Percy - Neptune Microfluidics Project

Matthew Brawley, Vrushali Mahajan, Michael Webster

Project Description:

The Neptune Microfluidics project involved creating a fluid-functional microfluidic device, creating a software tool to utilize a user-entered script to control an experiment on a microfluidic device using asynchronous timing as per the user's description, and also research further primitives that could be used to create, advance, or supplement microfluidic devices.

Microfluidic Devices:

The first goal of the project was to create a fluid functional microfluidic device which is based in a biological design. We have multiple microfluidics fabricated:

- Device 1 was created with 3DμF and has anomalies in the mixers and the control layer was not flipped correctly to properly control the device (3DμF bug).
- Device 2 was created with 3DμF and is fluid functional, with a control layer flipped manually.
- Device 3 was described with MINT and translated with Fluigi but is not fully fluid functional due to the placement of parts. All layers output correctly, however.

Ideas for future primitives:

- Multilevel devices that can be CNC milled.
- A device to assist with microfluidic initialization by preventing bubbles from traveling into the device - perhaps modified from a cell trap primitive.
- Improvements to cell visualization with designs that do not trap cells and using smaller bits on the CNC to etch grids for volume identification.

Software Tool Percy:

Please view the README file on our GitHub for detailed information on how to use Percy.

The software tool Percy consists of the following software elements:

- index.html - First webpage to open to run Percy. Requires Bootstrap and jQuery, these are linked to web hosts though so they don't require installation.

- scheduler.html - This page is linked to from index.html. It includes a text area for the user to enter a script and then automates the entire schedule for the script when the user hits the run button.

- parser.js - This is a javascript file called from scheduler.html when the user runs the script. This file reads in the user input commands (view readme) and parses them. It checks for syntax errors and alerts the user of syntax errors. The parser.js eventually uses a setTimeOut() function to call either sendDispense (in pumpDriver.js) or valveControl at the time specified by the user. This setTimeOut function is asynchronous in nature. The code stores each dispenser command into a JS object, places it in an

array and compares it to previously read in dispenser commands also stored as JS objects. This maintains the correct dispenser status.

- pumpDriver.js - Contains function sendDispense(). This script is called from the parser file. It calls functions in pwmCal.js and maintains their returned values to determine what commands to send to the the Arduino to control the servo motors which run the dispensers. Eventually, pumpDriver calls sendCommandDispense() in the DispenserCommands.js file.

- pwmCal.js - Previously developed by CIDAR lab, calculates the PWM values and the displacement of the dispensers. These are the physical commands that are passed to the Arduino board.

- sendCommandDispense.js - This script is called from pumpDriver.js and sends commands to the Arduino to control servo motors. Currently does not actually send the commands to the Arduino due to the Arduino not functioning with Neptune's current system. Instead it logs the command on the webpage.

- valveControl.js - Similar to the files and functions which send dispenser commands, this file contains the functions which send the commands for valves opening and closing to the Arduino. This function also calls pwmCal.js to calculate values. Currently does not actually send the commands to the Arduino due to the Arduino not functioning with Neptune's current system. Instead it logs the command on the webpage.

- toastr.js - A javascript library for asynchronous notifications. This script was a part of the software developed by CIDAR lab.

To compile the code, simply download it from the Github repository and ensure you have an internet connection to connect to web hosted jQuery and Bootstrap.