

Acquisition of Violin Instrumental Gestures Using an Infrared Depth Camera

Zacharias Vamvakousis
Universitat Pompeu Fabra
zacharias.vamvakousis@upf.edu

Alfonso Prez
Universitat Pompeu Fabra
alfonso.perez@upf.edu

Rafael Ramirez
Universitat Pompeu Fabra
rafael.ramirez@upf.edu

ABSTRACT

We present a low-cost six-degrees of freedom violin and bow pose-tracking system. The infrared and depth streams of an infrared-based Microsoft Kinect Mbox One depth camera are utilized to track the 3-dimensional position of 6 infrared markers attached on the violin and the bow. After computing the bow pose on the violin coordinate system, a number of bowing features are extracted. In order to evaluate the system's performance, we recorded 4 bowing exercises using simultaneously our system along with a commercial two-sensor 3D tracking system based on electromagnetic field (EMF) sensing. The mean pearson coefficient values were 0.996 for the bow position, 0.889 for the bow velocity, 0.966 for the bow tilt, 0.692 for the bow-bridge distance and 0.998 for the bow inclination. Compared to existing bowing-tracking solutions, the proposed solution might be appealing because of its low cost, easy setup and high performance.

1. INTRODUCTION

Violin performance tracking can be useful in many different contexts. For example, a system that tracks the bowing gestures of a violin student, combined with an appropriate software, could provide useful feedback, enhancing the learning process [1]. Music performance tracking could also be used for enriching a performance with audio or visual effects [2–4]. Another application might be studying ensemble cohesion [5].

In order to achieve 6 degrees of freedom pose estimation of the bow and the violin, different technologies might be applied. Existing commercial products include electromagnetic field (EMF) sensing devices [6], or multiple infrared cameras marker-based systems [1, 7]. More sensors might be attached on the violin/bow for measuring additional features, such as bow acceleration and bow weight [1, 2, 7, 8].

Depth sensors, like Microsoft Kinect or Intel RealSense, make possible 6 degrees of freedom (dof) pose estimation of rigid objects with a lower cost. De Sorbier et. al. [9] report on a system that achieves 6-dof pose estimation of the violin and the fingering, using the color and depth stream of the Kinect for Xbox 360 camera. To the best of our

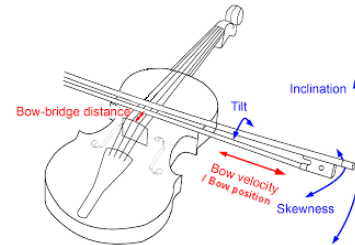


Figure 1. Commonly computed features in bowing tracking. Figure copied from www.bowing3d.info

knowledge, tracking of the bow using a depth sensor has never been reported. The reason might be that depth data are noisy in thin objects.

In this study we report on a system that achieves 6 degrees of freedom pose estimation of the bow and the violin using Kinect for Windows v2 camera. Then following bowing features are computed: position (distance between the point that is in touch with the string and the frog), velocity, tilt, skewness, inclination and distance to bridge 1. Compared to existing solutions for violin bowing tracking, the proposed solution is low-cost, easy to setup and non-invasive.

2. MATERIALS AND METHODS

2.1 Markers position and calibration

The raw infrared and depth streams of Microsoft Kinect v2, along with reflective markers are used to extract the pose of the bow and the violin. Microsoft's Kinect sensor offers a resolution of 512x424 pixels for the infrared and depth stream at a framerate of 30 fps.

In order to achieve pose estimation of both the violin and the bow, 3 markers were attached to each of them. A marker consists of a reflective sphere and piece of paper attached next to it (figure 2). As Microsoft Kinect does not capture depth data on reflective or thin surfaces, the pieces of paper serve as reliable surfaces for capturing depth data. In both rigid objects the markers are placed in a shape of an isosceles triangle, with the equal sides being significantly longer than the base of the isosceles triangle.

After placing the markers, a short calibration is required, in which the distances between the markers in each rigid body are saved, along with the position of the bridge.

During the calibration, initially the user is asked to place one by one the violin and the bow in front of the camera. For each of them the physical distances in millimeters be-

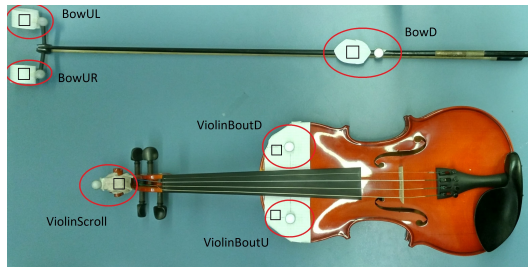


Figure 2. Three markers are placed on each rigid body.

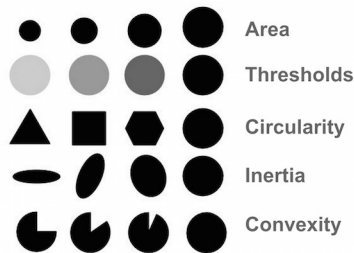


Figure 3. Parameters for rejecting reflections not caused by the reflective markers. Figure copied from www.learnopencv.com/

tween the 3 markers are saved in a file. Later, when tracking the bow and the violin, these dimensions are used for detecting whether both objects are tracked in the scene and for labeling each of the 6 markers.

The SimpleBlobDetector function, part of the OpenCV library, is utilized for retrieving the positions of the markers in the infrared image. The algorithm implemented in SimpleBlobDetector function consists of the following steps: (i) Convert to binary image through thresholding. (ii) Extract connected components and calculate their centers. (iii) Group centers: close centers form one group that corresponds to one blob. (iv) Estimate final centers of blobs. (v) Finally blobs that do not satisfy specific constrains are rejected. These constrains include: minimum and maximum blob area, minimum and maximum circularity, ratio of the minimum inertia to maximum inertia, minimum and maximum convexity (see figure 3).

If exactly 3 markers are detected, the algorithm passes to the next phase, which is estimating the real world coordinates of each marker. Microsoft Kinect's depth data are absent on reflective surfaces. An algorithm for estimating the depth of a reflective marker was developed. This algorithm looks for depth data on neighbouring surfaces. Initially the center of each marker is extracted from the infrared frame. Then in the depth frame: the pixel 'minDepth' with the minimum depth value in a square region 11x11 pixels centered around the marker is detected. In order to discard depth data of the background, all pixels with depth value more than 10cm of the depth of 'minDepth' are rejected. The median depth value of the remaining pixels is returned as the depth of the marker. Finally the MapDepthPoint-ToCameraSpace function, provided by micorsoft Kienct's SDK is used for retrieving the real-world coordinates, in

the camera coordinate system, of each marker in mm.

Three seconds are recorded for each rigid object. For each frame the distances between the markers are computed. For each object two values are saved: the length of the base side of the triangle and the length of the equal sides. Then for each side, the average value among all frames is computed and saved.

2.2 Labeling blobs

This task consists of recognizing the violin and bow markers. The algorithm for recognizing each marker is based on applying constrains related to the geometrical properties of the points formed by the markers. We will use the labels (names) shown in figure 2. Additional constrains are applied for rejecting shapes with similar geometry, that can be randomly formed when moving the bow. The following algorithm describes how detected blobs are labeled in each frame.

1. Make an assignment of all blobs to labels.
2. Compute the length of equal sides and base of the triangles on the bow and the violin.
3. If all distances are equal to the distances retrieved from the calibration (with an error margin of 6cm) and the BowUL and BowUR markers are further from the ViolinScroll marker than the ViolinBoutU and ViolinBoutD markers, then return the XYZ positions of all 6 blobs, else return to step 1 until all possible assignments are made.
4. If no assignment resulted to a valid tracking, return null.

In step 2, distances are computed bit differently that in the calibration process. The same method is applied, but in a smaller area of 7x7 pixels, and around a point 5 pixels next to the blob. These areas are shown as squares in figure 2. The purpose of the pieces of paper pasted next to the reflective spherical markers is to increase the accuracy of the depth data in these areas.

2.3 Calibrating Performer's position

In order to optimize the performance of the tracking, the distance and orientation of the violin to the camera must be optimized, otherwise markers might be out of frame, hidden by performer's body, or too small to be tracked.

Once the 3d coordinates of all labeled markers are computed, the euler angles and the local to world transformation matrices of both rigid objects are calculated. The euler angles, along with the 3d coordinates of each rigid body are then used to instruct the performer to appropriately place himself/herself in front of the camera. Additionally the transformation matrices for the violin and bow coordination systems (see figure Figure 4) are computed.

Figure 5 shows the feedback given to the performer. In this feedback two circles appear. The left circle corresponds to the position and the right to the rotation of the violin. The size of the left circle becomes smaller as the

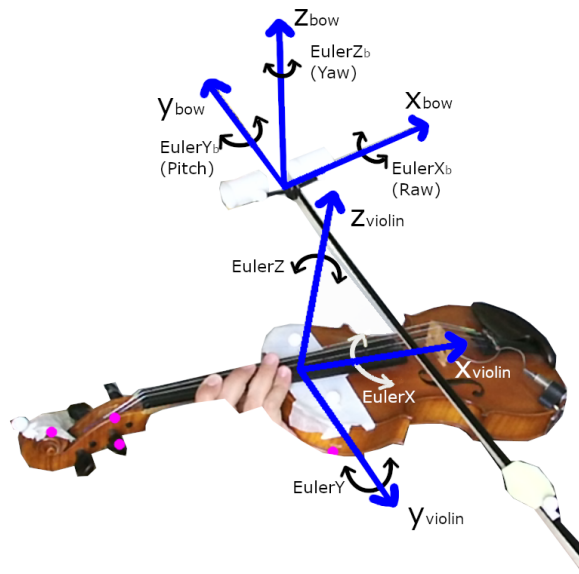


Figure 4. The violin and bow coordinate systems as computed after labeling the markers in each rigid body.

violin moves away from the camera and bigger as it moves closer. The same circle moves left/right and up/down, following the violin's movements. The performer's objective is to place this circle on top of a white circle that appear on the left side of the screen. When the position is within "optimum" limits, the circle becomes green. If the position is good, but not optimum, the color of the circle turns yellow. Finally if the position of the violin is not acceptable, the circle turns red.

Similarly, the right circle shown in figure 5 corresponds to the violin's rotation. The circle moves up/down, left/right as the user rotates the violin on the X and Y axis (the Z axis does not affect the tracking, so no feedback is given). The purpose of the performer is to place the circle on top of the white circle on the right side. Colors change as described in the circle corresponding to the position.

2.4 Calibrating bridge and Placing virtual markers

In order to calibrate the bridge position, the user is asked to place the BowD marker (see figure 2) at the lowest point of the bridge. The local XYZ-point of BowD in the violin coordinate system is then recorded during one seconds and the average of all frames is saved.

Similarly, the frog and tip points on the bow are saved as local points in the bow coordinate system. In the current version of the system, a default calibration is used for these points. The bowTip virtual marker is placed 3.5 cm under the mid point between the BowUL and BowUR markers. The BowFrog virtual marker is placed 11 cm from the bowD marker and towards the direction of the frog, and 2 cm towards the hair of the bow.

2.5 Computing bowing features

The following features are computed:

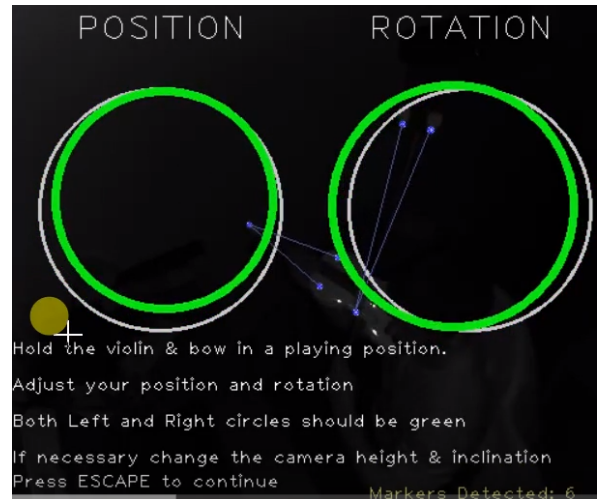


Figure 5. Visual feedback given to the user in order to place him/herself in front of the Kinect camera. The user has to adjust the violin's position and rotation. The left (position) and the right (rotation) circle should be placed at the center, denoted by the two white circles. Better positioning of the violin, results in more accurate tracking, as reflections on the violin body are avoided, and it is ensured that the marker will always remain visible to the camera.

1. Position: As bow position we refer to the distance between the point that the bow touches the strings (ContactPoint) and the frog. The point ContactPoint is computed by projecting the BowTip and BowFrog on the plane of the Violin, and then computing the intersection point of the lines (ContactPoint, BowFrog) and (bridge, violinScroll). Then the bow Position is computed as the euclidean distance between the Contact Point and the BowFrog.
2. Distance to Bridge corresponds to the distance between the ContactPoint and the bridge.
3. Velocity: the derivative of bow position.
4. Inclination: the first euler angle (Raw) of the bow rigid object in the violin coordinate system.
5. Tilt: the second euler angle (Pitch) of the bow rigid object in the violin coordinate system.
6. Skewness: the third euler angle (Yaw) of the bow rigid object in the violin coordinate system.

3. EVALUATION

3.1 The experimental setup

A professional violin player was asked to perform 4 recordings, about 40 seconds each, in which data were recorded simultaneously using a Kinect-v2 and the Polhemus commercial 3D tracking system based on electro-magnetic field (EMF) sensing. Details about the system used for computing bowing features with Polhemus can be found in a study

	Position	Velocity	Tilt	BridgeDist	Inclin
1	0.995	0.918	0.972	0.605	0.999
2	0.997	0.937	0.978	0.555	0.998
3	0.998	0.942	0.985	0.850	0.998
4	0.992	0.758	0.927	0.758	0.998
AVG	0.996	0.889	0.966	0.692	0.998

Table 1. Pearson coefficient values between the Kinect and the Polhemus solution. Each row corresponds to a recording. The first two recordings are focused on the distance to the bridge, the third on the tilt, and the fourth on the tremolo. Each column corresponds to a different bowing feature.

by E. Maestre et. al in 2007 [6]. As skewness is not computed by the Polhemus system we do not report on this feature.

The recordings consisted of 2-octaves G-major ascending and descending scales, and each recording had indications to put the concentration on different bowing features. The first two recordings were focused was on the bow-bridge distance: the first recording was performed with the bow close to the bridge, while the second one with the bow away from the bridge. In the third recording, the performer was asked to continuously modify the bow tilt from one edge to the other. Finally, the fourth recording was performed with tremolo in order to analyze the behaviour during very fast changes in bowing position and velocity.

3.2 Data Synchronization and Alignment

In order to compare both measuring devices, a synchronization and alignment of the data streams was required. The first step is to find a common time scale and sampling instants. The Polhemus device has a framerate of 240Hz, and the kinect of around 30Hz. As the framerate of the kinect is unstable, in order to match both frame-rates and sampling instants, we perform an interpolation on the kinect data using the timestamps. Once the signals share the same time scale, we proceed to align them by computing the cross-correlation function (*CCF*). The delay in number of samples is equal to the location of the maximum peak in the *CCF*.

3.3 Results

The metric used to evaluate the similarity between both signals is the *Pearson coefficient*, which measures the linear correlation between two variables. It has a value between +1 and 1, where 1 is total positive linear correlation, 0 is no linear correlation, and -1 is total negative linear correlation. The obtained results can be found in Table 1. Additionally to the numerical results, we provide plots of the computed (aligned and synchronized) features for all recordings in Figures 6, 7, 8, 9 and 10, where the inter-correlations can be better observed.

4. DISCUSSION

As shown in figures 6, 7, 8, 9, 10, the data of the two systems are highly correlated for most features. In all record-

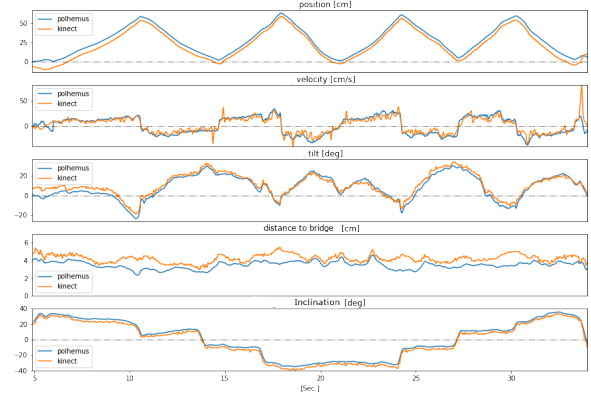


Figure 6. Plots of the first exercise, in which the performer played with the bow close to the bridge, as recorded by the Microsoft Kinect V2 and Polhemus systems. X axis is in seconds and Y is in degrees for the features referring to angles, and in cm for the features referring to distance. In yellow are plotted the data recorded by Kinect and in blue the data recorded by Polhemus.

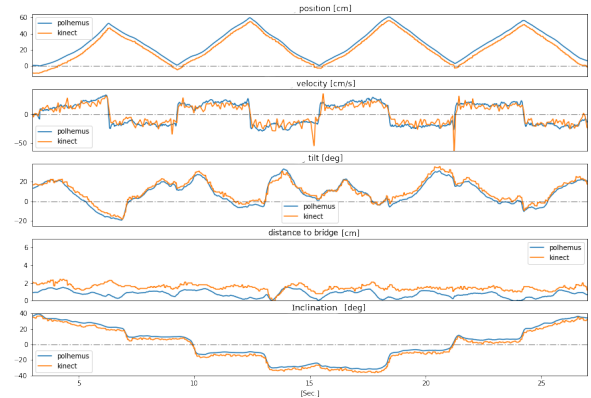


Figure 7. Plots of the second exercise, in which the performer played with the bow away from the bridge, as recorded by the Microsoft Kinect V2 and Polhemus systems. Units and colors as in figure 6

ings, the bowing position as recorded by Kinect system looks almost identical to the Polhemus's recording. This is also reflected in the pearson coefficient value in all recordings: it is always greater than 0.992. Bow inclination has also a very high correlation. In all recordings the pearson correlation value is greater 0.998. The tilt and velocity features as detected by Kinect also compare very well with features detected by Polhemus. The average pearson correlation value among all recordings is 0.966 and 0.889 respectively.

The bow-bridge distance pearson correlation value is not as good as with the other features. The reason might be that in the Kinect system, no calibration is performed for placing the BowFrog and BowTilt virtual markers. A default ,roughly estimated, point is chosen instead. This has as a result poor bow-bridge distance estimation when the bow tilt reaches the extremes. In all plots, we can observe

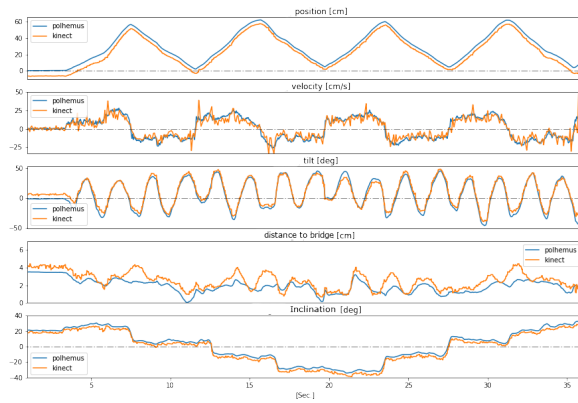


Figure 8. Plots of the third exercise, in which the performer changed the tilt periodically, as recorded by the Microsoft Kinect V2 and Polhemus systems. Units as in figure 6.

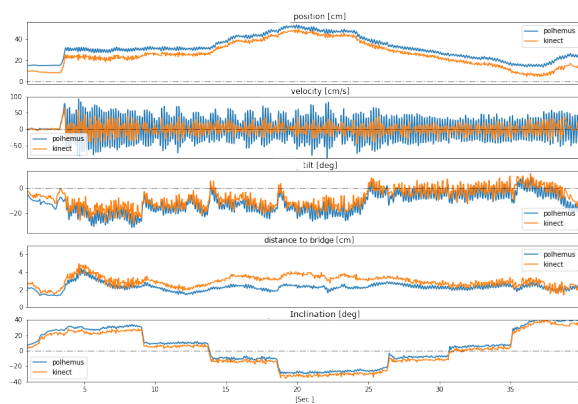


Figure 9. Plots of the fourth exercise, in which tremolo was performed, as recorded by the Microsoft Kinect V2 and Polhemus systems. Units as in figure 6.

that when the bow tilt reached a minimum value, there is a positive displacement of bow-bridge distance as tracked by Kinect. This indicates that BowTilt and BowFrog virtual markers are misplaced. As future work, a calibration process for the BowTilt and BowFrog should be added. This would also improve an offset observed in the bow position feature. This offset probably exists because the position of the frog is not exactly 11 cm away from the bottom marker of the bow, as the virtual BowFrog marker was placed.

For all features, the minimum correlation is performed in the tremolo exercise. This might be due to the low framerate of Kinect when compared to Polhemus. As shown in figure 10, although the tremolo frequency could be computed using Kinect, the width of the tremolo appears smaller than the tremolo width detected with Polhemus. Using a different infrared-based depth-sensing camera with higher framerate might improve the sensing of fast bowing gestures.

Since October 2017, the Microsoft Kinect v2 camera has been discontinued. Nevertheless, similar cameras that outperform the Kinect in technical characteristics have recently been released in the market. Intel Realsense D435 out-

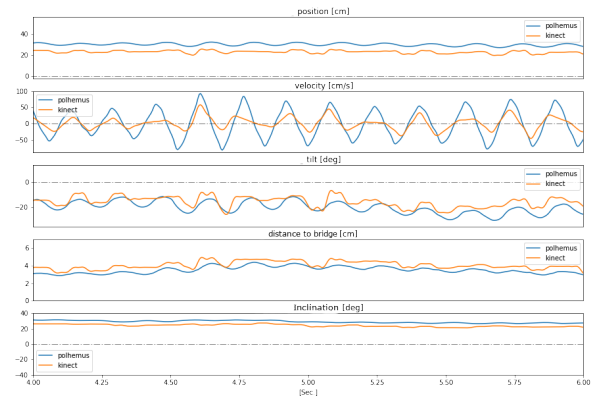


Figure 10. A closer look to the tremolo performed in the fourth exercise. Units as in figure 6.

performs Kinect v2 in both depth resolution and framerate (1280 x 720 @ 90fps). Since the proposed solution uses the raw depth and infrared streams, it can be easily adapted to work with any sensor that provides these streams. There is an increasing availability of laptops and smartphones with built-in depth-sensing cameras in the market. This tendency might make the bowing tracking solution we propose even more accessible to violin students and performers.

The results of this study indicate that the proposed low-cost bowing tracking system, well compares to a high-end alternative system. The fact that it offers an easy setup and accurate bowing tracking, might make it appropriate for usage in music schools or at home, combined to specialized software for enhancing the violin learning process. As future work a more user-friendly markers mounting solution should be developed. Possible future violin students should be able to quickly mount the markers on their violin and bow.

Acknowledgments

This work was partly sponsored by the Spanish TIN project TIMUL (TIN2013-48152-C2-2-R), and the European Union Horizon 2020 research and innovation program under grant agreement No. 688269 (TELMi project).

5. REFERENCES

- [1] E. Schoonderwaldt and M. Demoucron, "and sensors Extraction of bowing parameters from violin performance," vol. 2695, no. 2009, 2011.
- [2] F. Bevilacqua, N. Rasamimanana, E. Fléty, S. Lemouton, and F. Baschet, "The augmented violin project: research, composition and performance report," in *Proceedings of the 2006 conference on New interfaces for musical expression*. IRCAMCentre Pompidou, 2006, pp. 402–406.
- [3] L. Turchet, A. McPherson, and C. Fischione, "Smart instruments: Towards an ecosystem of interoperable devices connecting performers and audiences," in *Pro-*

ceedings of the Sound and Music Computing Conference, 2016, pp. 498–505.

- [4] K. A. McMillen, “Stage-worthy sensor bows for stringed instruments.” in *NIME*, 2008, pp. 347–348.
- [5] T. Grosshauser, V. Candia, H. Hildebrandt, and G. Tröster, “Sensor based measurements of musicians’ synchronization issues.” in *NIME*, 2012.
- [6] E. Maestre, J. Bonada, M. Blaauw, A. Perez, and E. Guaus, “Acquisition of violin instrumental gestures using a commercial emf tracking device.” in *ICMC*. Citeseer, 2007.
- [7] N. Rasamimanana, D. Bernardin, M. Wanderley, and F. Bevilacqua, “String bowing gestures at varying bow stroke frequencies: A case study,” in *International Gesture Workshop*. Springer, 2007, pp. 216–226.
- [8] L. S. Pardue, C. Harte, and A. P. McPherson, “A Low-Cost Real-Time Tracking System for Violin,” *Journal of New Music Research*, vol. 44, no. 4, pp. 305–323, 2015. [Online]. Available: <http://dx.doi.org/10.1080/09298215.2015.1087575>
- [9] F. De Sorbier, H. Shiino, and H. Saito, “Violin pedagogy for finger and bow placement using augmented reality,” in *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC), 2012 Asia-Pacific*. IEEE, 2012, pp. 1–5.