Pecking test computers

The pecking test is currently set up on 4 boxes, controlled by 2 computers. The computers are both Intel NUCs - Product #BOXD34010WYKH1 - with 4 USB ports and headphone audio out. The two computers are:

- Chubbyninja
  - Controls boxes 5 and 6
  - IP Address:
  - Connected to keyboard, mouse and monitor
- Pumpkin
  - Controls boxes 2 and 3
  - IP Address:

Network

- NWAF network IP:
- WiFi SSID:
- WiFi Password:
- WiFi Router admin password:

Audio

Audio is run out of the headphone jack where it connects to an amplifier. The amplifier should be adjusted to provide the desired output sound level within its corresponding boxes. The left and right channels output from the amplifier are sent one to each box that the computer controls. The ability to output sounds to a single channel of the headphone port was done using custom alsa configurations. These modifications were:

Add custom udev rules. Since Ubuntu uses udev to handle loading up devices on boot, all devices (e.g. sound cards, webcams, other usb devices) are initialized in parallel and thus their names can change depending on which one initialized first. The following rule tells udev to name the "PCH" card with the id "Analog" which is required in the asound.conf file described next.

```
# NUC
KERNELS=="card[0-9]*", SUBSYSTEMS=="sound", ATTRS{id}=="PCH", ATTR{id}="A
```

Add a custom asound.conf file to /etc. This creates interfaces that python can talk to called "speaker0" and "speaker1", corresponding to the left and right channels, respectively.

```
# Pieced together from http://alsa.opensrc.org/Asoundrc#Splitting_front_a
pcm.split {
    type dmix
    ipc_key 2048
    slave {
        pcm "hw:Analog"
        channels 2
    }
}
pcm.speaker0 {
```

```
    type plug
    slave.pcm "split"
    hint {
        show on
        description "Plays sound through the left channel only"
    }
    ttable.0.0 1
}
pcm.speaker1 {
    type plug
    slave.pcm "split"
    hint {
        show on
        description "Plays sound through the right channel only"
    }
    ttable.0.1 1
}
```

Controlling the boxes and collecting data

The boxes are interfaced with using an Arduino Uno hooked up to each chamber. The Arduino outputs 3 digital signals to control the pecking button light, the feeder, and the main house light (I don't think this one works though...). It has 1 digital input from the pecking button. The functioning of the chamber and programmatic interface to it should be discussed elsewhere, but there are a few things that are needed to make things work on the computers.

- The Arduino software must be installed
    - Can be done through `sudo apt-get install arduino`
- User must be added to the "dialout" group. This is done automatically when the Arduino software is run for the first time. You must log out and log back in before this takes effect.
- Again, udev will initialize the Arduinos in arbitrary order, so custom rules should be applied. These rules will create a symbolic link in /dev that points from a consistent and interpretable name like "ttyArduino_box2" to the randomly given name like "ttyACM1".

```
# Box 2:
SUBSYSTEM=="tty", SUBSYSTEMS=="usb", ATTRS{manufacturer}=="Arduino (www.a

# Box 3:
SUBSYSTEM=="tty", SUBSYSTEMS=="usb", ATTRS{manufacturer}=="Arduino (www.a

# Box 5:
SUBSYSTEM=="tty", SUBSYSTEMS=="usb", ATTRS{manufacturer}=="Arduino (www.a

# Box 6:
SUBSYSTEM=="tty", SUBSYSTEMS=="usb", ATTRS{manufacturer}=="Arduino (www.a
```

Video