications on the same device.

The parameters obtained from sensors and camera are sent to the OSC server using the OSC protocol. OSC is an open, transport-independent, message-based protocol developed for communication among computers, sound synthesizers, and other multimedia devices. OSC server listens at the certain port and accepts messages.

OpenCV integration

All algorithms for video analysis in the application are implemented in the C++ programming language using the OpenCV library. The integration of the C++ code with the rest of the mobile application developed in Java has been made using the Java Native Interface (JNI).

OSC integration

The OSC integration is based on the JavaOSC library [19]. As networking is not supported on the main thread in Android applications, communication with the server is realized by using special background thread. Every supported parameter is sent to the server using unique path and a single value, for example `/orientation-coordinateX 0.481`.

EXPERIMENTS AND USE CASES

In order to evaluate and demonstrate how the application works in different scenarios, we conducted several experiments some of which are described in this section. The server side was implemented using Pure Data, a visual programming environment for music and multimedia projects. The purpose of the server side was to read parameters received from the mobile application and use those parameters to evaluate whether they match expected values and to demonstrate how to control generated sounds or graphics.

Case 1: Playing notes using the touchpad

The first use case shows how the touchpad can serve as a simple interface for playing notes. One dimension is used for controlling the pitch of the note, while the other is mapped to the perceived loudness. Instead of allowing continues frequencies, in this experiment we selected discrete pitches which harmonically match the background music. Figure 5 shows data captured during one experiment and shows the relations between inputs from the touchpad and the acoustical qualities of the synthesized sound.

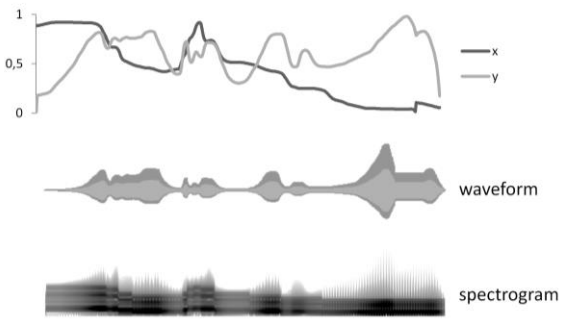


Figure 5. Data captured during a time frame of 12 seconds. The top chart shows values of the x and y coordinates where the user touched the touchpad, while the bottom images represent the waveform and the spectrogram of the synthesized sound.

Case 2: Controlling the 3D graphics and music

The second use case refers to controlling the rendered 3D graphics and music using the orientation of the mobile device. The orientation of the mobile device in the horizontal plane reflects to the orientation of the 3D object in the scene rendered by GEM for Pure Data [20]. If the user tilts the device down, than a violin appears in the scene and violins become prominent in the generated music. The loudness of the violins and the size of the violin object in the scene are determined by the amount of tilt. Similarly, when the user tilts the device to the side, a drum appears in the scene and the drum beat becomes audible. Figure 6 shows several snapshot captured during the experiment.

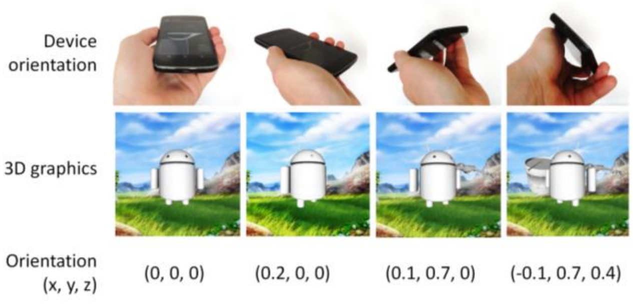


Figure 6. Dependence of the device orientation and the rendered 3D graphics.

Case 3: Demonstration of video features

In order to evaluate the accuracy of the video feature extractor, we conducted several experiments and compared the results with the expected values. Figures 7 do 9 show snapshot captured during the experiments. Below each image there are values of observed video features.

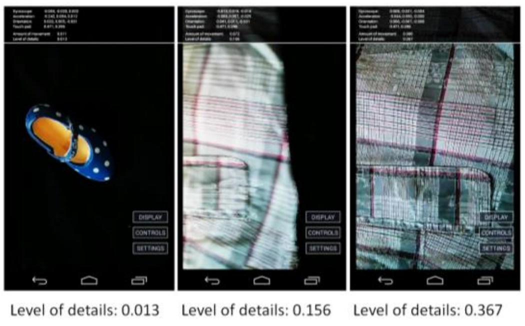


Figure 7. Correlation between the image texture and the level of details. Original image is here displayed with detected edges.

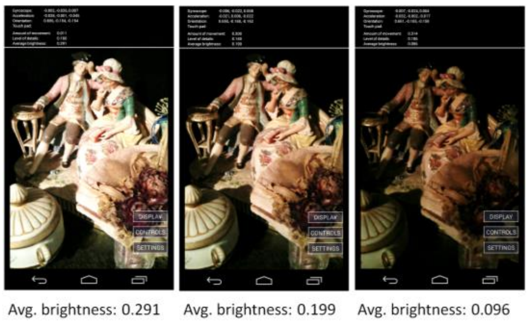


Figure 8. The average brightness in the image.

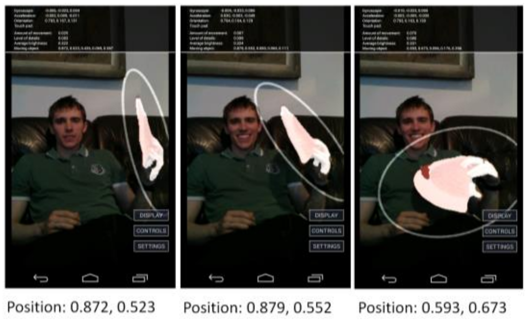


Figure 9. Moving object.

CONCLUSIONS

While most of existing mobile controllers of music and multimedia rely exclusively on the data from input devices and sensors, Sound My Vision additionally extracts video features from the camera image. Having a camera with real-time video analysis in a smartphone provides a lot of benefits starting from accessibility, affordability, and convenience to philosophical and artistic implications in sense of new opportunities of controlling music and multimedia.

In order to ensure usage of the application in various scenarios, the video features were selected to be general, low-level, and low-dimensional.

Based on the conducted experiments, we believe that Sound My Vision can contribute to the popularization of using multimedia in performance arts and for new possibilities in various modalities of artistic expression.

REFERENCES

[1] A. Tanaka, A. Parkinson, Z. Settel, and K. Tahiroglu, “A survey and thematic analysis approach as input to the design of mobile music GUIs,” In Proceedings of the International Conference on New Interfaces for Musical Expression, Michigan, 2012.

[2] T. Kell and M. M. Wanderley, “A high-level review of mappings in musical iOS applications,” In Proceedings of the International Computer Music Conference joint with Sound and Music Computing Conference, Athens, 2014, pp. 565–572.

[3] I. Neuman, C. Okpala, and C. E. Bonezzi, “Mapping motion to timbre: orientation, FM synthesis and spectral filtering,” In Proceedings of the International Computer Music Conference joint with Sound and Music Computing Conference, Athens, 2014, pp. 671–677.

[4] G. Wang, “Designing Smule's iPhone ocarina,” In Proceedings of the International Conference on New Interfaces for Musical Expression, Pittsburgh, 2009.

[5] T. Stool, “SoundScapeTK: a platform for mobile soundscapes,” In Proceedings of the International Computer Music Conference joint with Sound and Music Computing Conference, Athens, 2014, pp. 1731–1735.

[6] J. Cannon and S. Favilla, “The investment of play: expression and affordances in digital musical instrument design,” In Proceedings of the International Computer Music Conference, Ljubljana, 2012, pp. 459–466.

[7] S. Fels, A. Gadd, and A. Mulder, “Mapping transparency through metaphor: towards more expressive musical instruments,” Organised Sound, vol. 7, no. 2, 2002, pp. 109–126.

[8] R. Wechsler, W. Frieder, and P. Dowling, “Eyecon: a motion sensing tool for creating interactive dance, music and video projections,” In Proceedings of the Society of the Study of Artificial Intelligence and Simulation of Behaviour convention, 2004.

[9] G. Kreković and A. Pošćić, “Shaping microsound using physical gestures,” In Proceedings of the International Conference on Computation, Communication, Aesthetics and X, Glasgow, 2015.

[10] J. C. Schacher, “davos soundscape, a location based interactive composition,” In Proceedings of the In Conference on New Interfaces for Musical Expression, Genova, 2008.

[11] H. Zaho et al., “Interactive sonification of choropleth maps: Design and evaluation,” IEEE multimedia, Special issue on Interactive Sonification, vol. 12, no. 2, 2005, pp. 26–35.

[12] E. R. Miranda and M. M. Wanderley, *New Digital Musical Instruments: Control and Interaction Beyond the Keyboard*, A-R Editions, 2006.

[13] A. Camurri, C. Canepa, F. Orlich, and R. Zaccaria, “Interactions between music and movement: a system for music generation from 3D animations,” In Proceedings of the International Conference on Event Perception and Action, Trieste, 1987.

[14] R. Rowe, *Interactive Music Systems*, Cambridge (MA): MIT Press, 1993.

[15] O. Nieto and D. Shasha, “Hand gesture recognition in mobile devices: enhancing the musical experience,” In Proceedings of the International Symposium on Computer Music Multidisciplinary Research, Marseille, 2013.

[16] C. Stauffer and W. Grimson, “Adaptive background mixture models for real-time tracking,” In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2, 1999, pp. 246–252.

[17] J. Canny, “A computational approach to edge detection,” IEEE Transactions on Pattern Analysis and Machine Intelligence, No. 8, Vol. 6, 1986, pp. 679–698.

[18] K. Pulli et al., “Real-time computer vision with OpenCV,” Communications of the ACM, vol. 55, no. 6, 2012, pp. 61–69.

[19] C. Ramakrishnan, “JavaOSC” [*Online*]. Available: http://www.illposed.com/software/javaosc.html Accessed: April, 2015

[20] M. Danks, “Real-time image and video processing in GEM,” In Proceedings of the International Computer Music Conference, Thessaloniki, 1997, pp. 220–223.