computing@computingonline.net
www.computingonline.net

Print ISSN 1727-6209
On-line ISSN 2312-5381
International Journal of Computing

CREATING AN INTERACTIVE MUSICAL EXPERIENCE FOR A CONCERT HALL

Svitlana Antoshchuk [1], Mykyta Kovalenko [1], Jürgen Sieck [2]

[1] Odessa National Polytechnic University, Ave. Schevchenko 1F, 65044 Odessa, Ukraine, {asg, kovalenkonv}@opu.ua, www.opu.ua
[2] University of Applied Science Berlin, Wilhelminenhofstr. 75A, 12459 Berlin, Germany & Namibia University of Science and Technology, Windhoek, Storch Street 5, juergensieck@acm.org, www.htw-berlin.de

Paper history:
Received 05 December 2017
Received in revised form 22 August 2018
Accepted 20 September 2018
Available online 30 September 2018

Keywords:
gesture recognition;
pose estimation;
computer vision;
machine learning;
Augmented Reality;
virtual instrument.

Abstract: In this paper we introduce a Kinect based posture recognition approach that can classify the user's pose and gesture and match them to a set of predefined musical instruments. The efficiency of the approach is then demonstrated using two applications. The Virtual Orchestra system uses pose- and gesture-recognition along with Augmented Reality technology to add a virtual musical instrument into the scene, both visually and audibly: the visual representation of the instrument is placed into the user's hands and the sound of the corresponding instrument is played. An additional functionality is that the user can control the intensity and the pitch of the sound by changing the speed of his hand or finger movements. The Magic Mirror game is the second application developed for the Berlin Concert Hall that uses the posture recognition approach to introduce the visitors to some classical music pieces and familiarize them with various classical musical instruments.

1. INTRODUCTION

A lot of new possibilities as well as new challenges were brought into many aspects of life with the development of information technologies in the last decade. As with many other things, the synergy of information technology and culture, like art or music, was unavoidable. Incidentally, this not only results in the appearance of new forms and aspects of culture and art, but also helps to expose newer and wider audiences to them.

Experiencing classical music at a live concert has always been popular with the masses. This is true even today in the age of digital recordings and internet-streaming services like Spotify, iTunes or YouTube.

Unfortunately, many concert halls and music theatres cannot be open 24 hours a day and have to stay mostly empty when there is no live performance. This may in some cases have a negative effect on the profitability of such establishments, which might even result in them having to close down. Along with various new marketing strategies, using information technologies to aid the preservation and propagation of culture is a very promising solution.

Augmented Reality, which becomes more and more popular, has an excellent potential to keep the interest of visitors and even to introduce more people to the world of classical music – either as listeners, or even as new performers.

Many solutions utilizing Augmented Reality can be developed to accomplish that, including making use of the fact that vast majority of people nowadays have smartphones with cameras, which are excellent for Augmented Reality applications [1].

However, another solution exists, that can take advantage of the emerging Mixed Reality trend [2] and make the entire process more social and physical. The Kinect device, developed by Microsoft, is very useful tool for bridging the gap between reality and virtuality, and provides an easy and intuitive way for people to interact with the virtual content [3].

In this chapter, we discuss the interactive system that we developed that uses a Kinect device to detect if a user is imitating playing a musical instrument in front of the screen and recognized the pose and gesture matching it to corresponding musical instrument.

We then demonstrate the approach using two applications: an Augmented Reality media application and an interactive game developed for the Berlin Concert hall.

2. RELATED WORK

A popular medium for creating virtual musical instruments is the use of Augmented Reality and object detection [4]. An example of this is the use of in-air gestures, as with Lages, Nabiyouni, Tibau, and Bowman's Interval Player [5]. The system uses a Leap Motion controller. The developed virtual instruments employed the use of both hands to play notes and chords. The dominant hand specifies intervals between successive notes, while the non-dominant hand plays the chords. However, the interface has not been able to handle half step intervals, such as sharps and flats, and it cannot handle intervals greater than five.

Another group [6] modified the existing drumstick so that it outputs sound when the drummer strikes a virtual drum. They have equipped drumsticks with an accelerometer, a gyroscope, a PC and a Musical Instrument Digital Interface (MIDI) sound generator. The data measured by the sensors are sent via Bluetooth. The PC sends MIDI messages to the MIDI sound generator when the system recognises that the virtual drum is being struck. The proposed system is designed to be used with physical drums because only less frequently used parts such as the cymbal and the cowbell are being simulated by the Airstic.

A further interesting research [7] features "Drum", a virtual instrument using the Microsoft Kinect and Arduino. The researchers employed the 3D sensing framework OpenNI as the driver and API to communicate with the Kinect. When the wrist enters the region of the virtual drum sets, their program triggers static WAV sound files corresponding to the virtual drum. The researchers were only able to create virtual drums for the Snare, Tom, Crash Cymbal and Ride Cymbal. The foot pedals were simulated using the Arduino and a cantilever switch. They programmed the Arduino to register signals from the cantilever switch and to play a sound whenever a trigger signal is received.

In one of our recent publications [8], we proposed a gesture-recognition system that uses a simple colour camera for colour segmentation, hand tracking and fingertip extraction. The approach uses contour analysis to extract and track the coordinates of the fingertips, which allows controlling an Augmented Reality object or a Virtual Reality environment using simple gestures. The main idea behind the approach was to use a probabilistic network to predict and recognise different gestures, described with an ontology.

Sensors are one of the most fundamental things in developing a virtual system. One of these systems is described in a research article by Liu, Fan, Li, and Zhang [9]. Using the particle filter model, they developed a hand tracking and gesture-recognition system that has a high tracking accuracy. The researchers utilised formulas that identify hand-skin colour, temporal motion and depth to identify the hand in the presence of background. Initialisation happens during start-up when the actor is asked to place the hand above the actor's body. The system described is only able to detect hand gestures when the actor is in a static position (sitting down in front of the camera), and is only able to track one hand.

A slightly different approach was presented in another research [10]. They presented the AirPiano, an enhanced music playing system to provide touchable experiences in HMD-based virtual reality with mid-air haptic feedback. AirPiano allows users to enjoy enriched virtual piano-playing experiences with touchable keys in the air. The researchers implemented two haptic rendering schemes to mimic the resisting force of piano keys using ultrasonic vibrations. Constant Feedback is designed to provide short and intense feedback whereas Adaptive Feedback is designed to follow the changes in feedback from a real keypress.

In another research, the Leap Motion controller was used to develop an application to learn Indonesian traditional musical instrument such as: angklung, saron, kenong and kendang [11]. The researchers have implemented the Virtual Indonesian Musical Instruments using leap motion, laptop and speaker. Leap motion was used to capture the user's motion, laptop for the view of Indonesian traditional musical instrument, and speaker for the output sound.

A virtual xylophone was developed by a different group of researchers [12]. The program registers a background depth image, generated by a Kinect sensor, and the user interacts with the program to identify the colour of tone bars and to select a restricted track space for tracking mallet locations. During a play phase, the program tracks mallet heads by locating pixels that are in front of the pixels registered in the background image.

3. THE APPROACH OUTLINE

The proposed gesture recognition approach [21] can be roughly divided into two fundamental phases: the learning phase and the recognition phase, as well as several steps that are not involved with either of the phases. The first main step of the approach is grabbing the colour and depth images from the Kinect device and extracting the necessary information from the frames, i.e. the skeleton joints and the pose features. The Kinect's depth sensor is based on the time-of-flight (ToF) principle and gives us the distance between the sensor itself and every point on the sensor's field of view.

Then during the learning phase, the features are matched to the poses and recorded for further use as the training data.

During the recognition phase, the extracted features are compared to the training data and the user's pose is recognized, which is then matched to the corresponding musical instrument in the database.

In one of our applications that use the developed approach [21], an additional phase is added, where the information about the pose is further enriched by additional information about hand movement. This extra information is used to augment the images by adding the visual effect of a musical instrument and the corresponding instrument's sound (Fig. 1).

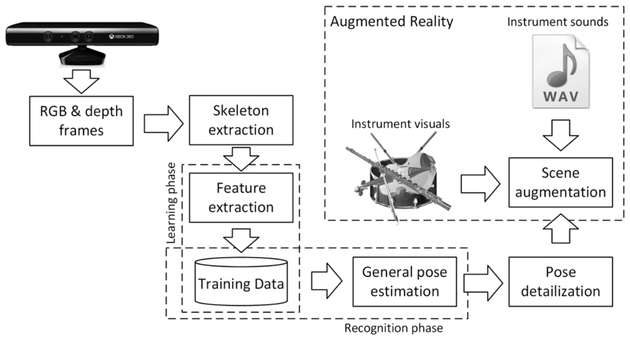


Figure 1 – The schematic overview of the system.

The Kinect device does all the necessary calculations to track people in front of the camera and extracts the coordinates of their skeleton joints (up to 20 joints), as well as provides an easy to use API. However, the Kinect in itself does not have possess any embedded functionality to estimate and recognise the user's pose or gestures, as well as extract any additional information about the nature of their movements.

This, along with the fact that we need to recognize a rather specific set of poses and gestures, means that we have to develop our own algorithms for pose recognition that can also match the poses to the corresponding musical instruments and sounds. The system that we have developed uses the OpenNI open-source framework to interact with the Kinect device [13], as well as the OpenCV framework for image processing and frame augmentation [14]. The reason for choosing the former lies in the fact that OpenCV has built-in support of OpenNI as well as very convenient interfaces to work with image data.

3.1 Model Selection

We have decided to treat the pose recognition in our application as a pattern recognition problem and use machine learning to achieve the task. For that purpose, we have tested and evaluated several popular machine-learning algorithms, such as the k-Nearest Neighbours classifier (KNN), the Multi-Layered perceptron (MLP), Support Vector Machines (SVM), Random Forests (RF), Naïve Bayes (NB) and Logistic Regression (LR).

Every algorithm was evaluated with cross-validation using accuracy as the metric. For every algorithm, the evaluation was performed 20 times, with different parameters and random states. Based on the results, means and standard deviations were calculates. The results are shown on Fig. 2.

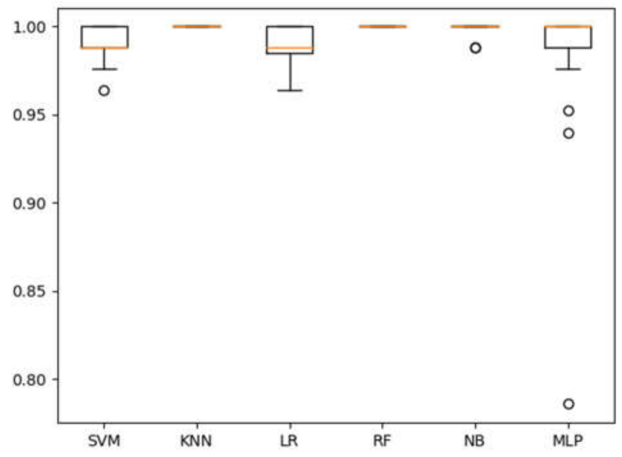


Figure 2 – Results of models' evaluation.

As seen from the graph, all models perform more or less the same. In particular, the k-Nearest Neighbours algorithm shows one of the highest accuracies (as well as the f1-measures) among other algorithms. This, along with the algorithm's simplicity, speed [15] and the fact, that it doesn't need to be trained and new training samples can be easily added at any moment, is why we decided to use *k*NN for our system.

We have previously developed our own modification of *k*-Nearest Neighbours algorithm for use in the Real-time hand tracking and gesture-recognition system [16] and we have tested its efficiency in that scenario. Our implementation, aside from assigning a label to the sample, also provides a similarity score based on the Gaussian kernel function. This allows us to determine, how close is the user to correctly mimicking one of the required poses and is very useful for the applications where our system was employed.

For every pose that we plan to recognize, we prepare a set of templates that are to be matched with the current pose shown by the player using the *k*NN algorithm. During the experimental testing of our system, we have decided to set the k value to 7 , which means we need at least 7 training samples prepared for every pose. For better accuracy and flexibility of the recognition, the training samples are provided from multiple people each doing their own take on the desired pose. The training samples are taken using the Kinect device by standing in front of it, extracting the salient features and recording them as training data.

3.2 FEATURE EXTRACTION

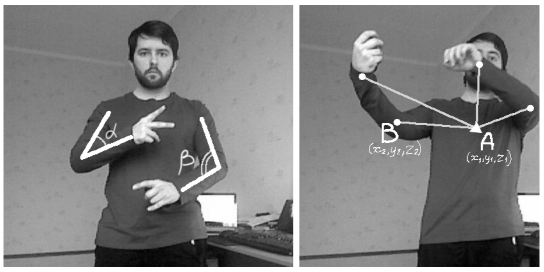
As with any machine learning application, the main problem is choosing the correct set of significant features to use as training data that will provide the best ratio of recognition speed and accuracy. We have to consider the fact that the users in front of the Kinect device might have different height and width, as well as stand in different positions relative to the camera [17]. More importantly, the users might not be facing the camera directly, i.e. their bodies might be at a slight angle in relation to the camera. This eliminates the possibility of directly using the coordinates of the joints or using the distances between them or using the relative positions of the joints. Another problem is the fact that some users might be left-handed and will be more comfortable with gestures that are mirrored relatively to the training data.

In order to solve these problems we have decided on using relative angles between body-parts as features. More specifically, we are using the following features:

– angles between the forearm and the upper arm on each side (α and β on Fig. 3a);

– angles between the body's centre of mass and the four essential joints of the arms: wrists and elbows (AB on fig. 3b);

– two motion features of hand movement averaged acceleration.



(a) (b)

Figure 3 – The main features used in the system: elbow angles (a) and relative joint angles (b).

This means, that no matter where the user is standing in the frame, or at what angle relatively to the camera he is positioned, the feature vectors will not be affected and the recognition will still be correct [18].

Additionally, these features are scale invariant, i.e. they do not vary for people of different height, body type, gender or distance from the sensor.

To calculate the angles α and β between the forearm and upper arm on each side, we use the formula (1) for the angle between two three-dimensional vectors:

$$\cos(\alpha)=\frac{\vec{x}\cdot\vec{y}}{\left\|\vec{x}\right\|\cdot\left\|\vec{y}\right\|} \tag{1}$$

The vector coordinates and vector norms are easily calculated from the 3D-locations of the joints. The vector $\vec{x}$ is the vector from the shoulder to the elbow and $\vec{y}$ is the vector going from the elbow to the wrist.

The angles between the body centre of mass and the joints are calculated as the angles between the vector going from the centre of mass to the joint (vector AB on fig. 3) and the normal vector to the user's torso (at the point A on Fig. 3).

To calculate the normal vector we assume that the three points, corresponding to the locations of the two shoulder joints and the torso joint, form one planar surface. Then the normal vector coordinates $N(x_N;y_N;z_N)$ can be expressed as (2):

$$N(x_N;y_N;z_N)=N\begin{pmatrix}(y_2-y_1)(z_3-z_1)-(y_3-y_1)(z_2-z_1)\\(x_2-x_1)(z_3-z_1)-(x_3-x_1)(z_2-z_1)\\(x_2-x_1)(y_3-y_1)-(x_3-x_1)(y_2-y_1)\end{pmatrix} \tag{2}$$

With this calculation, we can then use the equation (1) to calculate the angle between the normal vector and the "joint" vector. This gives us information about the user's posture, i.e. the spatial relation between his arms and his body, and the way the hands are positioned.

Additionally, we also use two motion-features, to recognise between postures that use similar hand positions, but use different hand motions. For that purpose, we also store the averaged acceleration of both hands over the span of the last five consecutive frames:

$$acc=\frac{1}{n-1}\sum_{i=0}^{n-1}\|p_i-p_{i+1}\| \tag{3}$$

The acquired features, including the angles between the upper arm and lower arm, are robust towards the variation in user's posture, as well as his position and his body's rotation relative to the camera. The chosen features, along with the chosen machine-learning algorithm, also provide a margin of error for the user's attempt at recreating one of the template postures.

Figure 4 shows the visualization of our training data using the t-SNE (t-distributed Stochastic Neighbor Embedding) algorithm, where for every pose we have pre-recorded at least 200 training samples:

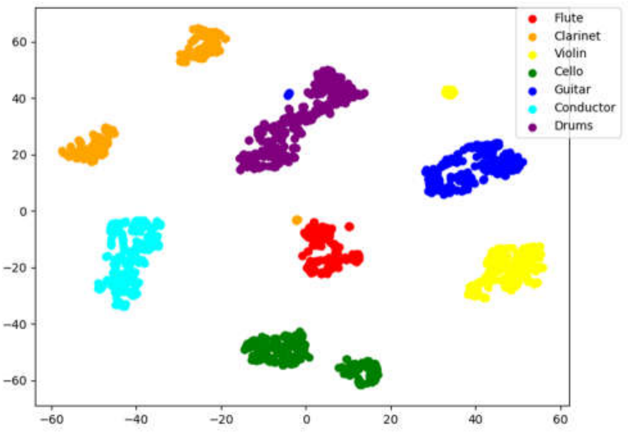


Figure 4 – Visualisation of the training data

t-SNE is an algorithm used for dimensionality reduction, that converts similarities between data points to joint probabilities and tries to minimize the Kullback-Leibler divergence between the joint probabilities of the low-dimensional embedding and the high-dimensional data.

As seen from fig. 4, the features that we have chosen allow for densely clustered classes (with a negligible amount of outliers), where each cluster of features corresponds to one pose.

It should be also noted, that these features were chosen as a result of a feature selection process. Here we used a wrapped selection process using a linear SVM model penalised with the L1 norm and the Extra-Tree method.

Originally, during the feature creation process we have also extracted additional features:

– distances from the sternum to the hand and elbow joints;

– angles between the shoulder line and the upper arms (calculated as angles between two 3D-vectors);

– angles between the forearms and the wrist.

Both features selection methods have allowed us to exclude the aforementioned features as being irrelevant.

From the features that were retained two were also given low scores: the angle of the left elbow and the angle to the left wrist (fig. 3b). However, that can be explained by the fact that all the training samples were recorded for a right-handed person. This was determined by including training samples for a left-handed person and running the feature selection algorithm again.

3.3 POSE CLASSIFICATION

The extracted features provide information about the user's posture, mainly the position of his arms and hands, and can serve as templates for the poses that our system aims to recognize, meaning that the feature vectors may directly be used as the training data for the *k*NN machine-learning algorithm. During the training stage, the feature vectors are recorded separately for every pose, corresponding to the various musical instruments.

Then during the recognition phase, when the user once again stands in front of the Kinect device, the system extract the features in real-time and tries to match the current feature vector to every feature vector present in the training data by using a simple Euclidian metric. Then, according to the *k*NN algorithm, the calculated distance values are sorted in the ascending order and the first *k* values are considered. This allows us to classify the current pose by a majority vote of the neighbours.

Considering the number of training samples, the Kinect's framerate of 30 fps and the fact that our system is supposed to work in real time without any delay (also 30 fps), we have, as previously mentioned, decided on a value of 7 for *k*, which provides the best ration of recognition accuracy and speed.

Furthermore, our algorithm has to deal with possible noise, jittering and false positives caused by the fluctuations in the joint coordinates provided by the Kinect device. The noise and the jittering can be caused by poor resolution of the depth sensor, as well as lightning changes and the presence of reflective surfaces, which are a common problem with most ToF-based devices. The idea behind the algorithm is using a queue of "tumblers" of a fixed length *N*. At every frame, after we extract the features and use the *k*NN algorithm to classify the pose, we push the recognition result to the back of the queue. If the queue is full, we also add the element to the front of the queue, so that the size of the queue is always kept lower or equal to *N*. After some experiments, we have found the most optimal values of *N* to be between 3 and 5.

When the tumbler queue is full, we have two options:

– if all *N* values in the queue are the same (i.e. the pose classification result was the same in *N* consecutive frames), or

– if *M* of *N* values in the queue are the same (i.e. the majority of the classifications results in the previous *N* frames were the same).

Regardless of the option chosen, we can conclude that the pose was recognised and classified correctly. Here, the first option allows us to eliminate false positives caused by joint jittering and measurement noise, making sure that our system is certain about the recognition result, although this may result in the system taking slightly longer to arrive to a conclusion. The second option allows us to make our system more fluid and forgiving toward sudden measurement noise; however, this requires setting a good enough *M* to *N* ratio.

The experimental testing shows that the proposed pose-recognition approach provides a high recognition accuracy. The test results are summarised in Table 1.

Table 1. Summary of the accuracy testing

Pose name	**Tumbler queue type**		**Response time (ms)**
	Full *N* ($N = 4$)	*M* of *N* (3 of 4)	
Flute	0.98	0.97	240
Violin	0.94	0.93	320
Drums	0.96	0.95	235
Cello	0.89	0.90	360
Trombone	0.93	0.94	270
Clarinet	0.97	0.98	260
Guitar	0.96	0.97	230
Triangle	0.92	0.93	290
Conductor	0.97	0.94	210

The average response time for a switch between different poses is 268 ms.

The tests were performed by the members of the development team for the Magic Mirror project, consisting of six adults, both male and female, with the height ranging from 170 cm to 190cm.

The real world testing of the posture recognition system was performed during an Open Doors day at the Berlin Concert Hall, where it was used as a key module of the Magic Mirror application (more details about the project are given in chapter 4.2).

During the event, a total 26 children (ages 11-14) and 8 adults have tried out the application.

Both the real world testing and the indoors testing with the development team have shown good results, despite a high variation of body types and height between the users. This can be attributed to the fact, that the features, used in our posture recognition system, only use relative angles between body-parts and are both scale- and orientation-invariant.

3.4 POSE DETAILING

It is clear, that simply standing in a position matching the musical instrument is not enough for playing an instrument as that process is normally much more dynamic. For instance, playing a flute requires moving fingers, while playing a violin involves moving the hand holding the bow. To solve this problem, we add the pose recognition system with an additional hand-gesture recognition subsystem described in [8]. This subsystem will further elaborate the pose, providing additional information about the posture and gesture, allowing a more detailed description of gestures and the recognition of more intricate poses.

When a pre-recorded pose is successfully recognised and maintained for several consecutive frames, we run the pose-elaboration procedure that consists of several key elements:

– calculating the movement speed and direction of each hand

– extracting the fingertips on both hands (when possible) and counting the number of outstretched fingers.

To calculate the movement speed and direction of the hand we only require the coordinates of both wrist joints. Using the coordinates of the joints, we can calculate the track line of the hands and their average speed, velocity and movement direction (Fig. 5).

This information is used to control the speed and intensity of the musical instrument, e.g. the force with which the user bangs the drums or the movement speed of the bow while the user is playing the violin. In our system, we use three gradations for the hand movement speed (slow, medium, fast) as well as the four main movement direction (up, down, left, and right).

We additionally implement a variation of the Kalman filter to smooth out the track line, further removing the joint coordinate jittering and making the gesture recognition more stable and accurate [19].

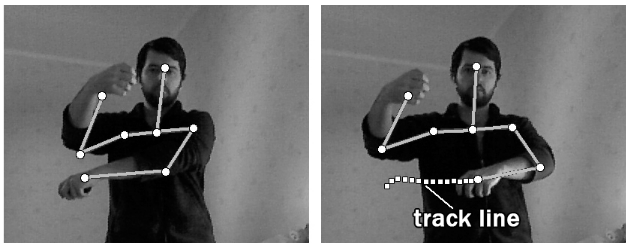


Figure 5 – Tracking the wrist joints and calculating the movement speed and direction.

To extract the fingertips we also use the coordinates of the wrist joints. However, in this case, since Kinect does not provide the coordinates of fingers, we have to use the extracted images to locate the fingers ourselves. To do that we combine the information from both the colour frame and the depth frame.

In the colour frame, we estimate the average colour of the user's skin and then use colour segmentation to get the hand contours. In the depth frame, we can refine the hand contour thanks to the stark difference between the background pixels and the foreground pixels, with the foreground pixels belonging to the area segmented on the colour frame.

After combining this information and receiving a reliable hand contour, we can now perform a contour analysis [16] to extract the positions of the fingertips by executing a clockwise contour traversal and calculating the angles between points of the contour (Fig. 6).

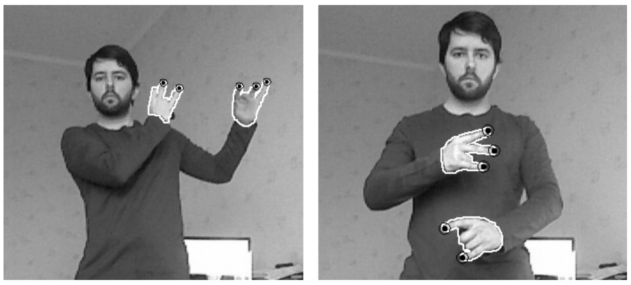


Figure 6 – Extracting fingertips.

The total number of detected fingertips will correspond to the total number of outstretched fingers, which means that we can easily calculate the number of fingers that are applied to the instrument. This information can be used to control the pitch of the instrument, e.g. pressing fingers to the holes of a flute or a clarinet.

Our chosen approach for pose- and gesture-recognition allows us to identify a multitude of instruments. In addition, the system also recognises the user's gestures, specifically the speed of the hand and finger movements to control the sound of the instruments. It should be noted, that there are many instruments played using very similar poses and similar gestures. An example of such a group of instruments are wind instruments (clarinet, bassoon and oboe).

This may also apply to some other instruments that are not similar in nature, but use seemingly similar movements (e.g. playing drums and playing a piano). The Kinect device unfortunately does not provide enough information to differentiate between such cases.

That is why in this paper we limit ourselves to only one instrument out of such a group of similar instruments, i.e. out of the wind instrument group we only choose the clarinet.

4. DEMONSTRATION AND PROOF OF CONCEPT

By this point, we have gained the following information from the Kinect device: the positions of every skeleton joint; the pose; the movement speed and direction of the hands; and the number of fingertips.

Having developed and tested the posture and gesture recognition approach, we will now discuss two possible practical applications for it.

The first one is a Virtual Orchestra project that uses the information about the classified posture and the additional pose detailization along with the Augmented Reality approach to overlay the image and the sound of the corresponding musical instruments, allowing the user to actually play the instruments in real-time. The second application is the Magic Mirror project, developed for the Berlin Concert hall.

The Magic Mirror is essentially an interactive game for the visitors of the concert hall, where they can familiarize themselves with pieces of classical music and learn how the musical instruments involved in the piece are played.

4.1 VIRTUAL ORCHESTRA PROJECT

This project will demonstrate how the posture recognition approach can be used along with the Augmented Reality technology to provide an interactive solution for the study and better appreciation of classical musical instruments.

Aside from the general postures, we have previously also extracted additional detailed information about the gestures. This information can be used to augment the scene visually as well as auditively by adding the visual effect of an instrument in the hands of the user as well as adding the sound of the corresponding instrument [20]. The auditive part of the augmentation can be further supplemented by adding variation of the sound according to the movements of the hands and the positions of fingers.

In essence, this means, that thanks to the gesture recognition, we can configure our system in such a way, that for example the speed of the hand movements will control the intensity of the sound for the violin, cello and drums, while the finger positions will change the pitch for the flute or clarinet.

Finally, both the hand movement and finger positions will control the loudness and the pitch for the trombone. An example of scene augmentation is show in Figure 7.

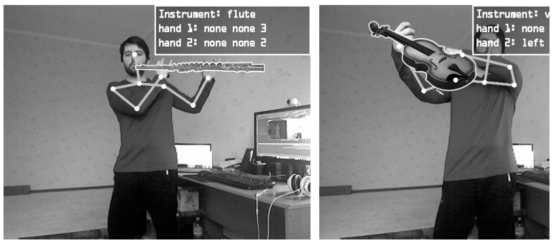


Figure 7 – Example of visual scene augmentation.

Additional information is displayed for clarity, such as the name of the instrument, as well as information about each hand, i.e. movement direction, movement speed and number of fingers.

As for the sound augmentation, our system includes a collection of WAVE sound files that contain the sounds of every instrument on the system. For each instrument we keep several sound files. For movement-based instruments we have three sound effects for every movement speed gradation. For the "finger movement" instruments the system has up to five sound effects with different pitches that match different finger positions. For example, there are three sound files for the violin (one for each movement speed) and five sounds for the flute (here we are only counting the number of fingers on the left hand). After the pose is recognised, our system decides whether it should take into account the hand movements or the finger positions. Then the system extracts the additional features (movement or fingers) and plays the matching sound.

4.2 MAGIC MIRROR PROJECT

In this project the posture and gesture recognition approach is used as a basis for developing an interactive game for the Berlin Concert Hall. The purpose of the game is to attract new visitors and familiarize them with the classical music pieces and to teach them about different musical instruments.

The players are met with a musical piece, during which they are asked simple questions about different musical instruments and then encouraged to give their best shot at demonstrating the way a musical instrument is played.

In particular, the game will ask the player "How is this instrument played?" and show him the image of a musician playing, for instance, a clarinet. The player then has to assume the pose associated with playing a clarinet and the game will grade him on his performance.

To do that the posture recognition approach extracts the skeleton joints using Kinect and recognizes the posture (Fig. 1). The chosen features as well as the use of the *k*NN algorithm allows the system to calculate a score from 0 to 1, which shows how close the player has gotten to matching the posture he was asked to demonstrate. If the player failed to do the pose correctly in the short allotted amount of time or did not reach a high enough score, the game will not count the attempt as successful.

Figure 6 demonstrates the concept: we can see that the player has already been previously asked to show how the drums are played, and he has been assigned 2219 points for his performance. This means that he managed to show the correct pose almost perfectly, approaching a score of approximately 0.9 of 1.

In the current round he has been asked to do the clarinet pose and we can see that the player is only barely approaching the 0.5 mark, which means the game is not going to count this attempt as successful.

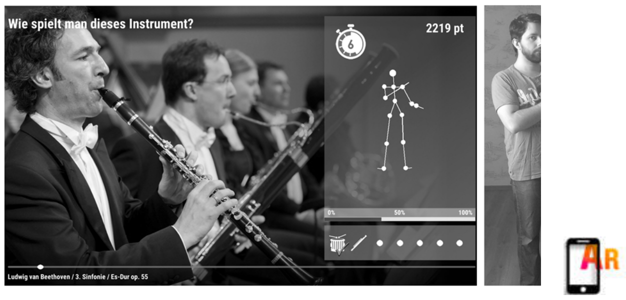

Wie spielt man dieses Instrument?
2219 pt
6
0%
50%
100%
Ludwig van Beethoven / 3. Sinfonie / Es-Dur op. 55
AR

Figure 8 – A screenshot from the Magic Mirror game where the player is asked to demonstrate how a clarinet is played.

In the end, the player receives his total score and is shown the leaderboard, where he has a chance to see himself if his score is high enough.

5. CONCLUSION

We have developed a posture recognition approach that utilizes a Kinect device to extract the visual features. The approach classifies the user's posture using a set of robust features with the help of a *k*-Nearest Neighbours algorithm. Additional hand features are then extracted to provide detailed information about the gesture.

We then demonstrate the efficiency of the developed approach using two applications: a Virtual Orchestra project and the Magic Mirror game.

The first one uses the Augmented Reality technology to overlay the instrument that matches the recognised pose so that an image of the instrument is placed in the user's hands. The sound of the corresponding musical instrument is played, with its speed and intensity depending on the movements of the user, as well as the positions of the user's fingers. This means, if the user is imitating playing drums, faster hand movements will result in a faster and louder drum sound or while moving the fingers to play the flute the sound will change corresponding to the position of the fingers and the speed of the movement. In the future, we are planning to improve the gesture recognition by implementing an algorithm that would analyse the hand motion pattern, which would allow the recognition of a much wider range of gesture and provide finer control over how the instrument sounds. Using the detailed information about the hand positions to implement Augmented Reality with the correct placement of 3D instrument models is also a possibility.

The Magic Mirror app is a game developed for the Berlin Concert Hall in collaboration with the INKA team that employs the posture recognition approach in a game to provide interactive experience to the visitors of the Concert Hall, attract new visitors and interest them in classical music. The Magic Mirror game was tested during the Doors Open Day 2017 in the Concert Hall and sever avenues of future work were considered. In particular, we are planning to add functionality for several people to be able to play the game at the same time, which is already supported by our posture recognition approach. We also plan to integrate our detailed gesture recognition into the Magic Mirror app for more advanced game modes.

ACKNOWLEDGMENTS

This paper describes the work undertaken in the context of the project APOLLO hosted by the research group "Information and Communication Systems" INKA that is gratefully funded by the European regional development fund (ERDF).

8. REFERENCES

[1] F. J. Detmer, J. Hettig, D. Schindele, M. Schostak, C. Hansen, "Virtual and augmented reality systems for renal interventions: a systematic review," in *IEEE Reviews in Biomedical Engineering*, vol. 10, pp. 78-94, 2017.

[2] K. F. Lee, Y. L. Chen, H. C. Hsieh, K. Y. Chin, "Application of intuitive mixed reality interactive system to museum guide activity," *Proceedings IEEE International Conference on Consumer Electronics*, Taiwan (ICCE-TW), Taipei, 2017, pp. 257-258.

[3] L. A. Hernández-Ibáñez, V. Barneche-Naya, R. Mihura-López, "A comparative study of walkthrough paradigms for virtual environments using kinect based natural interaction," *Proceedings of the 22nd International Conference on Virtual System & Multimedia (VSMM)*, Kuala Lumpur, 2016, pp. 1–7.

[4] S. Serafin, C. Erkut, J. Kojs, N. C. Nilsson, R. Nordahl, "Virtual reality musical instruments: state of the art, design principles, and future directions," in *Computer Music Journal*, Vol. 40, No. 3, pp. 22–40, 2016.

[5] W. Lages, M. Nabiyouni, J. Tibau, D. A. Bowman, "Interval player: Designing virtual musical instrument using in-air gestures," in *Proceedings of the IEEE International Conference 3D User Interfaces (3DUI)*, 2015.

[6] H. Kanke, Y. Takegawa, T. Terada, M. Tsukamoto, "Airstic drum: a drumstick for integration of real and virtual drums," in *Advances in Computer Entertainment*, pp. 57–69, Springer, 2012.

[7] P. Sarang, A. More, A. Gaikwad, T. Varma, "Air drum using Kinect and Arduino," *International Journal of Computer Science and Information Technologies*, Vol. 6, Issue 2, pp. 1153-1155, 2015.

[8] M. Kovalenko, J. Sieck, S. Anotshchuk, "Markerless augmented reality approach based on hand posture and gesture recognition," in *Proceedings of the International Conference "Kultur und Informatik: Augmented Reality"*, 2016, pp. 171–182.

[9] W. Liu, Y. Fan, Z. Li, Z. Zhang, "RGBD video based human hand trajectory tracking and gesture recognition system," *Mathematical Problems in Engineering*, Volume 2015, Article ID 863732, 15 pages, 2015.

[10] I. Hwang, H. Son, J. R. Kim, "AirPiano: Enhancing music playing experience in virtual reality with mid-air haptic feedback," *Proceedings of the IEEE World Haptics Conference (WHC)*, Munich, 2017, pp. 213-218.

[11] R. R. Hariadi, I. Kuswardayan, "Design and implementation of virtual Indonesian Musical Instrument (VIMi) application using leap motion controller," *Proceedings of the International Conference on Information & Communication Technology and Systems (ICTS)*, Surabaya, 2016, pp. 43–48.

[12] N. Burks, L. Smith, J. Saquer, "A virtual xylophone for music education," *Proceedings of the IEEE International Symposium on Multimedia (ISM)*, San Jose, CA, 2016, pp. 409-410.

[13] Open-source SDK for 3D sensors – OpenNI. [Online]. Available: http://www.openni.ru/, Retrieved January 18th, 2017

[14] Open Computer Vision Library, Retrieved January 17th, 2017, [Online]. Available: http://opencv.org/.

[15] J. Hou, H. Gao, Q. Xia, N. Qi, "Feature combination and the kNN Framework in object classification," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 6, pp. 1368–1378, June 2016.

[16] M. Kovalenko, J. Sieck, S. Anotshchuk, "Real-time hand tracking and gesture recognition using a semantic-probabilistic network," *Proceedings of the 16th IEEE International Conference on Computer Modelling and Simulation*, Cambridge, 2014, pp. 269-274.

[17] A. B. M. T. Islam, C. Scheel, R. Pajarola, O. Staadt, "Robust enhancement of depth images from Kinect sensor," *Proceedings of the IEEE International Conference on Virtual Reality (VR)*, Arles, 2015, pp. 197–198.

[18] N. Shirbhate, K. Talele, "Human body language understanding for action detection using geometric features," *Proceedings of the 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, Noida, 2016, pp. 603–607.

[19] N. H. Nguyen, K. Doğançay, "Improved pseudolinear Kalman filter algorithms for bearings-only target tracking," in *IEEE Transactions on Signal Processing*, issue 99, pp. 1–1, 2017.

[20] F. Berthaut, M. Hachet, "Spatial interfaces and interactive 3D environments for immersive musical performances," in *IEEE Computer Graphics and Applications*, Vol. 36, No. 5, pp. 82–87, 2016.

[21] M. Kovalenko, S. Antoshchuk, J. Sieck, "Virtual orchestra: an interactive solution for a concert hall," in *Proceedings of the XV International Conference "Culture and Computer Science,"* Berlin, 2017, pp. 169–179.

Svitlana Antoshchuk *received her Dr. Habil. degree in Computer Science in 2005. Currently a professor and the Director of the Institute of Computer Systems in Odessa National Polytechnic University, Ukraine. Her scientific interests include Information Technology; Computer Vision; Artificial Intelligence Systems. Supervisor of Master and PhD students in areas of computer vision.*

Mykyta Kovalenko*, born in Odessa, Ukraine. Received his Master's degree in Informatics in 2011 and his PhD in Information Systems in 2015 from the Odessa National Polytechnic University, Ukraine, where he worked as an Assistant Lecturer since 2011 and is currently a Senior Lecturer since 2015.*

Works in the field of Image Processing. His scientific interests include Artificial Intelligence and Machine Learning, Pattern Recognition and Augmented Reality.

Jürgen Sieck*, received his degree in mathematics in 1981 and his PhD in computer science in 1989 from the Humboldt University Berlin, Germany. Now he is the head of the research group "Informations- und Kommunikationsanwendungen" (INKA) and professor for computer sciences with a specialisation on algorithms, multimedia, mixed reality and mobile application at the University of Applied Sciences HTW Berlin. Previously, he was visiting professor at Monash University Melbourne, Australia, at the University of Cape Town, South Africa and at Old Dominion University, USA. In February 2013 he was awarded an honorary doctorate from Odessa National Polytechnic University, Ukraine.*

Since 2013, he is Principal investigator of the cluster of excellence "Bild Wissen Gestaltung" at the Humboldt-University Berlin.

Since 2015, he is also a professor of computer science at Namibia University of Science and Technology.