ME 134 Final Report

Diana Frias Franco, Chase Blagden, Joaquin Gomez, and Lorenzo Shaikewitz

I. Overview and Basic Task

We make a piano playing robot that is capable of playing from a pre-selected list of 10 songs from different starting positions and orientations of a keyboard.

II. The Robot

We constructed a 7 DOF robot with a panning base joint and two 3 DOF arms. Walking from base to tip: we have a pan joint connected to a bar. On each end of the bar is a 3 DOF (of the same style we built: pan joint, then two up/down joints). At the end of each 3 DOF arm will be a gripper, which closes to act as a finger. The upper and lower links for both arms are 30 cm.

III. Kinematics

We have coordinates in three different spaces: relative to the robot, the table, and the keyboard. When we receive a particular note to play, we compute its offset relative to a fixed point on the keyboard. We then compute the location of the key in the robot space using the position and orientation of the keyboard in the table space and the key's location in the keyboard space.

All spaces are done in Cartesian coordinates relative to the two tips of the closed robot grippers. We use quintic splines to interpolate between positions, and when moving between multiple notes we first move to a position above the note before moving down to press it and then back up again to the same position.

To ensure the robot is always facing the right way, we have a secondary task on the pan joint enforcing that the base faces the center of the keyboard.

We have a weighted pseudoinverse to handle points near singularities, and we will check the singlar values and return to floating mode if one of them is too low.

IV. Detectors

We use the RGB images of a Realsense D435 camera sending 950x540 images at 30 Hz. The system uses the four Aruco markers at known positions on the corners of the table to find the translation and rotation of the camera relative to the table, and then uses two Aruco markers on the keyboard to determine it's postion and rotation in terms of the table coordinates. To calculate the rotation we compare the positions of the markers on the corners of the keyboard, whereas for the position we just use one of the markers. The detector itself runs at 30 Hz, and recomputes the transforms live as long as the four corner tags are available. The detector will send out the last known position and orientation of the keyboard if one of the keyboard markers is covered.

Fig. 1. The robot.

V. Software Architecture

The main node is a planner node which does the main motion planning and inverse kinematics to control the robot at 100 Hz. It listens to a song node, which in turn listens to midi keyboard events at 30 Hz to determine what song to play and whether or not the robot should be in teaching mode. The planner also listens to a keyboard detector node, which outputs the position and orientation of the keyboard at 30 Hz. The planner node will then output commands to the joints which go to the HEBI node (running at 100 Hz) which then sends the joint commands to the motors themselves. The planner node also outputs the expected pose of the robot. The planner is a state machine that will switch between a floating state, when only the gravity model is being applied, and a play state that will listen for and then play a particular song or note.

VI. Behaviors

By default, the robot starts in "floating" mode, meaning the arms are only running their respective gravity models. In this mode, the arms are free to be moved as the robot awaits input. Input is provided via the keyboard - when the first 3 notes of any song the robot knows are played on it, the robot will load the song and begin playing it given the keyboard's current position and orientation.

The robot will play the keyboard starting from any position and (almost) any orientation within its task space. There are two behaviors the robot will exhibit given a moving keyboard, depending on its capability to play reliably in any of these cases. If the piano begins moving slowly while the robot is playing a song, it will adjust live where it is moving to account for the movement (this is made possible by planning doing

our planning in keyboard space). If the keyboard moves too quickly, the robot will stop at its current position and move its arms to a position just above the piano before continuing to play where it was left off. Additionally, if the robot detects the keyboard has been moved outside of its "playing range" (as determined by the distance to the center of the keyboard and the orientation of the keyboard relative to the robot base), it will attempt to move the keyboard into its range: if the keyboard is too far away, the robot will pull a loop attached to its center to bring it closer; if it is too close, the robot will push it away; if it is too askew, the robot will attempt to pull the farther side closer. If any of these repositioning behaviors fail, it will re-attempt them until the keyboard is in range. Finally, if the keyboard is too far outside the playable range, the robot will not attempt to play it. The state machine graph of this is available in Figure 2.

By pressing a special button on the keyboard, the robot can also be put into teaching (or Simon) mode. The robot will then play a note from the selected song, and wait for the user to mimic it. If the user is correct, it will play the same note and then the next note in the song. If the user messes up, the robot will repeat the same sequence. In this manner, the robot can "teach" the user how to play a particular song.

Past this, if the robot hits something too hard (i.e. an effort threshold is exceeded) or if it goes too close to a singularity, then it will switch back into floating mode.

VII. Shortcomings

We always struggled with playing the right notes correctly, and the robot would often consistently miss notes towards the ends of the keyboard - although the systematic error itself would change from day-to-day. The robot itself is also very slow at playing songs - we can only play consistently play songs at speeds up to 45 BPM - whereas songs typically range from 70-90 BPM. We are also not playing notes at their true duration because we subtract out the move time between notes from how long they are supposed to be pressed so that the timing is still consistent. Although we have the grippers and manufactured an attachment to play chords, we ran out of time to actually add it to the system so that we could play chords and not just individual notes. Finally, the robot is unable to play any 5SOS song or the Ride of the Valkyries.

VIII. Appendix

- Video summary
- Github

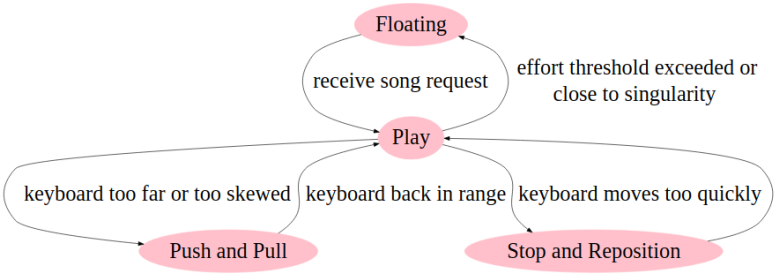

Fig. 2. A simplified graph of the state machine of the robot. The nodes represent the different possible states, and the edges represent the conditions necessary to transition between the respective states.