Digito: A Fine-Grain Gesturally Controlled Virtual Musical Instrument

Nicholas Gillian
Responsive Environments
MIT Media Lab
75 Amherst Street, E14-574B
Cambridge, MA 02139, USA
ngillian@media.mit.edu

Joseph A. Paradiso
Responsive Environments
MIT Media Lab
75 Amherst Street, E14-548P
Cambridge, MA 02139, USA
joep@media.mit.edu

ABSTRACT

This paper presents Digito, a gesturally controlled virtual musical instrument. Digito is controlled through a number of intricate hand gestures, providing both discrete and continuous control of Digito's sound engine; with the fine-grain hand gestures captured by a 3D depth sensor and recognized using computer vision and machine learning algorithms. We describe the design and initial iterative development of Digito, the hand and finger tracking algorithms and gesture recognition algorithms that drive the system, and report the insights gained during the initial development cycles and user testing of this gesturally controlled virtual musical instrument.

Keywords

Gesture Recognition, Virtual Musical Instrument

1. INTRODUCTION

Advances in sensing technology, gesture recognition algorithms and personal computational resources are slowly enabling gesturally controlled electronic instruments to become feasible for the everyday musician. Gesture recognition facilitates many new musical-interaction paradigms, such as enabling a musician to simultaneously interact with a machine *and* other performers with one succinct movement. It also facilitates a performer to extend and dynamically manipulate an existing musical instrument by using expressive gestures to digitally modify the real-time acoustic signals generated by the instrument [15]. In addition, gesture recognition also enables a performer to play what Mulder refers to as *Virtual Musical Instruments* (VMIs) [12]; that is, digital instruments analogous to physical musical instruments, but controlled exclusively via gestural interfaces. Further, gestural interfaces allow a performer to simultaneously control multiple instrumental parameters, such as pitch, onset attack, etc., and apply intricate many-to-many mapping techniques. Complex, high-dimensional, mapping techniques can be difficult even with more conventional interfaces [17], thus making gestural interaction a powerful tool for musician-computer interaction (MCI).

Designing a viable gestural controller is, however, by no means a rudimentary task. Choosing from the infinite combinations of which gestures to use, how these should be mapped and what sounds they should be mapped to are all integral features that result in the success, or indeed failure, of such an interface. How then should one approach the design of a gestural controller? What are the differences between designing a gestural controller for a musical interface, and designing a gestural controller for a conventional, human-computer interaction task, such as navigating through a user's photo library? What audio, visual and haptic feedback does such an interface require and exactly how expressive, or indeed practical, can a gestural musical controller really be? This paper describes the design and initial iterative development of Digito, a gesturally controlled VMI which is currently being applied to help answer such questions.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
NIME'12, May 21 – 23, 2012, University of Michigan, Ann Arbor.
Copyright remains with the author(s).

2. DIGITO

Digito [1] is a gesturally controlled virtual musical instrument that can be played through a subset of subtle hand gestures. A performer can trigger a note to be played in Digito by using a 'tap' style gesture with the tip of the index finger of their right hand. This note-on gesture is performed in free-space. The performer can gesturally tap anywhere in mid-air, rather than being constrained to touching a physical surface, such as a table, wall or touchscreen. The note-on gesture is affected by both the three-dimensional location of the gesture and how the gesture is performed, with the 3D position of the tap controlling the frequency - and the velocity of the tap controlling the amplitude and timbre - of the note to be triggered. In addition, the performer can continually affect the timbre of the note they have just played after they have triggered it through a number of continuous hand gestures. To aid the user in navigating Digito's virtual space, visual feedback is provided. This feedback represents the 3D virtual space in which the VMI exists, and provides important visual feedback on the current tracking status of the system (see figure 1b). The 3D virtual space has been created with a number of 'layers', with each layer containing twelve equally spaced targets arranged in a circular distribution (see figure 1a). Each layer represents an octave, with each target representing a note. A performer triggers a note to play by gesturally tapping any of these virtual targets. A performer can dynamically move through each layer by 'reaching' into the screen to access the next layer, or 'retracting' away from the screen to recall the previous layer.

2.1 Sound Engine

Digito's sound engine is built in SuperCollider[2] using frequency modulation (FM) synthesis [3]. A performer's gestures are used to control the carrier frequency, modulation

[1]`www.nickgillian.com/research/projects/digito`
[2]http://supercollider.sourceforge.net/

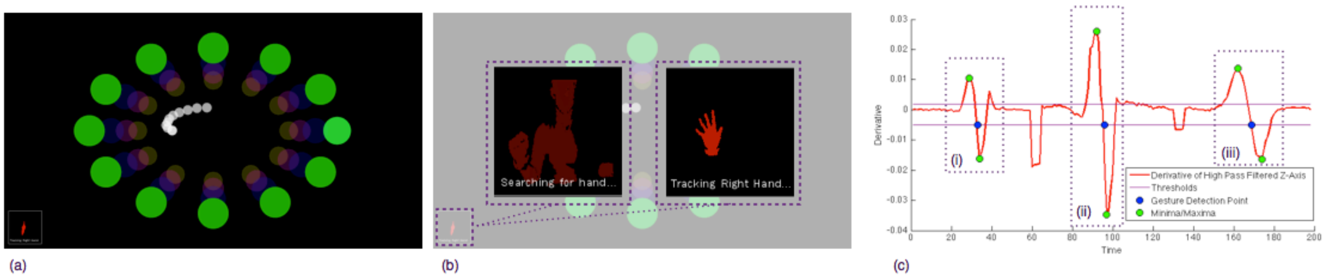

Figure 1: (a) Digito's main visual feedback illustrating the continuous gesture feedback and 12 notes of the current layer, with several layers visible in the distance, (b) the enlarged tracking status feedback, (c) the peak detection algorithm used to detect a tap gesture, showing a (i) soft tap, (ii) fast hard tap, (iii) slow soft tap

frequency, modulation index and two ADSR[3] envelops that drive the FM synthesis algorithm. These envelops are used to control the index of the modulation waveform component of the FM synth; along with controlling the amplitude of the synth's carrier waveform. The velocity of a performer's tap gesture is mapped to control the attack and decay elements of the ADSR envelopes. The sustain and release components are controlled by a user's continuous hand movements; performed after a note has been triggered using the tap gesture. A performer can also use their continuous hand gestures to change the modulation index and manipulate the modulation frequency parameters of the FM synthesis algorithm. This enables a performer to trigger a note, that can initially start with a simple timbre, similar to that of a resonating glass, and then modify the timbre of the note, using continuous hand gestures, to create a wide range of more complex timbres. With this interaction technique, a performer can gain dynamic control over a large number of Digito's settings, using a small subset of intuitive, subtle, hand gestures.

Digito provides a performer with two mapping options after they have triggered a note. The first is an automatic mapping in which the sustain and release parameters of the ADSR envelopes controlling the FM synthesis engine for the triggered note are automatically set in relation to the velocity of the performer's tap gesture. The second option is a continuous mapping of the sustain, release, modulation index and modulation frequency parameters, controlled by the performer in real-time using a number of fine-grain hand gestures. A performer can dynamically select either option by varying the movement of their hand just after they have triggered a note. If they remain relatively still or move into the circular distribution after hitting a note then the first mode will be triggered. Alternatively, moving their hand away from the circular distribution will give the performer continuous control over the FM synthesis engine. The performer can then move their hand into the center of the circular distribution and continue to perform a number of fine-grain hand gestures, controlling the sustained note for as long as they continue to gesticulate. In addition, the performer can also continue to trigger other notes, creating chords, with the entire chord being controlled by the user's continuous hand gestures. The note(s) will finally be released when the performer stops moving their hand. A note or chord can also be rapidly stopped by performing a closed fist gesture.

2.2 Visual Feedback

The visual component for Digito, built using openFrameworks[4], represents the 3D virtual space in which the VMI exists. This 3D virtual space contains six layers (representing octaves) each containing twelve targets (representing notes), equally distributed around the exterior of a large circle (or ellipse, depending on the screen setup). In the continuous mapping mode, specific visual feedback is provided to the performer, in the form of a continuously evolving trajectory trail that appears to extend from their triggering finger tip (see figure 1a). Visual feedback is also provided to inform the user of the current tracking status of the system, with the user's entire body being displayed until their hand is detected, after which just their segmented hand will be shown (see figure 1b).

3. HAND & FINGER TRACKING

To make this VMI a reality, we required a method of sensing a user's fine-grain hand gestures, i.e. detailed information such as the position of the finger-tips or if the hand is open or closed. We chose to avoid constraining the user to wear sensor gloves or colored markers, as have been used in previous systems [11, 16]. This was to enable the musician, if required, to simultaneously play other instruments while using the VMI. We therefore developed a form of markerless computer vision to facilitate fine-grain hand control. Hand tracking and the recognition of hand gestures are well established fields within the computer vision and machine learning literature [4, 14]. Hand tracking via computer vision has also been applied to several musical controllers, including music software controlled exclusively via hand and finger motion [10], or the tracking of a guitarist's [2] or violin player's [18] left hand finger position for computer-aided pedagogical tools.

Digito uses a Microsoft Kinect[5] depth camera to sense a user's gestures. A number of libraries[6] and SDKs[7] are available for interfacing with the Kinect. Although some of these resources feature skeleton tracking algorithms which provide an estimation of the 3D position of the user's hands, they do not provide any fine-grain information. A number of fine-grain hand and finger tracking algorithms have recently been developed [7, 13]. However, such algorithms are computationally intensive and currently function at a maximum recognition rate of $\sim$ 15Hz, even when the algorithms are heavily optimized and running on a machine's GPU. Therefore, to achieve a useable compromise between fine-grain tracking and low-latency, we adopted a similar approach to that used by several authors [5, 8, 9], in which the hand and fingers are tracked using common 2D computer vision techniques.

The input to Digito's hand tracking system consists of a 640 x 480 depth image which is streamed from a Kinect at

[3]ADSR: Attack, Decay, Sustain, Release
[4]http://www.openframeworks.cc/

[5]`http://www.microsoft.com/en-us/kinectforwindows/`
[6]`https://github.com/ofTheo/ofxKinect`
[7]`http://openni.org/`

30 frames per second. An initial two-fold depth thresholding operation is performed on the raw depth image, removing any depth pixels that do not fall within the desired field of interaction (FOI). This process performs a foreground/background subtraction task. Following the depth thresholding operation, remaining objects in the FOI are then grouped using blob tracking. Unsuitable objects, such as those that are too small or large to be a hand, are removed and the remaining objects are processed to find a potential hand. If a potential hand is found then each finger is classified, labelled and tracked, with the position and orientation of the hand and visible fingers being used as input to the gesture recognition system. A critical phase of the system lies in the finger tracking and estimation stages. This phase of the system is key as any tracking or estimation errors made here will be propagated throughout the remaining stages; resulting in either legitimate gestures being missed or non-gestural movements being falsely recognized.

The hand tracking process is segmented into two states, an initial search phase in which the user's hand is located in the continuous input depth image stream, and a tracking and estimation phase, in which the fine-grain hand movements of the user are tracked over time. For both states, each potential hand object is scanned for the presence of cylindrical-like regions which could indicate potential fingers. Neighboring cylindrical-like regions are combined to form possible finger tip locations. If the search state is in the initial search phase then an object will not be classified as a hand until five finger-tips are recognized within the same image frame. If a hand has been found then the system labels each finger (to later allow specific key fingers such as the index finger to be identified) and increments the hand tracking process from the initial search phase to the tracking and estimation phase.

In the tracking and estimation phase, the system tracks the fine-grain movements of the hand, propagating the predicted 3D position and orientation of the fingers and hand through to the system's gesture recognition phase. To enable continuous tracking of the hand, as well as continuous estimation and classification of each visible finger, we employed a tracking and prediction algorithm similar to that used by Argyros et. al. [1]. This online adaptive tracker dynamically creates a number of discrete models, one for the hand and five for the fingers, when the hand is first classified in the initial search phase. The models, which are built using the position, orientation and area of each hand or finger, are then used to continually track the hand and label each visible finger by using the model estimated in the previous frame to predict the position of each tracked object in the current frame. This enables the user to freely relax individual fingers or open and close their hand, while still retaining track of the hand and being able to classify each visible finger.

4. GESTURE RECOGNITION & MAPPING

Digito's note-on events are triggered by the user performing a subtle tap-style gesture at various locations in space with the index finger of their right hand. This tap gesture is recognized using a peak detection algorithm that searches for rapid changes in the z-axis (which represents the distance of an object from the camera) of the tip of the index finger. The peak detection algorithm works by using a high pass filter to suppress slow changes in the z-axis data. The first derivative of this filtered data is computed and a search algorithm continually scans the derivative signal until it finds a positive, then negative threshold crossing, which must both occur within the same search window, as illustrated by figure 1c. If both threshold crossings are detected, a peak, i.e. tap gesture, is considered as recognized. The amplitude of the tap is then mapped to control the velocity and timbre of the triggered note.

4.1 Layer Transition Classification

A user can move through each of Digito's layers by performing a 'reach' or 'retract' style hand gesture, which will move the system to the next or previous layer respectively. These gestures are recognized using Dynamic Time Warping [6], with the number of layers transitioned being proportional to the amplitude of the gesture. This enables a user, for example, to move to two layers above the current layer by reaching further into the space.

4.2 Continuous Mapping

Digito continuously computes two values that are then mapped to control various parameters of the FM synth. These values are the *movement index* and the *contraction index*. The movement index expresses how vigorously the performer is moving their hand. The contraction index, given by the normalized distance between a user's thumb and little finger, expresses how open or closed a performer's hand is.

The movement index is computed by storing the previous n position coordinates for the centroid of a user's hand in a circular buffer. At each frame, the contents in the first and latter halves of the buffer are averaged to create two 3D points (p_1, p_2). A leaky integrator is then used to provide some 'memory' of a performer's hand movements, computed by: $y_t = (\kappa\, y_{t-1}) + d_t$; where κ is the leak rate, in the range [0 1.0], and d is the distance between the points p_1 and p_2 at time t. The output of the leaky integrator, y_t, is limited to provide a maximum movement value, y_{max}, which is also used to normalize y_t. The normalized movement index is then mapped to continuously control the modulation index and modulation frequency of the FM synth. The leaky integrator provides a useful mapping technique, as it allows the user to quickly build up some energy in the instrument, which corresponds to more complex spectral timbres as the modulation index and modulation frequency of the FM synth increase; which then slowly return to their original values as the energy 'leaks' from the system as the performer either slows down their movement, or alternatively, stops moving altogether.

5. DISCUSSION & CONCLUSION

To help guide the design and future direction of this VMI, ten users, the majority of which had some musical training, were taught to play the instrument over a 30 minute exploratory session and provided some informal feedback. A number of important observations were noted across almost all test users. The first was the significance of the continuous control gestures. Test users reported that these gestures "*felt fluid and natural*" and allowed them "*to really explore the wide range of possible sounds of the instrument*". A second key observation was the importance of the visual feedback. All users found that the visual feedback was essential to enable them to trigger a note and to understand what tracking state the system was in. Interestingly, the visual feedback was not as important for the continuous control gestures. One possible reason for this is that the audio feedback for the continuous gestures independently provides sufficient information. The discrete trigger gesture, however, provides no audio feedback until the user performs a successful tap gesture.

The majority of users were able to quickly learn how to control the various aspects of the instrument, such as tapping a note, gaining continuous control and moving between layers, within the first few minutes of playing the instrument. One user was even able to play "twinkle twinkle little star" after just two minutes of using the interface. Moreover, the majority of users were able to continue to refine their skills in playing the gestural instrument over the 30 minute session. Users were able to learn the subtleties of the amplitude control for the tap gesture along with learning the continuous control gestures to achieve specific timbres. They were also able to perform complex exercises, such as allowing the modulation frequency to drop to a point were it produced a vibrato style effect on the modulation index, and then subtly moving their hand just enough to sustain this effect, without the note being released due to a small movement index value.

It should be noted that one disadvantage in using hand gestures to control a VMI, such as Digito, is arm fatigue, which was common across a significant number of participants after using the system for just a few minutes. This was somewhat alleviated by ensuring that the user's FOI was not calibrated too high, which resulted in the user having to extend their reach to trigger the top-most notes, and that the participant was able to rest their elbow on a chair arm-rest or other body part, rather than having to continually hold their arm elevated and unsupported. While there is commonly some discomfort whilst learning to play acoustic musical instruments as the player develops the required tendon and muscle strength, such as learning how to play chords on a guitar for the first time, arm fatigue could be mitigated by more powerful tracking algorithms that allow the user to interact with the system, with their hands in any location, such as close to their torso, or orientation, i.e. not necessarily orthogonal to the tracking system. We are currently developing algorithms that mitigate these interaction limitations.

So what have we learned through the process of designing this instrument? Foremost, we discovered that, although the classification of free-air gestures can emulate discrete trigger events, such as Digito's tap gesture, the dynamics for gesturally musical interaction fundamentally change when gesture recognition is used to enable a performer to gain continuous control over an instrument, parameter or effect. This is especially true when gestures provide a performer with access to intricate, high-dimensional, many-to-many forms of control. We also learned that while a number of gestures could be used to control virtual buttons and sliders, similar to those found on many physical MCI devices, simply mirroring the functionality of an existing hardware controller in a virtual space using gestural control may not be the most appropriate application of such technology. The task and context in which gestural interaction is applied is therefore important; in the same way that a keyboard and mouse are more applicable interaction devices to use when writing a long essay, compared with using a gesturally controlled 'air' keyboard. One key lesson that we learned during the initial design process and that was reinforced by our user tests was the importance of some form of feedback to indicate to the user the current state of the system. Feedback, either auditory or visual, was critical to inform the user that the system was acting in an expected manner, and perhaps more importantly, to guide the user when the system either failed to recognize a gesture, entered an unwanted state, or lost tracking. Given all that we have learned through designing this instrument, a prominent focus of our future work will be to investigate which gestures, recognition algorithms, mapping techniques and forms of feedback, are best suited to facilitate users to control live performance systems through gestural interaction. Digito is currently being used in a number of longitudinal studies to evaluate the possible strengths and limitations of virtual musical instruments and to assist in uncovering new questions regarding gestural musical interfaces.

References

[1] A. Argyros and M. Lourakis. Real-time tracking of multiple skin-colored objects with a possibly moving camera. *Computer Vision-ECCV*, 2004.

[2] A. Burns and M. Wanderley. Visual methods for the retrieval of guitarist fingering. In *Proceedings of the 2006 conference on New Interfaces for Musical Expression*, 2006.

[3] J. Chowning. The synthesis of complex audio spectra by means of frequency modulation. *Journal of the Audio Engineering Society*, 1973.

[4] A. El-Sawah, C. Joslin, N. Georganas, and E. Petriu. A framework for 3d hand tracking and gesture recognition using elements of genetic programming. In *Computer and Robot Vision*. IEEE, 2007.

[5] V. Frati and D. Prattichizzo. Using kinect for hand tracking and rendering in wearable haptics. In *World Haptics Conference, IEEE*, 2011.

[6] N. Gillian, R. B. Knapp, and S. O'Modhrain. Recognition of multivariate temporal musical gestures using n-dimensional dynamic time warping. In *Proceedings of the 2011 International Conference on New Interfaces for Musical Expression*, 2011.

[7] S. Guomundsson, J. Sveinsson, M. Pardàs, H. Aanæs, and R. Larsen. Model-based hand gesture tracking in tof image sequences. *Articulated Motion and Deformable Objects, Springer*, 2010.

[8] G. Hackenberg, R. McCall, and W. Broll. Lightweight palm and finger tracking for real-time 3d gesture control. In *Virtual Reality Conference, IEEE*, 2011.

[9] C. Harrison, H. Benko, and A. Wilson. Omnitouch: wearable multitouch interaction everywhere. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 2011.

[10] C. Kiefer, N. Collins, and G. Fitzpatrick. Phalanger: Controlling music software with hand movement using a computer vision and machine learning approach. In *Proceedings of the 2009 Conference on New Interfaces for Musical Expression*, 2009.

[11] T. Mitchel and I. Heap. Soundgrasp: A gestural interface for the performance of live music. In *Proceedings of the 2011 Conference on New Interfaces for Musical Expression*, 2011.

[12] A. Mulder. Virtual musical instruments: Accessing the sound synthesis universe as a performer. In *Proceedings of the First Brazilian Symposium on Computer Music*, 1994.

[13] I. Oikonomidis, N. Kyriazis, and A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *Proceedings of BMVC*, 2011.

[14] K. Oka, Y. Sato, and H. Koike. Real-time fingertip tracking and gesture recognition. *Computer Graphics and Applications, IEEE*, 2002.

[15] D. Overholt, J. Thompson, L. Putnam, B. Bell, J. Kleban, B. Sturm, and J. Kuchera-Morin. A multimodal system for gesture recognition in interactive music performances. *Computer Music Journal*, 2009.

[16] B. Pritchard and S. Fels. Grassp: Gesturally-realized audio, speech and song performance. In *Proceedings of the 2006 International Conference on New Interfaces for Musical Expression*, 2006.

[17] M. M. Wanderley and N. Orio. Evaluation of input devices for musical expression: Borrowing tools from hci. *Computer Music Journal*, 2002.

[18] B. Zhang, J. Zhu, Y. Wang, and W. Leow. Visual analysis of fingering for pedagogical violin transcription. In *Proceedings of the ACM International Conference on Multimedia, ACM Press*, 2007.