reihlo.com
github.com/reihlo

Gestur
by reihlo

Kyle Carson, Ryan Kaveh, Jon Young, Ryan Lee, Ryan Tsukamoto

Introduction & Vision

Virtual Reality (VR) promises to be the next step in human-machine interaction; from films to video games to enterprise, there is an immense potential for immersion. Current VR tools are limited to bulky setups with unwieldy controllers, none of which can accurately track fine motor movements in 3D spaces. The Gestur Glove aims to further user immersion in VR environments. Using an array of different sensors, Gestur can wirelessly track a user's fingers and hand movements, while simultaneously providing feedback based off of virtual interactions. Our glove is a prototype for the immersive, wireless and universal controllers we aim to build in the future.

System Overview

The PCB reads touch, flex, and motion tracking data from the glove, frames/packages the measurements, and sends them to the host computer for processing. When the user interacts with VR objects, the host computer sends haptic data to the glove.

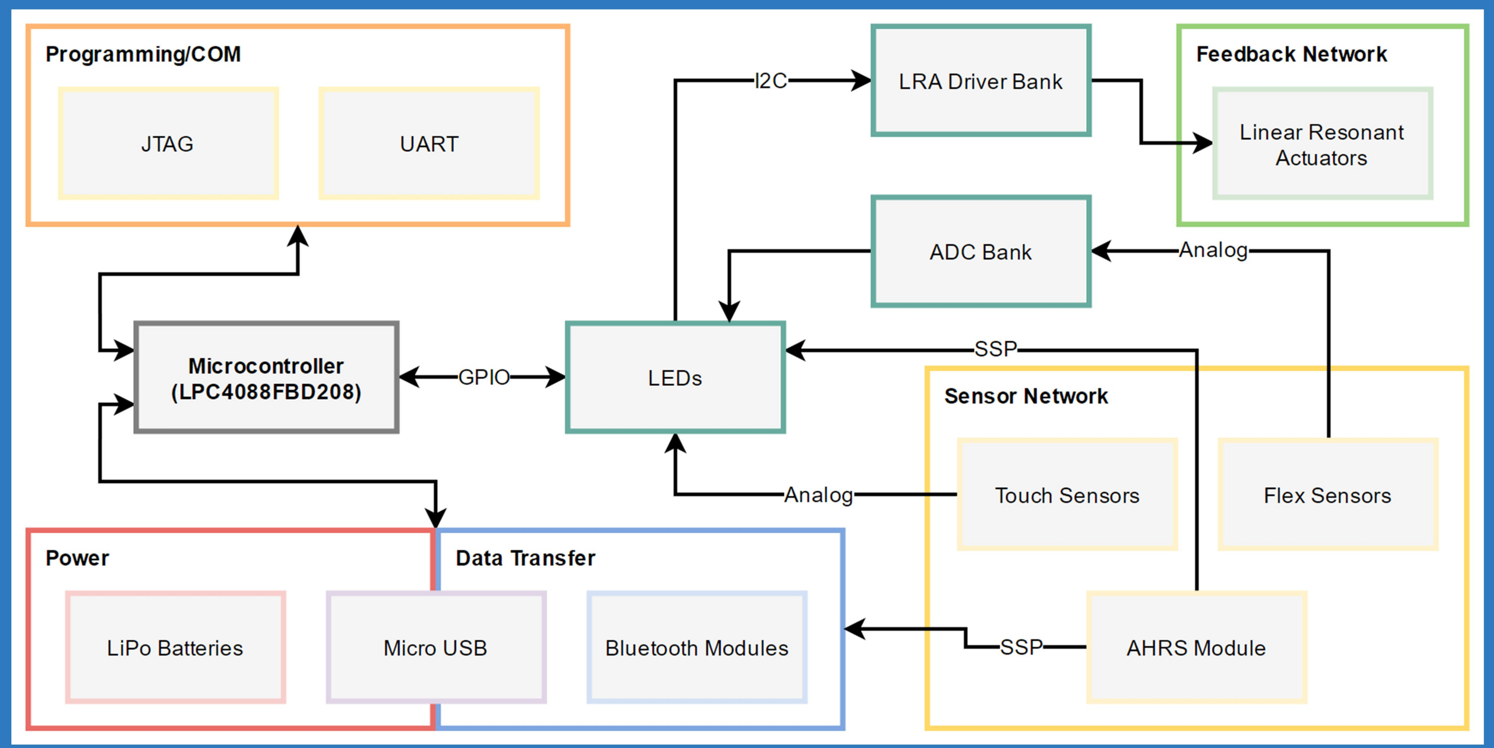

Figure 1: Gestur PCB block diagram

Iterative Development

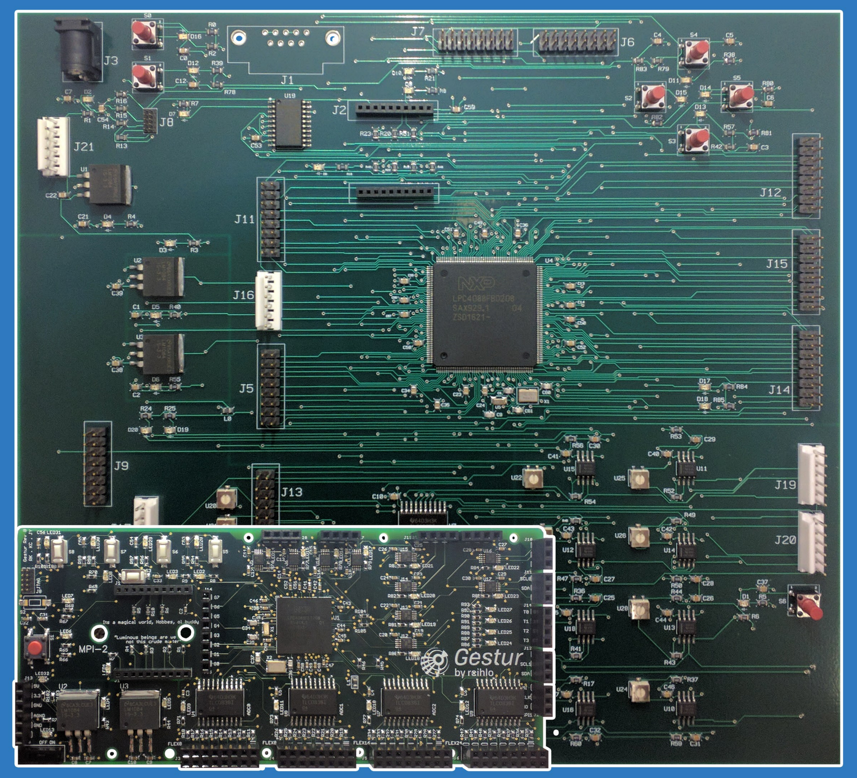

Figure 2: Gestur PCB Rev 1 and Rev 2 at relative scale

- First Gestur board was development kit for the embedded software design
- With Laritech's support and capabilities, the second board used BGA packages, new motor drivers for the LRAs, & smaller components
- Second spin is ~80% smaller than the first (from 8.6" x 7.9" to 5.5" x 2.5")
- Second board smaller than a conventional smartphone

Glove Design

- 20 flex sensors, 10 LRAs, 4 touch sensors and 1 IMU sewn into glove for strain relief
- Flex sensors are situated around all finger and hand joints, while motors target large muscle groups and areas with highest nerve ending densities
- IMU sits behind the palm to provide the smoothest tracking
- Challenges: glove longevity, component modularity, and cable management

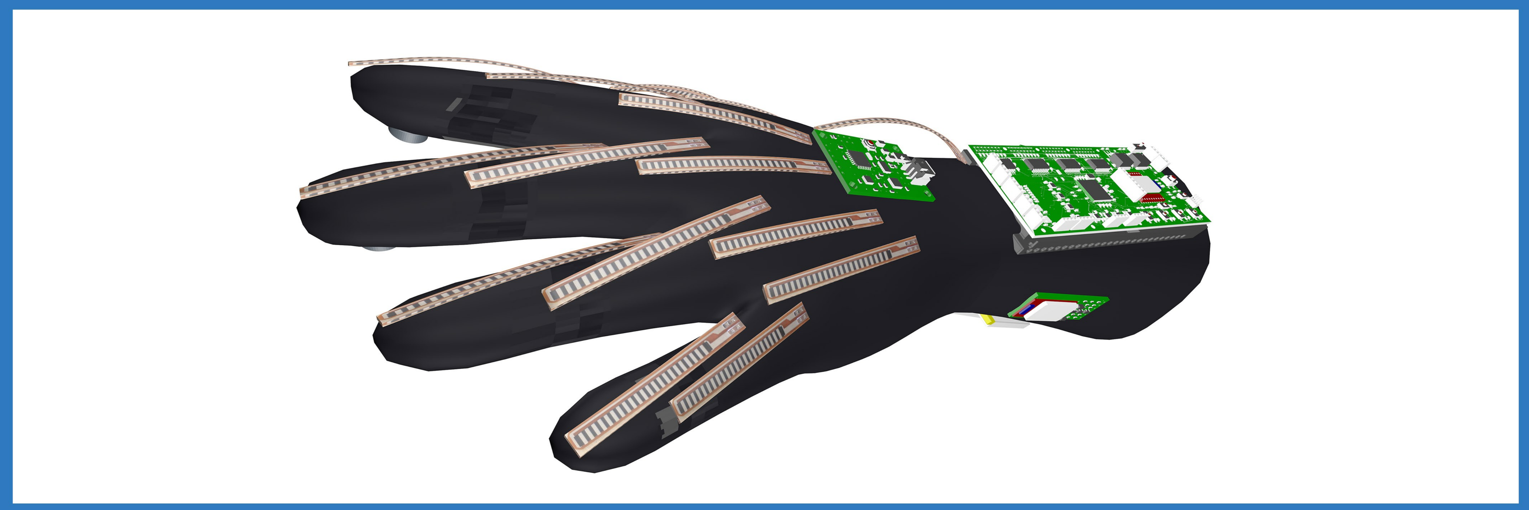

Figure 3: A 3D model of the glove's construction

Motion Tracking

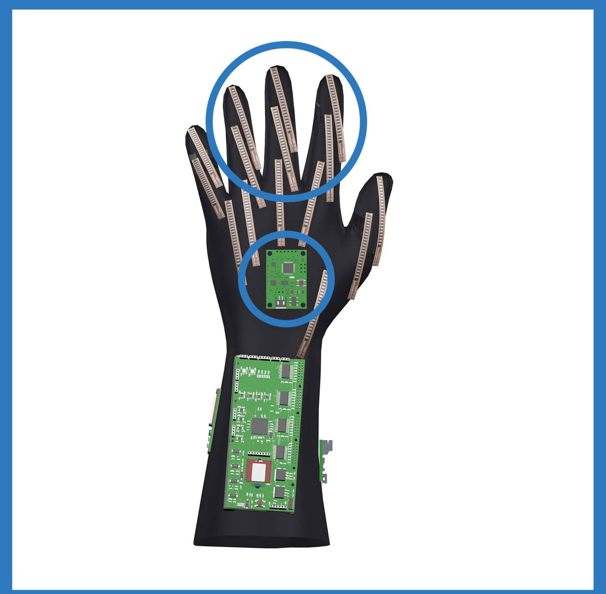

Figure 4: Flex sensors and IMU

- Flex sensors measure joints
- Extrapolate fine finger movement with LPF
- IMU tracks hand movement in 3D space
- Transmitted wirelessly via bluetooth to host computer
- Applications beyond VR: medical & design

Haptic Feedback

Figure 5: Haptic feedback motors (LRAs)

- Built with Individually controlled LRAs
- Targets large muscle groups & areas with highest densities of nerve ending
- Can provide varying responses to simulate different textures or degrees of pressure
- Triggered via bluetooth signal form host computer

Software Overview

The software is divided into two primary parts: the embedded software running on the board and the application software running on the host computer. The embedded code handles all of the sensing, data processing, and data-packing needed to transmit the glove's state. The application code provides a core API four separate components. Both parts interface using COM ports to asynchronously send and receive data via bluetooth.

Embedded

- FSM handles all I/O - State transitions depend on the input data from bluetooth
- Input transitions are interrupt-based; bluetooth transitions are cyclic/host dependent
- Output data is obtained via mixed polling and interrupting
- Goes into low power state when idle

Application

- Gestur Core API (GCORE) creates abstractions for all serial communication and representation of the glove's data
- Gestur Model Data Generator (GMDG) uses GCORE to emulate the glove in software, providing a WPF GUI interface
- Gestur API (GAPI) is built on GCORE to provide a higher-level API abstraction
- TouchBox is a demonstration game built using the Unity engine

By developing GMDG before the second revision of the board was finished, we were able to forward-develop the rest of the application software without a physical glove. We used com0com to establish virtual pairs of COM ports on the same computer, creating a pipeline with the GMDG simulated glove and any application software that would later be interacting with the physical glove.

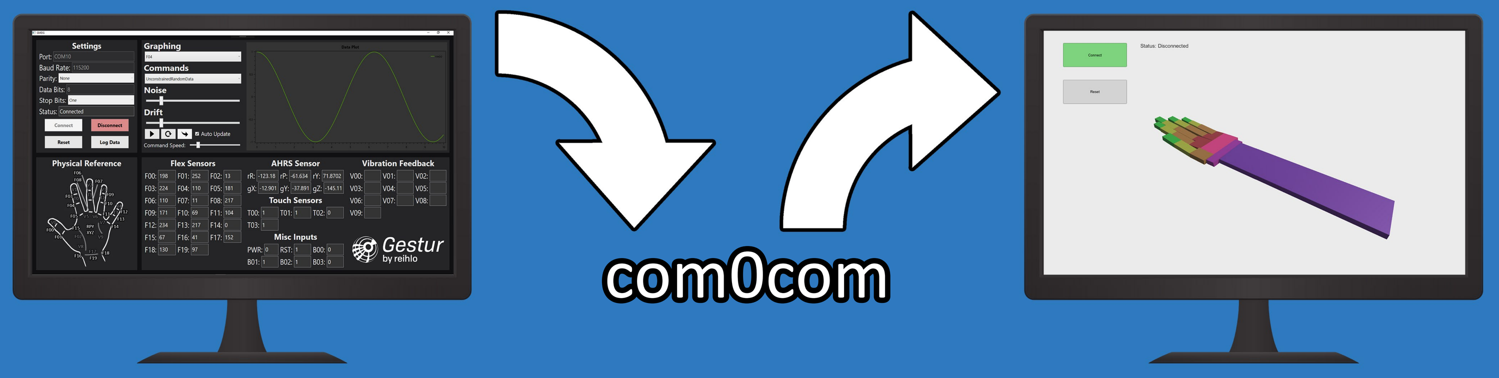

Figure 6: Forward-development pipeline out of GMDG, through com0com, and into TouchBox

Acknowledgments

Thank you Bill, Lilly, Kristin, Veronica, Eddie, and everyone else from Laritech for making all of this possible. Thank you Caio, Celeste, and Will for always helping us at a moment's notice. Thank you John Johnson for sticking around one last year to guide us with Gestur. Thank you Yoga Isukapalli for helping organize this year's Capstone events. Thank you Forrest Brewer and Yon Visell for your consultation and advice. Lastly, thank you to the College of Engineering for additional funding and support.

UC SANTA BARBARA engineering

ECE 189A/B: UCSB Computer Engineering Capstone (2016 - 2017)

LARITECH, INC. ODM SOLUTIONS