Freepad: A Custom Paper-based MIDI Interface

Sungkuk Chun
HCI Lab.
School of Media
Soongsil University
Seoul, South Korea
+82-2-812-7520
k612051@ssu.ac.kr

Andrew Hawryshkewich
School of Interactive Arts & Technology
Simon Fraser University
Surrey, BC, Canada
+1 778.785.0746
aha50@sfu.ca

Keechul Jung
HCI Lab.
School of Media
Soongsil University
Seoul, South Korea
+82-2-812-7520
kcjung@ssu.ac.kr

Philippe Pasquier
School of Interactive Arts& Technology
Simon Fraser University
Surrey, BC, Canada
+1 778.782.8546
pasquier@sfu.ca

ABSTRACT
The field of mixed-reality interface design is relatively young and in regards to music, has not been explored in great depth. Using computer vision and collision detection techniques, Freepad further explores the development of mixed-reality interfaces for music. The result is an accessible user-definable MIDI interface for anyone with a webcam, pen and paper, which outputs MIDI notes with velocity values based on the speed of the strikes on drawn pads.

Keywords
Computer vision, form recognition, collision detection, mixed-reality, custom interface, MIDI

1. INTRODUCTION
Musical interfaces for MIDI are becoming more and more accessible, and with current software, can frequently be used for a variety of different control modalities. This type of accessible and customizable interface is important as it provides musicians, and those exploring musical expression with a customizable form of interaction.

Freepad takes custom interface design one step further, allowing users to draw their own musical interfaces. While simple MIDI trigger pads require users to purchase hardware, Freepad will take a set of shapes on a piece of paper and through using computer-vision techniques, recognize them as interface objects to generate MIDI notes from. The system will work with a variety of shapes and automatically detect them for use as a music interface.

Taking into consideration musical interface design, Freepad focuses on developing an extremely customizable and accessible mixed-reality interface. This system offers the benefits of being able to prototype interface designs, quickly create a performance tool, and all with a webcam, pen and a piece of paper.

2. BACKGROUND
Freepad focuses on three key areas – customization, accessibility and mixed-reality – each of which are discussed to provide scope for the system. In particular, Freepad is compared to a small range of systems each of which embodies some of the elements Freepad is capable of. The Lemur, is an example of a customizable multi-touch piece of music hardware [1]; as it allows users to design and use their own MIDI or OSC (Open Sound Control) based controls on a digital display. Leaning more towards physical gestures approximating musical ones are Wii Music [2] and ZOOZBeat [3], both which use accelerometers in Wii-motes and iPhones to enable musical expression. For example, in the case of Wii Music the percussion playback with the Wii-mote takes on a similar physical action to the playing of a drum, whereas in ZOOZBeat, the shaking of an iPhone influences the pitch of the drum being struck. Moving away from musical interfaces, VooDooSketch is another reference point in some ways closer to Freepad. As an example of a user-definable interface, VooDooSketch will accept drawn interface elements, though at the same time differs by using dials and sliders that can be pinned to a reactive surface to detect their state [6]. This presents some of the similarities to Freepad's user-drawn interface, though further distancing VooDooSketch is its demonstrated application in the context of Photoshop [6].

2.1 Accessibility
Freepad offers a MIDI interface which only requires the use of a webcam, paper and a pen, where the majority of interfaces for the computer require the purchasing of additional hardware to function. In some cases, one is able to employ the computer keyboard as a basic MIDI interface [5], but this does not offer the customizability of Freepad, and could not be struck like the table under the paper on which an interface is drawn.

In comparison with the other examples, Freepad offers a much higher level of accessibility as it uses only one piece of equipment – a webcam – in addition to standard computer hardware. Wii Music requires a Wii and at least one Wii-mote, and ZOOZBeat the ownership of an iPhone. Both Wii Music and ZOOZBeat are also geared for entertainment, not for more rigorous music interface applications. The Lemur requires the purchase of their multitouch interface, and VooDooSketch requires the use of a digital pen (Anoto) and a special conductive mat for assembling the controls on [6].

2.2 Customizability
Both the Lemur and VooDooSketch offer a very high level of user customization: The Lemur provides users with an extensive set of control elements which can be set up however the user sees fit. Similarly VooDooSketch provides elements, but are in this case a mix of physical and drawn elements such as sliders and buttons defined or pinned down by the user. For

NIME2010, June, 2010, Sydney, Australia

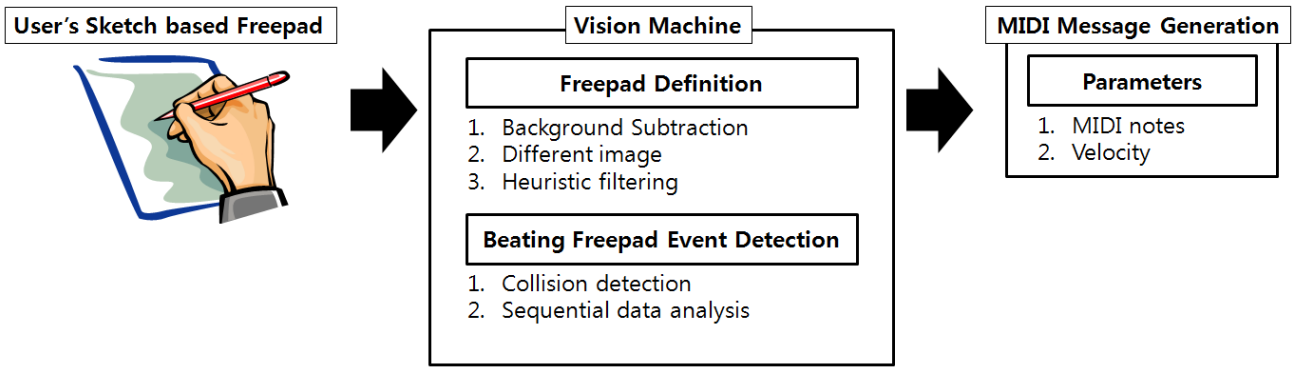

Figure 1. An overview of Freepad's architecture.

customization, both Wii Music and ZOOZBeat offer little control in the type of interactions the user can perform with the phone. At the same time, it is important to keep in mind that both ZOOZBeat and Wii Music are not designed with interface customizations in mind.

Freepad currently offers users the ability to draw their own interface for output of MIDI notes. Though it does not offer the extensive amount of control demonstrated by the Lemur or VooDooSketch, Freepad's further development will involve the integration of more complex MIDI control objects (see Section 4.1.1), beyond its current performance focus.

2.3 Mixed-Reality Interfaces

Mixed-reality interface design is a relatively new field which strives to integrate physical and digital aspects to achieve interactions in a smooth and usable way [7]. The main drive behind mixed-reality interfaces is the need to integrate these two aspects more smoothly, and includes research in tangible interfaces, augmented reality, augmented virtuality and embodied interfaces.

Of this set of systems, the only one which really exemplifies a mixed-reality interface is VooDooSketch. It integrates the ability of users to define their own controls on a physical object, and then use them as an interface to digital software. Similarly, Freepad seeks to achieve this kind of balance between the physical and digital by using common physical elements – paper and pens – to generate an interface for playing MIDI notes on. Unlike other computer-vision based systems, Freepad also does not require the use of fiducial markers to identify where pads have been drawn.

3. SPECIFICATIONS

As has been discussed, Freepad works with user-drawn interfaces to generate MIDI notes using variables such as which form – known as pad from now on – is being struck and the speed of the hit. To determine these variables, the system uses computer vision techniques such as background subtraction [8], connected component labeling [9], image moments [10], and collision detection [11] by employing an OpenCV library [12]. These techniques allow the system to parse all closed shapes and assemble the interface in three stages (overview in Figure 1): Drawing the pad, analyzing the real-time video stream, and generating MIDI messages.

3.1 Freepad Setup

In Freepad, a user can draw multiple pads without limitations to what shape or number can be drawn. Outside of these elements, there is only one limitation, being that the drawn pads must be within the visual space of the webcam being used. Figure 2 shows the setup and drawing the virtual pads with the webcam.

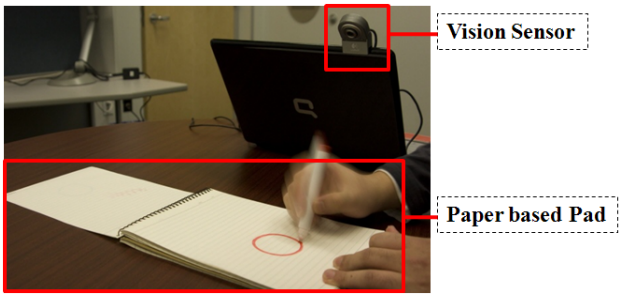

Figure 2. Setup and drawing of virtual pads.

Freepad works with any surface on which a user could draw pads to form their interface. The pads can then be played either with hands or 'sticks', such as a pair of pens. For each detected hit the system outputs a MIDI note message, which can then be routed into software or hardware synthesizers or samplers. Within the area viewed by Freepad, each hit is determined using collision detection algorithms detailed in the next section.

3.2 Pad Definition & Collision Detection

The computer vision component of Freepad deals with two processes: 1. Defining locations of drawn pads, 2. Detecting pad hits, which includes calculating the speed of the hit. To extract multiple pads, the system applies background subtraction and heuristic filtering based on connected component labeling [9], and image moments [10] techniques which are both described below. Then, it continuously surveys the webcam input to detect hits (collisions) within the area of each pad. Additionally, if an event occurs, Freepad calculates the properties of the detected collision: Such as which pad was struck and the speed of the hit with which it generates an appropriate MIDI message.

3.2.1 Defining Pads

Interfaces drawn for Freepad are not expected to be drawn the same way twice, and are therefore not restricted to one shape. The interface itself is entirely dependent on how the user draws it. To deal with different shapes, the system extracts the drawn regions using the computer vision methods of background subtraction and heuristic filtering.

Background subtraction is a technique used to sort out objects of interest in a scene and involves comparing an observed image with a modeled background image. This method looks for and assesses if any objects of interest exist using comparison algorithms to model the background against the observed image.

Freepad first constructs background image. The pixels of which are modeled by a single value, and updated by:

$$B_t = (1 - \alpha)B_{t-1} + \alpha I_t \quad (1)$$

B_t is a background pixel and I_t is an input pixel at time t. α is a value dependant on how the background is modeled, and is kept small to prevent artificial noises which are contained in the input image. In Freepad, α is assigned 0.01, to reflect that Freepad use the average of 100 frames as the background image.

In Figure 3, the difference image (Figure 3(c)) is calculated by subtracting the initial pad image (Figure 3(b)) from a background image (Figure 3(a)), which is an average of 100 frames worth of images before the user has drawn an interface. As shown in Figure 3(c), the difference image contains not only the drawn pads but also some background noise which appears as the edge of the pad of paper.

To reduce the noise in the image, heuristic filtering is applied based on the geometric information obtained from connected component labeling and image moment techniques. Connected component labeling is used to detect connected regions in binary digital images, and enables Freepad to detect closed shapes. Image moment techniques are used to describe the basic properties of objects such as the centroid, width, height, density, and orientation of the shape. From these properties, Freepad calculates the number of pixels and the aspect ratio that is the ratio of the width of the shape to its height, and then cuts out the drawn shape if the shape is too small or the of width and height are extremely different. The used minimum number of pixels is 100, and the aspect ratios are 10:1 and 1:10. Nearing completion, the system reduces the noise and defines the index of the shapes using image moments on the detected closed shapes. The detected pad image appears as shown in the final frame of Figure 3(d).

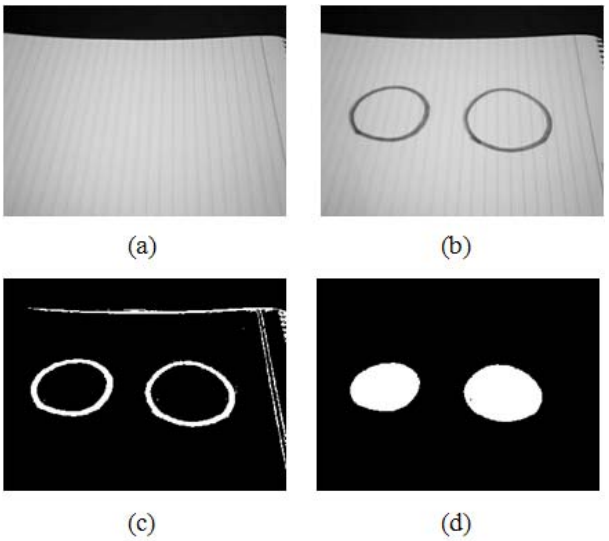

Figure 3. Result of defining pads; (a) background image, (b) initial pad image, (c) difference image, (d) final pad image.

3.2.2 Collision Detection

After defining the pad regions of the input interface, Freepad has to detect when and how collisions occurs. In Freepad, a collision occurs when the user's hand, finger or some form of stick hit the defined pad. Note that the system uses one camera without image calibration and the captured image is two-dimensional. However, the real space in which virtual pads are played is three-dimensional, which means that Freepad needs to be able to detect the three-dimensional collision event using a sequence of two-dimensional reference images.

Figure 4. Distance between top of the pad's region and bottom of the stick.

We assume that a collision occurs when the object striking the pad appears in the pad's region. In Freepad, the movement of the object is detected by tracking the distance between the top point of the region and the bottom point of the striking object in the set of collision detected frames. At the point which the distance is determined to be the maximum among that, the collision event is generated.

Collision Detection Pseudo Code

Variables

1. (x_t, y_t) : top point of the pad's region.
2. (x_i, y_i) : bottom point of the stick.
3. d_i : distance between (x_t, y_t) and (x_i, y_i) at time i.
4. T_s : time of collision beginning.
5. d(i) : one dimensional discrete function, $d(i) = d_i$.
6. δ(i) : differentiation function.
7. collision_begin : flag to represent beginning of collision.
8. T_{peak} and T_{adj} : threshold value. Local maximum which is more than T_{peak} are detected by finding zero crossings of the difference function δ(i). And the detected local maximum is far from other local maxima more than T_{adj}.

Pseudo Code

Begin

For(i = 0 ; ; i++){

Compute distances $d_i = \sqrt{(x_i - x_t)^2 + (y_i - y_t)^2}$.

IF($d_i > 0$),

Then collision_begin is true, $T_s = i$.

IF(collision_begin is true),

Then

Compute $\delta(i) = d(i) - d(i-1)$

If($\delta(i) == 0, d(i) > T_{peak}$ and $T_s - i > T_{adj}$),

Then

collision occurs at time i.

velocity is decided based on $1/(T_s - i)$ and $d(i)/(T_s - i)$.

}

End

Figure 5. Algorithm for collision detection

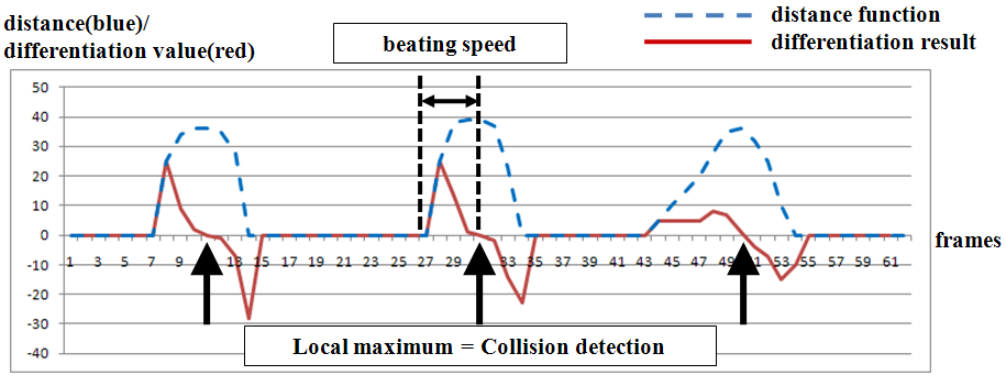

Figure 6. Collision detection using local maximums; distance function (blue), differentiation result (red), collision point (local maximum; arrow), and beating speed (duration of collision).

To detect collision events, Freepad first calculates a subtracted image, which is the difference between the input image and the initial pad image (Figure 3(b)). It then multiplies the subtracted image by the final pad image (Figure 3(d)). Then a subtracted image of the pad's regions is obtained (Figure 4). From the pad's subtracted image, the system calculates the distance between the top of the pad and the bottom of any colliding element within the pad's region. Feepad tracks the distances of every frame in the distance function, and collision is detected at the local maximum in the distance function. This maximum is found by extracting the zero-crossing point in the differentiation of distance function. The collision detection algorithm used in Freepad is detailed in Figure 5, and Figure 6 illustrates the detection of pad hits using a distance function.

As shown in Figure 5, the system uses the thresholds of T_{peak} and T_{adj} to prevent faulty local maximum detection, because finding local maximums by differentiation makes it susceptible to noise in the sequence of data. For example, if the stick only appears in one frame or some camera noise suggests the movement of a stick, the system may misunderstand it as a collision event. Additionally, noise located closely to the local maximum can result in two collision events from one hit. To reduce these faulty detections, the system recognizes collisions only if the distance is over T_{peak}, or the detected local maximum is far from other local maxima more then T_{adj}. Through testing T_{peak} has been given a value of 10, and T_{adj} is defined by the user, since it is related to collision detection sensitivity (see Section 3.4). Small T_{adj} values enable Freepad to capture faster hits, whereas, large T_{adj} values are capable of reducing faulty collision detections from the camera noise.

As shown in Figure 6, the local maximum can be found from calculating the zero-crossing point in the differentiation of distance function. Also, the speed of the hit is based on the amount of time and the average distance which transpires from the beginning of a collision to the local maximum occurring in the distance function. For example, if the number of frames is smaller and the average distance is higher, the speed would be higher accordingly. Consequently, the index of a beaten pad and speed of the hit is used for the generation of MIDI messages, which is explained in the next section.

3.3 MIDI Message Generation

Using MIDI for output provides users with a variety of options regarding design and use of a custom interface for music. The ability to select a variety of synthesizers, samplers or to even patch the data into other applications such as Max 5 or PureData could be extremely beneficial to interested users. Considering these advantages of MIDI, Freepad provides MIDI note output with two parameters; a note-value and velocity based on the properties of the detected collision.

Table 1: MIDI Message properties and parameters

Collision property	Parameter in MIDI message	Range
Pad index	Note-value	0~127
Beating speed	Velocity	0~127

Table 1 shows the parameters used in MIDI message generation. The note-value is the pitch at which a note-on and a note-off pair occur, and velocity is equal to the volume of a note. Both are respectively determined by the index and speed of the pad hit. For example, if the detected pad hit is at index 0, and the speed of the hit is moderate, the output note would have a pitch equal to that which has been set by the user (see Section 3.4) and a velocity around 60. With regards to velocity, if user hits the pad faster, the output velocity increases accordingly.

3.4 GUI

The GUI for Freepad focuses on easing setup of a newly drawn interface, so it may be as quick and straightforward as possible. The result is a three-step setup process to ensure that the interface works well and consistently for users. The steps are: Setting the background, parsing the interface and playing it, in addition to a preference panel for additional setup.

The first step in the process requires the user to establish the background image which Freepad uses to discern its interface from. The only requirement is that the user places the desired pad of paper in front of the camera, and clicks to set the interface (Figure 7). Once this background image is set, the paper cannot be moved or the system will lose the ability to track any of the drawn pads.

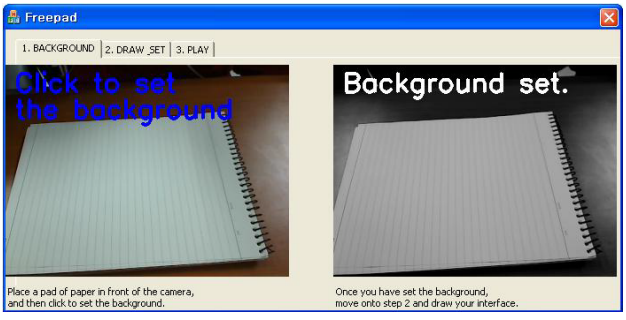

Figure 7. 1st step, setting a background image.

In the second step the user is actually drawing and then setting the interface. The system works with shapes such as rectangles or ellipses, and cannot detect shapes which are too small, long, or not closed (see Section 3.2 for details). Similar to the first step, once the interface is drawn, the user simply needs to click to set the interface. Freepad then prompts the user with an image of the detected forms to indicate that it has detected an interface to play (Figure 8).

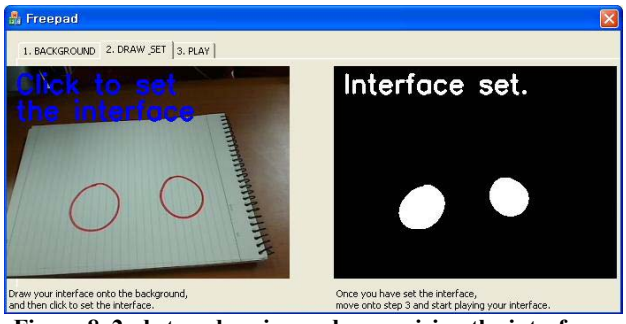

Figure 8. 2nd step, drawing and recognizing the interface.

The final step lets the user start playing their custom interface. Within Freepad is indicated the pad that has been struck, and the velocity at which it was struck. To the right of the visual indicators, there are also a set of controls for setting up the MIDI output of the system (Figure 9). MIDI controls available to the user include setting a pitch for each pad, instrument values, and a MIDI channel. The final control available is a three-level sensitivity setting for collision detection, which adjusts the T_{adj} value discussed in Section 3.2.2.

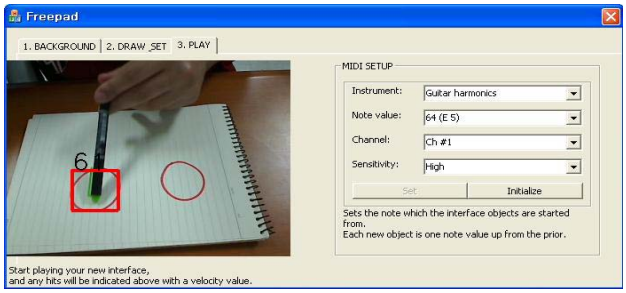

Figure 9. 3rd step, playing the new interface.

The final portion of the GUI which is not presented above is the preferences for Freepad. These preferences currently offer the user the option of adjusting camera settings such as frame-rates.

3.5 Current Limitations

As Freepad is the first iteration of a mixed-reality interfacing system for music, there are obviously some limitations which should be addressed. In particular, since the system works with video input, there are latency issues that warrant discussion.

3.5.1 Latency

As a result of the speed with which the system can parse the information from the video, there is latency between 30-90 milliseconds (ms) between when the user strikes the pad, and Freepad sends a MIDI signal. In terms of usability, the latency issues are an important consideration, and an issue that can be reduced to about 30-40 ms with webcams capable of higher frame-rates than the standard 24-30 frames-per-second (fps). So Freepad was tested 100 times with a webcamera set at four different frame-rates, and the averages and SDs of latency were calculated for each frame-rate. As shown in Figure 10, the average and SD gradually decline as the frame-rate increases. The small average and SD demonstrate Freepad's ability to quickly respond to a pad being struck, and the stability of the system. It is also reasonable to suppose that higher frame-rates would garner an improvement in collision detection.

With a 30-40ms latency, Freepad falls into the approximated 30-40ms threshold, at which the latency between action and sound is not perceived [13]. Below is a figure illustrating the changes of average latency and the standard deviation (SD) on various frame-rates.

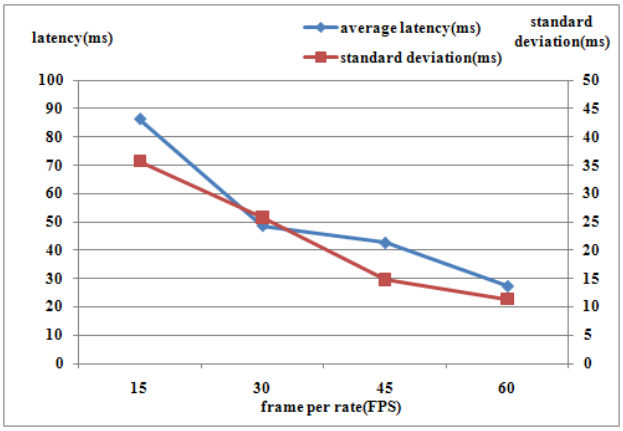

Figure 10. Changes of average latency and standard deviation dependent on camera's frame-rate.

3.5.2 Web-Camera Frame-Rate

Due to the fps of standard webcams sitting between 24 to 30 FPS, Freepad cannot always read pad hits as quickly as the user may perform them, as the action could occur between frames and is never in fact 'seen' by the system. This hinders the usability of Freepad for more common webcameras, but can be easily remedied with higher frame-rate webcams, which are slowly becoming more common. To keep up with ever improving video technologies, webcamera specifications are starting to provide more motion sensitive frame-rates of up to 90 fps in consumer-grade cameras [14]. Additionally, we experimented with the recognition rate of user's hits in contrast with changing frame-rates on the webcam. The results are shown in Table 2.

Table 2: Recognition rate of user's hitting on different frame-rates.

Frame-rate(fps)	**15**	**30**	**45**	**60**
Recognition rate(%)	45	88	91	96

Freepad was tested 100 times at each frame-rate with randomly assigned hitting speeds to check whether the system generated a MIDI signal or missed it. The result of the experiment showed that Freepad works better with the higher frame-rate cameras as expected.

These current limitations also present a possible direction for the future development of Freepad which involves furthering Freepad's abilities as a MIDI controller. This is due to the fact that the latency with which Freepad responds would not be as imperative when employed as a controller as compared to its use as an instrument interface.

4. FUTURE WORKS

Freepad is a first step towards user-definable mixed-reality interfacing in music, and as a result, offers a variety of areas in which it can be developed. There are two areas of development which involve Freepad: Control and research.

4.1 Further Control in Freepad

Currently, Freepad only offers users the ability to draw simple shapes and use them as a form of pad-based MIDI controller or note interface. There are two specific areas which this paper identifies as means of improvement for Freepad: Control Types and Playback Styles.

4.1.1 Control Types

Rather than working with basic sliders, the inclusion of other control types would be a worthwhile avenue of exploration. Including control elements such as buttons or dials similar to VooDooSketch [6] would make for an extremely portable and customizable MIDI interface, and an interface which could be entirely scalable based on the application. Rather than carrying around large MIDI devices – each for its own set of controls – Freepad offers the possibility of simply drawing out an entire interface for use in performance or otherwise.

4.1.2 Playback Styles

In addition to developing on control abilities, different playback styles or shapes could be worked into Freepad. For example, when one plays a hand drum such as a djembe, the drum can be struck in the middle for lower pitches, or near the edge for a higher pitched sound. Exploring and integrating similar sorts of mappings could easily be applied to Freepad based on the distance of the strike into the drawn pad. This offers further benefits in the system possibly having more refined controls for fine or detailed musical applications.

4.2 Mixed-reality interface research

Freepad offers users a mixed-reality interface for musical expression, an area which has not been looked at in depth with relation to musical interaction. Research comparing how users perceive and interact with mixed-reality interfaces such as Freepad to their singular-reality counterparts such as a MIDI trigger pad would be the next step. This would be beneficial in helping understand what further benefits mixed-reality interfaces offer users, apart from those already discussed or readily apparent.

4.3 Further Uses

The customizable nature of Freepad also offers possibilities for broader applications. Freepad could offer a quick method for custom musical interface prototyping, a means for designing audio controller layouts, or branching outside of music, a custom video game controller. Camera-based interfaces such as Freepad can offer a lot of flexibility in the realm of quick and easy interaction.

5. CONCLUSION

Freepad contributes to the development of mixed-reality interfaces for musical expression. It offers users an accessible and customizable interface which could be employed with a wide variety of synthesizers and audio software due to its MIDI output. It is only of further benefit in its sheer accessibility by only requiring a webcam to create a functioning interface for the user. As Freepad is currently limited somewhat in real-time interaction due to latency, future development of it as a control interface will enable users to develop their own MIDI control interfaces without the need for substantial hardware. Freepad develops a new way to approach customizable and accessible music interfaces for any types of users.

6. ACKNOWLEDGMENTS

This research was conducted with the support of the Natural Sciences and Engineering Research Council of Canada, in addition to the National Research Foundation of Korea (NRF), Grant (NRF-2008-000-10175-0), and the Ministry of Culture, Sports, and Tourism, Korea, under the CTRC (Contents Technology Research Center) support program supervised by the KOCCA.

7. REFERENCES

[1] "JazzMutant - Lemur, multitouch modular controller for sequencers, synthesizers, virtual instruments, vj and light." Accessed February 1, 2010. <http://www.jazzmutant.com/lemur_overview.php>.

[2] "Wii Music | Nintendo." Accessed February 1, 2010. <http://www.wiimusic.com/>.

[3] G. Weinberg, A. Beck and M. Godfrey. "ZooZBeat: a Gesture-based Mobile Music Studio," *Proceedings of the 9th International Conference on New Interfaces of Musical Expression (NIME)*, Pittsburgh, PA, USA: 2009, pp. 312-315.

[4] F. Block, C. Gutwin, M. Haller, H. Gellersen, and M. Billinghurst, "Pen and paper techniques for physical customisation of tabletop interfaces," *3rd IEEE International Workshop on Horizontal Interactive Human Computer Systems,* 2008, pp. 17–24.

[5] "Sweet Little Piano / A MIDI computer keyboard." Accessed February 1, 2010. <http://www.ronimusic.com/sweet_pi.htm>.

[6] F. Block, M. Haller, H. Gellersen, C. Gutwin, and M. Billinghurst, "VoodooSketch: Extending Interactive Surfaces with Adaptable Interface Palettes," *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction (TEI)*, New York, NY, USA: 2008, pp. 55–58.

[7] E. Dubois, P. Gray, and L. Nigay, eds., *The Engineering of Mixed Reality Systems*, London: Springer, 2010.

[8] A. McIvor, "Background subtraction techniques," In *Proceedings of Image and Video Computing*, 2000, pp. 147–153.

[9] H. Samet and M. Tamminen, "Efficient Component Labeling of Images of Arbitrary Dimension Represented by Linear Bintrees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, 1988, pp. 579-586.

[10] C. Teh and R. Chin, "On Image Analysis by the Methods of Moments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, 1988, pp. 496-513.

[11] S. Kawabata, S. Hiura, and K. Sato, "3D Intrusion Detection System with Uncalibrated Multiple Cameras," *Lecture Notes in Computer Science*, vol. 4843, 2007, p. 149-158.

[12] "Welcome - OpenCV Wiki." Accessed February 1, 2010. <http://opencv.willowgarage.com/wiki/>.

[13] D.J. Levitin, K. Maclean, M. Mathews, L. Chu and E. Jensen, "The Perception of Cross-Modal Simultaneity," *Computing Anticipatory Systems Conference*, vol. 517, no.1, 2000, pp. 323-329.

[14] "Philips - Webcams - Webcams with headsets - Headsets - Notebook webcam - Webcams and headsets - PC products and phones." Accessed February 1, 2010. <http://www.consumer.philips.com/c/webcamsheadsets/13022/cat/gb/>