Multi-Location Droplet Management for Digital Microfluidics

Sarang Joshi

2017-2018

Contents

1 Abstract

The digital control and management of fluid droplets, or digital microfluidics, has applications in a wide array of fields, from chemical synthesis to diagnostics to analytic biological procedures. The vast majority of existing work in this field, however, is limited to the manipulation and control of single-electrode droplets. As the applications of this technology grow, the necessity for a more flexible system for control of larger droplets is becoming more and more evident. In this paper, I describe a proof-of-concept for an end-to-end system to add support for managing multi-location droplets to the University of Washington (UW) Molecular Information Systems Laboratory (MISL) PurpleDrop/Puddle microfluidics control system.

2 Introduction

Digital microfluidics (DMF) is an interdisciplinary area of work that sits at the intersection of computer engineering, physics, chemistry, and biotechnology. UW MISL has the Puddle and PurpleDrop projects that delve deep into the field of DMF. PurpleDrop is the UW MISL's microfluidic device project, based on the OpenDrop system, but tailored for MISL's specific needs. The PurpleDrop cartridge design includes a grid of electrodes that can be controlled by a microcontroller (such as an Arduino or a Raspberry Pi). This system is actively under development say more?. The other piece of the puzzle is Puddle, which is a microfluidics operating system that provides fluid-based primitives that allow a user to control a DMF device without having to deal with the low-level details.

The goal with this project is to expand the end-to-end system with support for droplets that can occupy more than a single grid cell on the DMF devices, which can allow for more flexible operation of DMF algorithms and applications. The project spanned from the top-level software implementation of Puddle to implementing and experimenting with multi-location droplet patterns at the PurpleDrop level.

3 Software

The first part of the end-to-end implementation was putting into place a workable software model for managing and controlling multi-location droplets.

The Puddle system underwent some major upgrades through the course of the project, starting from a pure Python core to a Rust core combined with a Python user-level API. The architecutre considers the PurpleDrop cartridge as a Cartesian grid, with every location on the board represented by an (x, y) tuple.

3.1 Support for Multiple Locations

The primary goal of the requisite work in the Puddle system was adding support for multi-location droplets. The single-element `location` field that represented the droplet's location on the board needed to be accompanied by a `shape`, which is a set of "offset" tuples. An "offset" is defined as a tuple representing the offset of every location that this droplet occupies from its base location. For example, if a droplet occupies {`(1, 1), (2, 1), (1, 2)`}, the droplet could be represented as:

```
location: (1, 1),
shape: \{(0, 0), (1, 0), (0, 1)\}}
```

However, simply adding a shape field to the Droplet class was not sufficient to support multilocation droplets. One critical aspect of multilocation droplets was verifying whether the locations that the droplet occupies are contiguous — without this invariant, a multilocation droplet cannot be considered a single droplet. I went through a primitive version of implementing this:

```
for each offset o in droplet's shape:
    if no other offset o' has a manhattan distance == 1 from
    o, the shape is not contiguous
```

In the end, I used a graph-based approach, using the Python NetworkX library, which was already being used for the routing algorithm. With this approach, every cell in the electrode grid is considered a node in doubly-connected graph. Thus if the graph represented by the shape of the droplet is connected, then that means that the shape is contiguous.

As the core system transitioned from Puddle to Rust, the multi-location support was simplified to more accurately reflect a workable model, where the shape of the droplet was restricted to being rectangular instead of a fully flexible (contiguous) set of offsets. A square droplet shape, or a rectangular shapes that is square-like (such as 2x1, 3x1, 3x2), is one of the more physically stable shapes. Thus the droplet was now represented as a base location and a pair of dimensions representing the width and height of the droplet. This simplification did away with the need for any contiguity checking.

4 Hardware

With the basic software support in place in the Puddle software stack, the next step would be to implement support for these operations on the PurpleDrop cartridge. While the PurpleDrop software layer hopes to abstract away the device specifities, there has to be some acknowledgement of the different specifications that are mandated by the actual physical DMF device.

One important simplification that is important to note here is only considering relatively smaller-scale droplets, i.e. nothing more than 3x2 droplets. Once the size of the droplet crosses that magnitude, the magnitude of the intermolecular forces supersedes that of the electrowetting, and any sort of operations on the droplets cannot be consistently observed or analyzed.

The main operations that the Puddle system supported were `mix`, `move`, and `split`. The implementation of the first two of these operations did not differ significantly from the single-location droplet implementation that was already in place. For **moving**, in order to move a droplet, the pattern is still involves activating the adjacent electrode and deactivating the location that the droplet is no longer occupying. For **mixing**, the pattern still involved moving the two droplets together and moving them in a loop to ensure that the contents of the droplets are evenly mixed.

However, the operation of **splitting** brings a new level of complexity that is not experienced with single-location droplets. The key difference here is that the pattern of splitting is no longer the same for all multi-location droplets. Every different shape has its own splitting pattern that results in the best 50-50 split, entirely dependent on the shape. On top of this, the splitting mechanism is also more dependent on physical variables that affect the PurpleDrop cartridge, which need to be abstracted away in Puddle.

So the next stage of the project involved experimenting with the feasibility of splitting multi-location droplets, with a specific focus on the effect that various variables have on the process. The primary droplet under experimentation was the 2x1 droplet, as it can be thought of as the natural successor of the simple 1x1 droplet. The 3x1 and 2x2 droplets also were experimented with in a cursory fashion.

4.1 Approach and Methods

The primary physical variables that I tested were the following:

- Splitting pattern, i.e. the exact electrode actuation patterns requisite for splitting
- Volume per grid cell (for a 2x1 droplet, the overall volume of the droplet would be twice this value, since the droplet occupies 2 grid cells)
- Spacing between base plate and top layer
- Viscosity of oil in which the droplets are suspended

Each experiment constituted a certain permutation of these variables, and each experiment was run 10 times, to ensure statistical accuracy. For each experiment, a collection of the needed number of 1x1 droplets were added at specific locations on the board, and then were combine them to create the larger droplet. The results were measured by the ratio of the split droplets, and the amount of residual liquid that was left outside of the final 2 droplets created by the split operation.

The experimentation code was implemented using the PurpleDrop Python control repository, which connects to the on-board Arduino microcontroller over a Serial port. The PurpleDrop control source code has implemented a basic architecture that allows the program to activate and deactive specific electrodes and send the corresponding commands to the board. The Arduino ran the `PurpleDropClient` file, which essentially listens for Serial input, and once received, converts the electrode map from the Python frontend to the corresponding pins to activate.

After a sequence of trial-and-error experiments, the baseline experiment was determined to constitute the following parameter values: **1.75 μL, 1-layer spacing, 196V top plate voltage, 1.5 cSt oil suspension viscosity**.

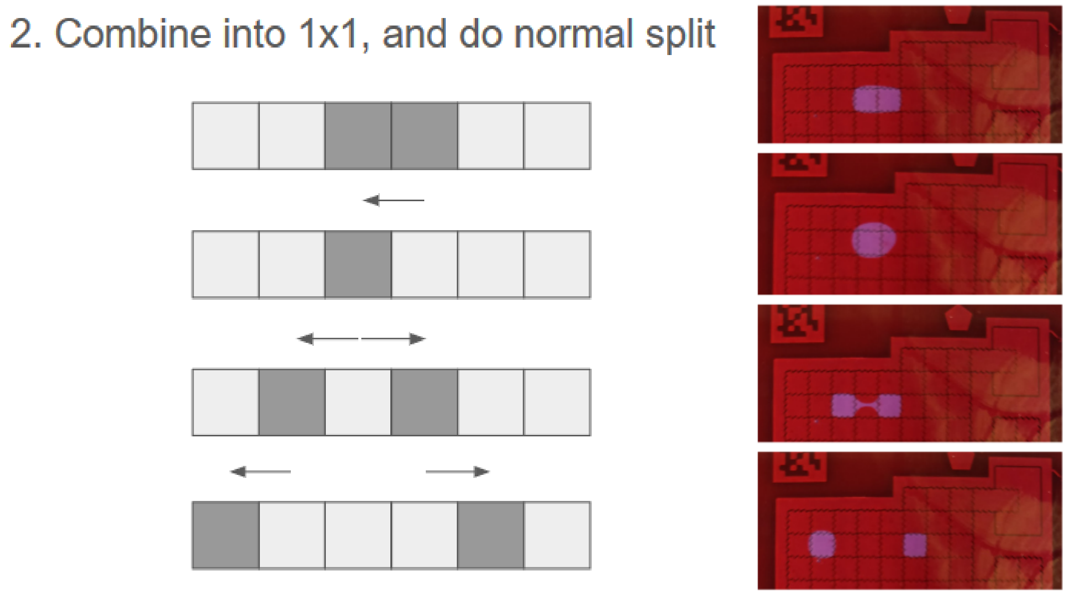
4.2 Experimentation

4.2.1 2x1 Droplet Experimentation

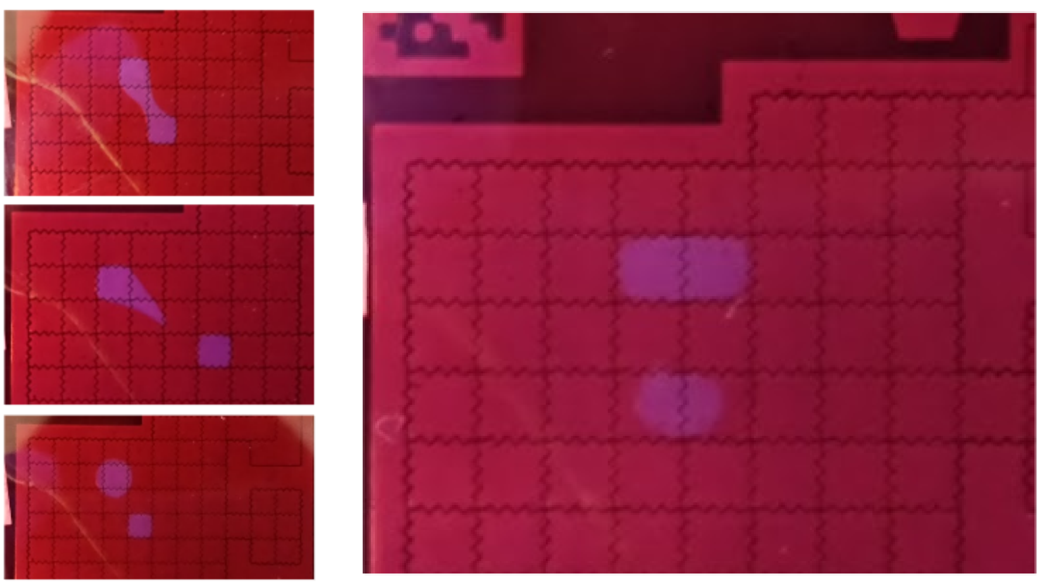
The first set of experiments were run with varying split patterns for the 2x1 droplet. *Pattern 1* was the most basic split, as described below:

The second half above describes the process of residue collection. Depending on the specific experimental setup, there might be some amount of the original liquid remaining in the original droplet location. Thus, in order to retain control over the residue and not lose fluid, the split pattern takes this into account and collects the residue.

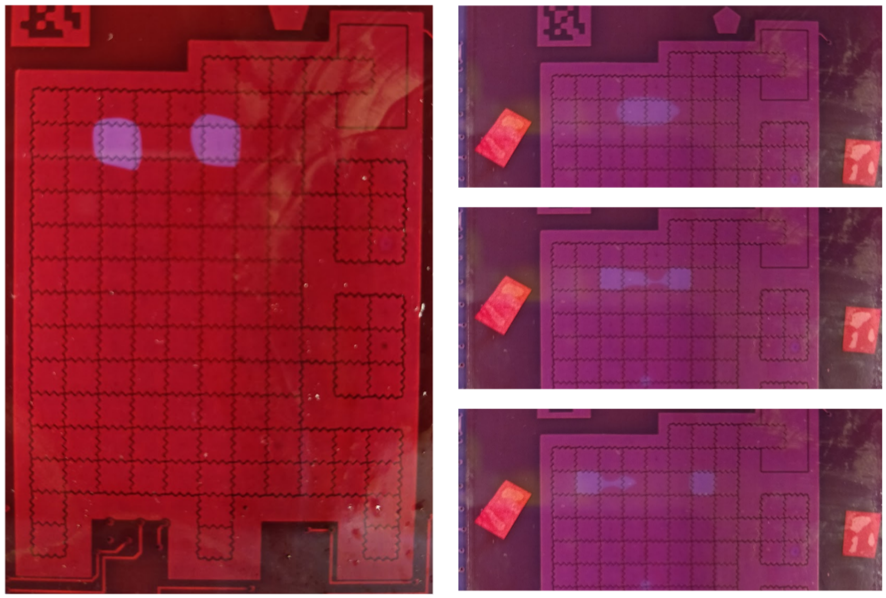
The second pattern that was explored was *Pattern 2*, which combines the 2x1 droplet into a 1x1 location, and performs the existing 1x1 split method.



Other generally less successful patterns include *Pattern 3* (left), i.e. moving one half of the droplet `[Up, Left]` and the other `[Down, Right]`, and *Pattern 4* (right), i.e. splitting the 2x1 lengthwise as opposed to heightwise as in *Pattern 1*.



For 2x1 droplets, the remaining experiments explored varying the volume, top-bottom plate spacing, and oil viscosity. Below are examples of the visual differences associated with larger volume, and higher oil viscosity.



4.2.2 Larger Droplet Experimentation

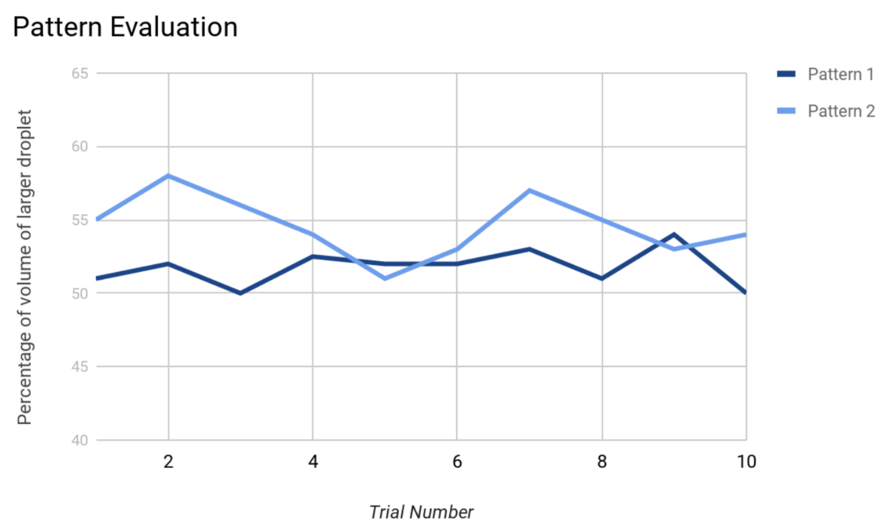
Beyond the simple 2x1 droplet, I also ran some primitive experiments with 3x1 and 2x2 droplets. These experiments were not as exhaustive as the 2x1 permutation testing, but I was still able to observe results, as seen here:

3x1 split, 2x2 split.

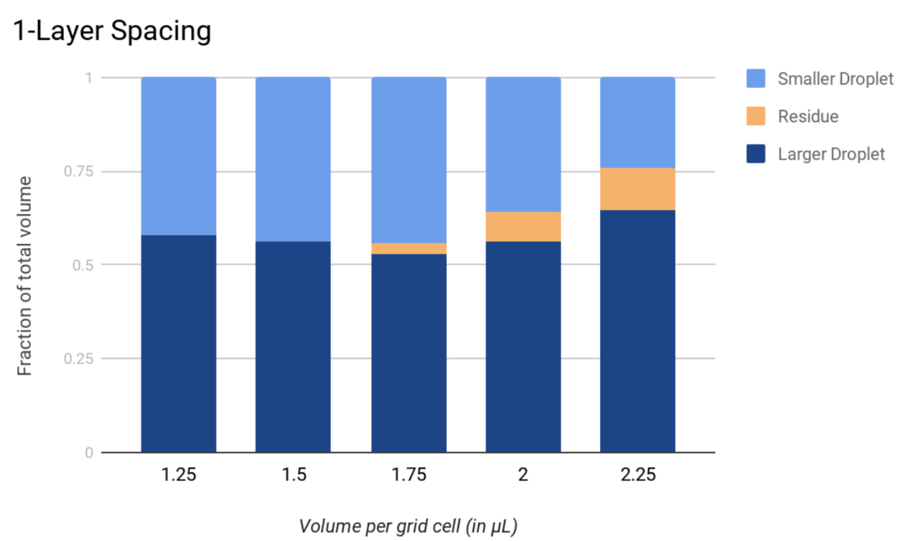
4.3 Results

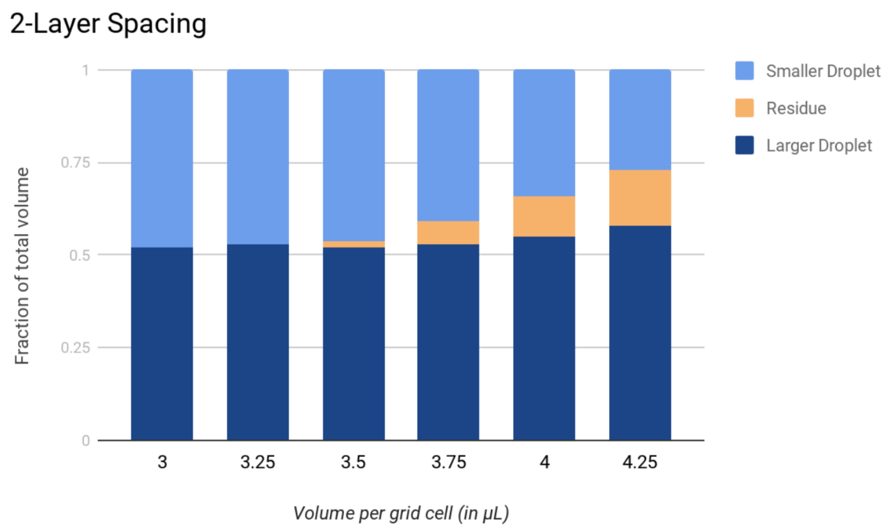
The results from these experiments were meticulously collected and tabulated. The process of evaluating the actual split ratio was completed using visual analysis. One possible approach that was considered to measure the resulting droplet volumes was manually re-pipetting the split droplets and measuring with maximum precision, but the time taken per experiment for this procedure was sufficient impetus to discard this approach. Instead, each experiment featured a fixed camera mounted directly above the device, facing it from a top-down view. Appropriate lighting was necessary to ensure consistent lighting for all the pictures taken by the camera. Pictures were taken manually at the start and end of every experiment trial. Once the images were collected, the open-source image processing program GIMP was used to compare the size of the droplet as seen in the images. This was also used to measure the residue, if any.

4.3.1 Pattern Results

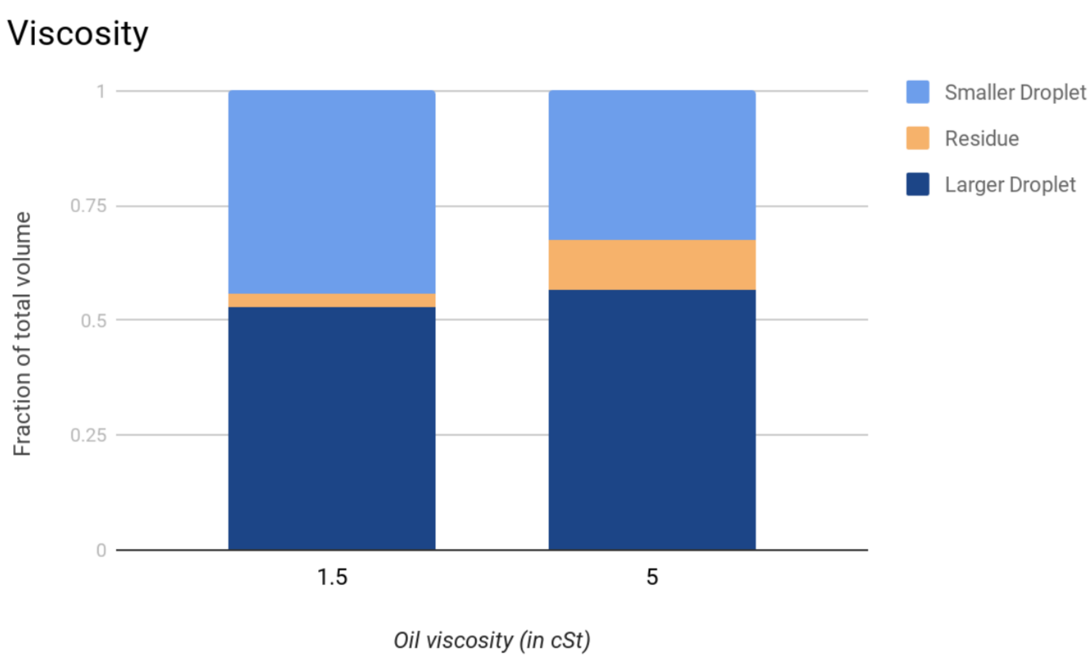


4.3.2 Volume-Spacing Results

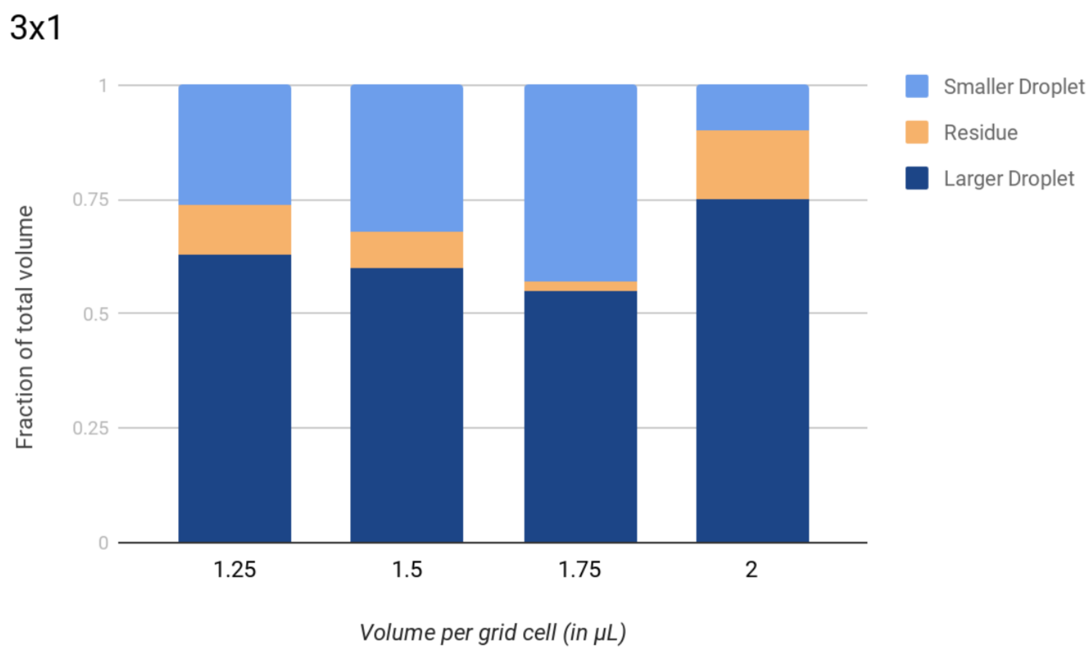




4.3.3 Viscosity Results



4.3.4 3x1 Results



4.4 Result Analysis

The above graphs provided me with some baseline data upon which to analyze the efficacy of splitting. While the ratio of the split droplets themselves is important to analyze, what is almost more valuable is analyzing the residue left behind by the split operation. As mentioned earlier, the amount of residue left behind correlated with the variables controlled as part of the experimentation. The presence of residue is also a clear indicator of the possibility of splitting in that particular permutation of values: this means that the electrowetting force that should be able to pull a droplet apart is not strong enough to do so, and some amount of liquid is left behind.

4.4.1 Pattern Analysis

It is evident from the results seen above that *Pattern 1* is consistently the most accurate split. In the pattern comparison, *Pattern 1* had an average 51.75%-48.25% split, while *Pattern 2* had an average 54.60%-45.40% split. The key reason for this was the fact that during the 2x1 to 1x1 combination, it was not always possible for the whole volume of the droplet to be contained in the grid cell. As a result, when the droplet was pulled apart in the split step, one side would regularly have an uneven portion of the original droplet volume.

Patterns 3 and *4* were too erratic to be considered valid in the environment constraints as defined by the parameters of the experimentation, and thus were not explored further.

4.4.2 Volume and Spacing

It is evident from the results seen above that for a given spacing, there is a particular value for the droplet volume that seems to give the best results. On the one hand, smaller droplet sizes lead to inaccurate splits, because the size of the to-be-split droplets are not large enough to be controlled by the electrowetting force. On the other hand, larger droplet sizes lead to a greater frequency of residue buildup due to the reasons mentioned above. In this case, for 1-layer spacing, 1.75 μL per grid cell seemed ideal, and for 2-layer spacing, 3 - 3.5 μL per grid cell seemed ideal.

As for spacing, 1-layer spacing fared much better in getting a cleaner split than 2-layer spacing, which can be attributed to the effect that spacing has on the contact angle of the droplet with the hydrophobic surface (the base plate). Because the top and bottom plates are closer together, the contact angle is more harsh, resulting in a weakening of intermolecular forces, allowing for the electrowetting force to take over.

The far-reaching impact of this analysis is the notion that there is an ideal volume-to-spacing ratio that represents the ideal area that a droplet occupies on the board, if considered from a top-down 2D perspective. According to the above experimentation, this sweet spot seems to be roughly in the 1.5 to 1.75 μL range.

4.4.3 Oil Viscosity

As mentioned in the introduction, the primary function of the oil in the DMF device is to provide a sort of lubricant for the water droplets thanks to the immiscibility of the droplets and the oil. However, while oil does provide this assistance to the control and ease of movement of the droplets, the viscosity of the oil also negatively impacts how fast the droplets can actually move around. The above experiments demonstrated that, critically, the higher viscosity oil resulted in almost 10% more residue, and less even droplet splitting. The most problematic situation with the higher viscosity oil substrate was the fact that this slowed down the system greatly. One key aspect of the abstract nature of the Puddle software stack is to remove the dependency on timing, and thus higher oil viscosities cannot be considered feasible for large droplet splitting.

5 Conclusion and Future Work

All in all, the physical variables affecting multi-location droplets were accurately assessed, and provide a clean picture of the ideal setup for larger droplet manipulation and splitting. The brand new result that is a valuable takeaway as mentioned earlier is the notion of an ideal volume-to-spacing ratio which could definitely serve as a starting point for future analysis.

As for future work, I hope that the two (somewhat) disparate pieces of controlling multilocation droplets can be connected together, to give truly end-to-end control to the Puddle high-level software user, even with fine-grained flexibility over the intended patterns for splitting. In terms of the actual chemical in the droplets that are being manipulated, some future work for other chemicals suspended in water would be also beneficial, considering that most biological experiments feature 80-20 water-chemical mixtures. Since this has already been tested by the UW MISL lab, it would be easily possible to extend this testing to larger droplets. Some other variables that were out of control for this experimentation phase would also be beneficial to test, such as the maximum droplet size, different droplet fluids, temperature, oil-free movement, and more.

At the end of the day, I hope that this work contributes in some small way to the bigger movement from silicon-only computer systems to combined silicon/biological systems, which allow for the easy interpolation between what once seemed to be incompatible domains.

6 Referenced Work

Some referenced work on the foundations and applications of DMF can be read in [1], [2], [3]. More details of UW MISL's work can be found at the lab website for Microfluidic Automation [4] and the Puddle software repository

[5].

7 Acknowledgements

I would like to extend my gratitude to my faculty advisor Luis Ceze for his guidance and help in finding a research project, my project lead and manager Max Willsey for his start-to-finish assistance with work and endless patience to answer my questions, and Ashley Stephenson for her help with managing and controlling the PurpleDrop cartridge and familiarizing me with independent operation of the board.

References

[1] Sung Kwon Cho, Hyejin Moon, and Chang-Jin Kim. Creating, transporting, cutting, and merging liquid droplets by electrowetting-based actuation for digital microfluidic circuits. *Journal of Microelectromechanical Systems*, 12(1):70–80, Feb 2003.

[2] Kihwan Choi, Alphonsus H.C. Ng, Ryan Fobel, and Aaron R. Wheeler. Digital microfluidics. *Annual Review of Analytical Chemistry*, 5(1):413–440, 2012. PMID: 22524226.

[3] Richard B. Fair. Digital microfluidics: is a true lab-on-a-chip possible? 2007.

[4] UW MISL. Misl: Microfluidic automation, 2018.

[5] UW MISL. uwmisl/puddle: A system for high-level, reliable programming of microfluidic devices, 2018.