Piano player

This project was developed in a ROS environment using the Robotis OP3 Framework, Python 2.7, OpenCV 3, and the United-Ros-Platform.

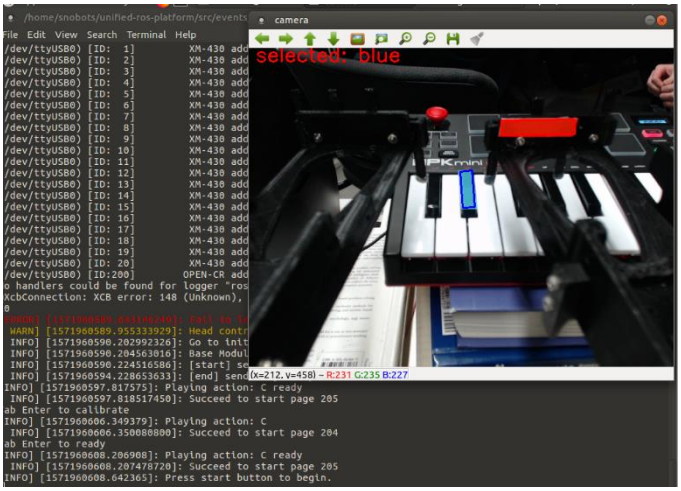
Vision

The “Object Detector” project was incorporated and improved to help in my implementation. Instead of ball recognition, a rectangle recognition was implemented. The function will return the x, y, width, height, area and size of the object detected. Because this is in ROS, those values will be published as an ObjectCoords message and will be used by the piano player script.

Motions

Using the Robotis-OP3 Action Editor, I created hard-coded motions to play all the keys in an AKAI MPK mini play keyboard, with a fixed distance and positioning. The key-playing was designed given the number of fingers we give to the robot: if we use one finger per hand, the robot will play each individual white key, the left hand oversaw the bass keys (to the left of the central C) and bice-versa for the right key. The other option was to use one finger in the left and two in the right to play chords; for simplicity only some chords were used, which some of them were chord variations.

Visual servoing



Using two markers (one on the robot’s right hand and the other on the middle B), I first placed the robot in the perfect position for playing, recorded the coordinates and calculated the optimal distance. Then, implemented a function called align_hand(), which will first move the hand left or right, depending on the distance along the x-axis and then will move along the y-axis (down), comparing the y and height of each marker so it knows how hard to press.

The servos will be moved by 1 degree, using Robotis’ direct control for each arm. After all the visual servoing is done, the goal angles will be recorded. These angles will become in the “home” angles, which everything will be based on

Inverse Kinematics

Now, for simplicity, I used the bass line of Billie Jean by Michael Jackson for the calculation of all the keys needed to play it, which are only 4. Based on the keys to be reached and the keys’ width, plus the values got from visual servoing, the coordinates for those keys were calculated by a formula only for the shoulder roll servo. These calculates values were then recorded to a json file, which it is used to demo the song

Results

As expected, using only motions only worked when positioning the robot in the correct distance away from the middle C. The motions for Billie Jean went until 78 bpm, very slow comparing to the original 120 bpm. If positioning correctly, it will work 10/10.

The results were better with inverse kinematics. If the robot was positioned a little far from the middle C, it could find the correct position and play it with no problem. Because this was a calculation mostly in 2d, there will be scenarios when it will fail if the robot was too close or too far from the keyboard, also because of arm limitation it won't arrive to the goal coordinate (for example is it is 10cm to the right of middle C). The speed was a bit better (around 85 bpm), and it could be done faster (like 90-100 bpm), but it won't give the servos enough time to arrive to the goal positions. I researched a little bit how this could be improved: one is writing the values directly to servos (without publishers) like the walking module does, the other one and more simple is to use the framework's set_joint_states, somehow it is not even working when using the read_write demo from Robotis, this may be because a bug which was not updated to the newest version or because the change it was done to the framework to support both Polaris and OP3.