A Human-Machine Interface to Assess Tactile Perception in Individuals with Stroke

Jackson Todd
CS 4624 - Multimedia, Hypertext and Information Access
Professor: Dr. Edward Fox
Virginia Tech, Blacksburg, VA 24061
11 May 2023
Outline

1. Background
2. Plan
3. Results
4. Future work
5. Acknowledgements
The Client

- Dr. Netta Gurari, head of the Robotics and Sensorimotor Control Laboratory in the Department of Biomedical Engineering & Mechanics
  - The name of the lab is often abbreviated RoSenCo
- Researchers, primarily Anna Feldbush
Motivation

- RoSenCo endeavours to understand how people move after brain injury.
- Stroke is a leading cause of disability, and the asymmetric nature of strokes allows comparing the two sides of the brain to gain insight.
The Problem

Currently:
- Old software written in a mix of MatLab, C, and Python
- Bespoke hardware, difficult to make and use

Goal:
- Python only code-base, open and more well-known
- Off-the-shelf hardware
- Improved and extensible functionality
Plan for collaboration

- Not very relevant for a “team” of one
- Met every Monday morning with the clients
Deliverable(s)

- Packaged program that can be easily distributed
- Python code that meets the requirements
- Excellent documentation
System capabilities/requirements

1. Must control actuators & read sensors at 1000 Hz to 4000 Hz
2. Display live plots of data to the experimenter
3. Output audio/visual feedback for the participant
4. Save data for processing after experiment is completed
5. Must use Python
Physical setup

- Researcher
- Computer running RoSenCoExMan
- External system
- Brain scans
- "Picture taken" signal
- Participant in MRI
- Participant's hand
- Air compressor
- Pressure regulator
- Pneumatic actuator
- Audio prompts
- Pressure control signal
Begin experiment using: 'C:\Users\RoSenCoLab-1\Desktop\Reference_During_Experimentation_TEMP_01.csv' [Y/n] y
Connected to DAQ
Select a Procedure:
1) Anatomical Scan
2) Perceptual Threshold Testing(Arm 1)
3) Perceptual Threshold Testing(Arm 2)
4) Functional Localizer Scan(Arm 1)
5) Functional Localizer Scan(Arm 2)
6) Trial Scan(Arm 1)
7) Trial Scan(Arm 2)
Procedure: _
When you no longer feel pressure, press and release the button.
Text to speech

- Initially selected Mozilla TTS as the text-to-speech engine
- Produces fairly natural-sounding results
- Generation is done locally, so no internet required
- Uses a ~300MB model
- Dependency conflict with the live-plotting library
Text to speech cont.

- Pivoted to gTTS
- Wrapper for calls to Google Translate’s tts
- Sounds even better
- Much faster than Mozilla TTS
- Requires internet at time of generation, but results are cached
- No dependency conflict!
Performance

- The data input/output needs to run at 1000Hz to 4000Hz.
  - Currently running at 1600Hz.
- Simple reading and writing uses 8% of the period, leaving plenty of time for experiment logic.
- Currently, most “expensive” experimental procedure takes about 30%
- Time-sharing OSs have inherent non-determinism.
  - Not an issue because DAQ hardware reads samples exactly on time, and buffers them.
Future work

- Integration of GUI created by other student teams this semester.
- More advanced visuals for planned interactive experiments.
Acknowledgements

- Dr. Fox
- Dr. Gurari
- Anna Feldbush
- Grants
  - Department of Biomedical Engineering and Mechanics Start-up Funds, Virginia Tech
  - Institute for Critical Technology and Applied Science Junior Faculty Program, Virginia Tech
Image credits

[1] https://github.com/mozilla/TTS