GROUP #67: ALEX BRANNICK, DARYL DRAKE

VR HAND SIMULATOR

INTRODUCTION

- Problem: VR has hit a wall
  - Interacting with environment is limited to your controller
  - Need a more immersive experience
- Solution: Your hand is the controller
  - Interact in VR without needing a controller

OBJECTIVES

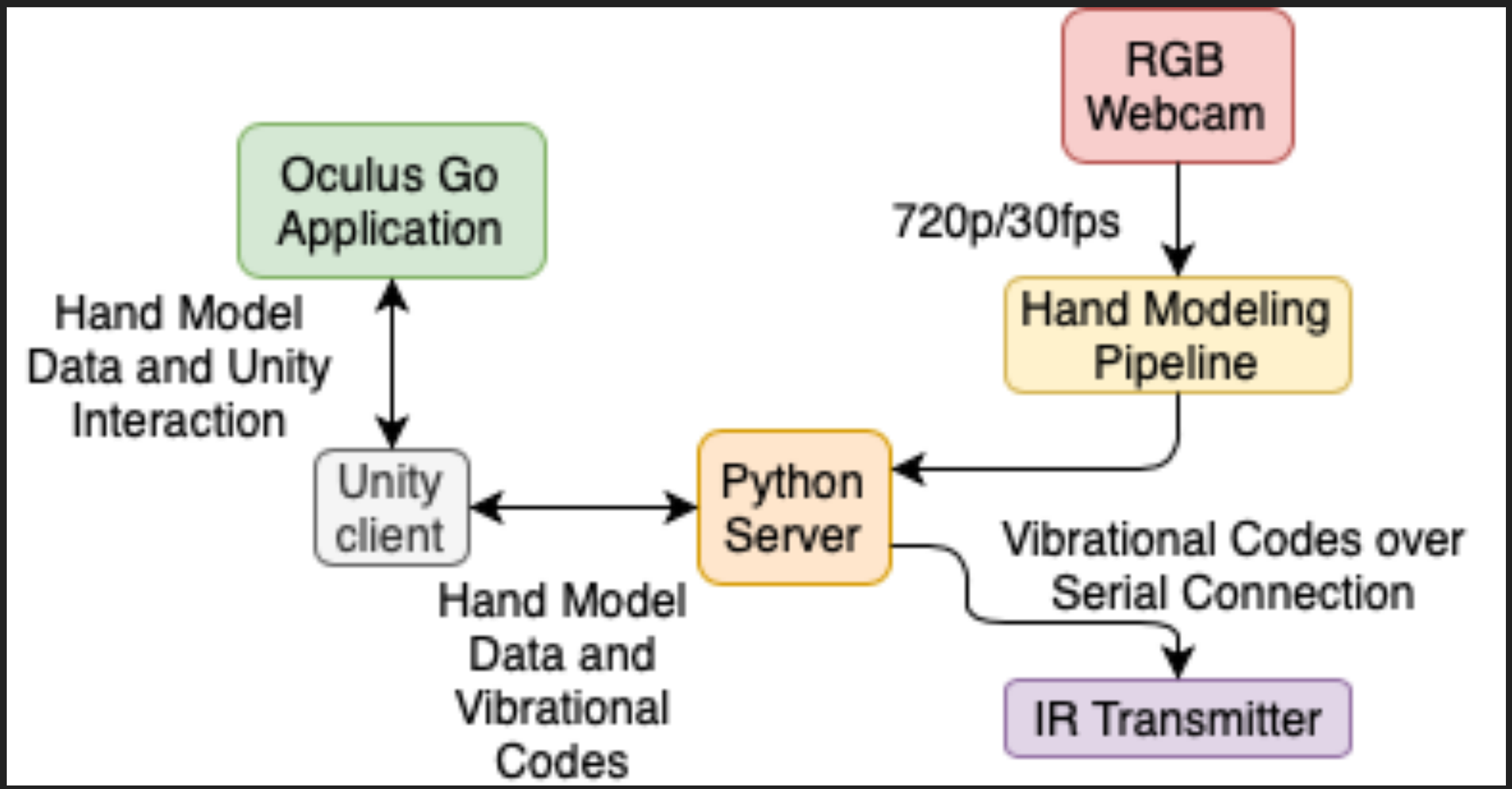
- Populate virtual environment with hand in real time
- Add sensory feedback to in game interaction
- Make the device available in Unity for game developers

DESIGN OVERVIEW

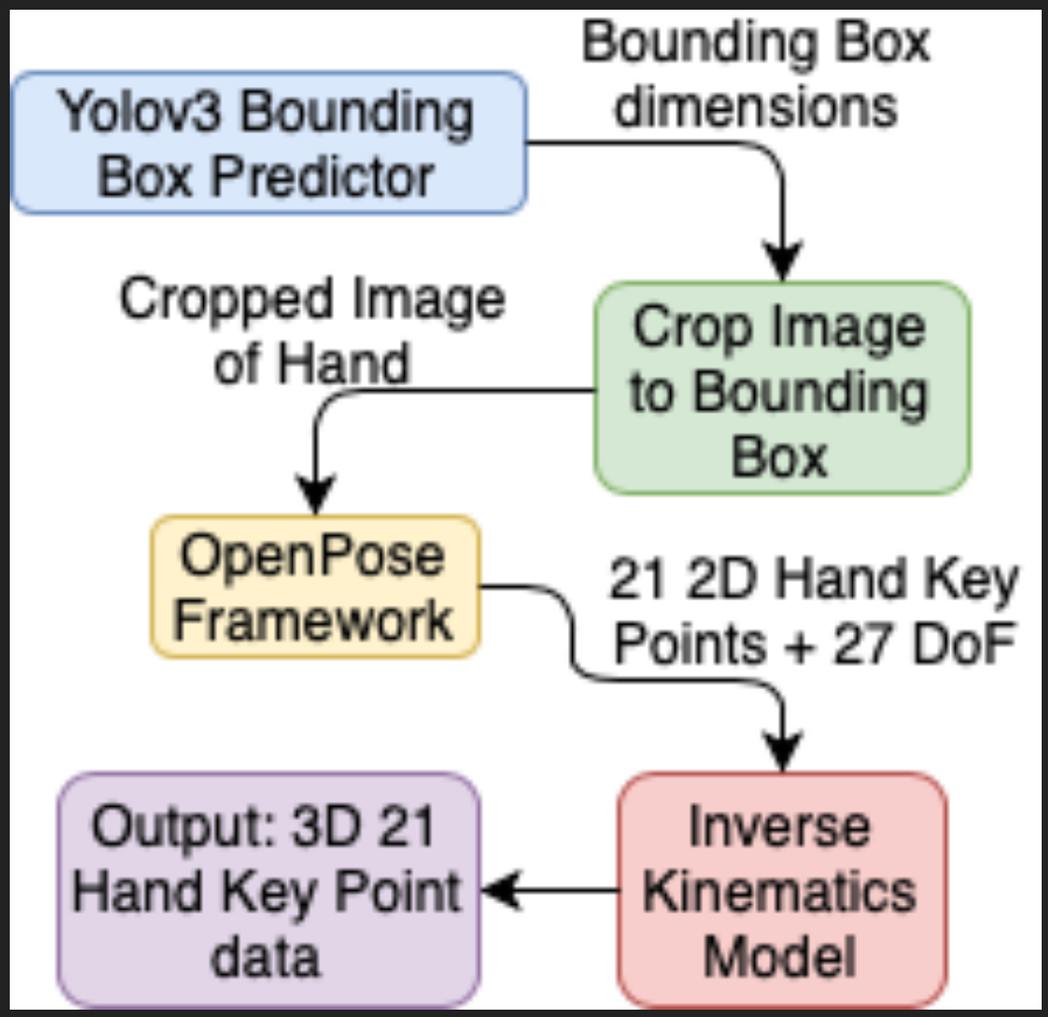
- Headset
  - RGB Camera
  - Hand Modeling Pipeline/Server
  - IR Transmitter
- Bracelet
  - IR Receiver
  - Vibrational Motor/LED circuit

PART 1: HEADSET

HEADSET BLOCK DIAGRAM

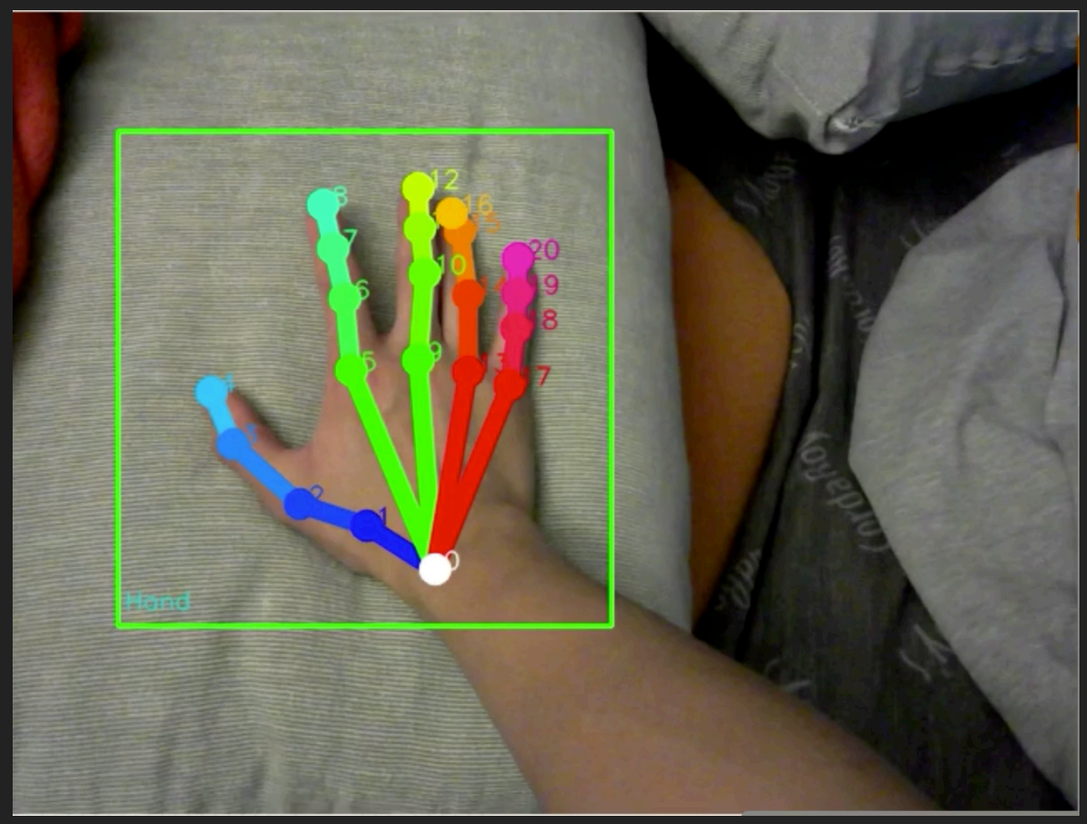


HAND MODELING PIPELINE



BOUNDING BOX PREDICTOR

- YOLOv3 Architecture
  - Real-time object detection
  - Trained on Egohands dataset
- Input: Single RGB frame
- Output: Cropped image of hand from frame

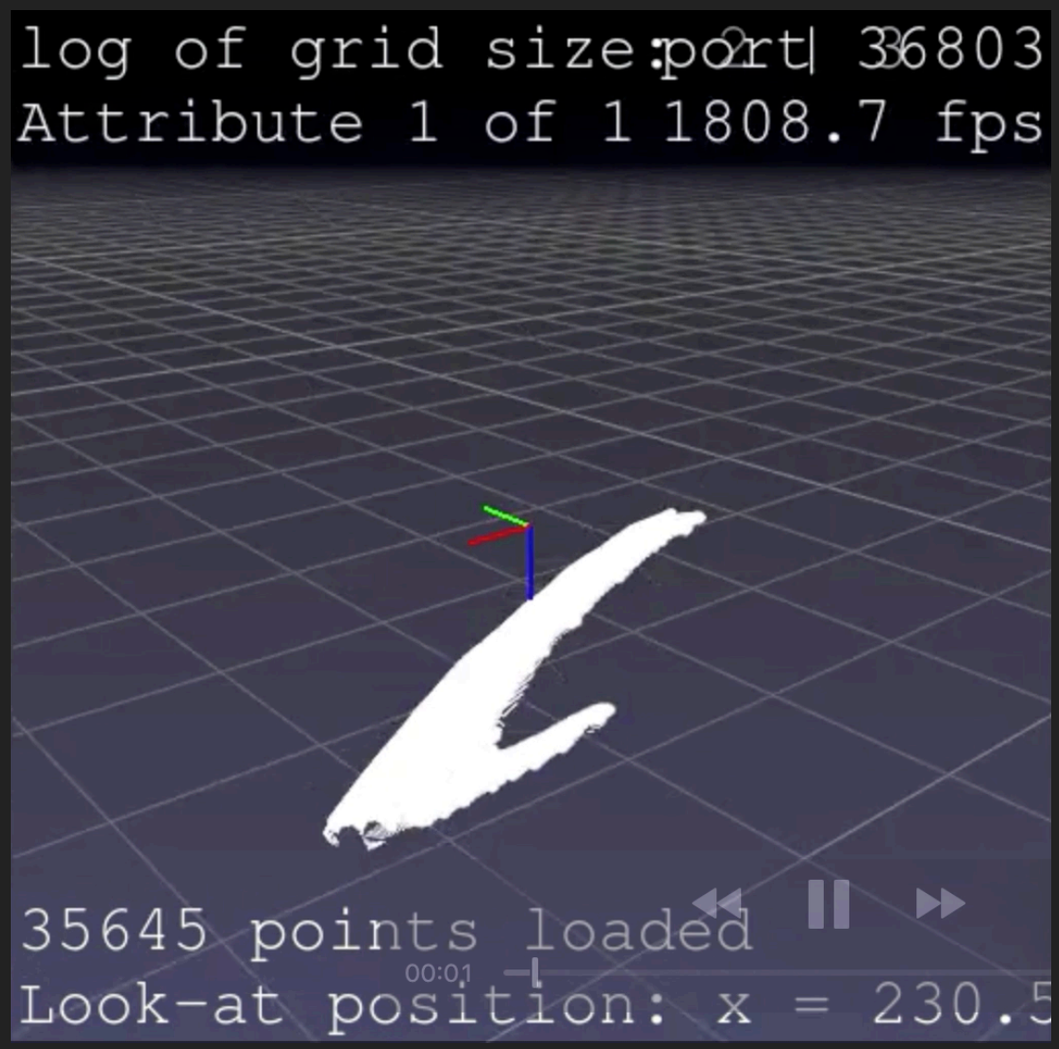


OPENPOSE

- Open-source library for hand/body pose estimation
- Input: Cropped hand image
- Runs through CNN to find hand peaks
- Converts peaks into joint estimates
- Output: 21 2D hand key points

INVERSE KINEMATICS MODEL

- Compiled Ubuntu binaries for 3D joint estimation
- 2D key points used to find 27 DoF
- Input: 27 DoF
- Output: 3D joint data



PYTHON SERVER

- Runs the hand modeling pipeline
- Works on the local network
- Responsible for sending 3D hand data and vibration codes to the bracelet
- Client must poll server to receive hand data

UNITY PLUGIN (CLIENT)

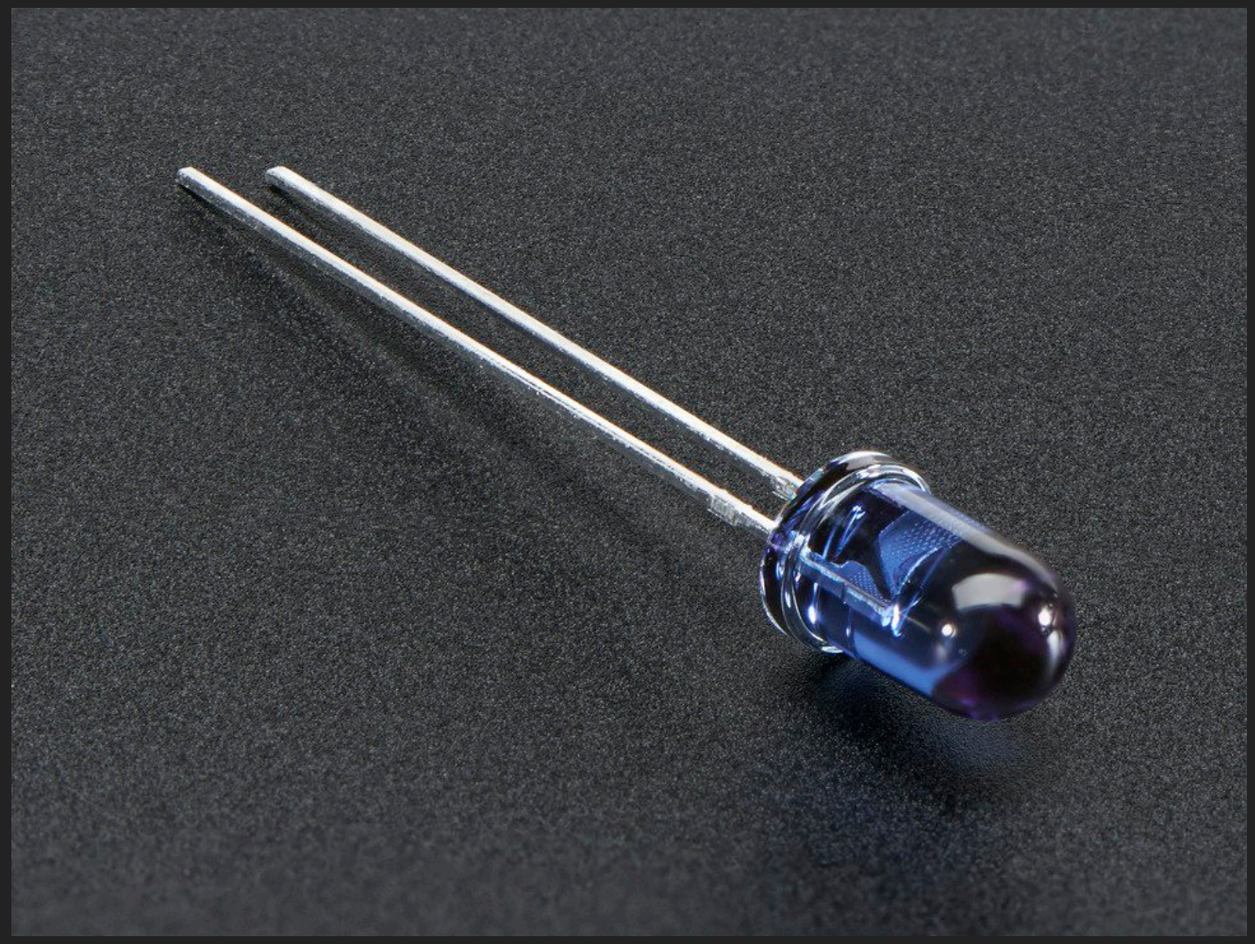
- Interface for sending and receiving data to the Oculus
- Works when server running on local network
- Polls server for hand data
  - Sends vibrate codes to server as well
- Interface for Unity developers to use in their games

UNITY DEMO GAME

- Simple game for showing proof of concept
- Uses client interface to populate hand model
- Creates hand of 21 joint colliders
  - “Collider”: triggers event when touched

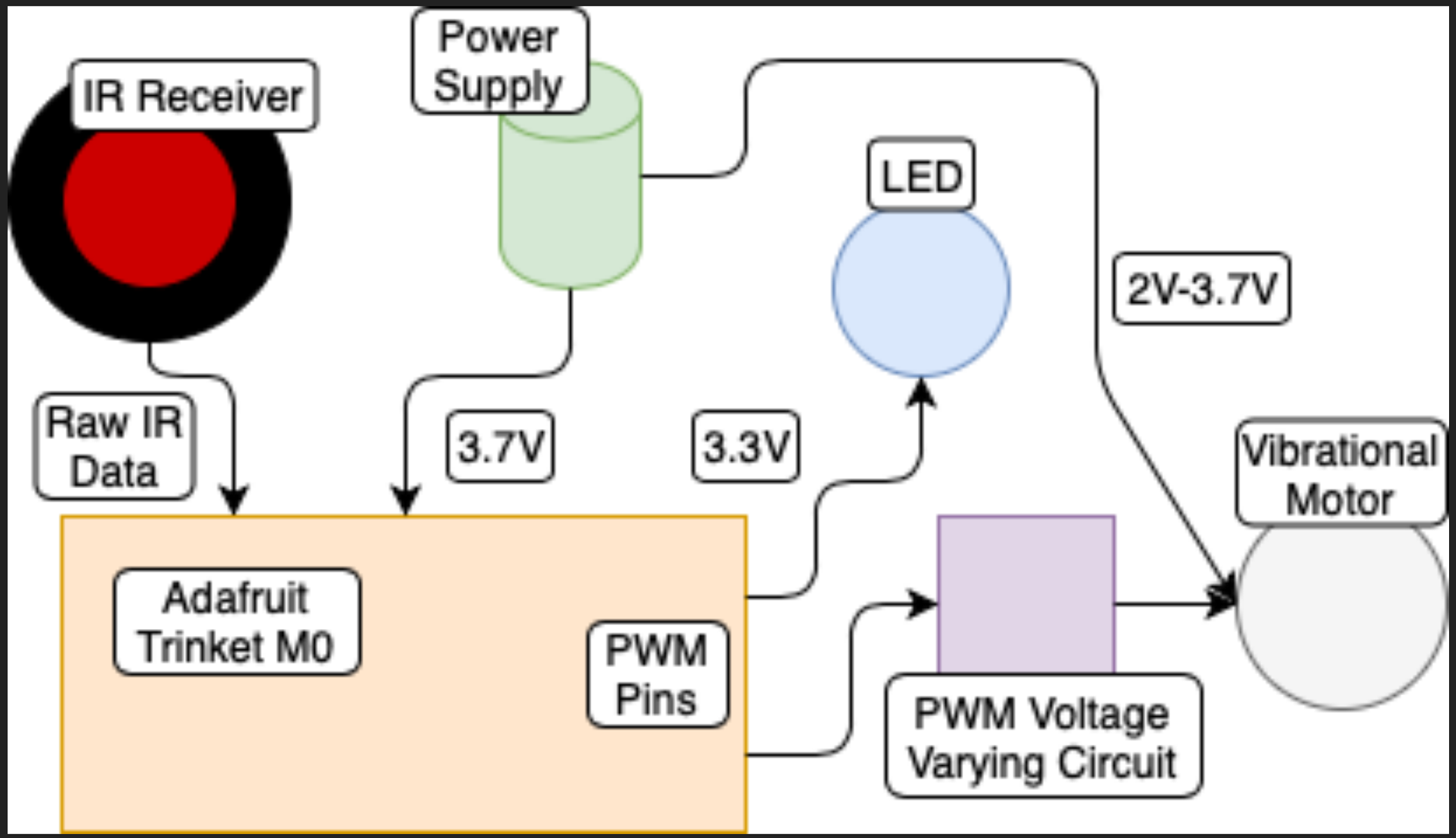
IR TRANSMITTER

- IR LED, <$1.00
- Sends message to vibrate bracelet
  - Uses standard IR codes (like tv remote)
- Three codes for three levels of vibration
- Notified by python server what to send



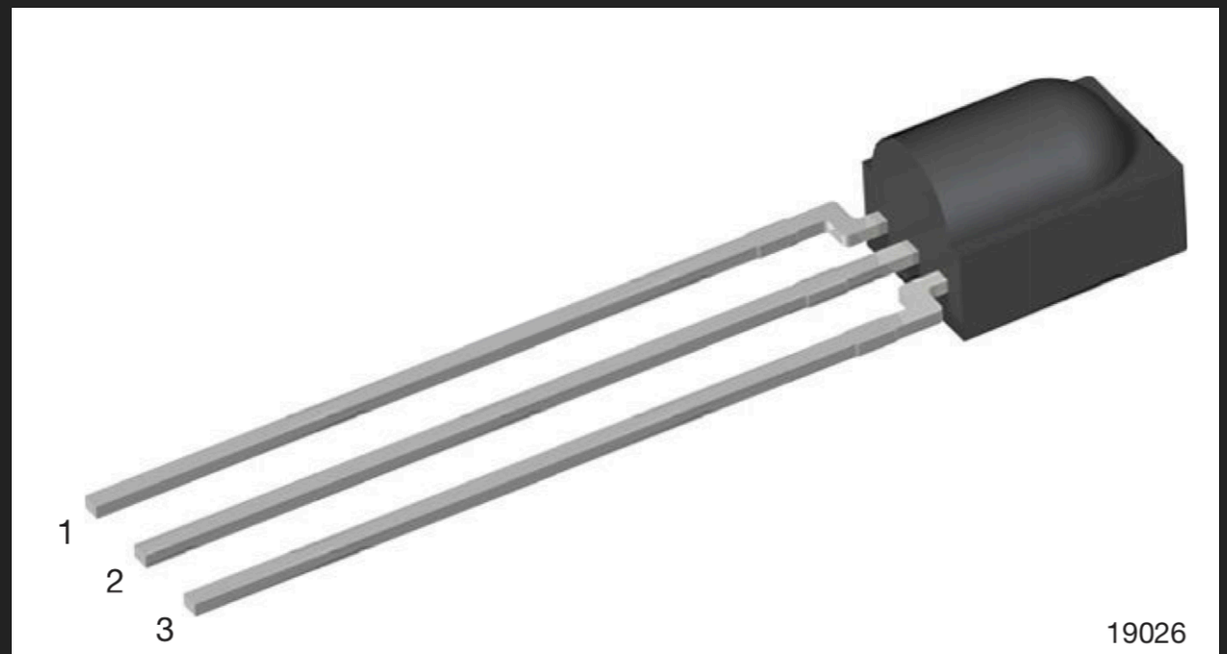
PART 2: BRACELET

BRACELET BLOCK DIAGRAM



IR RECEIVER

- $1.95/piece
- Outputs raw IR data from transmitter
- Data interpreted by IR library



DESIGN SUPPORT TOOLS *click logo to get started*

3D Models Available

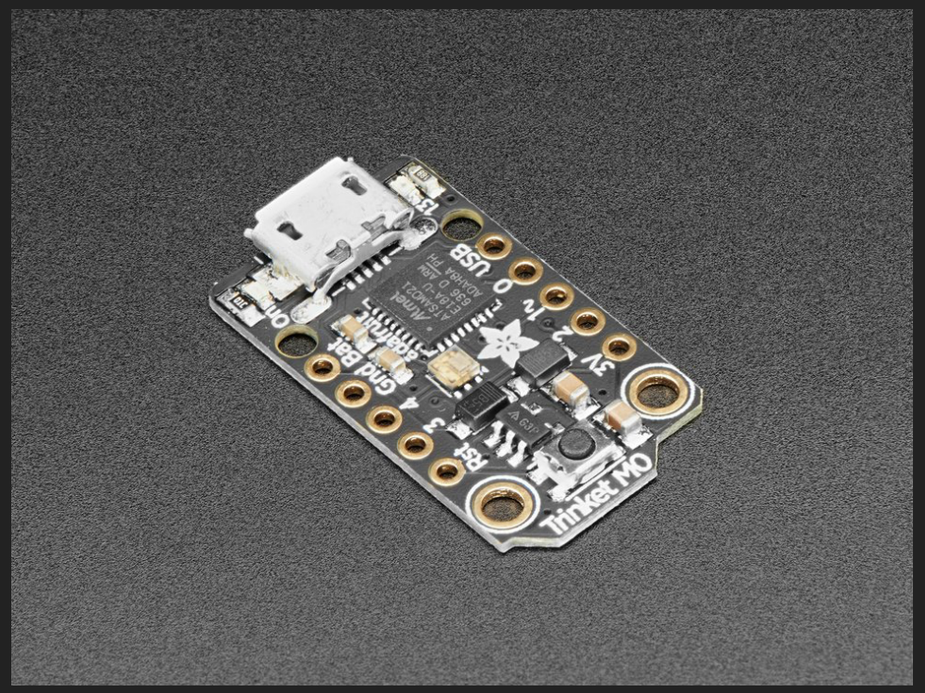
MECHANICAL DATA

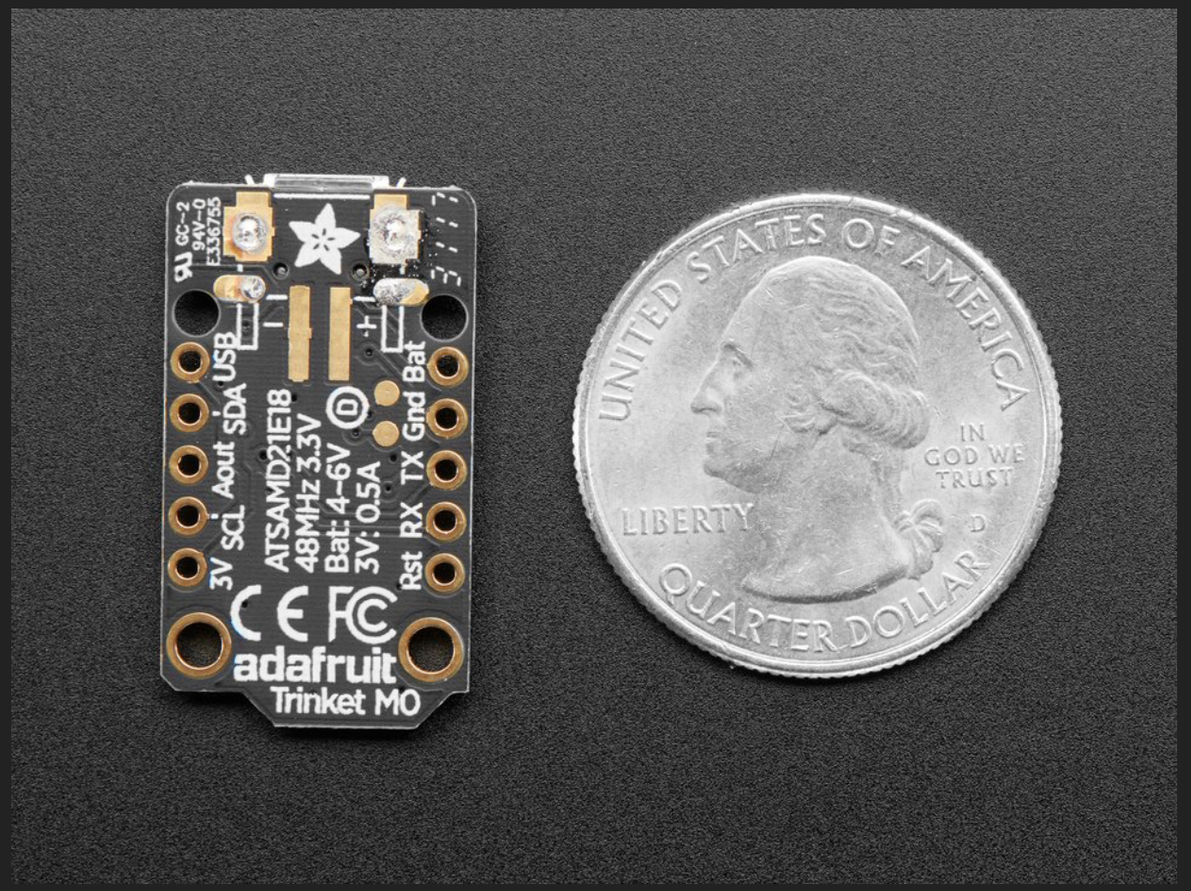
Pinning for TSOP382.., TSOP384..:

1 = OUT, 2 = GND, 3 = V_S

ADAFRUIT TRINKET M0

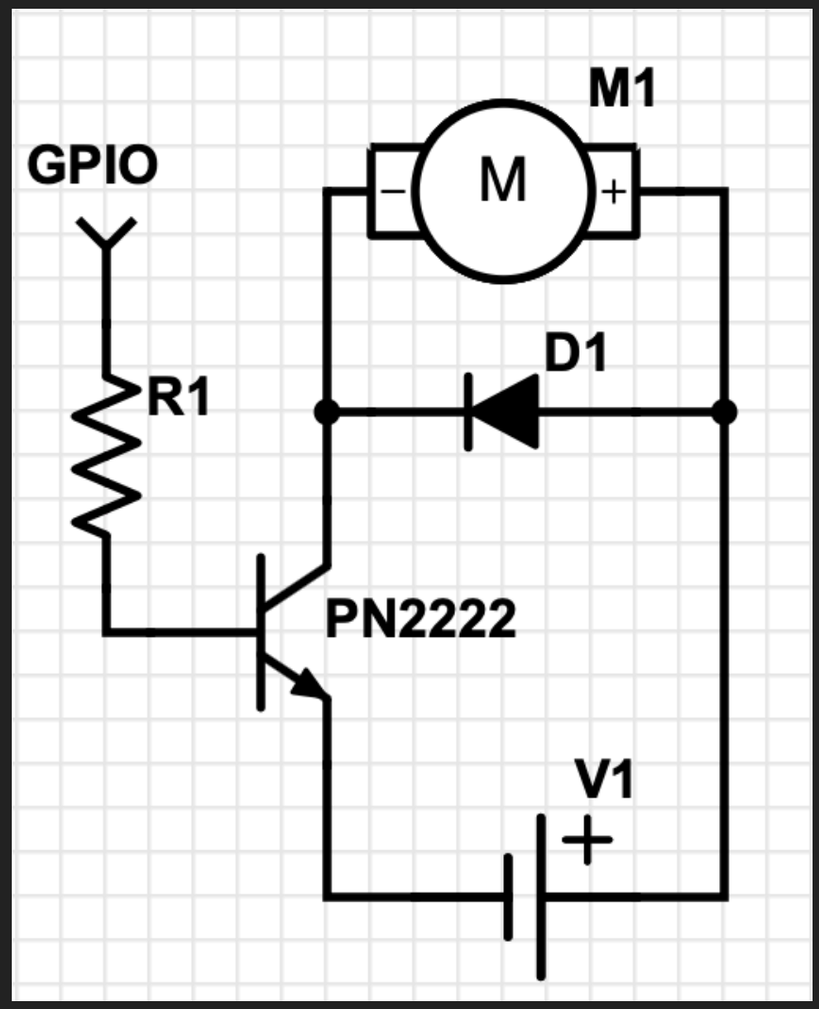
- Circuit Python
  - Easy to develop and debug quickly
  - Small size
- Ties all components together
- Parses IR data
- Varies voltage across motor





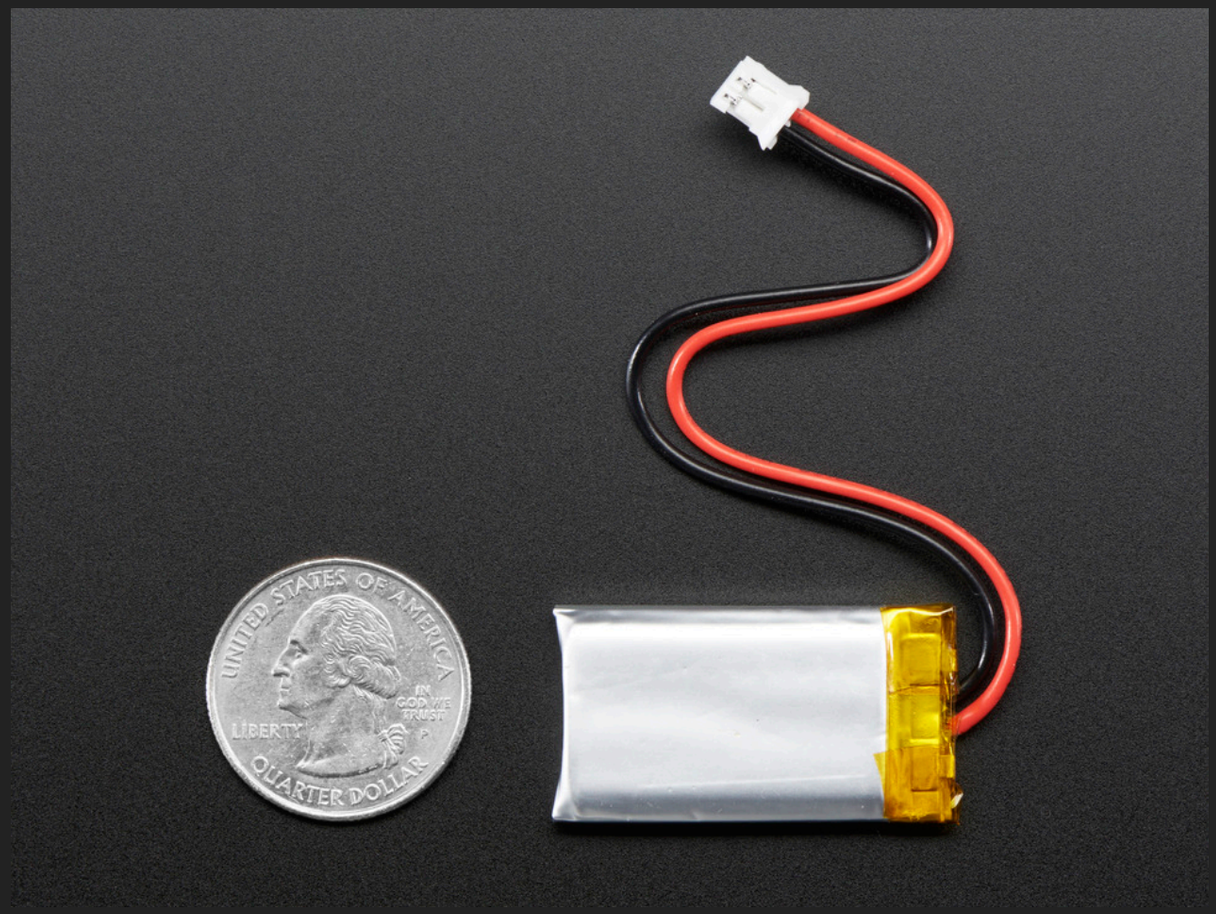
VIBRATIONAL MOTOR/PWM CIRCUIT/LED

- Three levels of intensity
- Not enough current from GPIO on trinket
- Use PWM circuit to vary voltage

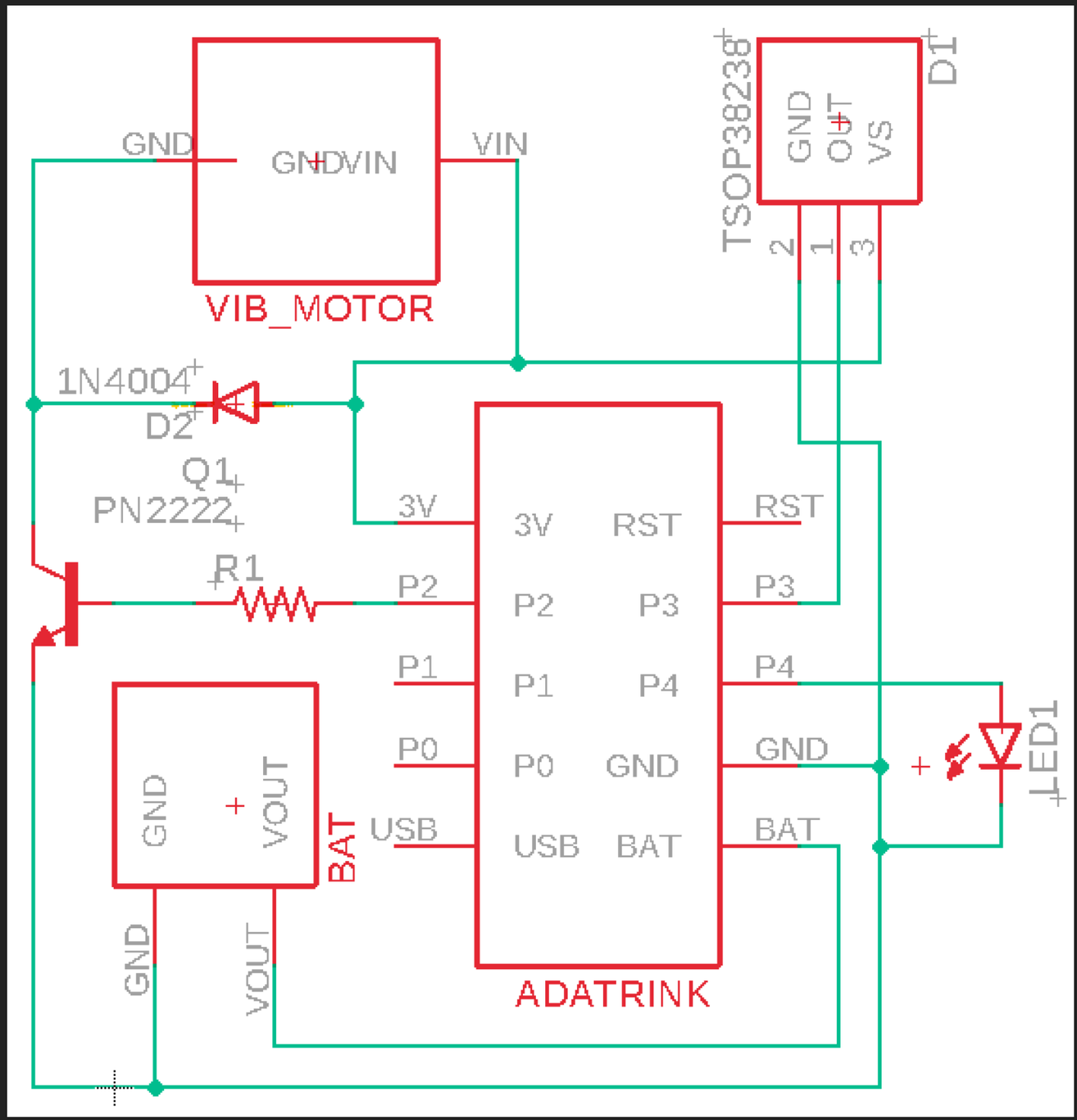


POWER SUPPLY

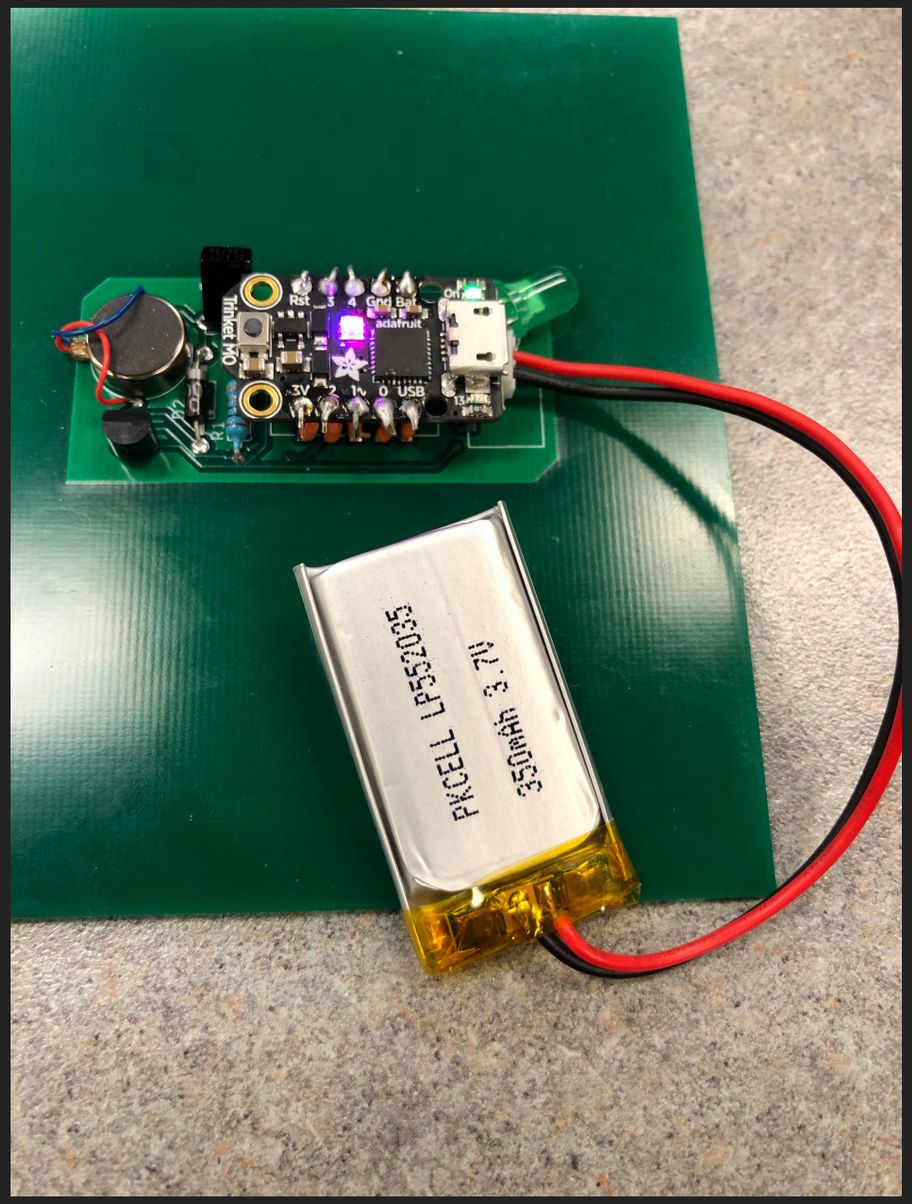
- 350mAh, 3.7V
  - Power bracelet for 4+ hours
- Chose for small size
  - Thinner than a quarter



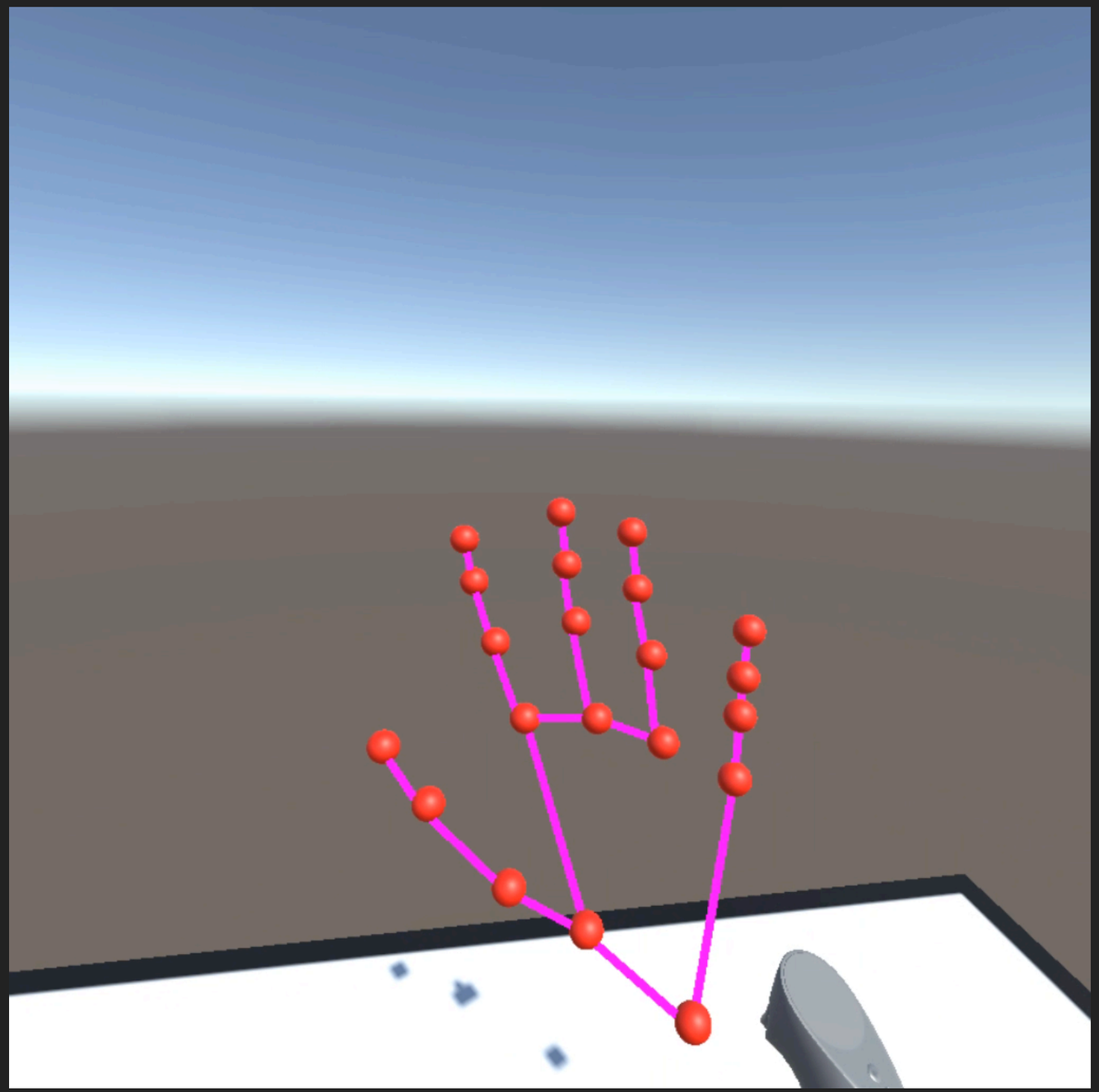
CIRCUIT SCHEMATIC



FINAL BRACELET



RESULTS



CHALLENGES

- TPU was incompatible with binaries
  - Some libraries were precompiled, no way to modify
- Hard to get hand data into Oculus Go
  - Can not send data over bluetooth or micro-usb

CHALLENGES (CONT.)

- Development for Oculus Go is not great
  - Poor documentation
  - Oculus forum is unhelpful
- Unable to use in Unity preview editor
  - Need to build the app to preview
- Restrictive
  - No open ports, can not send data except over network

FUTURE IMPROVEMENTS

- Haptic-feedback glove
  - Easy to expand bracelet to full glove
    - Just need more IR codes
- Reduce latency of network
  - Pipeline runs at ~15fps, but network is slow
  - Find new solution or reduce this latency
- Find a way to port software to raspberry pi

CONCLUSION

- Bracelet fully tested/verified
- Hand simulator works with specific hardware requirements, but not on raspberry pi
- Demo game works with some latency issues

THANK YOU

- Questions?