Capture of Dynamic Piano Performance with Depth Vision

Albert Nisbet
Department of Computer Science
University of Canterbury
Christchurch, New Zealand
Email: ajn75@uclive.ac.nz

Richard Green
Department of Computer Science
University of Canterbury
Christchurch, New Zealand
Email: richard.green@canterbury.ac.nz

Abstract— **This paper proposes a method to capture keypresses and their velocity on a piano keyboard using a depth camera. Previous vision-based research generally does not capture velocity information along with keypress events.**

In the proposed method, the depth image captured by a Kinect v2 camera had a bilateral filter applied. The keyboard was registered manually by applying a perspective transform to generate a top-down view onto which key bounding boxes were overlaid. A vertically-dominant correlation kernel was used to filter noise this keyboard frame while minimising blur key edges in the horizontal direction. A difference image was generated based on an initial frame with no pressed keys or hands present. A threshold was applied to this difference image to isolate pixels above the keyboard's initial position such as hands. Hand detection involved applying opening and dilation operations to this frame. This approximate hand location frame was subtracted from individual key rectangles, which were then used as masks to average the difference image on a key-by-key basis to obtain an instantaneous depth for each key. A rolling average was applied and tracked, enabling calculation of velocity as each key surpassed the point of sound, entering a pressed state.

A successful linear relationship was obtained between the calculated keypress velocity and the baseline of measured peak audio amplitude for each press. Tested white and black keys yielded a sound ratio ranging from 0.78 to 0.94 mm s^{-1} dB^{-1}. This sound ratio was fitted to keypress data with high coefficient of determination values ranging from 0.81 to 0.92.

Keywords—performance capture; hand detection; velocity calculation, keypress detection

I. Introduction

Piano performances are highly dynamic and are imbued with the performer's individual style and musical interpretation. This fact, when combined with the dynamic notation of the composer, means that different notes will be played at different volumes by the performer. These variations can be large (for example where the performer is reproducing a sudden emphasis or *sforzando* notated by the composer) or subtle (such as slight leans or voicing adjustments made by the performer).

To thoroughly capture a piano performance, the dynamic properties of each keypress must be recorded. There exist two major methods of recording performances in such a way: MIDI capture and audio recording. Generally speaking, MIDI capture requires an electronic keyboard. Keypresses are recorded as a digital event stream [1]; the volume of each keypress is detected by physical sensors in each key and included in the stream. Audio recording involves capturing the keyboard output after sound synthesis, or the use of microphones to capture acoustic instruments.

Compared to audio recording, MIDI capture has several notable advantages. As it is a raw event stream, MIDI can be directly interpreted by production or synthesis software. This has a wide range of applications, including automatic music transcription (AMT), instrument remapping, acoustic remapping, technique analysis and entertainment. Compared to MIDI, use of audio recording in such software involves algorithmic interpretation to extract notes from the captured waveform. This process can be convoluted and imprecise, especially in complex performances [2].

Due to the requirement for an electronic keyboard, MIDI capture is seldom used for recording piano concerts at a professional level. The proposed method seeks to overcome the electronic keyboard requirement by using a vision-based approach to capture keypress events. These events must include volume information, even on acoustic or non-powered instruments.

Figure 1: Potential applications for the proposed method include non-intrusive performance capture of acoustic instruments, such as using a downwards-facing depth camera suspended above the piano [3].

II. Background

A. Previous Keypress Detection Systems

In [4], Akbari develops a method of automatically transcribing sheet music based on an RGB camera feed of a piano keyboard during a performance. Akbari's implementation, known as claVision, attained a very high accuracy of 95% for a range of performed pieces and implemented detection of pressed ranges of keys. claVision utilises on a camera mounted off-axis above the keyboard. This positioning causes pressed keys to cause signature lines to appear in the image relative to an initial frame. The keypress detection process utilised in claVision is summarised in Figure 2.

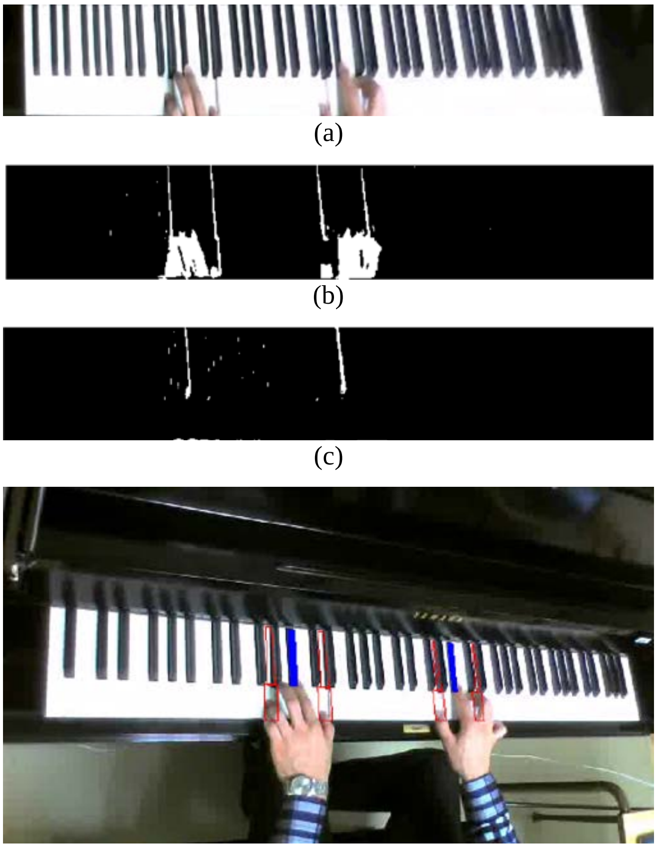


Figure 2: Summary of Akbari's keypress detection method: (a) transformed view of keyboard resulting from prior keyboard registration, (b) negative binary difference image used to extract pressed white keys, (c) positive difference image used to extract pressed black keys, and (d) keypress detection result overlaid on original RGB camera frame [4].

A large limitation of Akbari's work is the absence of dynamic capture. The use of a camera with depth recording capabilities should help overcome this limitation.

For keyboard recognition and key press detection, Akbari built on the work of Suteparuk [5] shown in Figure 3. A limitation of this prior report is greatly decreased accuracy for complicated pieces of music. This is due to Akbari's use of an off-axis camera position; the difference images generated using Suteparuk's top-down mount contained much less contrast. As with Akbari's work, Suteparuk did not address the capture of dynamics, focusing more closely on keyboard registration and binary keypress detection.

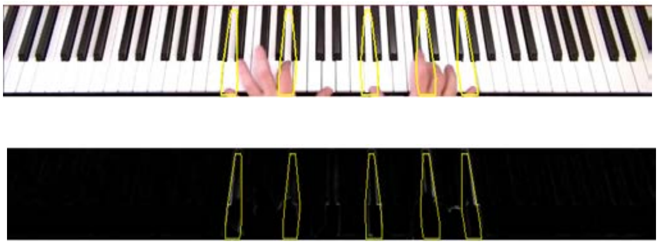


Figure 3: Key press detection by Suteparuk [5].

Work has also been carried out in detecting virtual keypresses on a hand-drawn paper keyboard through an RGB camera. Stenfert Kroese [6] obtained success rates of up to 90% for one finger on such a keyboard. There are significant limitations of the virtual keyboard approach, including an inherent lack of dynamic capture and a lack of application to real keyboards – important for production and performance analysis applications.

B. Hardware Keypress Detection Systems

In 2003, Moog introduced the since-discontinued PianoBar [7], a device which is physically fitted to the piano as shown in Figure 4. It used individual sensors to measure the depth of each key. Key press data including velocity was captured as a MIDI stream via an included control box. The device sat slightly above the keys and as a result did not impact the feel of the piano. The Moog PianoBar was accurate and received a good deal of professional use. However, the device was cumbersome in shape and size. It was also expensive, selling for over 1000USD. The vision-based system implemented in this research aims to be a step towards a significantly cheaper and more portable solution than the Moog PianoBar.

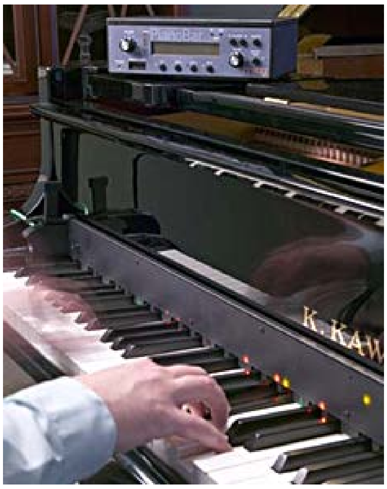


Figure 4: Moog PianoBar MIDI converter [7].

In 2015, Steinway & Sons introduced the Spirio piano. Solenoids built into the piano itself are capable of measuring velocity and replicating performances [8]. The built-in nature of this hardware within a high-end grand piano means the Spirio is extremely expensive and cumbersome.

C. Velocty Detection Methods

Various researched methods exist for calculating the velocity of objects in three dimensions. In optical flow tracking, a changing image is used to track relative motion between the observer and a scene.

In [9], Alexander, Guo, Koppal, Gortler and Zickler present a "focal flow" sensor which combines principals of differential optical flow and depth from defocus. This solution provides benefits of depth from defocus, in that no object or camera motion is required, and removes the need for lens actuation by detecting focal changes on a differential basis. Depth values can be obtained for local areas of the frame, and additional information is taken from optical flow. Velocity is calculated using central differences. A limitation of the focal flow method is computational intensity: the generation of depth and velocity maps can take multiple seconds for a typical webcam frame. More significantly, a high-contrast texture must be present everywhere in the scene, as can be seen in the example of Figure 5. Such high-contrast textures are largely absent from keyboard key surfaces, and therefore some amount of intrusiveness (for example, adhering stickers to keys) would be required for using flow-based systems.

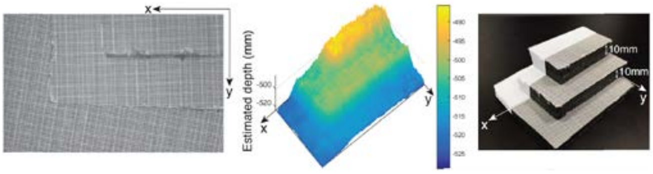


Figure 5: Example of computed depth map using focal flow method [9].

D. *Audio-Based Melody Extraction*

Audio-based methods of extracting keypress information are being actively researched and improved. While a different approach to overcoming the problem, audio-based melody extraction is worth considering in the context of this research as it provides a set of accuracy values against which the suitability of a vision-based system can be evaluated.

An example of an audio-based note recognition system can be found in the form of Paiva's melody detection algorithm, summarized in Figure 6.

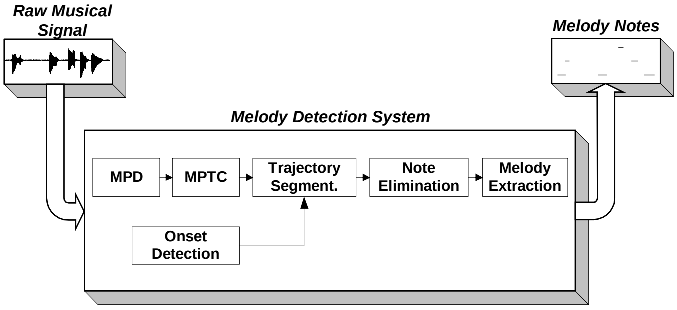


Figure 6: Paiva's melody detection system overview [10].

In their early 2014 paper, Salamon, Gómez, Ellis and Richard presented a summary of 16 such melody extraction algorithms [11]. The algorithms tended to attain 70%-80% accuracy for extracting a single melody line. The proposed vision-based system in this research aims to overcome the limitation of single-line extraction by physically recording data for each individual key.

E. *Keypress Mechanics*

In an acoustic piano, the volume of a keypress is determined by the speed at which the hammer strikes the string. The hammer speed is, in turn, directly and solely determined by the speed at which the key is pressed. More specifically, the crucial key speed is the one that is present as the key crosses the "point of sound". This point of sound is described by Mark, Gary and Miles [12] as the point at which the hammer is thrown toward the string. Beyond this point, downwards key velocity does not further affect the hammer's travel speed.

Palmer and Brown [13] investigated the relationship between this speed and the resulting sound amplitude of the associated string, determining the relationship as linear as shown in Figure 7.

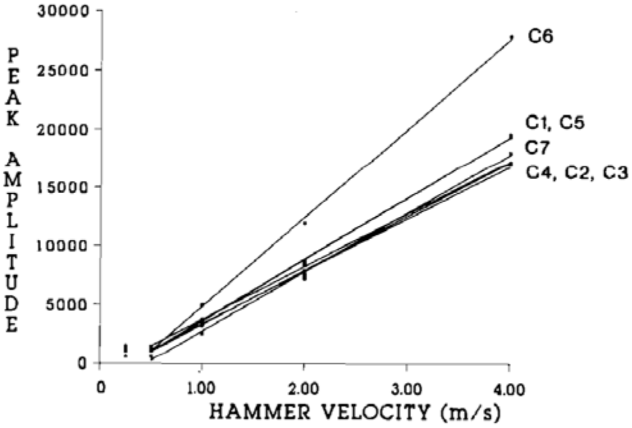


Figure 7: Peak amplitude as a function of hammer velocity [13].

III. Method

A. *Keyboard Registration*

The Kinect camera was fastened to a camera tripod and set to face downwards towards the keyboard at a height above the keyboard of around 500mm. A typical depth frame from the Kinect in this position is illustrated in Figure 8. This frame contains data for all visible pixels.

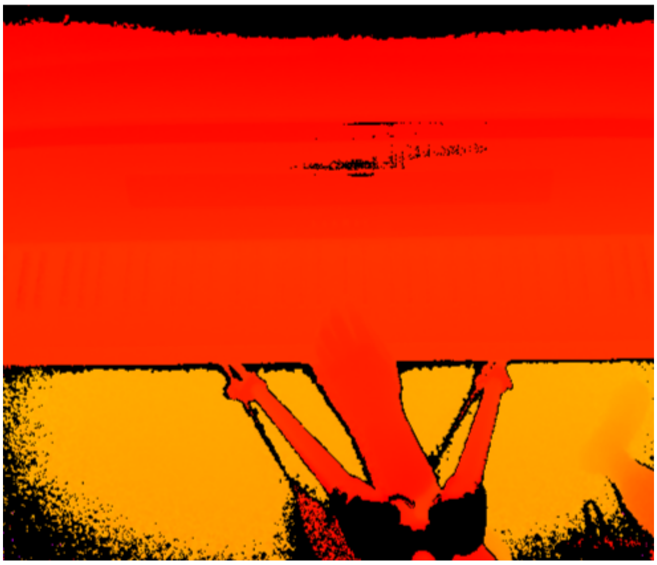


Figure 8: Kinect depth frame with colour ramp applied. Relatively small variations in depth mean the keyboard is not clearly visible.

A bilateral filter was applied to remove some noise from the source frames while reducing blurred edges. A threshold was then applied to the depth image to generate the result shown in Figure 9. This was done to remove excessively far or near pixels, which may contribute to noise within the keyboard area.

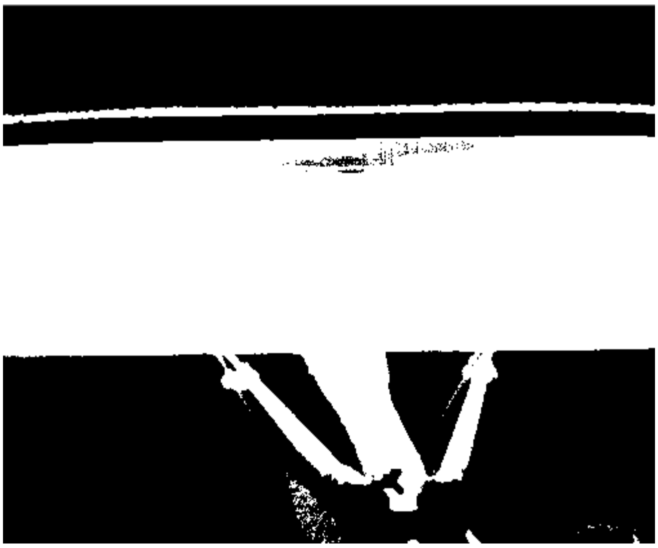


Figure 9: Thresholded and bilaterally filtered frame (not normalised) with shallow and deep extremes removed. White pixels contain a depth value, and black pixels are 0.

The keyboard was registered manually for the purposes of this research. A perspective transform was carried out to extract the keyboard. The points used for this transform were selected to simulate a top-down view of the keyboard and to correct the Kinect's horizontally flipped image. To reduce noise in the resulting image, a long vertical kernel was then used as a correlation kernel in a filter operation. The vertically-dominant kernel shown in (1) was used to reduce the blurring of lines in the horizontal direction (this would blur keys together and make depth detection less accurate).

$$k_{20x1} = \frac{1}{20}\begin{bmatrix}1\\1\\\vdots\\1\end{bmatrix} \qquad (1)$$

The result of perspective transforming then filtering the image is shown in normalised form in Figure 10.

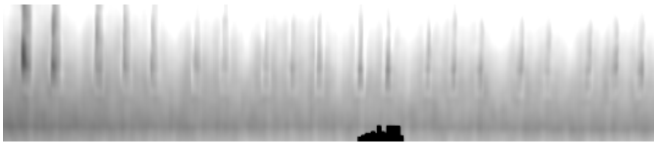


Figure 10: Perspective transformed and filtered keyboard image.

B. Difference Image

The first keyboard image captured is used as a baseline for calculation of successive difference frames. This was calculated as in (2).

$$difference = current - first \qquad (2)$$

Therefore, pixels in the difference image have a value defined as depth in millimetres below the original pixel positions. It is important the first frame be an image of the keyboard with no hands or pressed keys present. This difference image is not absolute: in the difference image, hands above the keys will have negative pixel values and pixels on depressed keys will have positive values.

C. Hand Detection

For this research, precise hand locating or finger identification is not required. However, approximate hand detection is vital to reduce the presence of hands above keys affecting keypress tracking.

Approximate hand detection was implemented in four stages. First, a threshold was carried out on the difference image to isolate pixels greater than 6mm above the keyboard. This included the hand and some noise. An opening operation was carried out with a circular structuring element of diameter 11 to remove noise in a similar fashion to Figure 11, with additional dilation to create a "finger region" with a margin of safety. This region was subtracted from key masks when calculating depth values to ensure fingers were not considered as negative keypresses. The entire process is shown in Figure 12.

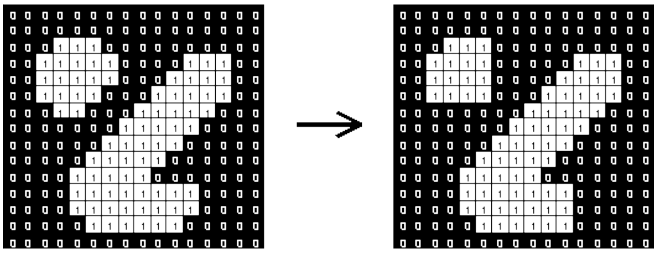
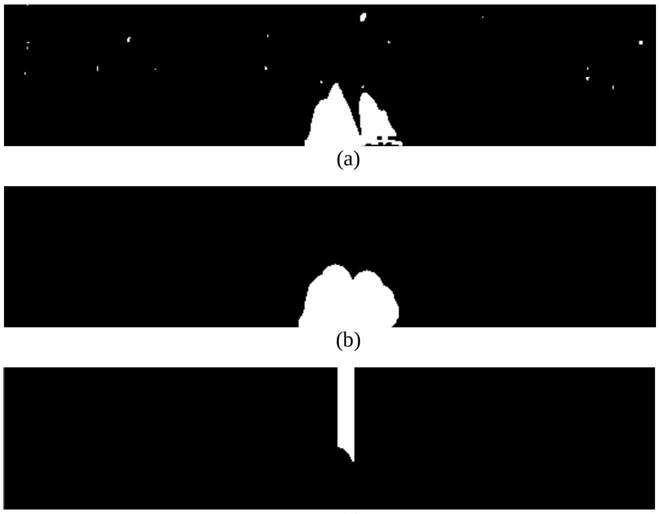


Figure 11: Effect of opening with 3x3 square structuring element [14].



(a)

(b)

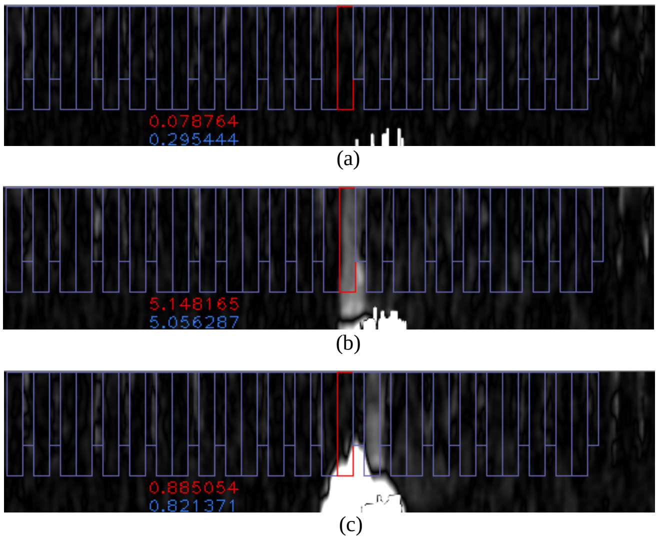
(c)

Figure 12: Basic hand detection procedure, involving (a) thresholding for pixels over 6mm above original key position, (b) opening and additional dilation of threshold frame to remove noise and add margin of safety, and (c) subtraction of the dilated frame from the key mask used for averaging.

D. Depth Detection

To enable dynamic capture, each key has an associated depth value at each frame. This is in comparison to the binary nature of prior research, where each key is either pressed or not pressed.

For each key, the keyboard depth frame is masked using the mask generated as in Figure 12c, based on the key's bounding box and the approximate hand location frame. Within this mask, the mean of the frame's depth pixels is calculated to give an instantaneous depth value for that key. For noise removal and tracking purposes, a rolling average of this depth value is stored for each key. A buffer size of n=5 was chosen for the rolling average as a good compromise between response time and noise removal. Examples of the calculated values are overlaid on the normalised difference image as shown in Figure 13 for three different cases.



(a)

(b)

(c)

Figure 13: Obtained depth values in millimetres for the C4 key as outlined in red, with instantaneous depth in red and n=5 rolling average in blue, for (a) no key press, (b) key press, and (c) no key press with hand interference. Note: this image is normalised and shown as an absolute difference image. Numbers are derived from the non-normalised image.

E. Velocity Calculation

When a key reaches the point of sound, a keypress is registered. At this stage, the velocity is to be calculated and output. The point of sound was taken to be 4.5mm based on measurements on the upright Yamaha piano used for testing.

To implement this, the rolling average calculated previously is continuously monitored. As soon as this smoothed depth value exceeds the point of sound, the key velocity is estimated using the reverse finite difference method applied to the rolling average values as shown in (3).

$$vel_{press} = \frac{depth_{current} - depth_{previous}}{\Delta t_{frame}} \qquad (3)$$

This calculation is demonstrated graphically in Figure 14. The first averaged depth value exceeding the point of sound is used in conjunction with the previous point to calculate a gradient value which is then associated with the keypress. This gradient is illustrated as a projected line based on the two key points. The steeper this line, the faster the keypress and the louder the note.

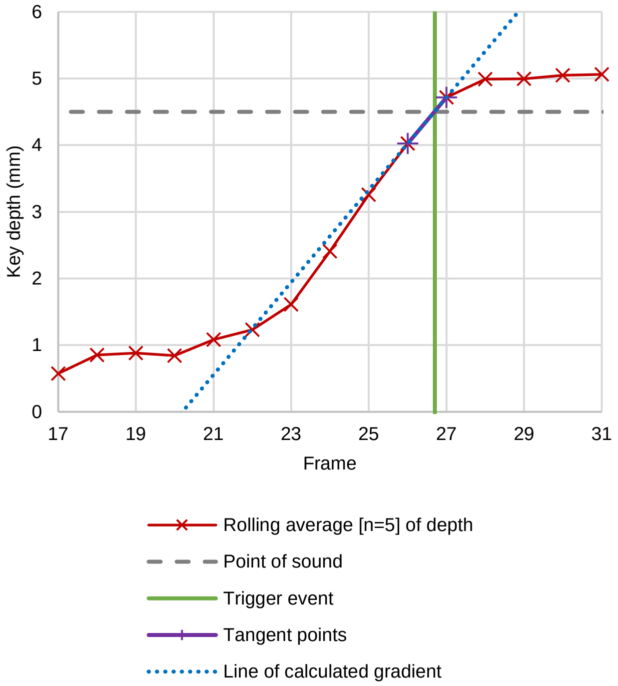


Figure 14: Graphical representation of velocity calculation when point of sound is reached.

The velocity is only calculated at the frame in which the key surpasses the point of sound. Before the next velocity is calculated, the key must be "lifted" by returning to within 2.5mm of the original depth.

IV. Results

Results were obtained on a test machine with an Intel i5-6400 processor at 3.20 GHz, 16GB of DDR4 RAM at 2133MHz and an NVIDIA GTX1060 graphics card with 6GB of GDDR5 VRAM. Code was written in C++ using Microsoft Visual Studio Community 2017 on Windows 10 Professional. OpenCV 3.2.0 [15] and Kinect Studio 2.0.1410 [16] were utilised. A Microsoft Kinect V2 camera was used, with a depth frame of resolution 512x424 and of frame rate 30 frames per second.

Detected keypress velocities were compared to a baseline of relative peak amplitude level for each press, as captured by a stationary microphone in proximity to the piano and analysed using Adobe Audition CS6 as shown in Figure 15.

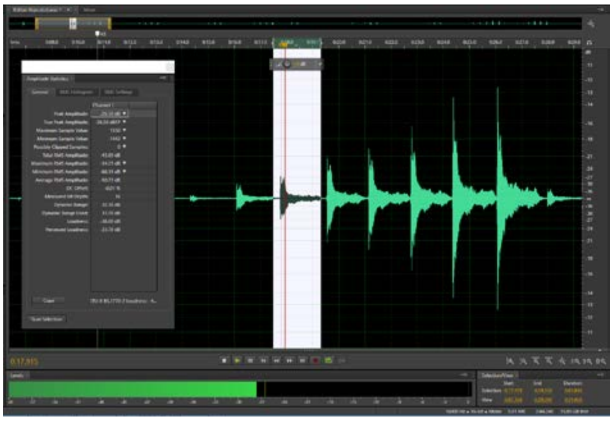


Figure 15: Extraction of peak amplitude information from microphone audio recording as point of comparison.

Figure 16 shows the calculated and recorded data over time for a series of the keypresses increasing in volume from *piano* to *forte*. As expected, the increase in measured relative peak amplitude can be proportionally matched to calculated keypress velocity.

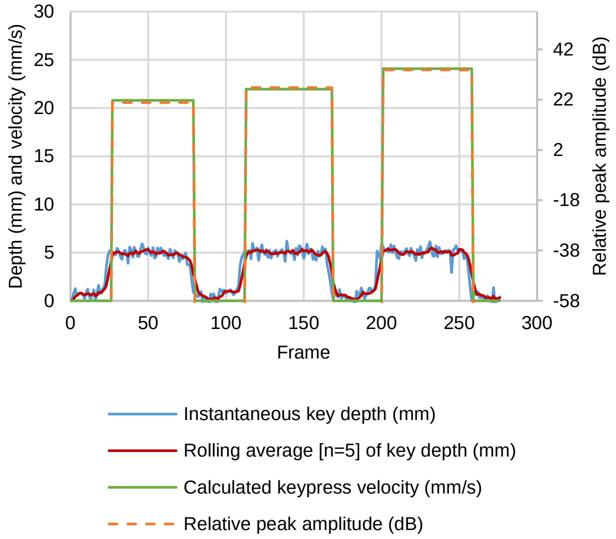


Figure 16: Keypress sequence of C4 key, showing close correlation of calculated keypress velocity and measured peak amplitude over time.

Repeated keypresses were made in a similar fashion for different white and black keys, with a peak amplitude and detected velocity recorded for each press. The results were organised by key and are shown in Figure 17 with best-fit lines. As expected from Palmer and Brown [13], linear best-fit lines were appropriate.

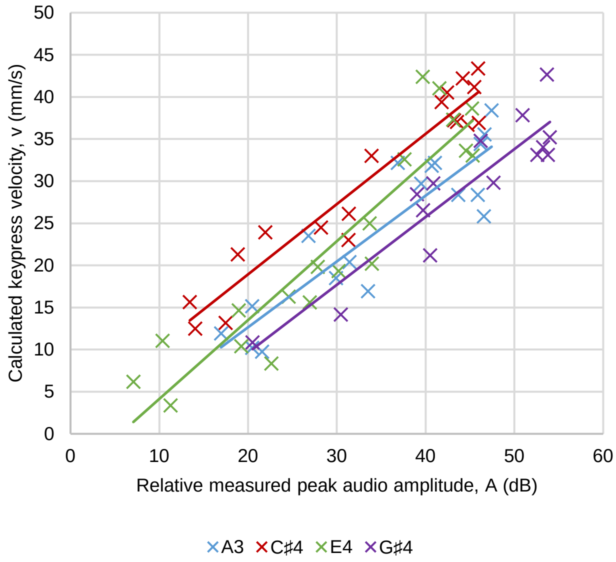


Figure 17: Correlation between measured peak audio amplitude and calculated key velocity for repeated keypresses of two black notes and two white notes.

The gradient of each key's best-fit line was calculated and named sound ratio, m, in mm s^{-1} dB^{-1}. The coefficient of determination, R^2, was additionally obtained as an indication of the tightness of overall fit. Results are summarised in Table 1.

TABLE 1 RESULTS SUMMARY

Key	Presses	Note Correctly Detected	Sound Ratio, m (mm s^{-1} dB^{-1})	Coefficient of Determination, R^2
A3 (White)	18	18	0.78	0.81
C♯4 (Black)	18	18	0.83	0.92
E4 (White)	19	19	0.94	0.85
G♯4 (Black)	14	14	0.81	0.84

For all keypresses in testing, no false positive keypress was registered. Additionally, the note associated with each press was correctly detected each time. The program ran in real time, with an execution duration of around 21ms per frame.

V. Conclusion

A. Summary

The proposed method was successful in obtaining a relationship between vision-derived keypress velocity calculations and an audio amplitude-based baseline: tested keys yielded a sound ratio ranging from 0.78 to 0.94 mm s^{-1} dB^{-1}. This sound ratio was fitted to keypress data with high coefficient of determination values ranging from 0.81 to 0.92.

B. Comparison with Prior Research

The correct note detection rate of 100% with 0% false positives compares favourably with the 95% accuracy delivered by Akbari's claVision [4], however it is important to note that the present research focused on velocity detection of individual keypresses rather than the accurate capture of complex performance as evaluated by Akbari. Nevertheless, the depth method of keypress detection certainly appears comparable in performance to the RGB methods used in claVision, while overcoming the limitation of absent velocity capture.

Compared to Moog's PianoBar [7] and Steinway's Spirio [8], the proposed system managed to capture velocity data in an affordable and relatively portable package. While it is unlikely this vision-based system is as accurate as the PianoBar's hardware approach, the system is able to run in real time and would therefore be able to facilitate instrument remapping with minimal latency, in a similar fashion to the PianoBar.

C. Limitations of Research

As with most vision-based approaches, this system has no way of detecting keypresses for keys which are obscured by hands. While hands are detected and removed from the key masks, significant hand coverage will cause unreliable readings and complete hand coverage will cause zero readings.

The method proposed in this research uses a manually-registered keyboard. This registration is a time-consuming process and requires entry of warp points into the program.

The Kinect camera used for depth detection output depth frames at a rate of 30 frames per second. A camera with a higher frame rate would be likely to give higher accuracy, provided noise characteristics were comparable.

D. Future Research

As a proof of concept, much of the testing for this research was focused on velocity calculations for single and repeated keypresses. The system is capable of registering any number of simultaneous keypresses and it would be valuable to research the versatility of the system in more realistic performances containing fast-paced melodic sections and broad harmonic chords. These situations may lead to effects such as significant hand coverage which are likely to occur in concert recording or technique analysis.

For this research, the camera was placed nearly directly above the keyboard at the minimum useful distance from it, in an attempt to maximise the signal-to-noise ratio. Further research into the effects of camera angle and height on the captured data should be carried out. This would be particularly applicable in situations such as concert capture, where minimisation of intrusiveness is crucial, or public entertainment, where the threat of theft is to be minimised.

It would be useful to carry out research into the integration of the velocity-detection system into an AMT workflow such as claVision, between automatic keyboard registration and transcription. Sheet music resulting from this process could be tagged with dynamics.

The results of this research show slightly varying sound ratios and line intercept values for different keys. This is likely due to various sources of error as well as apparent volume differences for varying frequencies. Further data collection could be carried out for a wide range of keys. The resulting data may be useful as a calibration step for registering a particular keyboard's sound ratio profile.

VI. References

[1] D. M. Huber, The MIDI Manual, Carmel, USA: Focal Press, 1991.

[2] O. Derrien, "A Vert Low Latency Pitch Tracker for Audio to MIDI Conversion," *Conference on Digital Audio Effects,* vol. 1, no. 17, p. 4, 2014.

[3] The Musical Touch, "Events," 2012. [Online]. Available: http://www.themusicaltouch.com/events.html. [Accessed 5 May 2017].

[4] M. Akbari, "Clavision: Visual Automatic Piano Music Transcription," University of Lethbridge, Lethbridge, Alberta, Canada, 2014.

[5] P. Suteparuk, "Detection of Piano Keys Pressed in Video," 2014.

[6] M. Stenfert Kroese, "A Touch Piano - Improved Paper-actuated MIDI," University of Canterbury, Christchurch, 2015.

[7] Synthtopia, "Save up to $500 on Moog Piano Bar," Synthtopia, 2007. [Online]. Available: http://www.synthtopia.com/content/2005/09/16/save-up-to-500-on-moog-piano-bar/. [Accessed 25 April 2017].

[8] Steinway & Sons, "Spirio," Steinway & Sons, 2015. [Online]. Available: https://www.steinway.com/spirio. [Accessed 27 April 2016].

[9] E. Alexander, Q. Guo, S. Koppal, S. Gortler and T. Zickler, "Focal Flow: Measuring Distance and Velocity with Defocus and Differential Motion," Harvard SEAS, Cambridge, 2016.

[10] R. P. Paiva, "An Algorithm For Melody Detection In Polyphonic Recordings," University of Coimbra, Coimbra, Portugal, 2005.

[11] J. Salamon, E. Gómez, D. Ellis and G. Richard, "Melody Extraction from Polyphonic Music Signals," *IEEE Signal Processing Magazine,* pp. 118-134, March 2014.

[12] T. C. Mark, R. Gary and T. Miles, What Every Pianist Needs to Know about the Body: A Manual for Players of Keyboard Instruments, GIA Publications, 2003, pp. 128-129.

[13] C. Palmer and J. Brown, "Investigations in the Amplitude of Sounded Piano Tones," *The Journal of the Acoustical Society of America,* vol. 90, no. 1, pp. 60-66, March 1991.

[14] R. Fisher, S. Perkins, A. Walker and E. Wolfart, "Opening," Image Processing Learning Resources, 2003. [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/HIPR2/open.htm. [Accessed 25 April 2017].

[15] Open Source Computer Vision Library, "OpenCV," Sourceforge, 23 December 2016. [Online]. Available: https://sourceforge.net/projects/opencvlibrary/files/opencv-win/3.2.0/. [Accessed 3 March 2016].

[16] Microsoft Corporation, "Kinect for Windows SDK 2.0," Microsoft Corporation, 21 October 2014. [Online]. Available: https://www.microsoft.com/en-us/download/details.aspx?id=44561. [Accessed 3 April 2016].