Enabling Practical and Accessible Automatic Droplet Microfluidics Platforms

by

Kevin Haiqiao Chen

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2023

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

Noura Ezzo contributed comments on hydrogel spheroid encapsulation in Chapter 7, and supported with PDMS chip fabrication.

Abstract

Droplet microfluidics has emerged as an innovative technology enabling high-sensitivity, high-resolution chemical and biological analyses via precise manipulation of picoliter-to-microliter fluid droplets. The ideal end goal of this technology is a general-purpose droplet microfluidics platform (DMP) composed of simple building blocks or modules that can be configured to perform arbitrary manipulations and analyses of individual droplets autonomously, returning desired outputs (e.g. nanoparticle synthesis) or insights (e.g. heavy metal detection) to the end-user.

Although numerous innovations in droplet manipulation have emerged in the literature, most existing techniques — broadly categorized as passive vs active — optimize for a single application and act on continuous droplet trains, hence are difficult to generalize to arbitrary manipulation of individual droplets. Passive techniques rely on specific microfluidic chip geometries to be designed by a skilled user to perform a fixed sequence of droplet manipulations, and thus cannot be used for individual droplet control. Most active techniques embed custom actuators (electrodes, membranes, etc) within a passive system which only allows individual droplet control in localized areas, limiting the precision and resolution of droplet manipulations. A simpler and more generic technique is pressure-driven feedback control, in which droplets are sensed visually within simple passive chip geometries (e.g. T-junctions) and actuated by off-chip pumps that adjust chip inlet pressures in response to visual feedback. This approach shows that individual droplets can be stabilized and driven to arbitrary locations on-chip without the need for complex chip designs or embedded actuators, opening the door to modular automation.

However, bridging the gap from this proof-of-concept to a fully automated modular platform for non-expert users requires overcoming significant practicality and accessibility challenges. Existing feedback control systems for droplet manipulation ignore time-varying behavior in the system, which gradually degrades performance and reliability, necessitating frequent manual tuning and calibration. Additionally, current software workflows require the end-user to manually set up each droplet manipulation, which does not generalize to longer manipulation sequences necessary for practical applications. Moreover, standard pressure-driven flow generation methods are either too slow and imprecise for individual droplet control, or too complex and costly to be effectively modularized.

This thesis aims to address these key challenges in feedback control, software workflow, and droplet actuation to pave the way for modular automated DMPs that are practical and accessible for end-users. On feedback control, a new adaptive control system is designed to automatically perform model parameter identification online, compensating for changes in

system dynamics as droplet manipulations are performed (Chapter 4). Regarding software, a new DMP workflow is developed to allow end users to validate and execute arbitrary manipulation sequences automatically (Chapter 5). For pressure-driven flow generation, off-the-shelf piezoelectric micropumps are evaluated as a modular, low-cost alternative to existing methods, demonstrating comparable performance in droplet manipulation (Chapter 6).

Acknowledgements

I would like to express my deepest thanks to my supervisor Prof. Carolyn Ren for her patience, guidance, and belief in my abilities, pushing me to see the forest beyond the trees. I am also grateful to Prof. Baris Fidan for taking the time to join my seminar committee, and for his insightful and intuitive approach to controls, a source of inspiration through my foray into this field.

I also appreciate Marie Hebert for her initial guidance through the existing platform.

Additionally, I extend my thanks to Tomasz Zablotny, Matt Courtney, Runze Gao, Dylan Hahn, and Noura Ezzo for the stimulating discussions and ideas that have influenced my research.

Dedication

To my Mom and Dad, for their love, encouragement, and support all this time.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The need to perform high sensitivity, high resolution chemical and biological analyses on small sample quantities led to the development of microfluidics, the science and technology for manipulating microliter to picoliter fluids in microscale channels [11].

The advantages of shrinking to micron scale include dramatic reductions in sample volume, cost, and analysis time. As dimensions shrink, the dominance of surface forces (e.g. surface tension, viscosity) over volume forces (e.g. gravity, inertia) leads to changes in fluid behavior — importantly the prevalence of laminar flow — that are advantageous in many circumstances [11].

The extension from single-phase to multi-phase flows consisting of at least two immiscible fluids (e.g. water and oil) allowed monodisperse microfluidic droplets to be generated in a controlled manner [12]. This spurred the field of droplet microfluidics, finding applications in chemistry and biology including the production of polymer particles, emulsions, and foams, as well as the isolation and confinement of cells and chemical reactions [13].

To facilitate these applications, various techniques have been devised for precise control of the desired droplet manipulations (droplet generation, split, merge, sort, mix, etc) [13]. The challenge lies in precisely generating the necessary flow field for each manipulation in real-time. Passive techniques generate a constant pressure-driven flow field to perform a fixed sequence of droplet manipulations according to microchannel geometry [14], meaning the performance of each manipulation depends on physical factors (e.g. fluid properties, manufacturing tolerances, microchannel surface wettability [15]), and individual manipulations cannot be performed on-demand. This necessitates the design, fabrication, and

testing of custom application-specific microchannel architectures, making passive techniques inaccessible to users without microfluidics expertise. Active techniques on the other hand add external energy (e.g. electrical, mechanical, thermal) to modify the flow field locally, facilitating on-demand manipulations [16]. However, this often requires custom actuators (electrodes, membranes, lasers, etc) and sensors to be manufactured and embedded into the device [16], making it expensive and impractical for real-world use. Most active techniques are also limited in the precision and types of manipulations they can control [17], again requiring microfluidic expertise to build custom devices for each use-case. For the potential of droplet microfluidics to be fully realized beyond the research lab, there is a need to simplify these techniques to make them more practical and accessible for non-expert users. Ultimately, a general purpose droplet microfluidics platform (DMP) is envisioned that would enable users to perform a variety of chemical and biological analyses with minimal user intervention or microfluidics expertise.

As elucidated by Prof. Whitesides, a luminary in the realms of microfluidics and chemistry, simple technologies have a few defining characteristics [18]:

- **Predictability and reliability** — function is obvious and repeatable
- **High performance-to-cost ratio** — cheap enough to be compelling in its niche
- **Stackability** — can be combined/stacked as building blocks for higher abstraction levels and broader application scopes (e.g. transistors to general-purpose computers)

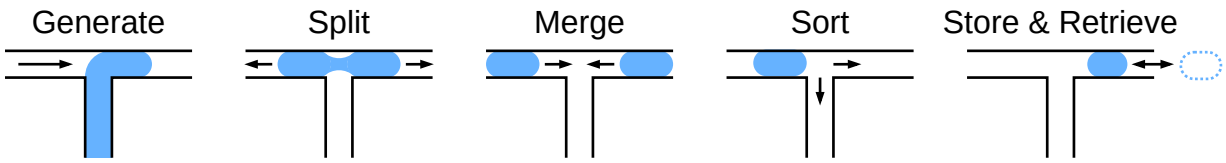


Figure 1.1: Individual droplet manipulations using T-junction geometry

Following these guiding principles, the vision of a general-purpose DMP has been further developed by Hebert [17]. The building blocks of droplet microfluidics applications are the individual droplet manipulations, most of which can be performed with a simple T-junction microchannel geometry (Figure 1.1). Thus one potential platform architecture consists of a set of stackable modules, with each module based around a single T-junction microchannel. This allows a single module to perform any individual droplet manipulation while minimizing cost and complexity. For predictability and reliability, each module would include some method for droplet actuation and sensing to ensure manipulations are fully autonomous, thereby minimizing user error. By stacking/combining modules fluidi-

cally/electrically/mechanically, arbitrary sequences of manipulations can be autonomously executed, increasing the scope and complexity of applications. Additionally, stackability would enable higher level user inputs that approach natural language, maintaining ease-of-use as complexity grows [19]. Given these benefits, it's evident that a modular and automated approach will be key to realizing general-purpose DMPs that are practical and accessible for end-users [17].

1.2 Progress and Challenges

As a first step toward fully automated droplet manipulation, Wong identified a droplet motion model and designed a feedback controller around it to stabilize droplet interfaces in T-junction microchannel networks and decouple droplet dynamics between junctions [4]. Droplet actuation was performed by externally adjusting inlet pressures to produce pressure-driven flow (as in passive microfluidics), and droplet sensing was performed for feedback control (an active microfluidics approach) via digital microscopy. Combining the benefits of passive (i.e. leveraging geometry for droplet manipulations) and active techniques (i.e. real-time flow field control) enabled individual droplet manipulations on-demand without the need for custom-built embedded actuators/sensors. Wong also developed a software application — RoboDrop — to provide a graphical user interface (GUI) for individual droplet manipulations [4]. Further work by Hebert and Zablotny [20, 21] implemented higher levels of automation and software abstraction, reducing the amount of manual user intervention required to set up and execute individual droplet manipulations. Lensless imaging techniques have also been applied by Zablotny to significantly reduce the cost and complexity of droplet sensing hardware [22].

However, the use of the proposed platform in real-world physical and life science applications is still limited by practicality and accessibility challenges in software workflow, control system tuning, and droplet actuation methods.

Applications of droplet microfluidics often require long droplet manipulation sequences over extended time periods [23], but RoboDrop's droplet manipulation workflow requires the user to manually intervene at different key points to set up and execute each manipulation [20]. Each type of manipulation also has associated parameters that must be fine-tuned by the user to ensure it is performed successfully [4]. Thus as droplet manipulation sequences grow in length and complexity, this workflow becomes impractical for end-users due to the lack of full automation and inability to generalize manipulations.

Additionally, previous feedback control systems using linear time-invariant (LTI) droplet

models [4, 20, 21] neglect non-linear and/or time-varying dynamics (behavior near junctions, accumulation of generated droplets, variability of microchannel surface wetting, dust/defects, etc), degrading performance and reliability of longer droplet manipulation sequences. Optimal model and controller parameters also require manual tuning by the end-user for each application, limiting accessibility for users without controls background.

On droplet actuation, common methods [24] for pressure-driven flow generation are impractical for the modular and automated nature of the proposed platform, being either too slow and imprecise for controlling individual droplet manipulations (e.g. peristaltic/syringe pumps) or too complex to be modularized in a cost-effective way (e.g. pressure-driven flow controllers).

1.3 Thesis Overview

This thesis aims to address the aforementioned challenges in control, software workflow, and droplet actuation to make the envisioned DMP design more practical and accessible for end-users.

To tackle the control challenge, we aim to apply adaptive control to automate the model parameter identification process, as well as to automatically compensate for changes in droplet dynamics over longer sequences of manipulations. Regarding software, our goal is to develop a fully automated end-user workflow, allowing users to perform arbitrary manipulation sequences without manual intervention. For droplet actuation, we will investigate recent commercially available piezoelectric micropumps as a more practical (in terms of cost, footprint, and portability) droplet actuation method for the proposed DMP.

Chapter 2 provides an overview of existing techniques in passive, active, and feedback-controlled droplet microfluidics, as well as a brief background on droplet models and control theory. Chapter 3 describes the hardware and software components used throughout this thesis. Chapter 4 discusses the design and evaluation of an Adaptive Model Predictive Controller (AMPC) for automated droplet manipulation. Chapter 5 details the design of a new automated end-user workflow for designing and executing sequences of droplet manipulations. Chapter 6 validates a low-cost, modular piezoelectric micropump design for automated droplet manipulation. Chapter 7 summarizes the thesis and makes some suggestions for improvements as well as future work.

Chapter 2

Literature Review and Background

This chapter provides a review of the literature in the field of droplet microfluidics, specifically focusing on the techniques developed for droplet manipulation. This is followed by a brief background on modelling and control of droplet microfluidics systems.

2.1 Microfluidic Droplet Manipulation Techniques

The ability to control and manipulate droplets in microfluidic devices has been a cornerstone in the advancement of the field. This has led to a myriad of applications demonstrated in various domains [17], including cell biology (e.g. single [25] and multi-cell analysis [26]), biochemistry (e.g. drug screening [27] and protein crystallization [28]), material synthesis (e.g. nanoparticles [29] and emulsions [30]), and environmental monitoring (e.g. heavy metals detection [31]).

2.1.1 Passive Microfluidics

Passive techniques rely on microchannel geometry and a fixed flow field to implement specific droplet manipulations (e.g. droplet generation through T or flow focusing junctions, droplet trapping [32, 33]). The flow field is usually generated with regulated pressure or flow sources such as pressure pumps or syringe pumps. These techniques have been widely used due to their simplicity and reliability. However, they lack the flexibility and adaptability provided by active techniques.

2.1.2 Active Microfluidics

Active techniques, on the other hand, add external energy (e.g. electrical, mechanical, thermal) to the flow field locally to control droplet manipulations. These techniques usually require external actuators (e.g. pneumatic valves [34], piezoelectric actuators [35], electric fields [36], heating elements [37], or lasers [38, 39]) to modify the flow field at specific points.

Digital microfluidics, particularly electrowetting-on-dielectric (EWOD), has seen significant advancements and applications in the past few decades. EWOD allows for the manipulation of small liquid droplets on a two-dimensional surface, making it particularly promising for portable systems due to its mechanical simplicity and low energy consumption [40]. Li et al. (2019) introduced a novel method of droplet manipulation that uses electrical signals to induce the liquid to dewet, rather than wet, a hydrophilic conductive substrate [41]. EWOD has found commercial applications in areas such as optical devices, including liquid lenses and reflective displays, and biomedical devices, such as DNA library preparation and molecular diagnostics [42].

Surface Acoustic Wave (SAW) microfluidics is a technology that utilizes sound waves to manipulate fluid droplets on a microscale. Franke et al. (2009) demonstrated the use of SAW devices for directing the motion of droplets in microfluidic channels. This method allows individual droplets to be directed along separate microchannel paths at high volume flow rates, which is useful for droplet sorting [43].

Both SAW and EWOD offer high precision and flexibility, but they also require complex setups and high energy inputs. Additionally for EWOD, droplet sizes are difficult to vary, and minimum droplet volumes are fundamentally limited. Droplet evaporation and contamination are also challenges, but can be resolved through the use of disposable microfluidic chips.

Quake's approach, often referred to as "Quake's valves," leverages the soft nature of polydimethylsiloxane (PDMS) to create on-chip valves. By combining a control and sample layer, the valves can block or allow flow at specific points in each channel. The actuation of multiple valves can confine small sample quantities similar to droplets, but in a purely single-phase flow [44].

Other approaches similar to Quake's valves have been investigated, including electroactive polymer-based valves [45], pH-activated valves [46], and wax valves [47].

Garstecki's research group developed an off-chip valve-based approach for the continuous growth and monitoring of bacterial populations. Although the system is single-phase, automated on/off valves allow accurate droplet splitting and manipulations via real-time

video feedback. This method allows for parallel, long-term studies of microbial ecology, physiology, evolution, and adaptation to chemical environments [48].

In another work, Garstecki's group developed an active control method using in-line solenoid valves. Unlike Quake's parallel approach, these valves are directly integrated in series with the flow source, avoiding complex multi-layer chip fabrication [49]. Furthermore, the microflow involved is two-phase flow (i.e., droplet microfluidics) instead of single-phase flow, which offers many advantages, including compartmentalization of reactions and high-throughput without compromising uniformity. However, the in-line solenoid valve system still requires involved user interaction to operate the system, and microfluidics knowledge to set up and operate the actuation pressure behind each of the solenoid valves.

2.1.3 Feedback-Controlled Microfluidics

There have been many applications of feedback control in droplet microfluidics, but most have been targeted toward digital microfluidics platform due to the ease of control. In more traditional pressure-driven droplet microfluidics systems, feedback control has largely only been applied to tune parameters in a passive droplet generation process, e.g. droplet size and spacing between droplet trains [50, 51].

Wong's approach was the first to aim toward a droplet microfluidics platform that could perform a variety of required droplet operations on the same chip architecture. Utilizing computer vision for droplet sensing, pressure-driven flow for droplet actuation, and a multiple T-junction microfluidic chip architecture, a feedback control system was created that generated and stabilized droplets in each channel despite external disturbances and system uncertainties [52, 53]. Unfortunately, the workflow was manual, requiring the user to break down each droplet manipulation into the required objectives in each channel, defining each objective by dragging a cursor over the channel.

Building on Wong's work, semi-automated manipulations combining multiple intermediate objectives were realized [20], but manual intervention was still required for synchronization.

Zablotny's approach fully automates basic droplet manipulations (generation, splitting, merging, sorting, move) by autonomously completing sequences of low-level objectives or "instructions" [21]. Unfortunately, the droplet manipulations are prone to error and can be easily destabilized by transients or unmodelled dynamics, necessitating significant manual tuning of model/controller parameters and instruction sequences.

An overarching challenge of the aforementioned feedback control techniques is the difficulty of modelling the inherently non-linear and time-varying dynamics encountered in droplet microfluidics.

2.2 Microfluidic Droplet Models for Control

In modeling physical systems, the ideal model is often the one that most accurately explains the relationship between input and output data. In contrast, feedback control relaxes the objective to merely identifying a model-controller combination that ensures a stable closed-loop system while meeting desired control specifications (e.g. rise time, overshoot, control effort) [54].

2.2.1 Interface Displacement Models

We start with Poiseulle flow, common in microfluidics, which involves a steady-state, pressure-driven flow of a single phase incompressible fluid in a long, straight, rigid channel [55]. The analytical solution of the Navier-Stokes equation for Poiseulle flow shows that pressure drop ΔP across this channel is directly proportional to the volumetric flow rate Q [55], producing the Hagen-Poiseulle law

$$\Delta P = R_{hyd} Q \tag{2.1}$$

analogous to Ohm's law ($\Delta V = RI$), with R_{hyd} being the hydraulic resistance.

As with previous approaches, visual detection of droplet position is used for feedback. Simplistically, we can model droplet displacement d as the integral of Q, analogous to electric charge.

The relevant dynamic equation becomes:

$$Q = \dot{d} = \frac{1}{R_{hyd}} \Delta P \tag{2.2}$$

or in Laplace domain:

$$d(s) = \frac{1}{R_{hyd} s} P(s) \tag{2.3}$$

However, integrators tend to be non-ideal and leaky in physical systems due to unmodelled dynamics from non-ideal components. In microfluidics, a physical source of this leakage comes from the gutter flow (Figure 2.1) present in microchannels with non-circular cross-sections (common due to their ease of fabrication), as the thin film between the droplet and channel walls has a non-uniform width.

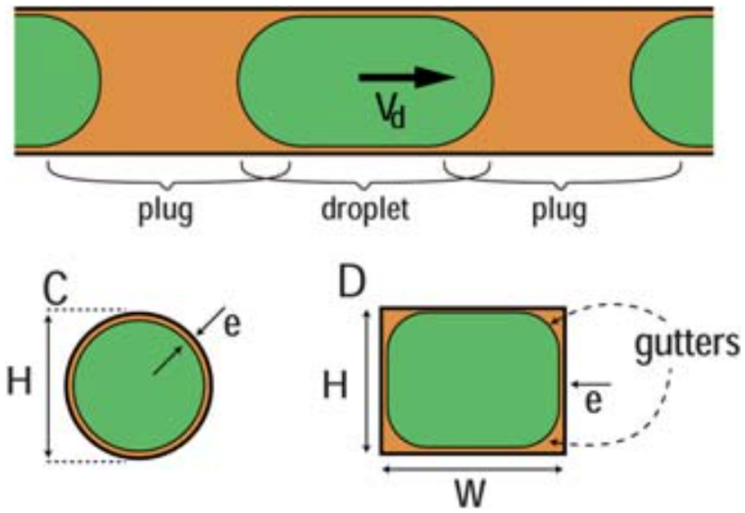


Figure 2.1: Droplet flow through circular vs rectangular cross-section microchannel, gutter locations indicated [1]

As an analogy consider a cylindrical bucket with cross-section area A being filled with liquid at a volumetric flow rate of w. Its height h would be given by integrating the flow, i.e.

$$\dot{h} = \frac{w}{A} \tag{2.4}$$

If a hole is added to the bottom of the bucket, there is now a leakage flow rate that depends on liquid height due to gravity. As a first order approximation let's assume this leakage flow rate is directly proportional to height with a factor τA, the relevant differential equation becomes:

$$\dot{h} = \frac{w}{A} - \tau h \tag{2.5}$$

or in Laplace domain:

$$h(s) = \frac{1/A}{s+\tau} w(s) \tag{2.6}$$

In the same way, we can assume this gutter flow removes a constant but small proportion $\tau \ll 1$ of the ideal droplet displacement d from the overall flow, producing the relevant dynamic equation:

$$Q = \dot{d} = \frac{1}{R_{hyd}} \Delta P - \tau d \tag{2.7}$$

or in Laplace domain:

$$d(s) = \frac{1/R_{hyd}}{s + \tau} P(s) \tag{2.8}$$

Now we will extend this model to a multi-input, multi-output (MIMO) system, using the ubiquitous T-junction droplet generator as an example (Figure 2.2).

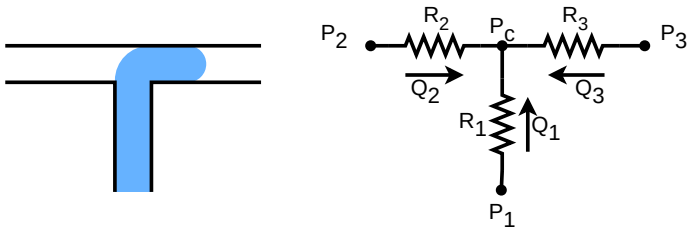


Figure 2.2: T-junction microchannel network and equivalent circuit

Hagen-Poiseulle law with leak modification gives the dynamic equations:

$$\dot{d_1} = Q_1 = -\tau_1 d_1 + \frac{P_1 - P_c}{R_1} \tag{2.9}$$

$$\dot{d_2} = Q_2 = -\tau_2 d_2 + \frac{P_2 - P_c}{R_2} \tag{2.10}$$

$$\dot{d_3} = Q_3 = -\tau_3 d_3 + \frac{P_3 - P_c}{R_3} \tag{2.11}$$

To convert to state-space form

$$\dot{x} = Ax + Bu \tag{2.12}$$

$$y = Cx + Du \tag{2.13}$$

we'll use $x = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}$ as the state vector, $u = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \end{bmatrix}$ as the control input vector, and $y = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}$ as the output vector.

The state matrices A, C, and D are obvious:

$$A = \begin{bmatrix} -\tau_1 & 0 & 0 \\ 0 & -\tau_2 & 0 \\ 0 & 0 & -\tau_3 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$D = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

To find an expression for B, we'll use superposition. Each input pressure and output position relation is found by setting all other input pressures to 0, producing

$$B = \begin{bmatrix} \frac{R_2+R_3}{R_1R_2+R_1R_3+R_2R_3} & \frac{-R_3}{R_1R_2+R_1R_3+R_2R_3} & \frac{-R_2}{R_1R_2+R_1R_3+R_2R_3} \\ \frac{-R_3}{R_1R_2+R_1R_3+R_2R_3} & \frac{R_1+R_3}{R_1R_2+R_1R_3+R_2R_3} & \frac{-R_1}{R_1R_2+R_1R_3+R_2R_3} \\ \frac{-R_2}{R_1R_2+R_1R_3+R_2R_3} & \frac{-R_1}{R_1R_2+R_1R_3+R_2R_3} & \frac{R_1+R_2}{R_1R_2+R_1R_3+R_2R_3} \end{bmatrix}$$

This is a highly simplistic model that doesn't account for channel compliance (hydraulic capacitance), fluid inertia (hydraulic inductance), as well as external components of the system, such as pump and tubing dynamics. However, in the microfluidics context (i.e. small pressure gradient, low Reynolds number, laminar flow), the integral action has been shown to dominate the droplet displacement response in feedback controlled systems [21]. Additionally, unnecessary model parameters can cause the model order to grow very quickly as microchannel networks become more complex due to coupling effects between channels. This can lead to ill-conditioned models [21] that are much more sensitive to numerical precision and model parameter variations, making it harder to design robust controllers that can meet performance specifications.

2.2.2 Optimal Control Objectives

Linear Quadratic Regulation (LQR)

With our state-space droplet displacement model, our objective now is to drive the trajectory of each state $\mathbf{x}_a$ to any user-defined reference trajectory $\mathbf{x}_d$. Additionally, a common objective is to minimize actuator effort $\mathbf{u}$, which can reduce power consumption and prolong the lifetime of the actuator. We can convert this trajectory tracking and control effort problem to an optimization problem by defining the appropriate cost function. Let $\mathbf{E}(t) = \mathbf{x}_a(t) - \mathbf{x}_d(t)$ be the state trajectory error (i.e. difference between actual state and desired state), and Q and R be constant positive semidefinite weight matrices. Then we can define an infinite-horizon (i.e. acting over all time) quadratic cost function:

$$\mathbf{J} = \int_0^\infty (\mathbf{E}^T Q \mathbf{E} + \mathbf{u}^T R \mathbf{u}) dt \tag{2.14}$$

Minimizing this cost function produces the optimal state, output, and input trajectories given the reference trajectory $\mathbf{x}_d(t)$. In a perfect world, applying the optimal input trajectory in open-loop (without feedback) will produce the optimal output trajectory. However, real systems have uncertainties (e.g. actuator/sensor noise, unmodelled plant dynamics) that cause both output and state trajectories to diverge from optimality over time. By closing the loop, i.e. measuring the current state trajectory through state feedback (assuming the system is stabilizable and detectable), the optimal input trajectory becomes robust to noise and disturbances. Solving the associated Riccati equation produces the optimal state feedback gain K and associated control law (for infinite-horizon cost) relating current state to control input known as the linear quadratic regulator (LQR):

$$\mathbf{u} = -K\mathbf{x} \tag{2.15}$$

Because LQR optimizes over the entire time horizon without constraining inputs/outputs, the resulting control law has guaranteed stability margins independent of the plant [56].

Linear quadratic Gaussian (LQG)

In linear physical systems we can often model uncertainties as additive white-noise Gaussian distributions $B_v\mathbf{v}(t), \mathbf{w}(t)$:

$$\dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t) + B_v\mathbf{v}(t) \tag{2.16}$$

$$\mathbf{y}(t) = C\mathbf{x}(t) + \mathbf{w}(t) \tag{2.17}$$

the cost functional becomes the expected value of the LQR cost:

$$\mathbf{J} = \lim_{T\to\infty} \frac{1}{T} \int_0^T E\big[(\mathbf{E}^T Q\mathbf{E} + \mathbf{u}^T R\mathbf{u})\big] dt \tag{2.18}$$

Minimizing this cost functional is known as the linear quadratic Gaussian (LQG) problem. Assuming the system is stabilizable and detectable, minimizing $\mathbf{J}$ produces the optimal state feedback control law. This is equivalent to minimizing the steady-state mean square error:

$$\lim_{t\to\infty} E\big[(\mathbf{E}^T Q\mathbf{E} + \mathbf{u}^T R\mathbf{u})\big] \tag{2.19}$$

In many physical systems, only output feedback is available (i.e. some states cannot be directly measured). Thus we can apply the separation principle to split the LQG problem into a linear quadratic estimator (LQE or Kalman filter) and LQR problem that can be independently solved as two LQR problems due to duality [57]. Two Riccati equations are derived, the solutions of which produce an observer law (Kalman gain) and control law (state feedback gain).

This is an elegant solution, but the lack of perfect state feedback means stability margins are no longer guaranteed [56].

State and Disturbance Estimation

Since our model has no unmeasurable states, we could directly apply state feedback without state estimation. In practice, a state estimator is still beneficial here. Apart from filtering measurement noise, it can compensate for delays or outliers in measurements by relying on the predicted output of its internal model.

Specific to this system, one physical phenomena that is not directly measured is the Young-Laplace pressure, a pressure differential generated across a droplet interface due to interfacial tension [55].

Along a channel with a fixed cross-section, Laplace pressure remains constant, producing a constant flowrate and a linearly increasing displacement. But near junctions or situations where channel cross-sections change, Laplace pressure can change non-linearly, making it difficult to identify the exact value of Laplace pressure in any particular scenario. Thus we will model this as a ramp disturbance scaled by some Gaussian white noise, i.e.

$$y_{od} = 1/s^2 w_{od},\ w_{od} \sim N(0,1), y_{od} \sim N(0, t^2/2) \tag{2.20}$$

or in state-space:

$$\dot{x}_{od}(t) = A_{odc}x_{od}(t) + B_{odc}w_{od}(t) = \begin{bmatrix}0 & 0\\ 1 & 0\end{bmatrix} x_{od}(t) + \begin{bmatrix}1\\ 0\end{bmatrix} w_{od}(t) \tag{2.21}$$

$$y_{od}(t) = C_{odc}x_{od}(t) + D_{odc}w_{od}(t) = \begin{bmatrix}0 & 1\end{bmatrix} x_{od}(t) \tag{2.22}$$

$$\tag{2.23}$$

This will allow the state estimator to more accurately predict random ramp disturbances caused by Laplace pressure and other non-linearities so the controller can better reject them.

As our control system will be implemented on a digital computer, we need to discretize all prediction models to match some desired sampling period T. The standard approximation to obtain a discrete equivalent model is the zero-order-hold (ZOH), where model inputs and outputs are held constant between samples. Using this approach, the discrete disturbance model is given in state-space as:

$$x_{od}(k+1) = A_{od}x_{od}(k) + B_{od}w_{od}(k) \tag{2.24}$$

$$= e^{A_{odc}T}x_{od}(k) + \int_0^T e^{A_{odc}\tau}d\tau B_{odc}w_{od}(k) \tag{2.25}$$

$$\approx (I + A_{odc}T)x_{od}(k) + \int_0^T \begin{bmatrix}1\\ \tau\end{bmatrix} d\tau w_{od}(k) \tag{2.26}$$

$$= \begin{bmatrix}1 & 0\\ T & 1\end{bmatrix} x_{od}(k) + \begin{bmatrix}T\\ T^2/2\end{bmatrix} w_{od}(k) \tag{2.27}$$

$$y_{od}(k) = C_{od}x_{od}(k) + D_{od}w_{od}(k) \tag{2.28}$$

$$= \begin{bmatrix}1 & 0\end{bmatrix} x_{od}(k) \tag{2.29}$$

We can also model the measurement uncertainty w_n as Gaussian white noise:

$$x_n(k+1) = A_n x_n(k) + B_n w_n(k) = 0 \tag{2.30}$$

$$y_n(k) = C_n x_n(k) + D_n w_n(k) = w_n(k) \tag{2.31}$$

$$w_n(k) \sim N(0,1) \tag{2.32}$$

For brevity only the single-input single output (SISO) — i.e. single channel — disturbance and measurement noise models are described. In reality a MIMO system is modelled, meaning these SISO models are duplicated for each channel, and cross-coupling between

channels are considered (disturbances in a single channel couple to all other connected channels).

Now we can augment the plant (T-junction) model with our disturbance and noise models. Let's discretize the original plant so it can be used by the estimator (the calculation for B_{pu} becomes quite complex, so we'll represent each element using R_{ij}):

$$x_p(k+1) = A_p x_p(k) + B_{pu} u(k) \tag{2.33}$$

$$\approx (I + AT) x_p(k) + \int_0^T e^{A\nu} d\nu B u(k) \tag{2.34}$$

$$= \begin{bmatrix} 1-\tau_1 T & 0 & 0 \\ 0 & 1-\tau_2 T & 0 \\ 0 & 0 & 1-\tau_3 T \end{bmatrix} x_p(k) + \begin{bmatrix} R_{11} & -R_{12} & -R_{13} \\ -R_{21} & R_{22} & -R_{23} \\ -R_{31} & -R_{32} & R_{33} \end{bmatrix} u(k) \tag{2.35}$$

$$y_p(k) = C_p x_p(k) + D_{pu} u(k) \tag{2.36}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x_p(k) \tag{2.37}$$

The combined prediction model is then defined:

$$x_c = \begin{bmatrix} x_p \\ x_{od} \end{bmatrix}, \quad u_o = \begin{bmatrix} u \\ w_{od} \\ w_n \end{bmatrix} \tag{2.38}$$

$$x_c(k+1) = \begin{bmatrix} A_p & 0 \\ 0 & A_{od} \end{bmatrix} x_c(k) + \begin{bmatrix} B_{pu} & 0 & 0 \\ 0 & B_{od} & 0 \end{bmatrix} u_o(k) \tag{2.39}$$

$$y(k) = \begin{bmatrix} C_p & C_{od} \end{bmatrix} x_c(k) + \begin{bmatrix} D_{pu} & D_{od} & D_n \end{bmatrix} u_o(k) \tag{2.40}$$

with plant states x_p, disturbance states x_{od}, combined states x_c, and combined inputs u_o (including both controlled and uncontrolled inputs).

Model Predictive Control (MPC)

Infinite-horizon LQR/LQG finds a single optimal state/output trajectory over all time, and the corresponding optimal input trajectory. However, in physical systems one must consider the presence of input/output constraints, as well as input slew rate limitations. The addition of these constraints make the optimization problem non-linear and much more difficult to solve over the entire time horizon.

Alternatively, we can find an input/output trajectory that is optimal for a finite prediction horizon starting at each point in time a new measurement is received, reducing the problem to a series of convex optimizations (if constraints are linear). This technique is known as linear Model Predictive Control (MPC). Because the control law does not necessarily use the measurement for feedback directly, MPC is essentially solving a sequence of open-loop optimization problems, with the current measurement only providing the initial conditions (indirect feedback) [57].

The cost function for the linear MPC optimization problem is very similar to LQR, but only optimizes over the prediction horizon p:

$$\mathbf{J} = \sum_{i=0}^{p-1} \left\{ \mathbf{E}^T Q \mathbf{E} + \mathbf{u}^T R \mathbf{u} \right\} \tag{2.41}$$

while satisfying input, input slew rate, and output constraints:

$$y_{min} \leq y[k+i|k] \leq y_{max} | i = 1:p \tag{2.42}$$

$$u_{min} \leq u[k+i-1|k] \leq u_{max} | i = 1:p \tag{2.43}$$

$$\Delta u_{min} \leq \Delta u[k+i-1|k] \leq \Delta u_{max} | i = 1:p \tag{2.44}$$

Given that the cost function is quadratic with linear constraints, a quadratic programming (QP) problem can be formulated at each time step to find the optimal input trajectory [58].

Chapter 3

Methodology

This chapter describes each hardware and software component involved in the operation of the proposed feedback-controlled droplet microfluidics platform (DMP). These components are organized into subsystems, with the flow of information between them shown in Figure 3.1.

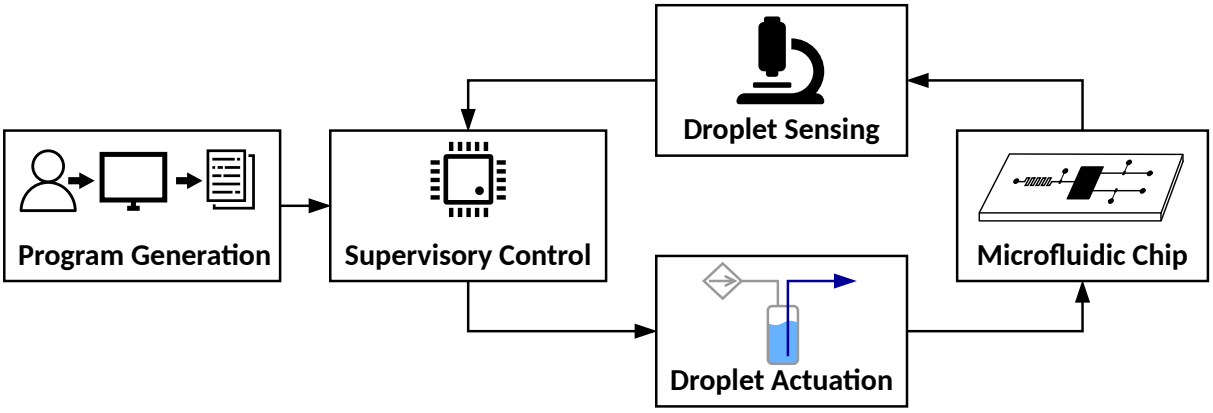


Figure 3.1: Overview of feedback-controlled DMP subsystems

3.1 Hardware

3.1.1 Microfluidic Chip

A microfluidic chip consists of a network of micrometer- to millimeter-width fluid channels where droplets can be generated and manipulated. Droplet microfluidics requires at least two phases of fluid to be present: dispersed phase (which make up the droplets) and continuous/carrier phase (which surround the droplets). In many applications, the dispersed phase is aqueous and thus hydrophilic, meaning microchannels must be hydrophobic to prevent dispersed phase droplets from wetting channel walls. Additionally for many microfluidics applications and feedback control especially, microchannels must be transparent to ensure droplets are observable.

In this thesis, Polydimethylsiloxane (PDMS) and glass microfluidic chips are utilized. Both PDMS and glass are transparent to light, can be modified to become hydrophobic, and their use in microfluidic chip construction is quite mature. PDMS chips (Dow Corning Sylgard 184) are fabricated in-house using standard soft lithography techniques [59]. Unfortunately, the porous nature of PDMS causes contamination, and poor long-term hydrophobicity degrades the performance of droplet manipulations over time, rendering each chip as single-use for practical applications. As an alternative, glass chips are obtained off-the-shelf from Dolomite Microfluidics, who fabricate them through a wet-etching process common in the semiconductor industry [60]. Glass chips have higher up-front cost, but can be readily cleaned and reused, providing a long-term platform for droplet manipulations.

The most fundamental manipulation, droplet generation, is commonly implemented using the T-junction. By combining multiple T-junctions, additional sequences of manipulations can be performed on the generated droplets (Figure 1.1). To demonstrate droplet manipulations, deionized (DI) water and 5 [cSt] Silicone oil (Sigma-Aldrich) are used as the dispersed and continuous phases.

3.1.2 Droplet Sensing

Droplet sensing is achieved by integrating an inverted microscope (Nikon Eclipse Ti-E), machine vision camera (Andor Zyla 5.5 sCMOS), and laptop PC (Figure 3.2).

The microscope objective is selected to provide the optimal magnification to observe the Region-of-Interest (ROI) for a given chip. The camera then captures and sends frames within the ROI at 40 [Hz] to the PC over USB 3.0. Finally, an image processing algorithm

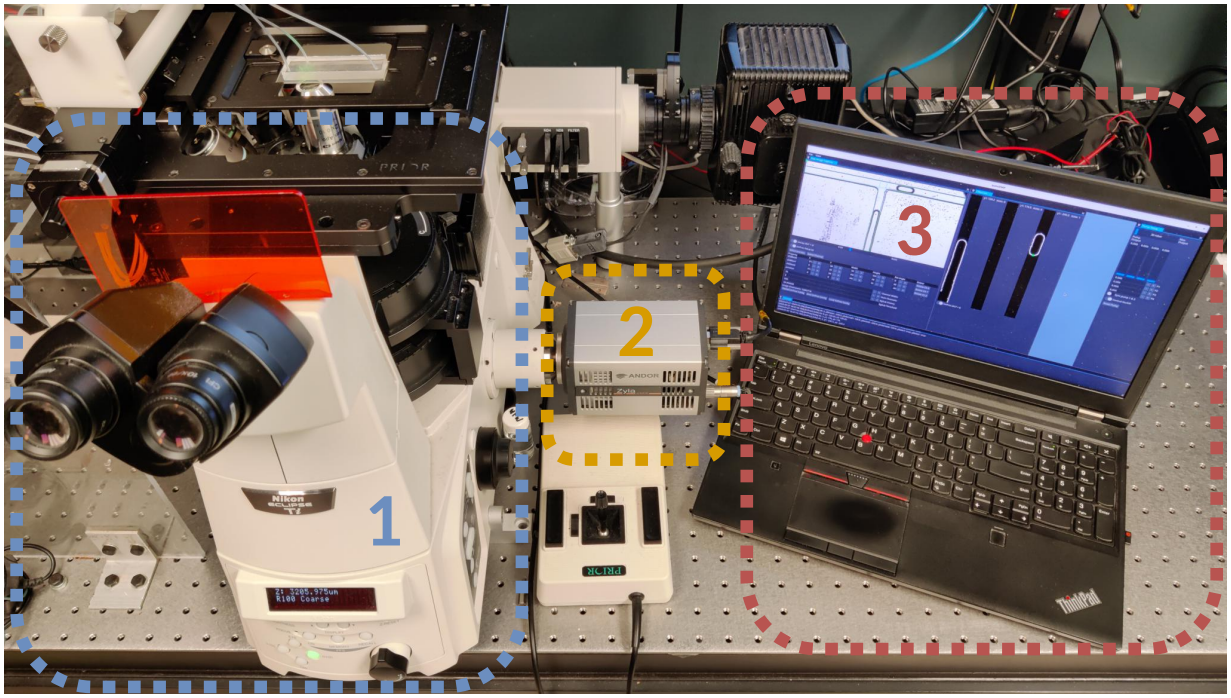


Figure 3.2: Droplet sensing setup (1) microscope (2) machine vision camera (3) laptop PC

runs on the PC to track droplet positions in real-time. The implementation of the droplet tracking algorithm is detailed in Appendix B.

3.1.3 Sample Transport

Before droplets can be actuated on-chip, we need a way to bring the dispersed and continuous phases to the ROI. Fluids of each phase are collected in an array of reservoirs (Fluigent Fluiwell) and connected via perfluoroalkoxy (PFA) tubing (Sigma-Aldrich) to each inlet of the microfluidic chip (parts 2 and 3 in Figure 3.3).

Providing a robust, leak-free interface (i.e. the macro-micro interface) between tubing and chip is a common problem in microfluidics. For PDMS chips, the elastic nature of PDMS allows tubing to be directly plugged into chip inlets without leakage (Figure 3.3). For glass chips, an external connector is required to provide a leak-free seal while interfacing external tubing to the chip (Figure 3.4).

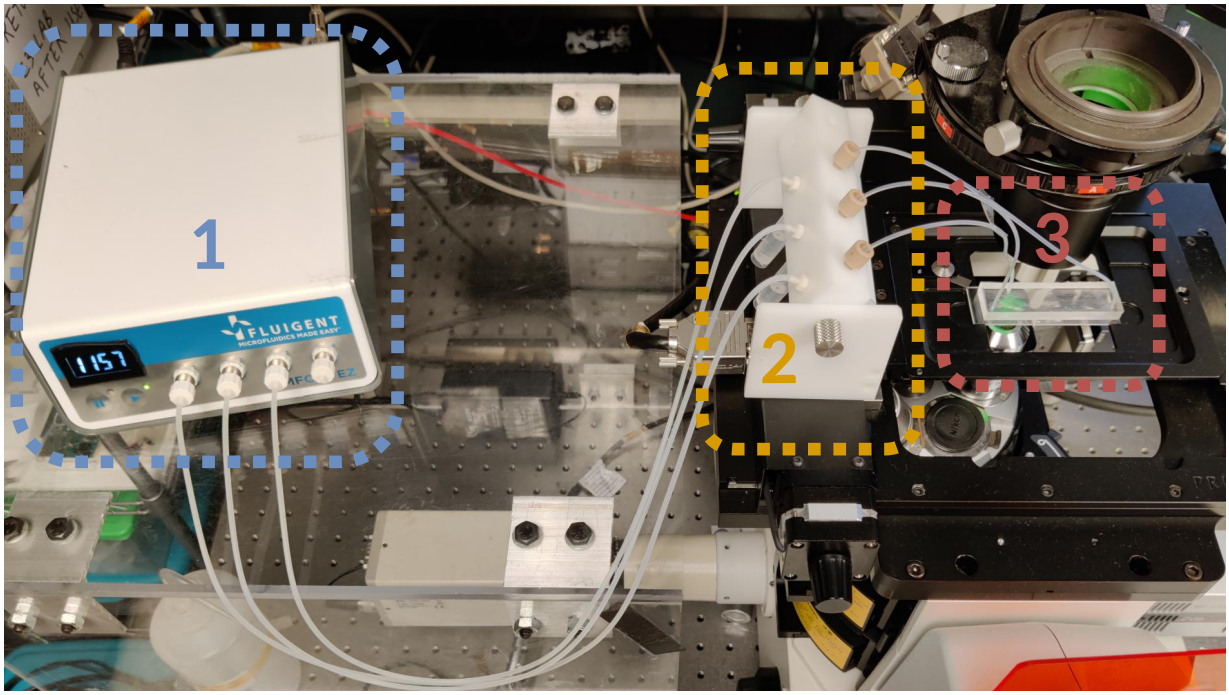


Figure 3.3: Droplet actuation and sample transport setup (1) pressure source (2) reservoir array (3) microfluidic chip

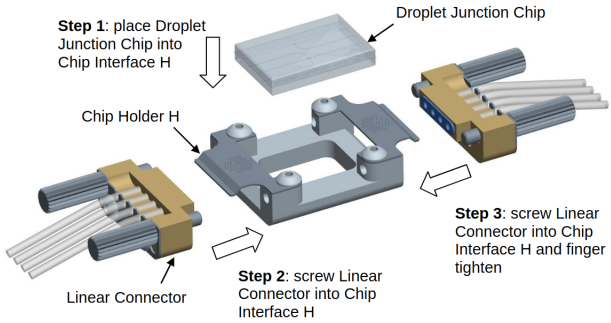


Figure 3.4: Macro-to-micro interface for Dolomite glass microfluidic chip [2]

3.1.4 Droplet Actuation

To drive the fluid phases to the ROI, each reservoir is pneumatically pressurized by a variable pressure source through Silicone tubing (part 1 in Figure 3.3). Since the pressure source is only in contact with air, there is no contamination between the actuator and

sample. By modulating each pressure source, droplets are provided the maximum degrees of freedom.

Unless otherwise stated, the pressure source used is a commercial pressure pump (Fluigent MFCS-EZ). The pump is provided with a fixed input pressure of 1.3 [bar], and communicates with the laptop PC over USB. Fluigent provides a public application programming interface (API) to interface custom software to the pump [61]. When a pressure command is sent to the pump, it uses internal pressure feedback to drive the input pressure to the desired pressure.

3.2 Software

3.2.1 Supervisory Control

The supervisory controller is composed of a high-level supervisor and low-level feedback controller. Both the supervisor and controller are implemented in MATLAB/Simulink, and deployed to a laptop PC via C++ code generation.

The controller is based on a LTI state-space model derived in Chapter 2 that captures the dynamics of the system, where inputs represent chip inlet pressures, and outputs represent droplet positions. Based on this model, an Adaptive Model Predictive Controller (AMPC) is used to drive the outputs according to the trajectories provided. The AMPC implementation is detailed in Chapter 4.

The supervisor is given a list of waypoints and generates a trajectory to be tracked by the controller. The supervisor implementation is detailed within the software workflow developed in Chapter 5.

3.2.2 Program Generation

At the highest level, the user will issue a list of commands that correspond to a sequence of fundamental droplet manipulations. Each fundamental droplet manipulation is broken down into a list of necessary waypoints to be reached in each microchannel. Thus any sequence of droplet manipulations can be converted to a list of waypoints to be given to the supervisor, analogous to compilation of a high-level programming language to low-level assembly,

Implementation of the program generation method is detailed within the software workflow developed in Chapter 5.

Chapter 4

Adaptive MPC Design for Droplet Control

4.1 Introduction

Stable feedback control of droplet interfaces is challenged by unavoidable non-linear and time-varying dynamics that occur during droplet manipulation. Each stage of droplet generation (Figure 4.1) locally alters the pressure field around the junction in a highly non-linear manner [3]. As each droplet is detached, it also contributes a persistent local pressure drop in its associated channel due to interfacial tension between different fluid phases [62]. Over time, changes in microchannel hydrophobicity [15], manufacturing defects, and debris can affect the pressure field locally or over large parts of the microchannel network. Each change in the local pressure field contributes to a change in the overall hydraulic resistance of the corresponding microchannel.

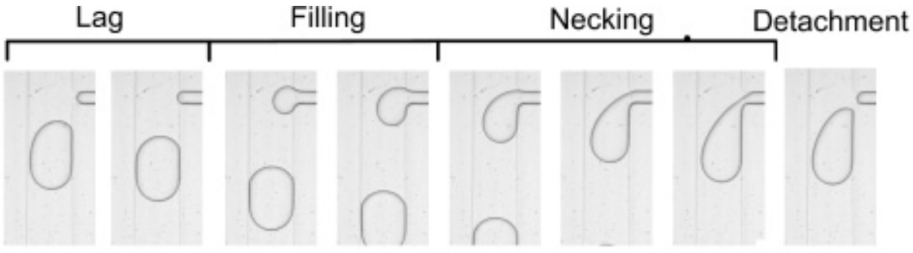


Figure 4.1: Stages of droplet generation in T-junction microfluidic chip [3]

Since the majority of non-linear dynamics appear when droplet interfaces are near the junction, the prevalent approach [4, 21] for droplet control — gain scheduling — is to identify a set of models and controllers at different operating points in the system (far from the junction) and switch between them when specific conditions are met (e.g. junction crossing occurs). In this approach, the non-linear dynamics are treated as temporary disturbances to be rejected when switching controllers (e.g. when droplet interfaces cross the junction into a new microchannel). Thus each model is approximated as linear time-invariant (LTI), greatly simplifying the associated controller design.

On the other hand, time-varying effects can permanently alter system dynamics, increasing the risk of instability as model errors accumulate over longer droplet manipulation sequences. Moreover, any change in tubing diameter/length, channel width, fluid viscosity, etc will require model parameters to be manually updated and tuned to match the physical system. Crucially, aside from trial and error, end-users without system identification expertise cannot determine if the current model parameters are adequate for droplet manipulation in their specific system.

The need for manual tuning by a skilled end-user and the inability to model time-varying dynamics limit the accessibility and practicality of this feedback control strategy. In this chapter, an adaptive model predictive control (AMPC) strategy is proposed to address these limitations.

Model predictive control (MPC) produces optimal control signals for trajectory tracking while taking input and output constraints into consideration. However, large changes in system dynamics can cause significant performance degradation due to inaccurate predictions from its internal model. AMPC augments linear MPC with an online parameter estimator, allowing model parameters to quickly converge to those of the true physical system without user intervention. We show that AMPC is able to maintain droplet trajectory tracking performance despite large changes in system dynamics, while the equivalent non-adaptive MPC massively deteriorates in performance.

4.2 AMPC Design

4.2.1 Overview

The overall AMPC architecture is shown in Figure 4.2 below. The plant encompasses all DMP components, including pump, sample transport, and microfluidic chip dynamics. The DMP model, linear MPC and state estimator formulation are derived in Chapter 2.

Trajectory generation produces a smooth path to drive droplets to desired positions under feedback control, and is discussed in detail in Chapter 5.

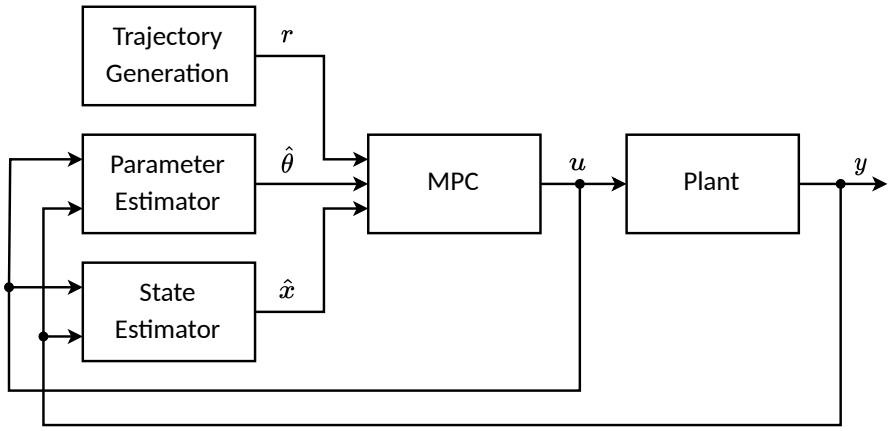


Figure 4.2: Adaptive model predictive controller architecture

4.2.2 Online Parameter Estimator Design

In this section, we derive the parametric model, associated estimation model, and recursive least squares (RLS) adaptive law for online estimation of DMP model parameters.

Model Parametrization

For the i-th microchannel in an n-channel, single junction microfluidic chip, the state space model of droplet interface displacement vs pressure is given by (Chapter 2):

$$x_{i,k} = A_i x_{i,k-1} + B_i u_{k-1} \tag{4.1}$$

$$= (1 - \tau_i)\, x_{i,k-1} + \begin{bmatrix} -b_{i,1} & -b_{i,2} & \cdots & b_{i,i} & \cdots & -b_{i,n} \end{bmatrix} u_{k-1} \tag{4.2}$$

$$y_{i,k} = C_i x_{i,k} = x_{i,k} \tag{4.3}$$

We can form a static parametric model (SPM) [63] by a linear combination of param-

eters θ and regressors ψ (moving all signs to the regressor ψ for convenience):

$$z_{i,k} = x_{i,k} - x_{i,k-1} \tag{4.4}$$

$$= \begin{bmatrix} -x_{i,k-1} & -u_1 & -u_2 & \cdots & u_i & \cdots & -u_n \end{bmatrix} \begin{bmatrix} \tau_i \\ b_{i,1} \\ b_{i,2} \\ \vdots \\ b_{i,i} \\ \vdots \\ b_{i,n} \end{bmatrix} = \psi_{i,k}^T \theta_i \tag{4.5}$$

Extending to multiple channels:

$$z_k = x_k - x_{k-1} = \begin{bmatrix} \psi_{1,k}^T & & & \\ & \psi_{2,k}^T & & \\ & & \ddots & \\ & & & \psi_{3,k}^T \end{bmatrix} \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_n \end{bmatrix} = \psi_k^T \theta \tag{4.6}$$

$$\psi_k = \begin{bmatrix} -x_{1,k-1} & & & & & & & & & \\ u_{1,k-1} & & & & & & & & & \\ -u_{2,k-1} & & & & & & & & & \\ \vdots & & & & & & & & & \\ -u_{n,k-1} & & & & & & & & & \\ & -x_{2,k-1} & & & & & & & & \\ & -u_{1,k-1} & & & & & & & & \\ & u_{2,k-1} & & & & & & & & \\ & \vdots & & & & & & & & \\ & -u_{n,k-1} & & & & & & & & \\ & & \ddots & & & & & & & \\ & & & -x_{i,k-1} & & & & & & \\ & & & -u_{1,k-1} & & & & & & \\ & & & -u_{2,k-1} & & & & & & \\ & & & \vdots & & & & & & \\ & & & u_{i,k-1} & & & & & & \\ & & & \vdots & & & & & & \\ & & & -u_{n,k-1} & & & & & & \\ & & & & \ddots & & & & & \\ & & & & & -x_{n,k-1} & & & & \\ & & & & & -u_{1,k-1} & & & & \\ & & & & & -u_{2,k-1} & & & & \\ & & & & & \vdots & & & & \\ & & & & & u_{n,k-1} & & & & \end{bmatrix}, \ \theta = \begin{bmatrix} \tau_1 \\ b_{1,1} \\ b_{1,2} \\ \vdots \\ b_{1,n} \\ \tau_2 \\ b_{2,1} \\ b_{2,2} \\ \vdots \\ b_{2,n} \\ \vdots \\ \tau_n \\ b_{n,1} \\ b_{n,2} \\ \vdots \\ b_{n,n} \end{bmatrix} \tag{4.7}$$

Recursive Least Squares Estimation

The online parameter estimator is implemented by a recursive least squares (RLS) algorithm with modifications for robustness [54, 64].

Given a linearly parameterized plant model (with measured output z, regressor vector ψ, true parameter vector θ)

$$z_k = \psi_{k-1}^T \theta,$$

and associated estimation model (with estimated output $\hat{z}$, regressor vector ψ, estimated parameter vector $\hat{\theta}$)

$$\hat{z}_k = \psi_k \hat{\theta}_{k-1},$$

least squares regression aims to find the $\hat{\theta}$ that minimizes a weighted sum of squared differences between z and $\hat{z}$ as defined by the cost function

$$\hat{\theta}_k = \arg\min_{\theta} \sum_{r=0}^{k} \beta(k,r) \left(z_r - \hat{z}_r\right)^2 .$$

To perform this regression online, $\hat{\theta}$ is updated at each time step k according to an adaptive law with adaptation gain L_k and estimation error $\epsilon_k = z_k - \hat{z}_k$:

$$\hat{\theta}_k = \hat{\theta}_{k-1} + L_k \left(z_k - \hat{z}_k\right)$$

Knowledge of physical constraints in the system allows us to constrain parameter estimates using parameter projection. The equations for adaptive gain and parameter projection are detailed in Appendix A. The adaptive law is implemented in MATLAB and Simulink, and C++ code generation is used to deploy it to the physical system.

4.2.3 MPC Tuning

Linear MPC minimizes a quadratic cost function with linear constraints over a finite prediction horizon p and control horizon m by solving a quadratic program (QP) at each time step. For convenience, the QP solver used is the KWIK algorithm [65] implemented in MATLAB's MPC Toolbox.

The MPC's internal model structure is derived in Chapter 2, with the output variable (OV) being droplet position in each channel, and manipulated variable (MV) being the inlet pressures set by external pressure pumps.

We select the standard cost function J defined in MATLAB's MPC Toolbox, the objectives being the minimization of OV trajectory tracking error (J_y), MV magnitude (J_u), and MV slew rate ($J_{\Delta u}$):

$$J = J_y + J_u + J_{\Delta u} \tag{4.8}$$

At each time step k, the relevant objective costs for a single channel/pump are defined as:

$$J_y = \sum_{i=1}^{p} w_y^2 \left(r_k - y_k\right)^2 \tag{4.9}$$

$$J_u = \sum_{i=0}^{p-1} w_u^2 \left(u_k - \bar{u}_k\right)^2 \tag{4.10}$$

$$J_{\Delta u} = \sum_{i=0}^{p-1} w_{\Delta u}^2 \left(u_k - u_{k-1}\right)^2 \tag{4.11}$$

with relevant tuning weights w_y, w_u, $w_{\Delta u}$.

Weights are tuned via iterative closed-loop simulation to maximize robustness by penalizing aggressive control moves, i.e. penalizing MV slew rate more than OV tracking or MV magnitude. For each channel being controlled, the weights are:

$$\begin{aligned} w_y &= \beta \\ w_u &= \beta \\ w_{\Delta u} &= T/\beta \\ \beta &= 0.13534 \end{aligned}$$

$1/T$ is the MPC update rate, set to 40 [Hz] to match the sampling rate of droplet sensing (Chapter 3). β is an overall adjustment factor tuned to maximize closed-loop robustness.

Furthermore, a prediction horizon p of 20T [s] and control horizon m of T [s] was found to maintain adequate performance without significant computational overhead, with an estimated maximum memory requirement of only 160 [kB] when executing on the physical system.

Finally, OV constraints of 0 to 4.8 [mm] are set from droplet sensing field-of-view limits while conservative MV constraints of 0 to 30 [mbar] are set from pressure pump output range limits (Chapter 3).

4.3 Methodology

4.3.1 Materials and Hardware

A T-junction PDMS (Dow Corning Sylgard 184) microfluidic chip with rectangular 200 [μm] x 200 [μm] microchannel cross-section is fabricated from a 3D-printed (Formlabs)

master. Deionized (DI) water and 5 [cSt] Silicone oil (Sigma-Aldrich) are used as the dispersed and continuous phases. Refer to Chapter 3 for detailed description of supporting hardware.

4.3.2 Parameter Estimation

To compare the performance of adaptive vs non-adaptive MPC, the same internal model structure (Chapter 2) is used for both, with parameter estimation performed online vs offline.

Offline Parameter Estimation

Following standard open-loop system identification procedure [54], a pressure excitation signal is applied to the microfluidic chip at each inlet to obtain input-output data for training. For excitation, a pseudorandom binary sequence (PRBS) is designed in MATLAB (10th order, sampling rate of 40 [Hz] with clock period of 4) to provide uniform excitation to all inputs within a conservative frequency range between 0 and 5 [Hz]. Given the model structure (Chapter 2) and training data, model parameters are identified using prediction error minimization (PEM) in MATLAB's System Identification Toolbox [54].

For model validation, a different PRBS excitation is used to generate input-output validation data. To measure the percent goodness of fit fit_{gof} between simulated model output $\hat{y}$ and measured validation data output y, we use normalized root mean square error (NRMSE), defined as follows for each output:

$$\text{NRMSE} = \frac{\|y - \hat{y}\|}{\|y - \text{mean}(y)\|} \tag{4.12}$$

$$\text{fit}_{\text{gof}} = (1 - \text{NRMSE}) \times 100\% \tag{4.13}$$

Further, residual analysis is used to find the causes of model residuals — parts of the validation data the model cannot explain. Model residuals at each time step are defined as the difference between $\hat{y}$ and y assuming both started at the same location in the previous time step. Auto-correlation of residuals at any time lag except 0 indicates a correlation between past and future outputs that the model has not explained. Cross-correlation of residuals with inputs indicate a correlation between past inputs and future outputs that the model has not explained [54].

Online Parameter Estimation

Online parameter estimation is performed by applying the RLS algorithm to the parameterized estimation model, both derived in Section 4.2.2. The online parameter estimator is implemented in MATLAB/Simulink and deployed to the physical system via C++ code generation. Relevant codes and simulation files can be found in the associated GitHub repository (https://github.com/KevinHQChen/autoDMP).

4.3.3 Droplet Trajectory Tracking

The droplet trajectory tracking performance of both controllers are evaluated for a predefined trajectory (Figure 4.3). To evaluate robustness to time-varying dynamics, we apply a 4x reduction in pump output for an unobserved channel to emulate resistance increase over time, then repeat the same droplet trajectory.

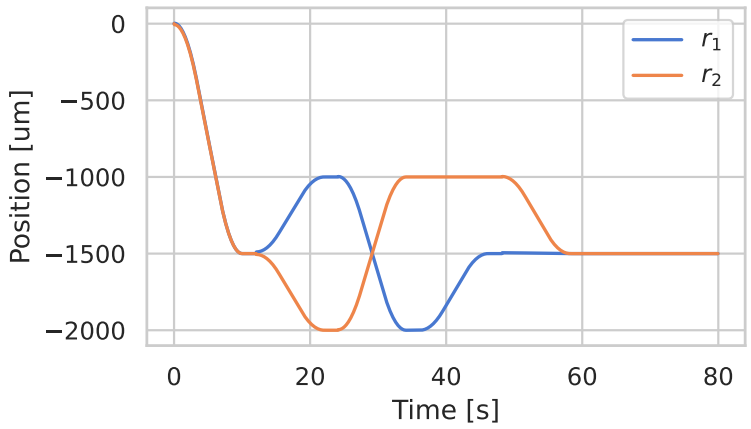


Figure 4.3: Droplet interface reference trajectory for controller validation

Controllers are validated on a simulated plant before deploying to the physical system. To simulate the true system, a physical microchannel model based on the electric circuit analogy is built in Simulink (Figure 4.4) with parameter values taken from system identification results in previous work [4].

A T-junction is modelled based on the microchannel model, and actuator/sensor noise is added (Figure 4.5).

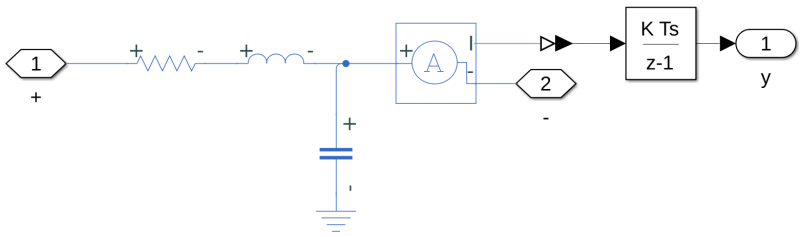


Figure 4.4: RLC model of single microchannel [4]

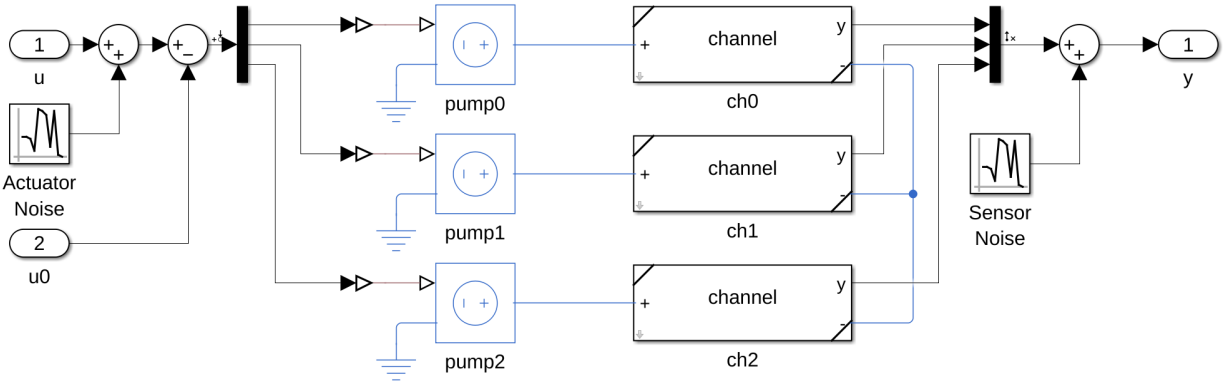


Figure 4.5: RLC T-junction model with actuator and sensor noise

Adaptive MPC and regular MPC Simulink blocks are used to design and test feedback control of the plant in simulation, then C++ code generation is used to deploy each controller to the physical system. Relevant codes and simulation files can be found in the associated GitHub repository (https://github.com/KevinHQChen/autoDMP).

4.4 Results and Discussion

This section analyzes offline vs online parameter estimation results as well as associated non-adaptive vs adaptive closed-loop trajectory tracking performance.

4.4.1 Parameter Estimation

Parameter estimation is performed on a T-junction geometry with two visible droplet interfaces, thus the model structure requires 3 inputs and 2 outputs.

Offline Parameter Estimation

From Section 4.2.2, the relevant state-space model is

$$\begin{bmatrix} x_{1,k} \\ x_{2,k} \end{bmatrix} = \begin{bmatrix} 1-\tau_1 & 0 \\ 0 & 1-\tau_2 \end{bmatrix} \begin{bmatrix} x_{1,k-1} \\ x_{2,k-1} \end{bmatrix} + \begin{bmatrix} b_{11} & -b_{12} & -b_{13} \\ -b_{21} & b_{22} & -b_{23} \end{bmatrix} \begin{bmatrix} u_{1,k-1} \\ u_{2,k-1} \\ u_{3,k-1} \end{bmatrix} \tag{4.14}$$

$$\begin{bmatrix} y_{1,k} \\ y_{2,k} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1,k} \\ x_{2,k} \end{bmatrix} \tag{4.15}$$

Based on the state space model structure above, offline parameter estimation shows good fit to validation data (Figure 4.6) with NRMSE under 21%.

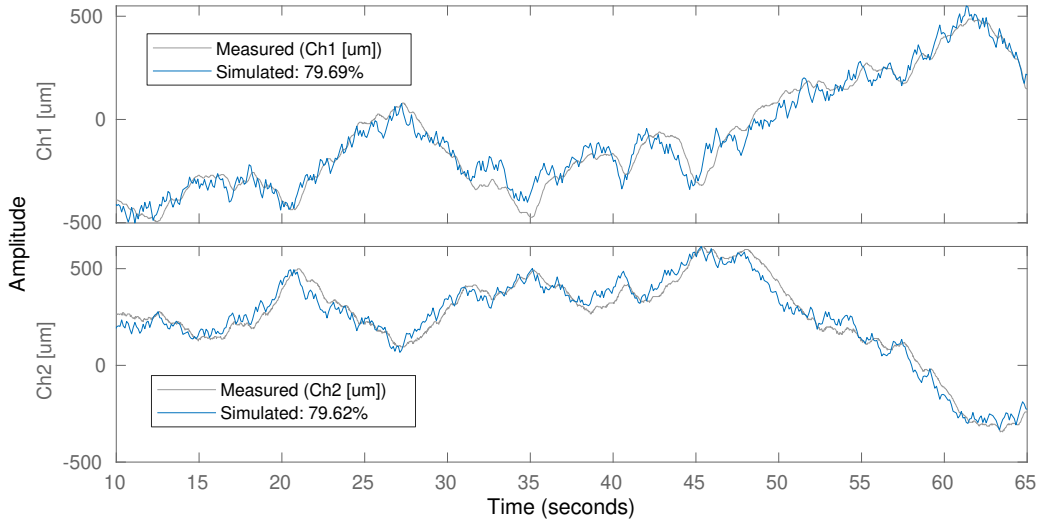


Figure 4.6: Model output prediction, goodness-of-fit labelled (as defined in Equation 4.13)

Model residuals and associated 99% confidence intervals are shown in Figure 4.7. Residuals are mostly uncorrelated with themselves, indicating residual distribution is similar to white noise and any correlations between past and future outputs are well-modelled. There are some slight cross-correlations between residuals and pump inputs, indicating additional dynamics that our simplified model structure may not be able to capture. However in practice this did not significantly impact closed-loop control performance.

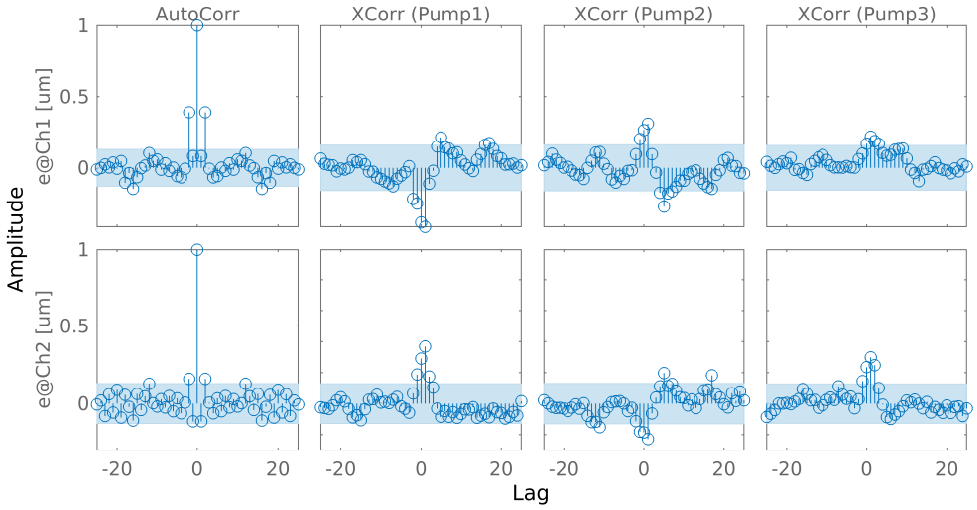


Figure 4.7: Model residual correlation with 99% confidence intervals

Online Parameter Estimation

From the state-space model above, the parameterized estimation model becomes (from Appendix A):

$$\begin{bmatrix} \hat{z}_{1,k} \\ \hat{z}_{2,k} \end{bmatrix} = \begin{bmatrix} x_{1,k} \\ x_{2,k} \end{bmatrix} - \begin{bmatrix} x_{1,k-1} \\ x_{2,k-1} \end{bmatrix} = \begin{bmatrix} -x_{1,k-1} & & \\ u_{1,k-1} & & \\ -u_{2,k-1} & & \\ -u_{3,k-1} & & \\ & -x_{2,k-1} & \\ & -u_{1,k-1} & \\ & u_{2,k-1} & \\ & -u_{3,k-1} & \end{bmatrix}^T \begin{bmatrix} \hat{\tau}_1 \\ \hat{b}_{11} \\ \hat{b}_{12} \\ \hat{b}_{13} \\ \hat{\tau}_2 \\ \hat{b}_{21} \\ \hat{b}_{22} \\ \hat{b}_{23} \end{bmatrix}$$

Results of online parameter estimation on the parameterized system are shown in Figure 4.8. All parameters reached near converged values in under 10 seconds, with parameter projection ensuring negative parameters were immediately corrected.

Output estimation error $\epsilon = y - \hat{y}$ (Figure 4.9) also drops to acceptable values after roughly 15 seconds.

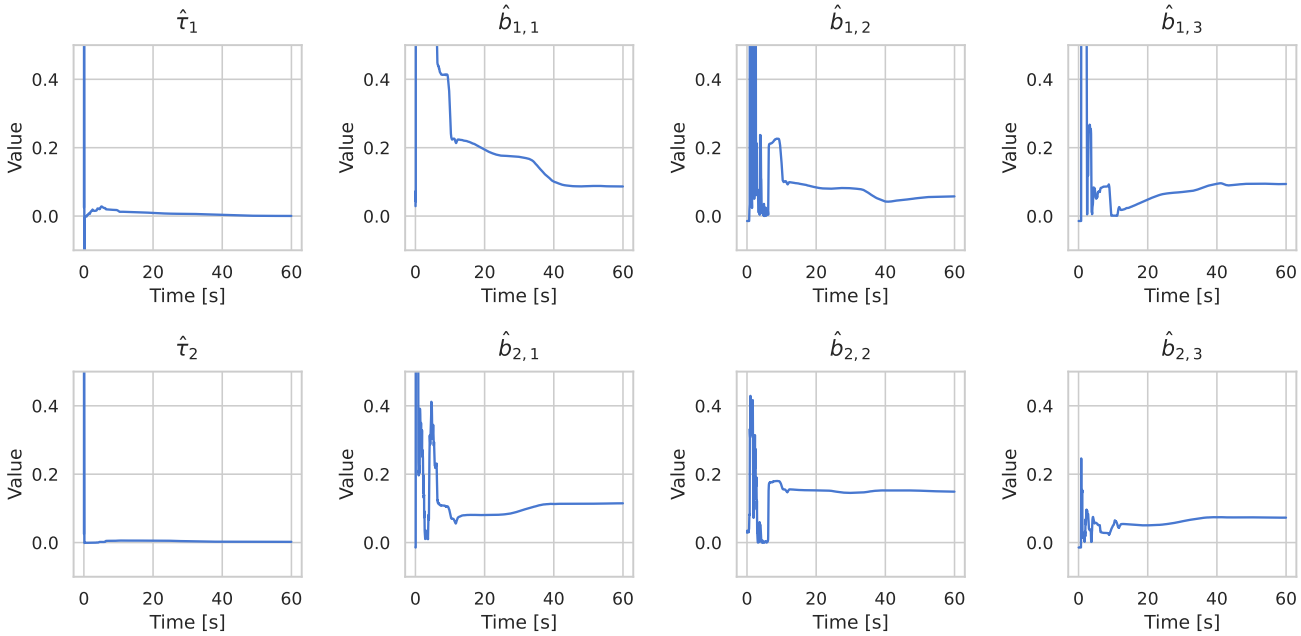


Figure 4.8: Convergence of online parameter estimates

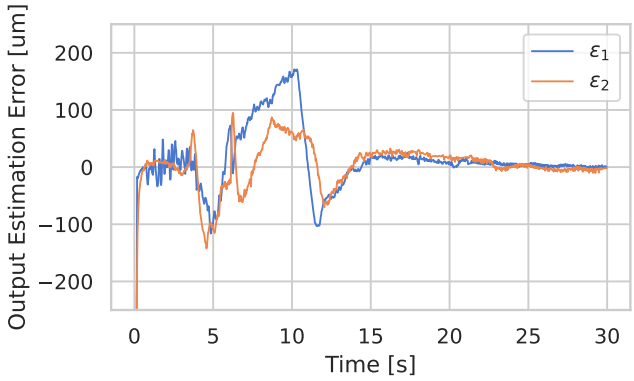


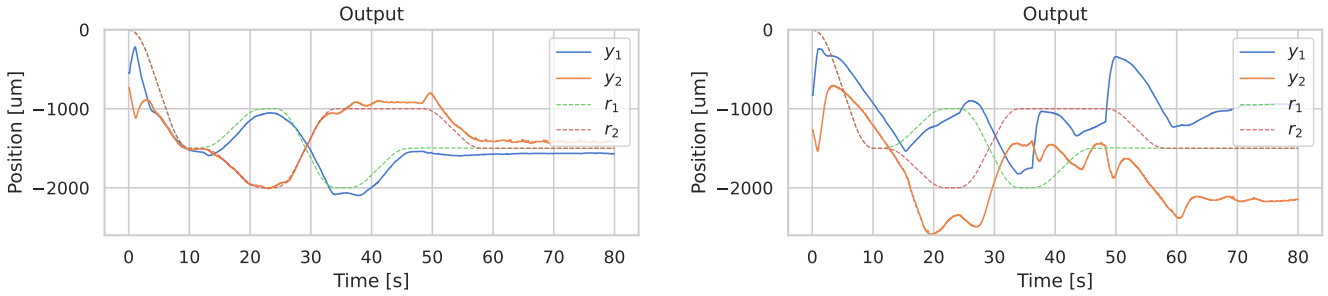
Figure 4.9: Output estimation error

4.4.2 Droplet Trajectory Tracking

Closed-loop droplet trajectory tracking performance of both controllers are shown in Figures 4.10 and 4.11. Adaptive MPC is given an additional 10 seconds before the trajectory

is performed for parameter estimates to converge.

Non-adaptive MPC is able to perform droplet trajectory tracking under normal conditions (Figure 4.10a), although some steady-state error persists due to the lack of integral action in the controller. However, a simulated 4x reduction in channel 3 pump pressure u_3 significantly degrades tracking performance, as the MPC's predictions of u_3's effect on droplet position become highly inaccurate (Figure 4.10b).



(a) No changes in system dynamics

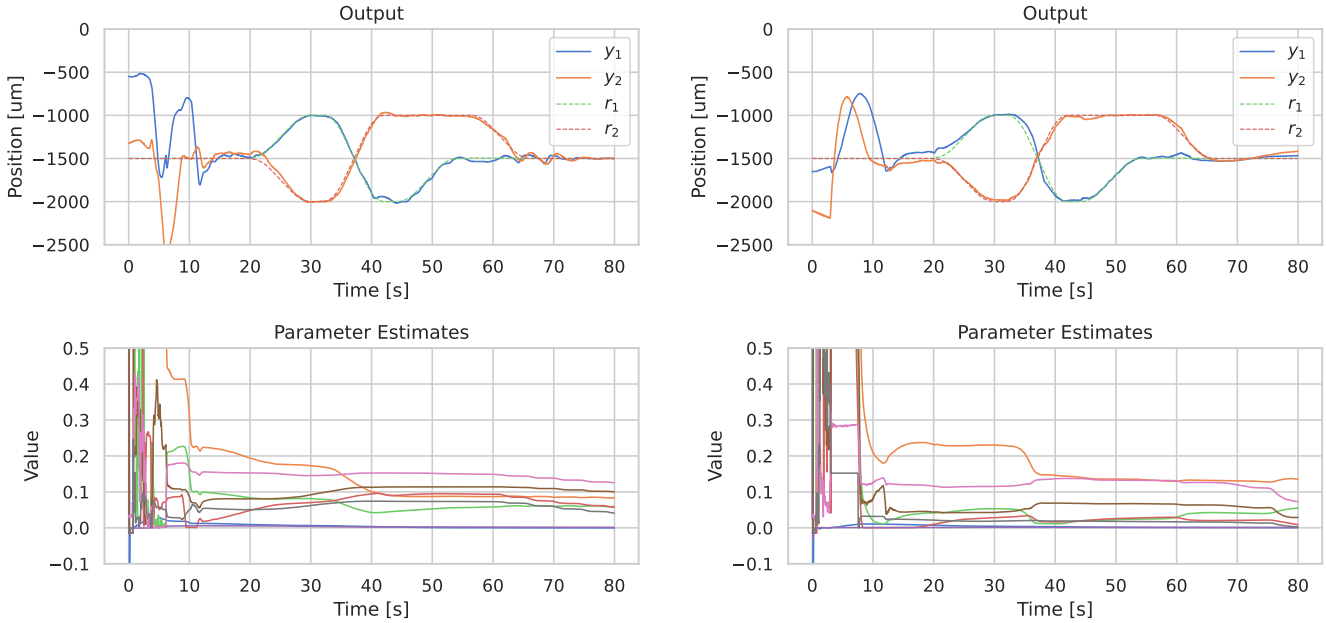
(b) Simulated 4x reduction in u_3

Figure 4.10: Non-adaptive MPC trajectory tracking of droplet interface position

On the other hand, once parameter estimates converge, adaptive MPC is able to smoothly track output trajectories with almost zero steady-state error, even in the absence of integral action (Figure 4.11a). When a reduction in u_3 is applied, parameter estimates are updated to account for the increase in control effort required on channel 3, producing almost identical tracking performance (Figure 4.11b).

4.5 Summary

In this chapter we designed an adaptive law based on RLS with parameter projection and used it to implement an adaptive MPC controller for droplet trajectory tracking. Despite using the same model structure and controller design as linear MPC, the addition of online parameter estimation results in significant improvement in droplet trajectory tracking performance and reduction in steady-state error without the need for integral action. Using RLS, adaptive MPC is able to learn the current model parameters faster and more accurately compared to offline techniques, and update them as conditions change. Automating the parameter estimation process and improving robustness to changes in system



(a) No changes in system dynamics

(b) Simulated 4x reduction in u_3

Figure 4.11: Adaptive MPC trajectory tracking of droplet interface position

dynamics will improve the practicality and accessibility of the proposed DMP platform for end-users.

Chapter 5

Development of an Automated Droplet Manipulation Software Workflow

5.1 Introduction

Software is crucial in the automation of advanced scientific workflows, enabling researchers to perform experiments and analyses accurately and repeatably with minimal manual intervention. A general definition of a workflow is "a network of tasks with rules that determine the (partial) order in which the tasks should be performed" [66]. In software workflows, there is often a tradeoff (Figure 5.1) between simplicity (accessibility to a wide audience) and generality (applicability to a wide variety of use-cases).

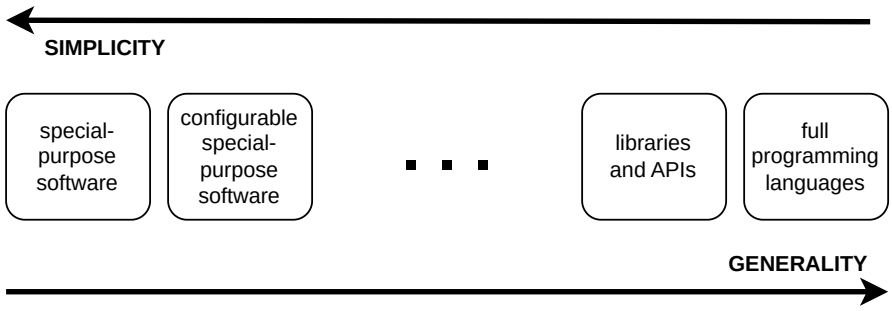


Figure 5.1: Simplicity vs generality in software environments [5]

A software wizard which guides the user through a fixed sequence of simple tasks is easy to use but hard to generalize. A full programming language allows much more flexible workflows, but requires the user to gain proficiency in that language to realize any gains in productivity. Balancing simplicity and generality in software workflows requires us to understand how much complexity should be abstracted away vs exposed to the user. Exposing more details of the workflow gives users more power and flexibility but also requires a deeper level of understanding — with great power comes great responsibility.

From Lamprecht [5], we identify some key requirements of scientific software workflows:

- **Domain-specific abstraction** — implementation-specific and domain-specific aspects of the workflow are separated via a software abstraction layer, with only domain-specific details exposed to the user, commonly via a graphical user interface (GUI)
- **Workflow modelling** — a set of data flows (transformations of input data through relevant software services to produce meaningful output data) and control flows (structures that define the order and conditions in which tasks are executed) are defined to characterize each workflow, allowing highly complex workflows to be modelled from simple reusable components
- **Workflow validation/verification** — users can validate that the modelled workflow is physically realizable and produces the desired outputs at model-time (via unit testing, simulation, etc), and verify that the modelled workflow is correctly executed at run-time (via fault detection, error handling, real-time feedback, etc) on the physical system

Previous work on the proposed DMP platform have seen workflows shift from a fixed flowchart of tasks manually performed by the user for each manipulation [53] to a library of instruction templates that can be arbitrarily combined and automatically executed [21]. This shift from simple to general workflows is important for the proposed platform to support a wider range of real-world droplet microfluidics applications [23]. However, many aspects of previous DMP workflows still present practicality and accessibility barriers to potential users. Existing workflows require ad-hoc open-loop procedures to handle nonlinearities (e.g. necking in droplet generation) or discontinuities (e.g. droplet leaving/entering a ROI), as well as significant manual tuning and calibration to ensure successful manipulations [4, 20, 21]. Moreover, due to inadequate workflow modelling, users cannot validate DMP workflows before running them on the physical system, increasing the likelihood of failed manipulations which can cost time and material.

This chapter addresses the identified challenges in workflow automation and modelling to develop a generic model-based software workflow for automated droplet manipulation.

A program generation method and associated software platform is developed to convert a high-level user-defined workflow into sequences of low-level droplet manipulation objectives. Using a model-driven approach, a supervisor is designed to generate droplet trajectories to reach each objective. To generate smooth trajectories between potentially discontinuous manipulation objectives (e.g. multiple droplet generation, droplet storage/retrieval), a new trajectory planning method is developed which removes the need for ad-hoc open-loop commands. AMPC (Chapter 4) is applied to ensure trajectory tracking with minimal error, with online parameter estimation eliminating the need for manual calibration and model parameter tuning.

We evaluate the DMP workflow for automating the fundamental manipulation — droplet generation, showing that each stage of droplet generation is automatically executed as defined by the user, with close correlation between simulated and experimental performance.

5.2 Supervisor Modelling

Figure 5.2 shows an overview of supervisor components in context.

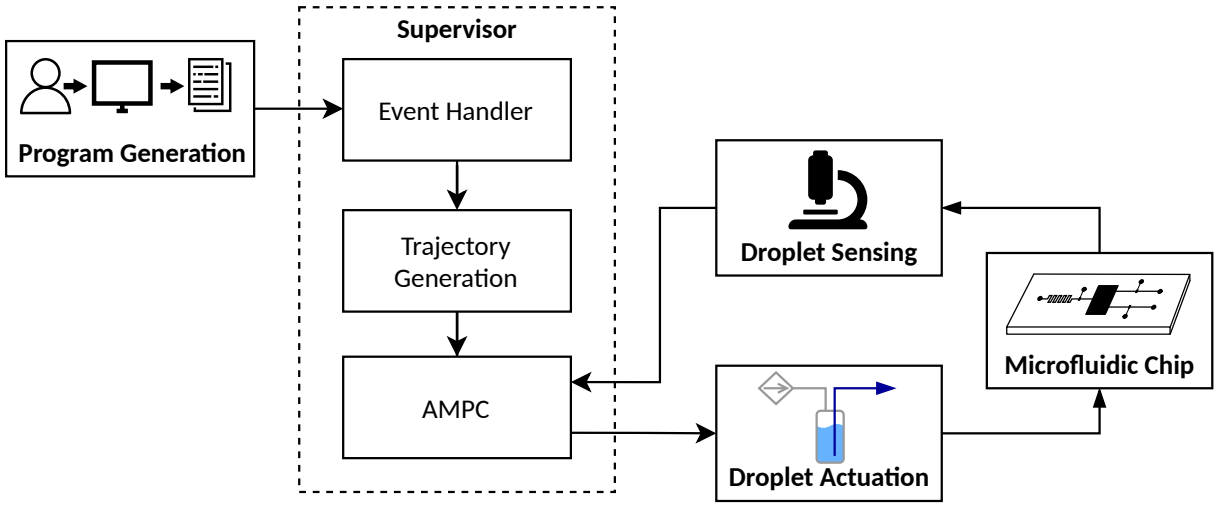


Figure 5.2: Simplified supervisor architecture

Program generation (Section 5.4) converts a user-designed workflow to a sequence of droplet manipulation objectives/events to the supervisor. The supervisor implements objective/event handling as a state machine that passes the current event to the trajectory

generator, and requests new events when the current one is completed. Trajectory generation produces a smooth trajectory between each objective, and the optimal control law (AMPC) minimizes trajectory tracking error.

The supervisor is designed using a model-driven approach, in which a model is developed based on the specified architecture and validated in simulation (Simulink). To simulate arbitrary droplet manipulation sequences, the RLC microchannel model (detailed in Chapter 4) is augmented with a simplified model of junction behavior (based on zero-crossing detection with hysteresis) dictating how droplet interface measurements pass through junctions to other channels. Code generation tools (Simulink Coder) are used to produce a matching C++ implementation based on the model which can be deployed to hardware. A simplified overview of the supervisor simulation model is provided in Figure 5.3.

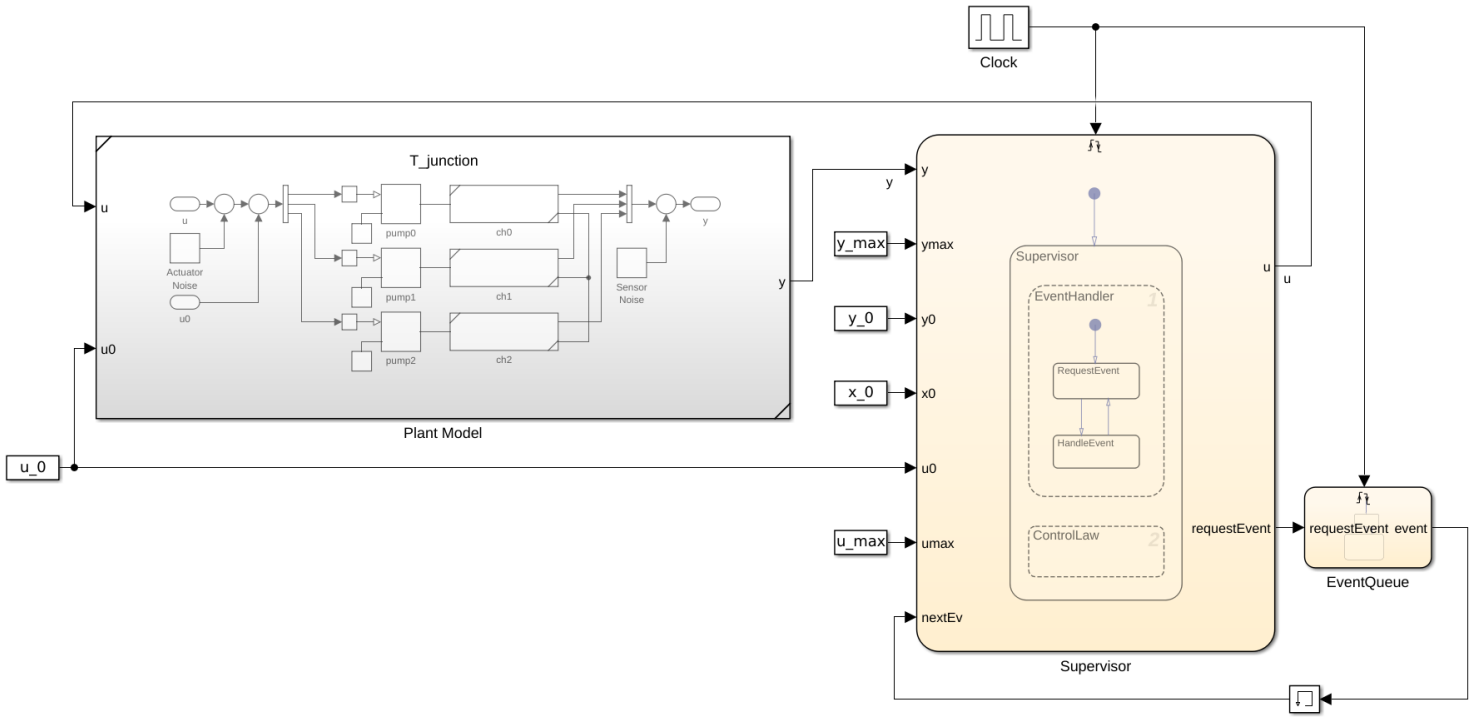


Figure 5.3: High level overview of supervisor simulation environment

This approach drastically reduces the number of development iterations, as well as the lines of code written by hand during development (over 3x reduction relative to RoboDrop [4]), minimizing the likelihood of error. Using this approach, we can rigorously define how the supervisor interfaces with external software (i.e. type and size of each input/output).

This allows the supervisor to be safely integrated into any user-facing software applications (see Appendix C), and signals within the model to be easily monitored in real-time to verify proper operation.

The accompanying source files can be found at https://github.com/KevinHQChen/autoDMP.

5.3 Trajectory Planning

This section describes a general framework to generate smooth trajectories for droplet manipulations in the presence of discontinuous droplet interface measurements.

5.3.1 Trajectory Generation

Given desired droplet interface locations from program generation (Section 5.4) and current droplet interface locations from droplet sensing, a smooth trajectory can be defined to drive droplets to desired positions. For simplicity a trapezoidal velocity profile (Figure 5.4) is defined in MATLAB (Robotics System Toolbox) to provide constraints on maximum velocity and acceleration in the trajectory.

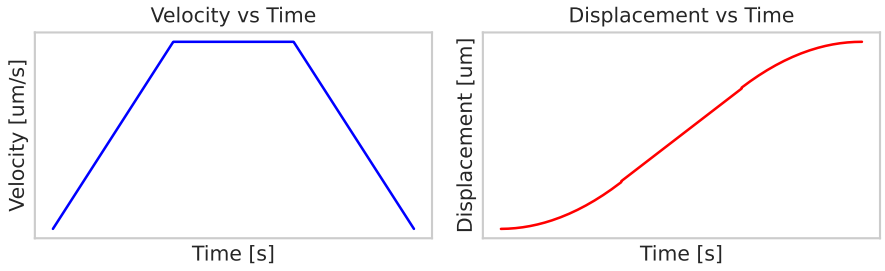


Figure 5.4: Trapezoidal velocity profile and corresponding displacement profile

5.3.2 Interface Measurement Framework

To fully control droplet manipulations, ideally the entire pressure/flow field throughout the microfluidic chip can be measured for feedback. In practice, only visible droplet interfaces provide any clue as to the current state of the system, and channels without visible

interfaces are completely unobservable. Previous work handled this by defining a set of state-space models corresponding to all combinations of observable channels, and used gain-scheduling control to switch between them automatically as needed [4, 20, 21]. However, this causes discontinuous jumps when droplet interfaces cross junctions or leave the region of interest, necessitating additional compensation techniques or special open-loop commands to smooth these transitions.

Here a new interface measurement framework is developed that attempts to maintain measurement continuity as much as possible. First, we define two types of measurements for each channel, direct and inferred. Direct measurements in any channel can be made whenever an interface is present in that channel, denoted as negative in our coordinate system (Figure 5.5). Once the interface crosses a junction, it is no longer directly measurable in the original channel, but we can infer an equivalent measurement (denoted as positive in our coordinate system) based on measurements in all other channels connected to that junction. In a single-junction, n-channel geometry, this framework allows a single interface to be represented as a continuous (direct or inferred) measurement vector $\mathbf{y} \in \mathbb{R}_n$ regardless of which channel it happens to be in.

Using this framework, any droplet manipulation that doesn't create new interfaces can be modelled as a series of continuous trajectories, including droplet move, hold, and sort. An example droplet sorting trajectory is simulated in Figure 5.5, showing smooth junction crossings (i.e. zero-crossings).

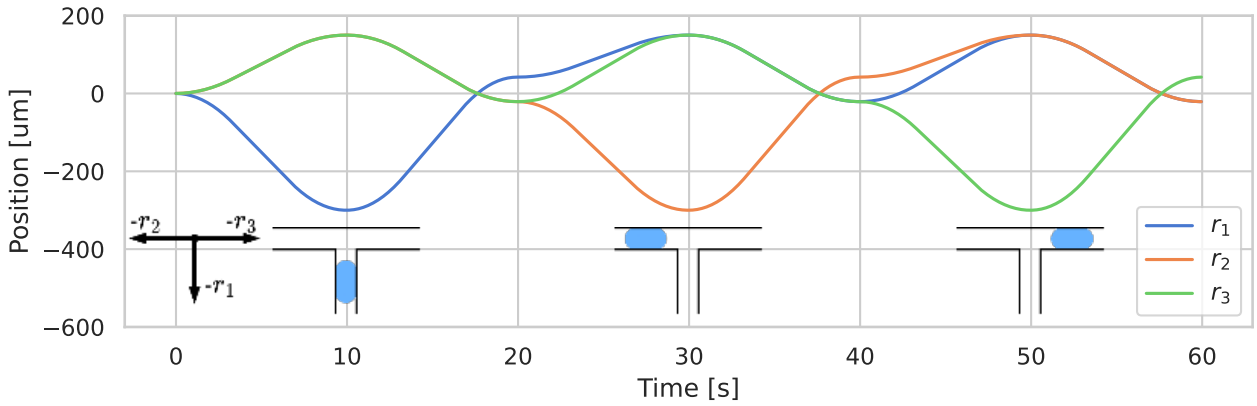


Figure 5.5: Simulated droplet sorting trajectory in T-junction

To capture manipulations like droplet generation, splitting, or merging where interfaces are created/combined (causing discontinuous measurement offsets), we define a primary

and secondary measurement vector $\mathbf{y}_\alpha = \begin{bmatrix} y_{1\alpha} & y_{2\alpha} & \dots & y_{n\alpha} \end{bmatrix}^T$ and $\mathbf{y}_\beta = \begin{bmatrix} y_{1\beta} & y_{2\beta} & \dots & y_{n\beta} \end{bmatrix}^T$ in $\mathbb{R}_n$ that each represents a continuous interface measurement. Initially both vectors are identical, but whenever any component of one vector crosses the junction, we reset the other vector to the interface closest to the junction. Two sets of measurement vectors allow us to fully characterize the motion of all individual droplet manipulations, providing the maximum amount of information to the supervisor at all times. As an example, primary and secondary direct measurements for different stages of droplet generation are shown in Figure 5.6.

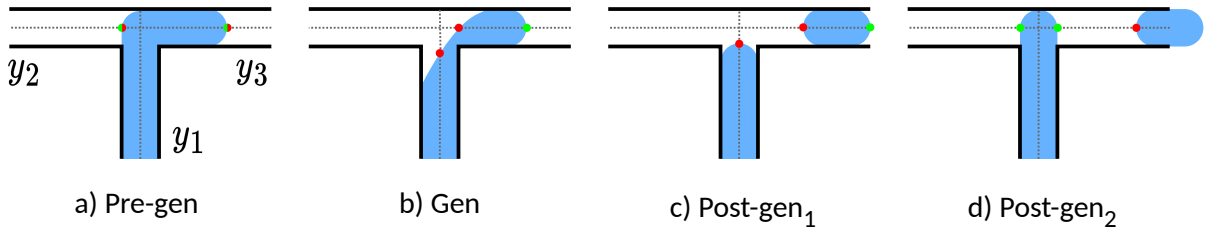


Figure 5.6: droplet generation waypoints for $\mathbf{y}_\alpha$ (green) and $\mathbf{y}_\beta$ (red)

The full implementation of the droplet measurement framework is detailed in Appendix B.

5.3.3 Online Modification of Control Objectives

Although $\mathbf{y}_\alpha$ and $\mathbf{y}_\beta$ fully characterizes individual droplet manipulations, we need a way to provide the most relevant measurement for the current control objective. Since the two measurement vectors will diverge over longer manipulation sequences, abruptly switching measurements will appear to the controller as a large output disturbance which it will try to reject, potentially causing significant overshoot and oscillation.

Wong's approach [4] relied on applying a measurement offset to remove any discontinuities from the controller's perspective. However, this approach is impractical for longer and more complex sequences of droplet manipulations as the offsets can become very large and difficult to keep track of. Instead, the new measurement framework is used to provide both primary and secondary measurements to the controller at all times, but at any time only weighing the most relevant measurements in the MPC cost function by tuning w_y (Chapter 4).

Consider the waypoints involved in droplet generation (Figure 5.6). After the droplet is generated (i.e. Gen to Post-gen1), the only direct measurement in $\mathbf{y}_\alpha$ is $y_{3\alpha}$, which is just a fixed offset from $y_{3\beta}$. Clearly, the primary measurement is no longer providing enough information for the controller. Thus to ensure the most relevant outputs are being weighted, the MPC can smoothly decrease $w_{y,\alpha}$ and increase $w_{y,\beta}$ as soon as the droplet is generated. This can be applied to any droplet manipulation where there are multiple interfaces of interest in a single channel. For example, the overall reference trajectory with output weight modification for droplet generation is shown in Figure 5.7.

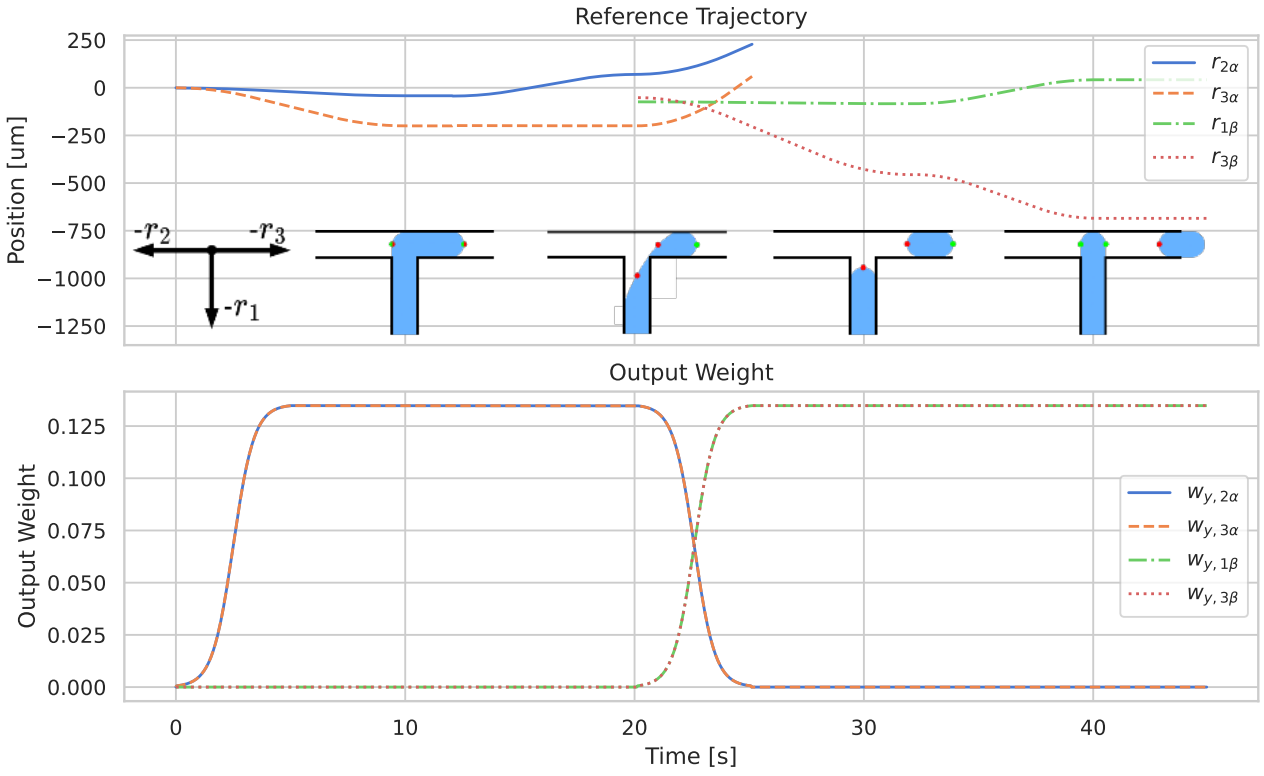


Figure 5.7: Simulated droplet generation trajectory in T-junction, with output weight modification using sigmoid function

5.4 Program Generation

Using the new measurement framework, we can break down complex trajectories into a set of simple objectives/waypoints that can be executed without opening the control loop. Similar to Zablotny's approach [21], each objective is encoded in a standardized instruction containing the following components:

- $\mathbf{r}_\alpha \in \mathbb{R}_n$: Primary waypoint vector
- $\mathbf{r}_\beta \in \mathbb{R}_n$: Secondary waypoint vector
- T_{pre}: Initial hold time
- T_{move}: Move time
- T_{post}: Final hold time

The waypoint vectors hold the desired droplet interface positions (any element set to 0 means there is no objective for that channel — i.e. unweighted in MPC cost function). Additionally, we specify a move time between waypoints which the trajectory generator can use to solve for the constant acceleration and max velocity required to reach the target waypoint. A hold time is also defined to hold droplet interfaces in position for the specified amount of time before or after the instruction. Arbitrary droplet manipulation sequences can be encoded as lists of instructions (i.e. programs) to be fed to the supervisor.

A single instruction can capture any manipulation that does not produce new interfaces (e.g. move, hold, mix, sort). For example, the droplet sorting trajectory in Figure 5.5 is defined by the following program:

$\mathbf{r}_\alpha$ $[\mu m]$	$\mathbf{r}_\beta$ $[\mu m]$	T_{pre} $[s]$	T_{move} $[s]$	T_{post} $[s]$
$(-300, 0, 0)$	$(0, 0, 0)$	0	10	0
$(42, 0, 0)$	$(0, 0, 0)$	0	10	0
$(0, -300, 0)$	$0, 0, 0)$	0	10	0
$(0, 42, 0)$	$0, 0, 0)$	0	10	0
$(0, 0, -300)$	$0, 0, 0)$	0	10	0
$(0, 0, 42)$	$0, 0, 0)$	0	10	0

Table 5.1: Example droplet sort program corresponding to Figure 5.5

In general, we can describe and generate these instructions using higher level functions with syntax that approaches natural language (e.g. Python):

```
# move interface in channel 1 to dest in tau seconds
move(ch=1, destination=dest, moveTime=tau, hold=false)
# move interfaces in channels 2 and 3 to dest1 and dest2 in tau seconds
move(ch=[2, 3], destination=[dest1, dest2], moveTime=tau, hold=false)
```

For operations that change the total number of interfaces (e.g. generation, merge, split), we can define sets of instructions to be executed in order. For example, the droplet generation trajectory in Figure 5.7 is defined by the following instruction set:

$\mathbf{r}_\alpha$ $[\mu m]$	$\mathbf{r}_\beta$ $[\mu m]$	T_{pre} $[s]$	T_{move} $[s]$	T_{post} $[s]$
$(0, -42, -200)$	$(0, 0, 0)$	0	10	2
$(0, 84, -200)$	$(0, 0, 0)$	0	8	0
$(0, 0, 0)$	$(-84, 0, -450)$	0	12	0
$(0, 0, 0)$	$(42, 0, -675)$	0	8	0

Table 5.2: Example droplet generation program corresponding to Figure 5.7

More generally, we can define a parameterized droplet generation instruction set (detailed derivation in Appendix D):

$\mathbf{r}_\alpha$	$\mathbf{r}_\beta$	T_{pre}	T_{move}	T_{post}
$(0, -w_{ch}/2, -L_d)$	$(0, 0, 0)$	$\tau_{\text{pre pre-gen}}$	$\tau_{\text{pre-gen}}$	$\tau_{\text{post pre-gen}}$
$(0, \delta_{neck}, -L_d)$	$(0, 0, 0)$	$\tau_{\text{pre gen}}$	τ_{gen}	$\tau_{\text{post gen}}$
$(0, 0, 0)$	$(\delta_{junc}, 0, -\delta_{\text{drop}})$	$\tau_{\text{pre post-gen}}$	$\tau_{\text{post-gen}}$	$\tau_{\text{post post-gen}}$

Table 5.3: Generic T-junction droplet generation instruction set

with the following parameters:

- L_d - controls length of generated droplet
- w_{ch} - channel width
- δ_{neck} - additional displacement (from junction) required to completely detach the droplet neck
- δ_{junc} - additional displacement (from junction) required to completely plug the junction
- δ_{drop} - controls spacing between generated droplet and next droplet

w_{ch}, δ_{neck}, and δ_{junc} can be calculated from geometry with relatively minor tuning, so the only necessary parameters for user-tuning are L_d and δ_{drop}. This also allows us to easily describe and generate these instructions sets using a higher level language (e.g. Python):

```
# generate 3 droplets sourced from channel 1 to channel 3 with lengths L_d1, L_d2, L_d3, spacing delta_d
generate(qty=3, src_ch=1, dest_ch=3, length=[L_d1, L_d2, L_d3], delta=delta_d)
```

Similar instruction sets can be derived for merge, split, etc, but due to time constraints will not be discussed.

Combining individual instructions and instruction sets allows custom user-defined programs that realize complex droplet manipulation workflows.

5.5 User Interface

A custom user interface is developed to allow users to set up and execute droplet manipulation programs. The typical layout is shown in Figure 5.8, with setup steps highlighted.

The setup process involves the following:

1. Image capture and image processing
   (a) Select camera source in config file (`config/setup.toml`) — currently supported sources: generic webcam, generic USBcam, Andor Zyla 5.5
   (b) Select junction(s) and channel(s) in *Image Capture Window* (or click *Load Im-Proc Config* to load existing config from file)
2. Chip priming and interface balancing
   (a) Select pump type in config file (`config/setup.toml`) — currently supported pumps: Bartels mp6, Fluigent MFCS
   (b) Manually drive continuous phase channel through the microfluidic chip via *Pump Setup Window* until fully primed
   (c) Similarly, manually drive dispersed phase channel through the microfluidic chip until it is visible in *Image Capture Window*
3. Upload user program and run supervisory controller through *Ctrl Setup Window*

Furthermore, an embedded *Console Window* and *Real Time Plot Window* show the current status, warnings/errors, and hardware faults.

Figure 5.8: Typical user interface layout, highlighting windows for each step of the setup process - 1) image processing 2) chip priming and balancing 3) upload user program

5.6 Experimental Evaluation

In this section, we simulate and experimentally verify an automated droplet generation workflow on a T-junction microfluidic chip (setup identical to Chapter 4).

The workflow is defined by the following program:

$\mathbf{r}_\alpha$ $[\mu m]$	$\mathbf{r}_\beta$ $[\mu m]$	T_{pre} $[s]$	T_{move} $[s]$	T_{post} $[s]$
$(-1500, 0, 0)$	$(0, 0, 0)$	0	5	2
$(-500, 0, 0)$	$(0, 0, 0)$	0	5	2
$(-1500, 0, 0)$	$(0, 0, 0)$	0	5	2
$(210, 0, 0)$	$(0, 0, 0)$	0	7	0
$(0, -210, -1000)$	$(0, 0, 0)$	0	10	2
$(0, 350, -1000)$	$(0, 0, 0)$	0	7	0
$(0, 0, 0)$	$(-420, 0, -2000)$	0	12	0
$(0, 0, 0)$	$(210, 0, -3200)$	0	7	0

Table 5.4: Droplet generation evaluation program

The initial single-channel instructions provide excitation for online parameter estimation by the AMPC and sets up the interface for droplet generation. The remaining multi-channel instructions perform the droplet generation.

The program execution is validated on the simulated supervisor model, then uploaded via the user interface to the software application (Appendix C) to be executed on the DMP platform. The resulting simulated vs experimental droplet trajectory for a typical program execution is shown in Figure 5.9, with snapshots of each multi-channel instruction shown in Figure 5.10.

The AMPC is given 10 seconds of excitation to identify model parameters, and we observe trajectory tracking error decrease significantly afterwards. On trajectory tracking, experimental performance deviates from simulated performance at junction crossings (at $\sim$45 to 50 [s]), which is expected as the plant model does not simulate the additional non-linear dynamics. Notably, the AMPC was able to maintain stability and produced adequate manipulation outcomes in both simulation and experiment.

To evaluate experimental droplet generation accuracy and precision, the program was run repeatedly for different droplet lengths. The resulting droplet length distribution is shown in Figure 5.11.

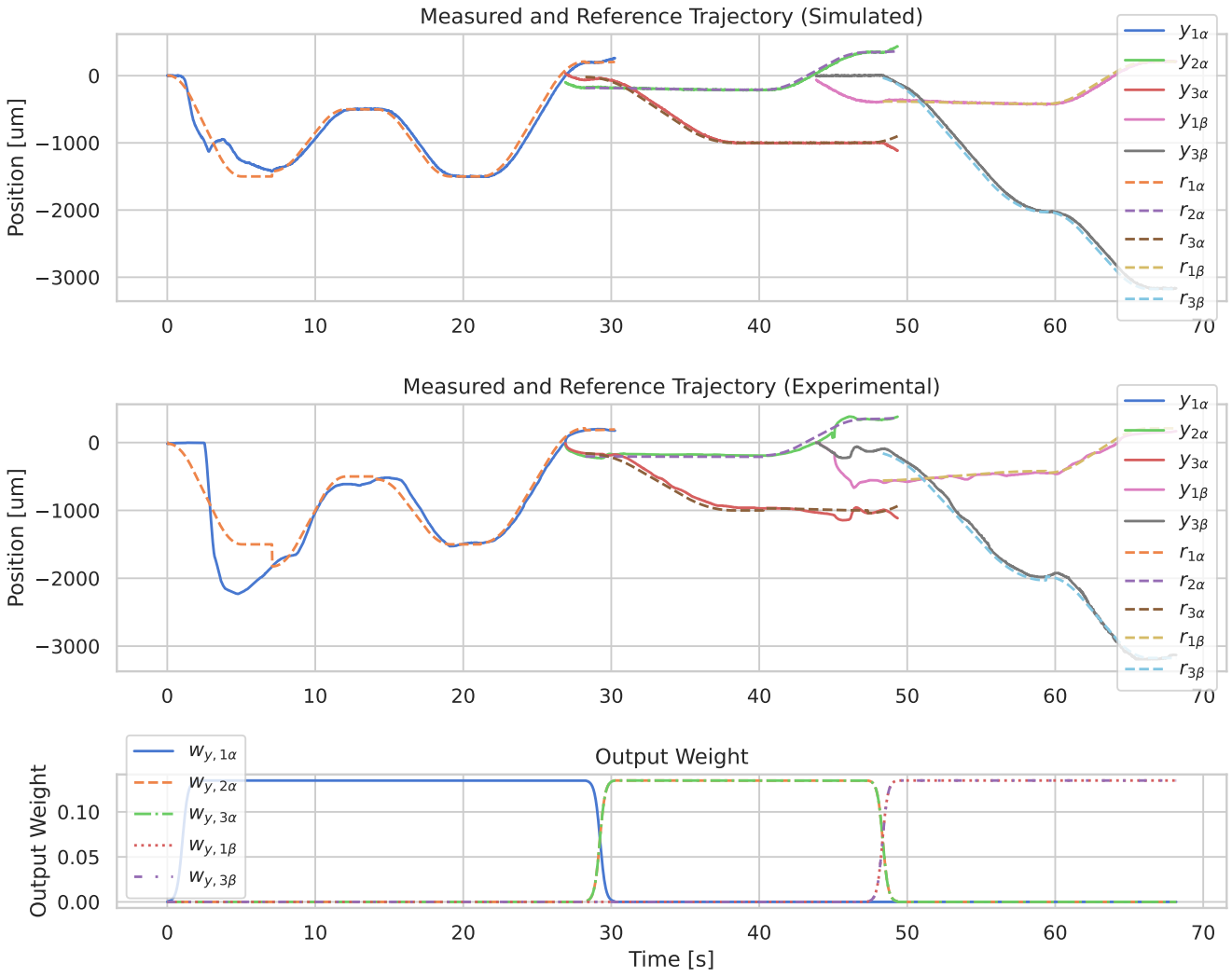


Figure 5.9: Simulated vs experimental trajectory tracking for automated droplet generation

Generated droplets achieve under ±5% error relative to calibrated reference, with slightly higher error at lower droplet lengths. The minimum achievable droplet length under fully automatic control was 460 [μm], or a droplet length-to-width aspect ratio of ∼2.3:1 with 200 [μm] channel width (compared to ∼1.5:1 for semi-automated methods [20]). The monodispersity is calculated for each cluster of generated droplets via coefficient of variation, with an average value of 2.12%, similar to the 2% value reported using passive

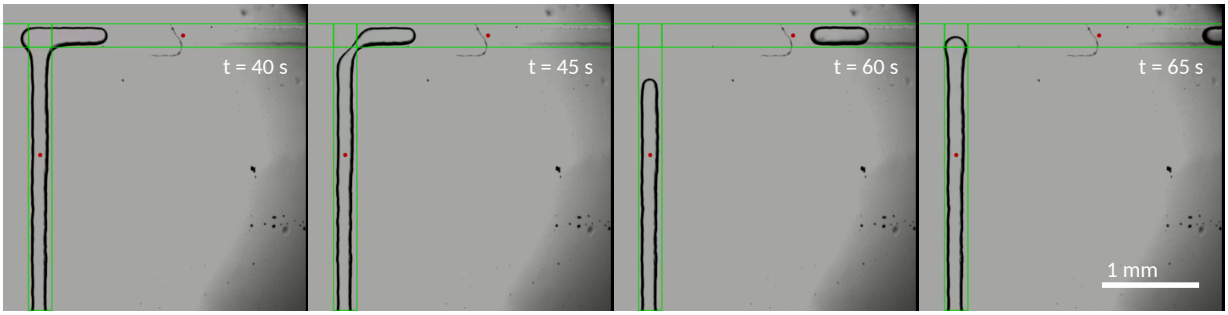


Figure 5.10: Experimental droplet generation waypoints corresponding to instruction set in Table 5.4 and trajectory in Figure 5.9

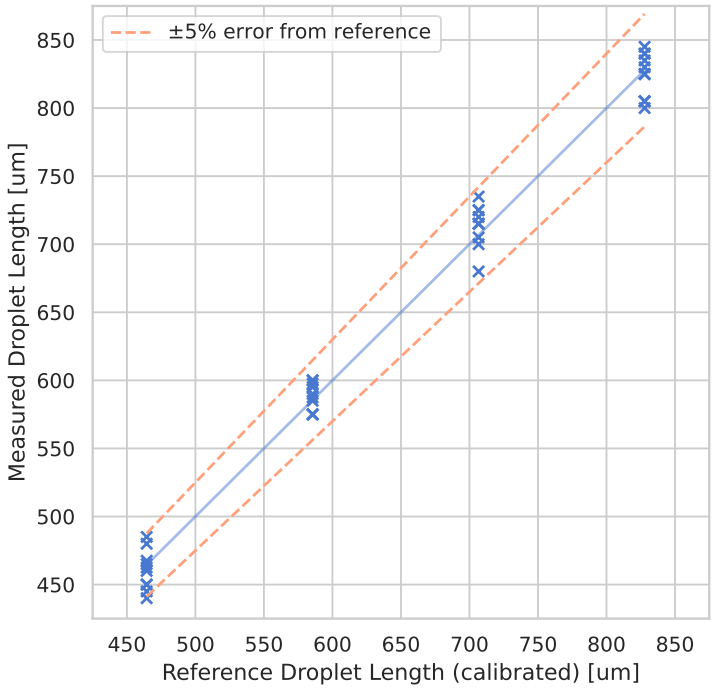


Figure 5.11: Droplet length distribution using automatic droplet generation workflow, calibrated via linear regression with best-fit line $y = 1.21x - 383$

droplet microfluidics techniques [67].

Some of these limitations can be attributed to the simplistic control model used by the AMPC, meaning that transient disturbances (seen in the experimental trajectory in Figure 5.9) cannot be predicted and rejected as effectively.

The vision system also contributes slightly to these limitations. The resolution of the combined microscope and camera system is 5 [μm] per pixel. Assuming optimistically that measurements are pixel-accurate, the worst-case error of ± 0.5 pixels contributes up to $\pm 0.5\%$ relative error at the minimum achievable droplet length ($460\mu m$).

Another contributing factor is the chip fabrication method. Due to time constraints, T-junction PDMS chips were fabricated using 3D-printed masters (Formlabs), which limited the achievable junction quality compared to traditional soft lithography techniques. However, results show that cheaper and lower quality chips can still achieve precision comparable to previous semi-automated methods [20] (in the shared range of droplet aspect ratios) without the significant calibration and model parameter tuning required in previous fully automated methods [21].

To demonstrate the flexibility of the proposed workflow, we extend the droplet generation program to generate variable length droplet trains (full instruction set and video provided in Appendix D) with the following pseudocode:

```
# generate 3 droplets sourced from channel 1 to channel 3
# with calibrated lengths 640 [um], 700 [um], 825 [um], and 4200 [um] spacing
generate(qty=3, src_ch=1, dest_ch=3, length=[640, 700, 825], delta=4200)
```

5.7 Summary

We have leveraged a new droplet measurement framework along with AMPC to enable smooth control of droplet interfaces when performing automated droplet manipulation workflows. Software modelling and program generation techniques resulted in improved robustness and practicality in longer instruction sequences, and will be crucial for future work in scaling up DMP workflows to more complex applications.

Due to time constraints, move times were not optimized and conservative values were selected to ensure stability, resulting in slow manipulation throughput. But much lower move times are possible by using higher quality chips, smaller microchannel widths, and higher order control models. Preliminary evaluation on circular cross-section glass microfluidic chips (Dolomite) showed more ideal behavior (closer to the control model used),

potentially allowing much faster droplet manipulations with the existing control system. Moreover, experiments run on glass chips over a period of multiple months show minimal channel surface degradation (with proper cleaning/storage), as opposed to PDMS whose channel surface degrades over a period of days to weeks.

Other soft parameters can also be further optimized, such as the AMPC forgetting factor, which is related to how fast system dynamics change. Most applications of adaptive control are valid when system dynamics change slowly, allowing the parameter estimator to forget older data at a constant rate specified by the forgetting factor. However for applications like droplet generation or splitting, there is a large change in dynamics before and after the operation. One possible solution is variable forgetting factor (VFF), i.e. updating the forgetting factor to discount older data faster if the system is highly volatile, and slower if the system is near steady state.

Chapter 6

Evaluating Piezoelectric Micropumps for Automated Droplet Manipulation

6.1 Overview

Visual feedback control is a promising technique for modular, general-purpose droplet microfluidics platforms (DMP), allowing arbitrary manipulation of individual droplets. To facilitate arbitrary droplet manipulation, droplet actuation methods must be fast and responsive. The most performant methods use pressure-driven flow control (more simply referred to as pressure pumps). This typically involves modulating a fixed external input pressure by applying pressure feedback to high precision proportional valves (Figure 6.1 top), pressurizing sample reservoirs and driving droplet motion in the microfluidic chip through pressure-driven flow.

However, the requirement for regulated external pressure and high precision valve control contributes to high cost, complexity, and bulkiness, limiting their practical application in modular DMP's.

In this study, piezoelectric (PZT) diaphragm micropumps are applied to meet both the pressure source and control requirements for arbitrary droplet manipulation. We evaluate this pumping method as it is well-studied and widely available, with a balance of fast response, simple structure, and reliability.

We show that piezoelectrics have a more predictable and linear pressure response that removes the need for pressure sensing (a necessity in conventional pressure pumps) for the purposes of automated droplet manipulation. The PZT pump is integrated into the

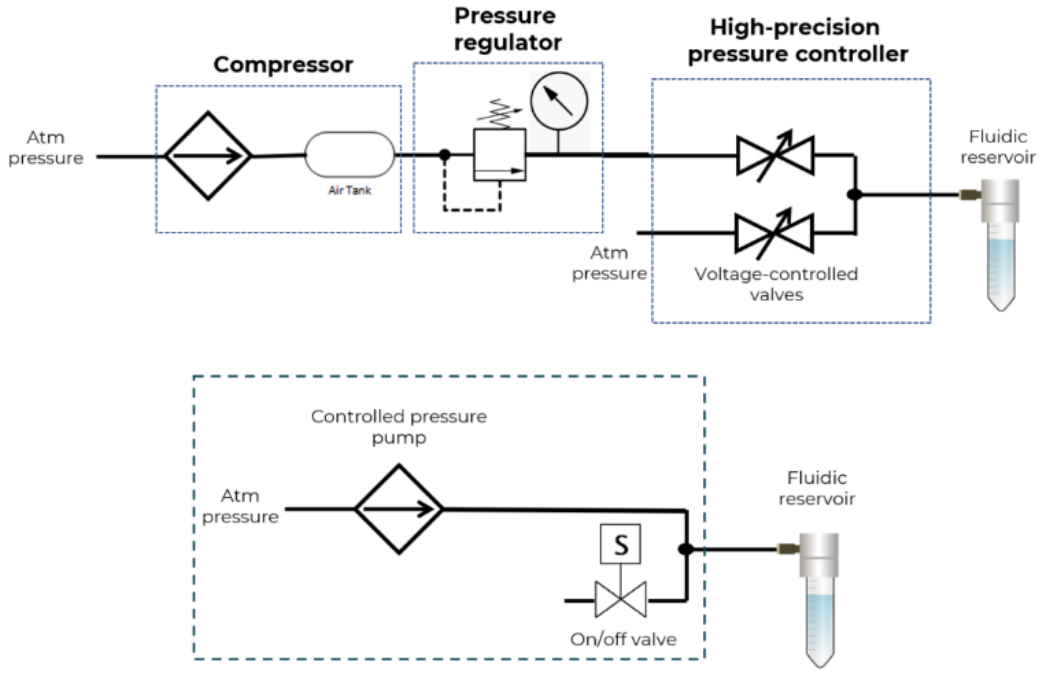


Figure 6.1: Conventional (top) vs portable (bottom) pressure control systems [6]

automated DMP workflow (Chapter 5), demonstrating adequate automated droplet manipulation accuracy and repeatability. The pump design is shown to be significantly more modular, portable, and cost-effective compared to conventional high performance pressure control systems. These findings suggest that PZT micropumps are a viable alternative to conventional pressure pumps for the proposed automated DMP platform.

6.2 Introduction

By combining pressure-driven flow control with vision-based feedback on a simple passive droplet generator architecture (T-junction), repeatable on-demand droplet manipulations (generation, trapping, merging, splitting, sorting) were demonstrated [52, 53]. Building on this work, semi-automated manipulations combining multiple intermediate steps were realized [20].

Visual-feedback-based droplet manipulation requires micropumps that actively and pre-

cisely control the pressure and flow field in a microchannel network. The most popular standalone micropumping systems include syringe pumps and pressure pumps [24]. Syringe pumps are simple, low-cost and relatively portable, but are hindered by slow response time and steady-state flow oscillations [68]. Energy-transfer-based pumping techniques (e.g. electro-osmotic, electrohyrodynamic) are another option, but for a variety of reasons (e.g. complex fabrication protocols, very high voltage requirements) are not yet mature enough for practical adoption [7]. Pressure pumps overcome these limitations by producing pressure-driven flow based on pressure feedback, utilizing a variety of valves and pressure regulators to modulate a fixed pressure source [69, 70]. Thus pressure pumps have been applied to visual-feedback-based droplet control [52, 53, 20] due to their high bandwidth, output range, and stability relative to common pumping techniques.

However, conventional pressure pumps that meet the requirements for visual-feedback-based droplet control are complex, nonportable and expensive, often requiring advanced pressure feedback mechanisms and external pressure sources [71, 69, 70].

More recently, a portable pressure pump design has been introduced (Figure 6.1 bottom) that claims comparable performance to conventional pressure pumps. Although little information is given on its cost, complexity, and actuation method, it highlights the increasing need for practical modular micropumps that are more easily integrated into existing microfludics systems.

The diaphragm-driven pump is one such pump architecture that requires much fewer components and generates pressure internally. Diaphragm-driven pumping is a well-studied technique that combines an oscillating diaphragm with valves to produce directed flow (Figure 6.2).

Valves can be actively controlled (allowing bidirectional flow), but most designs use passive check valves for simplicity (preventing backflow). A variety of diaphragm driving methods are available [7], but piezoelectrics have emerged as the most mature technique due to its ability to generate large deflections while maintaining a fast response time (Figure 6.3, Table 6.1).

PZT micropumps have previously been applied to active droplet sorting [72] and droplet generation [73, 74]. Pressure-feedback-based PID control has also been applied to PZT pumps for droplet generation [75]. Commercial PZT pumps have also been used to build smart, standalone pumping systems [76].

However, the PZT pumps used are often limited to unidirectional flow due to the use of passive check valves, which hinders the ability to change droplet direction in all channels, a requirement for performing arbitrary droplet manipulations. The proposed solution combines a PZT pump (Bartels mp6-gas+) and solenoid bleed valve to allow full

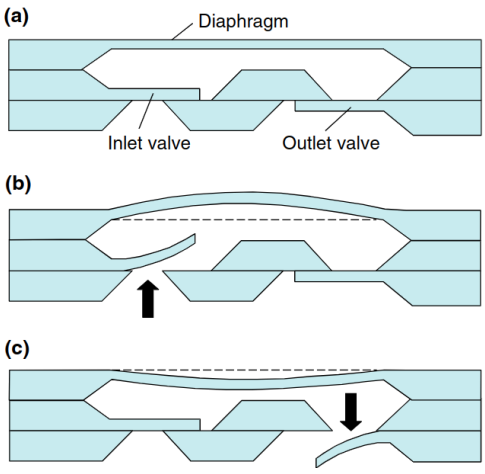


Figure 6.2: Diaphragm micropumping stages: a) undeflected, b) expansion, c) contraction [7]

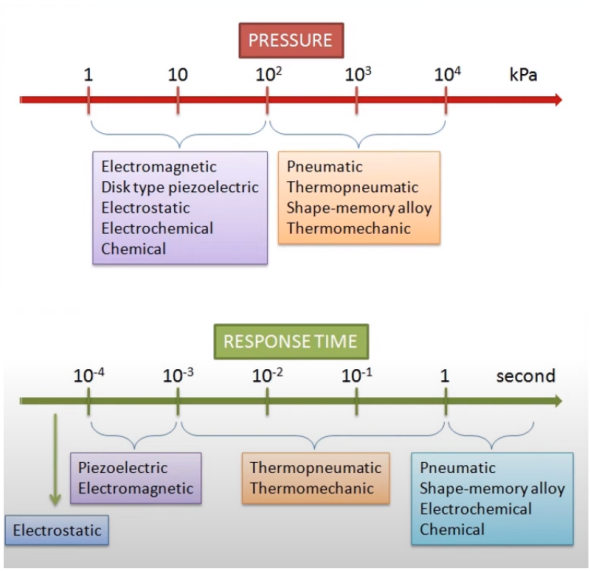


Figure 6.3: Pressure and response time scale for various micropump driving methods [8]

pressurization and depressurization of the chip, enabling arbitrary vision-based droplet control within each microchannel. Results show the proposed pump architecture is suitable for feedback-controlled droplet manipulation at a much lower price point compared to conventional high-performance pressure pump systems.

	Thermal	Electro-static	Electro-magnetic	Shape-memory	Piezoelectric-stack	Piezoelectric-film
Load	Small	Small	Small	Medium	Large	Small
Stroke	Medium	Small	Large	Large	Small	Medium
Response	Medium	Short	Medium	Long	Short	Short
Flowrate	Medium	Low	High	Medium	Low	High
Pressure	Medium	Small	Large	Medium	Large	Small
Frequency	Low	High	Medium	Low	High	Medium
Structure	Simple	Simple	Complex	Simple	Simple	Simple
Reliability	Medium	Excellent	Good	Poor	Good	Good
Anti-interference	Weak	Strong	Weak	Weak	Strong	Strong

Table 6.1: Comparison of different micropump driving methods [10]

6.3 Working Principle

Diaphragm pumps are a type of mechanical displacement pump that operate by expanding/contracting a flexible diaphragm at high frequency and using check valves to rectify the direction of flow [7]. The generated pressures and flowrates are inversely related, with max pressure occurring at zero flowrate and vice versa (Figure 6.4)

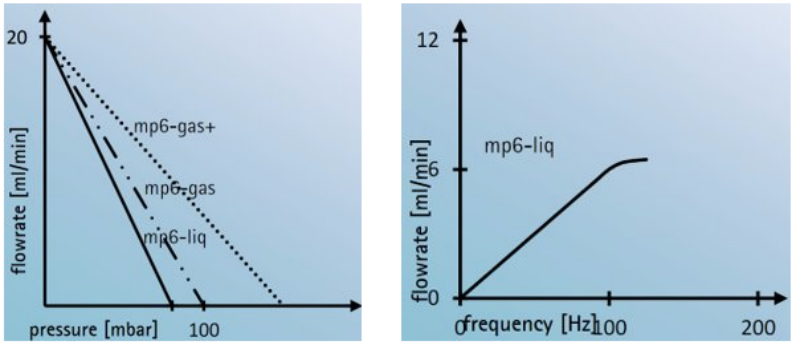


Figure 6.4: Typical flowrate vs pressure and flowrate vs frequency relationship in diaphragm pumps [9]

However, in visual-feedback-controlled applications, flowrates are relatively low (under 1 μ L/min range), allowing us to operate diaphragm pumps near max pressure. Max pressure also depends on fluid compressibility κ and diaphragm deflection (measured as a volume reduction ratio $\frac{\Delta V}{V_0}$, V_0 being the pump chamber volume when diaphragm is at rest) [7]:

$$\Delta P_{max} = \frac{1}{\kappa}\frac{\Delta V}{V_0}$$

Thus for our intended application, generated pressure is directly proportional to diaphragm deflection amplitude.

Diaphragm deflection frequency also corresponds to flowrate/pressure amplitude and stability up to the bandwidth of the diaphragm material (Figure 6.4 right). PZT diaphragms have relatively fast response times, allowing deflection frequencies up to tens of kHz. The inverse piezoelectric effect, which converts an applied electric field to mechanical deformation, enables large deflections to be produced given corresponding input voltages, generating relatively high pressures [10]. Additionally, multiple diaphragm pumps can be connected in series or parallel to increase both the maximum flow rate and maximum back pressure.

Diaphragm pumps are typically limited to unidirectional flow due to the use of passive check valves, but active valves can be added to enable various bidirectional flow configurations [10].

6.4 Methodology

6.4.1 Pump design

The proposed pump design is shown in Figure 6.5 based on the architecture in Figure 6.1 (bottom). A unidirectional PZT pump (Bartels mp6-gas+) is used as the controlled pressure source. A normally closed 5V solenoid air valve (0520D) is used for depressurization and to enable bidirectional flow in the feedback-controlled DMP. All components are combined into a stackable 3D printed PLA case without the need for fasteners.

The PZT pump is actuated by a high voltage waveform generator (Bartels Highdriver4) that applies up to a ± 250 [V] 800 [Hz] sine wave across each PZT element. A laptop PC powers the entire pump assembly and issues commands to a microcontroller (Arduino Pro Micro) which controls amplitude and frequency of the waveform generator over I2C (detailed schematics can be found in Appendix E).

6.4.2 Pump characterization

Pressure Response

Figure 6.6 shows the schematic for characterizing pressure response for both a commercial pressure pump (Fluigent MFCS-EZ) and the proposed modular design.

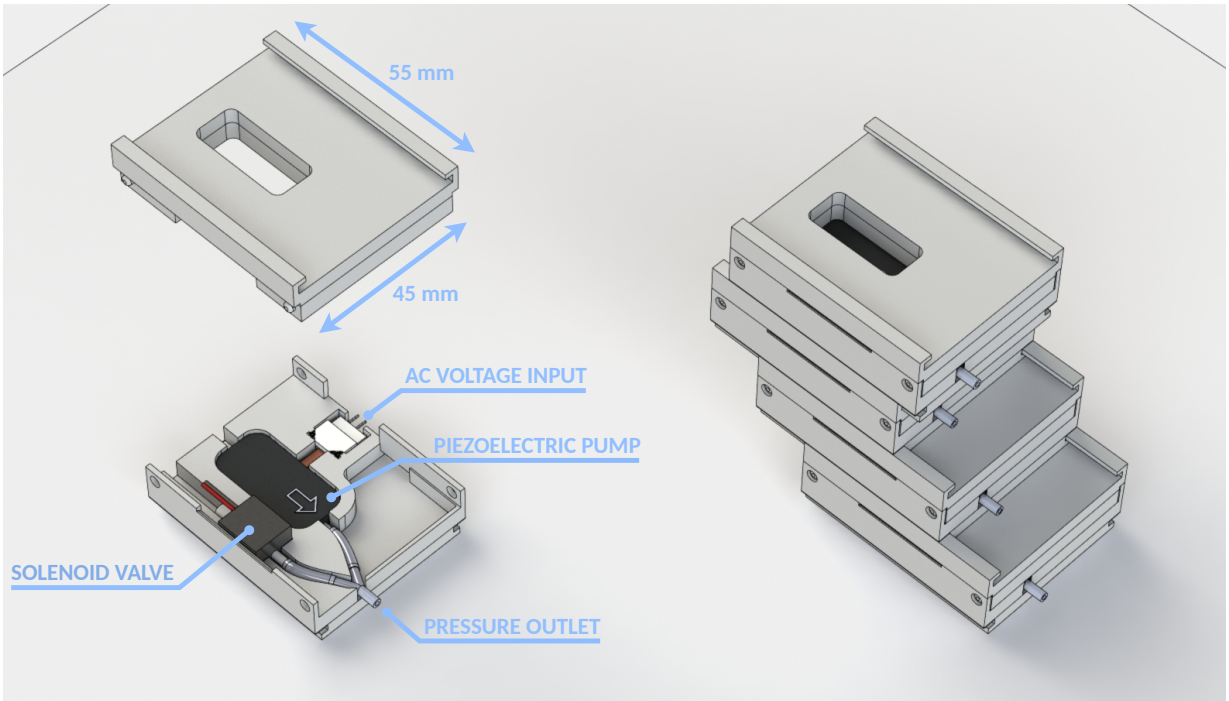


Figure 6.5: Proposed pump design exploded CAD assembly and 4-pump stack

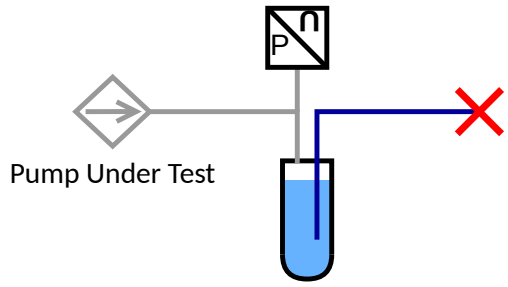


Figure 6.6: Pressure response characterization schematic

To isolate pump dynamics from chip dynamics, pressure response is evaluated up to the chip inlet, which includes pneumatic Silicone tubing, liquid reservoirs (Fluigent 4C), and liquid PFA+ tubing (Idex 1902L). 5 [cSt] Silicone oil (Sigma-Aldrich) is used as the single phase fluid. Air pressure is measured (Honeywell HSCDRRN001BDSA3) at the reservoir for the pumps under test.

The liquid tubing is plugged at the chip inlet to maintain zero flowrate for the pumps under test. This approximation is valid as standard flowrates for feedback-controlled droplet microfluidics are much lower than passive microfluidics, which are already quite low (typically on the order of uL/min).

A laptop PC (Thinkpad P51) is used to issue commands to both pumps over USB. The pressure pump is also provided with 1.3bar regulated pressure, and powered by a 12V supply.

Fluid Interface Response

Combining pump and chip dynamics, the fluid interface displacement response is evaluated by fully pressurizing a microfluidic chip, the schematic of which is shown in Figure 6.7.

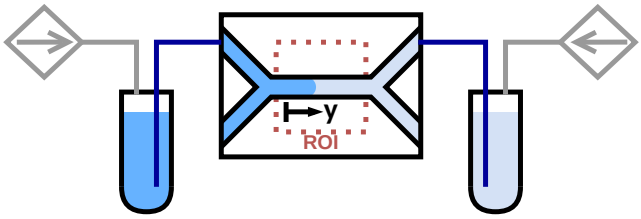


Figure 6.7: Droplet interface displacement response characterization schematic

Here the pumps under test are combined with a T-junction PDMS chip (fabricated using 3D-printed masters), and two immiscible fluid phases - 50 [cSt] Silicone oil, deionized water (Sigma-Aldrich) - produce a visible interface for measurement. By plugging all inlets/outlets except one inlet/outlet pair, the interface can be displaced within a single microchannel.

Since interface displacement integrates the effect of pump pressure (Chapter 2), time domain analysis is less straightforward, so frequency domain analysis is used instead. A zero-mean pseudo-random binary (PRBS) signal (sampling rate 40 [Hz], clock period 4, order 10) is designed to excite the system equally between 0 and 5 [Hz].

Droplet interface detection is performed through a real-time droplet tracking algorithm (see Appendix D).

6.4.3 Pump integration

Figures 6.8 and 6.9 shows the integration of a PZT pump array into the feedback-controlled DMP.

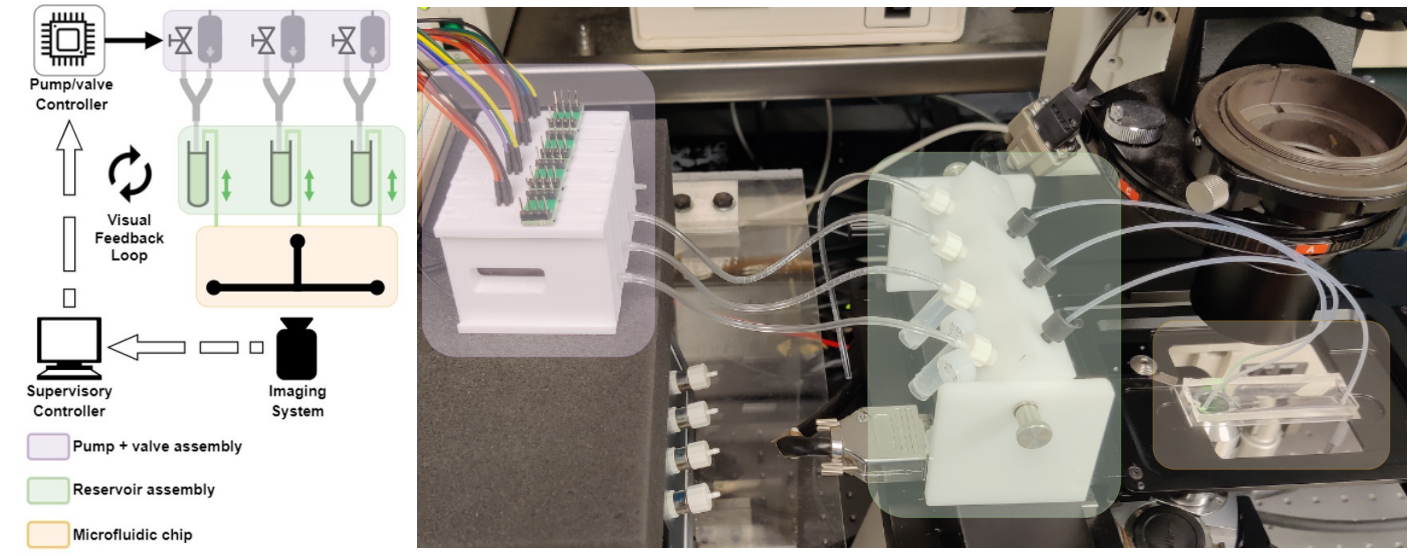


Figure 6.8: Piezoelectric pump integrated into automated DMP

Outside of actuation, the platform consists of a microfluidic chip (T-junction PDMS chip), sample transport mechanism (Fluigent Fluiwell 4C), supervisory controller (PC), and vision system (Nikon Eclipse Ti-E, Andor Zyla 5.5).

Each pump under test is connected to each inlet/outlet to fully pressurize the microfluidic chip, allowing bidirectional droplet control. Pump communications and power are provided identically to the previous section.

6.4.4 Automatic Droplet generation

The methodology for automatic droplet generation follows that of Chapter 5, with the only difference being the replacement of the pressure pump with the PZT pump.

Inlets/outlets are selected by filling the appropriate reservoirs with continuous (5 [cSt] silicone oil, Sigma Aldrich) and dispersed phase (deionised water, Sigma Aldrich) fluid. The microfluidic chip is primed with continuous/carrier phase fluid until it appears at the

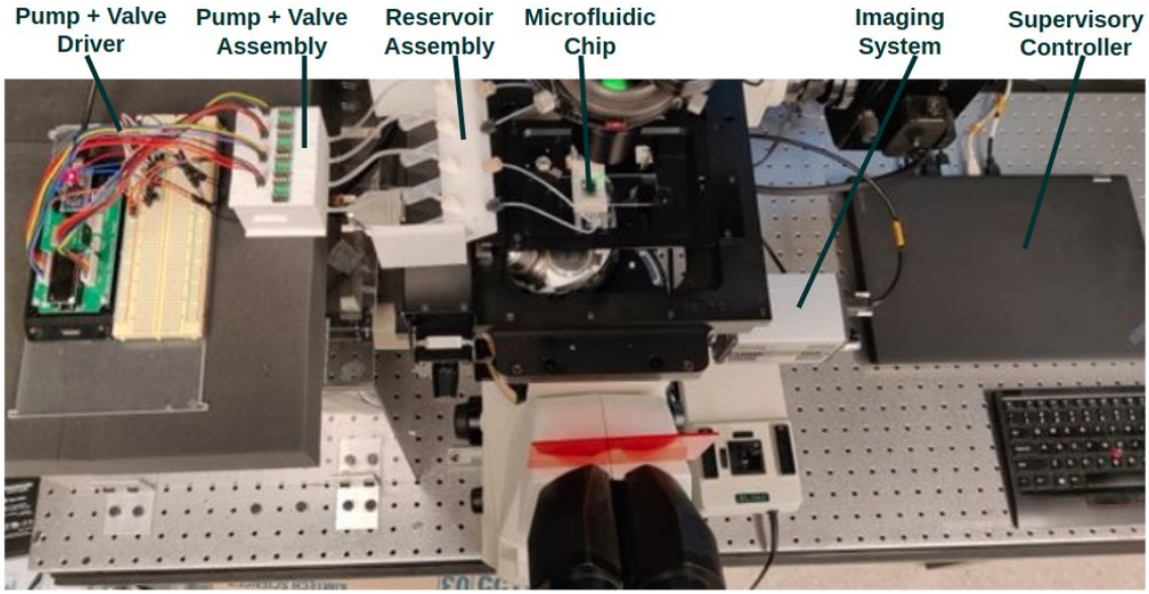


Figure 6.9: Complete setup

outlet reservoir. The dispersed and carrier phases are then balanced by adjusting pump ratios until the interface is stable and detected in the region of interest.

6.5 Results and Discussion

6.5.1 Cost

The proposed pump system is over an order of magnitude cheaper than the conventional pressure pump used for evaluation. A 4-channel array costs $650 USD using manufacturer-supplied pump drivers, but more cost-effective pump driver designs may further reduce this amount. Comparatively, high performance pressure pumps that match the channel count can exceed $10K USD. Even with cheaper off-the-shelf components, BOM costs for conventional pressure pump architectures are still $3000 USD for the same number of channels [71].

6.5.2 Power consumption

The selected PZT pumps each consume 150 [mW], while each solenoid valve was measured to consume 250 [mW] (5 [V], 50 [mA]) in the open state. Thus a 4-channel array will have a worst-case power draw of 1.6 [W], or a current consumption of 320 [mA] at 5 [V]. Given the max current specification of 1.5 [A] for USB 3.0 ports that support dedicated charging, a pump array of up to 18 channels can be powered from a single USB port. Note that since the system is fully pressurized during operation, all valves can remain in their normally closed states most of the time, drawing no power.

6.5.3 Portability

Each pump module takes up $\sim$50 [cm^3] in volume (roughly the size of a matchbox), requires no external pressure source, and can be powered from compact battery banks like Li-ion. The pump drivers used are from evaluation kits and are standalone, but alternatives exist that integrate the driver into the pump with minimal increase in footprint.

6.5.4 Open-loop Performance

Pressure Response

Figure 6.10 shows the pressure step response of both pressure pump and the proposed pump design using the test setup shown in Figure 6.6. Rise time and approximate bandwidth are shown in table 6.2.

Parameter	Pressure Pump	PZT Pump
Rise Time [s]	0.05	2.2
Bandwidth [Hz]	7	0.16

Table 6.2: Comparison of 10-90% Rise Time and First-Order Bandwidth Approximation for Pressure Response of Pumps Under Test

The pressure pump response is much faster, with a 10-90% rise time of $\sim$50 [ms], which at a first-order bandwidth approximation would be $\sim$7 [Hz] ($f_{3dB} = \frac{0.35}{\tau_r}$). However, the non-linear behavior near the setpoint complicates the response, likely due to the proprietary internal control systems involved. The response for the PZT pump on the other hand is

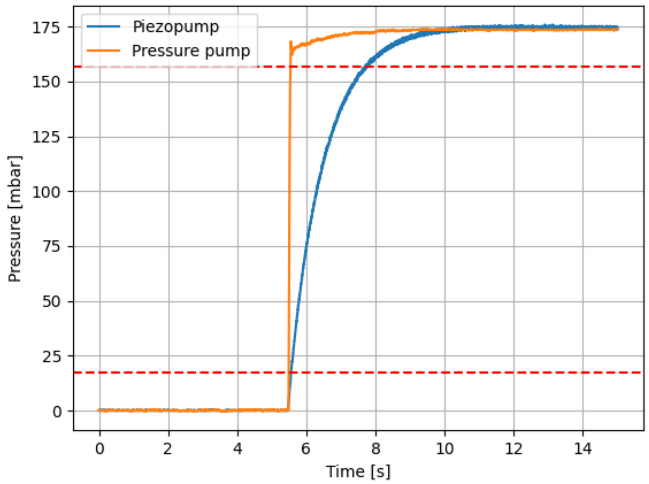


Figure 6.10: Pressure step response under load for pumps under test showing 10 - 90 % rise time thresholds

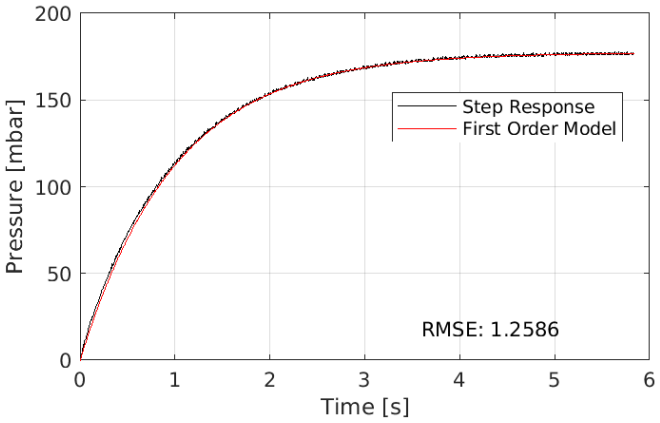


Figure 6.11: Piezopump pressure step response first order fit

clearly linear and can be modelled as first-order (Figure 6.11) with a bandwidth of 1 [rad/s] or 0.16 [Hz].

The pressure response comparison obviously favors the pressure pump, as modulating a proportional valve is faster than actuating a diaphragm multiple times to achieve the same pressure. However, for the purposes of feedback control, the linear and predictable response

of the PZT pump may be more desirable — especially for lower throughput applications — as it would lead to a simpler controller design that is more robust to model uncertainties.

Displacement Response

The interface displacement response in time and frequency domain for both pumps (using test setup in Figure 6.7 are shown in Figure 6.12 and 6.12. Estimated -3dB bandwidth for pumps under test are shown in table 6.3.

Parameter	Pressure Pump	PZT Pump
Bandwidth [Hz (rad/s)]	3.09 (19.4)	0.88 (5.54)

Table 6.3: Estimated -3dB bandwidth based on displacement response of pumps under test

Although there was a large difference in pressure response between pumps under test, the displacement response is more comparable due to the effect of the dominant poles in the overall system. Pressure pump dynamics appear to contribute to resonance in the overall system (Figure 6.13), which can amplify oscillations and introduce instability in feedback control.

6.5.5 Closed-loop Performance

As with the conventional pressure pump in Chapter 5, the droplet generation evaluation program (Table 5.4) was executed repeatedly for different droplet lengths using the PZT pump design. The resulting droplet distribution is shown in Figure 6.14.

Generated droplets achieve under ±5% error relative to calibrated reference, but exceeds those bounds at the minimum achievable droplet length of ∼600 [μm], or a droplet length-to-width aspect ratio of ∼3:1 with 200 [μm] channel width. The monodispersity is calculated for each cluster of generated droplets via coefficient of variation, with an average value of 3.45%.

PZT-generated droplets were generally less monodisperse than pressure-pump-generated droplets (Figure 5.11), but their monodispersity was more consistent across the range of lengths tested, likely owing to their more predictable response. A contributing factor to the inconsistency of pressure pumps is the need for calibration of internal pressure sensors, which have been observed to drift over the duration of an experiment. The linear response

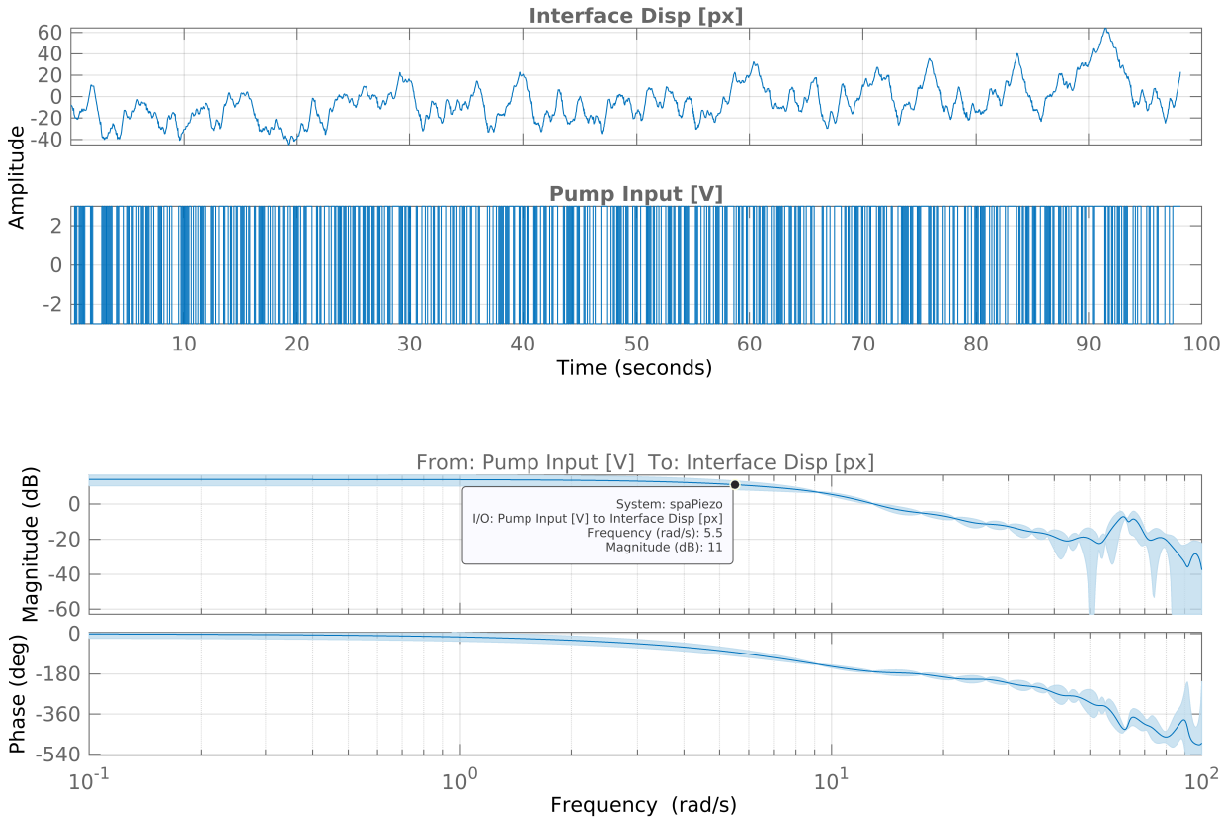


Figure 6.12: Input-output data (top) and Bode plot (bottom) of interface displacement response (with -3dB bandwidth marked) for proposed pump design

of PZT's obviates the need for pressure sensing, contributing to a more consistent response across droplet lengths.

The lower precision and larger minimum droplet length can be mostly accounted for by hardware limitations. The particular PZT pump driver used (Appendix E) has dead zones throughout the input voltage range, adding additional variability at the point where droplet generation occurs. As well, its upper frequency of 800 [Hz] limits the pressure stability especially at higher amplitudes due to increased pulsatility of diaphragm-driven flow, contributing to a reduction in droplet generation precision.

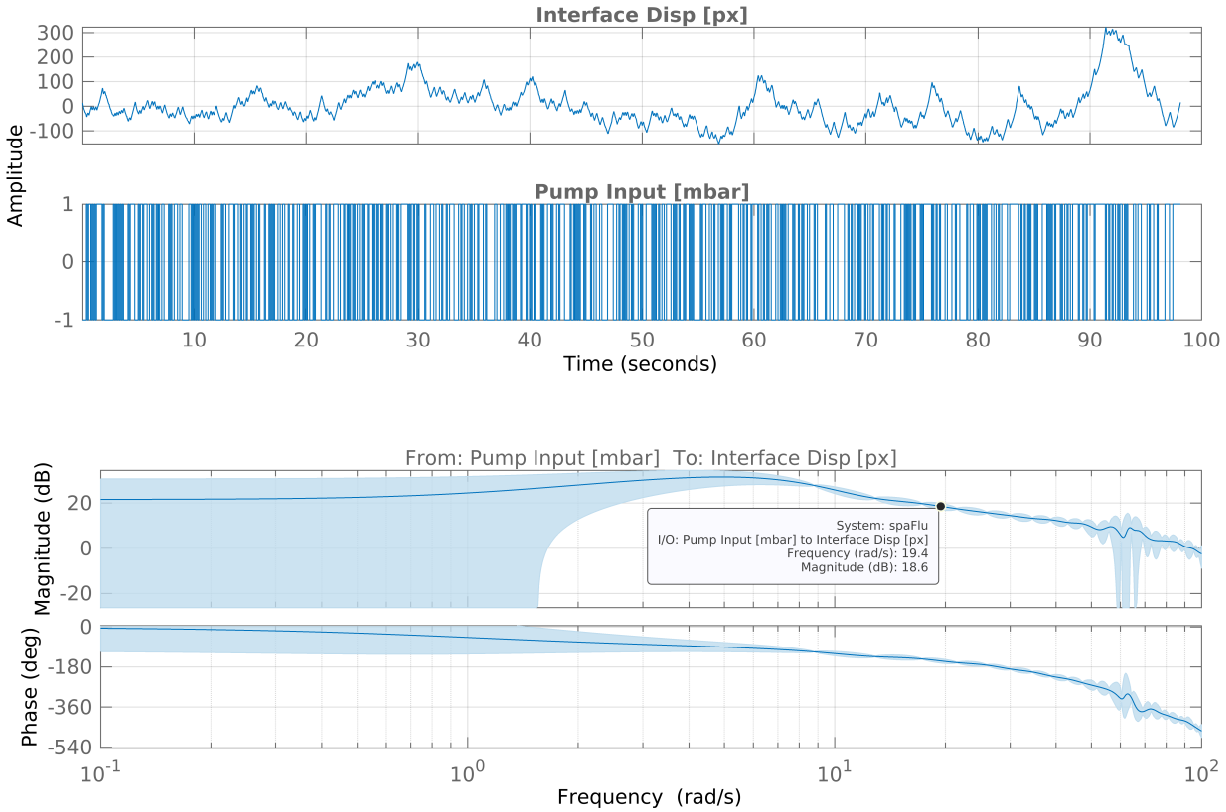


Figure 6.13: Input-output data (top) and Bode plot (bottom) of interface displacement response (with -3dB bandwidth marked) for pressure pump

6.6 Summary

Droplet microfluidic platforms have the potential to be modular and general-purpose, allowing automated arbitrary manipulation of individual droplets via visual feedback. However, the technology is limited in accessibility and practicality in part by the cost and complexity of suitable pumping mechanisms.

This study addressed these limitations by applying low-cost commercial PZT diaphragm micropumps to visual-feedback-controlled droplet manipulation. The proposed pump design was able to reliably perform automated droplet generation on the DMP platform while being simpler, highly portable, and significantly more cost-effective compared to conventional high performance pressure pumps. These findings further improve the viability of automated DMP platforms.

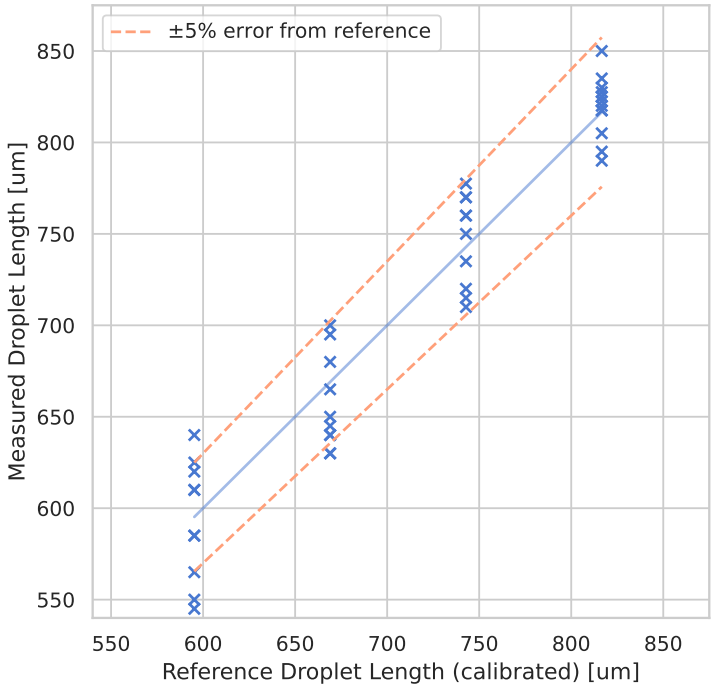


Figure 6.14: Droplet length distribution using automatic droplet generation workflow, calibrated via linear regression with best-fit line $y = 0.79x - 25.7$

As a future direction, it may be worthwhile to evaluate higher frequency/resolution voltage waveform generators (e.g. using techniques like direct digital synthesis) to further improve pressure stability and reduce flow pulsatility. More performant PZT architectures (e.g. PZT standing wave generators) should also be evaluated to improve the maximum pressure/flowrate of individual pumps.

Chapter 7

Conclusion

7.1 Summary

Microfluidics has undeniably driven innovation in the natural sciences. As the field has matured, the need for precise droplet manipulation techniques has continued to grow, leading to the development of more responsive, performant, and robust droplet microfluidics platforms utilizing feedback control. However, full automation of these platforms is impossible without a stable hardware and software foundation to stand on. In this thesis, these more mundane challenges are tackled through the enhancement of controller robustness via online parameter estimation, the development of a practical/accessible software workflow, and the evaluation of more cost-effective modular micropump architectures.

By resolving these issues, future development can be focused on realizing the impact of fully automated DMP's in real-world applications.

7.2 Future Work

Further development of the automated DMP platform should aim toward practical applications.

One promising area of focus is the encapsulation of loose cell aggregates (spheroids) in hydrogel droplets to support their growth into organoids (3D self-organized cell clusters) [77, 78]. Hydrogels are three-dimensional scaffolds that accurately mimic the cell's natural environment, allowing them to support cell growth into organoids. However,

conventional encapsulation methods, such as the pipetted hanging droplet method, are not high-throughput and may lead to unreliable results, due to the lack of repeatability [79]. Droplet microfluidics has the potential to improve the spheroid encapsulation process [80]. However, passive techniques cannot reliably encapsulate individual spheroids without also generating a large number of droplets which are either empty or contain too many spheroids. This necessitates additional downstream sorting techniques which add significant complexity and sample waste. The proposed DMP platform could offer an alternative approach in which droplets are only generated when a spheroid is present, and continuous phase fluid without spheroids are passed to a waste channel. This would ensure output droplets only contain a single spheroid, removing the need for additional sorting.

On modularity and portability, the major components for a truly portable automated DMP have now been validated. By combining lensless imaging for droplet sensing, PZT micropumps for actuation, an embedded processor (e.g. Raspberry Pi), and a portable power source, modules can be made fully portable, allowing much more flexibility in combining them. The next challenge to overcome would be droplet travel between modules, as droplet sensing is currently limited to a single region of interest on a single chip. Analogous to programs that are compiled differently on a sequential (i.e. CPU) vs parallel (i.e. GPU) processor, instructions in droplet manipulation programs would need to be distributed in a way that utilizes all available modules and synchronizes their inputs/outputs. Thus a communication protocol would need to be designed such that modules only produce or consume droplets as necessary for their operation.

Another practical problem is the macro-to-micro interfacing problem inherent to all microfluidic devices. The sample transport has long been a challenge in feedback control, as each additional path the sample must flow through adds additional dynamics to the system for the controller to stabilize. A method of combining all sample transport components could remove all unnecessary tubing in the system, significantly improving droplet response.

A more theoretical area of investigation is whether robust control can be applied to reject the unpredictable and non-linear fluctuations in droplet dynamics by modelling them as uncertainties. This would allow controllers that don't require online optimization, significantly reducing the computational requirements of the processors used.

Since junction dynamics have a clear and significant impact on droplet manipulation precision, another approach to improve precision might be combining pressure-driven flow with localized actuation at the junction, for example with EWOD which has seen extensive use in digital microfluidics.

References

[1] Charles N. Baroud, Francois Gallaire, and Rémi Dangla. Dynamics of microfluidic droplets. *Lab on a Chip*, 10(16):2032, 2010.

[2] Dolomite Microfluidics. Droplet chip user instructions.

[3] Tomasz Glawdel, Caglar Elbuken, and Carolyn L. Ren. Droplet formation in microfluidic t-junction generators operating in the transitional regime. ii. modeling. *Physical Review E*, 85(1):016323, 2012.

[4] Yuk Hei Wong. Feedback controls in droplet microfluidics. Master's thesis, 2016.

[5] Anna-Lena Lamprecht. *User-Level Workflow Design.* Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013.

[6] Fluigent. Compact all-in-one microfluidic micropump, Jan 2023.

[7] Brian D. Iverson and Suresh V. Garimella. Recent advances in microscale pumping technologies: a review and evaluation. *Microfluidics and Nanofluidics*, 5(2):145–174, 2008.

[8] Amar Basu. Ece 7995: Biomems and bioinstrumentation, 2015. Lecture series offered in the Winter semester 2015 at Wayne State University. Available online: `https://www.cytofluidix.com/biomems-and-bioinstrumentation-lecture-series-from-wayne-state-university/`.

[9] Bartels mikrotechnik. *data sheet mp6 micropump*, 2023.

[10] Hengyu Li, Junkao Liu, Kai Li, and Yingxiang Liu. A review of recent studies on piezoelectric pumps and their applications. *Mechanical Systems and Signal Processing*, 151(nil):107393, 2021.

[11] George M. Whitesides. The origins and the future of microfluidics. *Nature*, 442(7101):368–373, 2006.

[12] Shengqing Xu, Zhihong Nie, Minseok Seo, Patrick Lewis, Eugenia Kumacheva, Howard A. Stone, Piotr Garstecki, Douglas B. Weibel, Irina Gitlin, and George M. Whitesides. Generation of monodisperse particles by using microfluidics: Control over size, shape, and composition. *Angewandte Chemie International Edition*, 44(5):724–728, 2005.

[13] H.A. Stone, A.D. Stroock, and A. Ajdari. Engineering flows in small devices: Microfluidics toward a lab-on-a-chip. *Annual Review of Fluid Mechanics*, 36(1):381–411, 2004.

[14] Yung-Chieh Tan, Jeffrey S. Fisher, Alan I. Lee, Vittorio Cristini, and Abraham Phillip Lee. Design of microfluidic channel geometries for the control of droplet volume, chemical concentration, and sorting. *Lab on a Chip*, 4(4):292, 2004.

[15] Piyush Kumar and Manabendra Pathak. Dynamic wetting characteristics during droplet formation in a microfluidic t-junction. *International Journal of Multiphase Flow*, 156(nil):104203, 2022.

[16] Zhuang Zhi Chong, Say Hwa Tan, Alfonso M. Gañán-Calvo, Shu Beng Tor, Ngiap Hiang Loh, and Nam-Trung Nguyen. Active droplet generation in microfluidics. *Lab on a Chip*, 16(1):35–58, 2016.

[17] Marie Hébert, Jan Huissoon, and Carolyn L Ren. A perspective of active microfluidic platforms as an enabling tool for applications in other fields. *Journal of Micromechanics and Microengineering*, 32(4):043001, 2022.

[18] George M. Whitesides. Toward a science of simplicity, 2010. TED talk.

[19] Junjie Zhong, Jason Riordon, Tony C. Wu, Harrison Edwards, Aaron R. Wheeler, Keith Pardee, Alán Aspuru-Guzik, and David Sinton. When robotics met fluidics. *Lab on a Chip*, 20(4):709–716, 2020.

[20] Marie Hébert, Matthew Courtney, and Carolyn L. Ren. Semi-automated on-demand control of individual droplets with a sample application to a drug screening assay. *Lab on a Chip*, 19(8):1490–1501, 2019.

[21] Zablotny, Tomasz. Towards feedback controlled droplet microfluidic platforms. Master's thesis, 2021.

[22] Tomasz Zablotny, Matthew Courtney, Jan P. Huissoon, and Carolyn L. Ren. Lensless imaging for droplet identification towards visual feedback-based pressure controlled droplet microfluidic platforms. *Sensors and Actuators A: Physical*, 334(nil):113338, 2022.

[23] Thomas Moragues, Diana Arguijo, Thomas Beneyton, Cyrus Modavi, Karolis Simutis, Adam R. Abate, Jean-Christophe Baret, Andrew J. deMello, Douglas Densmore, and Andrew D. Griffiths. Droplet-based microfluidics. *Nature Reviews Methods Primers*, 3(1):32, 2023.

[24] Thomas Grandry. Researchers' opinion on droplet generation in microfluidics: Syringe pumps or pressure control?, 2013.

[25] Sanjay M. Prakadan, Alex K. Shalek, and David A. Weitz. Scaling by shrinking: Empowering single-cell 'omics' with microfluidic devices. *Nature Reviews Genetics*, 18(6):345–361, 2017.

[26] Yu-Hsiang Hsu, Wei-Wen Liu, Tung-Han Wu, Carina Jean-Tien Lee, Yu-Hsi Chen, and Pai-Chi Li. Study of diffusive- and convective-transport mediated microtumor growth in a controlled microchamber. *Biomedical Microdevices*, 21(1):7, 2019.

[27] Matthew Courtney, Xiaoming Chen, Sarah Chan, Tarek Mohamed, Praveen P. N. Rao, and Carolyn L. Ren. Droplet microfluidic system with on-demand trapping and releasing of droplet for drug screening applications. *Analytical Chemistry*, 89(1):910–915, 2016.

[28] Liang Li and Rustem F. Ismagilov. Protein crystallization using microfluidic technologies based on valves, droplets, and slipchip. *Annual Review of Biophysics*, 39(1):139–158, 2010.

[29] Suhanya Duraiswamy and Saif A. Khan. Droplet-based microfluidic synthesis of anisotropic metal nanocrystals. *Small*, 5(24):2828–2834, 2009.

[30] Wynter J. Duncanson, Tina Lin, Adam R. Abate, Sebastian Seiffert, Rhutesh K. Shah, and David A. Weitz. Microfluidic synthesis of advanced microparticles for encapsulation and controlled release. *Lab on a Chip*, 12(12):2135, 2012.

[31] K. Karthikeyan and L. Sujatha. Fluorometric sensor for mercury ion detection in a fluidic mems device. *IEEE Sensors Journal*, 18(13):5225–5231, 2018.

[32] Pingan Zhu and Liqiu Wang. Passive and active droplet generation with microfluidics: a review. *Lab on a Chip*, 17(1):34–75, 2017.

[33] Xiaoming Chen and Carolyn L. Ren. A microfluidic chip integrated with droplet generation, pairing, trapping, merging, mixing and releasing. *RSC Advances*, 7(27):16738–16750, 2017.

[34] Cheng-Tso Chen and Gwo-Bin Lee. Formation of microdroplets in liquids utilizing active pneumatic choppers on a microfluidic chip. *Journal of Microelectromechanical Systems*, 15(6):1492–1498, 2006.

[35] Avishay Bransky, Natanel Korin, Maria Khoury, and Shulamit Levenberg. A microfluidic droplet generator based on a piezoelectric actuator. *Lab Chip*, 9(4):516–520, 2009.

[36] Haejune Kim, Dawei Luo, Darren Link, David A. Weitz, Manuel Marquez, and Zhengdong Cheng. Controlled production of emulsion drops using an electric field in a flow-focusing microfluidic device. *Applied Physics Letters*, 91(13):133106, 2007.

[37] Claudiu A. Stan, Sindy K. Y. Tang, and George M. Whitesides. Independent control of drop size and velocity in microfluidic flow-focusing generators using variable temperature and flow rate. *Analytical Chemistry*, 81(6):2399–2402, 2009.

[38] Charles N. Baroud, Matthieu Robert de Saint Vincent, and Jean-Pierre Delville. An optical toolbox for total control of droplet microfluidics. *Lab on a Chip*, 7(8):1029, 2007.

[39] Charles N. Baroud, Jean-Pierre Delville, François Gallaire, and Régis Wunenburger. Thermocapillary valve for droplet production and sorting. *Physical Review E*, 75(4):046302, 2007.

[40] Jian Gong and Chang-Jin "CJ" Kim. All-electronic droplet generation on-chip with real-time feedback control for ewod digital microfluidics. *Lab on a Chip*, 8(6):898, 2008.

[41] Jia Li, Noel S. Ha, Tingyi 'Leo' Liu, R. Michael van Dam, and Chang-Jin 'CJ' Kim. Ionic-surfactant-mediated electro-dewetting for digital microfluidics. *Nature*, 572(7770):507–510, 2019.

[42] Jia Li and Chang-Jin "CJ" Kim. Current commercialization status of electrowetting-on-dielectric (ewod) digital microfluidics. *Lab on a Chip*, 20(10):1705–1712, 2020.

[43] Thomas Franke, Adam R. Abate, David A. Weitz, and Achim Wixforth. Surface acoustic wave (saw) directed droplet flow in microfluidics for pdms devices. *Lab on a Chip*, 9(18):2625, 2009.

[44] Marc A. Unger, Hou-Pu Chou, Todd Thorsen, Axel Scherer, and Stephen R. Quake. Monolithic microfabricated valves and pumps by multilayer soft lithography. *Science*, 288(5463):113–116, 2000.

[45] Cong Yu, Senol Mutlu, Ponnambalam Selvaganapathy, Carlos H. Mastrangelo, Frantisek Svec, and Jean M. J. Fréchet. Flow control valves for analytical microfluidic chips without mechanical parts based on thermally responsive monolithic polymers. *Analytical Chemistry*, 75(8):1958–1961, 2003.

[46] Qing Yu, Joseph M. Bauer, Jeffrey S. Moore, and David J. Beebe. Responsive biomimetic hydrogel valve for microfluidics. *Applied Physics Letters*, 78(17):2589–2591, 2001.

[47] María Díaz-González, César Fernández-Sánchez, and Antonio Baldi. Multiple actuation microvalves in wax microfluidics. *Lab on a Chip*, 16(20):3969–3976, 2016.

[48] Slawomir Jakiela, Tomasz S. Kaminski, Olgierd Cybulski, Douglas B. Weibel, and Piotr Garstecki. Bacterial growth and adaptation in microdroplet chemostats. *Angewandte Chemie International Edition*, 52(34):8908–8911, 2013.

[49] Olgierd Cybulski, Slawomir Jakiela, and Piotr Garstecki. Whole teflon valves for handling droplets. *Lab on a Chip*, 16(12):2198–2210, 2016.

[50] Deqing Huang, Kang Wang, Yaolei Wang, Hejia Sun, Xingyuan Liang, and Tao Meng. Precise control for the size of droplet in t-junction microfluidic based on iterative learning method. *Journal of the Franklin Institute*, 357(9):5302–5316, 2020.

[51] Alexander E. Siemenn, Evyatar Shaulsky, Matthew Beveridge, Tonio Buonassisi, Sara M. Hashmi, and Iddo Drori. A machine learning and computer vision approach to rapidly optimize multiscale droplet generation. *ACS Applied Materials and Interfaces*, 14(3):4668–4679, 2022.

[52] David Wong and Carolyn L. Ren. Microfluidic droplet trapping, splitting and merging with feedback controls and state space modelling. *Lab on a Chip*, 16(17):3317–3329, 2016.

[53] David Wong, Kaan Erkorkmaz, and Carolyn L. Ren. Robodrop: a multi-input multi-output control system for on-demand manipulation of microfluidic droplets based on computer vision feedback. *IEEE/ASME Transactions on Mechatronics*, 25(2):1129–1137, 2020.

[54] Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall PTR, 2nd edition, 1999.

[55] Henrik Bruus. *Theoretical Microfluidics*. Oxford University Press, 2008.

[56] Kemin Zhou and John C. Doyle. *Essentials of Robust Control*. Prentice Hall, Louisiana State University and California Institute of Technology, 1998.

[57] Desineni Subbaram Naidu. *Optimal Control Systems*. []. CRC Press, 2018.

[58] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, UK, 2004.

[59] Dong Qin, Younan Xia, and George M Whitesides. Soft lithography for micro- and nanoscale patterning. *Nature Protocols*, 5(3):491–502, 2010.

[60] The Dolomite Centre Ltd. Glass chip design guide.

[61] Fluigent. fgt-sdk, 2023.

[62] P. Sajeesh, M. Doble, and A. K. Sen. Hydrodynamic resistance and mobility of deformable objects in microfluidic channels. *Biomicrofluidics*, 8(5):nil, 2014.

[63] Petros Ioannou and Bariş Fidan. *Adaptive Control Tutorial*. Society for Industrial and Applied Mathematics (SIAM), 2008.

[64] K.J. Åström and B. Wittenmark. *Adaptive Control*. Dover Books on Electrical Engineering. Dover Publications, 2008.

[65] C. Schmid and L.T. Biegler. Quadratic programming methods for reduced hessian sqp. *Computers and Chemical Engineering*, 18(9):817–832, 1994.

[66] Kees van Hee. *Workflow Management*. The MIT Press, 2002.

[67] J. H. Xu, S. W. Li, J. Tan, Y. J. Wang, and G. S. Luo. Preparation of highly monodisperse droplet in a t-junction microfluidic device. *AIChE Journal*, 52(9):3005–3010, 2006.

[68] Guilhem Velvé Casquillas and Timothée Houssin. Responsiveness in microfluidics: Syringe pumps and microfluidic research, Oct 2022.

[69] C. Fütterer, N. Minc, V. Bormuth, J.-H. Codarbox, P. Laval, J. Rossier, and J.-L. Viovy. Injection and flow control system for microchannels. *Lab Chip*, 4(4):351–356, 2004.

[70] Ki Wan Bong, Stephen C. Chapin, Daniel C. Pregibon, David Baah, Tamara M. Floyd-Smith, and Patrick S. Doyle. Compressed-air flow control system. *Lab Chip*, 11(4):743–747, 2011.

[71] Run Ze Gao, Marie Hébert, Jan Huissoon, and Carolyn L. Ren. μpump: an open-source pressure pump for precision fluid handling in microfluidics. *HardwareX*, 7(nil):e00096, 2020.

[72] Jonathan Shemesh, Avishay Bransky, Maria Khoury, and Shulamit Levenberg. Advanced microfluidic droplet manipulation based on piezoelectric actuation. *Biomedical Microdevices*, 12(5):907–914, 2010.

[73] Yuki Oda, Hirofumi Oshima, Masaya Nakatani, and Masahiko Hashimoto. Vacuum-driven fluid manipulation by a piezoelectric diaphragm micropump for microfluidic droplet generation with a rapid system response time. *ELECTROPHORESIS*, 40(3):414–418, 2018.

[74] Gehan Melroy, Amith Mudugamuwa, Samith Hettiarachchi, Ranjith Amarasinghe, Van Dau, Pubudu Kumarage, Nirosh Jayaweera, and Chen Qing-guang. *PZT Based Active Microfluidic Droplet Generator for Lab-on-a-Chip Devices*, pages 277–289. Sustainable Design and Manufacturing. Springer Singapore, 2021.

[75] R. Joemaa, M. Grosberg, T. Rang, and T. Pardy. Low-cost, portable dual-channel pressure pump for droplet microfluidics. In *2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO)*, page nil, 5 2022.

[76] Timothy Kassis, Paola M. Perez, Chloe J.W. Yang, Luis R. Soenksen, David L. Trumper, and Linda G. Griffith. Piflow: a biocompatible low-cost programmable dynamic flow pumping system utilizing a raspberry pi zero and commercial piezoelectric pumps. *HardwareX*, 4(nil):e00034, 2018.

[77] Abhijith Shankaran, Keshava Prasad, Sima Chaudhari, Angela Brand, and Kapaettu Satyamoorthy. Advances in development and application of human organoids. *3 Biotech*, 11(6):257, 2021.

[78] Jihoon Kim, Bon-Kyoung Koo, and Juergen A. Knoblich. Human organoids: Model systems for human biology and medicine. *Nature Reviews Molecular Cell Biology*, 21(10):571–584, 2020.

[79] Vanessa Velasco, S. Ali Shariati, and Rahim Esfandyarpour. Microtechnology-based methods for organoid models. *Microsystems and Nanoengineering*, 6(1):76, 2020.

[80] Thomas Heida, Jens W. Neubauer, Maximilian Seuss, Nicolas Hauck, Julian Thiele, and Andreas Fery. Mechanically defined microgels by droplet microfluidics. *Macromolecular Chemistry and Physics*, 218(2):1600418, 2016.

APPENDICES

Appendix A

Adaptive Law

A.1 Recursive Least Squares

For estimation, we create an estimation model from the parametric model. For a single channel,

$$\hat{z}_{i,k} = \psi_{i,k}^T \hat{\theta}_{i,k-1} = \begin{bmatrix} -x_{i,k-1} \\ u_{1,k-1} \\ u_{2,k-1} \\ \vdots \\ u_{i,k-1} \\ \vdots \\ u_{n,k-1} \end{bmatrix}^T \begin{bmatrix} \hat{\tau}_i \\ \hat{b}_{i,1} \\ \hat{b}_{i,2} \\ \vdots \\ \hat{b}_{i,n} \end{bmatrix} \tag{A.1}$$

The estimation model can be updated recursively with an adaptive law (with adaptive gain L_k and estimation error ϵ_k) generally of the form

$$\hat{\theta}_k = \hat{\theta}_{k-1} + L_k \epsilon_k = \hat{\theta}_{k-1} + L_k (z_k - \psi_k^T \hat{\theta}_{k-1}) \tag{A.2}$$

More specifically, using the discrete-time recursive least squares (RLS) algorithm [54], the adaptive gain is derived as

$$L_k = P_k \psi_k = \frac{P_{k-1} \psi_k}{\lambda_k + \psi_k^T P_{k-1} \psi_k} \tag{A.3}$$

with parameter estimation covariance

$$P_k = \frac{1}{\lambda_k}\left[P_{k-1} - \frac{P_{k-1}\psi_k\psi_k^T P_{k-1}}{\lambda_k + \psi_k^\top P_{k-1}\psi_k}\right] \tag{A.4}$$

Pure RLS algorithms can suffer from covariance wind-up, in which the covariance matrix can become arbitrarily small [63]. We can alleviate this by tuning the forgetting factor λ_k. In a discrete-time system with sampling period $T = 0.025$ [s], λ_k can be selected by tuning the memory horizon τ_m [s], which specifies how quickly older samples are discounted (larger memory horizons will remember older samples for longer) [54]:

$$\tau_m = \frac{T}{1-\lambda} \tag{A.5}$$

The forgetting factor is calculated from memory horizon:

$$\lambda = 1 - \frac{T}{\tau_m} \tag{A.6}$$

Iterative tuning in simulation found a 10 [s] memory horizon was adequate, producing a forgetting factor of 0.9975.

Another issue to consider is the problem of persistence of excitation. If ψ_k is persistently exciting, the parameter estimation error will converge to 0 over time. Given that each channel has $n+1$ regressors, for $\psi_{i,k}$ to be persistently exciting, the input signal must contain at least $\frac{n+1}{2}$ frequencies [63]. However, this condition conflicts with the objective of the optimal control law, which is to regulate the output trajectory tracking error to 0. Because our plant is stable, in steady state there is no input excitation.

A.2 Parameter Projection

To mitigate the loss of persistence of excitation, a group of techniques known as robustness modifications have been developed [63, 64] to guarantee the adaptive law is robust to non-ideal parameter estimation conditions. Here we will consider only the parameter projection modification due to its simplicity.

From the parametric model derived in Equation 4.7, physical constraints in the system dictate that the sign of each model parameter in θ must be positive. To elaborate, if τ_i

becomes negative, the eigenvalues of the plant become unstable. This does not match reality as droplet position is at worst marginally stable within a microchannel. On the other hand, the sign of $b_{i,j}$ is determined by geometry. Any $b_{i,j}$ becoming negative would indicate that an input/output relationship has been inverted, which is physically impossible during operation.

Parameter projection can be used to augment the RLS adaptive law with these parameter constraints:

$$\hat{\theta}_k = \begin{cases} \hat{\theta}_{k-1} + P_{k-1}\psi_k\epsilon_k, & \text{if } \hat{\theta}_k \in S^0 \text{ or } \hat{\theta}_k \in \delta(S) \text{ and } (P_{k-1}\psi_k\epsilon_k)^T\nabla g \leq 0 \\ \hat{\theta}_{k-1}, & \text{otherwise} \end{cases} \tag{A.7}$$

$$P_k = \begin{cases} \frac{1}{\lambda_k}\left[P_{k-1} - \frac{P_{k-1}\psi_k\psi_k^T P_{k-1}}{\lambda_k + \psi_k^T P_{k-1}\psi_k}\right], & \text{if } \hat{\theta}_k \in S^0 \text{ or } \hat{\theta}_k \in \delta(S) \text{ and } (P_{k-1}\psi_k\epsilon_k)^T\nabla g \leq 0 \\ \frac{1}{\lambda_k}P_{k-1}, & \text{otherwise} \end{cases} \tag{A.8}$$

The allowed parameter space is defined as $S = \{\theta \in \mathbb{R}^n \mid g(\theta) \leqslant 0\}$. Its boundary and interior are defined as $\delta(s) = \{\theta \in \mathbb{R}^n \mid g(\theta) = 0\}$ and $S^0 = \{\theta \in \mathbb{R}^n \mid g(\theta) < 0\}$ respectively.

Given the constraint $\theta_i \geq 0$, we can define $g(\theta_i) = -\theta_i$, $\nabla g = -1$. Thus for the i-th parameter, the parameter space is constrained to:

$$S_i = \{\theta_i \in \mathbb{R} \mid g(\theta_i) = -\theta_i \leqslant 0\} = \{\theta_i \in \mathbb{R} \mid \theta_i \geqslant 0\} \tag{A.9}$$

Appendix B

Droplet Measurement Framework

B.1 Overview

For feedback controlled droplet microfluidics, ideally we would be able to provide the entire pressure/flow field throughout the microchannel network for feedback. In practice, only the fluid interface provides any clue as to the dynamics of the system, and these interfaces may only be present in certain channels, leaving the state of other channels completely unobservable. Previously this problem was resolved by removing those states from the overall state-space model [4]. However, this leads to discontinuous jumps when droplet interfaces cross between different channels, necessitating additional compensation techniques to smooth these transitions [4].

Instead we try a new approach that attempts to maintain measurement continuity as much as possible. First, we define two types of measurements for each channel, direct and inferred. Direct measurements in any channel can be made whenever an interface is present in that channel. As the interface leaves a channel, it is no longer directly measurable. However, analogous to Kirchoff's Current Law, or more specifically conservation of charge, that same interface must now appear as an interface in another channel or multiple interfaces in a number of other channels. Thus we can make an inferred measurement of interface position in that channel based on the direct measurements of interfaces in all other channels.

Now that we have an overview of the approach, we will now describe the algorithms for direct and indirect interface measurement in detail.

B.2 Directly measured interfaces

Given an n channel, single-junction network where m channels have one or more visible interfaces, we can define m lists $L_1, L_2, \ldots, L_m$ of direct interface locations.

To obtain L_i, we first apply background subtraction to obtain interface positions in the foreground. Morphological operations (dilation + erosion + skeletonization) are applied to close any openings in the interface, producing a smooth boundary between the fluid phases. Then the foreground is segmented into individual channels based on specified bounded boxes. The foreground pixels in the center of channel i is obtained, then the mean of each pixel cluster is found and appended to L_i (Figure B.1).

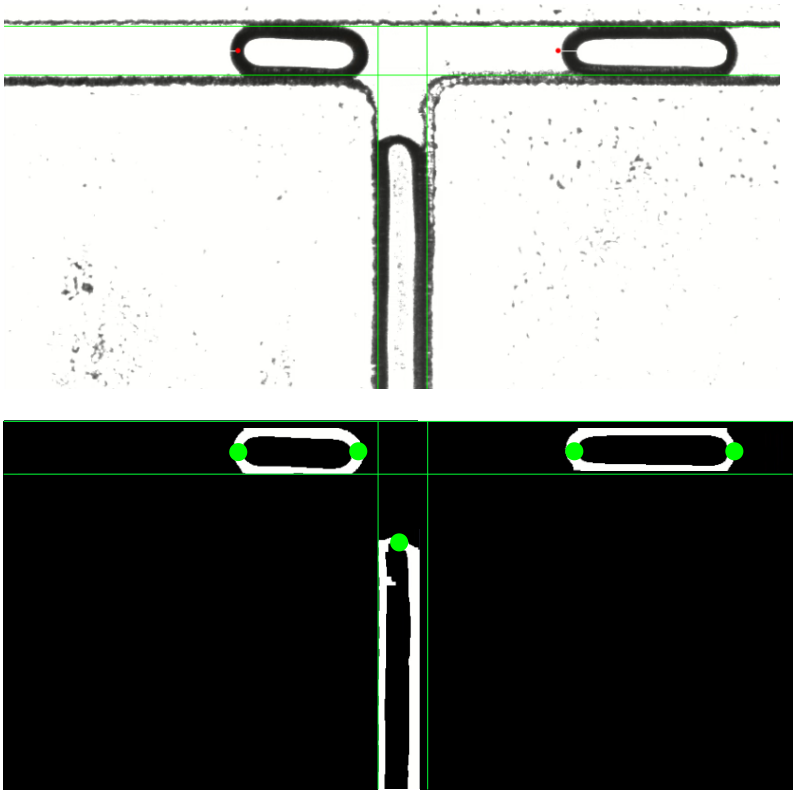


Figure B.1: Raw and processed frame

B.3 Inferred interfaces

We can define $n-m$ lists $K_1, K_2, \ldots, K_{n-m}$ of inferred interface locations for the remaining $n-m$ unmeasured channels.

Because multiple interfaces can be present in L_i, multiple inferred interface locations are possible. This makes the resulting algorithm slightly more involved, as each combination of direct interface locations must be evaluated to produce the list of all possible inferred interface locations. The resulting algorithm is shown below.

Algorithm 1 Find Inferred Droplet Interface Locations

Input:
    $L_1, L_2, \ldots, L_m$ (m lists of direct interface locations for m measured channels)
    n (total number of channels)
    m (number of measured channels)
Output:
    $K_1, K_2, \ldots, K_{n-m}$ ($n-m$ lists of inferred interface locations for $n-m$ unmeasured channels)
for $j = 1$ to $n-m$ **do**
    $K_j \leftarrow \emptyset$ ▷ Initialize inferred list for j-th unmeasured channel
    for all combinations $C = [c_1, c_2, \ldots, c_m]$ where $c_i \in L_i$ **do**
        $I_C \leftarrow \frac{-1}{n-1} \sum_{i=1}^{m} c_i$ ▷ Compute inferred measurement for combination C
        Append I_C to K_j
    end for
end for
return $K_1, K_2, \ldots, K_{n-m}$

A limitation of this algorithm is the fact that it currently only applies to single-junction microchannel networks. However, a shift in reference frame should make it compatible with arbitrary microchannel networks.

B.4 Interface re-identification

Now that we have a list of all measured and inferred interface locations, we need to narrow down the specific interfaces of interest. We do this by initializing the measurement with the closest interface to the junction, then re-identifying that interface as it moves throughout the microfluidic chip.

To re-identify droplet interfaces between frames, we find the direct/inferred measurement in each channel closest to the previous value.

The result of this approach is that a single interface can now be represented in $\mathbb{R}_n$ as a continuous measurement (direct/inferred) vector $\mathbf{y}$, regardless of which channel it happens to be in.

B.5 Capturing interface discontinuities

The above algorithm works well when the interface is smooth, but many droplet manipulations (generate, split, merge) create discontinuities in the interface (Figure 1.1). The difficulty in capturing this behavior boils down to the limited degrees of freedom in the system. Within each channel, there is only a single degree of freedom - along the channel. But if we want to perform more complex sequences of droplet operations, at some point we will have to switch our measurement to any new interfaces that are produced.

To capture this in our measurement, let's define a primary and secondary measurement vector $\mathbf{y}_\alpha = \begin{bmatrix} y_{\alpha 1} \\ y_{\alpha 2} \\ \vdots \\ y_{\alpha n} \end{bmatrix}$ and $\mathbf{y}_\beta = \begin{bmatrix} y_{\beta 1} \\ y_{\beta 2} \\ \vdots \\ y_{\beta n} \end{bmatrix}$ in $\mathbb{R}_n$ that each represent a continuous interface.

When the measurement begins, both vectors will be identical to each other. However, whenever any component of one vector crosses the junction, we will reset the other vector to the interface closest to the junction, which provides a much more useful measurement for feedback control (Figure 5.6).

Two sets of measurement vectors allows us to fully characterize the motion of each fundamental droplet operation, providing the maximum amount of information for the controller at all times. It also gives the supervisory controller the flexibility to switch between measurements on the fly.

Appendix C

Software Design

The source code for the software application is found at https://github.com/KevinHQChen/autoDMP.

C.1 Software Architecture

Figure C.1 shows the software architecture for the automated DMP.

Four major software subsystems are identified:

1. Image capture - updates shared frame buffer from machine vision camera
2. Image processing - detects droplet locations and updates shared droplet position queue
3. Supervisor - given current state from prediction model and current waypoint of desired trajectory, produces optimal control signals for droplet actuation
4. User interface - renders GUI windows to visualize data in each software subsystem, and provides a method for users to input waypoint sequences as plain text files

Each subsystem can be run as a standalone application, or in parallel with other subsystems. Interfaces between subsystems are clearly defined, with thread-safe buffers/queues used for all shared data structures, minimizing the risk of race conditions and deadlock. This modular multi-threaded architecture enables easier testing, maintainability and extensibility.

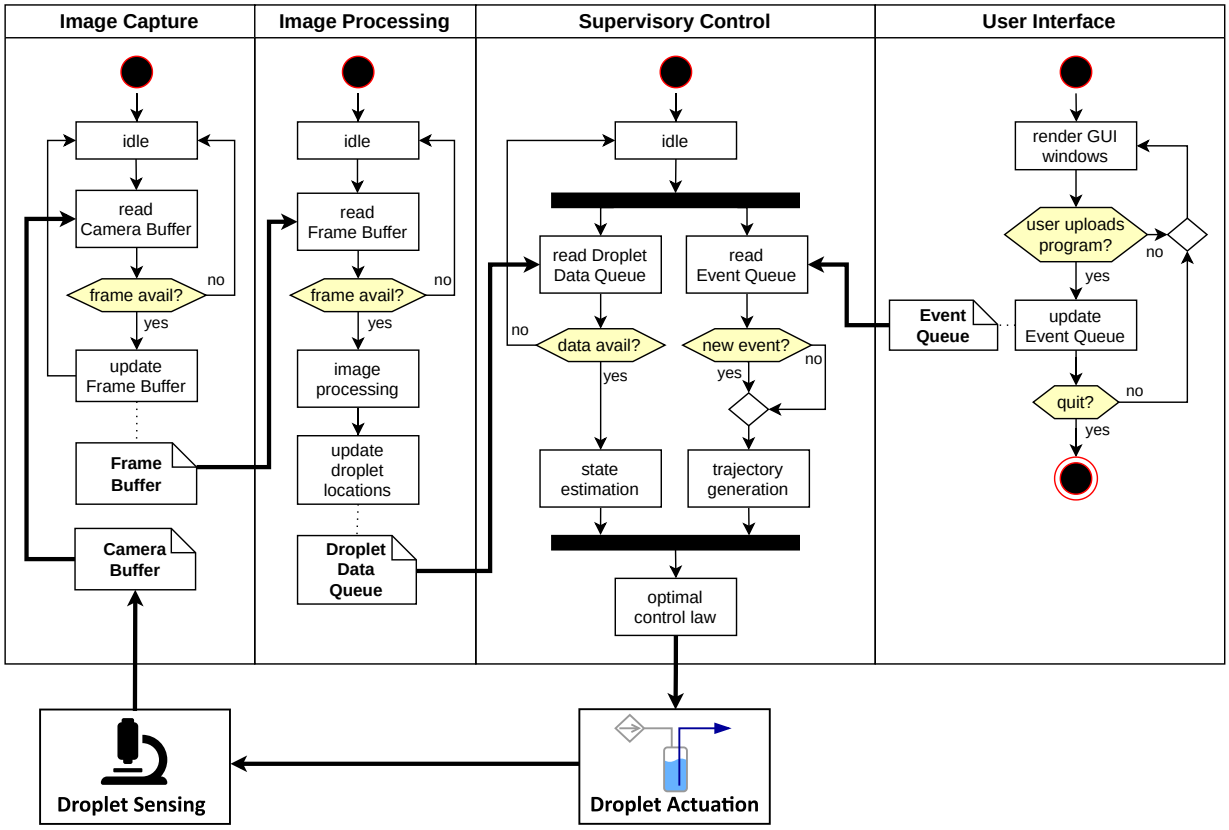


Figure C.1: Multi-threaded software architecture

C.2 Software Infrastructure

Software infrastructure starts from the operating system (OS) running on the bare metal processor, includes the compiler toolchain that builds, compiles, and packages the application (and its dependencies), and at the highest level involves cloud platforms (although those are irrelevant for our application).

Instead of manually keeping track of these large complex pieces of software, we can package together only the parts that are needed for our application. This technique is known as containerization (Figure C.2).

Unlike virtual machines which bundles an entire operating system, containers run on top of any OS, packaging only the application and its dependencies. This makes them much more light-weight and performant, enabling applications like real-time image processing

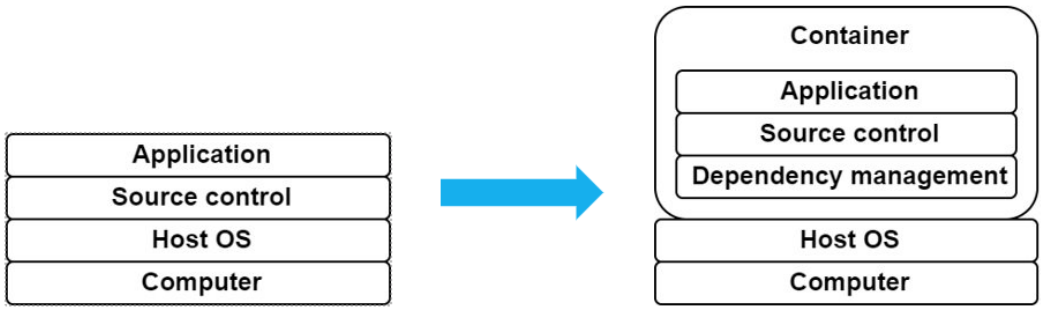


Figure C.2: Non-containerized vs containerized application

and control, while also being more accessible to users running the application on different platforms (e.g. Windows, MacOS).

A popular containerization tool is Docker, which solves the previously discussed challenges by allowing the user to define every part of the infrastructure (application dependencies, compilers, hardware drivers) within a configuration file. The first time the application is built, Docker builds the package using the configuration file as the recipe. This package becomes an immutable image (template) from which containers (instances) can be created. As the image will not change often, it can be uploaded to a shared repository (e.g. Docker Hub), allowing other users to immediately have access to the application along with all required software infrastructure.

To further improve accessibility, open-source tools are preferred when possible, such as the OS (Ubuntu Linux), compiler toolchain (GCC), and the containerization tool itself (Docker). The use of Linux-based operating systems enables the software platform to run on much lower-end x86 and ARM processors (e.g. Raspberry Pi), further reducing the barrier to entry.

Appendix D

Droplet Train Generation

The instruction set for a single droplet generation (Table 5.3) can be repeated to generate custom droplet trains:

$\mathbf{r}_\alpha[\mu m]$	$\mathbf{r}_\beta[\mu m]$	$T_{\text{pre}}[s]$	$T_{\text{move}}[s]$	$T_{\text{post}}[s]$
$(-1500, 0, 0)$	$(0, 0, 0)$	0	5	2
$(210, 0, 0)$	$(0, 0, 0)$	0	7	0
$(0, -210, -850)$	$(0, 0, 0)$	0	10	2
$(0, 467, -850)$	$(0, 0, 0)$	0	7	0
$(0, 0, 0)$	$(-420, 0, -2000)$	0	12	0
$(0, 0, 0)$	$(210, 0, -3200)$	0	7	0
$(0, -210, -950)$	$(0, 0, 0)$	0	10	2
$(0, 467, -950)$	$(0, 0, 0)$	0	7	0
$(0, 0, 0)$	$(-420, 0, -2000)$	0	12	0
$(0, 0, 0)$	$(210, 0, -3200)$	0	7	0
$(0, -210, -1050)$	$(0, 0, 0)$	0	10	2
$(0, 467, -1050)$	$(0, 0, 0)$	0	7	0
$(0, 0, 0)$	$(-420, 0, -2000)$	0	12	0
$(0, 0, 0)$	$(210, 0, -3200)$	0	7	0

Table D.1: Droplet train generation evaluation program

The associated video file "autoDropTrainGen.mp4" shows the execution of this program.

Appendix E

Piezoelectric Pump Electronics

The piezoelectric pump is actuated by a high voltage waveform generator (Bartels Highdriver4) that applies up to a ±250 [V] 800 [Hz] sine wave across each piezoelectric element. A laptop PC powers the entire pump assembly and issues commands to a microcontroller (Arduino Pro Micro) which controls amplitude and frequency of the waveform generator over I2C, as well as on/off state of each solenoid valve via GPIO.

The schematic in Figure E.1 shows the electrical connections between components. All electrical connections are facilitated through a carrier board (Bartels Mp-Multiboard) shown in Figure E.2.

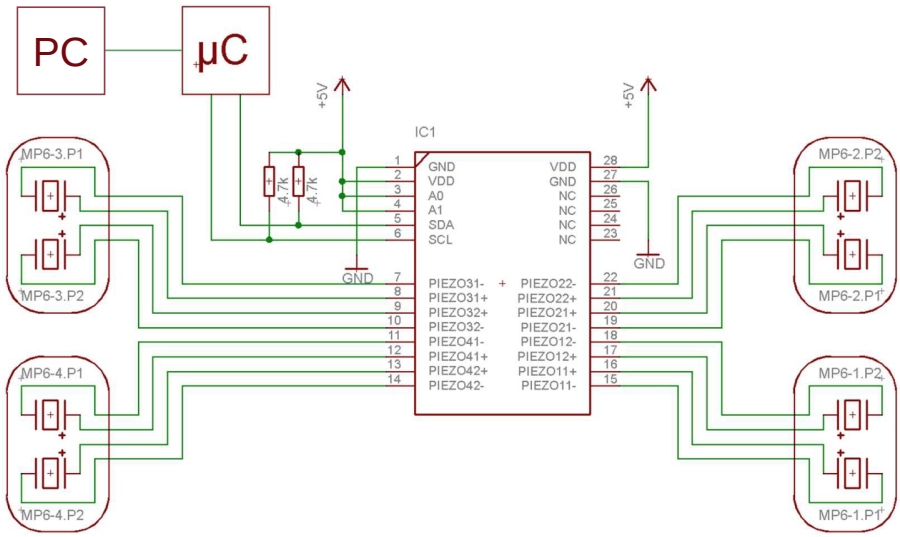


Figure E.1: Piezoelectric pump driver hookup schematic for 4 pumps

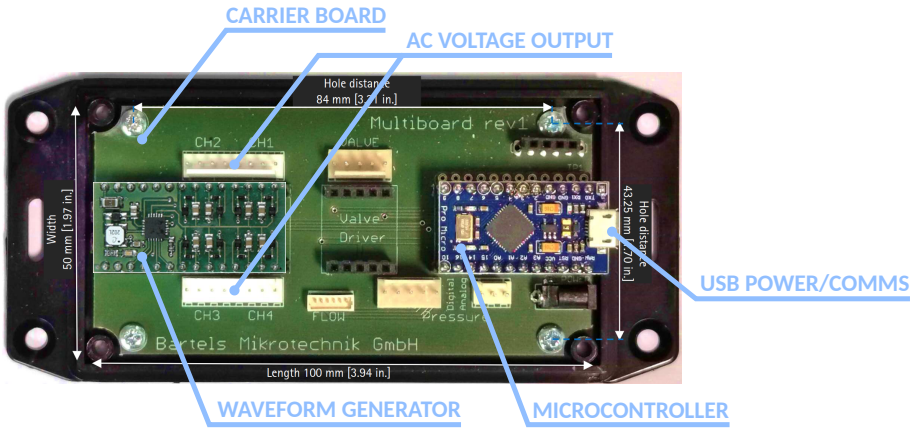


Figure E.2: Piezoelectric pump driver PCB assembly