

Animatronic Head

How it Works

Objectives

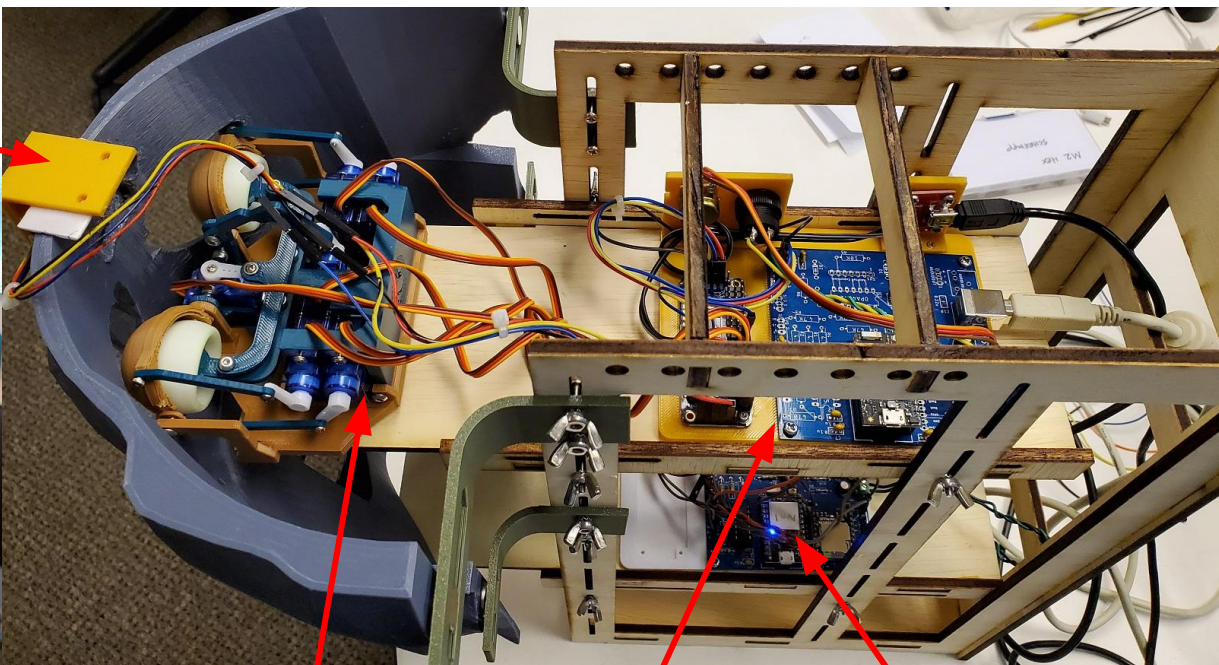
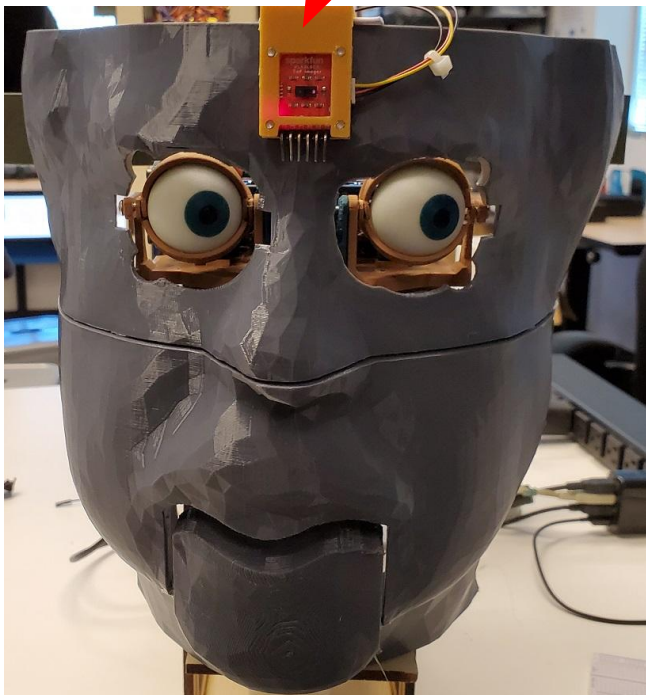
1. Relatively cheap, very creepy head that will surprise our Halloween guests.
2. A fun thing to show off at our mini-maker faires
3. Learn about animatronic control mechanisms
4. Engage creative makers in scripting and costuming the Animatronic Head for participation in other holidays and events.

All open source at <https://github.com/TeamPracticalProjects>



at Maker Nexus, Sunnyvale, Calif

TOF Sensor



Eyes
Mechanism

Eyes
Electronics

Mouth
Electronics

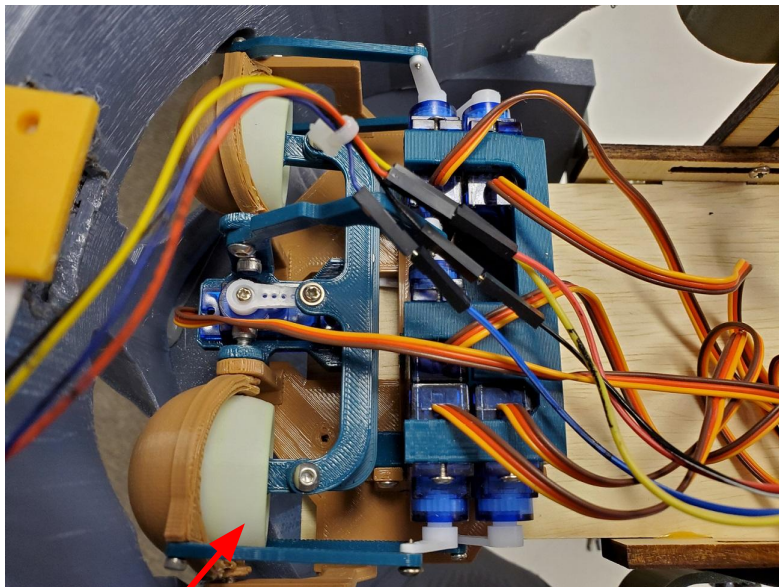
What we have now

1. A great 3D printed eye mechanism from Nilheim Mechatronics
 - a. <https://www.instructables.com/Simplified-3D-Printed-Animatronic-Dual-Eye-Mechani>
 - b. His YouTube channel: <https://goo.gl/7Cle6h>
2. A simple mouth mechanism we made ourselves
3. An 8 x 8 zone optical time-of-flight (TOF) sensor to determine where people are positioned when engaging with the head.
4. Arduino compatible code running on two Particle.io Photons
5. Larger than life 3D printed head (from a scan we made with an iPhone)
6. Various laser cut and 3D printed parts and armature to support it all
7. Custom designed printed circuit boards to house the electronics.

Architecture

- Two custom-designed PCBs
 - The boards are identical, but may be provisioned differently for the different functions
 - Boards host a Particle Photon microcontroller (Arduino compatible with integral WiFi)
- Eyes control system:
 - Connect to the TOF sensor and process sensor data to detect events, such as (1) person arrived into the field-of-view, (2) person leaves the field-of-view, (3) person too close to the head.
 - Publishes new sensor event data to the Particle Cloud (subscribed to by the Mouth control PCB)
 - Uses an Adafruit 16-channel servo control board to control the 6 servos in the eyes. Firmware on this board controls the eye motions according to sensor data and scriptable patterns.
- Mouth control system:
 - Subscribes to events from the Eyes control PCB
 - Plays pre-determined and pre-recorded audio clips for each new event
 - moves the mouth in synchronism to the audio.



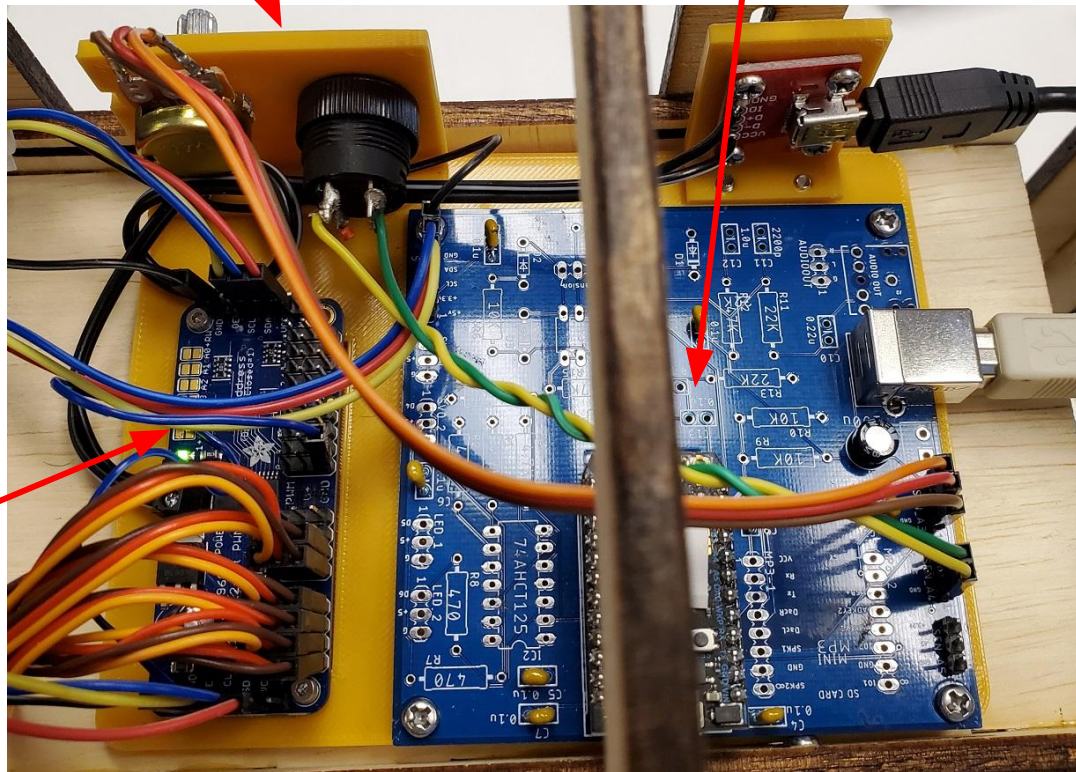


6 sevo eyes
mechanism (3D
printed)

16 channel
servo board

Eyes Calibration
Controls

Eyes Control
PCB



Eyes Animation Software Layers

main()	<p>Calls animationList.process()</p>
Class: animationList	<p>A singleton. Define a series of “scenes” (mechanism positions) and then run through them as the clock ticks.</p> <ul style="list-style-type: none">• addScene(scene, speed, delay) - predefined, or mechanism by mechanism• startAnimation()• stopAnimation()• clearSceneList()
Class: animatePuppet	<p>One for each puppet (we have only one); owns animateServo objects</p> <ul style="list-style-type: none">• eyeball.look(x, y, speed)• eyelid.open(position, speed)• mouth.open(position, speed)• Puppet<ul style="list-style-type: none">○ eyesOpen()○ blink()○ wink()
Class: animateServo	<p>An instance for each servo. Moves servo slowly, or quickly.</p> <ul style="list-style-type: none">• moveTo(x, speed)



The process() Stack

1. Mainloop calls `animationList.process()` every time
2. `animationList.process()`
 - a. notes elapsed time and decides if there is a new scene to set.
 - b. Calls `animatePuppet` to set new scene positions.
 - c. Calls `animatePuppet.process()`
3. `animatePuppet.process()`
 - a. Decides if the movement of one servo affects another (e.g. moving eyes all the way left might cause the head to rotate left)
 - b. Calls `animateServo.process()` for each servo it owns
4. `animateServo.process()`
 - a. Notes current servo position vs target servo position
 - b. Notes time and moves servo towards target position, if needed

The Mouth

- The Mouth has independent microcontroller software (from the eyes), running on a Particle Photon.
- The Mouth PCB is the same as the Eyes PCB to minimize parts buy. However, the Mouth PCB may be provisioned differently from the Eyes PCB.
- On-board MP3 audio player that can be controlled by software on the Photon microcontroller.
- MP3 Player can play audio clips directly through a small speaker or through a stereo headphone jack.
- Custom analog processing circuitry on-board:
 - Pre-amplifier and offset/scaling
 - Half wave rectification
 - Low pass filtering for envelope extraction (40 Hz)
- Software processing of the envelope data to drive the mouth servo
 - Scaling
 - Moving average
 - Optional non-linear processing
 - Mapping to mouth servo limit



Welcome



Walk-away



Pirate

Eye/Mouth Synchronization

- Triggered by external events
 - Events are processed out of TOF sensor data in the Eyes system
 - Currently supported events:
 - Person arrives into the sensor field-of-view
 - Person leaves the sensor field-of-view
 - Person is too close to the head
- The Mouth Photon subscribes to these events and plays an appropriate audio clip with synchronized mouth movements
- Provision is made for future detection of additional/new events and additional/altered audio clips
- Audio clips are in MP3 or WAV format, stored on a micro-SD card

Status and Future Plans

- Current target is Halloween 2022
 - All of the structural hardware is complete
 - New iterations of the head are in process to provide more realism and better incorporation of the TOF sensor
 - Electronic hardware is finalised and tested
 - Firmware is in the process of finalization
 - Final integration of the eye code with the scripting infrastructure
 - Final integration of the Eyes and Mouth via Particle publish/subscribe interface
- Broader engagement planned
 - Plan to engage creative people from Maker Nexus and other places to script and costume the Head for other holidays/venues