PianoAR

STUDENTS:

Gal Shalom Ariel Iny Alex Bondar

SUPERVISORS:

Boaz Sterenfeld Yaron Honen

Table of Content:

Abstract

As students who learn B.Sc. in Computers Science, we aspire to use computers and programs to make complex tasks simpler and easier to do.

We decided to develop an augmented reality application which will teach people to play a piano. We used Hololens' AR properties as our AR platform of development.

Our application uses the Hololens' webcam to recognize the piano keyboard. We used image processing to be able to recognize the piano and to detect the press of a key. While learning to play a song, the application will color the next key to press and will indicate with color change if the correct key was pressed.

We hope this application will be an example of the possibilities the AR world has to offer.

Introduction:

'Piano AR' is an AR application which aim to teach how to play a piano to anyone without assumption on previous knowledge of music and piano playing. By using AR application, we are able to use a real life piano as our teaching instrument – thus, making the teaching process as close to teaching by a human instructor as possible. Our application shows only a glimpse of the possibilities which lay in AR applications world.

The main components of our application is:

- Main menu:
  - Activated by voice commands or hand gestures.
  - Easily understandable and intuitive.
- Piano recognition:
  - Useable on any piano.
  - Placing a virtual piano on the real life piano.
- Keyboard calibration:
  - Using a green thimble to detect the tip of the finger.
  - Calibrating the middle octave to be detected.
- Freestyle play:
  - Detecting the press of a key and coloring the corresponding key.
- Song learn:
  - Marking the current key to be pressed.
  - Visual feedback according to the user correctness.

Develop Environment:

- **<u>Unity</u>**:
  - A cross-platform game engine that can be used to create both three-dimensional and two dimensional games, as well as simulations for desktops and laptops, home consoles, smart televisions, and mobile devices.
  - Unity is scripted with C# in Visual Studio.

- **<u>Visual Studio</u>**:
  - Visual Studio is an integrated development environment (IDE) from Microsoft.
  - It is used to develop computer programs, as well as web sites, web apps, web services and mobile apps.
  - Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight.
  - It can produce both native code and managed code.

- **<u>HoloLens</u>**:
  - Microsoft HoloLens is the first self-contained, holographic computer, enabling you to engage with your digital content and interact with holograms in the world around you.
  - known under development as Project Baraboo, is a pair of mixed reality smartglasses developed and manufactured by Microsoft. HoloLens gained popularity for being one of the first computers running the Windows Mixed Reality platform under the Windows 10 operating system. The HoloLens can trace its lineage to Kinect, an add-on for Microsoft's Xbox gaming console that was introduced in 2010.
  - Getting around HoloLens is a bit different from using Windows on other devices. Instead of moving the cursor with a mouse, you use your gaze. And instead of clicking or tapping, you use hand gestures, your voice, or the HoloLens clicker.

- **OpenCV**:
    - openCV is an open source library used for implementing Computer Vision algorithms, hence the name. The library provides the needed infrastructure for developing computationaly efficient, real-time computer vision applications.
    - The libraries provided in openCV vary from basic data structures such as Matrices and Vectors to complex algorithms such as GrabCut and SIFT.
    - openCV was introduced with a basic C interface, which was later updated to a C++ oriented one. Nowadays, openCV also has a full Python interface as well as a Java one, and was ported to various different platforms including Android and iOS. The portability of openCV made it ideal for our needs.
    - Though the final objective was to get the algorithm up and running on an Android device which uses the Java interface, we've decided to work primarly with the C++ interface. Thus retaining the portability of our code and allowing us to easily use other C++ libraries if we choose to.
    - Our main use of openCV was simplfing the working progess using the bulit in data structures, input and output functions and other general utilities, the actual algorithm was programmed as part of the project.

- **Mixed Reality Toolkit V2**:
    - The MRTK is an open source toolkit that has been around since the HoloLens was first released. After 3 years of listening to the feedback of HoloLens developer community, built MRTK v2 to take the biggest concerns into account.
    - The MRTK v2 with Unity is an open source cross-platform development kit for mixed reality applications. MRTK version 2 is intended to accelerate development of applications targeting Microsoft HoloLens, Windows Mixed Reality immersive (VR) headsets and OpenVR platform. The project is aimed at reducing barriers to entry to create mixed reality applications and contribute back to the community as we all grow.
    - Feature areas:
        - Input System
        - Voice Commanding
        - Gaze + Select
        - Teleportation
        - UI Controls
        - Solver and Interactions
        - Controller Visualization
        - Spatial Understanding
        - Diagnostic Tool
        - MRTK Standard Shader

Application Overview:

Main Menu

First the getting to the Main Menu, currently we have in the menu only the recognition button.

After clicking it or saying "Recognition", we moving to the recognition scene.

Recognition Scene

After reading the instructions, you can say "start" for starting the piano recognition, and after all the piano is being marked by the outline you say "stop" and you can see the model of the piano on top the real piano.

At the end, you need to say "done" and you move back to the menu that is now having one move button for piano key calibration.

Piano Key Calibration Scene

After getting into the keyboard calibration scene, you can say "start" after reading all the instructions.

Now the keys of the relevant Octave, is colored with green.

After saying "start" ones again, you can say the first key to calibrate in green, you need to put your finger in the current key and say "next" for the calibration of the next key. Keep going so, till you calibrating all the keys.

At last, you will go back to the Menu.

Piano Play Scene

After getting into the Play sence, you can say "one" for FreeStyle, "two" for "Mary had a little lamb" song, "three" for "twinkle twinkle little star" song.

Key in Yellow color is signs for key that next to be pressed,

Key in Green color is signs for key that was pressed correctly,

Key in Red color is signs for key that was pressed incorrectly.

Development Process:

As part of our development process, we started on the learning of Image Processing, 3D modeling and more.

In addition, we learn how to use and 101 develop in Unity, and after that we move to 101 develop applications for HoloLens.

After we got familiar with the basics of Unity, Images Processing, OpenCV and Unity for Hololens(via MRTK),

we started to recognize the piano.

As part of the recognition of the piano we tried to use Vuforia, DL and other ways of Object recognition.

Eventually, we decided because of all the limitations of the HoloLens to use Image Processing using OpenCV library for HoloLens. That was accomplished using assumptions on the geometry of the Piano Keyboard(like black/white colors, piano size and number of keys).

Because the OpenCV is 2D image Processing, we needed to convert the coordinates of the piano that we recognize, to the 3D world.

Ones again, we discovered that HoloLens doesn’t have any 2D to 3D conversion out of the box.

So, in order to do that, we used as a solution a “Laser”. We shooting 4 lasers from the point of view of the user(the HoloLens camera) through the 2D canvas, and it strikes into the Spatial mapping of our space/room.

One more challenge we had in the way, was the recognition of player’s fingers. And ones again, the HoloLens platform didn’t have any solution for hand/finger recognition. The main problem was with the limitation of HoloLens on object recognition under 2 meters.

Our solution for this was to recognize the fingers with Image Processing(using OpenCV). We were able to recognize the fingers and the tapping, under the assumption that we don’t move the head after the calibration of the keys through the learning the song.

Future work:

We think our project can be further improved. There are many directions one could further develop this project, and to name some:

- A transition from Hololens v1 to Hololens v2 - Hololens v1 processing power was a major limitation of our development. By moving to development on Hololens v2, we believe the application will have a better FPS.
- Improving the key press detection – our key press detection algorithm isn't accurate as we would like to. Additional work on the algorithm should be done.
- Overcoming the "No head movement" limitation – one of the main drawback of our application it is the limitation of the user on its head movement. The user isn't allowed to move its head at all for the application to run correctly.
- Additional work on the piano recognition – On some rare cases, e.g. bad lights, our piano recognition might malfunction. One can try to improve our algorithm to handle those cases too.

We hope our project to be a base of many future projects and to inspire students and developers to research the world of AR.