

# Real-time Control of Solute Plume in Closed Conduit Flow

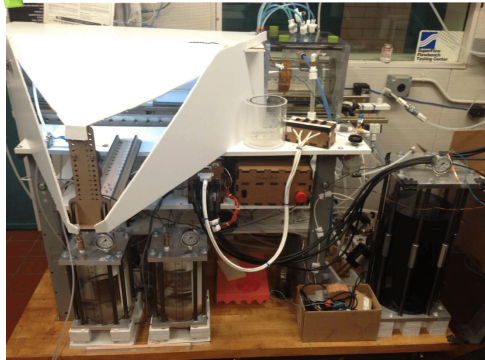
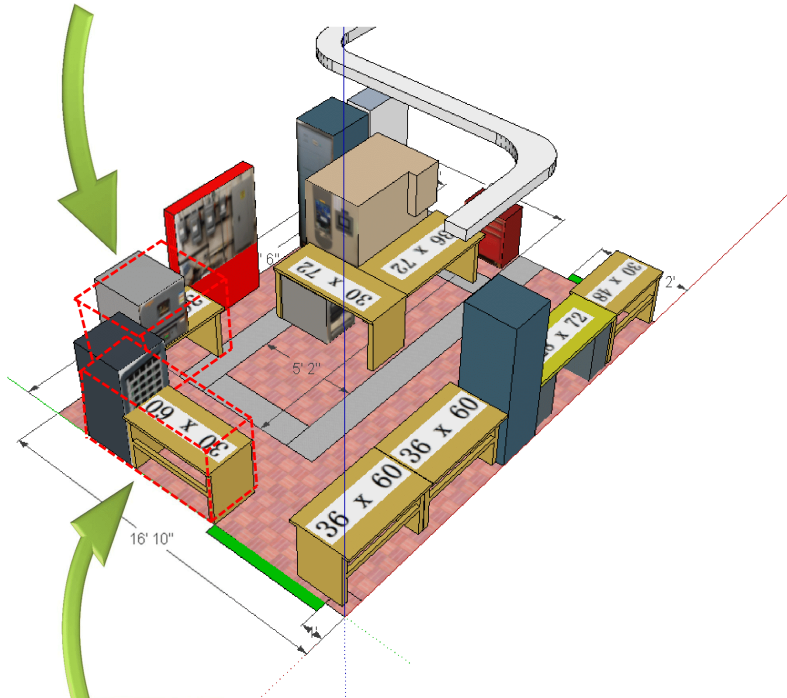
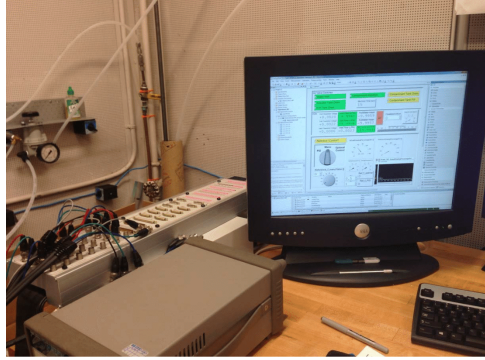
by

Boyun Wang

A dissertation submitted in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
(Mechanical Engineering)  
in The University of Michigan  
2015

## Doctoral Committee:

Professor Anna G. Stefanopoulou, Co-Chair  
Professor Nikolaos D. Katopodes, Co-Chair  
Professor Aline J. Cotel  
Professor Brent Gillespie  
Assistant Research Scientist Jason B. Siegel



© Boyun Wang 2015  
All Rights Reserved

This thesis is dedicated to my family members; My mother Ping, my father Ziyan,  
my grandparents Dan, Hebing, Yinglian, Yumao. And to Xiaozhi and Rudy.



## ACKNOWLEDGEMENTS

Life is a closed loop feedback control system, which is a cycle of seeking, receiving and giving. I cannot possibly acknowledge all those who have helped, guided and inspired me within this endless magic cycle. This list, therefore, is very incomplete.

My principal thanks go to all my family members for their support and love over the years. I'd like to especially thank my parents for giving me the right amount of guidance at the right time. They also gave me the freedom to choose my field of study, in which I enjoyed my life and study.

I would also like to express my sincere appreciation to my committee members: Professor Anna Stefanopoulou, for her endless and patient support from onset of my graduate career and life. She helped me not only from an academic perspective, but also on perfecting my character and building my inner strength; Professor Nikolaos Katopodes, for his invaluable advise and inspiration on my research. He also shared many his own stories, which inspired my understanding of life. I wish the book he is currently writing becomes a big success; Professor Brent Gillespie, for giving me much engineering experience while I was working in his lab, where I decided to pursue higher academic degree; Dr. Jason Siegel, for his countless help in the machine shop, in hardware design, in software programming, in learning Li-ion battery basics and etc; Professor Aline Cotel, for her insight, comments and encouragement on my work, especially on the fluid dynamic mathematical modeling.

I am grateful to my colleagues and friends. April Warnock and Sara Rimer, who we have worked together in this research project. Also to Xinfan Lin, Shankar Mohan,

Youngki Kim, Nassim Samad, Ying Wang, Huan Fu, Toyoaki Matsuura and Hector Perez who gave me much happiness and help in the amazing battery control lab. Bai Zou, Mathieu Davis, Matt Bennett, Zhichen Zhao, Qi Zhang, Fangwei Gu, Gang Su, Yifanyang Gong, Zhaori Cong, Kevin Kuo, Yizhou Fan, Bangxin Hu and Ye Tian made my off campus life rich and colorful. Please forgive me if I've forgotten anyone. I appreciate you all.

Finally, this work was supported by the National Science Foundation (NSF) through the Automotive Research Center. I appreciate that Professor Anna Stefanopoulou and Professor Nikolaos Katopodes gave me this tremendous research topic.

I would always keep in mind all those who have helped me throughout the years. I sought and received so much knowledge and help, and I would always be willing to give back.

Boyun Wang

May 2015

# TABLE OF CONTENTS

DEDICATION . . . . .	ii
ACKNOWLEDGEMENTS . . . . .	iii
LIST OF FIGURES . . . . .	viii
LIST OF TABLES . . . . .	xiii
LIST OF APPENDICES . . . . .	xiv
LIST OF ABBREVIATIONS . . . . .	xv
ABSTRACT . . . . .	xvi
<b>CHAPTER</b>	
<b>I. Introduction . . . . .</b>	<b>1</b>
1.1 Flow Control Using Boundary Actuations . . . . .	1
1.2 The Control Algorithm . . . . .	3
1.2.1 Optimal Control . . . . .	3
1.2.2 Feedback Control . . . . .	5
1.2.3 Discrete Time System . . . . .	6
1.2.4 Adaptive Control and Sensitivity Analysis . . . . .	7
1.3 Methods for Predicting Contaminant Spreading . . . . .	7
1.4 Scope, Objective and Significance . . . . .	9
1.5 Thesis Outline . . . . .	14
<b>II. The Real-time Control Architecture . . . . .</b>	<b>18</b>
2.1 Control Problem Order Reduction and Mathematical Formulation . . . . .	21
2.2 The Real-time Control Architecture and Controller Methodology	23
2.2.1 The Real-time Optimal Controller . . . . .	26
2.2.2 The Output Feedback Controller . . . . .	32

2.2.3	Mathematical Model Parameter Adaptation Scheme	33
2.2.4	Summary . . . . .	36
2.3	Conclusion . . . . .	37
<b>III.</b>	<b>The Prototype Experimental Setup . . . . .</b>	<b>38</b>
3.1	System Diagram . . . . .	38
3.2	Boundary Port Control . . . . .	44
3.3	Signal Processing and Prototype Repeatability . . . . .	45
3.3.1	Mode One - Steady Flow Experiment . . . . .	45
3.3.2	Mode Two - Transient Flow Experiment . . . . .	46
3.4	Worst Case Test and Repeatability . . . . .	48
3.4.1	Worst Case Scenario Experiment . . . . .	48
3.4.2	Experiment Repeatability . . . . .	50
3.5	Prototype Limitations and Summary . . . . .	51
<b>IV.</b>	<b>The FEA Model and CFD Simulation Setup . . . . .</b>	<b>53</b>
4.1	The 2D and 3D Finite Element Model . . . . .	53
4.2	CFD Software Environment Parameters Setup . . . . .	54
<b>V.</b>	<b>Mathematical Modelling for Real-time Control . . . . .</b>	<b>56</b>
5.1	The APT Model - Particle Location Prediction for Constant Boundary Control . . . . .	58
5.1.1	Modelling Technique . . . . .	59
5.1.2	APT Model Properties and CFD Validation . . . . .	64
5.1.3	APT Model for the 3D pipe . . . . .	67
5.2	The VPT Model - Particle Trajectory Approximation for Tran- sient Laminar Flow . . . . .	77
5.2.1	Modelling Technique . . . . .	78
5.2.2	VPT Model Parameterization . . . . .	81
5.2.3	VPT Model validations and extension . . . . .	83
5.2.4	VPT Model for Pipe of Different Flow Rates and Geometries . . . . .	91
5.2.5	Advantages of the VPT Model . . . . .	93
5.2.6	3D VPT Model for the Prototype . . . . .	96
5.3	Summary . . . . .	101
<b>VI.</b>	<b>Controller Implementations Using the Prototype System and CFD Method . . . . .</b>	<b>103</b>
6.1	The Optimal Controller . . . . .	105
6.1.1	Optimal Control Using the APT Model . . . . .	107
6.1.2	Optimal Control Using the VPT Model . . . . .	110

6.1.3	Summary . . . . .	126
6.2	The Feedback Control . . . . .	126
6.3	VPT Model Parameter Adaptation . . . . .	129
6.4	Summary . . . . .	136
<b>VII. Summary and Future Directions . . . . .</b>		<b>137</b>
7.1	Contributions . . . . .	137
7.2	Future Direction . . . . .	140
7.2.1	Apply 2D Control Algorithm in the 3D Pipe Problem	140
7.2.2	Extending the Mathematical Modelling Techniques .	143
7.2.3	The Prototype System . . . . .	145
7.2.4	Source Inversion Problem . . . . .	145
<b>APPENDICES . . . . .</b>		<b>147</b>
<b>BIBLIOGRAPHY . . . . .</b>		<b>182</b>

## LIST OF FIGURES

### Figure

1.1	The chemical cloud detection and elimination flow control problem.	10
1.2	A passenger terminal with the presence of a hazardous chemical cloud.	11
1.3	The real-time control loop for the chemical cloud elimination problem.	12
1.4	A 2D and a 3D representation of the measurement-control segment.	13
1.5	The thesis outline. . . . .	15
2.1	The overall real-time control architecture . . . . .	19
2.2	Solving the 3D control problem using the particle trajectory projection on the 2D symmetric plane. . . . .	21
2.3	The 2D measurement-control segment geometry quantities and notations. . . . .	22
2.4	The overall control architecture for the contaminant elimination problem . . . . .	24
2.5	The control architecture for controlling the contaminant propagation in the single measurement-control segment. . . . .	26
2.6	The motivation of using an optimal controller demonstrated by prototype experiments. . . . .	28
2.7	The motivation of using an optimal controller demonstrated by finite element analysis . . . . .	29
2.8	The model reference optimal controller. . . . .	30

2.9	The output feedback controller . . . . .	33
2.10	The mathematical model on-line adaptation process system block diagram. . . . .	34
3.1	The front view of the prototype . . . . .	39
3.2	The prototype ink injection unit . . . . .	40
3.3	The prototype boundary port installation. . . . .	41
3.4	The prototype flow system diagram. . . . .	43
3.5	The prototype hydraulic pump step response. . . . .	44
3.6	The prototype image processing steps for steady flow scenario. . . . .	46
3.7	The prototype image processing steps for transient flow scenario. . . . .	47
3.8	The prototype worst case scenario experiment - contaminant removal. . . . .	49
3.9	The prototype repeatability test. . . . .	50
3.10	The prototype repeatability when the ink cloud is partially removed. . . . .	51
4.1	The finite element model for CFD simulations. . . . .	54
4.2	The CFD solver settings used in the <i>Ansys-Fluent</i> software environment. . . . .	55
5.1	Notations for the pipe measurement-control segment. . . . .	57
5.2	The modelling principle for the 2D APT model. . . . .	60
5.3	CFD validation of the flow phenomenon that is implied by the APT model. . . . .	65
5.4	APT model validation using CFD simulated results. . . . .	66
5.5	Steady flow CFD simulation results showing the propagation future of the particles in the 3D pipe. . . . .	68
5.6	The 3D APT model integration region on the 3D pipe upstream inlet boundary. . . . .	70

5.7	The 3D APT model integration area approximated by combining a circle and a segment. . . . .	71
5.8	The furthest particle that can be removed by $U = 0.5$ boundary drawing action. . . . .	73
5.9	The 3D APT model compared with CFD simulated results. . . . .	75
5.10	CFD simulated results illustrating the 3D APT model can be applied for flow of different flow rate $Q_{in} - 1$ . . . . .	76
5.11	CFD simulated results illustrating the 3D APT model can be applied for flow of different flow rate $Q_{in} - 2$ . . . . .	77
5.12	The 2D pipe region enclosed by four boundaries defined for VPT model development. . . . .	78
5.13	Approximation of the 2D pipe bottom boundary velocity profile using a Gaussian function. . . . .	80
5.14	VPT model parameterization using CFD simulated particle spatial trajectories. . . . .	82
5.15	VPT model parameterization using CFD simulated flow velocity field along the particle trajectories. . . . .	83
5.16	3D visualization of the entire velocity field approximated by the VPT model. . . . .	84
5.17	VPT model validation using CFD simulated results for steady flow .	85
5.18	VPT model validation using CFD simulated results using square wave pulse boundary control. . . . .	86
5.19	VPT model validation using CFD simulated results using sine wave boundary control. . . . .	87
5.20	Additional VPT model validations using CFD simulated results using different sine waves for boundary control. . . . .	88
5.21	VPT model validation for a pipe with three boundary ports. . . . .	89
5.22	VPT model for different flow conditions and pipe geometries. . . . .	92
5.23	4D empirical lookup table for the 2D flow problem . . . . .	95



5.24	3D VPT model tuning - velocity field . . . . .	98
5.25	3D VPT model tuning - particle trajectories . . . . .	99
5.26	3D VPT model vs Prototype experiments . . . . .	100
6.1	The overall control architecture. . . . .	103
6.2	Monotonic relationship between the control magnitude $U$ and the output $y_f$ described by the APT model. . . . .	109
6.3	2D CFD parametric study showing that higher control strength for shorter period of time is more efficient than spreading out the control effort in space and time . . . . .	113
6.4	Proposed method for validating the optimality of using square wave shaped boundary control $U(t)$ against other wave forms . . . . .	114
6.5	The optimization iteration logic for solving the minimization problem. . . . .	116
6.6	The optimization computing speed requirement for implementing the optimal control algorithm in real-time. . . . .	118
6.7	The optimization iterations and results based on the 2D VPT model. . . . .	119
6.8	2D CFD validation of the optimal control strategy solved by the VPT model based optimization. . . . .	120
6.9	Reading the boundary port location from the camera image. . . . .	122
6.10	The optimization iterations for solving the 3D VPT model based minimization problem. . . . .	123
6.11	Prototype implementation and validation of the optimal control strategy solved by using the 3D VPT model. . . . .	124
6.12	Optimal control timing error caused by the size of the color cloud and the image processing algorithm. . . . .	125
6.13	The output feedback controller using the VPT model. . . . .	127
6.14	The feedback iterations and the feedback gain which is determined by the VPT model gradient. . . . .	128

6.15	The absolute sensitivities of the 3D VPT Model Parameters. . . . .	133
6.16	The relative sensitivities of the 3D VPT Model Parameters. . . . .	134
6.17	VPT Model parameters online tuning results. . . . .	135
6.18	Validation of the VPT Model parameter online tuning results. . . . .	135
7.1	The removed particles and the remaining ones . . . . .	141
7.2	The top most particles that are removed by the boundary port flow showing similar $y_{ini}$ s. . . . .	142
7.3	The concept for problem order reduction from 3D to 2D sub-problems.	143
7.4	The APT model proposed for a 2D pipe network. . . . .	144
7.5	The VPT model that can potentially approximate the contaminant plume shape. . . . .	145
A.1	dSPACE hardware loaded with the simulink program for valve control.	149
A.2	dSPACE hardware loaded with the simulink program for receiving flow meter and pressure sensor data. . . . .	150
A.3	dSPACE hardware loaded with the simulink program for controlling the water pump flow rate. . . . .	150
A.4	The dSPACE user interface on the host PC. . . . .	152
A.5	The simulink program that links the live camera to the matlab envi- ronment . . . . .	153

## LIST OF TABLES

### Table

3.1	Flow System Parameters and Notations . . . . .	40
3.2	Flow System Diagram . . . . .	42
4.1	Flow System Parameters and Notations . . . . .	54
6.1	Parametric study $U(t)$ parameters . . . . .	113
6.2	prototype feedback control implementation for $y_r = -3\text{cm}$ . . . . .	129

**LIST OF APPENDICES**

**Appendix**

- A. Prototype Real-time Control Program Implemented via dSPACE and Matlab . . . . . 148
- B. VPT Model Matlab Code . . . . . 163
- C. VPT Model Based Optimization Algorithm . . . . . 167
- D. VPT Model Sensitivity Function . . . . . 173
- E. VPT Model Parameter Adaptation . . . . . 179

## LIST OF ABBREVIATIONS

- CFD** Computational Fluid Dynamics
- APT model** Algebraic Particle Tracing model
- VPT model** Velocity Field Particle Tracking model
- 2D** 2-dimensional
- 3D** 3-dimensional
- FEA** Finite Element Analysis
- ODE** Ordinary Differential Equation
- PDE** Partial Differential Equation

# ABSTRACT

Real-time Control of Solute Plume in Closed Conduit Flow

by

Boyun Wang

Chair:

Anna G. Stefanopoulou - Mechanical Engineering

Nikolaos D. Katopodes - Civil & Environmental Engineering

The mitigation of accidental toxic chemical releases in a flow system such as a passenger terminal, a tall building or a municipal water channel is an important study area. The real-time, automatic detection and elimination of a chemical cloud is vital to protect human lives against threats. The control of the chemical cloud propagation (and finally elimination) in the aforementioned flow systems is achieved by forced tributary flow patterns that are produced by distributed boundary actuators such as the ventilation system of a terminal. Given suitable chemical sensors, the time of action and intensity of each actuator (the boundary control strategy) can be determined based on the location of the chemical plume. In the past several decades, using computational fluid dynamics (CFD), studies on the boundary actuators have proved the feasibility of controlling chemical cloud propagation in a channel, at least in theory. However, the success of a real-time control mechanism depends on whether a control strategy can be determined fast enough to allow its implementation. Due

to the intense computational effort associated with CFD methods, few attempts have been made to demonstrate real-time chemical cloud detection and elimination in practical applications.

This dissertation introduces an approximate but very fast plume control method. A prototype realization of the real-time control system is also demonstrated for controlling a contaminant plume propagation in a circular pipe. The contaminant plume is modeled as a collection of small massless particles that are passively subject to the advection of the ambient flow. In addition, incompressible laminar flow is assumed for simplicity. A complete flow control system is developed, including a real-time control architecture, based on two novel mathematical models that approximate particle propagation and enable the real-time controllers.

The prototype pipe system resembles the major components in the chemical cloud elimination flow control problem. The system includes a video camera that monitors the cloud propagation, a boundary port that produces tributary flow to manipulate the plume path, and a computing station that processes the camera images and determines the control strategy in real-time. The presented work demonstrates the feasibility of implementing flow control in a real world scenario with the proposed novel mathematical models, which predict the cloud propagation fast enough to implement a real-time control. The real-time control architecture consists of a feedforward optimal controller, an output feedback controller and an adaptation process. The feedforward controller employs a mathematical model that determines the boundary control strategy and moves the cloud from its initial location in the flow upstream region to the desired final location in the downstream region. The feedback controller adjusts the control strategy based on the measured error of the cloud's final location. The model adaptation algorithm tunes the mathematical model towards higher fidelity. Compared to traditional CFD methods, the proposed mathematical models reduce the computational time by more than 99%.

The successful prototype implementations elevate the state of the art of real-time flow control to real world scenarios. The proposed mathematical models introduce an efficient tool to link classical control algorithms with plume control applications in real time. This will ultimately allow the control of a hazardous chemical cloud in critical infrastructure systems.



# CHAPTER I

## Introduction

Mitigating accidental releases of hazardous chemicals in buildings, transportation tunnels or water supply systems, can reduce the threat to human life. For example, by implementing carefully designed boundary control strategy using the air blowing or drawing of the ventilation system, the propagation of the hazardous chemical cloud in the building can be manipulated in order to clear evacuation path, to produce more evacuation time or to eliminate the chemical cloud from the building. Such contaminant control system involves automatic sensing of the chemical cloud, predicting the contaminant propagation in real-time, and implementing controls by actuators that are usually installed on the system's boundary walls. Finding the solution to this contaminant control problem is an emerging research area. The realization of this contaminant control system addresses topics from different fields of study. For example, durable sensor fabrication for a particular chemical substance, boundary control algorithm development, and methods for predicting contaminant spreading. This dissertation emphasizes the latter two topics.

### 1.1 Flow Control Using Boundary Actuations

Over the past decades, a great deal of research has been conducted on using boundary actuations (blowing or drawing fluid) to manipulate the physical quantities

of interest, for example, the mixing (represented by turbulence) of two fluid substance, the boundary drag force, or the evolution of a solute plume. These studies provide the theoretical demonstrations showing the feasibility of using boundary actuations for the flow control problem. Enhancing mixing in channel flow using boundary feedback control was demonstrated by *Aamo et al. (2003)* and *Balogh et al. (2005)*. In their work, by implementing boundary control actuations, effective mixing result is obtained by consuming small amount of boundary control effort. Studies in boundary drag reduction also contribute to the knowledge of the boundary control effectiveness. Numerical experimentation has demonstrated about 20 ~ 25% drag reduction in turbulent flow (*Babcock et al. (1996)* and *Lee et al. (1997)*). Several other studies on drag reduction includes *Choi et al. (1994)* and *Hammond et al. (1998)*. The topics of controlling the evolution of a solute plume is closely related to the presented work. Minimizing a contaminant release impact in a vertically mixed river was discussed by *Piasecki and Katopodes (1997)* using an optimization method. A relatively complete system for detection and elimination of a chemical cloud using boundary blowing and suction was developed by *Wu and Katopodes (2006)*. Their work demonstrated the feasibility of implementing real-time flow control in a real world situation, where only limited number of sensors and boundary actuators can be installed. In a more recent study, *Alvarez-Vázquez et al. (2009)* demonstrated the purification of a polluted river section by injecting clean water at an optimized rate.

To the author's knowledge, the aforementioned studies were based on theoretical analysis and numerical simulations. Due to the intense computational effort associated with the numerical computing method, few attempts had been made to demonstrate these flow control applications in the real world scenario where real-time control is considered. The computational effort is mainly consumed by two processes. The first is predicting the contaminant spreading given a boundary control strategy. The second is the computing associated with the control algorithm. In the contami-

nant control problem, the control algorithm uses the predicted contaminant spreading future to determine the boundary control strategy in response to the observed contaminant location. This dissertation addresses both the problem of efficiently predicting the contaminant spreading and the problem of implementing control algorithms in real-time.

## 1.2 The Control Algorithm

To solve the aforementioned flow control problems, different control algorithms and methods have been proposed and applied in virtual experiments via simulated studies in the past. To solve the boundary control problem, optimal control and feedback correction techniques have been used in the previously mentioned studies. Adaptive methodology is also investigated via simulations, such as the drag reduction problem (*Babcock et al. (1996)*).

### 1.2.1 Optimal Control

Optimal control relies on mathematical optimization that can be used typically offline to derive control strategies for a system given control objectives and system constrains. The general optimal control theory has been developed and discussed in many books. A well organized introduction on optimal control with engineering application examples is written by *Evans (2005)*. Some common optimal control applications include minimizing fuel consumption during a soft landing of lunar lander (*Bennett et al. (1964)*, *Meditch (1964)*, *Shijie and Jianfeng (2007)*) and minimizing the time to bring a pendulum to rest (*Bryson (2002)*, *Åström and Furuta (2000)*). The issue of optimal control in fluid dynamics was addressed by *Abergel and Temam (1990)*. For the contaminant control problem discussed in this dissertation, the optimal control problem is formulated as minimization of the total fluid removed by the boundary control while bringing the solute cloud to a certain location or completely

remove from the stream.

The formulation of the optimal control problem requires four parts (*Becerra (2008)*): a mathematical model of the system that is to be controlled, a specification of the performance index which is usually referred to as the cost function, a specification of boundary conditions and constraints, and a statement of the free variables that are allowed to change during optimization. One widely used mathematical model describing fluid dynamics is the Navier-Stokes Equations (*Acheson (1990)*). Some other methods for modelling fluid dynamics will be reviewed later in Section 1.3. The performance index is formulated depending on the control problem. For example, the turbulence represents the performance of fluid mixing in *Aamo et al. (2003)*. The specification of the boundary conditions or constraints describe the mathematical space that the optimal solution exists within. In the boundary flow control problem for example, the constraints could be the physical limitation of the boundary actuation flow rate or the maximum fluid removal. Finally, the performance index is optimized by changing the free variables. For example, the thrust force used to decelerate the lunar lander during the landing process.

A success of optimal control for turbulent flow was demonstrated by *Choi et al. (1993)*. A procedure was developed to reduce the mean-square velocity gradient, which is the cost function for their study. The existence theory of optimal control for viscous flow was developed by *Fattorini and Sritharan (1992)*. This dissertation, however, emphasizes the control implementation rather than mathematical theory. The advantage of the optimal control method is that it can be applied in a variety of types of fluid dynamic governing equations or the flow control problems. The optimization cost function can be formulated for different boundary control objectives. In the contaminant control problem presented in this dissertation, the optimal control aims to reduce the energy consumption of the boundary control actuations.

### 1.2.2 Feedback Control

Feedback control is a widely used control method that refers to the control situation when some measured quantities from the system output are used to make adjustment of the system inputs. The feedback control is frequently used in many fields of study such as economic investment theory (*Soros (2008)*), biology (*Hellman et al. (2007)*), climate (*Deser et al. (2000)*) and many mechanical and electrical systems such as vehicle control.

The feedback method used in this study is the output feedback, which adjusts the system input using the error between the measured system output and the desired one. One most commonly used feedback law is to tune the system input magnitude by the amount of the output error scaled by a constant feedback gain. The output feedback for a linear system is well established and has been discussed in many text books such as the *Feedback System* by *Aström and Murray (2010)* and *PID control* by *Åström and Hägglund (2006)*. A good reference for linear system control is written by *Williams and Lawrence (2007)*. In this dissertation, however, the mathematical model proposed for approximating fluid dynamics (Chapter V) is nonlinear. The techniques to handle nonlinear system are more rigorous and are usually not general. Some relatively generic method includes Lyapunov stability theory (*Kalman and Bertram (1960)*, *Khalil (1995)*) and limit cycle approach (*Strogatz (2014)*).

For the specific problem of contaminant control in pipe investigated in this study, the physical quantities to be controlled is the spatial location of the contaminant. The nonlinear dynamic behavior of the contaminant movement trajectory is observed to be smooth in both space and time. Therefore, local linearization of the nonlinear system (*Bretscher (2013)*) is applied in this study so that control methods based on linear system can be performed. Linearization is a common approach to apply feedback control for a nonlinear system. The typical example of a successfully nonlinear control using a linearized system is the inverted pendulum problem (*Mirza and Hussain*

(2000)).

The output feedback control implemented in this study uses the linearized nonlinear mathematical model to approximate the local dynamic behavior of system. The gradient of the linearized model is used as the feedback gain to adjust system input. For a linear discrete time system (the discrete time system will be reviewed in the next section), this gradient based feedback law produces a dead-beat control problem (*Kailath (1980)*). The dead-beat controller brings the system output value to the desired value within the smallest number of time steps. The dead-beat control for a nonlinear system is still an open question. However, some notable results have been shown for simple nonlinear system. For example, global stability of the closed-loop control for a Single Input and Single Output (SISO) discrete-time nonlinear system was demonstrated by *Bastin et al. (1999)*. The presented work does not emphasize the mathematical foundation of the nonlinear feedback control, but checks its convergence both numerically and experimentally. Specifically, it is shown that the feedback control convergence is achieved within 3 discrete steps on the prototype system.

### 1.2.3 Discrete Time System

Optimal control or feedback control can be applied for both continuous or discrete time system. The discrete time system refers to the situation when the values of variables (measurements) are defined at distinct separate points in time (*Houpiis and Lamont (1992)*). In the contaminant control problem, the sensors are installed at discrete locations. Although the sensors continuously monitor the occurrence of the contaminant, the propagation of one contaminant cloud is provided in a discrete manner by sensors at different locations down the length of the pipe. Therefore, the control is formulated as a discrete system in Chapter II and VI.

#### 1.2.4 Adaptive Control and Sensitivity Analysis

Adaptive control addresses the problem of unknown or time-varying system (controller) parameters (*Åström and Wittenmark (2013)*). A close to life example of adaptive control is the digital active ear defender, which attenuates high impact sound noise (*Jianyong and Yu (1996)*) and has recently been applied to the ear protectors product used in industry manufacture environment. Adaptive control emphasizes parameter estimation and aims to adjust the parameter values according to measurements. Common adaptive methods include least square estimation (*Hayes (2009)*) and gradient descent (*Avriel (2012)*).

In this study, a mathematical model with six uncertain parameters is proposed for approximating the flow system. The adaptation method is applied to tune the model parameter based on the actual flow system output measurements. However, it is a normal situation that some of the system (model) parameter may be not easy to parameterize to estimate. The parameter estimability can be measured by conducting a sensitivity analysis (*Khalil (1995)*). The sensitivity analysis is a study of how the system output responses to a change of an input or a parameter. If changing a parameter is not influential to a measured output, this parameter is less likely to be estimated successfully by using the measurement of that specific output. Discussed in Section 6.3, sensitivity analysis is performed on the proposed mathematical model before it is determined that which model parameters are to be adapted.

### 1.3 Methods for Predicting Contaminant Spreading

In the problem of contaminant control, predicting the contaminant propagation is essential for designing the boundary control strategy. The spreading of a solute in a viscous fluid is usually modeled by systems of Partial Differential Equations (PDEs), which are solved using methods of Computational Fluid Dynamics (CFD). For real-

time control, a very fast mathematical model is needed for predicting the propagation of the contaminant substance. Despite advances in Central Processing Unit (CPU) clock frequency and faster CFD algorithms, the time required to solve the flow and mass transport equations is still far beyond the requirements of an iterative or a real-time controller. Several other modeling methods have been developed to reduce or to avoid such computational demands. One of the leading methods is the Proper Orthogonal Decomposition (POD) solution (*Lumley (1967)*) for the Navier-Stokes equations. By implementing the POD method, approximately 90% computational effort reduction was demonstrated by *Lieu et al. (2006)*. According to their paper, the CPU time for simulating the air dynamics around an F-16 aircraft in transonic flow was reduced from 1-3 days on a 128-processors computer cluster for the full CFD, to 3.8 hours on a Linux cluster with 15 Pentium 4.0Ghz processors for the POD based model. However, for the iterative, real-time flow control problem considered in this study, an even faster mathematical model is required.

Under ideal flow conditions, also known as potential flow, the flow can be approximated except at a thin boundary layer next to solid walls. The flow in a closed conduit is thus completely described by drawing the streamlines for the irrotational velocity field (*Kirby (2010)*). This approach constructs the flow geometry using basic flow elements (such as a wall element or a fluid sink element) and describes the velocity field. However, potential flow cannot be applied to viscous internal flows, and thus cannot be used for the types of pipe flow considered in this study. At the other end of the spectrum, a method known as Smoothed Particle Hydrodynamics (SPH) (*Monaghan (1992)*) has become very popular for the solution of multi-phase flow in recent years. This method represents the fluid using a set of discrete particles. The fluid dynamics is then approximated by a system of Ordinary Differential Equations (ODE) that are constructed based on the interaction among these particles. However, the SPH method requires a large number of particles to ensure accuracy, thus it is



not suitable for approximating flow phenomena in the context of real-time control.

Currently, there does not exist a rigorous method available that can simulate fluid flow and contaminant transport fast enough to permit real-time control. Thus the models proposed in this work deviate from classical fluid dynamics. These models are inspired by potential flow and the SPH method, and attempt to create a phenomenological approach to modeling plume propagation in a pipeline. The models are not recommended for general fluid flow simulation, but as approximations to the actual flow conditions that should be tuned first with experiments and then can be executed extremely fast. Therefore, repeated simulations within a control code can be completed in only a few seconds, thus allowing the prototype actuation to be implemented in real time.

## 1.4 Scope, Objective and Significance

This study is focused on demonstrating contaminant cloud elimination in an actual prototype system in real time. Therefore, the main tasks addressed in this study include control architecture and real-time controller development, prototype design and fabrication, mathematical model development and real-time control implementation. To the author's knowledge, the literature for contaminant control using boundary actuations is very limited. The aforementioned research was mainly conducted using numerical simulations. Due to the complexity of the flow system and the computational demands of the CFD method, no experimental results have been obtained to demonstrate real-time control of contaminant elimination under practical scenarios. Therefore, this study aims at developing a real-time flow control prototype system that resembles all the necessary components of the contaminant elimination control problem.

The subject of real-time chemical cloud elimination with a predictive control scheme was previously formulated in Fig. 1.1 by *Katopodes* (2009). A chemical cloud

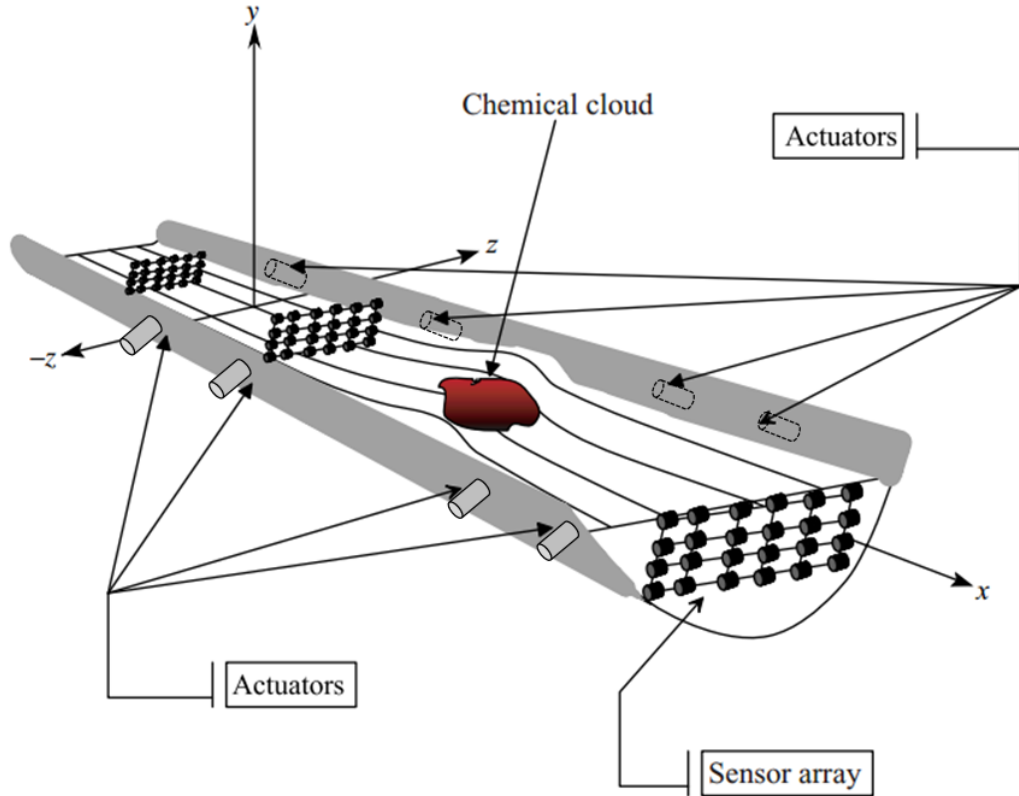


Figure 1.1: The chemical cloud detection and elimination flow control problem. (formulated by Katopodes *Katopodes* (2009))

is present in a section of a channel and needs to be removed by the actuators that are installed on the channel's side walls (pointed by the arrows). The chemical cloud represents an unexpected hazardous release from a point source. The boundary actuators provide blowing or injection of ambient fluid as the control actions with the intent to manipulate the location of the chemical cloud and to finally remove it from the system.

The flow control problem described in Fig. 1.1 involves turbulent open-surface flow, which is very complex to study using a prototype system. Therefore, this study considers a similar but simpler chemical cloud removal problem. A typical passenger terminal, for example, can be represented by the flow system described in Fig. 1.2. A hazardous chemical cloud is suddenly released and needs to be removed strategically

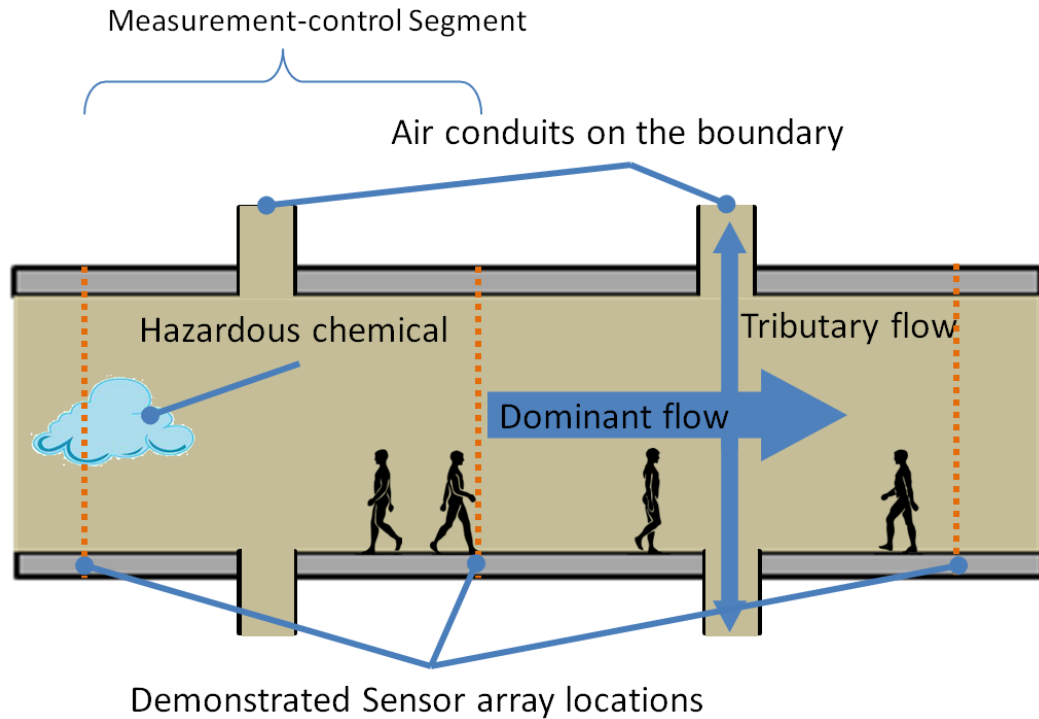


Figure 1.2: A passenger terminal with the presence of a hazardous chemical cloud. The cloud is to be removed by the air conduits if proper chemical detection and control are applied.

by controlling the lateral air flow that is created by a series of air vents on walls of the conduit. The chemical cloud location is measured by strategically installed sensor arrays. The proper boundary control strategy is then computed and implemented in real-time. The chemical cloud detection and control process is repeated until the chemical cloud is removed from the area of concern. Post treatment of the removed chemical is not discussed in this study.

This passenger terminal flow system is characterized by: a long conduit geometry containing an ambient fluid with a dominant flow direction; solid side walls as the boundaries; flow patterns induced by the lateral flows; and repeated measurement-control attempts. These characteristics can be demonstrated by the flow control system defined in Fig. 1.3. Sensor arrays and boundary control ports are installed in alternating patterns, forming several measurement-control segments in series over

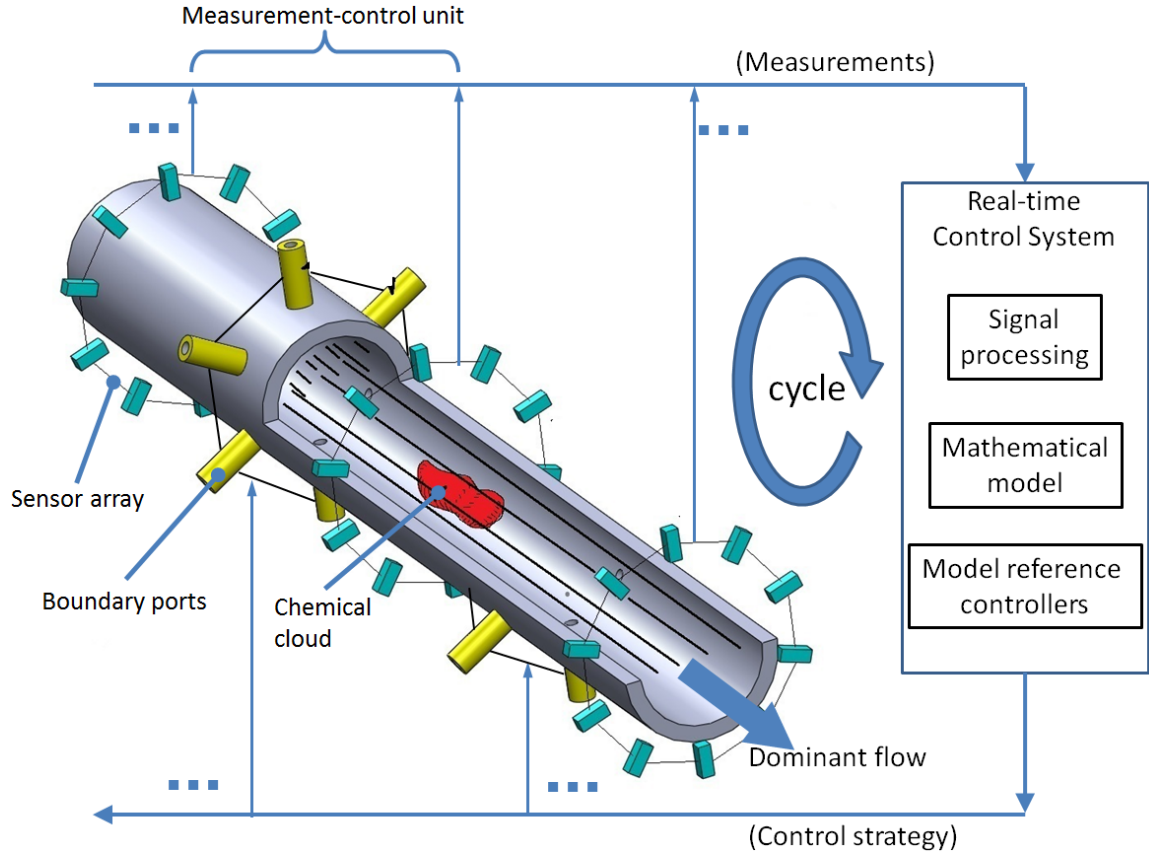


Figure 1.3: The real-time control loop for the chemical cloud elimination problem. The process of contaminant measurement, control strategy optimization and control implementation repeats until the chemical cloud is removed.

the length of the conduit. The real-time control system processes the sensor measurements and computes the control strategy for each boundary port. The control strategy is then implemented by the boundary actuators. The measurement, computing and control cycle is repeated until the hazardous chemical is removed from the flow system.

For the purpose of establishing a realistic starting point for solving this generic conduit control problem, this study emphasizes a simplified sub-problem, in which only one measurement-control segment is considered and only one boundary port is installed. This sub-problem is abstracted as a 2-dimensional (2D) or a 3-dimensional (3D) pipe illustrated in Fig. 1.4. The boundary port is positioned at  $x = x_p$ . Two

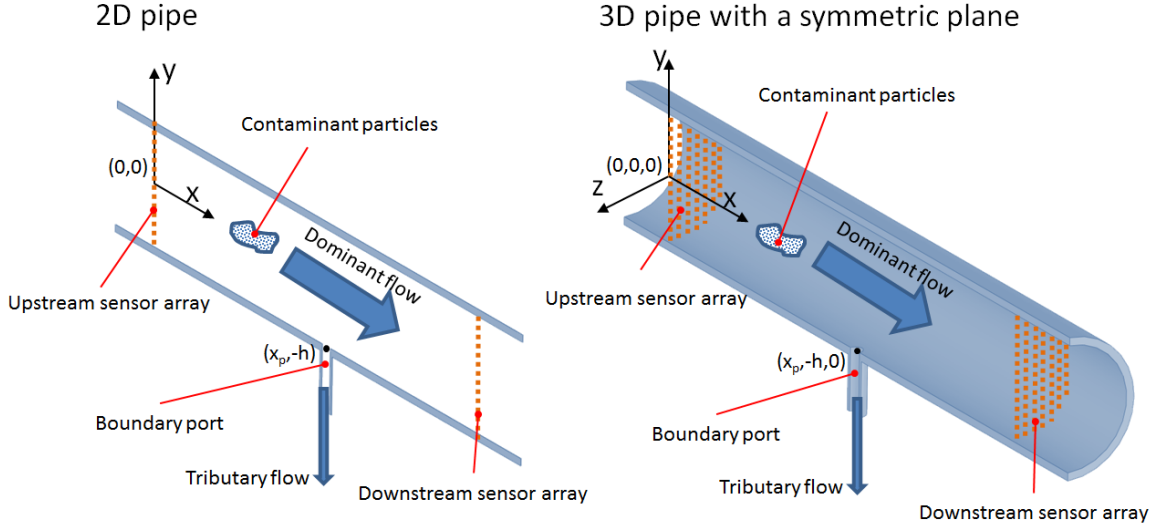


Figure 1.4: A 2D and a 3D representation of the measurement-control segment. The chemical cloud is represented by a collection of independent massless contaminant particles, which are subjected to the advection of ambient fluid.

sensor arrays are assumed in the upstream and downstream flow region, denoted by  $x < x_p$  and  $x > x_p$ . The contaminant plume is represented as a collection of small independent massless contaminant particles, which are passively subject to advection by the surrounding flow motion. This massless particle simplification is valid when the contaminant has similar physical properties when compared to the ambient fluid. In addition, only one type of contaminant particle was considered in this study for mathematical model development simplicity. To remove or to control the location of the contaminant particles, the control algorithm determines the boundary control that should be enforced by the boundary port. For the passenger terminal example shown in Fig. 1.2, the control solution may be trivial by drawing as much air as possible. However, when the boundary conduit drawing strength is limited or no boundary conduit exists near the chemical cloud, effective cloud propagation prediction and a robust control algorithm are needed.

Based on the simplified single boundary port flow control problem, a complete real-time control solution was accomplished in this study. The real-time control sys-

tem was successfully demonstrated via the prototype system, which resembles all the necessary components needed to simulate the real world scenario. The results presented in this dissertation validate the feasibility of implementing a real-time iterative control algorithm in real world situations. The successful outcome of this study elevates the state-of-the-art of real-time flow control, and stimulates relevant real world practices. The main achievements includes the overall real-time control architecture, the prototype pipe system, and two mathematical models that predict the propagation of the solute plume.

Furthermore, the proposed control architecture and the mathematical models can be potentially applied in many other engineering applications. For example, microfluid droplet generation (*Teh et al. (2008)*) and single cell analysis/sorting (*Mazutis et al. (2013)*). In these microfluidic applications, the flow Reynolds number is usually less than 1, which is the ideal low Reynolds number flow condition for implementing the proposed methodology. Although the presented material are mainly proposed for low speed pipe flow (Reynolds number less than around 100), the methodology developed in this study can be potentially applied for situations when the flow speed is very higher and fluid mixing/diffusion can be neglected. For example, for a air terminal of 10 meter in diameter, the flow Reynolds number may exceeds  $10^7$  with 20 m/s wind speed.

Finally, the mathematical models were originally developed for predicting solute propagation future given its initial condition, both models can be used for the subject of contaminant source inversion problem. An review of the source inversion problem is included in the Section 7.2.4

## 1.5 Thesis Outline

The remainder of this thesis is divided into six additional chapters. The thesis structure is shown in Fig. 1.5.

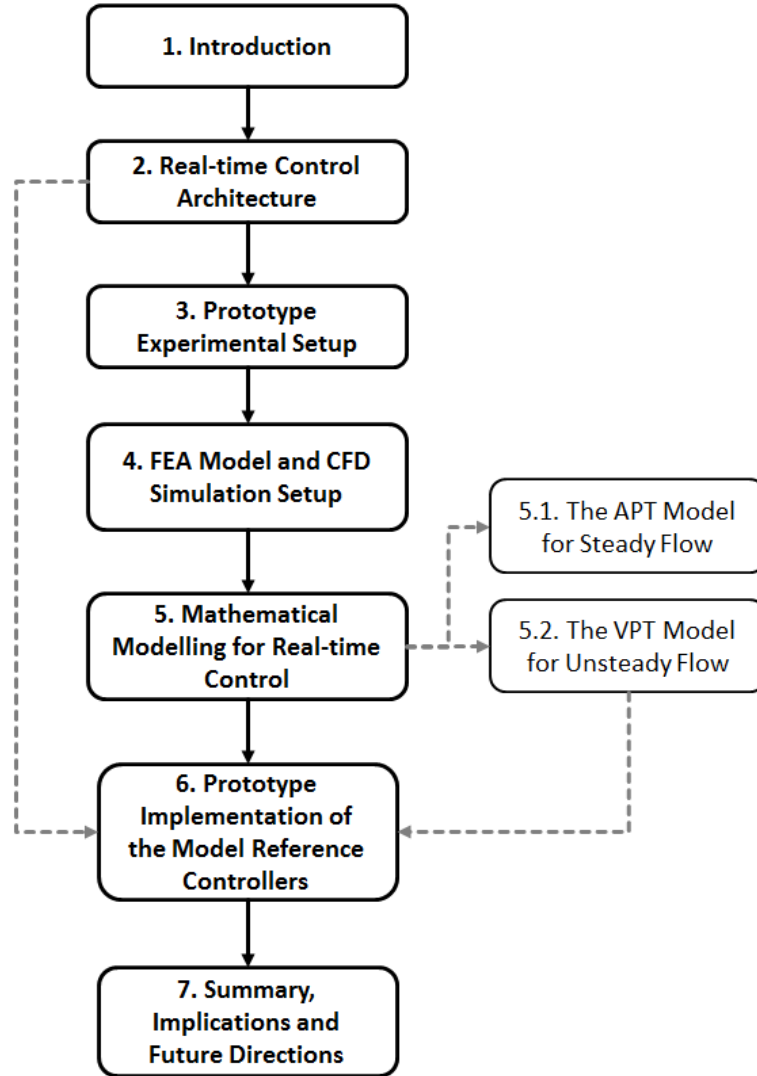


Figure 1.5: The thesis outline.

Chapter II defines mathematically the flow control problem based on the single measurement-control segment geometry. The overall high level real-time control architecture is proposed in this chapter. The control architecture consists of a feed-forward optimal controller, an output feedback controller and a mathematical model adaptation process. The feedforward controller employs the mathematical model to determine the boundary control strategy using the measured particle’s initial location. The feedback controller adjusts the control strategy based on the measurement error. The model adaptation algorithm tunes the mathematical model towards higher

fidelity.

Chapter III illustrates the prototype flow system. The prototype system resembles the basic components required to realize the real world contaminant control system. These components include: a live camera that emulates the sensor arrays; a work station that provides necessary computing power for processing measurements; and running controllers and a dSPACE embedded real-time system for controlling the boundary port flow.

Chapter IV presents the Finite Element Analysis (FEA) model and the Computational Fluid Dynamics (CFD) simulation setup. Full CFD was used as virtual experiments in this study. The mathematical models presented in Chapter V were mainly tuned and developed based on CFD simulation results.

Chapter V presents two fast solvable mathematical models that are used for approximating particle propagation in incompressible laminar pipe flow. These two novel models were named Algebraic Particle Tracing model (APT model) and Velocity Field Particle Tracking model (VPT model), and were used for steady flow and unsteady flow, respectively. Compared to the conventional CFD approach, these two models save more than 99 percent of the computational time and provide adequate accuracy for controller implementations. Therefore, they fit into the time frame of real-time control applications and can be used in the model reference iterative real-time controller. The presented mathematical models establish a convenient tool for applying classical control algorithms to flow control applications. Furthermore, the proposed mathematical models and the corresponding modeling techniques enable fast and easy simulation for flow systems of high complexity, which is further discussed in Chapter VII

Chapter VI presents the actual prototype implementation of the real-time controllers. The success of the prototype validations prove the functionality of the mathematical models and the feasibility of the real-time flow control algorithm.



Finally, Chapter VII discusses future study directions based on the results of this study. The success of this research elevates the state-of-art of flow control to real world practice.

## CHAPTER II

### The Real-time Control Architecture

The generic contaminant elimination problem described in Fig. 1.3 involves controlling multiple measurement-control pipe segments, which have multiple boundary ports in each. It is of interest, however, to establish a realistic starting point for solving this complex control problem. Therefore, this study emphasizes realizing real-time flow control based on one single measurement-control pipe segment that has only one boundary port installed. In addition, the contaminant is simplified as a small massless particle that is passively subject to the advection of the ambient flow. This chapter presents the high level control architecture and the mathematical formulation for this single boundary port particle propagation control problem.

Section 2.1 presents the mathematical representation of the particle propagation control problem. For the single boundary port symmetric pipe geometry, the removal of the particle can be sufficiently described using only the two coordinates that define the symmetric plane. Therefore, the particle propagation trajectory is orthogonally projected onto the geometry symmetric plane. By doing so, the 3-dimensional (3D) control problem is reduced to a 2-dimensional (2D) problem. The flow quantities and the notations are then defined based on this 2D problem.

The real-time control architecture is presented in Section 2.2. The overall control architecture illustrated in Fig. 2.1 is composed of a feedforward controller, a feedback

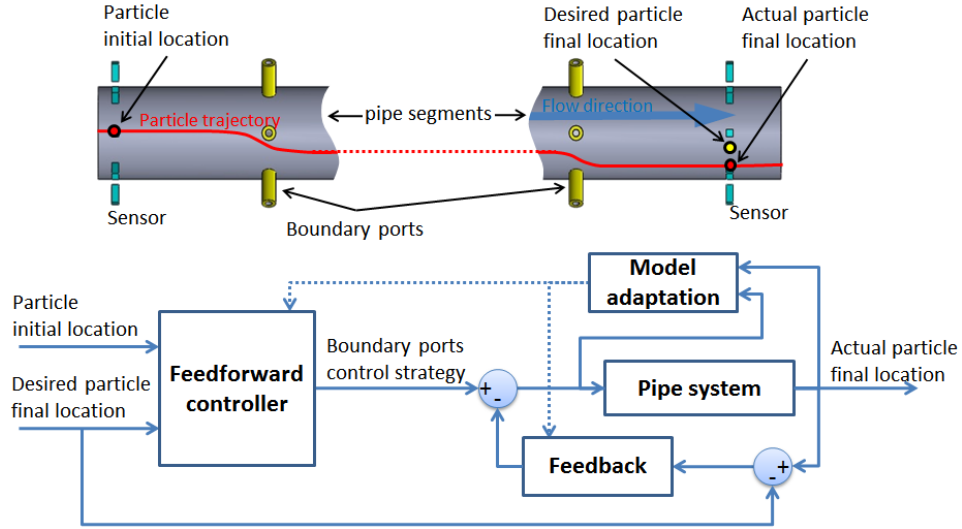


Figure 2.1: The overall real-time control architecture.

controller and a mathematical model adaptation process. The feedforward controller employs a mathematical model that approximates particle propagation to determine the boundary control strategy to be implemented by the following measurement-control pipe segments. The control strategy defines the flow rate profile for each boundary port down the length of the pipe. The feedback controller uses the same mathematical model to compute the appropriate feedback gain that adjusts the control strategy based on the measured error between the desired and the actual particle final location. The model adaptation aims to tune the mathematical model towards higher fidelity.

In this study, the feedforward controller is an optimal controller (Section 2.2.1), which iterates a reference mathematical model to optimize the boundary control strategy. The optimized control strategy aims to achieve the control objective by consuming minimum amount of boundary control effort. In this study, the control effort is quantified as the accumulated fluid exchange through the boundary ports. In addition, the control objective can be defined as the desired contaminant particle location measured by the downstream sensor or as removing the contaminant. In order to im-

plement the optimal controller in real-time, the iterative optimization process needs to be fast enough. This speed requirement is one of the crucial design specification for the mathematical model. In Chapter V, two novel mathematical models that can be solved fast enough for real-time optimal control implementations are presented.

In an actual system, a variety of factors causes error between the actual measured system output and the desired control objective. For example, observed in our prototype system, the density variation of the contaminant color cloud produces unexpected contaminant movement. When the mathematical model, which does not consider this density variation, is employed in the optimal controller, the actual system output by implementing the optimized control strategy will be different from the optimal control set point. Usually, it is not necessary and is difficult to construct a high fidelity mathematical model that captures all the physics. In order to achieve real-time control, we trade off the model fidelity by higher computing speed. At this situation, the output feedback controller discussed in Section 2.2.2 takes into account all the combined error and adjusts the boundary control strategy to drive the system output to the desired objective value. Considering the cloud density variation is potentially the future step to improve the mathematical model.

Finally, the mathematical model adaptation scheme is presented in Section 2.2.3. The model tuning aims to reduce the modelling error based on on-line measurements. A mathematical model of higher fidelity improves the controller performance. For example, using a higher accuracy mathematical model, the optimal controller will find a boundary control strategy closer to the true system optimum. However, due to the flawed experimental data caused by the un-modeled cloud density variation, the demonstration of model adaptation aims to illustrate the functionality of the model adaptation process based on the proposed mathematical model. The proposed model adaptation presents a framework for future studies, when a more accurate prototype system is available or a mathematical model that considers the density variation is

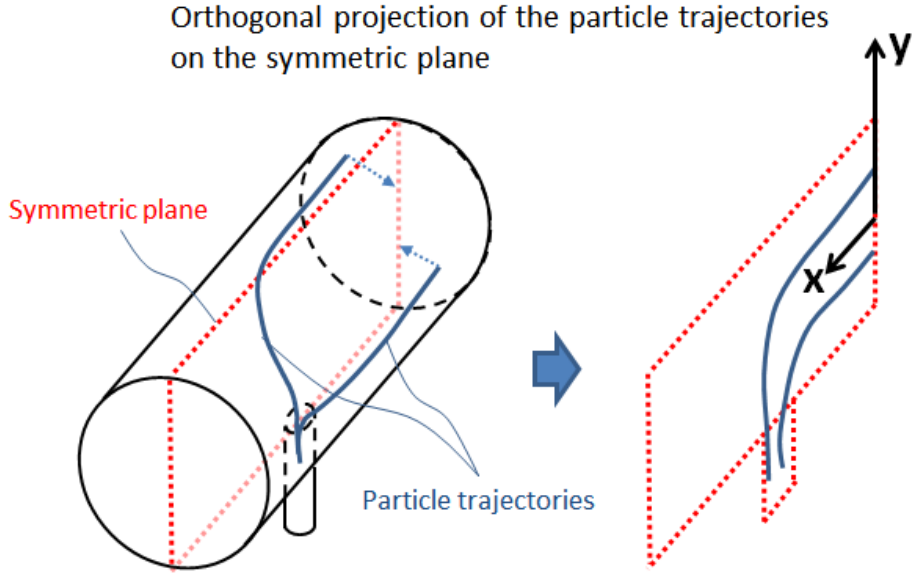


Figure 2.2: Solving the 3D control problem using the particle trajectory projection on the 2D symmetric plane. For the single boundary port geometry, the projected particle trajectory is sufficient to describe particle removal.

created.

## 2.1 Control Problem Order Reduction and Mathematical Formulation

In this study, one major task is to develop a proper mathematical model which approximates the necessary contaminant particle trajectory for controller design. For the single boundary port geometry, the removal of a contaminant particle can be sufficiently described by the particle trajectory that is projected on the pipe symmetric plane as illustrated in Fig. 2.2. Using the coordinate system (originated on the pipe central axle) defined in this figure, when and only when the particle is removed by the boundary port flow, its  $y$  coordinate becomes less than the pipe radius. Therefore, the 3-dimensional contaminant elimination control problem can be solved in a 2-dimensional space.

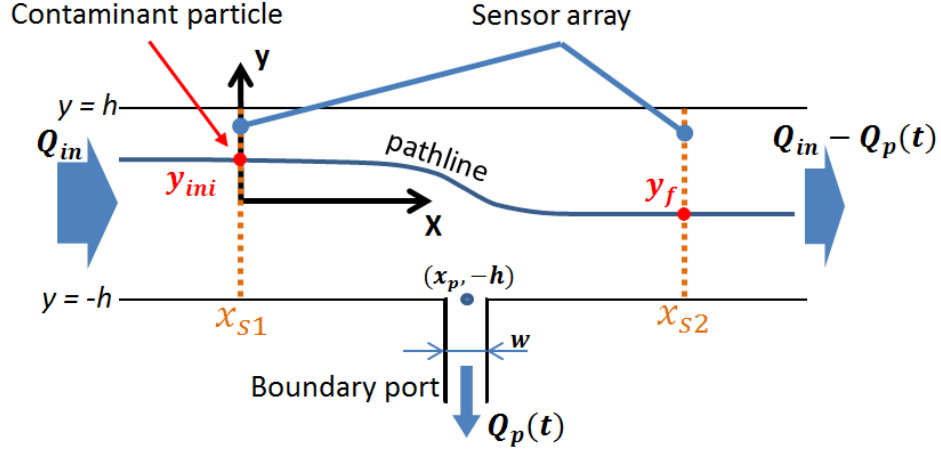


Figure 2.3: The 2D measurement-control segment geometry quantities and notations. The major quantities are: the upstream volume flow rate  $Q_{in}$ , the boundary control volume flow rate  $Q_p(t)$ , the particle initial location  $y_{ini}$  and the particle final location  $y_f$  after the control. The pipe size is  $2h$  in height or diameter. The boundary port is centered at  $x_p$ . The sensors are placed at  $x_{s1}$  for the upstream and  $x_{s2}$  for the downstream.

The geometries and the associated notations for this 2-dimensional geometry is defined in Fig. 2.3. The Cartesian coordinate origin is chosen on the pipe center line and is placed in the very upstream where the flow pattern is not influenced by the boundary port. The  $x$  axis is along the pipe center line. The  $y$  axis is perpendicular to the  $x$ , pointing away from the boundary port. The two virtual sensor arrays, which bound the measurement-control segment, are placed at upstream  $x_{s1} = 0$  and downstream  $x_{s2}$ . The flow is assumed to be fully developed at both sensor locations. The pipe is  $2h$  in height. The boundary port is centered at  $x = x_p$  with its width of  $w$ . The upstream total inlet flow rate  $Q_{in}$  and the boundary port flow rate  $Q_p(t)$  have unit of  $\text{cm}^2/\text{s}$  for a 2D geometry (or  $\text{cm}^3/\text{s}$  for the 3D case in the prototype). The downstream flow rate is directly computed as  $Q_{in} - Q_p(t)$  for incompressible fluid. The upstream sensor array  $s_1$  measures the contaminant initial location  $y_{ini}$ . After the boundary port control  $Q_p(t)$  is applied, the particle moves to its final location  $y_f$  and is measured again by the downstream sensor array  $s_2$ . The particle final location can

be either a new position in the pipe downstream or being removed by the boundary port. It will be shown later in Chapter V that, these two particle final conditions can be mathematically represented as  $-h < y_f < h$  and  $y_f \leq -h$  respectively.

In this study, the  $Q_{in}$  is assumed to be a constant. The  $Q_p(t)$  is the system input. In addition,  $Q_p > 0$  indicates boundary drawing action, while  $Q_p < 0$  indicates boundary injection. In this dissertation, the demonstrated contaminant elimination problem only utilizes  $Q_p > 0$ . The  $Q_p < 0$  condition may need to be considered for a problem of higher complexity. For example, in the case when multiple contaminant particles are to be controlled simultaneously, the control algorithm may wish to assign boundary injections for overall control effectiveness. The utilizing of  $Q_p < 0$  is not investigated in this study, but is briefly explained with the optimal controller in Section 2.2.1. Further more, it will be mentioned in Chapter V and Chapter VI that, the placement of the downstream sensor is not influential to the control problem as long as the sensor is placed in the fully developed laminar flow region. It is obvious that, in the fully developed laminar flow region where the velocity field is purely in  $x$  direction, the contaminant particle  $y$  measurement is not influenced by the location of the downstream sensor.

Although the study was focused on the 2D problem that resides on pipe symmetric plane, the result presented in this dissertation can be potentially applied to the full 3D multiple boundary ports problem directly. The methodology of utilizing the 2D control algorithm on the full 3D problem is later discussed in Chapter VII.

## **2.2 The Real-time Control Architecture and Controller Methodology**

This section presents the high level control system block diagram and the methodologies utilized in each controller. The overall block diagram is defined in Fig. 2.4

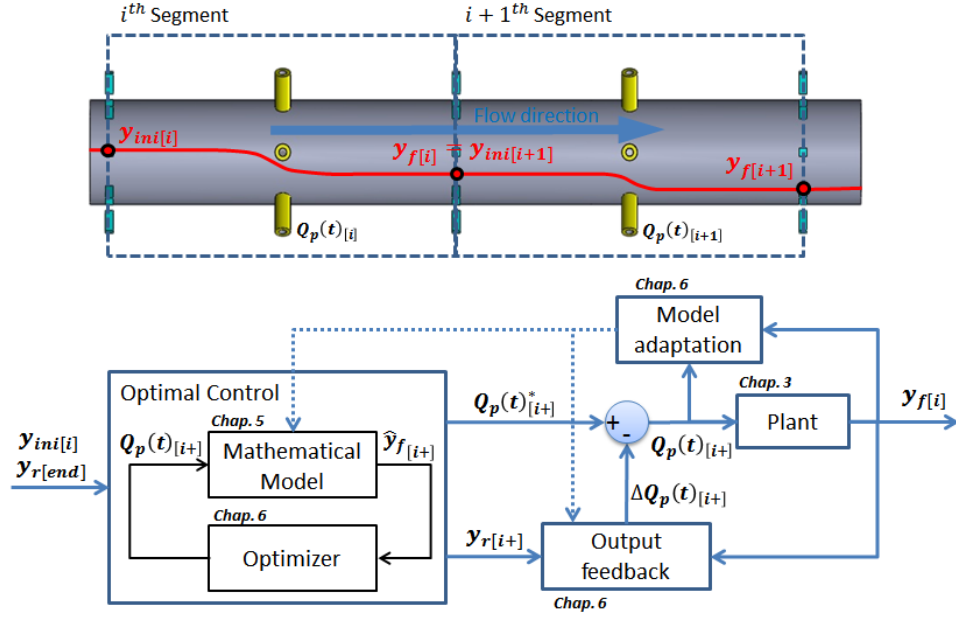


Figure 2.4: The overall control architecture for the contaminant elimination problem, which involves multiple measurement-control segments.

for the general multiple measurement-control segments control problem. The whole system is composed of three major components: the optimal controller, the output feedback controller and the control scheme for mathematical model parameter tuning.

The optimal controller starts with the contaminant particle measurement at the beginning of the  $i^{th}$  segment,  $y_{ini[i]}$ . In addition, the control objective set point is defined as  $y_{r[end]}$ , which is the desired contaminant particle at the end of last pipe segment. It will be later presented in Chapter V that contaminant removal can be mathematical represented as  $y_r \leq -h$ , where  $h$  is the half height of the 2D pipe or the radius of the 3D pipe. Given these two initial parameters, the optimal controller iterates a mathematical model to optimize the boundary control strategy  $Q_p(t)_{[i+]}^*$  for all the following control segments, which is subscript as  $[i+]$ . The \* denotes the optimized control strategy. The general form of the mathematical model used in each



controller is defined as Equation (2.1)

$$\hat{y}_{f[i]} = M(y_{ini[i]}, Q_{in}, Q_p(t)_i), \quad (2.1)$$

which creates the relationship among the particle initial location  $y_{ini}$ , the system flow conditions  $Q_{in}$ ,  $Q_p(t)$  and the system output  $\hat{y}_f$ . The  $\hat{\cdot}$  indicates model predicted quantities. The development of the mathematical model will be presented in Chapter V. The optimized control strategy aims to remove the contaminant particle by consuming minimum amount of boundary control effort. In this study, the control effort is quantified as the accumulated fluid exchange through the boundary ports. In this control architecture, the optimal controller is a feedforward control and keeps rolling forward to check the optimality of the optimized control strategy based on the new measurements. If the measurement  $y_{f[i]}$  is the same as the mathematical model prediction  $\hat{y}_{f[i]}$  solved during the optimization, the optimality holds for the  $i + 1^{th}$  segment and the optimization does not need to be conducted again. Difference between the measurement and the model prediction indicates that error and/or disturbance occurred during the implementation of  $Q_p(t)_{[i]}$ . At this situation, the optimization should be conducted again based on the new measurement. In addition, the new measurement can be utilized in the feedback controller and be used to tune the mathematical model. The feedback controller aims to regulate the contaminant particle's movement to follow the optimized control objectives in the following pipe segments  $\hat{y}_{f[i+]}$ . The model tuning algorithm is introduced to adjust the mathematical model, which is used as the reference in the optimal controller and the feedback controller. Using a mathematical model of higher fidelity, the controller is expected to deliver better performance. For example, the optimized boundary control strategy solved by the optimal controller will be closer to the true system optimum if a more accurate mathematical model is employed.

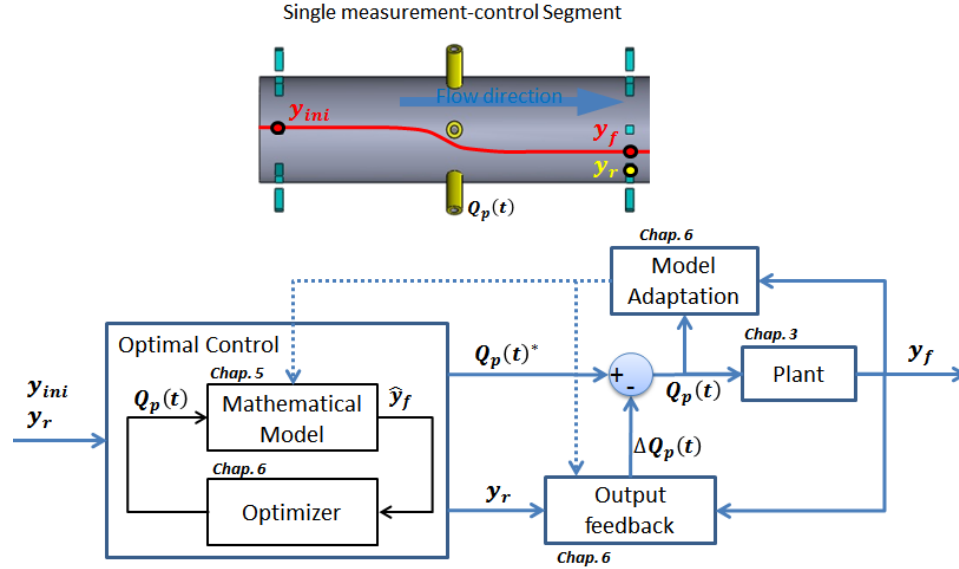


Figure 2.5: The control architecture for controlling the contaminant propagation in the single measurement-control segment.

This study, which presents a realistic starting point to solve above broad control problem, is focused on one single measurement-control segment subproblem. The corresponding control diagram is illustrated in Fig. 2.5. It is obvious that the overall control logic is identical to the control architecture defined in Fig. 2.4. Thus, demonstrating the functionality of the three controllers in the single pipe segment problem supports future study and implementation of the whole pipe system control problem.

The three controllers can operate independently from each other and they are discussed individually in this section.

### 2.2.1 The Real-time Optimal Controller

The worst case scenario prototype experiment demonstrated in Chapter II provides a feasible contaminant elimination solution, which is removing all fluid together with the contaminant. However, when the contaminant is not fully spread out in the pipe, such extreme control action wastes uncontaminated fluid and consumes unnecessary control efforts. In addition, when the boundary actuator is not strong enough

to deliver required control strength to remove all ambient fluid, a strategical distribution of the control effort among multiple measurement-control segments is needed. In this dissertation, the performance of the control strategy is quantified by the volume of removed fluid. The less fluid removed while ensuring contaminant elimination, the better the control strategy performance is. In future work, the energy consumed by the boundary port actuator can be introduced as the secondary measurement of the control performance. In this subsection, the motivation of implementing an optimal controller is first illustrated via prototype experiments and finite element analysis. Then the optimization problem is mathematically formulated.

Two examples shown in Fig. 2.6 and Fig. 2.7 demonstrates the motivation of using an optimized control strategy. These two examples were demonstrated via prototype experiments and 3D finite element analysis respectively. (The finite element analysis model setup is documented in Chapter IV). In both examples, the upstream total flow rate is  $Q_u = 83.33\text{cm}^3/\text{s}$ . The 3D pipe is  $h = 5.08\text{cm}$  in radius. The boundary port is  $w = 1.27\text{cm}$  in diameter. Two different control strategies are compared in each example. Both control strategies aim to eliminate the contaminant using two measurement-control segments. The difference between the two control strategy is that, the first strategy tries to move the contaminant downward as much as possible using the first control segment. While the second control strategy slightly moves the contaminant downward in the first segment.

In both examples, squared wave shaped boundary control action is used. Three variables are used to describe the boundary control action: the flow rate magnitude  $Q_p$ , the square wave starting time  $t_{on}$  and the wave ending time  $t_{off}$ . Further discussed in Chapter VI, the optimized boundary control strategy is indeed a square wave pulse. In addition, for easier demonstration, an upper limit is set for the boundary port flow  $Q_p \leq 25\text{cm}^3/\text{s}$ . It is convenient to define an extreme control action by using the upper limit. Further more, the upper limit can be used to mimic the boundary control pump

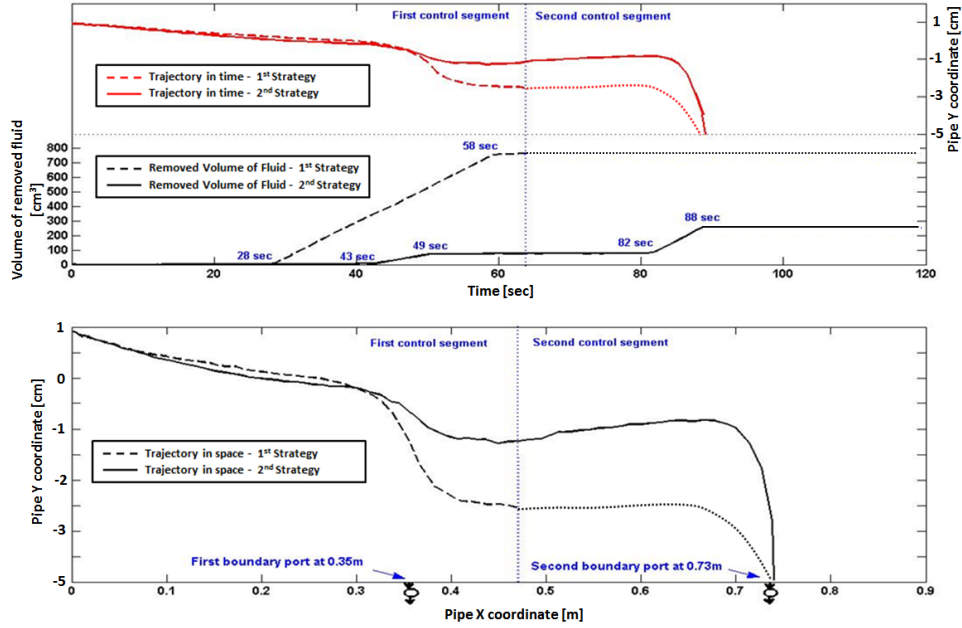


Figure 2.6: The motivation of using an optimal controller demonstrated by the prototype experiments.

physical limitations that may exist in an actual control application.

In the prototype experiments shown in Fig. 2.6, the contaminant particle was released on the pipe symmetric plane at  $y_{ini} = 1\text{cm}$ . In the first control strategy, the boundary port flow removed about  $800\text{cm}^3$  fluid in the first segment while the second strategy only removed  $100\text{cm}^3$  fluid. It is easily observed that, although the first control strategy achieved better result at the end of the first control segment that the contaminant particle moves closer to the bottom of the pipe, the second strategy better distributed the control effort and removed less fluid overall in order to eliminate contaminant particle. It should be noted that, the prototype only has one pipe segment and has only two contaminant injection spots on the symmetric plane. To generate the continuous particle trajectory for two pipe segments, the contaminant was first released by the top injection spot at  $y_{ini} = 1\text{cm}$ . The boundary control was then carefully tuned so that the final location of the contaminant is equal to the other injection spot, which is located at  $y = -1\text{cm}$ . Due to the same reason that

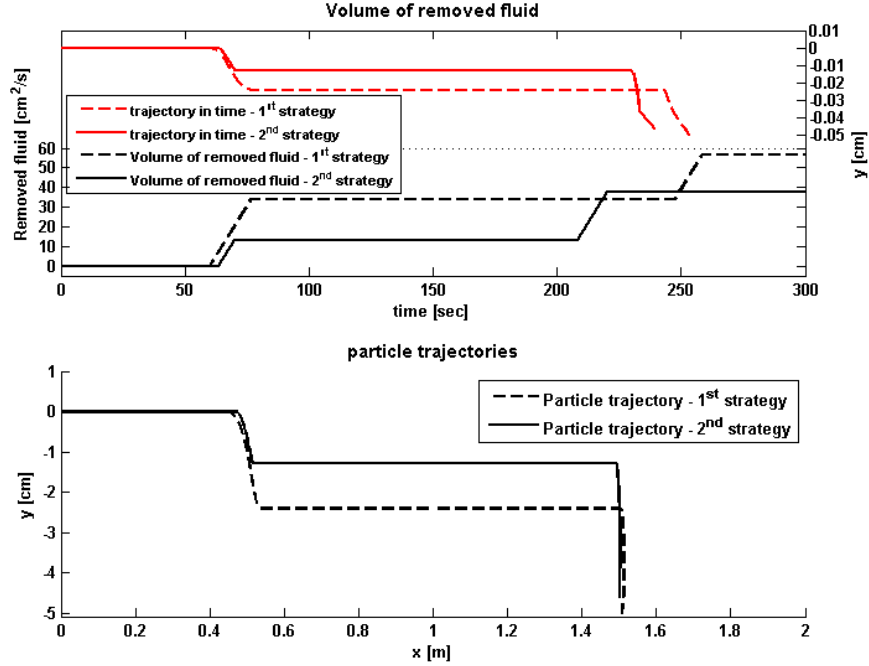


Figure 2.7: The motivation of using an optimal controller demonstrated by finite element analysis.

the number of color injection location is limited, the first control strategy was not possible to implement on the prototype in the second pipe segment. However, the demonstrated experimental results already show that, the removed amount of fluid of the first strategy in the first pipe segment already exceeds the overall removed fluid of the second strategy. It is clearly showed that the first control strategy consumed more fluid than the second control strategy and both control strategies can achieve the final goal of contaminant removal. The demonstration shown in Fig. 2.6 illustrates the benefit of a well designed boundary control strategy (2<sup>nd</sup> strategy) that conserves more than 80% of uncontaminated water.

The advantage of applying an optimized control strategy was also illustrated via 2D finite element analysis results shown in Fig. 2.7. In this example, the contaminant particle is released at the middle of the 2D pipe  $y_{ini} = 0\text{cm}$ . The same conclusion is arrived, namely, although the first strategy moved the contaminant particle closer

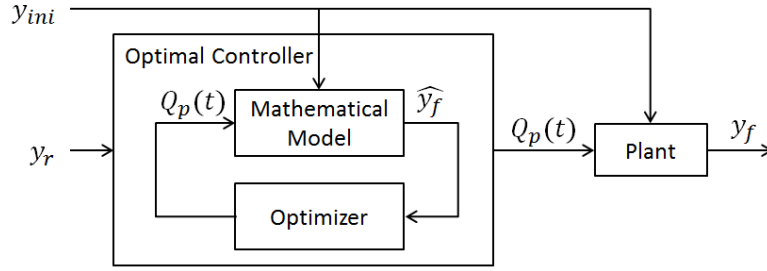


Figure 2.8: The model reference optimal controller. The controller iterates a mathematical model to optimize the boundary control strategy  $Q_p(t)$ . The mathematical model approximates the relationship among the system initial condition  $y_{ini}$ , the system flow conditions  $Q_{in}$ ,  $Q_p(t)$  and the system output  $y_f$ .

to the bottom of the pipe after the first pipe segment, the second control strategy overall removed less fluid.

In summary, the need for optimizing the boundary control strategy is now made more clear. Specifically, by strategically distributing and timing the boundary control, the same control objective can be achieved by consuming less energy and more uncontaminated fluid resource can be preserved. In this study, the optimization is formulated for the single measurement-control segment subproblem. For this subproblem, the control objective is generalized from contaminant removal to contaminant positioning. In other words, as described in Fig. 2.8, the optimal controller iterates a mathematical model to optimize the control strategy  $Q_p(t)$ , which produces the desired system output  $\hat{y}_f = y_r$  at the downstream sensor location. It will be later shown by the mathematical modelling in Chapter V that, contaminant removal can be mathematically described as  $y_r \leq -h$ , which represents the bottom of the pipe.

The optimality of a control strategy is evaluated by the amount of fluid exchange through the boundary port, which is a representation of boundary control effort. In the contaminant elimination problem, a better control strategy removes less fluid. For example, in an application such as contaminant removal from a water supply system, an optimized control strategy reduces the waste of clear water. Therefore,

the optimization problem for one measurement-control segment is mathematically formulated as the minimization problem in Equation (2.2)

$$\begin{aligned}
& \min_{Q_p(t)} \int |Q_p(t)| dt \\
& \text{s.t. } \hat{y}_f = M(y_{ini}, Q_{in}, Q_p(t)) \\
& \hat{y}_f \leq y_r \tag{2.2} \\
& Q_p(t) \leq Q_{max} \forall t \geq 0 \\
& Q_p(t) \geq Q_{min} \forall t \geq 0.
\end{aligned}$$

The variable to be optimized is the time dependent boundary control strategy  $Q_p(t)$ . The cost function is the accumulated fluid exchange through the boundary port. The equality constraint is the mathematical model, which predicts the particle final location  $\hat{y}_f$  given the initial particle location  $y_{ini}$ , the flow condition  $Q_{in}$  and the boundary control  $Q_p(t)$ . The first inequality constraint defines the control objective,  $y_r$ . The last two inequality constraints are the physical limitations of the pump, which provides the boundary port flow rate. This formulation is written for drawing and controlling the particle towards the boundary port, which is installed at the bottom of the pipe  $y = -h$ . For the case where particle is to be pushed away, the second constraint should be replaced as  $\hat{y}_f \geq y_r$ . The reason for formulating the control objective  $y_r$  in an inequality constraint is that  $y_f$  has monotonic relation with  $\int |Q_p(t)| dt$ , which is an obvious fact that the greater the boundary control effort the closer the particle will be to the boundary port. This monotonicity ensures that the optimal solution lies on the inequality constraint edge: either  $\hat{y}_f = y_r$  or  $Q_p(t) \equiv Q_{max}$ . This monotonicity will be later illustrated in Section. 6.1.1.

The optimization formulation in Equation (2.2) can be extended for multiple measurement-control segments scenario by:

1. integrating all boundary port flows as the cost function;

2. repeating the mathematical model for each measurement-control segment, where the particle final location from the upstream measurement-control segment is the particle initial location for the next measurement-control segment.

3. defining the pump strength limitation for each boundary port.

However, for the multiple boundary ports problem, the variables to be optimized include not only the  $Q_p(t)$ s for all boundary ports, but also the  $y_r$ s for each measurement-control pipe segments. In addition, the optimized boundary control strategies may be subject to change based on new measurements and the optimization iterations may keep rolling forward in time. This optimization problem of higher complexity does not change the fundamental methodology presented in this dissertation and is not further investigated.

Presented in Chapter V, two novel mathematical models were developed in this study and were employed by the optimal controller. The real-time implementation of the optimal controller is successfully demonstrated by both Finite Element Analysis method and prototype experimental results, which is presented in Chapter VI.

### **2.2.2 The Output Feedback Controller**

The output feedback controller reacts to the measured error between the actual measurement  $y_f$  and the desired objective value  $y_r$ , independently of the physical reason behind. The error can be caused by several factors. For example, the modelling error between the true system dynamics and the mathematical model cascades to errors in the optimal control strategy computed by the optimal controller, which uses the mathematical model as the reference. As a result, the actual system output  $y_f$  will be different from the optimal controller set point  $y_r$ . Another error observed on our prototype system was caused by the density variation of the color cloud, which produces a relatively consistent systematic error that the color cloud sinks downward by itself. This density variation is un-modelled in the mathematical models developed



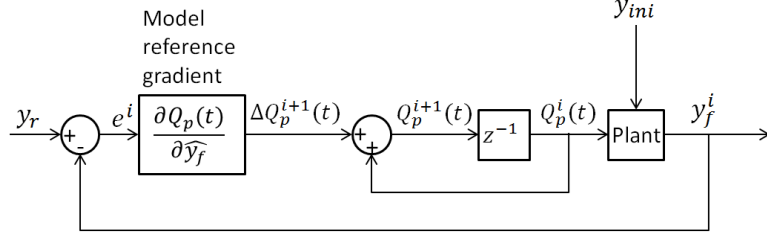


Figure 2.9: The output feedback controller.

during this study. Usually, it is difficult and is unnecessary to produce a control oriented mathematical model that captures all the physics. Based on the combined measured error, the output feedback controller is introduced to adjust the boundary control  $Q_p(t)$  and to drive the actual system output to the desired value. The output feedback control logic is illustrated in Fig. 2.9.

The overall feedback system is described as a discrete system, because the particle is only measured at discrete sensor locations. The zero-dimensional system output  $y_f$  is the accumulated effect produced by the time dependent boundary control  $Q_p(t)$ . The  $Q_p^1(t)$  in the first iteration is initialized in the  $z^{-1}$  block, which is the unit delay for a discrete system. At the  $i^{th}$  iteration, after the boundary control  $Q_p^i(t)$  is applied on the system, the error  $e^i$  compares the desired value  $y_r$  and the actual system output  $y_f^i$ . The boundary control applied in the next iteration is adjusted by the amount of  $\Delta Q_p^{i+1}(t) = e^i \times \partial Q_p(t) / \partial \hat{y}_f$ . For a linear system, the feedback logic presented in Fig. 2.9 is a deadbeat controller, which drives the system output to the target value in one discrete step. For a nonlinear system, such controller delivers the desired output in finite steps. The actual implementation of this feedback control logic will be presented in Chapter VI.

### 2.2.3 Mathematical Model Parameter Adaptation Scheme

Both the optimal controller and the feedback controller use the mathematical model as the reference. This section presents the mathematical model parameter

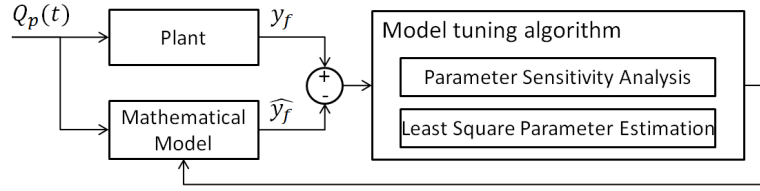


Figure 2.10: The mathematical model on-line adaptation process system block diagram.

tuning scheme (Fig. 2.10), which is aimed at adjusting the mathematical model parameters based on actual measurements to reduce the modelling error. A mathematical model of higher fidelity is beneficial to improve model reference controller design. For example, the optimized control strategy, that is closer to the true system optimum, can be found by the optimal controller if a more accurate mathematical model is employed.

The model tuning algorithm utilized in this study contains two processes. The first process is the sensitivity analysis of the mathematical model parameters *Khalil* (1995). The sensitivity of a model parameter indicates the influence of that parameter upon the system output. The sensitivity analysis helps to determine which parameters can be effectively tuned based on the measurement, which occurs at a specific location and time. In the second process, the selected parameters to be tuned are applied in a Least Square Parameter Tuning algorithm.

The sensitivity analysis used in this study requires the mathematical model be written as a set of Ordinary Differential Equation (ODE)s. Suppose the mathematical model can be generalized as Equation (2.3)

$$\dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}, \boldsymbol{\lambda}_0), \quad (2.3)$$

where  $\mathbf{x}$  is the mathematical model ODE states,  $\mathbf{f}$  is the mathematical model state equations, the  $\boldsymbol{\lambda}$  is the vector containing all the mathematical model parameters to

be analyzed, and the  $\lambda_0$  is the nominal value of  $\lambda$ . The sensitivity of the model parameters,  $\mathbf{S}$ , is defined as Equation (2.4)

$$\mathbf{S} = \frac{\partial \mathbf{x}(t, \lambda)}{\partial \lambda}, \quad (2.4)$$

which defines the sensitivity of the parameters with respect to the mathematical model ODE states. To solve for the sensitivity  $\mathbf{S}$ , the sensitivity function defined in Equation (2.5) is used,

$$\dot{\mathbf{S}}(t) = \mathbf{A}(t, \lambda_0)\mathbf{S}(t) + \mathbf{B}(t, \lambda_0), \quad \mathbf{S}(t_0) = \mathbf{0}, \quad (2.5)$$

where  $\mathbf{A}$  and  $\mathbf{B}$  are defined in Equation (2.2.3)

$$\begin{aligned} \mathbf{A}(t, \lambda_0) &= \left. \frac{\partial \mathbf{f}(t, \mathbf{x}, \lambda)}{\partial \mathbf{x}} \right|_{\mathbf{x}=\mathbf{x}(t, \lambda_0), \lambda=\lambda_0} \\ \mathbf{B}(t, \lambda_0) &= \left. \frac{\partial \mathbf{f}(t, \mathbf{x}, \lambda)}{\partial \lambda} \right|_{\mathbf{x}=\mathbf{x}(t, \lambda_0), \lambda=\lambda_0} \end{aligned}.$$

The sensitivity function is also a set of ODEs, which are solved simultaneously together with the mathematical model. After the sensitivity function is solved, the parameter sensitivities with respect to the system output,  $\partial y / \partial \lambda$ , are evaluated using Equation (2.6) based on where and when the output measurement takes place,

$$\frac{\partial y}{\partial \lambda} = C \left. \frac{\partial \mathbf{x}}{\partial \lambda} \right|_{\mathbf{x}=\mathbf{x}_f, t=t_f}, \quad (2.6)$$

where the  $C$  is the output matrix relating the system output and the ODE states. The  $\mathbf{x}_f$  and  $t_f$  represent where and when the  $y_f$  is measured. The sensitivity analysis will be applied on the Velocity Field Particle Tracking (VPT) model, which is later illustrated in Section 5.2.

After the parameter sensitivity analysis, the model parameters that can be ef-

fectively tuned are determined. These parameters are then identified using a least square parameter estimation algorithm. The detailed implementation of the sensitivity analysis and the parameter tuning process will be presented in Section 6.3.

#### **2.2.4 Summary**

In this section, the overall control architecture is defined together with the methodology to be used in each controller. The control architecture consists of an optimal controller, a feedback controller and a mathematical model parameter tuning scheme. The optimal controller, which is a feedforward controller, optimizes the boundary control sequence to achieve the control objective set point. The optimality of the boundary control is represented as the total volume of fluid that is removed by the boundary port flow. The control objective is the contaminant location in the downstream after the boundary control implementation. An optimal control strategy is demonstrated via finite element analysis and prototype experiments, where the optimized boundary control strategy achieves the particle elimination by consuming less boundary control effort and preserves more uncontaminated fluid. The feedback controller is introduced to overcome system error and to drive the actual measurement towards the control objective. The system error can be caused by a variety of factors, for example, boundary pump drift, contaminant physical property variations, and other unmodeled factors. Both the optimal controller and the feedback controller employ a reference mathematical model to compute the associated control parameters. A mathematical model of higher fidelity is beneficial to improve the controller performance. Hence, the model parameter tuning scheme is introduced in this study, aiming to demonstrate the possibility of tuning the proposed mathematical model in this dissertation.

## 2.3 Conclusion

In this chapter, a realistic starting point to solve the general contaminant elimination flow control problem is established. The original flow control problem involves controlling multiple boundary ports simultaneously in the 3-dimensional pipe. A problem simplification was introduced to represent the 3-dimensional control problem using 2-dimensional space which relies on the geometry symmetric plane. The orthogonal projection of the contaminant particle trajectory onto the symmetric plane is sufficient to define the contaminant removal. Based on the geometry symmetric plane, the flow control problem parameters and notations are defined.

The overall control architecture is presented in this chapter. The control architecture consists of:

1. a feedforward optimal controller that uses a reference mathematical model to optimize the boundary control strategy.
2. an output feedback controller that reacts to the measured error between the desired and the actual particle final location.
3. a model adaptation process that tunes the mathematical model towards higher fidelity.

## CHAPTER III

### The Prototype Experimental Setup

A table size prototype pipe flow system was designed and fabricated in this study. The flow system represents one single measurement-control segment with one boundary port. The system diagram, geometry parameters and the nominal flow conditions are presented in Section 3.1. The dynamic of the boundary port pump is discussed in Section 3.2. The prototype signal processing methods are presented in Section 3.3. Finally, one worst case scenario experiment and the prototype experiment repeatability are presented in Section 3.4.

#### 3.1 System Diagram

The front view of the prototype is shown in Fig. 3.1. To simulate the passenger terminal hazardous chemical cloud removal scenario (Chapter I, Fig. 1.2), the chemical cloud is represented by a blue color cloud. The ambient dominant flow uses clear tap water, which enters the transparent tube from the right shown in the figure. After a filter, the tap water flow encounters the ink injection unit, by which the ink can be injected at 6 different spots individually. The amount of the injected ink is controlled by the duration of the injection using a solenoid valve. The ink cloud is immediately subjected to the ambient flow and enters a high density sponge, which removes any flow turbulence that is caused by the injecting action or the prior geometries. The

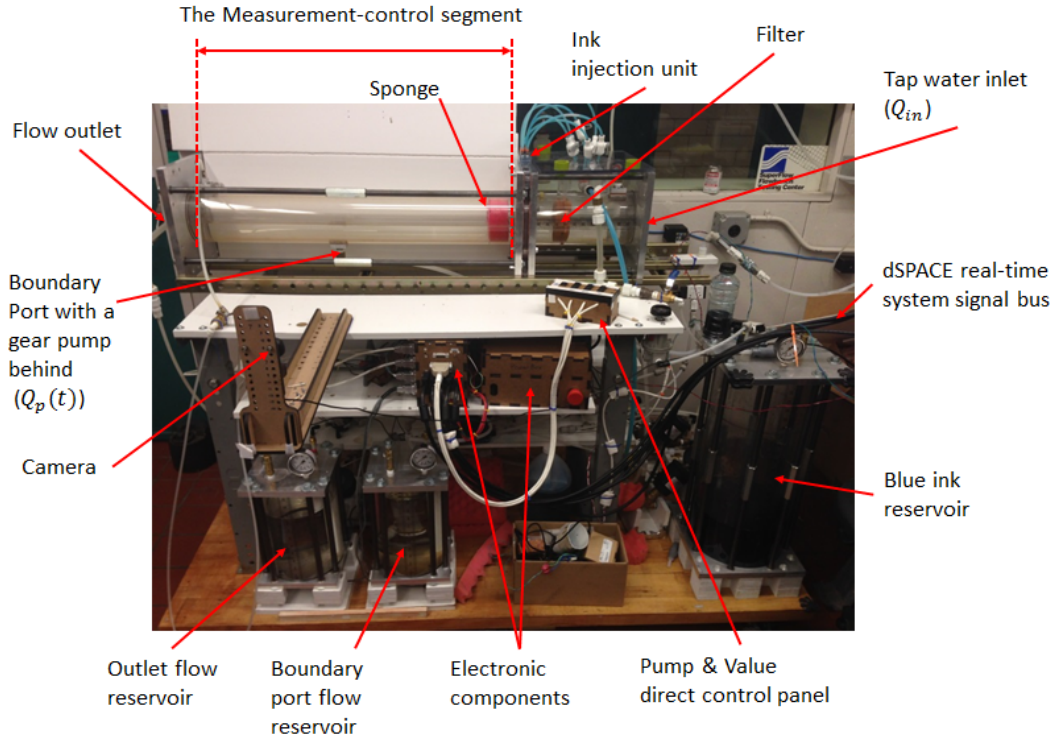


Figure 3.1: The front view of the prototype. The dSPACE real-time system hardware is to the right of the photo and is not included in this image.

flow right after the sponge is assumed a uniform velocity profile. The ink color cloud is then observed by a live camera. The real-time system processes the camera image for the cloud location, computes the desired boundary control strategy  $Q_p(t)$  and applies the boundary control to a gear pump. The image processing method is later illustrated in Section 3.3.

The major parameters for the prototype system are listed in Table. 3.1, which also includes some notations to be defined or used in later context.

The physical geometries of the ink injection unit is shown in Fig. 3.2. The ink injection unit is made of seven layers of acrylic board, which form a 3D geometry connecting the ink injection spot to the outside pipes. Layers are glued together by siphoning thin glue liquid from the gluing holes. Extending the injection spot, small pipes are inserted. After the ink injection unit is installed, these small pipes are

Table 3.1: Flow System Parameters and Notations

Parameter	Symbol	Unit	Value or Range
Inlet flow rate	$Q_{in}$	cm <sup>3</sup> /s	0 ~ 120
Boundary port flow rate	$Q_p$	cm <sup>3</sup> /s	0 ~ 120
Volume of removed fluid	$V = \int Q_p dt$	cm <sup>3</sup>	-
3D Pipe radius	$h$	cm	5.08
Boundary port location	$x_p$	cm	depends on camera setup
Boundary port diameter	$w$	cm	1.27
Pipe total length	$L$	cm	91

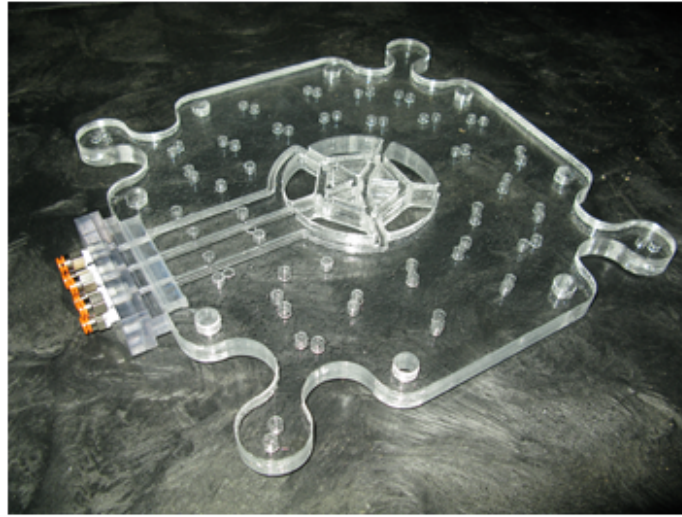


Figure 3.2: The prototype ink injection unit. The ink cloud represents the contaminant solute.

pushed into the sponge to reduce the turbulence that is caused by the injection flow. In this study, we were focused on the flow dynamics on the symmetric plane of the pipe and the boundary port. Therefore, only two injection spots, which are located on the symmetric plane, are needed. One is centered in the middle of the pipe, the other one is in the top area of the pipe. The prototype experiment results presented in this dissertation are mainly conducted using the center ink releasing location.

The boundary port installation design aims to minimize the influence of the installation geometry on the flow pattern as illustrated in Fig. 3.3. The measurement-control segment is constructed using two transparent acrylic tubes, naming the outer



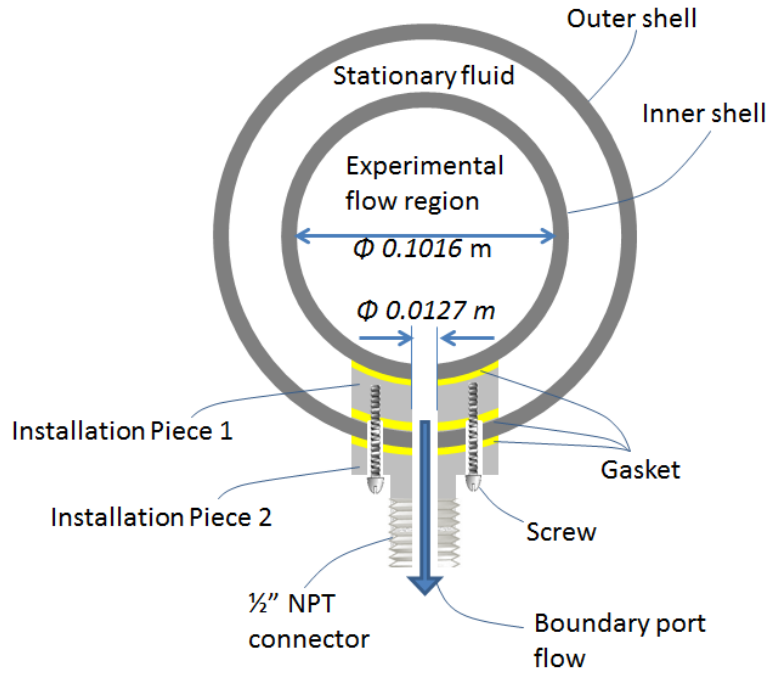


Figure 3.3: The prototype boundary port installation.

shell and the inner shell. The ink is injected into the experimental flow region, which is enclosed by the inner shell. The stationary fluid area between the inner shell and the outer shell is filled with clear water and is sealed before any prototype experiment setup. The two pieces of boundary port installation are screwed together, clamping the outer shell with sealing gaskets. The gasket between the installation piece 1 and the inner shell, however, does not experience pressure. Because the stationary fluid area is sealed, leaking through this gasket can be neglected. This boundary port installation design ensures that the experimental flow region is an ideal cylinder, which is intersected with another ideal cylinder shaped boundary port flow.

The prototype system diagram is shown in Fig. 3.4, with the numbered item listed in Table 3.2

An prototype experiment is prepared and conducted in the following procedure:

- 1: Open valve (2) and power on system electronics.
- 2: Check ink reservoir (18) condition. Add ink by opening valve (15) and/or add

Table 3.2: Flow System Diagram

<b>Item No.</b>	<b>Description</b>
1	45 psi tap water supply
2	main water supply shutoff valve
3	water filter
4	60 psi pressure relief valve
5	air piloted hydraulic directional control valves
6-1	single jet flow meter
6-2,3	ultrasonic flow meter
7	flow restrictor
8	pipe outer shell
9	pipe inner shell
10	ink injection unit
11	check valve
12	electric motor powered hydraulic gear pump
13	shutoff valve for selecting injection spot
14	undiluted blue ink tank
15	shutoff valve for adding ink
16	100 psi compressed air source
17	air pressure regulator
18	ink diluting tank
19	boundary port flow reservoir
20	outlet flow reservoir
21	shutoff valve for filling the stationary fluid area between 8 and 9
22	water drain reservoir

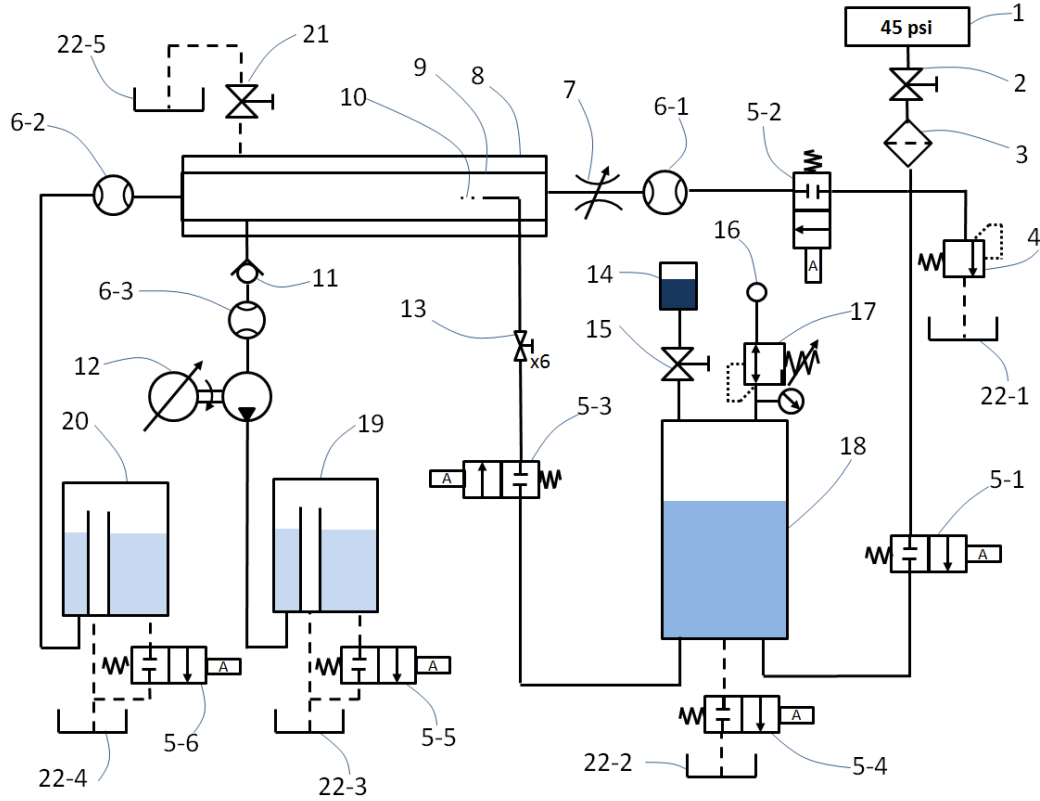


Figure 3.4: The prototype flow system diagram.

clear water through (5-1) to dilute the ink.

3: Open valve (5-2) to allow dominant flow into the testing pipe.

4: Open valve (21) to allow air leaving the pipe area. Close valve (21) when the entire pipe area is filled with clear water.

5: Adjust restrictor (7) for proper inlet flow rate  $Q_{in}$ .

6: Adjust air regulator (17) to pressurize ink reservoir (18) for ink injection.

7: Start running the dSPACE real-time computing software system.

8: Perform experiments: Inject ink using valve (5-3) and implement boundary control using the Pump (12). Both actions are controlled through real-time software

9: Close valve (2).

10: Open valve (21) and drain remaining water from the main pipe by running pump (12).

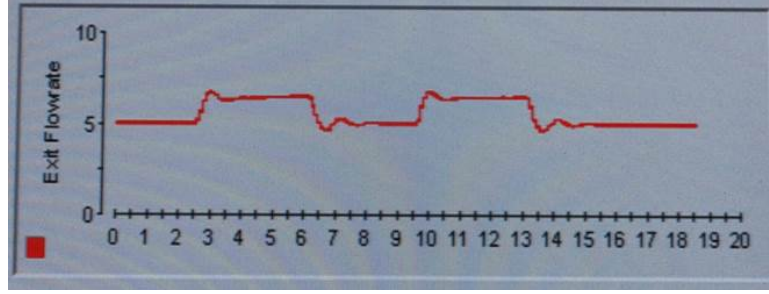


Figure 3.5: The prototype hydraulic pump step response. The pump provides the boundary control flow rate through the boundary port.

11: Close valve (21).

12: Power off system electronics.

The Reynolds number of the pipe flow around nominal flow conditions can be computed in Equation (3.1)

$$\begin{aligned}
 Re &= \frac{\rho Q_{in} D}{\mu A} \\
 &= \frac{\rho Q_{in} 2h}{\mu \pi h^2} \\
 &= 1044.3,
 \end{aligned} \tag{3.1}$$

where  $Q_{in} = 83.33 \text{ cm}^3/\text{s}$  (5 LPM), water density and dynamic viscosity near room temperature are  $\rho = 1000 \text{ kg/m}^3$  and  $\mu = 1e - 3 \text{ Pa}\cdot\text{s}$  respectively. The Reynolds number is within the laminar flow range, which is the suitable condition for this study.

## 3.2 Boundary Port Control

The step response of the boundary port hydraulic pump is illustrated using the screen snapshot in Fig. 3.5. The photo shows the displayed downstream flow rate (exit flow) in real-time during a prototype experiment. In this example, the boundary pump injects water with about 4 secs duration as the downstream. The rise time of the boundary pump is about 1 sec.

### 3.3 Signal Processing and Prototype Repeatability

Two technical difficulties are avoided by using a live camera instead of the discretely installed sensor arrays. First, the camera is installed outside the pipes and does not disturb the flow pattern. Second, any substance with color can be used to represent the contaminant, which simplifies the material selection and contamination post treatment. In this study, environmental friendly food color die was used. There are two operation modes of the prototype. The first considers steady state flow when the boundary port flow rate is constant in time. In this scenario, the contaminant is continuously released from one single spot. Flowing along with the ambient fluid, the contaminant forms a steady continuous line, which approximates the particle trajectory. The second mode refers to unsteady state flow scenario, where the boundary control can be dynamic. The operation modes and their image processing methods are presented in this section.

#### 3.3.1 Mode One - Steady Flow Experiment

The ink (contaminant) is slowly released from a contaminant injection spot continuously in this operation mode. The image processing steps are shown in Fig. 3.6. The camera captures one single image. The image is windowed and zoomed at the pipe segment that is of interest, naming after the injection location to the flow outlet (refer to Fig. 3.1). The windowed pipe image is then flipped for convention, showing dominant flow from left to right. (Note that, the prototype was modified at certain point during the study. The injection spot is covered within the high density sponge after the modification, unlike the photo shown in Fig. 3.6-a.) After a line by line scan of the processed image, the single 1D trajectory of the contaminant is generated. For steady flow, this curve represents the trajectory of a particle that is released at the start point of this curve. This prototype experiment mode is used to validate the first mathematical model in Chapter V, which model is called the Algebraic Particle

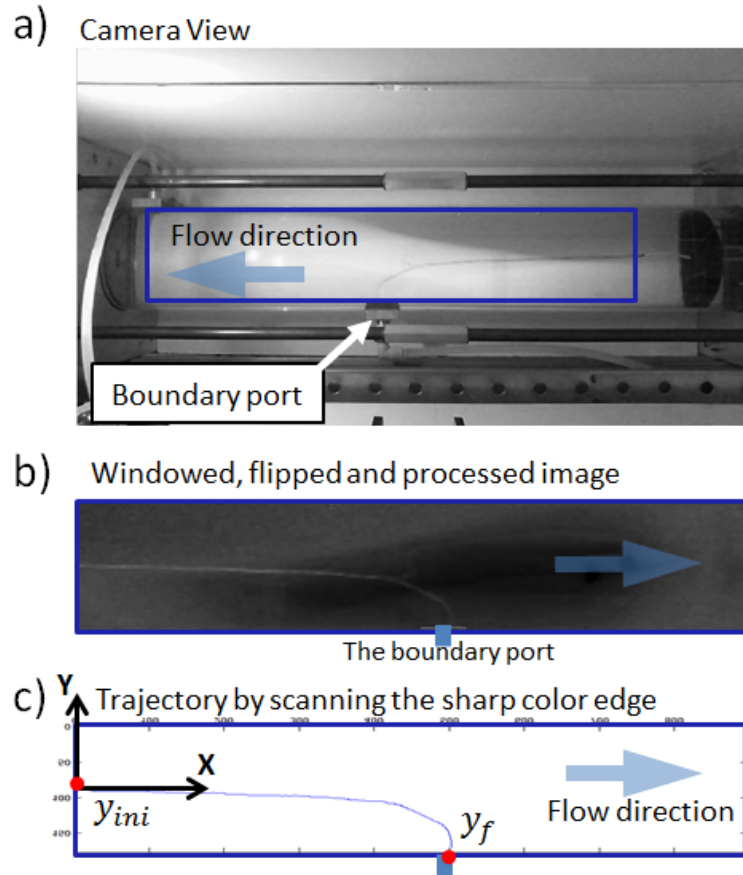


Figure 3.6: The prototype image processing steps for steady flow scenario. The boundary control flow rate is constant in time.

Tracing (APT) model developed for steady flow.

### 3.3.2 Mode Two - Transient Flow Experiment

In this mode, the ink (contaminant) is injected within a short time duration, forming a color cloud. To simulate discretely installed sensor arrays, although the entire pipe section is monitored by the camera, only the contaminant cloud locations captured at two specific virtual sensor spots are used for control purposes. The entire image processing steps are summarized in four steps in Fig. 3.7.

The image is sampled and processed at 1 frame/sec, which is sufficient to approximate the entire cloud trajectory. Several actual processed images are visualized in

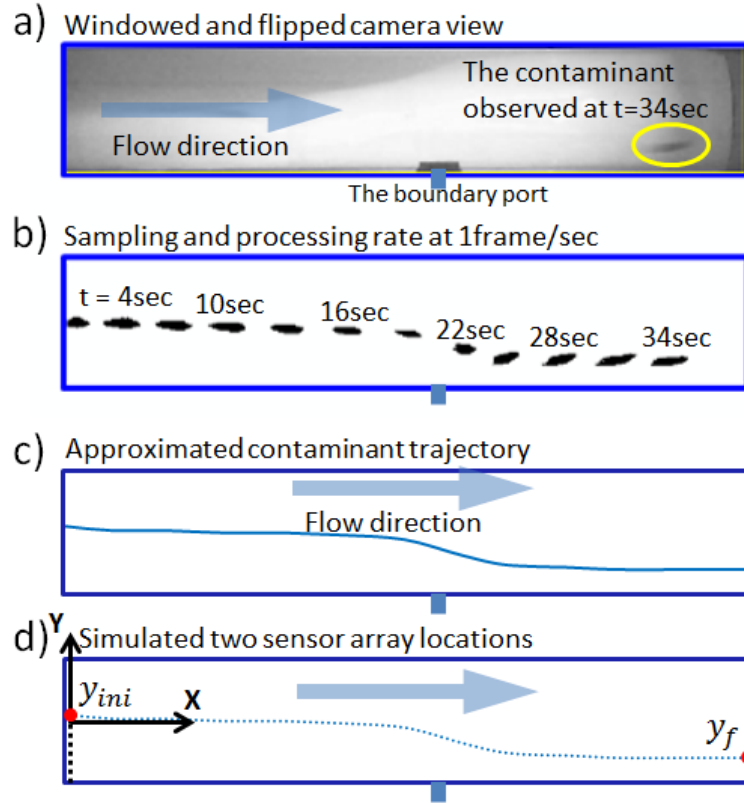


Figure 3.7: The prototype image processing steps for transient flow scenario. The boundary control flow rate is time varying.

Fig. 3.1 b), showing the evolution of the shape and the location of one contaminant cloud. In the case shown, the boundary port continuously drew fluid and the contaminant moves towards the boundary port. In this study, the contaminant cloud is simplified as one single particle, which is defined as the contaminant cloud visual center. The full trajectory of a contaminant particle is approximated by connecting the visual center at each time frame in Fig. 3.1 c). Finally, for real-time control purposes, only the contaminant initial location  $y_{ini}$  in the upstream and the final location  $y_f$  in the downstream are used by the controllers previously defined in Chapter II. The detailed matlab image processing program is documented in Appendix A. This operation mode of the prototype is closer to the hazardous chemical cloud control problem defined in Chapter I.

In both prototype operation modes, it was observed that, the ink cloud moved downward in the upstream region when no boundary port flow was applied. This phenomenon can be caused by several factors. The first factor is the color cloud density. By changing the dilution ratio between the color die and water, the ink cloud moving downward rate changes accordingly. The second factor is the water temperature. When the tap water supply temperature changes, the phenomenon also varies. This temperature variation can be potentially resolved if a large water reservoir is available to store and control the water source temperature. However, installing such water reservoir requires renovation of the entire laboratory. Therefore, the investigation of the temperature effect could be introduced in future studies. The third factor is the flow pattern produced by some unknown geometry variations. For example, a big air bubble in the high density sponge may lead to turbulence and change the flow pattern. In summary, this downward motion of the ink cloud was treated as a systematic error. For all the experiments shown in this dissertation, prototype repeatability was verified before and after each experiment group, so that the experimental results to be compared have the same systematic error.

### **3.4 Worst Case Test and Repeatability**

#### **3.4.1 Worst Case Scenario Experiment**

An open loop worst case scenario experiment was performed to test the strength of the boundary port pump. The ink was released from all injection spots and the ink spread throughout the entire cross section of the pipe. To remove all the ink cloud, the obvious control strategy is to draw all fluid through the boundary port, that is,  $Q_p = Q_{in}$ . As shown by the experiment snapshots in Fig 3.8, the ink was injected and dispersed during the time period  $t=0 \sim 8$  sec. After the detection of the contaminant cloud, the boundary port started drawing fluid at  $t_{on} = 10$  sec at



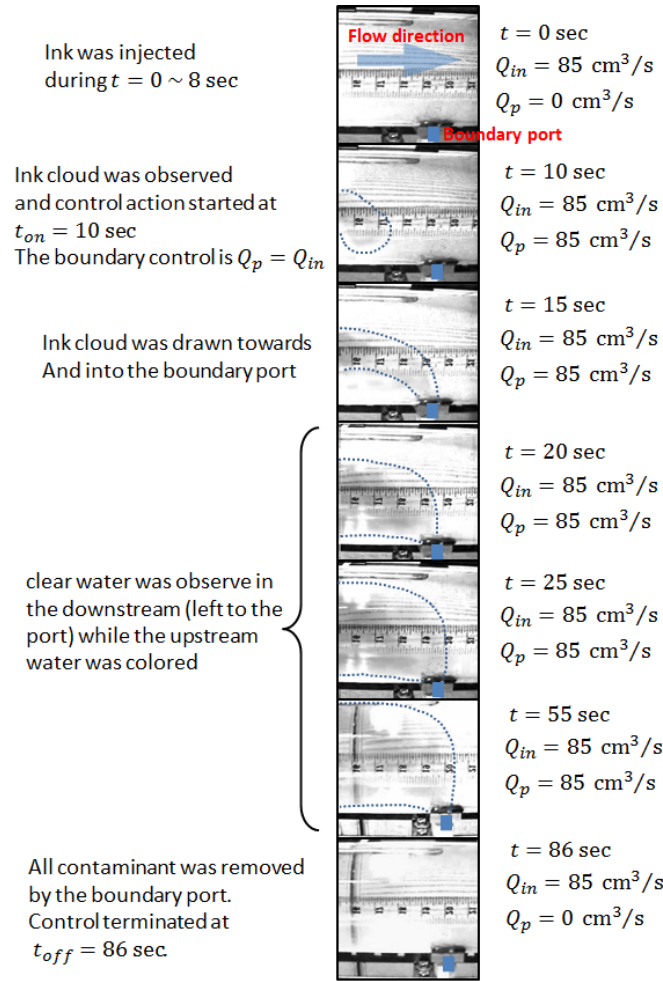


Figure 3.8: The prototype worst case scenario experiment - contaminant removal. The ink was injected through all injection spot and was spread out throughout the entire pipe cross section. The obvious boundary control is to remove all the fluid by applying  $Q_p = Q_{in}$ . The boundary control was turned on at  $t_{on} = 10$ sec and was turned off at  $t_{off} = 86$ sec

the speed of  $Q_p = Q_{in}$ . The control action was terminated at  $t_{off} = 86$  sec when the concentration of the contaminant was too low to be observed by the camera. The envelopes of the contaminant were approximately marked by the dot lines. During the control period, no colored water flow to the downstream region of the boundary port, indicating a successful control result. By performing this extreme case test, it was concluded that the physical strength of the water pump power was sufficient to provide enough boundary control for removing all bulk fluid.

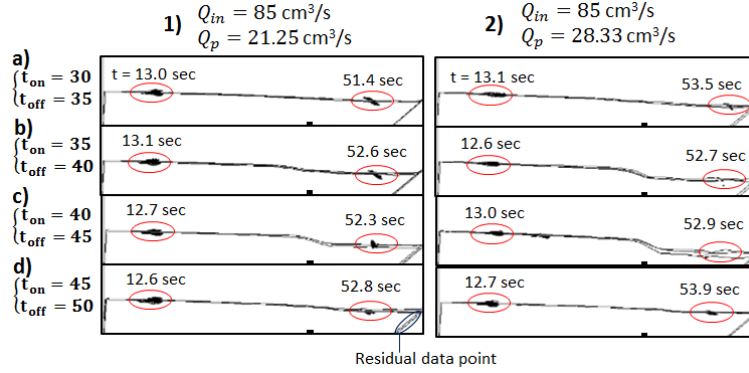


Figure 3.9: The prototype repeatability test. Each of the eight scenarios were repeated three times and the processed three trajectories are plotted in the save window. The repeatability drops if the the ink color cloud was partially removed by the boundary drawing action as in the 2)-c) scenario.

### 3.4.2 Experiment Repeatability

Repeatability tests are presented in this section to show the reliability of the prototype system. In the eight control scenarios shown in Fig. 3.9, the boundary control was defined as a square wave function, whose wave width is defined by the boundary control timing turn on and turn off time:  $t_{on}$  and  $t_{off}$ . The ink trajectories for two different boundary port flow rate magnitudes and four different control timing are graphed. The overall repeatability is high except for the special case when the ink color cloud was partially removed by the boundary port drawing action: scenario 2-c). The ink cloud partial removal case is further illustrated in Fig. 3.10.

When the ink cloud is partially removed or the cloud moves very close to the bottom of the pipe, the color cloud was stretched and the approximation of the single particle failed. As illustrated in Fig. 3.10 case a), at time 43.9 sec, the ink cloud started splitting into two directions. The lower direction led the colored water to the boundary port and the other direction passed the remaining colored water to the downstream. In addition, when the ink cloud was stretched, the image color density also decreased. Therefore, the image processing algorithm cannot predict the location

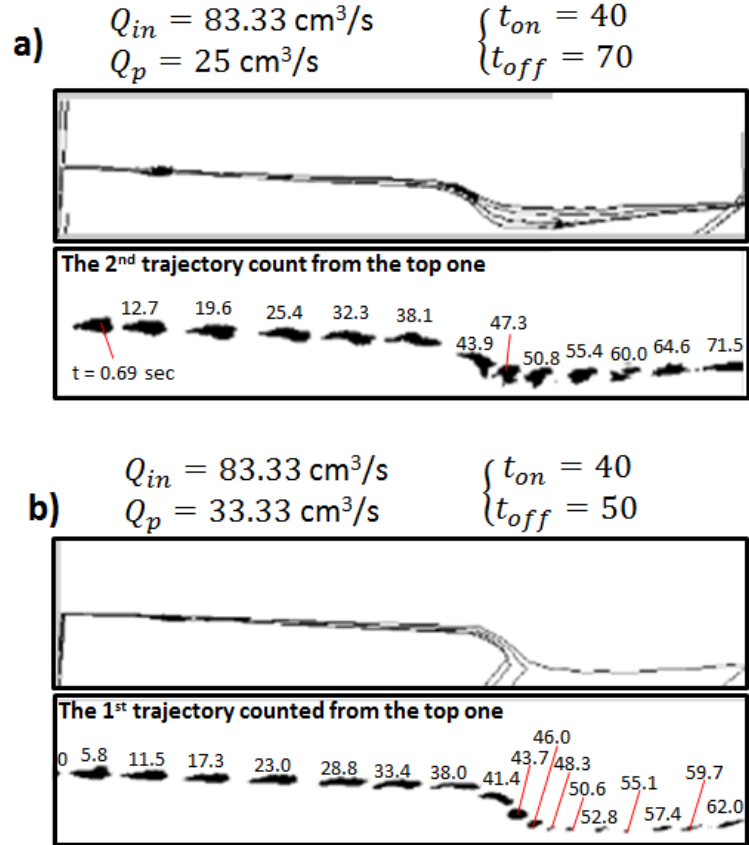


Figure 3.10: The prototype repeatability when the ink cloud is partially removed. Due to the resolution of the camera and the image processing algorithm, it is difficult to produce repeatable results of the deformed or dimmed color cloud.

of the ink cloud in a repeatable manner.

### 3.5 Prototype Limitations and Summary

The overall performance of the prototype system is presented in this chapter. There are several drawbacks and limitations of this prototype system. First, the ink injection spots are limited and there are only two injection spots located on the symmetric plane of the pipe. Second, due to several environmental factors, there is one unexpected but repeatable flow pattern: the ink cloud moves downward automatically without applying boundary flow control action. This flow pattern is treated as a

systematic error in the later context. Third, the experiment repeatability drops when the ink cloud is partially removed by the boundary port. Despite of these drawbacks, the prototype is suitable for the problem of particle trajectory control problem that is the main focus of this dissertation.

## CHAPTER IV

### The FEA Model and CFD Simulation Setup

Due to the relatively low resolution of the camera and the limitations of the prototype, CFD simulations were conducted as virtual experiments for mathematical model development. The mathematical models presented in Chapter V were first developed for 2D pipe geometry and were calibrated using 2D CFD simulation results. To implement the mathematical model on the 3D pipe prototype, the model parameters are then tuned using 3D CFD simulation results. The Finite Element Analysis (FEA) model used in the CFD simulation software environment, where the physical parameters can be fully controlled, provides a close approximation of the true physics.

#### 4.1 The 2D and 3D Finite Element Model

The 2D and the 3D pipe setups were previously described in Chapter I Fig. 1.4. The FEA model parameters are listed in Table 4.1.

The finite element model analysis was performed in the *Ansys* software environment. The 2D geometry was meshed with 5000 ~ 10000 quadrilateral elements sized from 1 to 3 mm. The 3D geometry was meshed with 1 ~ 2 million tetrahedral elements (mixed with 5-node and 6-node type) sized from 2.5 ~ 10 mm. The meshes of the two geometries are zoomed in near the boundary port in Fig. 4.1.

Table 4.1: Flow System Parameters and Notations

Parameter	Value or Range	
	2D pipe	3D pipe
$Q_{in}$	10 cm <sup>2</sup> /s	85 cm <sup>3</sup> /s
$Q_p$	0 ~ 10 cm <sup>2</sup> /s	0 ~ 85 cm <sup>3</sup> /s
$h$	5.08 cm	5.08 cm
$x_p$	21.4 cm	51.44 cm
$w$	1.27 cm	1.27 cm

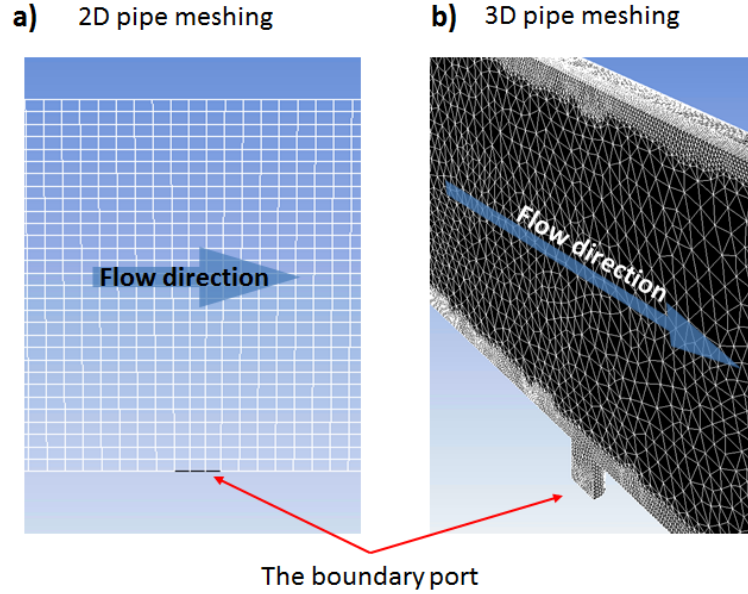


Figure 4.1: The finite element model for CFD simulations.

## 4.2 CFD Software Environment Parameters Setup

The *Ansys-Fluent* CFD solver settings are screen captured in Fig. 4.2. Double precision with serial processing were used for both 2D and 3D cases (Fig. 4.2-Top-Left). laminar viscous flow model was used in *Ansys-Fluent* (Fig. 4.2-Top-Right). In this study, pure water physical properties were used as the base reference values and the flow rate was maintained low to ensure low Reynold number laminar flow condition. For example, the Reynold number of the  $h = 2.54\text{cm}$  pipe is around 200 when the average flow velocity is 2 mm/s. When higher flow rate was considered, fluid viscosity may subject to change to produce similar Reynold number in each simula-

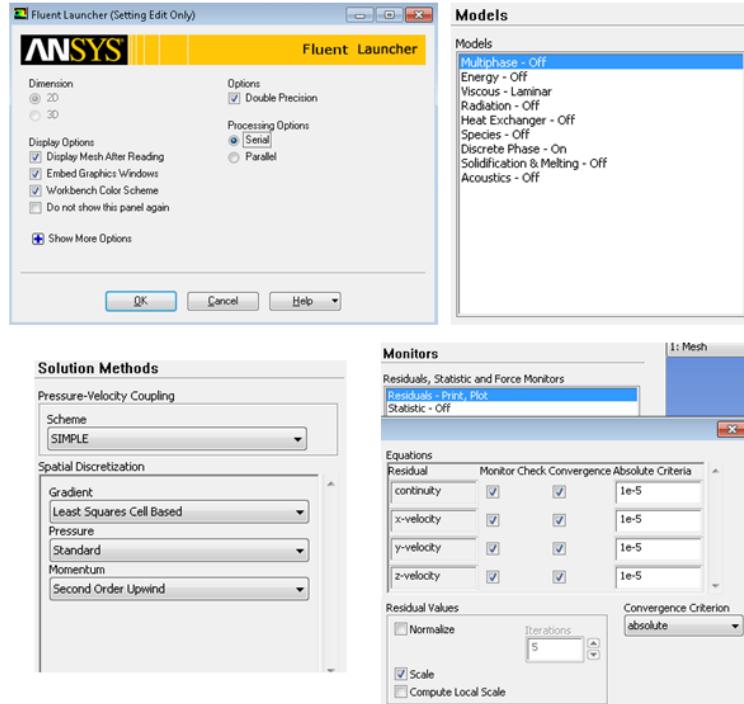


Figure 4.2: The CFD solver settings used in the *Ansys-Fluent* software environment.

tion case. The solution methods in *Ansys-Fluent* determine the Partial Differential Equation (PDE) to be solved (Fig. 4.2-Bottom-Left). Simple scheme was used with Least-squares cell based gradient, standard pressure model and second order upwind momentum settings. The convergence criteria range from  $1e-4$  to  $1e-7$  depending on the meshing size (Fig. 4.2-Bottom-Right). Identical meshing and convergence criteria are used for any CFD simulated results that are to be compared.

In summary, the finite element model provides a fluid model ideal conditions. The color cloud density variation effects are avoided in the CFD simulation. Therefore, the finite element analysis results were idea for mathematical model development, which will be presented in the next chapter.

## CHAPTER V

### Mathematical Modelling for Real-time Control

The three controllers defined in Chapter II are powered by the mathematical model which approximates the propagation of the contaminant particle given a boundary control action. The optimal controller iterates the mathematical model to optimize the boundary control strategy. The feedback controller uses the mathematical model to determine the feedback gain. The model adaptation scheme aims to increase the mathematical model fidelity by adjusting the model parameters.

As discussed in Chapter I, there are many existing methods that can be used to approximate the particle propagation within a flow field. Some of the methods involve computationally intensive algorithms, which are not suitable for the real-time control problem considered in this study. For example, the CFD algorithm (Chapter IV) consumes about 2 minutes to solve the finite element model of the 2D pipe geometry. But, this solving speed is too low to be utilized in the iterative optimal controller, which is expected to optimize the boundary control strategy within 1 minute for the prototype system. Analogy to an actual situation when toxic chemical is released into a passenger terminal, for example, suppose the chemical cloud removal solution has to be determined within 10 minutes before the chemical cloud spreads out. If 100 iterations is needed to find the control solution, the CFD algorithm needs to make cloud propagation prediction within 6 secs in each iteration. This speed requirement



is beyond the capability of the CFD algorithm based on the Author's knowledge.

As a result, we are motivated to develop a mathematical model, which approximates the contaminant particle propagation fast enough for real-time control purpose. Low Reynolds number laminar pipe flow is assumed for simplicity. Two suitable mathematical models for real-time control were developed in this study. The Algebraic Particle Tracing (APT) model presented in Section. 5.1 is used for static flow scenario when the boundary control flow rate is constant in time, specifically,  $Q_p(t) = Q_p$  for  $t \geq 0$ . This model is an algebraic relationship that describes the relation among the particle initial location  $y_{ini}$  at upstream sensor, the boundary control strength  $Q_p$ , and the particle final location  $y_f$  at the downstream sensor. Therefore, the APT model computes the  $y_f$  discretely without considering the dynamics between the two sensors. On the other hand discussed in Section. 5.2, the Velocity field Particle Tracking (VPT) model approximates the full spatial trajectory of the particle and is used for transient flow case when the boundary control strength is time varying. As is described in Fig. 5.1, given the contaminant particle initial location  $y_{ini}$  and a boundary control strategy  $Q_p(t)$ , both mathematical models are used to compute the contaminant particle final location  $y_f$  in the flow downstream region for the controller. The modelling techniques associated with the two mathematical models are also presented in this chapter.

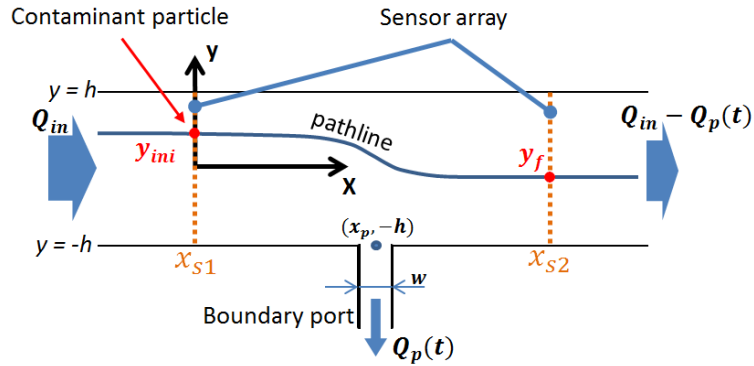


Figure 5.1: Notations for the pipe measurement-control segment.

Furthermore, although the models are used for predicting particle final location forward in time in this study given the initial condition, both models can be used in the reverse direction to back tracking the source of the particle given its final observed status. This source inversion problem is also an important area of study, which was frequently discussed in studies of groundwater, atmosphere or a body of surface water. For example, identifying the sources of groundwater pollution was studied by matching the simulated and measured solute concentration data (*Gorelick et al.* (1983)). The mathematical models presented in this thesis can be potentially applied in such source inversion problem for faster simulation of particle propagation.

## 5.1 The APT Model - Particle Location Prediction for Constant Boundary Control

For the one boundary port 2-dimensional pipe segment, the Algebraic Particle Tracing (APT) model creates a simple algebraic relationship among the particle initial location  $y_{ini}$  at the upstream sensor location, the particle final location  $y_f$  at the downstream sensor, the upstream total flow rate  $Q_{in}$  and the boundary port flow  $Q_p$ . This model is used for the steady flow scenario, where the boundary port flow rate  $Q_p(t)$  is assumed constant in time, specifically,  $Q_p(t) = Q_p$  for  $t \geq 0$ . This assumption is valid under the following conditions:

1. The upstream flow rate  $Q_{in}$  stays constant during the boundary control period.
2. The boundary port starts its constant control action when the contaminant particle is far away from the port in the upstream, where the flow is fully developed.
3. The boundary port ends its control action when the contaminant particle is far away from the port in the downstream, where the flow is fully developed.

The 3<sup>rd</sup> condition is satisfied by placing the downstream sensor away from the port at least the distance described by the laminar flow entrance length, which is

approximated by the empirical relationship in Equation 5.1 *Post* (2011)

$$L_{lam} = 0.05R_eD_h, \quad (5.1)$$

where the  $L_{lam}$  is the entrance length for laminar flow of Reynolds number  $R_e$  in a pipe of hydraulic diameter  $D_h$ . For the 2-dimensional pipe with height of  $2h$  or the 3-dimensional pipe with radius of  $h$ , the hydraulic diameter is  $D_h = 4h$  or  $D_h = 2h$  respectively. The entrance length refers to the distance until the flow velocity profile is fully developed after an obstruction. After the distance  $L_{lam}$  in the pipe downstream region where the flow is fully developed, the contaminant particle  $y$  location is not further influenced by the boundary port flow. The same distance can be used to place the upstream sensor before the boundary port. Although there is no such empirical relationship for describing flow backward into the upstream region for the 2<sup>nd</sup> condition, it can be observed via CFD simulated results (Fig. 5.3) that the boundary flow influential range into the upstream is shorter than the range into the downstream. Therefore, placing the upstream sensor at least  $L_{lam}$  from the port ensures the constant  $Q_p$  assumption in the upstream.

### 5.1.1 Modelling Technique

The modelling principle of the APT Model is based on the conservation of volume for incompressible fluids and is illustrated in Fig. 5.2. The 2D pipe is  $2h$  in height and the pipe is long enough (for instance, two times the entrance length  $2L_{lam}$ ) so that the flow is assumed fully developed in both upstream and downstream of the port. The corresponding fully developed parabolic velocity profile at the upstream and downstream region are  $v_{ini}^x(y)$  and  $v_f^x(y)$  respectively. Because the velocity in fully developed region is purely in  $x$  direction, the velocity profiles are functions of  $y$  coordination.

In this 2D pipe, following its pathline, the contaminant particle moves from its initial location  $y_{ini}$  to its final location  $y_f$  due to the boundary port drawing action. This pathline splits the flow field into the upper and the lower two areas between the two sensor arrays. Because a fluid element cannot have two different velocities at the same spatial location in steady flow, the streamlines do not intersect with each other. In addition, the pathline coincides with the streamline in steady flow. Therefore, no fluid exchange occurs between the upper and lower areas across the pathline associated with the contaminant particle. As a result, the total volume of fluid contained in the upper area does not change over time. In other words, the volume of fluid that flows into the upper area above the  $y_{ini}$  equals to the volume that exits the upper area above the  $y_f$ . This relationship is mathematically expressed

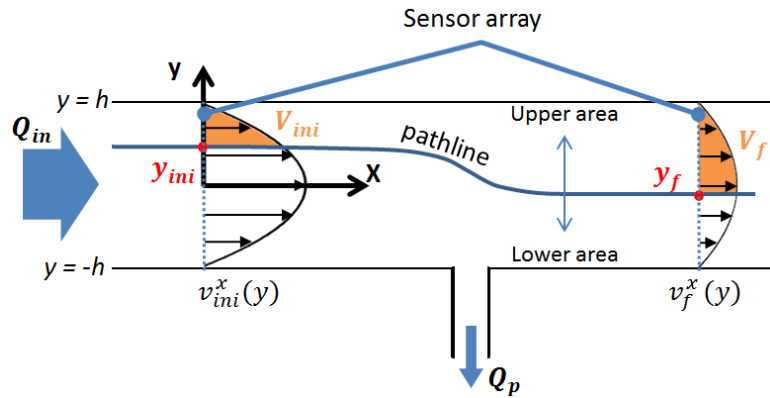


Figure 5.2: The modelling principle for the 2D APT model. The upper area is enclosed by the particle pathline, pipe upper boundary and the two sensor array. Because of steady laminar incompressible flow assumptions, the volume of fluid flows into the upper area above  $y_{ini}$  equals the volume that flows out above  $y_f$ .

in Equation (5.2)

$$\begin{aligned}
 V_{ini} &= \int_{y_{ini}}^h v_{ini}^x(y) dy \\
 V_f &= \int_{y_f}^h v_f^x(y) dy \\
 V_{ini} &= V_f.
 \end{aligned} \tag{5.2}$$

where  $V_{ini}$  is the volume that flows into the upper area in a unit time and  $V_f$  is the volume that flows out.

It should be noted that, because the velocity is purely in  $x$  direction, using the parabolic velocity profiles simplifies the problem. The integration path in Equation (5.2) is a straight line and is perpendicular to the  $x$  axis, making the integration straight forward to solve. In general, the integration path can be arbitrarily chosen. As long as the velocity profile along the integration path is known, Equation (5.2) can be formulated for conservation of volume and the following APT model formulation routine can be applied.

The parabolic velocity profiles in the upstream  $v_{ini}^x$  and the downstream  $v_f^x$  can be expressed as functions of  $Q_{in}$  and  $Q_p$  in Equation (5.3)

$$\begin{aligned}
 v_{ini}^x &= \frac{3}{4} Q_{in} \left( \frac{1}{h} - \frac{y^2}{h^3} \right) \\
 v_f^x &= \frac{3}{4} (Q_{in} - Q_p) \left( \frac{1}{h} - \frac{y^2}{h^3} \right).
 \end{aligned} \tag{5.3}$$

Substituting these expressions into Equation (5.2), the integration of the parabolic

velocity profiles are derived in Equation (5.4)

$$\begin{aligned}
V_{ini} &= \int_{y_{ini}}^h v_{ini}^x(y) dy \\
&= \int_{y_{ini}}^h \frac{3}{4} Q_{in} \left( \frac{1}{h} - \frac{y^2}{h^3} \right) dy \\
&= Q_{in} \left( \frac{y_{ini}^3}{4h^3} - \frac{3y_{ini}}{4h} + \frac{1}{2} \right),
\end{aligned} \tag{5.4}$$

and Equation (5.5)

$$\begin{aligned}
V_f &= \int_{y_f}^h v_f^x(y) dy \\
&= \int_{y_f}^h \frac{3}{4} (Q_{in} - Q_p) \left( \frac{1}{h} - \frac{y^2}{h^3} \right) dy. \\
&= (Q_{in} - Q_p) \left( \frac{y_f^3}{4h^3} - \frac{3y_f}{4h} + \frac{1}{2} \right).
\end{aligned} \tag{5.5}$$

The 2D APT model general form is established by linking  $V_{ini} = V_f$  in Equation (5.6)

$$Q_{in} \left( \frac{y_{ini}^3}{4h^3} - \frac{3y_{ini}}{4h} + \frac{1}{2} \right) = (Q_{in} - Q_p) \left( \frac{y_f^3}{4h^3} - \frac{3y_f}{4h} + \frac{1}{2} \right). \tag{5.6}$$

This equation is a simple third order polynomial algebraic relation among  $Q_{in}$ ,  $Q_p$ ,  $y_{ini}$  and  $y_f$ . Given the first three variables, only one of the three roots for  $y_f$  is the feasible solution within the range of the pipe size  $[-h, h]$ . In fact, any three of the  $y_{ini}$ ,  $y_f$ ,  $Q_{in}$ ,  $Q_p$  can be used as the model inputs to compute the fourth variable. For example, the APT model can be used for back tracking the source of a particle by providing  $Q_{in}$ ,  $Q_p$  and  $y_f$ . Furthermore, the APT model expression in Equation (5.6) can be rewritten as Equation (5.7) by introducing the dimensionless boundary control

representation  $U$

$$\frac{y_{ini}^3}{4h^3} - \frac{3y_{ini}}{4h} + \frac{1}{2} = (1 - U)\left(\frac{y_f^3}{4h^3} - \frac{3y_f}{4h} + \frac{1}{2}\right), \quad (5.7)$$

where  $U = Q_p/Q_{in}$  is the ratio between the boundary port flow rate and the upstream inlet flow rate. This dimensionless number represents the portion of the removed fluid from the total amount of fluid. No boundary control is expressed as  $U = 0$  and removing all fluid through the boundary port is  $U = 1$ . The conditions associated with particle on the boundary wall should be avoided because of the no slip boundary condition assumed in laminar flow.

The expression defined in Equation (5.7) is the APT model referred to in the following dissertation context. Common polynomial roots solver can be used on the APT model to compute  $y_f$  or  $y_{ini}$ . The APT model can be used in three different ways:

1. When the APT model is used to compute the  $y_f$  given  $U$  and  $y_{ini}$ , there is at most one distinct root to the polynomial within the range of the pipe geometry  $[-h, h]$ . If no solution exists within  $[-h, h]$ , the contaminant particle is removed by the boundary port flow. A special case is  $U = 1$ . In this case, the only mathematically feasible  $y_{ini}$  is  $y_{ini} = h$  and there is infinite number of solutions to  $y_f$ . However,  $y_{ini} = h$  refers to the situation when the particle is initially on the boundary wall. This situation is physically meaningless when zero slip boundary condition is used for laminar flow. The particle stays on the boundary wall forever.

2. When the APT model is used to compute the  $y_{ini}$  given  $U$  and  $y_f$ , there is at most one distinct root to the polynomial within the range of the pipe geometry  $[-h, h]$ . If no solution exists within  $[-h, h]$ , the contaminant particle is originated from the boundary port flow.

3. When the APT model is used to compute the required boundary control  $U$

that moves the particle from  $y_{ini}$  to  $y_f$ , domain  $y_f \in [-\infty, h)$  is applied for a feasible solution. In fact,  $y_f \in [-\infty, -h]$  computes the boundary control magnitude that removes the contaminant particle.  $y_f \geq h$  is physical meaningless.

For the contaminant elimination problem, the required boundary control that removes the contaminant can be computed using Equation (5.8)

$$U \geq 1 - \frac{\frac{y_{ini}^3}{4h^3} - \frac{3y_{ini}}{4h} + \frac{1}{2}}{\frac{y_f^3}{4h^3} - \frac{3y_f}{4h} + \frac{1}{2}} = 1 - \frac{y_{ini}^3}{4h^3} - \frac{3y_{ini}}{4h} + \frac{1}{2}, \quad (5.8)$$

where  $y_f = -h$  is used.

### 5.1.2 APT Model Properties and CFD Validation

The first observation of the VPT model in Equation (5.7) is that, given the contaminant particle initial location, its final location only depends on the ratio of the  $Q_{in}$  and  $Q_p$  instead of their absolute value. This observation is validated via finite element CFD simulated results shown in Fig. 5.3.

Two cases ( $U = 0.2$  and  $U = 0.4$ ) are shown as examples in Fig. 5.3. In each case, two particles, which were released at different initial locations, were tracked for different  $Q_{in}$  magnitudes. The boundary port was centered at  $x_p = 21.4$  cm with its width  $w = 12.7$  mm. Parabolic velocity profile was assigned at  $x = 0$  cm and the velocity profiles at the end of the pipe  $x = 60$  cm had already fully developed. It should be noted that, due to the laminar flow solver setting in the Ansys Fluent software environment, laminar flow is developed within much shorter distance than real world case, in which the entrance length is much longer than the CFD simulated results in Fig. 5.3. The Reynolds number in this example varies from 516 to 2581 for five different  $Q_{in}$  values with room temperature liquid water as the fluid. It is obvious that the tracked particles move to the same final locations regardless of the flow magnitude. These validations also suggest that the APT model is suitable for



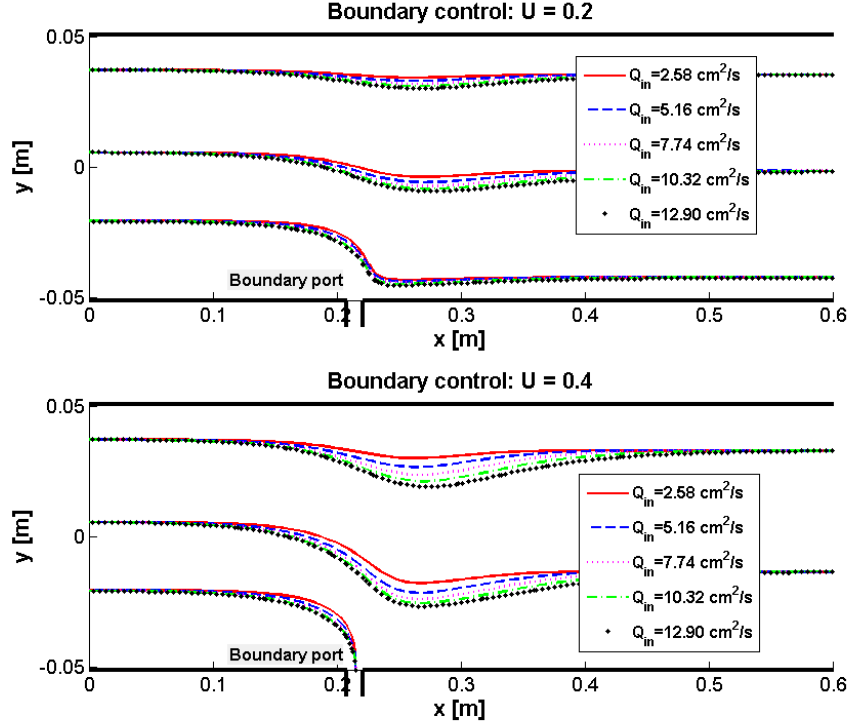


Figure 5.3: CFD validation of the flow phenomenon that is implied by the APT model. Given a particle initial location, its final location only depends on the ratio of  $Q_p$  and  $Q_{in}$ . In the example, the upstream total flow rate  $Q_{in}$  varies from  $2.58\text{cm}^2/\text{s}$  to  $12.90\text{cm}^2/\text{s}$ .

flow of different Reynolds number within the laminar limit.

The APT model is validated via CFD simulated results. Five cases of different  $U$  values are shown in Fig. 5.4 for illustration. Parabolic velocity profile was assigned on the inlet boundary at  $x = 0$  for the  $U = 0.1, 0.4, 0.7, -0.3$  cases. The last  $U = 0$  case used uniform velocity profile for demonstrating the APT model with different velocity profile. Fifteen particles with distinct initial locations  $y_{ini}$ s are tracked in each case. The top graph in Fig. 5.4 shows the CFD simulated particles trajectories (pathlines) for the  $U = 0.7$ . The graph below it extracts the particles initial locations  $y_{ini}$ s at  $x = 0$  and their final locations  $y_f$ s at  $x = 0.4$ , showing the relationship between  $y_{ini}$  and  $y_f$  given  $U = 0.7$ . The same data extraction process was applied for the other four cases. The red dashed line is the APT model solved curve, which lies exactly on

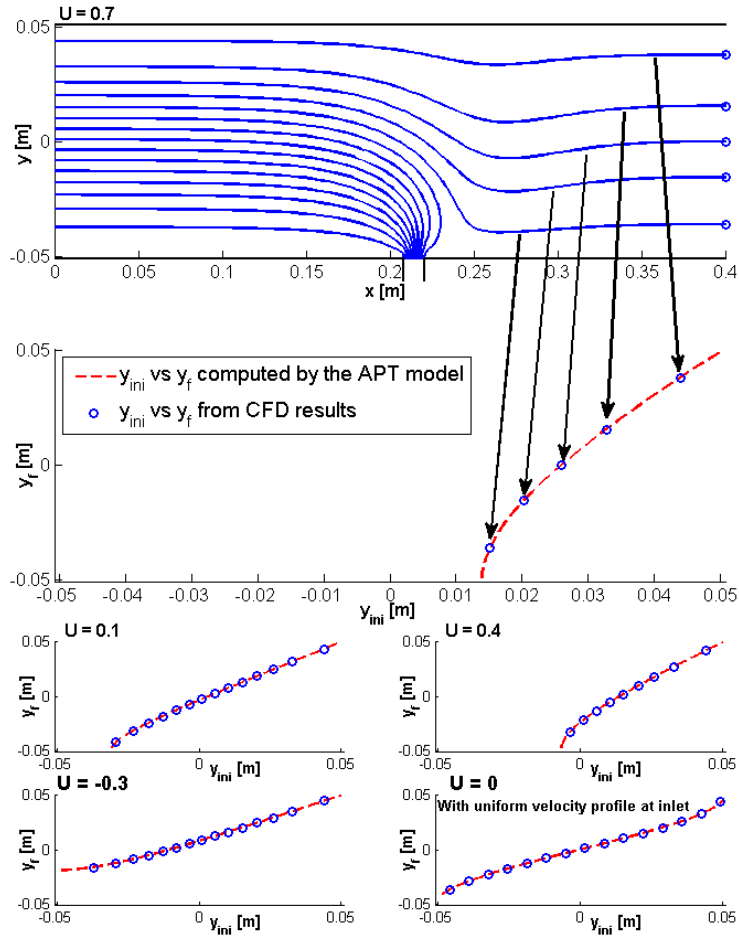


Figure 5.4: APT model validation using CFD simulated results. The top graph shows the CFD simulated 15 particle trajectories with  $U = 0.7$ . The particle initial locations  $y_{ini}$ s and the corresponding final locations  $y_f$ s are paired and plotted in the top second graph. The  $y_{ini}$   $y_f$  pairs lies exactly on the APT model predictions. The lower 4 graphs show the  $y_{ini}$   $y_f$  pairs for different control strength cases. The  $U = -0.3$  case represents port injection. The  $U = 0$  case used uniform velocity profile in the upstream, which case represents the pipe entrance flow dynamics.

the CFD simulated results in all cases.

The  $U = 0$  case in Fig. 5.4 refers to a different flow condition, where an uniform

velocity profile, defined in Equation (5.9)

$$v_{ini}^x = \frac{Q_{in}}{2h}, \quad (5.9)$$

was applied at  $x = 0$ . This case represents the laminar pipe flow entrance flow dynamics. The APT model used for this uniform velocity profile case was modified accordingly by integrating an uniform velocity profile for the upstream in Equation (5.2). The resultant APT model is expressed as Equation (5.10)

$$\frac{h - y_{ini}}{2h} = (1 - U) \left( \frac{y_f^3}{4h^3} - \frac{3y_f}{4h} + \frac{1}{2} \right). \quad (5.10)$$

When the particle was removed by the boundary port flow, the APT model simply does not have any feasible solution of  $y_f$  within the  $[-h, h]$  geometry range. When  $U < 0$ , which represents boundary port injection, the particles move toward the opposite side of the port. In all cases, the APT model exhibits high accuracy with nearly zero error for predicting particle location in steady 2-dimensional laminar flow.

### 5.1.3 APT Model for the 3D pipe

As discussed previously in Section 2.1, the 3D pipe control problem is reduced down to the 2D problem that relies on the 3D pipe symmetric plane. In addition, for the contaminant elimination problem, the main objective is to find the required boundary control to remove the contaminant. Therefore, the discussion of 3D APT model is focused on creating an empirical relationship that can be used to compute the required boundary control magnitude for eliminating a contaminant particle, which is released on the pipe symmetric plane. This empirical relationship is similar to the 2D APT model described in Equation (5.8), which computes the boundary control  $U$  given the particle initial location  $y_{ini}$ .

Using the analogy of the 2D APT modelling principle, where the 1-dimensional

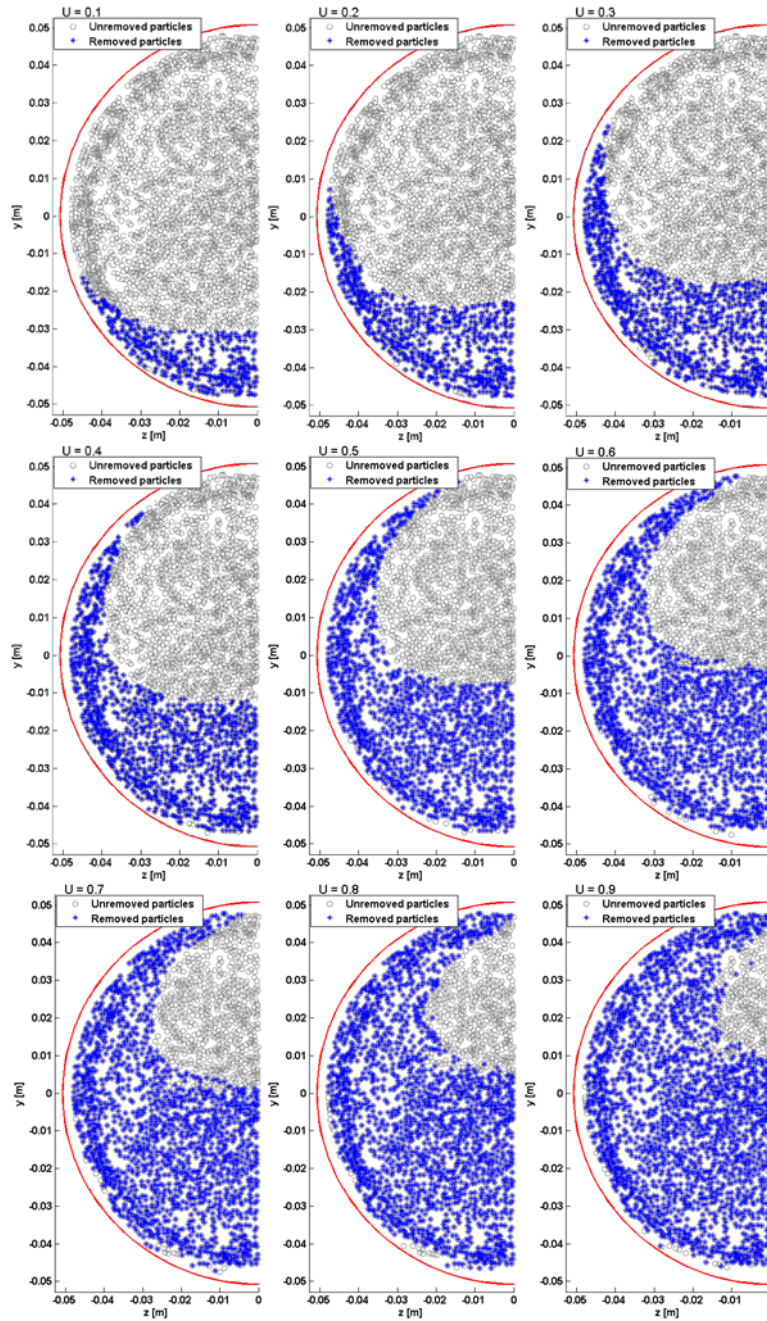


Figure 5.5: Steady flow CFD simulation results showing the propagation future of the particles in the 3D pipe. The graph shows the pipe inlet cross-section ( $x = 0$  slice), where hundreds of particles are released from. The blue marker represents the particles that will be removed by the boundary port flow, while the light gray marked particles flow into the downstream of the pipe without being removed. The boundary port is located at the bottom of the pipe  $y = -5.08$ cm

particle pathline splits the 2D pipe, the 3D APT model requires a 2-dimensional membrane to split the 3D pipe into two regions. Then, the conservation of volume principle is applied to establish the 3D APT model empirical relationship. Therefore, the main task is to define such a membrane for the 3D APT model.

For the contaminant elimination problem, given a boundary drawing  $U$ , the membrane is defined as the envelop that encloses all the pathlines that start on the upstream inlet boundary and end in the boundary port. The intersection of this membrane and the upstream inlet boundary is visualized in Fig. 5.5, where nine cases of different  $U$ s were generated using finite element CFD simulations. The inlet flow rate used in the CFD simulations is  $Q_{in} = 10.3 \text{ cm}^3/\text{s}$ . The pipe radius is  $h = 5.08 \text{ cm}$  and boundary port diameter is  $w = 1.27 \text{ cm}$ .

In each case in Fig. 5.5, the blue dots represent the starting points of the pathlines that ends in the boundary port. The contaminant particles that follow these pathlines are removed by the boundary port drawing flow. The gray dots, on the other hand, represent the pathlines of the unremoved particles. Therefore, the intersection of the membrane and the upstream inlet boundary is designated by the border that separates the blue dots region and gray dots region. Forming the whole membrane, this border line sweeps into the paper and gradually bends down into the boundary port. The membrane changes according to the boundary control  $U$  as shown by the nine cases in Fig. 5.5. Given a boundary control magnitude  $U$ , the membrane separates the removed and unremoved(remaining) particles and splits the 3D pipe into the upper and lower two regions. Analogous to the 2D APT model, the 3D APT model is formulated by applying conservation of volume in the upper region. The volume of fluid that flows into the upper region cross the inlet boundary can be computed by integrating the inlet boundary velocity profile over the gray dot region(the unremoved particle region shown in Fig. 5.5). For fully developed laminar flow with parabolic velocity profile, this integration is illustrated in Fig. 5.6 and is

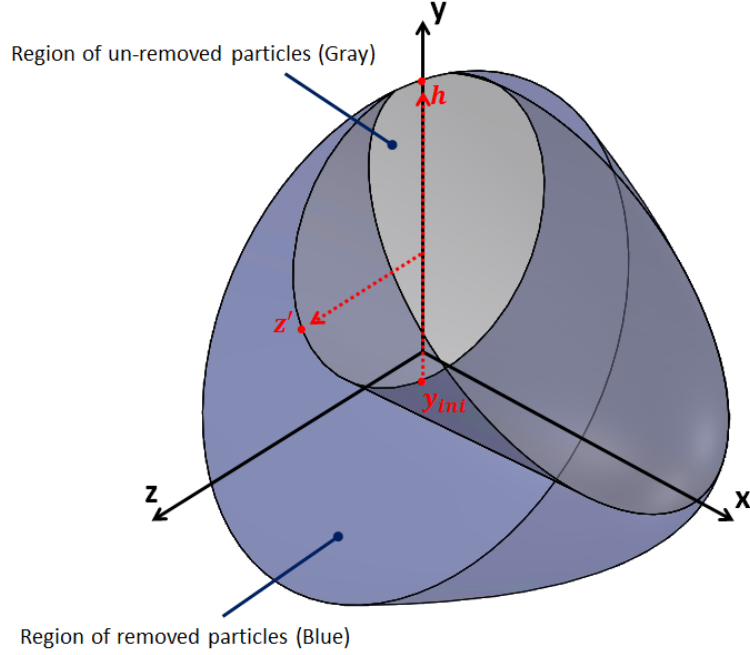


Figure 5.6: The 3D APT model integration region on the 3D pipe upstream inlet boundary. The blue region represents the area of fluid that will be removed by the boundary drawing action, while the gray region is the fluid that will not be removed. Based on the APT modelling methodology, the integration of parabolic velocity profile over the un-removed fluid region in the upstream inlet should equal to the overall integration of the the parabolic velocity profile in the downstream.

defined in Equation (5.11)

$$V_{ini} = \int_{y_{ini}}^h 2 \int_0^{z'} \frac{2Q_{in}}{\pi h^2} - \frac{2Q_{in}}{\pi h^4} (y^2 + z^2) dz dy, \quad (5.11)$$

where the  $V_{ini}$ , analogous to Equation (5.2), is the volume of fluid that flows into the upper gray region. The  $y_{ini}$  is the contaminant particle initial location on the symmetric plane. It should be noted that, circular shaped gray area is used in Fig. 5.6 for demonstration purpose only. The actual shape of the gray area is shown in Fig. 5.5.

To compute the integration defined in Equation (5.11), the shape of the gray area is required. In other words, the integration limit  $z'$  needs to be defined. However,

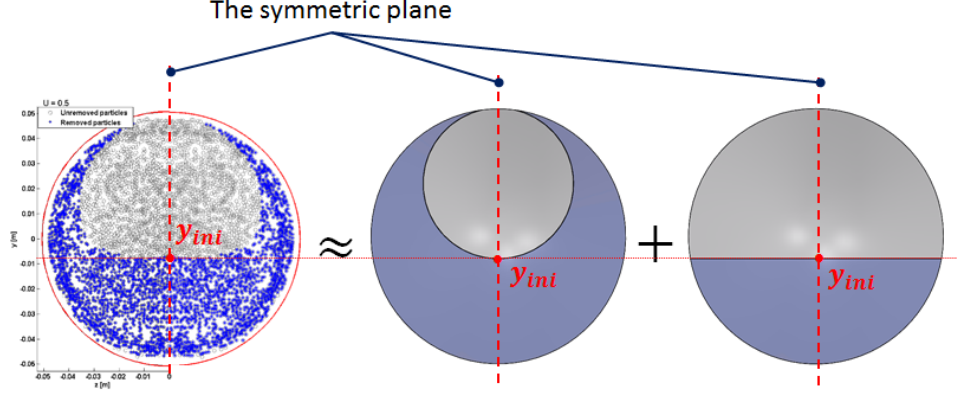


Figure 5.7: The 3D APT model integration area approximated by combining a circle and a segment. The left graph is the cross-section of the 3D pipe inlet boundary, on which the regions of fluid that will be removed and will not be removed by the boundary port flow are differentiated. Based on the APT model modelling methodology, an integration over the un-removed fluid region need to be computed. The integration of this un-removed fluid region is approximated by a combination of the integration over a circle area and the integration over an segment area.

a mathematical expression of the gray area envelop shape is complex and may leads to integration difficulty. By looking at the envelopes for different  $U$  values shown in Fig. 5.5, it is observed that, the envelope is similar to a horizontal cut when  $U$  is small, while when  $U$  gets larger, the envelope gradually turns to a circular shape. This observation leads to the empirical approximation of the integration over the gray area by combining two different integrations: an integration over a circle area and an integration over the segment area. This approximation is illustrated in Fig. 5.7. (The + sign does not refer to mathematical summation.)

The integration over the circular area from  $y_{ini}$  to  $h$  is computed in Equation (5.12)

$$\begin{aligned}
 V_{circle} &= \int_{y_{ini}}^h 2 \int_0^{\sqrt{d^2 - (b-y)^2}} \frac{2Q_{in}}{\pi h^2} - \frac{2Q_{in}}{\pi h^4} (y^2 + z^2) dz dy \\
 &= \frac{2Q_{in} b^2}{h^2} \left( 1 - \frac{b^2}{2h^2} - \frac{d^2}{h^2} \right),
 \end{aligned} \tag{5.12}$$

where  $V_{circle}$  is the volume that flows across the circular region in unit time. The  $b$  and  $d$  are coordinate transformers defined in Equation (5.13)

$$b = \frac{h + y_{ini}}{2}, d = \frac{h - y_{ini}}{2}. \quad (5.13)$$

The integration over the segment area from  $y_{ini}$  to  $h$  is computed in Equation (5.14)

$$\begin{aligned} V_{segment} &= \int_{y_{ini}}^h 2 \int_0^{\sqrt{h^2 - y^2}} \left( \frac{2Q_{in}}{\pi h^2} - \frac{2Q_{in}}{\pi h^4} (y^2 + z^2) \right) dz dy \\ &= Q_{in} \left( \frac{\arccos\left(\frac{y_{ini}}{h}\right)}{\pi} - \frac{y_{ini} \sqrt{h^2 - y_{ini}^2}}{\pi h^2} \frac{5h^2 - 2y_{ini}^2}{3h^2} \right), \end{aligned} \quad (5.14)$$

where  $V_{arc}$  is the volume that flows across the upper segment region in unit time.

Combining Equation (5.12) and Equation (5.14), the empirical approximation for the integration over the actual gray area  $V_{actual}$  is approximated by Equation (5.15)

$$V_{actual} = V_{circle} \operatorname{erf}\left(\frac{c_1(y_{ini} - c_2) + 1}{2}\right) + V_{segment} \left(1 - \operatorname{erf}\left(\frac{c_1(y_{ini} - c_2) + 1}{2}\right)\right), \quad (5.15)$$

where erf is the error function that smooth  $V_{actual}$  value from  $V_{arc}$  to  $V_{circle}$ . The  $c_1$  and  $c_2$  are two tunable parameters for this empirical approximation.

To formulate the 3D APT model for computing the required control strength given a  $y_{ini}$ , the volume flow rate  $V_{actual}$  in Equation (5.15) should equal to the total volume flow rate in the pipe downstream. This relationship is represented in Equation (5.16)

$$V_{actual} = Q_{in} - Q_p. \quad (5.16)$$

Finally, inserting Equation (5.15) into (5.16) and using the dimensionless boundary control representation  $U$ , the 3D APT model empirical relationship is formulated



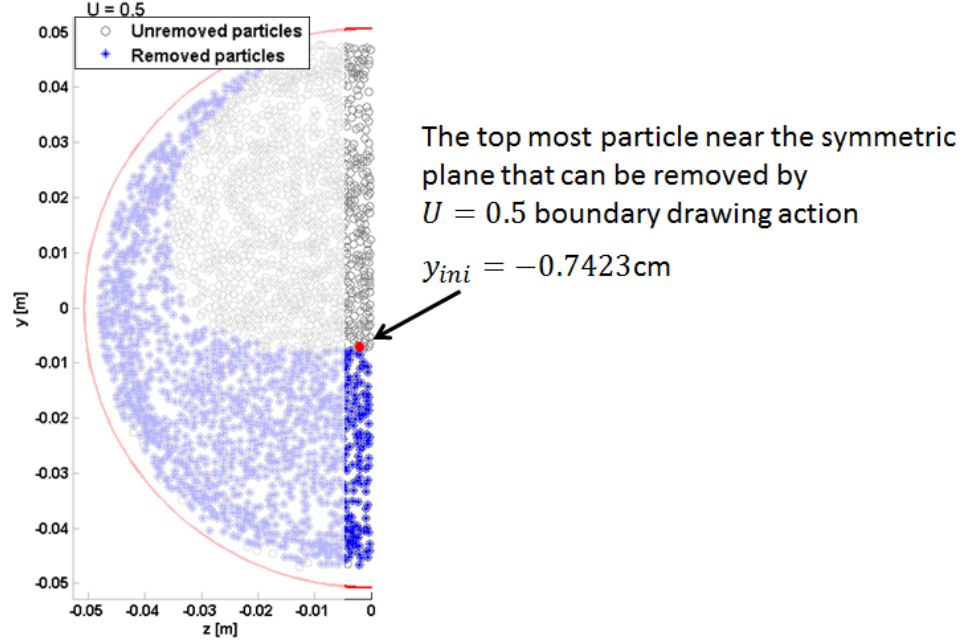


Figure 5.8: The furthest particle that can be removed by  $U = 0.5$  boundary drawing action has  $y_{ini} = -0.7423\text{cm}$ . In reverse,  $U = 0.5$  is the minimum control strength that should be applied to remove a particle that has  $Y_{ini} = -0.7423\text{cm}$ .

in Equation (5.17)

$$U = 1 - (V_{circle} \operatorname{erf}\left(\frac{c_1(y_{ini} - c_2) + 1}{2}\right) + V_{arc}(1 - \operatorname{erf}\left(\frac{c_1(y_{ini} - c_2) + 1}{2}\right))) \frac{1}{Q_{in}}. \quad (5.17)$$

This equation is used in the same way as the 2D APT model expressed in Equation (5.8) for computing the required boundary control magnitude to remove a contaminant particle of initial location  $y_{ini}$  on the pipe symmetric plane.

To tune the two parameters  $c_1$  and  $c_2$  in Equation (5.16), the top most (furthest away from the pipe bottom) particle that is removed by the boundary drawing action is picked from each case in Fig. 5.5. For example, the furthest particle that can be removed by  $U = 0.5$  boundary control is located at  $y_{ini} = -0.7423\text{cm}$  within a 1cm gap near the symmetric plane as shown in Fig. 5.8.

The top most particles for all 9 cases in Fig. 5.5 are extracted and are plotted in

Fig. 5.9, represented by the blue circles. The  $x$  axis is the coordinate of the furthest particle that can be removed given a boundary control  $U$ . The parameters  $c_1$  and  $c_2$  were tuned based on the first 7 data points, which are below  $y_{ini} = 0$  line. This is because in the multiple boundary ports control system described in Section 2.1, when the particle is observed in the upper half of the pipe, the boundary port installed on top of the pipe will be used. Therefore, only the data points with  $y_{ini} \leq 0$  were considered. The parameter tuning results are  $c_1 = 55.00$  and  $c_2 = -0.0138$  with average absolute error of 0.14 cm for the first 7 data points.

The tuned 3D APT model empirical relationship defined in Equation (5.17) is plotted in Fig. 5.9 as the solid blue line. The red dashed line is plotted using only the integration over the segment area as Equation (5.18)

$$V_{segment} = Q_{in} - Q_p. \quad (5.18)$$

The black dashed line is plotted using only the integration over the circle area as Equation (5.19)

$$V_{circle} = Q_{in} - Q_p. \quad (5.19)$$

The tuned APT model empirical relationship smooths from the red dashed line to the black dashed line as described in Equation (5.17).

Above 3D APT model was tuned based on  $Q_{in} = 10.3 \text{ cm}^3/\text{s}$  cases shown in Fig. 5.5. The parametric study shown in Fig. 5.10 suggests that the tuned 3D APT model empirical relationship is valid for other cases with different inlet flow rate  $Q_{in}$ . In Fig. 5.10, the six cases share the same boundary control  $U = 0.5$ . The total inlet flow rate varies from  $Q_{in} = 10.30 \text{ cm}^3/\text{s}$  to  $Q_{in} = 102.96 \text{ cm}^3/\text{s}$ . It can be observed that, near the symmetric plane, the furthest particles that can be removed by the boundary control has similar  $y_{ini}$ s. This observation is further illustrated in

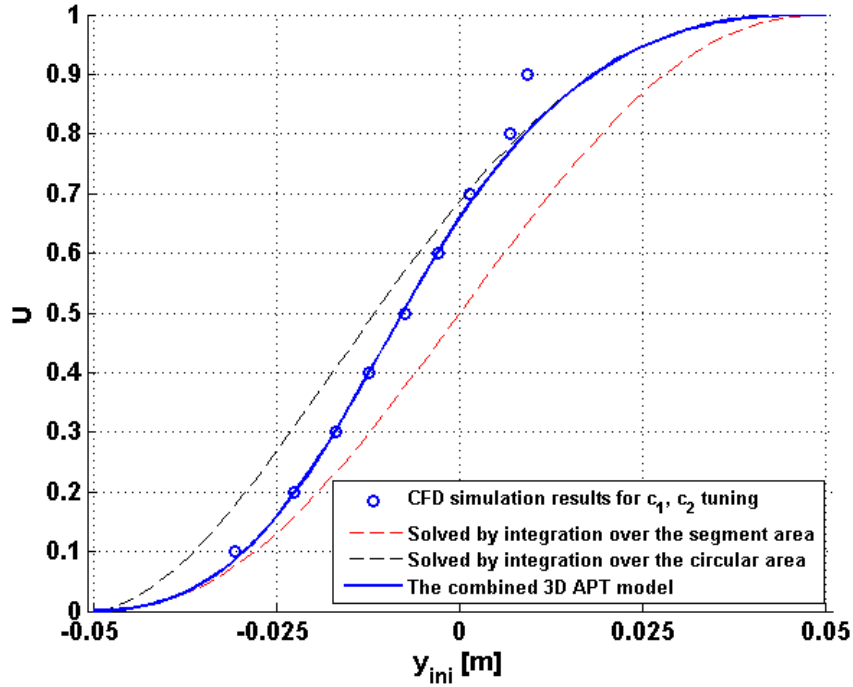


Figure 5.9: The 3D APT model compared with CFD simulated results. The error bar indicates variation when the total flow rate  $Q_{in}$  changes. The black dashed line is the  $U - y_{ini}$  relation by pure integration over the circle area, while the red dashed line is the pure integration over the segment area. The 3D APT model lies within the black and red curve by combining the two.

Fig. 5.11, where the  $y_{ini}$ s for different  $Q_{in}$ s are compared. When the boundary control magnitude  $U$  stays the same, the furthest distance that a particle can be removed by applying  $U$  does not vary much.

As a result, the 3D APT model empirical relationship can be universally used for various flow rates. However, it should be noted that the modelling is proposed for laminar flow. The Reynolds number of the  $Q_{in} = 102.96\text{cm}^3/\text{s}$  case in Fig. 5.10 reaches  $Re = 1290$ , which enters the transition region between laminar flow and turbulence flow. The 3D APT model for higher Reynolds number flow is beyond the scope of this dissertation. Due to the limited number of contaminant injection spots in the prototype system, prototype validation of the 3D APT model was not

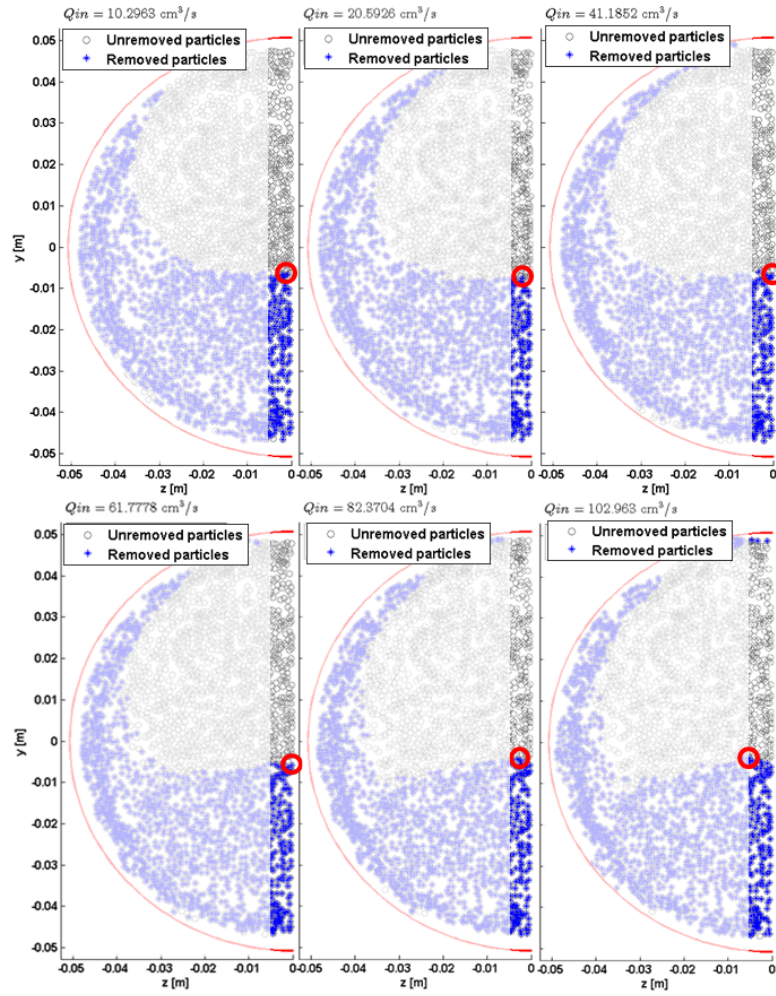


Figure 5.10: CFD simulations illustration showing the 3D APT model can be applied for flow of different flow rate  $Q_{in}$ . The furthest distance ( $y_{ini}$ ) of a particle that can be removed by the boundary control  $U = 0.5$  is similar when  $Q_{in}$  changes. Further illustration is shown in Fig. 5.11.

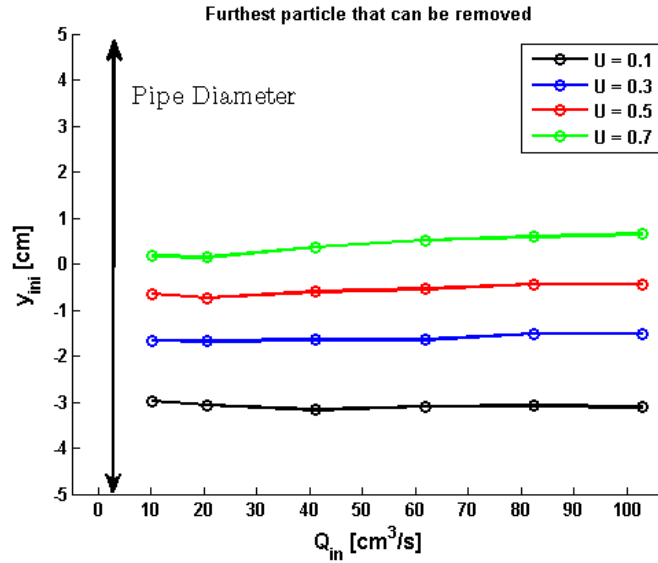


Figure 5.11: CFD simulated results illustrating the 3D APT model can be applied for flow of different flow rate  $Q_{in}$ . The  $x$  axis is the furthest distance of a particle that can be removed by the boundary control  $U$ .

presented studied.

## 5.2 The VPT Model - Particle Trajectory Approximation for Transient Laminar Flow

The 2D APT model presented in the previous section specifies the final location of a particle given its initial position in the pipe upstream and a constant boundary port flow. The advantage of the APT model is that it can be solved almost instantly. It is of interest however, to predict the particle location when the boundary port flow is non-constant in time. For example, a pulse of certain fixed port flow for a duration while the particle is traveling from the detection point (upstream sensor) till a bit after the boundary port location. The Velocity field Particle Tracking (VPT) model discussed in this section is a such model that approximates the full particle trajectory in both spatial and time domain. This model is a two states ODE, which can be solved using a common ODE solver. Although the VPT model requires more computational

time to solve than the APT model does, it demonstrates high efficiency compared to traditional CFD methods when an iterative controller is considered.

### 5.2.1 Modelling Technique

The general form of the VPT model is defined in Equation (5.20),

$$\begin{aligned}\dot{x} &= v_x(x, y, Q_{in}, U(t)) \\ \dot{y} &= v_y(x, y, Q_{in}, U(t)),\end{aligned}\tag{5.20}$$

where the boundary control  $U(t) = Q_p(t)/Q_{in}$  is inherited from the APT model (refer to Equation (5.7)). The right hand side of the VPT model,  $v_x(x, y, Q_{in}, U(t))$  and  $v_y(x, y, Q_{in}, U(t))$ , approximates the velocity field in the Cartesian coordinate system. The velocity field is a function of spatial location  $x$  and  $y$ , total flow rate  $Q_{in}$  and the boundary control  $U(t)$ . The VPT model uses the boundary velocity conditions around

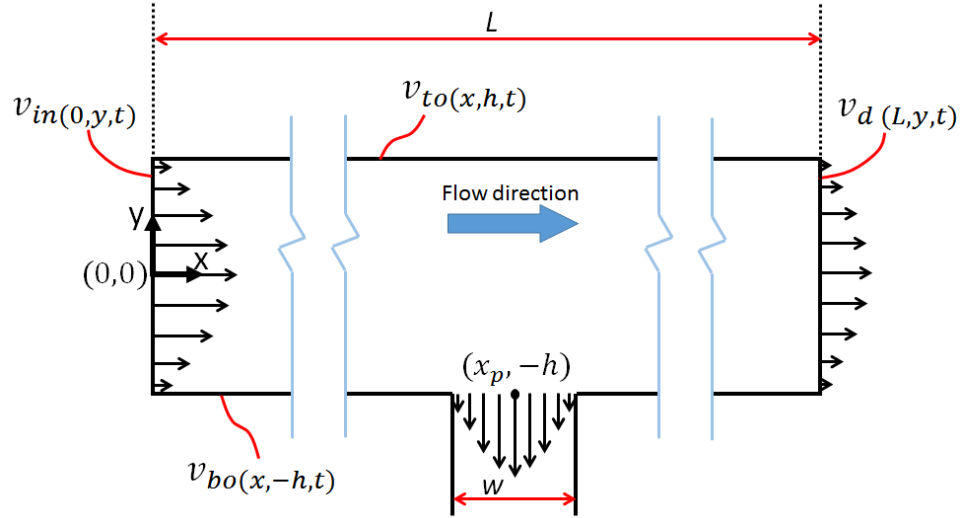


Figure 5.12: The 2D pipe region enclosed by four boundaries defined for VPT model development. The flow region to be modelled are enclosed by: the upstream boundary inlet boundary with parabolic velocity profile  $v_{in}$ , the no slip top boundary condition  $v_{to}$ , the downstream boundary with parabolic velocity profile  $v_d$  and the bottom boundary  $v_{bo}$ . The  $v_{bo}$  includes the information of the boundary port flow.

the flow region to approximate the velocity field in the two coordinate directions. In this study, low Reynolds number ( $Re \leq 100$ ) incompressible flow was considered for simplicity. Although not quantitatively analyzed, when  $Re \leq 100$ , the particle spatial trajectories for different Reynolds numbers have minor differences.

For the 2D pipe problem, four velocity boundaries are sufficient to enclose the flow region that is of interest. A 2D pipe of length  $L$  with a boundary port centered at  $x = x_p$  is defined in Fig. 5.12. The length of the channel is assumed long enough so that the flow is fully developed both in the very upstream  $x = 0$  and at the end of the pipe  $x = L$ . The four velocity boundaries in this example are top zero velocity boundary  $v_{to}$ , upstream parabolic velocity  $v_{in}$ , downstream parabolic velocity  $v_d$  and the bottom boundary  $v_{bo}$ . The  $v_{bo}$  includes the information of the boundary port flow. These four boundaries are mathematically represented in Equation (5.21)

$$\begin{aligned}
v_{in} &= \frac{3}{2} \frac{Q_{in}}{2h} \left(1 - \frac{y^2}{h^2}\right) \vec{i} \\
v_{to} &= 0 \\
v_{in} &= \frac{3}{2} \frac{Q_{in} - Q_p(t)}{2h} \left(1 - \frac{y^2}{h^2}\right) \vec{i} \\
v_{bo} &= \begin{cases} \frac{3}{2} \frac{Q_p(t)}{2w} \left(1 - \frac{(x-x_p)^2}{(0.5w)^2}\right) \vec{j} & : x \in [x_p - 0.5w, x_p + 0.5w] \\ 0 & : x \notin [x_p - 0.5w, x_p + 0.5w] \end{cases},
\end{aligned} \tag{5.21}$$

where no slip boundary wall condition is implied and parabolic velocity profile is assumed for the boundary port flow.

The major task of VPT modelling is to formulate  $v_x(x, y, Q_{in}, U(t))$  and  $v_y(x, y, Q_{in}, U(t))$  using the boundary velocity information in Equation (5.21). Due to the mathematical difficulty caused by the discontinuity in  $v_{bo}$ , the bottom boundary velocity is

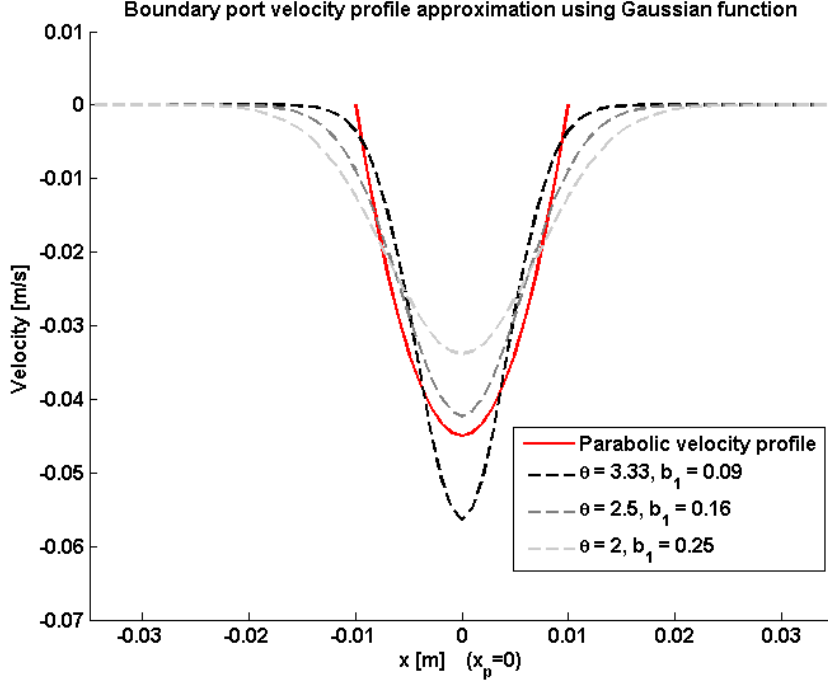


Figure 5.13: Approximation of the 2D pipe bottom boundary velocity profile using a Gaussian function.

first approximated by a Gaussian function in Equation (5.22)

$$\begin{aligned}
 v_{bo} &\approx -Q_p \theta \frac{1}{w\sqrt{\pi}} \exp\left(-\frac{(x-x_p)^2}{b_1 w^2}\right) \vec{j} \\
 &= -U Q_{in} \theta \frac{1}{w\sqrt{\pi}} \exp\left(-\frac{(x-x_p)^2}{b_1 w^2}\right) \vec{j},
 \end{aligned} \tag{5.22}$$

where  $\theta$  and  $b_1$  are two parameters to be tuned for the boundary port flow approximation. A comparison before and after this Gaussian function approximation is illustrated in Fig. 5.13 with different combinations of  $\theta$  and  $b_1$ .

The VPT model is finally defined in Equation (5.23)

$$\begin{cases} \dot{x} = \frac{3}{2} \frac{Q_{in}}{2h} \left(1 - \frac{y^2}{h^2}\right) (1 - \tau U(t)) \\ \dot{y} = -U(t) Q_{in} \theta \frac{(y-h)^2}{(2h)^2} \frac{1}{w\sqrt{\pi}} \exp\left(-\frac{(x-x_p)^2}{b_1 w^2 + b_2 (y+h)}\right) \end{cases}, \tag{5.23}$$



where  $\tau$  is a transition term defined in Equation (5.24),

$$\tau = 0.5 - \operatorname{erf}\left(-\frac{x - x_p}{a_1 + a_2(y + h)}\right) + 0.5\operatorname{erf}\left(-\frac{x - x_p}{a_3}\right). \quad (5.24)$$

There are totally six tunable parameters in the VPT model,  $\theta$ ,  $b_1$ ,  $b_2$ ,  $a_1$ ,  $a_2$  and  $a_3$ . The first two, as mentioned previously, approximate the bottom boundary velocity using the Gaussian function. Term  $(y - h)^2/(2h)^2$  and  $b_2$  create a smooth transition of the velocity field from the bottom boundary to the top boundary. The last three parameters used in  $\tau$  describe how  $v_{in}$  transitions to  $v_d$  over  $x$ . It can be easily checked by hand that, the right hand side of Equation (5.23) decays to the velocity representation defined in Equation (5.21) on the four boundaries, which are defined as  $x = 0$ ,  $x = L$ ,  $y = h$  and  $y = -h$ . In other words, the VPT model automatically satisfies the flow field boundary conditions.

### 5.2.2 VPT Model Parameterization

CFD simulation results were used for VPT parameter tuning and for model validations. In this study, the VPT model was tuned based on only one set of flow parameters:  $Q_{in} = 2.54\text{cm}^2/\text{s}$  and  $U(t) = 0.3$  constant in time. The 2D pipe has  $h = 5.08\text{cm}$ ,  $w = 1.27\text{cm}$  and  $x_p = 21.4\text{cm}$ . The six model parameters in Equation (5.23) were tuned by minimizing the error of spatial domain trajectory and the error of the particle velocity along these trajectories. The results after tuning are shown in Fig. (5.14) and Fig. (5.15) respectively for the nine spatial trajectories and the flow field velocities along these trajectories. The resultant parameters are  $\theta = 0.9571$ ,  $a_1 = 0.001$ ,  $a_2 = 0.5$ ,  $a_3 = 0.037$ ,  $b_1 = 1$  and  $b_2 = 0.01$ .

Due to the high nonlinearity of the VPT model equations, the parameter tuning process involved both manual observation and software package usage such as *Matlab* *Fmincon* function. Although the optimality of the parameterization result is not

discussed in this study, the tuning result shows high accuracy in both spatial domain and time domain.

The VPT model defined in Equation (5.23) approximates the full velocity field that is produced by the boundary control action. This velocity field is 3D visualized in Fig. 5.16. It is clearly illustrated that the four boundary velocity conditions are automatically satisfied: the flow is fully developed in upstream ( $x = 0$ ) and downstream ( $x = 0.4$ ) regions as the parabolic curvature; the flow has zero velocity on the upper boundary ( $y = h = 5.08$  cm); the boundary port flow is approximated by a Gaussian function on the  $y = -h$  boundary.

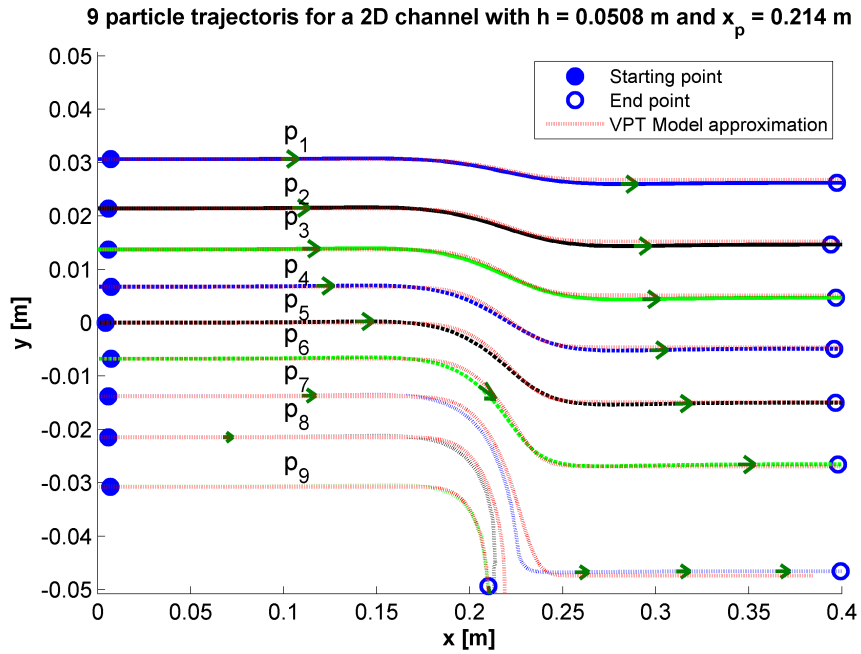


Figure 5.14: VPT model parameterization using CFD simulated particle spatial trajectories. The channel is  $2h = 0.1016$  meter in height with  $Q_{in} = 254$   $\text{mm}^2/\text{s}$  and  $U = 0.3$ . The boundary port is centered at  $x_p = 0.214$  m with width of  $w = 0.0127$  m. The trajectories of nine particles were used, out of which two particles were removed from the pipe by the boundary port flow.

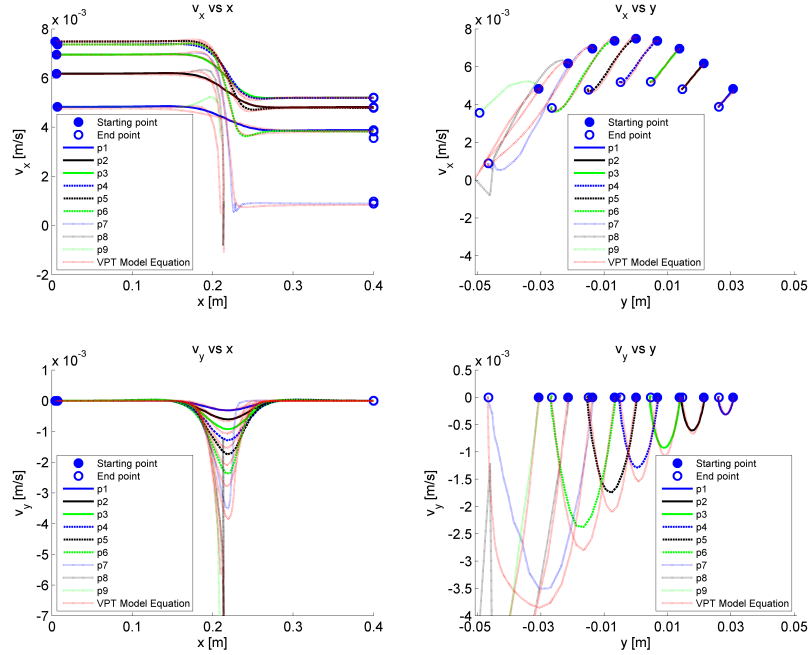


Figure 5.15: VPT model parameterization using CFD simulated flow velocity field along the particle trajectories. The particles' velocity components  $v_x$ s and  $v_y$ s along their trajectories represents the velocity field for steady flow. The solid blue dots represent trajectory start points on the upstream inlet boundary. The circles mark the particle final locations in the downstream(Refer to Fig. 5.14).

### 5.2.3 VPT Model validations and extension

The VPT model is validated using a variety of  $U(t)$ s and is extended to multiple boundary ports geometry in this section.

The first validation considers steady flow with different control magnitudes. Two examples ( $U(t) = 0.5$  and  $U(t) = -0.3$ ) are illustrated in Fig. 5.17, where the comparisons are shown in spatial domain and time domain separately. Seven particles were released and tracked from the flow inlet boundary ( $x = 0$ ) with different initial locations:  $y_{ini} = [-4, -3, -2, -1, 0, 1, 2, 3, 4]$  cm. Parabolic velocity profile was assigned on the inlet boundary. The boundary port was centered at  $x = 51.435$  cm with width  $w = 1.27$  cm. Despite of the minor error near the boundary port, which is

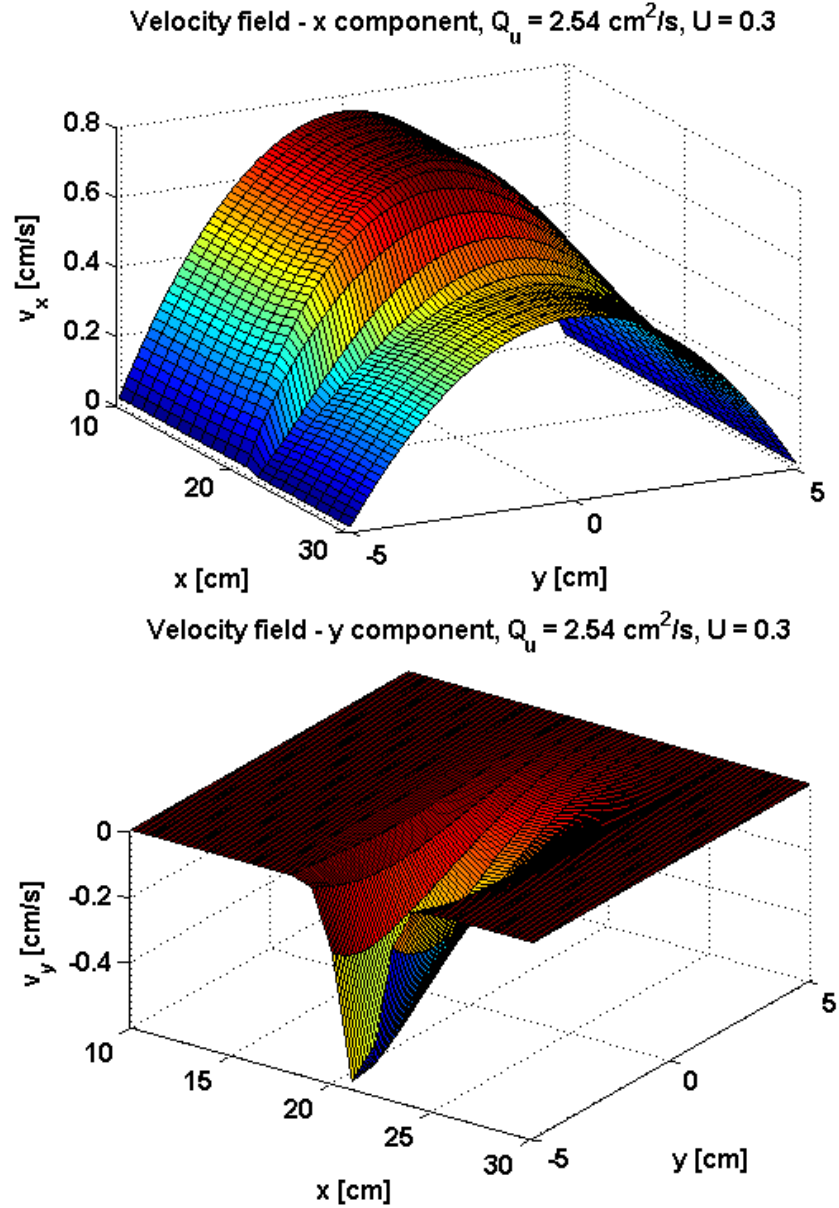


Figure 5.16: 3D visualization of the entire velocity field approximated by the VPT model. The flow conditions and geometries are  $Q_{in} = 2.54 \text{ cm}^2/\text{s}$ ,  $U = 0.3$ ,  $h = 5.08 \text{ cm}$ ,  $w = 1.27 \text{ cm}$  and  $x_p = 21.4 \text{ cm}$ .

mainly caused by the Gaussian function approximation of the boundary port flow, the VPT model exhibits high accuracy for approximating particle trajectories in both spatial domain and time domain. Remarkably, although only tuned by the  $U(t) = 0.3 > 0$  case, the VPT model can also be applied for boundary injection case when  $U < 0$ .

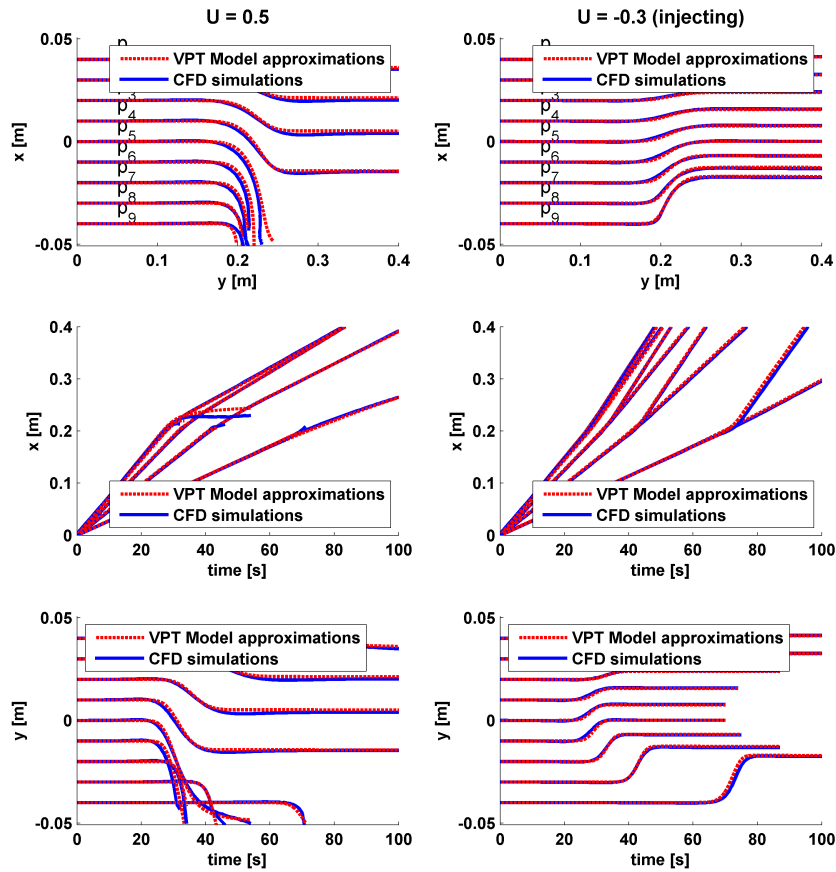


Figure 5.17: VPT model validation using CFD simulated results for steady flow. Two cases are shown for  $U = 0.5$  (left column) and  $U = -0.3$  (right column). In each case, the top graph shows the particle spatial trajectories; The middle and bottom graphs show the particle  $x$  and  $y$  propagation in time.

Using the VPT model, the computational effort to solve for each individual trajectory was around 0.11 second. Matlab ODE23 with tolerance of  $1e - 10$  was used on a 64-bit windows-7 laptop with Intel Core(TM) i5 M540 dual-core 2.53GHz CPU and 4GB RAM. While the steady state CFD simulation took around 25sec to solve on a 64-bit windows-7 workstation with Intel Xeon E5-1620 quad-core 3.6GHz CPU with 48GB RAM. The CFD model has around 40000 quadrilateral elements. The detailed VPT model *matlab* implementation code is included in Appendix B.

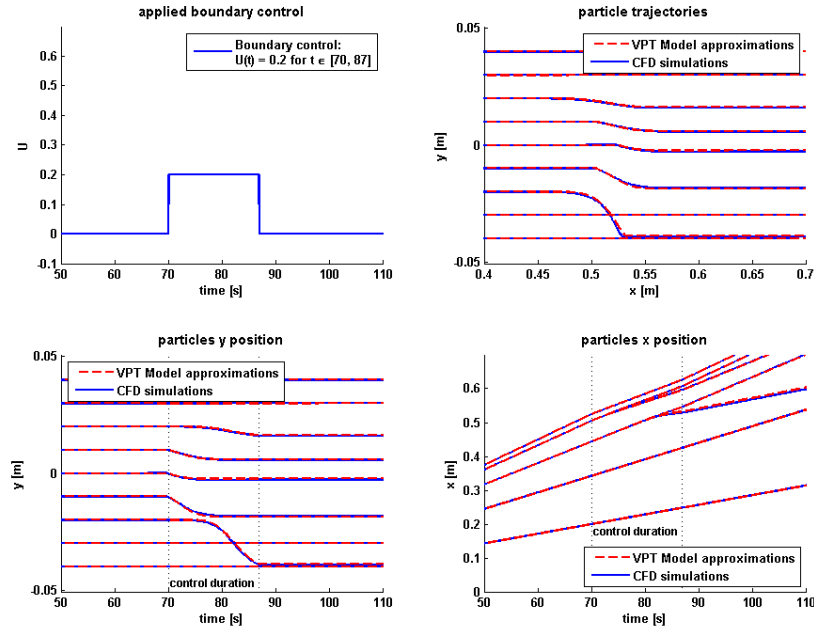


Figure 5.18: VPT model validation using CFD simulated results using square wave pulse boundary control. In this example, the boundary control is defined as a square wave function:  $U(t) = 0.2$  when  $70 \text{ sec} \leq t \leq 87 \text{ sec}$  and  $U(t) = 0$  elsewhere.

The second validation considers transient flow when the boundary control is not constant in time. Two examples are presented. The first example considers a square wave shaped boundary control. In this case, the control  $U(t) = 0.2$  was turned on at  $t = 70 \text{ sec}$  and was turned off at  $t = 87 \text{ sec}$ . The VPT model is compared with CFD simulated results in Fig. 5.18. The top-left graph shows the applied boundary control signal, which is mathematically represented as  $U(t) = 0.2$  for  $t \in [70, 78]$ . The top-right graph illustrates the seven particle spatial trajectories. Because the particles of different  $y_{ini}$ s have different  $x$  velocities described by the parabolic velocity profile, they advance towards the boundary port at different rates. If the particle moves too slow, it does not experience the boundary port downward drawing before the boundary control turns off at  $t = 87 \text{ sec}$ . For example, the two particles that starts from  $y_{ini} = -4 \text{ cm}$  and  $-3 \text{ cm}$  move straight right without any downward motion.

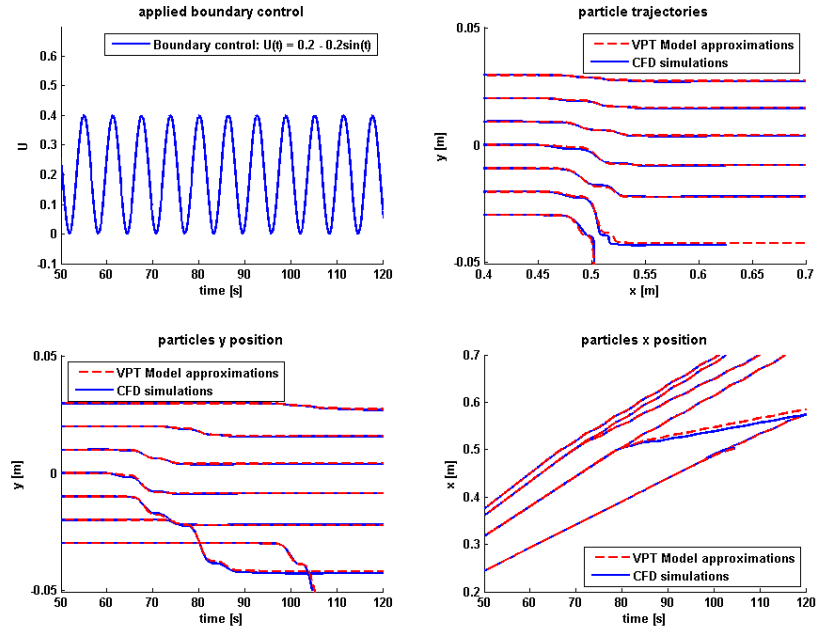


Figure 5.19: VPT model validation using CFD simulated results using sine wave boundary control. In this example, the boundary control is defined as a sine wave function:  $U(t) = 0.2 - 0.2\sin(1t)$  for  $t \geq 0$ .

The bottom two graphs in Fig. 5.18 compare the VPT model with the CFD simulated results in time domain. The second example shown in Fig. 5.19 considers a sine wave boundary control signal:  $U(t) = 0.2 - 0.2\sin(t)$  for  $t \geq 0$ . Both examples validate the functionality of the VPT model under time variant boundary control. Three more examples for validation are illustrated in Figure 5.20. In case c), good matching between the VPT model and finite element simulation results shows that, the VPT model accommodates to linear combinations of sine waves. This result implies that the VPT model can take any boundary control wave form, which can be expressed using Taylor expansion.

The third validation extends the VPT model to a flow system with multiple boundary ports that are installed at different locations. A three boundary ports pipe is shown as a particular example in Fig. 5.21. It can be observed from the VPT model Equation (5.23) that, the velocity field is linear with respect to  $U(t)$ . This linearity

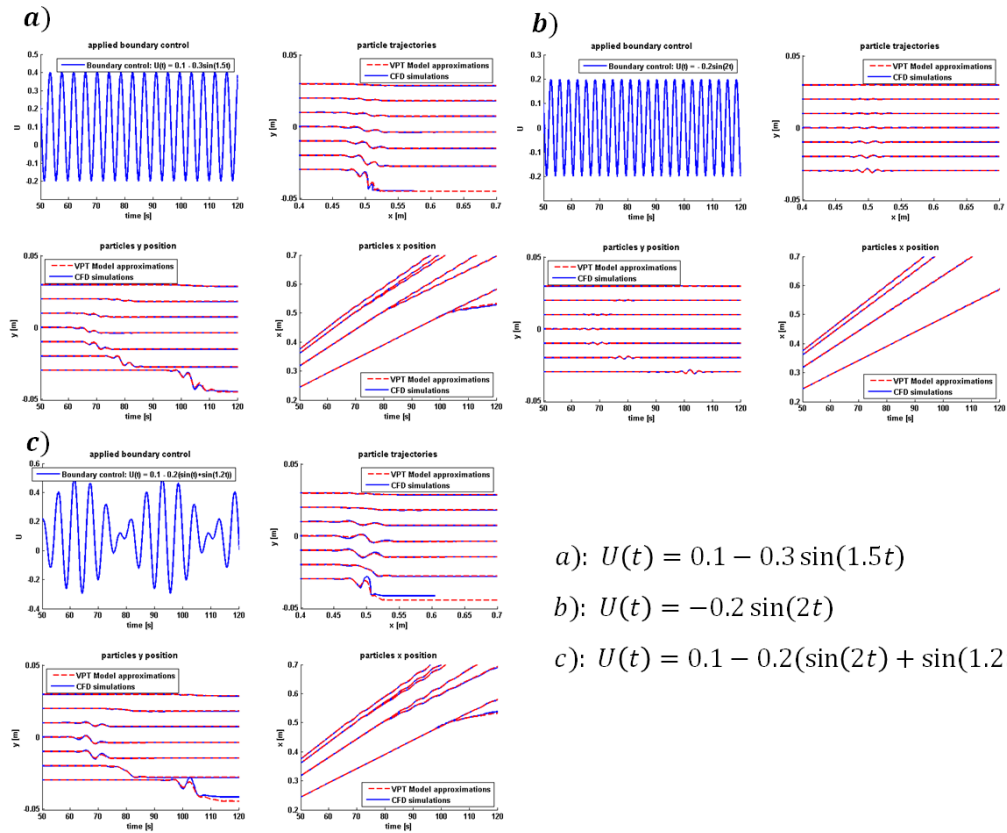


Figure 5.20: Additional VPT model validations using CFD simulated results using different sine waves for boundary control.



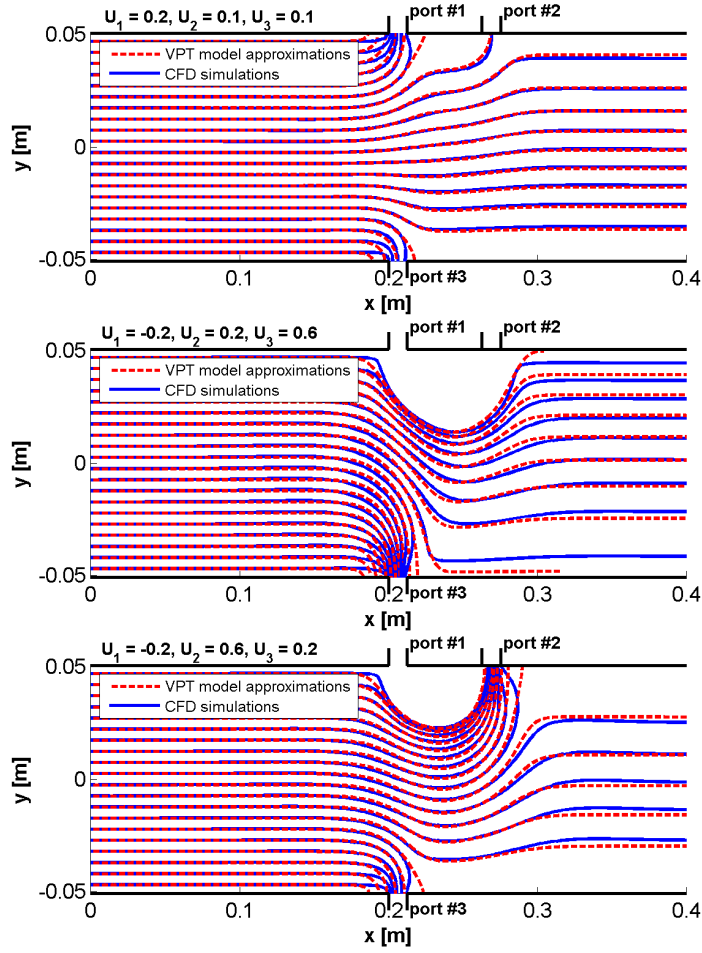


Figure 5.21: VPT model validation for a pipe with three boundary ports.

grants the VPT model the superposition property. The three boundary ports flow system can be modelled simply by superposing three VPT model equations without recalibrating the model parameters. The resultant 3-ports VPT model is formulated in Equation (5.25)

$$\begin{cases} \dot{x} = \frac{3}{2} \frac{Q_{in}}{2h} \left(1 - \frac{y^2}{h^2}\right) (1 - \tau_1 U_1 - \tau_2 U_2 - \tau_3 U_3) \\ \dot{y} = -v_{y1}(x_{p1}, U_1) - v_{y2}(x_{p2}, U_2) + v_{y3}(x_{p2}, U_2), \end{cases}, \quad (5.25)$$

where  $\tau_1, \tau_2, \tau_3, v_{y1}, v_{y2}$  and  $v_{y3}$  are defined in Equation (5.26)

$$\begin{aligned}
\tau_1 &= 0.5 - \operatorname{erf}\left(-\frac{x - x_{p1}}{a_1 + a_2(-y + h)}\right) + 0.5\operatorname{erf}\left(-\frac{x - x_{p1}}{a_3}\right) \\
\tau_2 &= 0.5 - \operatorname{erf}\left(-\frac{x - x_{p2}}{a_1 + a_2(-y + h)}\right) + 0.5\operatorname{erf}\left(-\frac{x - x_{p2}}{a_3}\right) \\
\tau_3 &= 0.5 - \operatorname{erf}\left(-\frac{x - x_{p3}}{a_1 + a_2(y + h)}\right) + 0.5\operatorname{erf}\left(-\frac{x - x_{p3}}{a_3}\right),
\end{aligned} \tag{5.26}$$

and Equation (5.27)

$$\begin{aligned}
v_{y1} &= -U_1 Q_{in} \theta \frac{(-y - h)^2}{(2h)^2} \frac{1}{w\sqrt{\pi}} \exp\left(-\frac{(x - x_{p1})^2}{b_1 w^2 + b_2(-y + h)}\right) \\
v_{y2} &= -U_2 Q_{in} \theta \frac{(-y - h)^2}{(2h)^2} \frac{1}{w\sqrt{\pi}} \exp\left(-\frac{(x - x_{p2})^2}{b_1 w^2 + b_2(-y + h)}\right) \\
v_{y3} &= -U_3 Q_{in} \theta \frac{(y - h)^2}{(2h)^2} \frac{1}{w\sqrt{\pi}} \exp\left(-\frac{(x - x_{p3})^2}{b_1 w^2 + b_2(y + h)}\right).
\end{aligned} \tag{5.27}$$

Each superposed equation used its corresponding boundary port locations  $x_{p1} = 0.206$ ,  $x_{p2} = 0.269$  and  $x_{p3} = 0.206$ . The associated boundary controls are named  $U_1, U_2$  and  $U_3$ . In this example, the #1 and #2 ports are installed on the pipe top boundary, and thus the  $y$  coordinate associated with these two ports is reversed in Equation (5.26). For illustration simplicity, all boundary controls are constants in time, making this example a steady flow problem. The results for three different boundary controls combinations are shown in Fig. 5.21. The computational effort for one trajectory by solving Equation (5.25) is around 0.219 seconds.

It is observed that the VPT modelling error is enlarged if the particle passes near a boundary port and is not removed by that port. This error is believed to be mainly caused by the Gaussian function approximation of the boundary port flow. Other methods for boundary port flow approximation can be studied in the future and the presented modeling routine can still be applied.

#### 5.2.4 VPT Model for Pipe of Different Flow Rates and Geometries

In Section 5.2.2, the VPT model was parameterized based on the reference 2D pipe geometry of height  $2h = 10.16$  cm, port width  $w = 1.27$  cm and upstream flow rate of  $Q_{in} = 2.54$  cm<sup>2</sup>/s. When the pipe geometry or the flow rate changes, the six model parameters may subject to change accordingly. This section presents a systematic method for adjusting the VPT model parameters according to geometry changes. This adjusting method enables the VPT model to be universally applicable on pipe of different sizes without re-conducting the parameter tuning process illustrated in Section 5.2.2.

In this study, three parameters define the pipe flow problem: the pipe height  $2h$ , the boundary port width  $w$  and the upstream flow  $Q_{in}$ . Therefore, three cases are investigated to cover all the combination of these three parameters. The first case considers changing the  $Q_{in}$ . The second case scales all three parameters by the same quantity. The third case considers a different boundary port size  $w$  without changing the  $h$  and  $Q_{in}$ . In addition, the VPT model is developed for low Reynolds number flow. The same Reynolds number  $Re = 10.16$  was used in all the examples presented in this section. The fluid kinematic viscosity changes accordingly to ensure same Reynolds number. For example, when  $Q_{in} = 2.54$  cm<sup>2</sup>/s,  $h = 5.08$  cm and  $w = 1.27$  cm, the required fluid kinematic viscosity is  $5e^{-5}$  m<sup>2</sup>/s.

##### Case 1:

The VPT model can be directly applied for different  $Q_{in}$  value without modifying its parameters. This is because the  $Q_{in}$  is an independent variable in the model equation (5.23). The VPT model is compared with CFD simulated results in Fig. 5.22 (left), in which the upstream inlet flow rate is  $Q_{in} = 5.08$ cm<sup>2</sup>/s, which is doubled from the original reference value of  $Q_{in} = 2.54$  cm<sup>2</sup>/s. The VPT model preserves its accuracy.

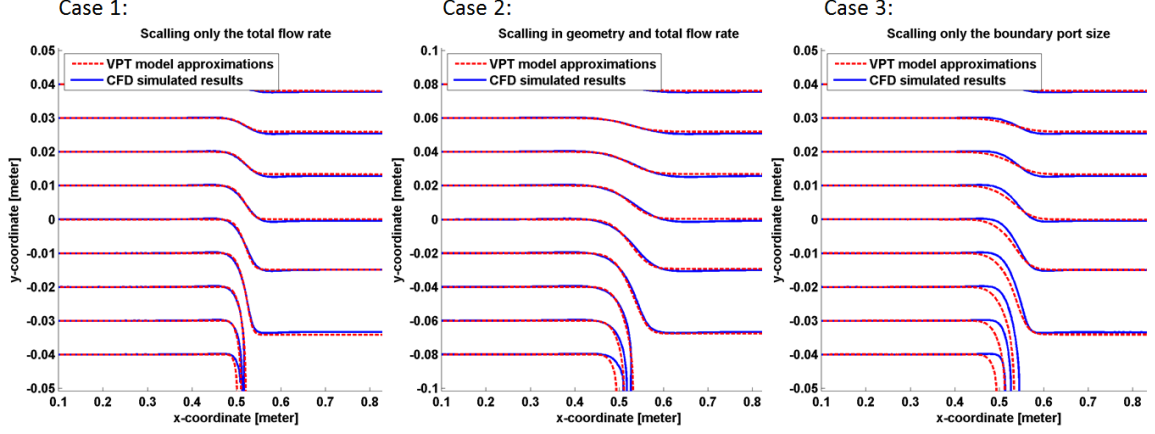


Figure 5.22: VPT model for different flow conditions and pipe geometries. Case 1: the flow rate  $Q'_{in} = 5.08\text{cm}^2/\text{s}$  is changed from the model reference value  $Q_{in} = 2.54\text{cm}^2/\text{s}$ , which was used for VPT model parameterization; Case 2: the geometry and flow rate are scaled up to  $h' = 10.16\text{cm}$ ,  $w' = 2.54\text{cm}$  and  $Q'_{in} = 5.08\text{cm}^2/\text{s}$ ; Case 3: only the boundary port size is increased to  $w' = 2.54\text{cm}$ .

### Case 2:

When all three parameters  $h$ ,  $w$  and  $Q_{in}$  are linearly scaled by the same amount  $\gamma_2$ , three of the VPT model parameters need to be zoomed accordingly as defined in Equation (5.28)

$$\begin{cases} a'_1 = \gamma_2 a_1 \\ a'_3 = \gamma_2 a_3 \\ b'_2 = \gamma_2 b_2 \end{cases}, \quad (5.28)$$

where the  $a'_1$ ,  $a'_2$  and  $b'_2$  are the VPT model parameters for the new geometry. Without losing generality,  $\gamma_2 = 2$  was used in the example in Fig. 5.22 (middle). Both pipe height and port width were doubled, resulting in  $h' = 20.32\text{ cm}$  and  $w' = 2.54\text{ cm}$ . The flow rate was also increased to  $Q'_{in} = 5.08\text{ cm}^2/\text{s}$ . The new VPT model is compared with a CFD simulated results in Fig. 5.22 (middle). The VPT model accuracy is lowered near the boundary port area. This lower accuracy is mainly caused by the worse boundary port flow Gaussian approximation when the boundary port is wider.

However, the VPT model overall accuracy is maintained for predicting particle final locations.

**Case 3:**

The boundary port size is changed in this case without changing the 2D pipe height and upstream flow rate. In the following example, the boundary port size is doubled,  $w' = \gamma_3 w = 2w = 2.54$  cm. Four VPT model parameters need to be adjusted as defined in Equation (5.29)

$$\left\{ \begin{array}{l} a'_1 = \gamma_3 a_1 \\ a'_2 = \gamma_3 a_1 \\ a'_3 = \gamma_3 a_1 \\ b'_2 = \gamma_3^2 b_2. \end{array} \right. \quad (5.29)$$

The resultant VPT model is compared with the CFD simulated results in Fig. 5.22 (right). The result shows lower accuracy near the boundary port region. This error is again mainly caused by the worse Gaussian approximation of the boundary port flow as the port size is enlarged.

In summary, the VPT model can be systematically adjusted for flow problems of any pipe sizes and flow rates, by combining the three cases shown above. For example, the VPT model for  $h = 2.54$  cm and  $w = 1.27$  cm can be formulated using  $\gamma_2 = 0.5$  and  $\gamma_3 = 2$ .

**5.2.5 Advantages of the VPT Model**

The VPT model establishes a convenient bridge between classical control algorithms and fluid dynamics. First, the VPT model is in the form of ODEs, which is the preferred mathematical formation for the classical control methods. For example, the state space oriented controllers require the ODEs to define the system states.

Secondly, the VPT model can be solved much faster than conventional CFD method, given that the particle trajectories are the physics that are of interest.

In real world applications, either a mathematical model or a empirical lookup table can be used for control purposes. Both the mathematical model and the lookup table are used to compute the output in response to an input. Usually, the lookup table is preferred if the database is small or the physics is too complex to model. However, when the database gets larger, both the data storage cost and lookup table index searching time cost increase. At this situation, a rational mathematical model comes into play.

For the contaminant elimination or relocation problem, given a boundary control action, the particle final location is the deterministic quantity need to be provided by the lookup table or the mathematical model. This section presents the advantage of the VPT model over a lookup table for flow control applications.

### 1. The VPT model requires less memory storage

For a lookup table, the required system memory depends greatly on how many variables are needed to specify a output. To simplify the discussion, square wave shaped boundary control is used for demonstration. A square wave boundary control can be defined using three variables, the magnitude  $U_o$ , the control turn on time  $t_{on}$  and turn off time  $t_{off}$ . With the addition of the contaminant initial location  $y_{ini}$  and the upstream flow rate  $Q_{in}$ , there are in total five inputs to define the output  $y_f$ . Thus, a five dimension lookup table is required. If each dimension takes 20 discrete values, the empirical lookup table contains  $20^5 = 3,200,000$  data points. A 3-dimensional sub-lookup table is illustrated in Fig. 5.23 for  $y_{ini} = 0$  and  $Q_{in} = 5.08 \text{ cm}^2$ . For easier illustration, the  $t_{on}$  and  $t_{off}$  are mapped into:  $V = U * (t_{off} - t_{on}) * Q_{in}$  and  $t_m = (t_{on} + t_{off})/2$ . The  $V$  represents the volume of fluid removed by the boundary port and the  $t_m$  represents the timing of the boundary control.

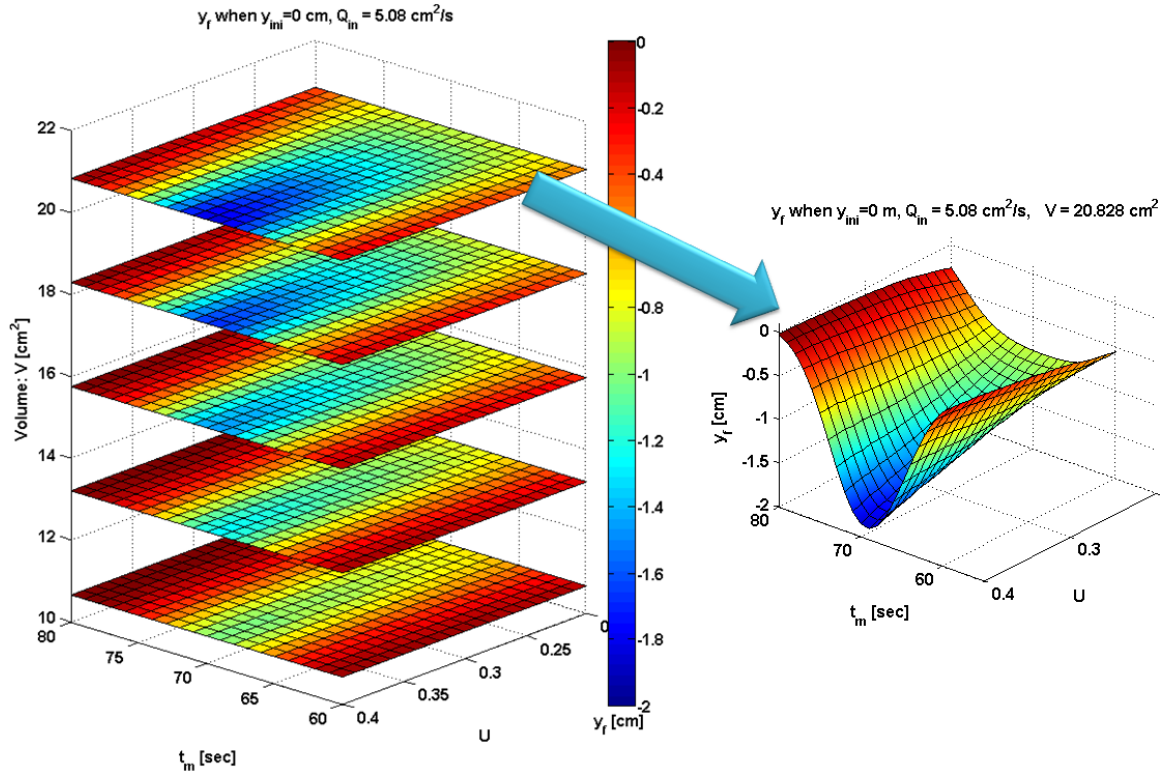


Figure 5.23: 4D empirical lookup table for the 2D flow problem.

Totally 2000 data points are shown in Fig. 5.23. The  $t_m$  and  $U$  take 20 values each. Five values for  $V$  are shown in the figure:  $[10.668, 13.208, 15.748, 18.288, 20.828] \text{ cm}^2$ . The color in the figure represents the control outcome  $y_f$ . Blue color indicates stronger control outcome. If 4 bytes float data type is used, the 2000 data points require  $2000 * 4 = 8000 \text{ byte}$  storage, which does not include the data for  $U$ ,  $t_{on}$ ,  $t_{off}$  and  $y_{ini}$ . It will consume  $3,200,000 * 4 \text{ byte} = 12.2 \text{ Mb}$  to store the full parametric study of 3,200,000 data points. While the VPT model takes 964 byte in the form of the matlab m file.

### The VPT model is flexible to construct flow system of higher complexity

As previously presented in Section 5.2.4, the VPT model can be easily stacked to construct a pipe flow system with multiple boundary ports, of which the veloc-

ity field overlaps with each other. However, the situation is hopeless when using a lookup table. A new parametric study has to be conducted for the new flow system. Moreover, each added boundary port contributes three more dimensions (the  $U$ ,  $t_{on}$ ,  $t_{off}$  associated with the new port) to the problem. In the example of the three boundary ports, if each dimension takes 20 values, the total required storage becomes  $20^{5+3+3} * 4\text{byte} = 745.1\text{Tb}$ . While the size of the VPT model matlab file is only tripled to be around 3Kb. Eventually, searching for a solution in the massive database will be more time consuming than solving the VPT model.

In summary, the VPT model is a convenient tool for applying control algorithm on flow control problem. To implement the VPT model in the contaminant control problem on the prototype system, the VPT model is tuned based on 3D pipe symmetric plane in the following section.

### 5.2.6 3D VPT Model for the Prototype

As previously discussed in Chapter II Section 2.1, the 3D pipe flow control problem is reduced down to 2D problem that relies on the pipe symmetric plane. Therefore, the VPT model was tuned based on the particle pathlines on the pipe symmetric plane. Due to the limitations of the prototype system, the velocity field in the prototype flow region cannot be accurately observed. Thus, in this study, the VPT model was first tuned using CFD simulated pathlines and velocity field. Then, the VPT model is adjusted according to prototype experimental observations.

#### Step 1: VPT model parameterization using 3D CFD simulated results

The same VPT model parameter tuning process presented in Section 5.2.2 was performed by matching the velocity field and particle trajectories for steady state CFD simulations. The CFD simulation flow conditions are  $Q_{in} = 40.54 \text{ cm}^3/\text{s}$  (equivalent to average velocity of 0.5 cm/s) and  $U(t) = 0.3$  constant. The pipe is 5.08 cm in



radius and the boundary port is 1.27 cm in diameter. The resultant VPT model parameters are  $\theta = 1.23$ ,  $a_1 = 0.001$ ,  $a_2 = 0.5$ ,  $a_3 = 0.037$ ,  $b_1 = 1.92$  and  $b_2 = 0.016$ . The 3D VPT model is expressed in Equation (5.30)

$$\begin{cases} \dot{x} &= 2\frac{Q_{in}}{\pi h^2}(1 - \frac{y^2}{h^2})(1 - \tau U(t)) \\ \dot{y} &= -U(t)(2h\frac{Q_{in}}{\pi h^2})\theta\frac{(y-h)^2}{(2h)^2}\frac{1}{w\sqrt{\pi}}\exp(-\frac{(x-x_p)^2}{b_1w^2+b_2(y+h)}) \end{cases}, \quad (5.30)$$

where  $\tau$  is the transition term defined in Equation (5.31)

$$\tau = 0.5 - \operatorname{erf}\left(-\frac{x - x_p}{a_1 + a_2(y + h)}\right) + 0.5\operatorname{erf}\left(-\frac{x - x_p}{a_3}\right). \quad (5.31)$$

Notice that the parabolic velocity profile for the 3D pipe is represented by  $2Q_{in}/(\pi h^2)$ , which is different from the 2D case where  $1.5Q_{in}/(2h)$  was used in the  $\dot{x}$  equation (refer to Equation (5.23)). In addition, the  $(2hQ_{in}/\pi h^2)$  term in the  $\dot{y}$  equation is the equivalent 2D flow rate for the symmetric plane.

The tuned 3D VPT model and the CFD tuning data set is compared in Fig 5.24 and Fig 5.25.

## Step 2: VPT model adjustment according to the prototype observation

It was previously presented in Chapter III that the contaminant color cloud gradually sinks downward due to the color cloud density variation. This downward motion accumulates error over time. To reduce this accumulating error, the upstream flow rate is set relatively high in the prototype to shorten the time period that this error accumulates. Therefore, the prototype experiments were conducted using  $Q_{in} = 7$  LPM ( $117 \text{ cm}^3/\text{s}$ ). At this flow rate, the Reynolds number is about 1500. The entrance length for laminar flow is about 0.06 times the Reynolds number *Post* (2011), which results in 90 meter. This entrance length is much longer than the prototype pipe system. In addition, uniform velocity profile is assumed after the high density sponge

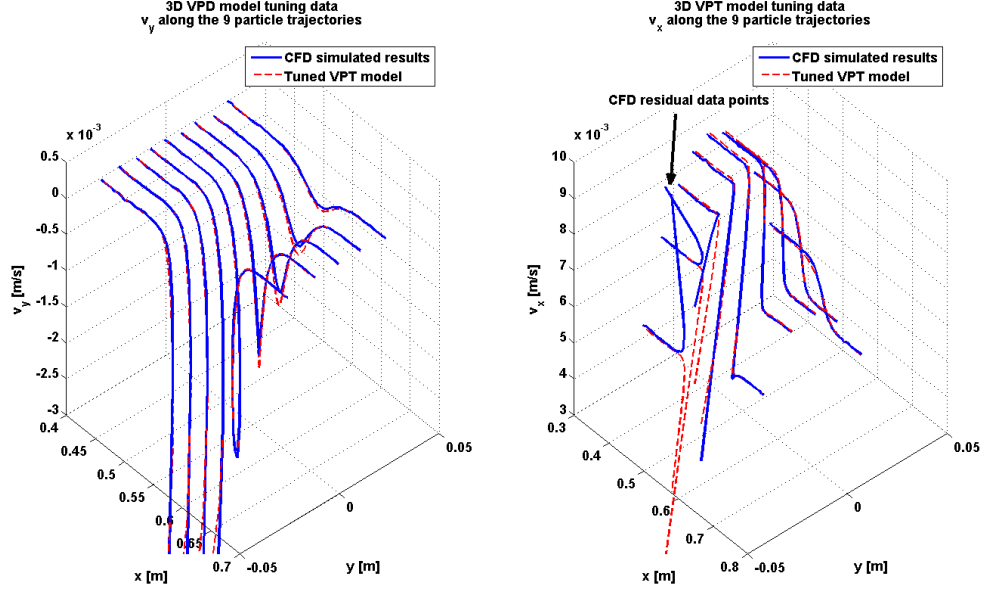


Figure 5.24: 3D VPT model tuning - velocity field. The graph shows the velocity field in the two coordinate on the 3D pipe symmetric plane. The  $y$  coordinate is the pipe radius direction and the  $x$  coordinate is the pipe length direction.

in the prototype pipe upstream. Therefore, instead of using parabolic velocity profile, the 3D VPT model  $\dot{x}$  equation was modified for uniform velocity profile as in Equation (5.32)

$$\begin{cases} \dot{x} = \frac{Q_{in}}{\pi h^2} (1 - \tau U(t)) \\ \dot{y} = -U(t) (2h \frac{Q_{in}}{\pi h^2}) \theta \frac{(y-h)^2}{(2h)^2} \frac{1}{w\sqrt{\pi}} \exp\left(-\frac{(x-x_p)^2}{b_1 w^2 + b_2 (y+h)}\right) \end{cases}, \quad (5.32)$$

in which equation, the upstream velocity profile implied in  $\dot{x}$  equation is represented as  $Q_{in}/(\pi h^2)$ , which is the mean velocity. In addition, it is also implied that the downstream flow also has uniform velocity profile. It should be noted that, the VPT model was originally proposed for low Reynolds number flow. Modelling error is expected when the VPT model is used to approximate particle trajectory in flow of higher Reynolds number. However, the VPT model is expected to capture the flow dynamics at least in a qualitative manner. Under this situation, the VPT model is

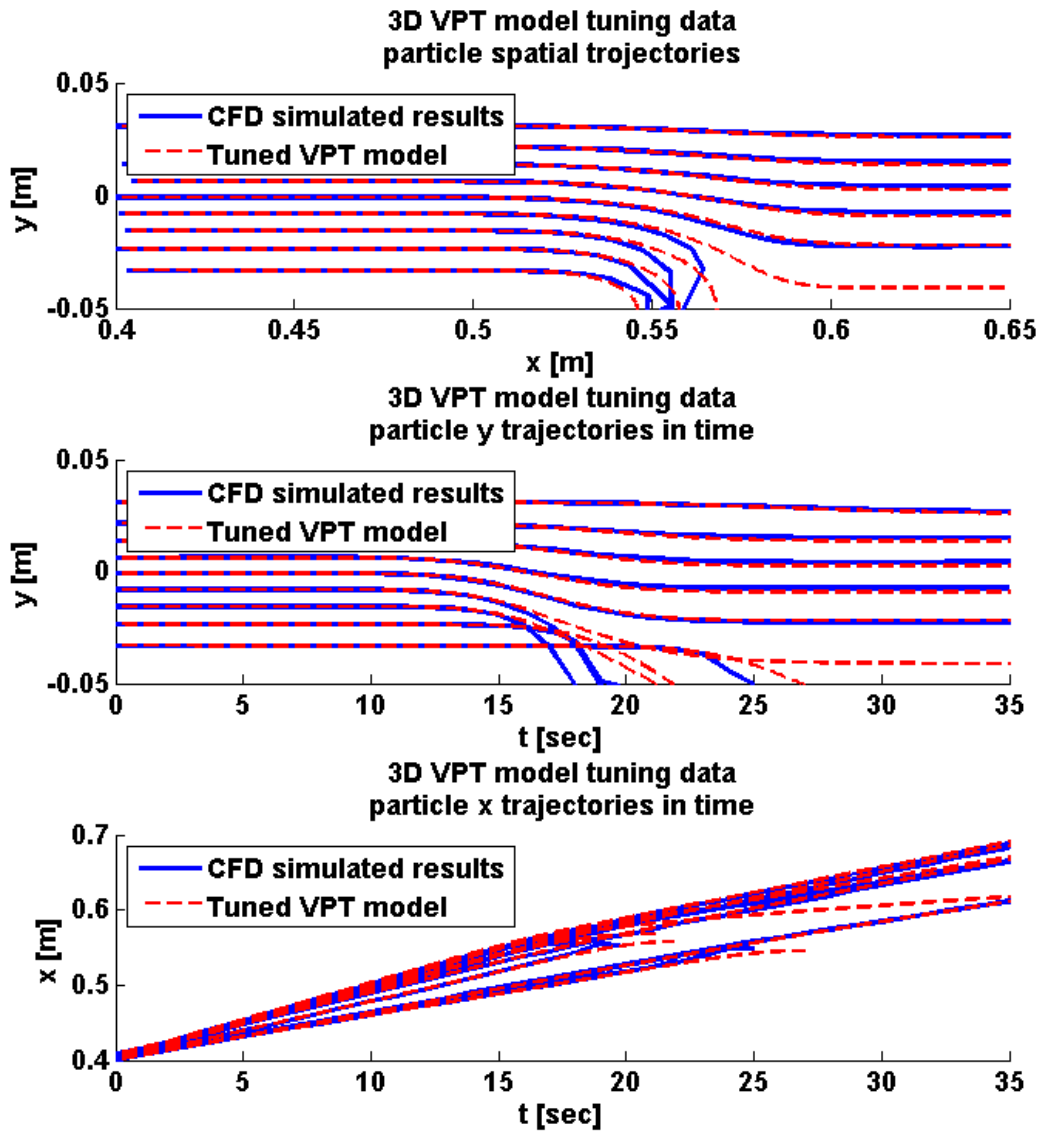
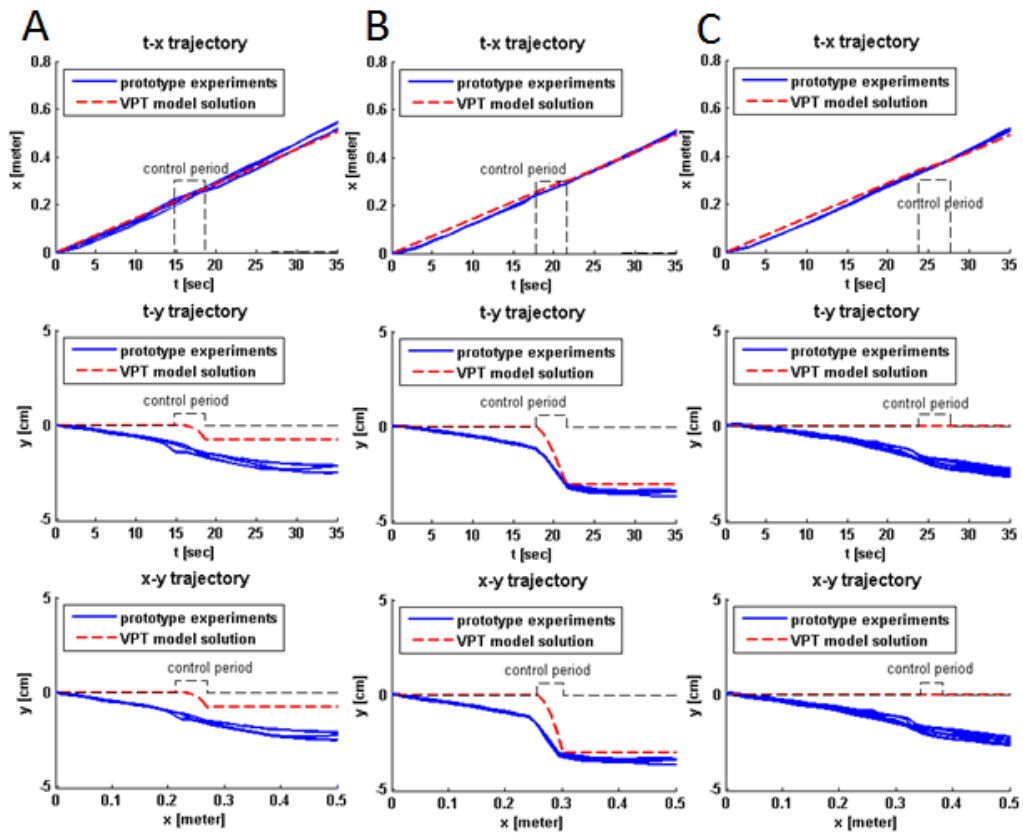


Figure 5.25: 3D VPT model tuning - particle trajectories.

still capable of being implemented in the real-time controller. Developing a prototype system for low Reynolds number flow is encouraged in future studies.

The resultant VPT model solutions are compared with prototype experimental results in Fig. 5.26. The bottom graph shows the contaminant cloud final locations given 10 different boundary control actions. The contaminant was released at the



3D VPT model vs Prototype experiments  
 $Q_{in} = 7\text{LPM}$ ,  $x_p = 0.2818\text{m}$ ,  $t_{off} - t_{on} = 3.82\text{sec}$

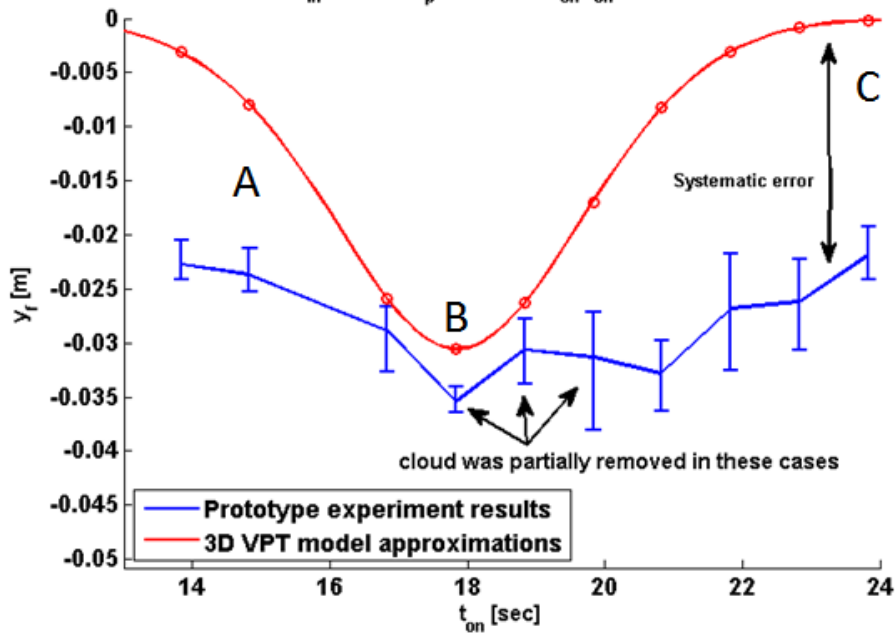


Figure 5.26: 3D VPT model vs Prototype experiments.

center of the pipe at  $y_{ini} = 0$  cm. The boundary control signals used in these examples were square wave pulses defined by  $U = U_o$  for  $t \in [t_{on}, t_{off}]$ . The  $U_o$  defines the pulse magnitude and the  $t_{on}$  and  $t_{off}$  define the square wave pulse width. In the 10 example cases shown in Fig. 5.26, the boundary control pulse magnitude was  $U_o = 0.3$  and the pulse width was  $t_{off} - t_{on} = 3.82$  sec. The control pulses were applied at different times as indicated by the  $x$  axis. The top three graphs illustrate in detail three of the cases.

The systematic error was caused by the color cloud density variation. This systematic error can be observed in the top  $y - time$  or  $y - x$  plot that, the color cloud sank downward without the boundary drawing action. In case A and C, the boundary control was applied too early and too late respectively. Therefore, the boundary control did not show significant influence on the particle  $y$  location, compared to the cloud self sinking motion. While in case B when the boundary control is applied at the right time, the boundary control effect is clearly observed. Despite of the systematic error, the 3D VPT model qualitatively approximates the trend between the boundary control timing and the control outcome. The significance of the VPT model will be presented with more detail in Chapter VI, where the optimal controller and the feedback controller uses the VPT model to determine the effective feedback gain.

### 5.3 Summary

Two mathematical models for fast approximating particle propagations in laminar pipe flow are presented in this chapter. The Algebraic Particle Tracing (APT) model is used for steady pipe flow when the boundary control is constant in time. The APT model is a simple third order polynomial relationship that assembles the flow control problem variables: contaminant initial location, the boundary control magnitude and the resultant contaminant final location. Given two of the variables, the third one can be solved using common polynomial solvers. Therefore, the APT

model can be used for particle forward prediction or source inversion problem. In addition, the development of the APT model introduced a dimensionless boundary control quantity  $U = Q_p/Q_{in}$  that characterizes the similarity among flows of different Reynolds number. This dimensionless quantity simplifies the controller design presented in Chapter VI. The Velocity field Particle Tracking (VPT) model, on the other hand, is used for transient case when the boundary port flow is time variant. The VPT model is a two states ODE model, which solves for the contaminant trajectory. This mathematical model not only solves for the contaminant final location like the APT model do, but also solves the contaminant location in time domain. Therefore, the VPT model is suitable for the proposed optimal control problem, in which the boundary control magnitude and timing are to be optimized. Furthermore, the VPT model implies a superposition property for low Reynolds number flow, which enables simple modelling for system of higher complexity. For example, a flow system of three boundary ports can be modelled by superposing three VPT model equations together. Although not further investigated in this thesis, both the APT and the VPT model can be potentially used for source inversion problem, which is reviewed in Section 7.2.4.

## CHAPTER VI

# Controller Implementations Using the Prototype System and CFD Method

This chapter presents the control system implementations using both the prototype system and the finite element method. The overall control architecture, as is illustrated in Fig. 6.1, is composed of three controllers, the optimal controller, the feedback controller and the model adaptation controller. In this chapter, the main discussion is divided into three sections, one for each controller.

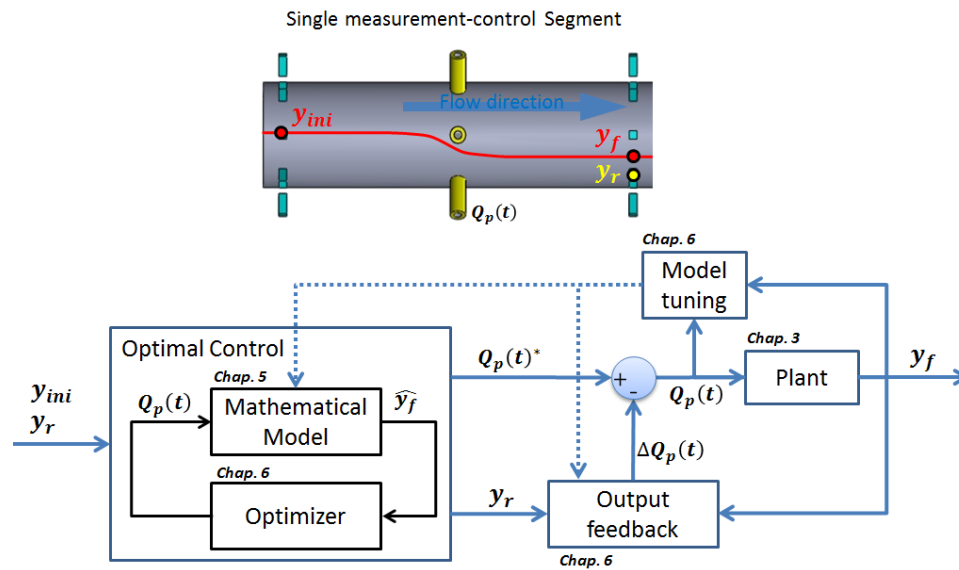


Figure 6.1: The overall control architecture.

Discussed in Section 6.1, the optimal controller is a feedforward controller which employs a reference mathematical model to optimize the boundary control strategy by minimizing the amount of removed fluid. In the cases such as the toxic chemical elimination problem, when a fast response is required and a control strategy need to be provided by the optimal controller immediately upon the toxin observation, the optimizer considers a steady flow control problem. In this case, the static APT model is employed to first determine whether the control objective (toxic chemical cloud elimination) can be achieved using the provided boundary ports. Then the magnitude of the static boundary control, which is constant in time, is optimized using the APT model as the reference. This optimized constant control strategy is improved from the worst case scenario control solution, which is removing all the fluid. In other cases when more computational time allows, the dynamic VPT model can be employed to find an even better optimized control strategy, in which the waveform of the boundary control is optimized. However, iterating the VPT model in the optimizer is more time consuming than the APT model based optimization. The feasibility of implementing the VPT model based optimal control in real-time is validated via both prototype experiments and finite element analysis method. The complete loop of contaminant measurement, control strategy optimization and boundary control deployment is demonstrated.

In Section 6.2, the model reference output feedback controller is demonstrated via prototype experiments. The feedback controller reacts to the measured error between the actual system output and the desired value, independently from the physical reasons that cause the error. The feedback controller, following the optimal controller, adjusts the optimized boundary control strategy to drive the system output towards the desired value. The effectiveness of the VPT model based feedback controller was demonstrated by prototype experiments, in which the prototype output converged to the desired value within 3 iterations.



Section 6.3 presents the concept for VPT model parameter on-line adaptation. Tuning the mathematical model towards higher fidelity improves the model referenced controller performance. For example, employing a more accurate VPT model, the optimal controller will find a boundary control strategy that is closer to the true optimum of the actual system. In this study, the model adaptation algorithm was validated using the prototype experiments and 3 out of the 6 VPT model parameters were tuned based on actual measurements. The successful prototype implementations of the controller emphasizes that the mathematical models developed in this study (Chapter V) have established a bridge to integrate classical control algorithms with real-time flow control problems.

## 6.1 The Optimal Controller

The worst case scenario prototype experiment shown in Chapter III illustrated a feasible contaminant elimination solution, which is removing all the fluid from the pipe. However, when the contaminant is not fully spread out in the pipe or only few contaminant particles are present, such extreme control action wastes uncontaminated fluid and consumes unnecessary boundary control effort. The optimal control problem addresses such consideration.

The optimization problem was previously formulated in Section 2.2.1 Equation (2.2). The optimization aims to find the best boundary control strategy which achieves the control objective. For the single measurement-control pipe segment case, the control objective is to move the contaminant particle to the desired location measured by the downstream sensor. In this study, the optimality of the boundary control strategy is quantified as the accumulated fluid exchange through the boundary port. In our case, the control strategy that removes less fluid is a better control strategy. This

optimization problem is reformulated in Equation (6.1)

$$\begin{aligned}
& \min_{U(t)} \int |U(t)| dt \\
& \text{s.t. } \hat{y}_f = M(y_{ini}, Q_{in}, U(t)) \\
& \hat{y}_f \leq y_r \tag{6.1} \\
& U(t) \leq U_{max} \quad \forall t \geq 0 \\
& U(t) \geq U_{min} \quad \forall t \geq 0.
\end{aligned}$$

In this formulation, the dimensionless boundary control representation  $U(t) = Q_p(t)/Q_{in}$  replaces the  $Q_p$  in the original formulation Equation (2.2). The  $M$  represents the mathematical model that is employed in the optimal controller. The  $\hat{y}_f$  and the  $y_r$  are the mathematical model approximated system output and the control objective respectively. The  $U_{max}$  and  $U_{min}$  are the physical limitations of the boundary port flow based on the pump characteristics. The upstream flow rate  $Q_{in}$  is constant in time, or its variability is slower than the control action, so that the upstream flow can be assumed as constant during the control duration. The dimensionless representation of the boundary control action  $U(t)$  is previously introduced in Chapter V by the APT model.

Either the static APT model or the dynamic VPT model can be employed in the minimization problem in Equation (6.1). As discussed in Chapter V, the APT model only considers the magnitude of the boundary control and assumes that the boundary control action is constant in time. Therefore, the optimization based on the APT model aims to find an improved boundary control strategy better than removing all the fluid, which is the control strategy demonstrated in the worst case scenario. On the other hand, the optimization based on the VPT model is more advanced that the waveform of the boundary control, which is a function of time, can also be optimized. For example, when the contaminant is far away upstream from the boundary port

early in time, the applied boundary control effort is wasted and has no influence on the contaminant location. Under this situation, the more advanced optimization based on the VPT model helps to determine not only the control magnitude but also when the boundary control should be applied. However, the computational time cost associated with solving the VPT model is higher than using the APT model. Therefore, the trade-off between computational time and control strategy optimality should be discussed depending on the flow velocity. If the problem is associated with relatively fast flow rate when the computational time is very limited, the APT model should be employed. While in the case when the flow rate is small and plenty of time is available before the control action need to be taken, the VPT model can be applied.

The steady flow scenario is discussed in Section 6.1.1, where the minimization problem is solved by employing the APT model. Due to the advantage of the APT mode, the optimization solution can be solved almost instantly, making the APT model a good candidate for the situation when fast reaction is preferred upon contaminant measurement.

Section 6.1.2 presents the VPT model employed optimization. It is proved that the VPT based minimization problem can be solved within the required time frame for real-time control application. Both CFD simulations and prototype experimental results demonstrates the accuracy and the efficiency of the optimal controller.

### **6.1.1 Optimal Control Using the APT Model**

The APT model based optimal control aims to find a boundary control strategy better than removing all the fluid. Because the boundary control is assumed constant in time, the minimization problem can be reformatted as Equation (6.2) by dropping

the time variable

$$\begin{aligned}
& \min_U \quad |U| \\
& \text{s.t.} \quad \hat{y}_f = APT(y_{ini}, U) \quad -h \leq \hat{y}_f \leq h \\
& \quad \quad \hat{y}_f \leq y_r \\
& \quad \quad U \leq U_{max} \\
& \quad \quad U \geq U_{min},
\end{aligned} \tag{6.2}$$

in which, the  $\hat{y}_f$  is computed by solving the cube root using the polynomial relationship described by the APT model that is rephrased in Equation (6.3)

$$\frac{y_{ini}^3}{4h^3} - \frac{3y_{ini}}{4h} + \frac{1}{2} = (1 - U) \left( \frac{\hat{y}_f^3}{4h^3} - \frac{3\hat{y}_f}{4h} + \frac{1}{2} \right). \tag{6.3}$$

This relationship is previously defined in Section 5.1 Equation (5.7), in which section, it is shown that only one distinct and feasible solution for  $\hat{y}_f$  exists within the pipe geometry range of  $[-h, h]$ .

An iterative optimizer can be applied to solve the single variable minimization problem defined in Equation (6.2). However, the following analysis of the APT model shows that no iterative optimization is required. The solution of the minimization problem can be solved directly using the APT model.

It is observed in Equation (6.3) that, given a  $y_{ini}$ , the  $U$  is monotonically and inversely related with the  $\hat{y}_f$  within the  $[-h, h]$  pipe geometry range. This monotonicity is visualized in Fig. 6.2. For any particle initial location  $y_{ini}$ , the  $\hat{y}_f$  decreases as the  $U$  increases. Therefore, the solution to the minimization problem in Equation (6.2) lies on the constraints' equality edge:  $\hat{y}_f = y_r$ ,  $U = U_{max}$  or  $U = U_{min}$ . The  $U = U_{max}$  is active when the boundary port pump is not strong enough to draw the particle to the target final location  $y_r$ ; the  $U = U_{min}$  is active when the boundary port injecting power is not sufficient to push the particle away (The boundary injection is not used in the contaminant elimination problem discussed in this dissertation); the  $\hat{y}_f = y_r$  is

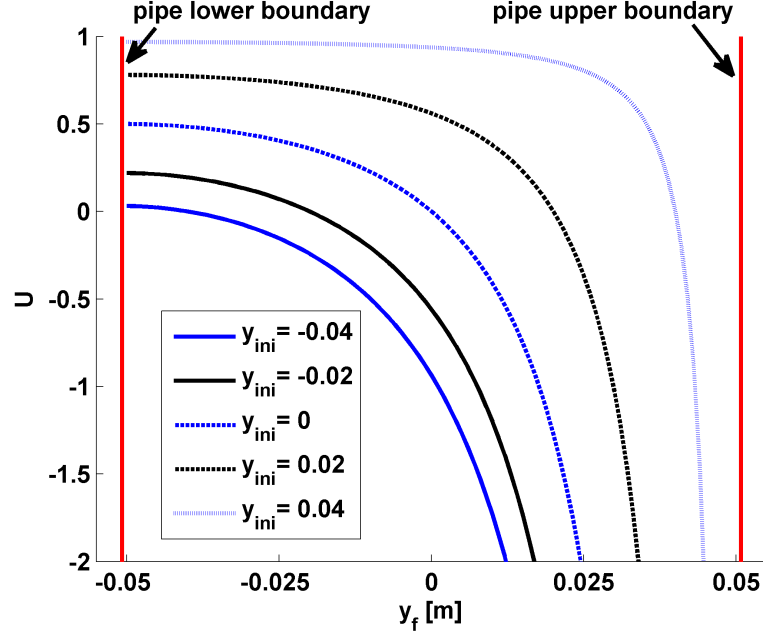


Figure 6.2: Monotonic relationship between the control magnitude  $U$  and the output  $y_f$  described by the APT model. Given a  $y_{ini}$ , the  $y_f$  decreases as  $U$  increases within the pipe geometry range of  $[-h, h]$ .

active when the right amount boundary control can be applied to achieve the control objective.

Therefore, instead of solving the minimization problem in Equation (6.2) iteratively, the optimization solution can be determined in the following simpler way. The APT model (Equation (6.3)) is reshaped into Equation (6.4) by replacing the  $\hat{y}_f$  with the  $y_r$

$$U_r = 1 - \left( \frac{y_{ini}^3}{4h^3} - \frac{3y_{ini}}{4h} + \frac{1}{2} \right) / \left( \frac{y_r^3}{4h^3} - \frac{3y_r}{4h} + \frac{1}{2} \right). \quad (6.4)$$

This equation computes the required boundary control magnitude  $U_r$  to achieve the control objective  $y_r$ . If the resultant  $U_r$  is larger than  $U_{max}$ , the solution of the optimization is  $U = U_{max}$ . This result indicates that the pump is not strong enough and more measurement-control segments are required to achieve the optimization set point  $y_r$ . The same principle follows for the  $U_r < U_{min}$  case. The advantage of above method is obvious that the required optimal control strength can be directly solved. If

the required control strength exceeds the boundary pump physical power, the control system can also be notified. In addition, Equation (6.4) is a simple algebraic equation that can be evaluated within millisecond, enabling fast response upon contaminant observation.

In summary, an one-to-one mapping between the contaminant particle location and the optimal boundary control strategy is created using the APT model. No iterative optimization is required in the static flow control problem. The entire optimization process is simply a static algebraic equation evaluation. To implement the APT model in an actual system, a 1D lookup table can be created. As a result, the control strategy can be almost instantly determined. Because the APT model is developed only for the 2D geometry, the existing prototype system is not capable of validating the APT model based optimal controller. The 2D finite element analysis results presented in Section 5.1 are sufficient to show the accuracy and the time efficiency of the APT model, and thus to illustrate the potential effectiveness of the APT model based optimal controller.

### **6.1.2 Optimal Control Using the VPT Model**

Using the APT model, the optimal boundary control magnitude can be determined almost instantly. However, a constant boundary control action can never be the best control strategy that takes into account the waveform of the boundary control action. A better optimized boundary control strategy can be solved by employing the dynamic VPT model, which requires more computing time to solve.

As is defined in Equation (6.1), the optimization of the time dependent variable  $U(t)$  is an infinite time horizon problem, which is usually very complex and time consuming to solve. It is even difficult to mathematical define the problem itself. In order to solve this optimization problem within the allowed time frame for real-time applications, the waveform of  $U(t)$  should be represented by finite number of

deterministic variables. Then, the minimization can be performed to optimize these variables.

Discussed in Section 6.1.2.1, one feasible wave form of the  $U(t)$  is a square wave, which requires three variables to define. It is illustrated via finite element analysis that an optimized boundary control strategy is indeed in a squared wave format. Then, the iterative optimization algorithm based on the square wave shaped  $U(t)$  is presented in Section 6.1.2.2. The optimal controller is implemented in real-time and is validated using both prototype implementations and finite element simulations.

### 6.1.2.1 Waveform of $U(t)$

An effective selection of the deterministic variables for defining the wave form of  $U(t)$  lowers the associated optimization computational time cost. Based on the following logical reasoning and the observations of velocity field induced by the boundary flow, the boundary control  $U(t)$  can be expressed as a square wave pulse, which can be expressed using only three independent variables.

It is an obvious fact that, the boundary control is meaningless if it is applied when the particle is far away from the boundary port. Therefore, the boundary control should be applied when the particle is as close to the port as possible. This reasoning can also be derived from the VPT model equation. The velocity field induced by the boundary flow is approximated by the right hand side of the VPT model, which is rephrased in Equation (6.5)

$$\begin{cases} \dot{x} &= \frac{3}{2} \frac{Q_{in}}{2h} \left(1 - \frac{y^2}{h^2}\right) (1 - \tau U(t)) \\ \dot{y} &= -U(t) Q_{in} \theta \frac{(y-h)^2}{(2h)^2} \frac{1}{w\sqrt{\pi}} \exp\left(-\frac{(x-x_p)^2}{b_1 w^2 + b_2 (y+h)}\right) \end{cases}, \quad (6.5)$$

where  $\tau$  is the transition term rephrased in Equation (6.6)

$$\tau = 0.5 - \operatorname{erf}\left(-\frac{x - x_p}{a_1 + a_2(y + h)}\right) + 0.5\operatorname{erf}\left(-\frac{x - x_p}{a_3}\right). \quad (6.6)$$

By examining the exponential term on the right hand side of the  $y$  equation,  $\exp(-(x-x_p)^2/(b_1w^2+b_2(y+h)))$ , it is observed that the particle  $y$  velocity magnitude has an inverse relationship with the  $x$ -distance away from the boundary port. In other words, the contaminant particle moves downward faster when the particle  $x$  location is closer to the boundary port location  $x_p$  (The boundary port is located at the bottom of the pipe).

Therefore, it is an obvious rational conclusion that the optimized boundary control should be formulated as a square wave pulse in Equation (6.7)

$$U(t) = \begin{cases} U_o = U_{max}, & t_{on} \leq t \leq t_{off} \\ 0, & t < t_{on} \quad \text{or} \quad t > t_{off} \end{cases}, \quad (6.7)$$

where the  $t_{on}$  and the  $t_{off}$  describe the boundary control turn on and turn off time, which define the width of the square wave. The  $U_o$  represents the magnitude of the boundary control applied between  $t_{on}$  and  $t_{off}$ . The boundary pump operates at its maximum allowed strength during the control period. Operating the boundary pump at a low strength leads to longer control duration, which implies that a portion of the boundary control effort is applied too early or too late in time. As a result, this portion of the control effort contributes less in moving the contaminant downward.

Shown in Fig. 6.3, a parametric study via 2D finite element CFD simulations was conducted to illustrate above logic reasoning of using square wave shaped  $U(t)$ . The parametric study kept the total amount of removed fluid ( $\int |U(t)| dt$ ) the same in all cases and investigated the outputs ( $y_f$ ) resultant from different combinations of  $t_{on}$ ,  $t_{off}$  and  $U_o$ . In this study, the volume of removed fluid represents the boundary control effort, which is later used as the cost function for the optimal controller. In the CFD simulation setup, the pipe is  $2h = 10$  cm in height with boundary port width of  $w = 2.5$  cm. Two particle initial locations are used for demonstration purposes,  $y_{ini} = 0$  cm and  $y_{ini} = -1$  cm. The upstream inlet flow rate is  $Q_{in} = 6.667$  cm<sup>2</sup>/s. The



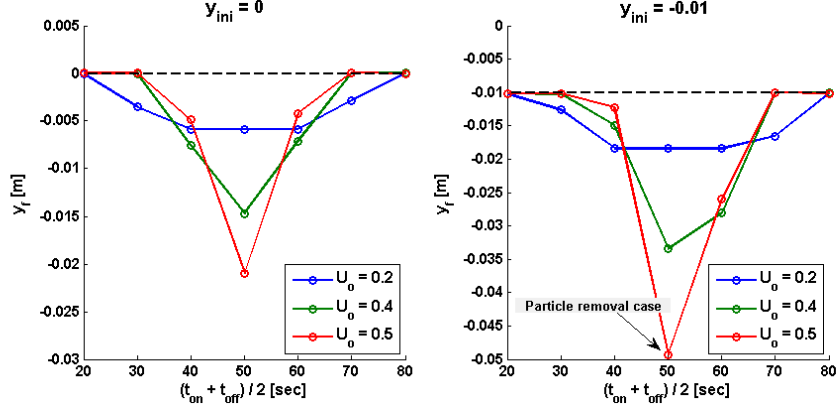


Figure 6.3: 2D CFD parametric study showing that higher control strength for shorter period of time is more efficient than spreading out the control effort in space and time. In each data point case, same amount of 53.34 cm<sup>2</sup> fluid is removed. Higher control strength for shorter period of time produces larger particle displacement.

boundary port is positioned at  $x_p = 50$  cm from the assumed upstream sensor location (which is the finite element simulation flow inlet boundary). In each parametric study case, the boundary control removed 53.34 cm<sup>2</sup> fluid. The parametric study boundary control magnitude ( $U_o$ ) and timing ( $t_{on}$  and  $t_{off}$ ) are listed in Table. 6.1, in which the  $(t_{on} + t_{off})/2$  is used as an indication of when the boundary control is applied. The utilization of  $(t_{on} + t_{off})/2$  also enhances the clarity in the graphic illustrations. The CFD simulated results ( $y_f$ s) are plotted in Fig. 6.3.

Table 6.1: Parametric study  $U(t)$  parameters

$\frac{t_{on}+t_{off}}{2}$	$U_o = 0.2$			$U_o = 0.4$			$U_o = 0.5$		
	Case	$t_{on}$	$t_{off}$	Case	$t_{on}$	$t_{off}$	Case	$t_{on}$	$t_{off}$
20	1	0	40	8	10	30	15	12	28
30	2	10	50	9	20	40	16	22	38
40	3	20	60	10	30	50	17	32	48
50	4	30	70	11	40	60	18	42	58
60	5	40	80	12	50	70	19	52	68
70	6	50	90	13	60	80	20	62	78
80	7	60	100	14	70	90	21	72	88

The result of this parametric study shows that, by removing same amount of

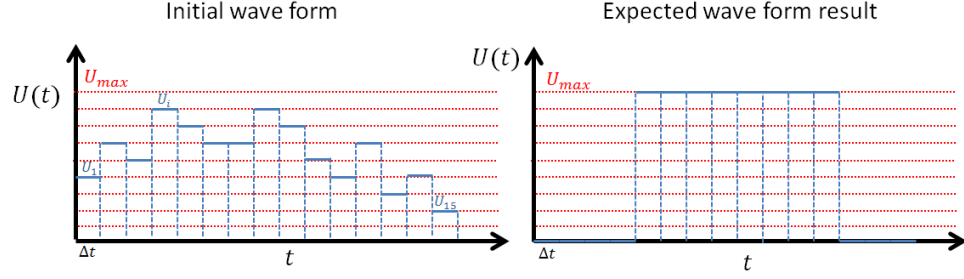


Figure 6.4: Proposed method for validating the optimality of using square wave shaped boundary control  $U(t)$  against other wave forms.

fluid, the control strategy using higher control strength for shorter period of time produces greatest influence on the particle  $y$  location. This result supports the earlier rational reasoning of using square wave shaped  $U(t)$ . In this thesis, the validation of the optimality of using square wave is not discussed against other wave forms. However, a method to validate the optimality of the square wave pulse is proposed in Fig. 6.4. The boundary control is discretized both in its magnitude and in time. In the example shown in Fig. 6.4, the control magnitude is discretized in 10 levels and the time span of interested is divided into 15 segments. An optimization can be conducted starting from an arbitrary assigned boundary control strategy as shown in the left graph in Fig. 6.4. The optimization keeps the total removed fluid the same ( $\sum_1^{15} \Delta t U_i$ ) and maximize the particle  $y$  displacement. If the control strategy that produces the maximum particle  $y$  displacement is in the shape shown in the right graph in Fig. 6.4, the square wave representation of the boundary control strategy is proved.

### 6.1.2.2 The Optimization Algorithm

Using the three variables that define the square wave pulse, the minimization problem previously defined in Equation (6.1) can be modified into the simpler mini-

mization with Equation (6.8)

$$\begin{aligned}
& \min_{U_o, t_{on}, t_{off}} |U_o| (t_{off} - t_{on}) \\
& \text{s.t.} \quad \hat{y}_f = VPT(y_{ini}, Q_{in}, U(t)) \\
& \quad \hat{y}_f \leq y_r \\
& \quad U_o \leq U_{max} \\
& \quad U_o \geq U_{min} \\
& \quad t_{on} \leq t_{off},
\end{aligned} \tag{6.8}$$

where  $U_o$  is the magnitude of the pulse,  $t_{on}$  and  $t_{off}$  are the on and off time of the pulse. The three variables are to be chosen so that the total "mass" during the pulse is minimized. In fact, as stated previously, the boundary pump should be operating at its maximum value, which makes it unnecessary to optimize the variable  $U_o$ . However, in the case when the slow dynamic response of the pump and/or the system time lag are considered, the resolution of the boundary control timing may limit the effectiveness of adjusting  $t_{on}$  and  $t_{off}$ . Thus, the  $U_o$  is kept as a variable to be optimized for generality.

The optimization algorithm logic for solving the minimization problem defined in Equation (6.8) is illustrated in Fig. 6.5. In general, the optimization starts with the maximum possible boundary control effort, for example, operating the boundary port at its maximum allowed value for all the time. Then, by employ the VPT model, the boundary control strategy is adjusted and the total control effort is reduced iteration by iteration until the final solution is found.

The optimization starts with the initial searching point of  $t_{on}^1$ ,  $t_{off}^1$  and  $U_o^1$ , which should be an extreme control action, for example,  $t_{on}^1 = 0\text{sec}$ ,  $t_{off}^1 \gg 1.5Q_{in}/h/x_p$  and  $U_o = U_{max}$ . The selection of  $t_{off}^1 \gg 1.5Q_{in}/h/x_p$  ensures that the boundary control action is turned off when the particle is in the very downstream region of the pipe.

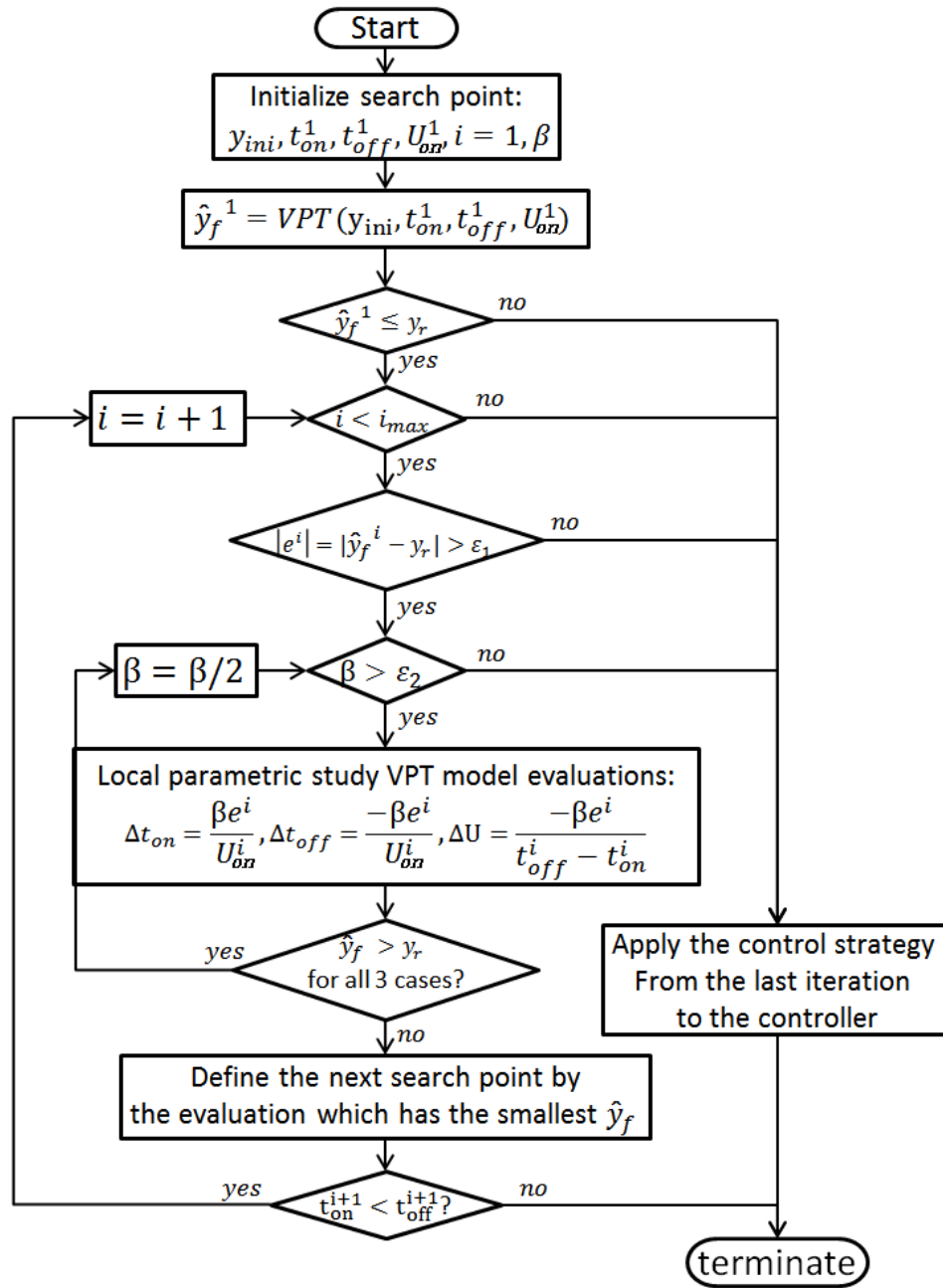


Figure 6.5: The optimization iteration logic for solving the minimization problem. The minimization problem is defined in Equation (6.8).

The optimization step size is defined by  $\beta$ , which is the reduction of the total control effort in each iteration. At the first iteration, the VPT model is evaluated outside the loop. If  $\hat{y}_f^1 > y_r$ , which indicates that the control objective cannot be achieved, the optimization process terminates. In such case, more measurement-control segments are required to achieve the control objective  $y_r$ . Continuing down the logic, if the maximum iteration steps  $i_{max}$  is not reached, the error  $e^i$  between the VPT model prediction  $\hat{y}_f^i$  and the objective  $y_r$  is computed. The algorithm uses the error threshold  $\epsilon_1$  to decide if the optimization process continues. The following sub-loop adjusts the searching step size  $\beta$  according to a local parametric study around the current searching point. Three VPT model evaluations are performed by perturbing the three control variables individually in each evaluation. If the VPT model predictions in all three evaluations violate the  $\hat{y}_f \leq y_r$  constraint, the searching step size  $\beta$  is reduced until at least one evaluation satisfies the constraint. The algorithm then uses the second threshold  $\epsilon_2$  to decide if the search step size is small enough to terminate the optimization process. If the optimization continues, the next searching point  $\hat{y}_f^{i+1}$ ,  $t_{on}^{i+1}$ ,  $t_{off}^{i+1}$  and  $U_o^{i+1}$  are defined by one of the three sub-loop evaluations which has the smallest  $\hat{y}_f$ . In addition, before the next iteration begins, the relationship between  $t_{on}^{i+1}$  and  $t_{off}^{i+1}$  determines if no boundary control is required. Satisfaction of the  $t_{on}^{i+1} \geq t_{off}^{i+1}$  condition indicates no boundary action is needed. This condition rarely occurs, unless a non-reasonable  $y_r$  is assigned. For example, if the particle initial location is already at the objective level,  $y_{ini} = y_r$ , no control action is needed. The optimization algorithm was programmed in Matlab 2009 and is documented in Appendix C.

### 6.1.2.3 Optimal Control Implementation via CFD and Prototype

This section presents the actual computing performance and the implementation of the optimal controller via both prototype experiments and finite element CFD sim-

ulations. As is previously presented in Chapter III, the prototype computing station communicates with the real-time system. The computing station is responsible for processing the image data obtained by the optical sensor and for solving the minimization problem defined in Equation (6.8). The real-time system is responsible for delivering the defined boundary control strategy by translating the control strategy into the water pump control command. In order to achieve real-time control, the requirement for the optimization computing speed is illustrated in Fig 6.6. At the time instance when the contaminant particle is observed, the computing station sends a command to synchronize the system time as  $t = 0$  with the real-time system (Appendix A). The computing station also starts solving the minimization problem at this moment. At the time ( $t = t_o$ ) when the VPT model based optimization process finishes, the resultant boundary control strategy (represented by  $U_o$ ,  $t_{on}$  and  $t_{off}$ ) is sent to the real-time system for actual implementation. The optimized boundary control strategy can be correctly implemented only if  $t_o \leq t_{on}$ . In this study, this timing requirement is successfully achieved and is demonstrated via both CFD co-simulations and with prototype hardware experiments.

### Optimal control implementation via 2D CFD simulations:

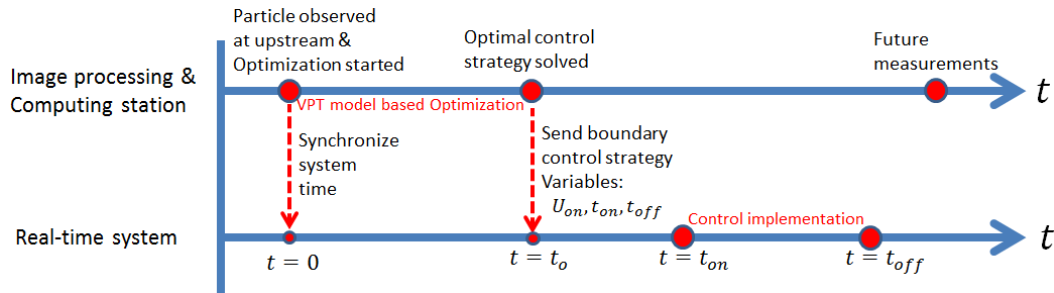


Figure 6.6: The optimization computing speed requirement for implementing the optimal controller in real-time. The time instance when the optimal control strategy is solved ( $t_o$ ) should be earlier than the solved  $t_{on}$  in the optimal control strategy.

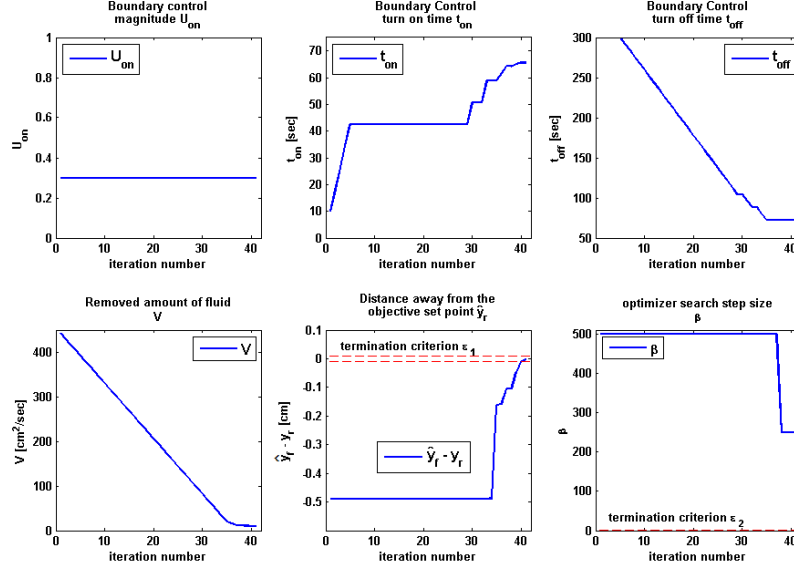


Figure 6.7: The optimization iterations and results based on the 2D VPT model. The control considered moving a particle from  $y_{ini} = 0\text{cm}$  to  $y_r = -1\text{cm}$  using the optimization algorithm illustrated in Fig. 6.5. The total time consumption to solve for this result is 20.07sec.

In the following example, the 2D VPT model is employed in the optimal controller. The performance optimal controller is demonstrated via 2D CFD simulations. The upstream flow rate is  $Q_{in} = 5.08 \text{ cm}^2$  and the pipe parameters are  $x_p = 51.435 \text{ cm}$ ,  $h = 5.08 \text{ cm}$  and  $w = 1.27 \text{ cm}$ . The control objective is to move a particle from the middle of the pipe  $y_{ini} = 0 \text{ cm}$  down to  $y_r = -1 \text{ cm}$ . The boundary control magnitude is limited to  $U_{max} = 0.3$ . The optimization iteration starts with  $U_o^1 = U_{max} = 0.3$ ,  $t_{on}^1 = 10 \text{ sec}$  and  $t_{off}^1 = 300 \text{ sec}$ . This initial search point ensures that the boundary flow is turned on when the particle is in the upstream region and is turned off when the particle is in the downstream.

By utilizing the optimization algorithm defined in Section 6.1.2.2 Fig. 6.5, the minimization problem took 20.07 sec to be solved on a laptop with Intel i5 M540 2.53Ghz CPU and 1G RAM. The optimization termination margins are  $|\hat{y}_f - y_r| \leq \epsilon_1 = 0.01 \text{ cm}$  and  $\beta \leq 5$ . The resultant optimized control strategy is defined by

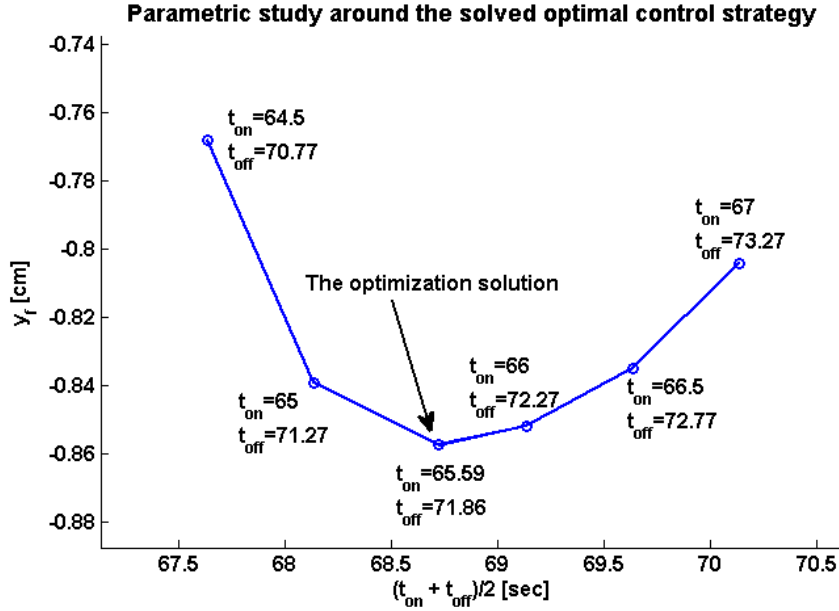


Figure 6.8: 2D CFD validation of the optimal control strategy solved by the VPT model based optimization. The solution suggested by the VPT model based optimization is very close to the true optimum implied by the CFD simulation results.

$t_{on} = 65.59$  sec,  $t_{off} = 71.86$  sec and  $U_o = U_{max} = 0.3$ . The optimization iteration process is plotted in Fig. 6.7. The control magnitude stays at the  $U_{max}$  level. The total fluid removed by the boundary port ( $V$ ) reduces over iterations, illustrating that the optimization algorithm is implemented correctly. The VPT model predicted output  $\hat{y}_f$  converges to the optimization set point  $y_r$  at the end of the optimization and the termination criterion  $\epsilon_1$  is satisfied. The matlab program code used for this example is documented in Appendix C.

To validate the optimality of the solved boundary control strategy, a local parametric study was conducted around the solution achieved by the optimal controller. Five other control strategies were implemented and compared with the achieved solution in Fig. 6.8. These five control strategies were created by shifting the optimized control strategy in time, without changing the total amount of removed fluid. It should be noted that, due to the CFD simulation time step resolution used in this



parametric study (0.5 sec), the actual boundary control timing  $t_{on}$  and  $t_{off}$  applied in the CFD simulation are different from the specified value. This time step resolution causes the 0.04 cm error between the objective value  $\hat{y}_f = y_r = -1$  cm and the CFD simulated result  $y_f = -0.86$  cm for the optimized boundary control strategy. Nonetheless, this parametric study result validates the optimality of the proposed control strategy within the CFD simulation resolution. In addition, in this example, the optimization time cost is 20.07 sec, which is less than the solved  $t_{on} = 65.59$ sec. Therefore, the requirement for real-time implementation (Fig. 6.6) is satisfied and it is demonstrated that the proposed optimal control strategy can be implemented in a real-time manner at least for these slow velocities and fluid conditions.

### **Optimal control implementation on the prototype:**

In the following prototype experimental implementation, the 3D VPT model presented in Section 5.2.6 was employed in the optimal control algorithm. The optimal controller was successfully implemented in real-time on the prototype. The control objective is to move a particle from the middle of the pipe  $y_{ini} = 0$  cm down to  $y_r = -3$  cm. (The geometry of the prototype pipe system can be found in Chapter III.) The actual measured prototype upstream flow rate is averaged at  $Q_{in} = 116.67$  cm<sup>3</sup>/sec (7 LPM). The boundary control strength is limited to  $U_{max} = 0.3$ . The optimization iteration starts with  $U_o^1 = U_{max} = 0.3$ ,  $t_{on}^1 = 2$  sec and  $t_{off}^1 = 50$  sec. The prototype boundary port was centered at 28.18 cm in the camera view by analyzing the camera image shown in Fig. 6.9, in which the boundary port mounting structure (5.08 cm wide) was used as the pixel-to-length reference scale.

The minimization problem took 9.80 sec to finish on the HP Z420 workstation with Intel Xeon(R) E5-1620 3.60Ghz and 48G RAM. This computational time cost was lower than the CFD implementation example shown previously because fewer iteration steps were needed to arrive at  $\hat{y}_f = y_r = -3$ cm than  $\hat{y}_f = y_r = -1$ cm. When

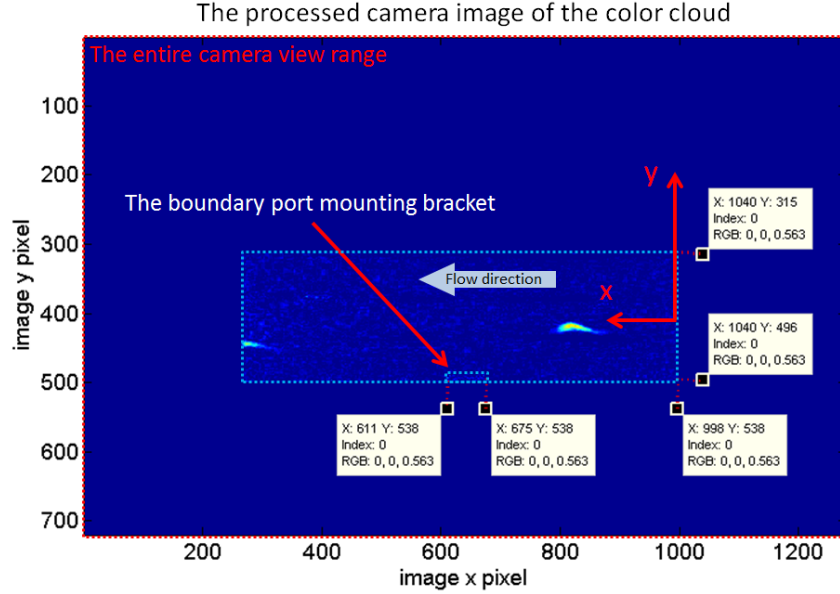


Figure 6.9: Reading the boundary port location from the camera image. The boundary port mounting structure is 5.08 cm wide and is used as the pixel-length reference scale.

the control objective  $y_r$  is closer to the bottom of the pipe, the solution is closer to the initial searching point. In addition, the higher computing power shortened the time cost in evaluating the VPT model. The optimization process is shown in Fig. 6.10, in which the resultant optimal boundary control strategy shows  $U_o = 0.3$ ,  $t_{on} = 17.83$  sec and  $t_{off} = 21.65$  sec.

At the time when the optimization process was finished, the resultant control strategy was automatically send to the prototype real-time system. The optimization time cost 9.8 sec was smaller than the  $t_{on} = 17.83$  sec, and thus, the optimal control strategy was successfully implemented in real-time. The prototype experimental result for implementing this proposed control strategy, which was previously shown in Fig. 5.26, is now explained with more detail in Fig. 6.11 case B. A parametric study is illustrated in the lower graph in Fig. 6.11, where nine other prototype experiments were conducted by shifting the optimized boundary control strategy. The  $x$  axis is the  $t_{on}$ , representing when the control was applied. The  $y$  axis is the system output,

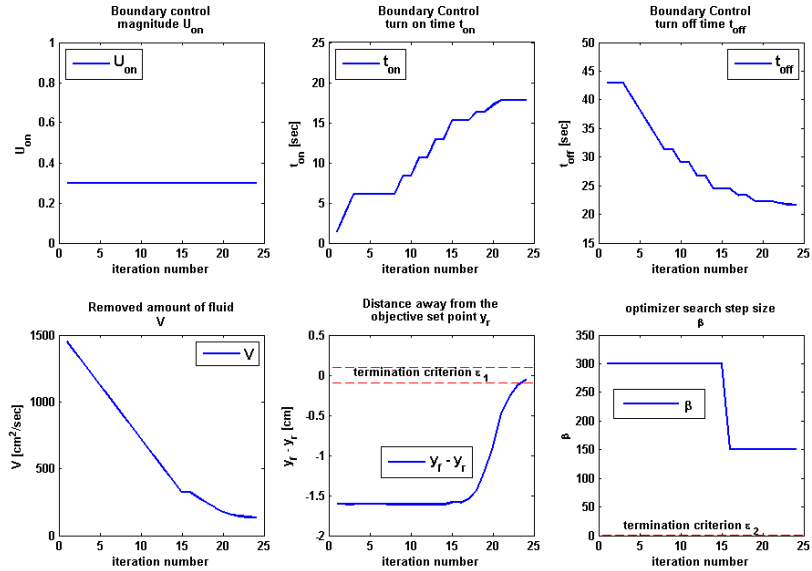


Figure 6.10: The optimization iterations for solving the 3D VPT model based minimization problem. The minimization problem considered moving a particle from  $y_{ini} = 0$  cm to  $y_r = -3$  cm. The optimization algorithm illustrated in Fig. 6.5 was applied. The total computational time cost is 9.80 sec on the HP Z420 workstation with Intel Xeon(R) E5-1620 3.60Ghz and 48G RAM.

which is the measured final location of the contaminant  $y_f$ . Despite the systematic error, the timing of the optimization solved boundary control is close to the true prototype optimum, which is indicated by the concave shaped curve produced by the prototype experimental results. In fact, the contaminant cloud was partially removed in three of the cases pointed in the figure, suggesting that the optimum timing of the boundary control lies within the time range indicated by these three cases. It could be observed in case A that, when the boundary control was applied too early in time, the boundary control action did not contribute to moving the contaminant downward. The downward movement of the contaminant was caused by the density variation discussed in Chapter III. The case C shows the same observation when the boundary control was applied too late. The implementation of the optimized control strategy is shown in case B, where the effect of the boundary control is optimized.

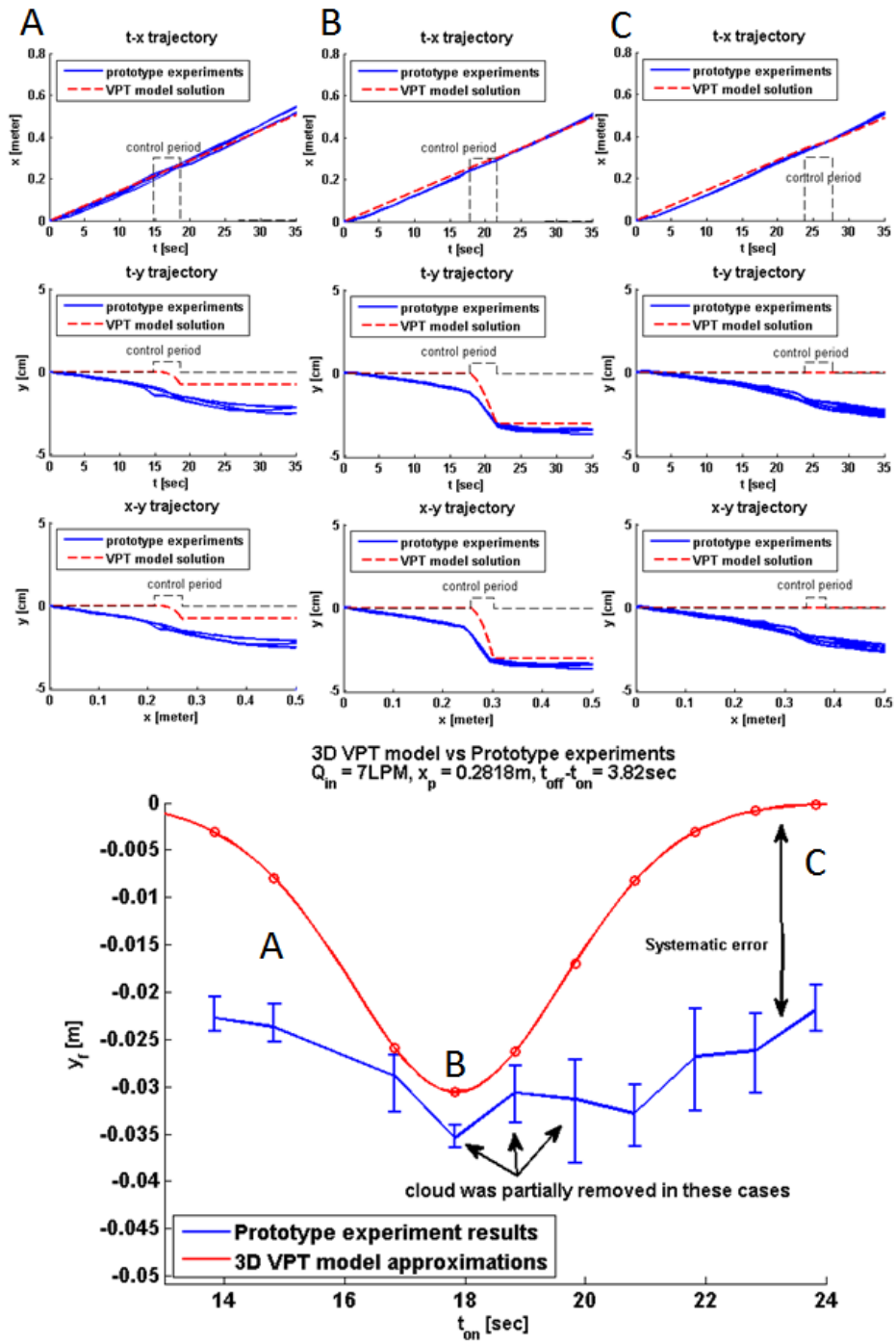


Figure 6.11: Prototype implementation and validation of the optimal control strategy solved by using the 3D VPT model.

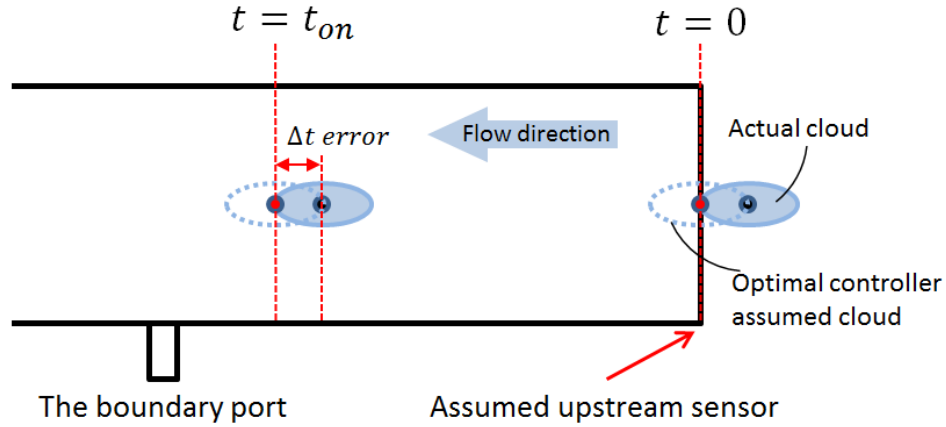


Figure 6.12: Optimal control timing error caused by the size of the color cloud and the image processing algorithm. The system time is initialized to  $t = 0$  when the color is first observed at the upstream sensor location. However, the actual whole color cloud body enters the pipe  $\Delta t$  later in time due to the size of the cloud. Thus, the assumed color cloud location in the optimization algorithm is slightly more advanced than the actual location. This  $\Delta t$  results in a slightly earlier timing in the optimization solution.

It is also observed that, the boundary control timing solved by the optimization seems slightly shifted earlier than the optimum boundary control timing indicated by the prototype parametric study. VPT model error is, of course, a potential cause of this observed timing shift. Another cause of this optimum timing shift comes from the size of the color cloud and the corresponding principle is explained in Fig. 6.12. The real-time system is initialized to  $t = 0$  when the color cloud is first observed at the assumed upstream sensor location. However, the whole color cloud body enters the pipe  $\Delta t$  later in time, due to the size of the cloud. Thus, the assumed color cloud location in the optimization algorithm is slightly more advanced than the actual cloud location. This  $\Delta t$  results in a slightly earlier boundary control timing in the optimization solution. This time shift error can be resolved by more careful image processing algorithm and is not further investigated in this study.

### 6.1.3 Summary

In summary, the real-time optimal control was successfully achieved by implementing the VPT model based optimization algorithm. The success was demonstrated via both CFD simulated results and the prototype experiments. The computational time cost associated with the optimization process is low enough to realize real-time control. The prototype experiments demonstrate the complete loop of contaminant measurement, control strategy optimization and control implementation.

## 6.2 The Feedback Control

Due to a variety of uncertainties, the mathematical model predictions differ from the actual prototype measurements. The uncertainties may include, for example, the prototype geometry variance, optical sensor observation error, unexpected flow pattern produced by the high density sponge, dynamic of the boundary port pump and etc. It is difficult and is not necessary to create a mathematical model that takes into account all these uncertainties. When a boundary control strategy is determined using the imperfect mathematical model, the actual plant output  $y_f$  will be different from the objective value  $y_r$ . For example, the optimal controller uses the VPT model to optimize control strategy, which is expected to deliver the optimization set point  $\hat{y}_f = y_r$ . When the VPT model prediction differs from the prototype output ( $\hat{y}_f \neq y_f$ ), the optimization set point is not achieved on the actual system.

Therefore, the output feedback controller is introduced to adjust the boundary control strategy  $U(t)$  directly based on the measured error  $y_f - y_r$ . As is defined in Section 6.1.2.1, the square wave shaped boundary control  $U(t)$  is ruled by three variables, its magnitude  $U_o$ , the control turn on time  $t_{on}$  and the turn off time  $t_{off}$ . The feedback process considers adjusting these three variables to drive the prototype output to the desired value.

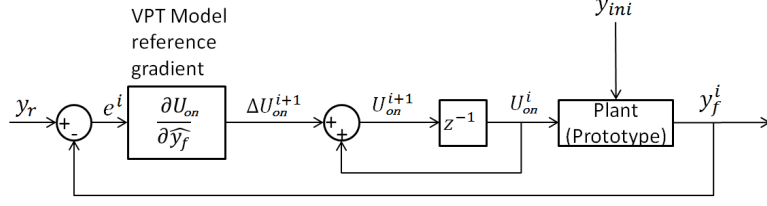


Figure 6.13: The output feedback controller using the VPT model.

In this study, defined previously in Section 2.2.2, a simple gradient based feedback control is implemented to demonstrate the effectiveness of using the VPT model as the controller reference. In the particular prototype demonstration presented in this section, only the boundary control magnitude  $U_o$  is adjusted by the feedback controller. This is because the image processing rate and the response of the water pump are too low, so that adjusting  $t_{on}$  and  $t_{off}$  become ineffective when the control period  $t_{off} - t_{on}$  is short (Refer to Chapter III for the prototype image processing and water pump dynamics).

The VPT model was used to approximate the location input-output gradient, which was the feedback gain in the controller. The prototype implemented feedback logic is defined in Fig. 6.13 and the feedback iteration is mathematically expressed as Equation (6.9)

$$\begin{cases} U_o^{i+1} &= U_o^i + \frac{dU_o}{d\hat{y}_f} e^i \\ e^i &= y_f^i - y_r \end{cases}, \quad (6.9)$$

where the superscript  $i$  represents the feedback iteration. The  $\hat{y}_f$  is the VPT model approximated system output and thus the  $dU_o/d\hat{y}_f$  is the VPT model approximated gradient. The output error  $e^i$  is the difference between the measured output  $y_f^i$  and the control objective  $y_r$  at  $i^{th}$  iteration.

The prototype demonstration of the output feedback control continuous from the optimal control strategy solved in the previous section:  $U_o = 0.3$ ,  $t_{on} = 17.83$  sec and  $t_{off} = 21.65$  sec. The prototype system output by applying the solved optimal

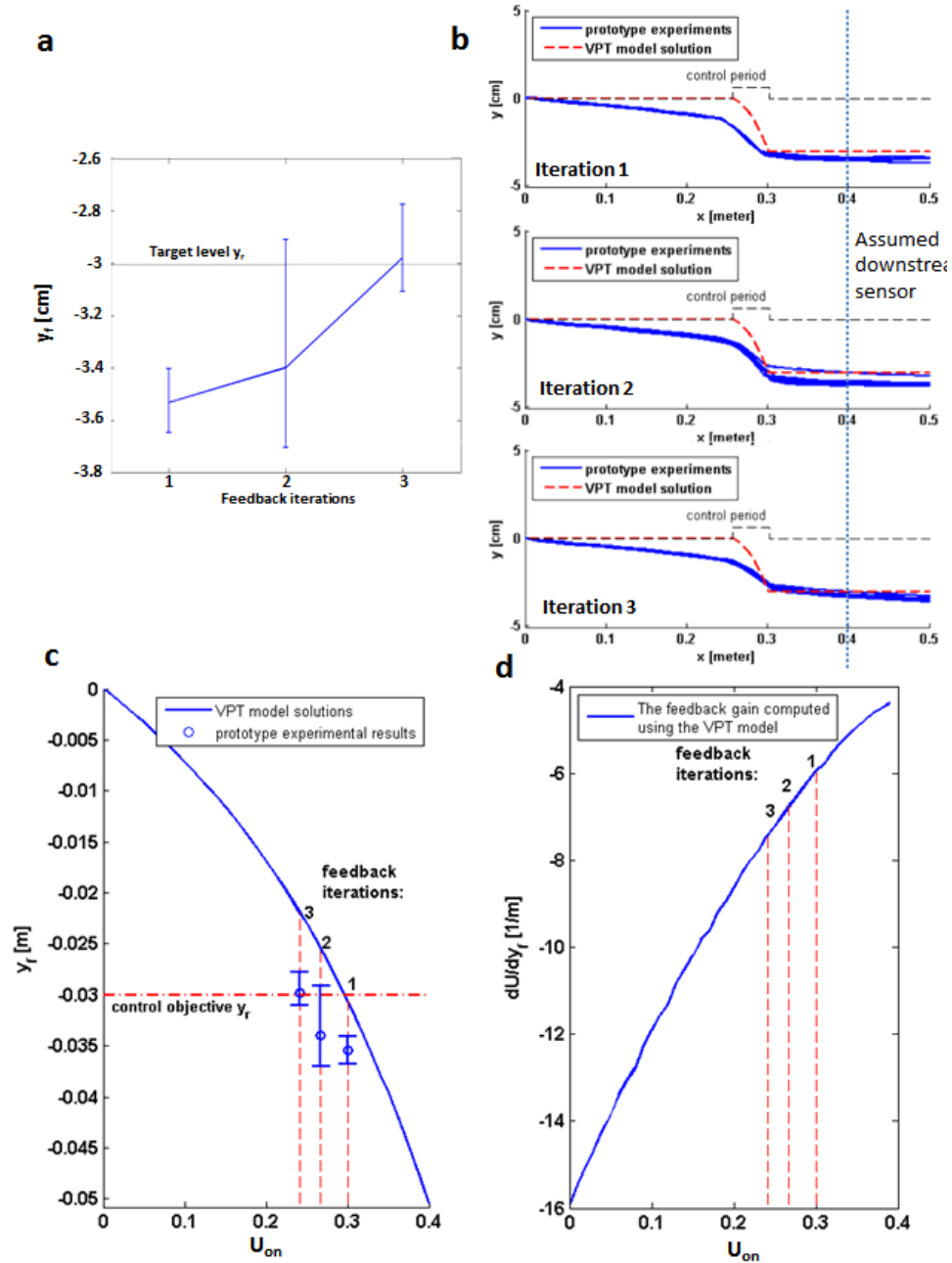


Figure 6.14: The feedback iterations and the feedback gain which is determined by the VPT model gradient. The system output by the prototype experiments converged to the objective value of  $y_r = -3\text{cm}$  within two steps.

control strategy can be read from Fig. 6.11 as  $y_f^1 = -3.532\text{ cm}$ . After implementing the feedback law in Equation (6.9) and applying the adjusted control strategies on the prototype, the feedback process terminated after two iterations. The feedback process



and the prototype experimental results are recored in Table 6.2 and are illustrated in Fig. 6.14.

Table 6.2: prototype feedback control implementation for  $y_r = -3\text{cm}$

iteration $i$	$U^i$	$y_f^i$	$e^i = y_f^i - y_r$	$dU^i/d\hat{y}_f^i$
0	0.3	-0.03546	-0.005461	-5.9514
1	0.2675	-0.03398	-0.003983	-6.674
2	0.2409	-0.02980	0.000201	

The convergence of the prototype outputs is illustrated in Fig. 6.14 a. The output converged to the target value with in 2 iterations. The prototype measured contaminant trajectories are plotted in Fig. 6.14 b, compared with the desired trajectory approximated by the VPT model. The feedback iteration mathematics are visualized in Fig. 6.14 c and d. The feedback gain, which is the VPT model input-output gradient  $dU_o/dy_f$ , is plotted in Fig. 6.14 d for  $t_{on} = 17.83$  sec and  $t_{off} = 21.65$  sec. It can be observed in Fig. 6.14 c that, although the VPT model approximated outputs are always higher than the prototype results, the VPT model approximations have the same trend compared with the actual prototype. This similar trend ensures that the VPT model approximated gradient can be utilized in the feedback algorithm.

### 6.3 VPT Model Parameter Adaptation

Increasing the fidelity of the mathematical model can improve the model reference controller performance. For example, the optimal controller will be able to find a solution closer to the true system optimum if a more accurate VPT model is employed. In this section, the concept for VPT model parameters online adaptation is presented. Due to the constraints of the prototype system, the demonstration shown in this section is limited in that only three of the model parameters were tuned. The discussion in this section is more focused on highlighting the usage and potential of the VPT model to be adapted on-line with continuous measurements.

Sensitivity analysis was first applied on the VPT model parameters. The parameter sensitivities help to determine which parameters can be effectively tuned based on measurements. The sensitivity analysis result shows that three out of the six VPT model parameters can be effectively tuned. The selected three parameters are then applied in a Least Square Parameter identification algorithm.

Totally 12 parameter sensitivities were involved interconnecting the six VPT model parameters and the two ODE states  $x$  and  $y$ . These sensitivities form the sensitivity vector expressed in Equation (6.10)

$$\mathbf{S} = \begin{pmatrix} \frac{\partial x}{\partial \lambda} \\ \frac{\partial y}{\partial \lambda} \end{pmatrix} = \begin{pmatrix} \frac{\partial x}{\partial a_1} & \frac{\partial x}{\partial a_2} & \frac{\partial x}{\partial a_3} & \frac{\partial x}{\partial b_1} & \frac{\partial x}{\partial b_2} & \frac{\partial x}{\partial \theta} \\ \frac{\partial y}{\partial a_1} & \frac{\partial y}{\partial a_2} & \frac{\partial y}{\partial a_3} & \frac{\partial y}{\partial b_1} & \frac{\partial y}{\partial b_2} & \frac{\partial y}{\partial \theta} \end{pmatrix}, \quad (6.10)$$

where  $\lambda = [a_1 \ a_2 \ a_3 \ b_1 \ b_2 \ \theta]$  is the vector containing the model parameters.

The sensitivity function that solves for the sensitivity vector  $\mathbf{S}$  is defined as Equation.(6.11)

$$\dot{\mathbf{S}} = \mathbf{A} \cdot \mathbf{S} + \mathbf{B}, \quad (6.11)$$

where  $\mathbf{A}$  and  $\mathbf{B}$  are defined in Equation (6.12)

$$\mathbf{A} = \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} \end{pmatrix}, \quad (6.12)$$

and Equation (6.13)

$$\mathbf{B} = \begin{pmatrix} \frac{\partial f_1}{\partial \lambda} \\ \frac{\partial f_2}{\partial \lambda} \end{pmatrix} = \begin{pmatrix} \frac{\partial f_1}{\partial a_1} & \frac{\partial f_1}{\partial a_2} & \frac{\partial f_1}{\partial a_3} & \frac{\partial f_1}{\partial b_1} & \frac{\partial f_1}{\partial b_2} & \frac{\partial f_1}{\partial \theta} \\ \frac{\partial f_2}{\partial a_1} & \frac{\partial f_2}{\partial a_2} & \frac{\partial f_2}{\partial a_3} & \frac{\partial f_2}{\partial b_1} & \frac{\partial f_2}{\partial b_2} & \frac{\partial f_2}{\partial \theta} \end{pmatrix}, \quad (6.13)$$

in which equations, the  $f_1$  and  $f_2$  stand for the VPT model equation right hand side

as defined in Equation (6.14)

$$\begin{cases} \dot{x} = f_1 = \frac{Q_{in}}{\pi h^2} (1 - \tau U(t)) \\ \dot{y} = f_2 = -U(t) (2h \frac{Q_{in}}{\pi h^2}) \theta \frac{(y-h)^2}{(2h)^2} \frac{1}{w\sqrt{\pi}} \exp(-\frac{(x-x_p)^2}{b_1 w^2 + b_2 (y+h)}) \end{cases}, \quad (6.14)$$

where  $\tau$  is expressed as Equation (6.15)

$$\tau = 0.5 - \operatorname{erf}\left(-\frac{x - x_p}{a_1 + a_2(y + h)}\right) + 0.5\operatorname{erf}\left(-\frac{x - x_p}{a_3}\right). \quad (6.15)$$

Totally 12 ODEs are formed in Equation (6.11). The detailed mathematical derivation of the four elements in  $\mathbf{A}$  and the 12 elements in  $\mathbf{B}$  are documented in Appendix D, which also includes the matlab code for solving the sensitivity function along with the VPT model.

Notice the fact that the downstream sensor is assumed to be fixed at a specific  $x$  location. Therefore, the particle  $x$  position should not be called a measurement. Instead, the time when the particle is observed at the sensor location is a measurement. Thus, six additional sensitivity functions for this time measurement should be added. The sensitivity for time,  $\partial t / \partial \boldsymbol{\lambda}$ , is defined as Equation (6.16)

$$\frac{\partial t}{\partial \boldsymbol{\lambda}} = \frac{\frac{\partial x}{\partial \boldsymbol{\lambda}}}{\frac{\partial x}{\partial t}} = \frac{\partial x}{\partial \boldsymbol{\lambda}} \frac{1}{\dot{x}}, \quad (6.16)$$

in which the  $\partial x / \partial \boldsymbol{\lambda}$  is the parameter sensitivity with respect the  $x$  coordinate. The corresponding sensitivity function for solving the  $\partial t / \partial \boldsymbol{\lambda}$  is defined in Equation (6.17)

$$\frac{\dot{\partial t}}{\partial \boldsymbol{\lambda}} = \frac{\dot{\partial x}}{\partial \boldsymbol{\lambda}} \frac{1}{\dot{x}}. \quad (6.17)$$

Finally, an ODE system of 20 states is formed: 2 ODEs for the VPT model, 6 ODEs for the  $\partial x / \partial \boldsymbol{\lambda}$ , 6 ODEs for the  $\partial y / \partial \boldsymbol{\lambda}$  and 6 ODEs for the  $\partial t / \partial \boldsymbol{\lambda}$ . The 18 initial conditions for the sensitivities are all zeros. The sensitivities are solved given

a specific  $y_{ini}$  and a boundary control strategy  $U(t)$ . In the following example, the 18 parameter sensitivities were solved using the optimal control strategy presented in the Section 6.1.2. The related quantities are  $y_{ini} = 0$  cm,  $U_o = 0.3$ ,  $t_{on} = 17.83$  sec and  $t_{off} = 21.65$  sec. The resultant  $\frac{\partial t}{\partial \lambda}$  and  $\frac{\partial y}{\partial \lambda}$  are plotted in Fig. 6.15. Please refer to Appendix D for the detailed process for solving the sensitivity functions.

The parameter sensitivities shown in Fig. 6.15 are plotted vs pipe  $x$  coordinate for easier correlation with the prototype sensor location. Given a location  $x$ , the absolute sensitivity magnitude represents how the change of the parameter influences the measurement if the measurement took place at  $x$ . In this study, the considered measurement is the particle  $y$  location observed at the downstream sensor. Therefore, the utilization of  $\partial y / \partial \lambda$  is demonstrated in the following example. Commonly used in practice, the relative sensitivity defined in Equation (6.18) represents the percentage change in the measured  $y_f$  resultant from a percentage change in the parameter.

$$\bar{S}_{2,j} = \frac{\lambda_{2,j}}{y_f} S_{2,j}, \quad (6.18)$$

where  $j = 1 \dots 6$  denotes the parameter index in the matrix.

The solution of the  $y$  related relative sensitivities are plotted in Fig. 6.16. The parameters with low sensitivity have weak influence on the measurement, and thus they are less estimable *Li et al.* (2004). As is shown in Fig. 6.16,  $\theta$  has the maximum sensitivity in the downstream where the sensor is present, meaning  $\theta$  has the most influence on the measurement. Parameter  $b_1$  and  $b_2$  are 10 times less influential than  $\theta$  but may still be estimable. While the low sensitivities of  $a_1$ ,  $a_2$  and  $a_3$  indicate that these three parameters could not be effectively estimated given  $y$  measurements. The estimations for  $a_1$ ,  $a_2$  and  $a_3$  may be possible based on time measurements as they have high sensitivities with respect to time (shown in Fig. 6.15). However, due to the low image sampling rate of the prototype (around 1 sec), tuning  $a_1$ ,  $a_2$  and  $a_3$  is not

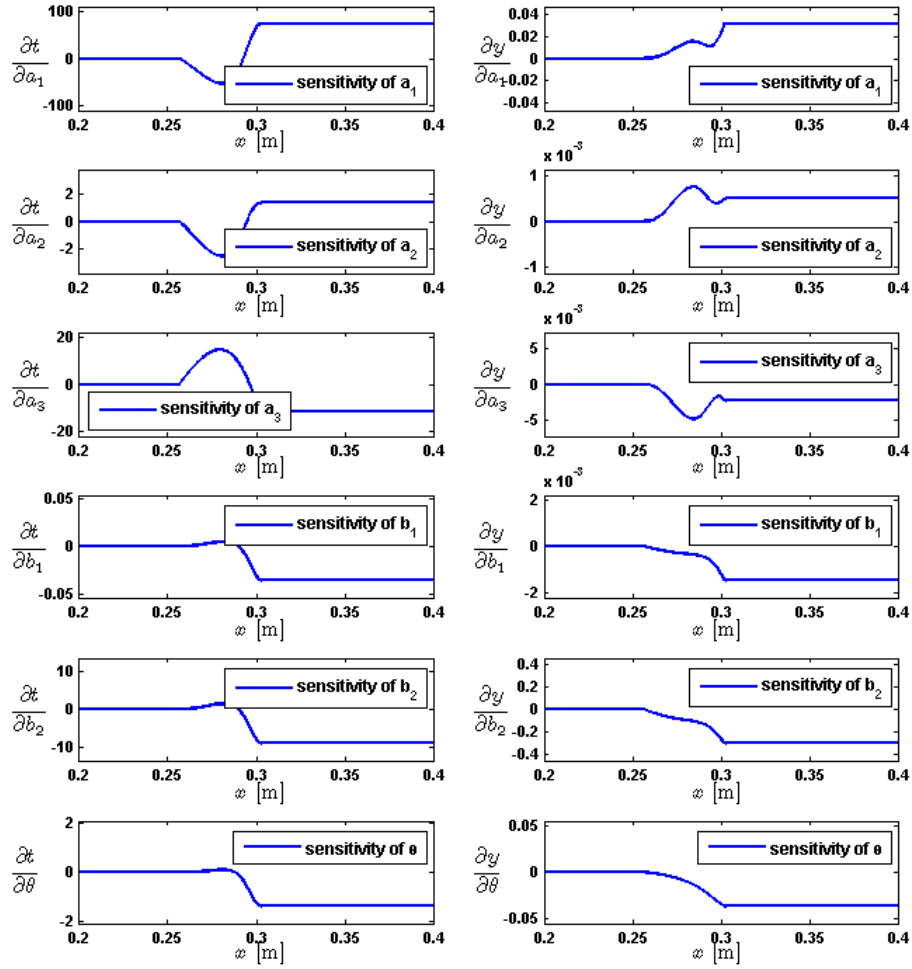


Figure 6.15: The absolute sensitivities of the 3D VPT Model Parameters. Left column plots the six model parameters' sensitivities with respect to time and the right column plots the y related sensitivities. The sensitivities are plotted vs pipe x coordinate for easier correlation with the prototype downstream sensor location.

considered in this study.

Therefore, the model parameter adaptation process was demonstrated based on  $\theta$ ,  $b_1$  and  $b_2$ . Matlab nonlinear least square optimization algorithm *lsqnonlin* was implemented. The detailed matlab implementation code is documented in the Ap-

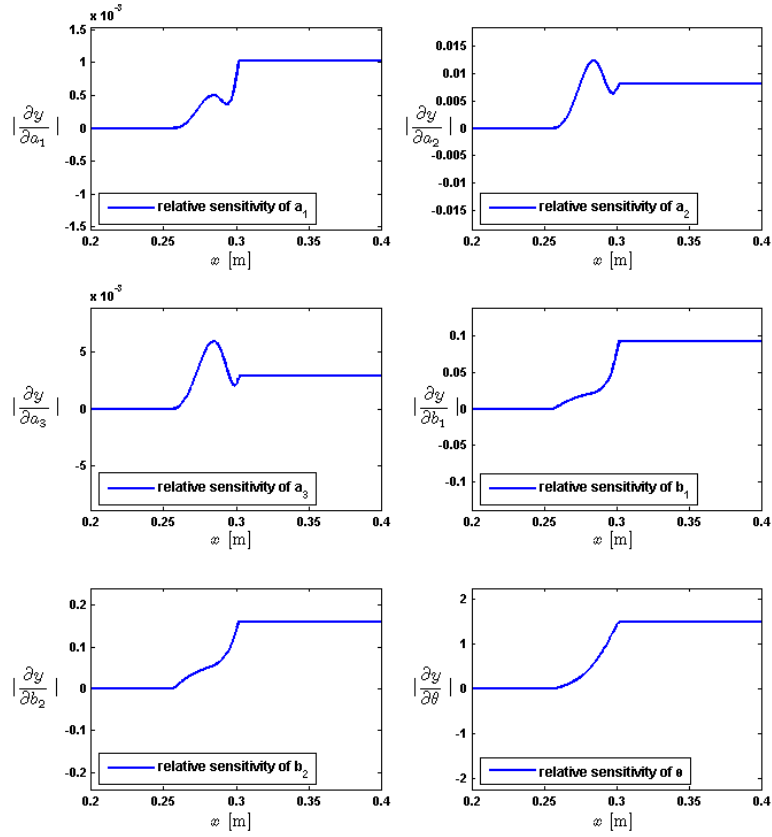


Figure 6.16: The relative sensitivities of the 3D VPT Model Parameters.

pendix E. The three prototype experimental results listed in Table. 6.2 in Section 6.2 were used as the parameter tuning data set. These experiments share the same  $t_{on}$  and  $t_{off}$ . Similar boundary control timing is expected to produce similar systematic error caused by the cloud density variation and thus these experiments are grouped together as the tuning data set.

The least square optimization (Appendix E) took 21.4969 sec on the 64-bit windows-7 laptop with Intel Core(TM) i5 M540 dual-core 2.53GHz CPU and 4GB RAM. The tuned VPT model are compared with the experimental tuning data set and the untuned VPT model in Fig. 6.17. The resultant VPT model parameters are  $b_1 = 1.9628$  (2.23% increase),  $b_2 = 0.0166$  (3.48% increase) and  $\theta = 1.4210$  (15.53% increase).

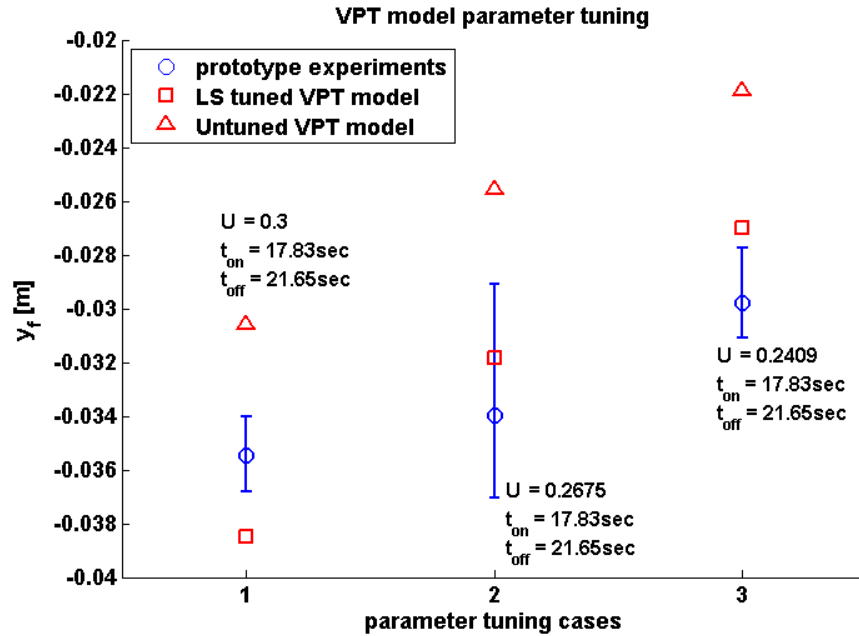


Figure 6.17: VPT Model parameters online tuning results.

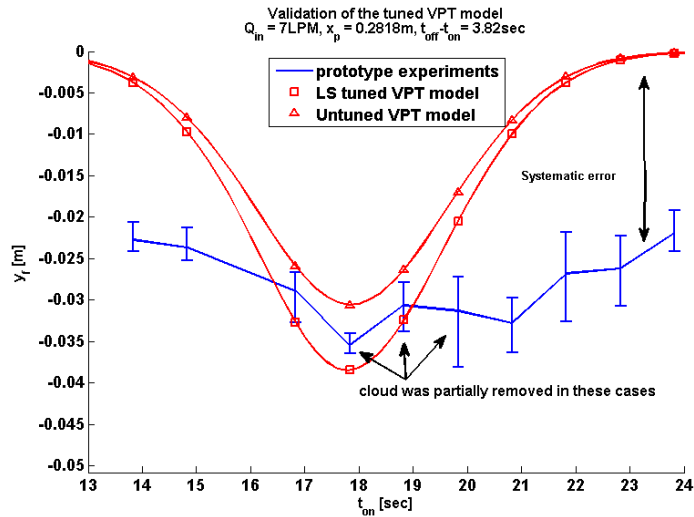


Figure 6.18: Validation of the VPT Model parameter online tuning results.

The tuned VPT model is validated against other prototype experimental cases for different  $t_{ons}$  and  $t_{offs}$  in Fig. 6.18, in which the experimental results was previously used in Fig. 5.26. The root mean square error for the 10 cases in Fig. 6.18 was reduced from 0.56 cm for the original VPT model down to 0.53 cm for the tuned

VPT model, showing a 5.12% improvements. The improvement is expected to be higher if the prototype system has lower systematic error. In this study, because the color cloud density variation is not modeled by the VPT model, tuning the VPT model parameters does not help to reduce the modelling error shown in the left and right region in Fig. 6.18 where the error caused by the density variation is dominant.

## 6.4 Summary

This chapter presents the actual implementations of the three controllers, the optimal controller, the feedback controller and the model parameter adaptation algorithm. The discussion of the optimal controller emphasizes on the VPT model based optimization. The optimal controller was successfully implemented on the prototype in real-time, realizing the complete real-time control cycle of contaminant measurement, control strategy optimization and control action deployment. The feedback controller used the VPT model to approximate the feedback gain. The feedback control algorithm was successfully validated via prototype experiments that the prototype output converged to the desired value within 2 iterations. Finally, the sensitivity analysis was demonstrated on the VPT model and the model adaptation scheme was illustrated.

The demonstrations presented in this chapter emphasis that the mathematical models developed in this study successfully enables the implementations of classical control algorithms in flow control problems.



## CHAPTER VII

### Summary and Future Directions

This dissertation presents a complete real-time control solution for controlling the contaminant particle propagation in pipelines. A real-time control architecture was proposed with two novel mathematical models that approximate the particle propagation. These two fast solvable mathematical models were employed in the real-time controllers and enabled real-time control. The real-time control was successfully demonstrated via both finite element simulations and prototype experiments.

The major achievements gained from this study is summarized in Section 7.1. Various research topics are made possible based on the work presented in this dissertation and are discussed in Section 7.2.

#### 7.1 Contributions

The flow control problem considers controlling the propagation of a contaminant plume in a circular pipe with incompressible laminar flow. The circular pipe is installed with sensor arrays that measure the location of the contaminant, and boundary ports that apply tributary boundary flow patterns to manipulate the propagation path of the contaminant. The contaminant plume is simplified as a small massless particle that is passively subject to the advection of the surrounding fluid flow. Down the length of the pipeline, the contaminant plume is measured and controlled pro-

gressively, and is finally removed from the pipeline. This study was focused on the simplified sub-problem, which considers a small massless contaminant particle propagation in a circular pipe segment that has only one boundary control port and two sensor arrays in the pipe upstream and downstream.

The overall real-time control architecture was presented in Chapter II. The control architecture consists of a feedforward optimal controller, an output feedback controller and a mathematical model adaptation process. Provided with the measured contaminant particle initial location, the feedforward optimal controller iterates a mathematical model to optimize the boundary control strategy that achieves certain control objective, for example, the particle final location measured by the downstream sensors. The feedback controller adjusts the control strategy based on the measurement error. The model adaptation algorithm tunes the mathematical model that is employed in the previous two controllers. The employed mathematical model approximates the particle propagation and can be solved fast enough to realize real-time control.

Two suitable mathematical models for the real-time flow control applications were presented in Chapter V. These two novel mathematical models approximate the contaminant particle location in the downstream given the particle initial location in the pipe upstream and the boundary port control action. The two mathematical models are used for steady flow and transient flow applications respectively. Compared to the traditional computational fluid dynamic algorithm, using the proposed mathematical models saves more than 99 percent computational time cost. This fast speed advantage of the proposed mathematical models realizes the real-time flow control implementations. In addition, demonstrated vis finite element computational fluid dynamics simulation under idea conditions, the accuracy of the presented mathematical models is high enough for control applications. Particularly, the Algebraic Particle Tracing(APT) Model (for steady flow) shows nearly zero error compared to

2D CFD simulated results. Furthermore, the Velocity field Particle Tracking(VPT) Model (for transient flow) demonstrates high flexibility for modelling a pipe flow system of higher complexity, for example, a pipe that has multiple boundary ports. A flow system of multiple boundary ports can be easily modeled using the VPT model as the basic building block, without re-calibrating the model parameters. This benefit of the VPT model enables faster and easier construction of flow problems of higher complexity.

In addition, the VPT model is formatted in the ordinary differential equation form, which is the common mathematical model formation used in many classical control algorithms. Therefore, the VPT model creates a convenient bridge that links classical control algorithms with flow control applications.

A prototype flow system was designed and fabricated in this study (Chapter III). This prototype system represents the one boundary port flow control sub-problem. The contaminant is represented by a color cloud and is monitored using a live camera. The prototype system resembles the basic necessary components to realize the flow control problem, including the camera which represents the sensor array, the boundary port powered by a water pump, a embedded real-time control using dSPACE system and a computing work station that provides computational power for processing the sensor data and running the controllers. The successful implementation of the real-time control on the prototype system (Chapter VI) validates the feasibility of applying flow control in real world applications, particularly for the contaminant elimination problem introduced in Chapter I.

Overall, the presented work established realistic starting point for the contaminant elimination flow control problem, elevating the state of art of real-time flow control closer to real world applications.

## 7.2 Future Direction

This study was focused on the simplified single boundary port sub-problem and emphasized the prototype realization of real-time control. Based on the presented work, both the breadth and the depth of this research can be extended.

### 7.2.1 Apply 2D Control Algorithm in the 3D Pipe Problem

The majority discussion in this dissertation emphasizes on the 2D problem and the 3D pipe symmetric plane. This section proposed a concept for implementing the presented 2D control algorithm directly on the 3D pipe control problem, which involves multiple boundary port around the pipe.

The first step to illustrate the concept is shown in Fig. 7.1. The figure shows several steady flow 3D Finite Element Analysis (FEA) simulation results. The top drawing in Fig. 7.1 shows the FEA simulated particle spatial trajectories in the 3D pipe, which has one boundary port at its bottom. Hundreds of massless and dimensionless particles are released and tracked from the inlet boundary. The spatial trajectories are plotted for those particles which are removed by the boundary port drawing flow. The 3D pipe has a geometry symmetric plane, which makes it sufficient to demonstrate the flow system only using half of the pipe. In this 3D demonstration, the trajectories are generated for the case when the upstream total inlet flow rate is  $Q_{in} = 10.3\text{cm}^3/\text{s}$  with pipe radius of  $h = 5.08\text{cm}$  and boundary port diameter of  $w = 1.27\text{cm}$ . In the lower 10 graphs in Fig. 7.1, each dot represents one tracked particle, positioned at the particle releasing location on the inlet boundary. The particles are marked blue if they are to be removed by the boundary port drawing flow, while the remaining particles are in gray color. The quantity  $U$  represents the boundary control strength, which is the ratio between the boundary port flow rate and the total upstream inlet flow rate. The definition of  $U$  is previously discussed in Chapter V.

It is observed in Fig. 7.1 that, within certain distance from the symmetric plane,

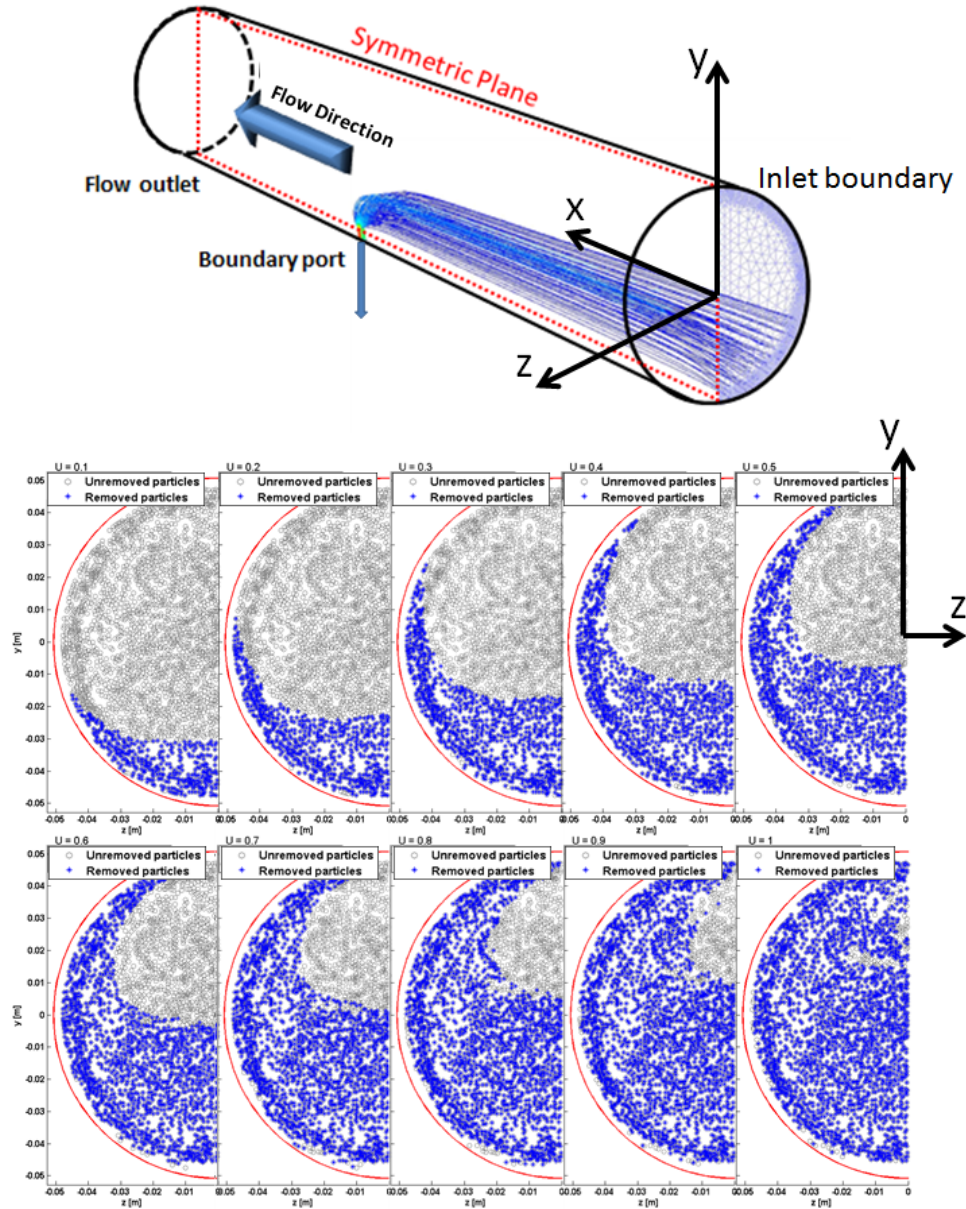


Figure 7.1: The removed particles and the remaining ones. A portion of the tracked particles are removed by the boundary port drawing action. The particles are released from random spots on the pipe inlet boundary. The removed particles are marked blue, while the remaining are in gray color. The inlet flow rate is  $Q_{in} = 10.3\text{cm}^3/\text{s}$  with pipe radius of  $h = 5.08\text{cm}$  and boundary port diameter of  $w = 1.27\text{cm}$ . The boundary port flow rate is  $Q_p = U * Q_{in}$ , where the  $U$  is defined as the ratio between the boundary port flow rate  $Q_p$  and  $Q_{in}$  in Section 5.1 .

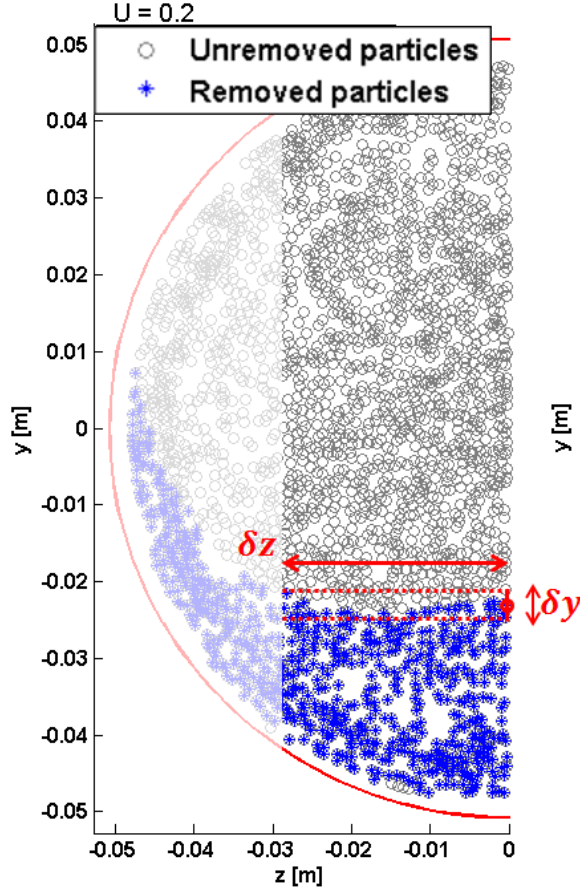


Figure 7.2: The top most particles that are removed by the boundary port flow showing similar  $y_{ini}$ s within a distance from the pipe symmetric plane.

if the initial  $y$  coordinates ( $y_{ini}$ s) of a group of particles are similar, these particles tend to have the same final conditions of being removed or not being removed. For example, as shown in Fig. 7.2, given a boundary flow rate, the top most particles that are removed by the boundary control have very similar  $y_{ini}$ s within a distance  $\delta z$  from the symmetric plane. In other words, for the particles with same initial locations, the control outcomes of these particles (with respect to the  $y$  coordinate within a tolerance of  $\delta y$ ) are approximately the same within a distance from the pipe symmetric plane. In Fig. 7.2,  $\delta y = 0.3\text{cm}$  is used for example, and the resultant  $\delta z$  is approximately 3cm.

Following the  $\delta y - \delta z$  routine for different boundary control magnitude  $Us$ , an

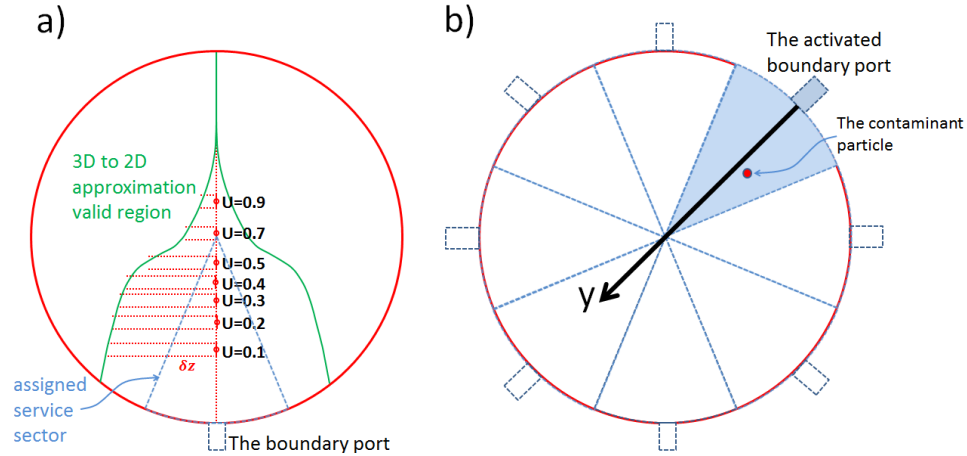


Figure 7.3: The concept for problem order reduction from 3D to 2D sub-problems. In a), the approximation valid region is the area in which the particle  $y$  dynamics can be approximated by the dynamics on the pipe symmetric plane. Graph b) shows that the entire 3D pipe is divided into 8 service sectors, each sector is assigned to a boundary port. If the sector fits in the approximation valid region for the corresponding boundary port, the dynamics within the sector can be reduced from 3D to 2D. Depending on where the contaminant particle is observed, the corresponding service sector and the boundary port will be activated.

envelope can be created in Fig. 7.3 a). The region within this envelope can be controlled using the 2D control algorithm by neglecting the  $z$  coordinate. As a result, a simple control strategy can be formulated by splitting the 3D pipe into several sectors as shown in Fig. 7.3 b). One boundary port is installed for each sector. The size of each sector is smaller than the envelope region. Therefore, the 2D control algorithm can be used for each individual boundary port. The initial location of the particle determines which boundary port will be active. However, operating multiple boundary ports at the same time is not discussed in above 3D pipe control concept.

### 7.2.2 Extending the Mathematical Modelling Techniques

The potential applications of the mathematical modelling techniques (Chapter V) can be investigated. The APT model modelling technique can be potentially used to model a pipe network as illustrated in Fig. 7.4, given the knowledge of the velocity

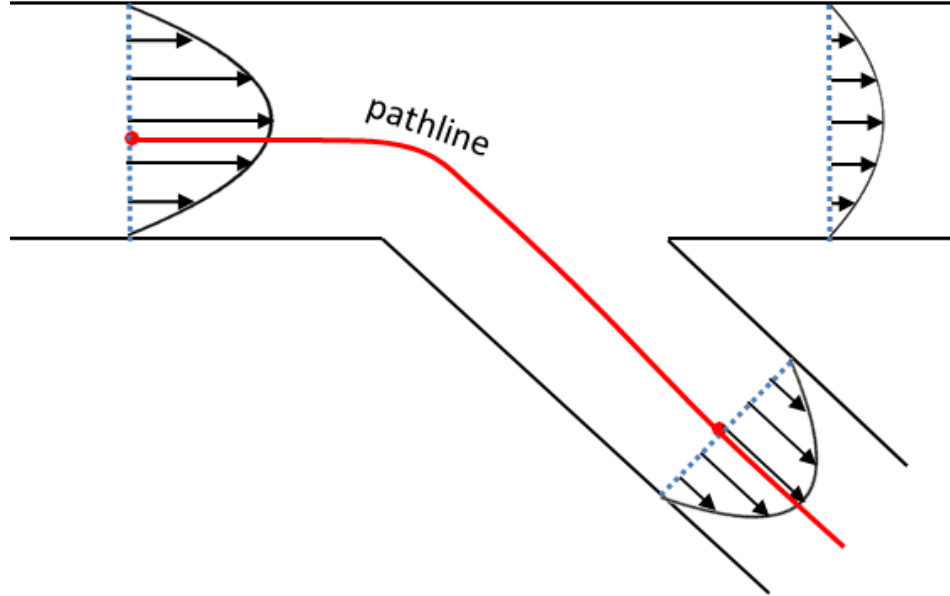


Figure 7.4: The APT model proposed for a 2D pipe network. The APT modelling technique can be potentially used to create a mathematical model for computing particle destination in a complex pipe network.

profiles at all pipe outlets. The VPT model modelling technique can also be extended for a pipe of different geometry. For example, approximating the velocity field in a round pipe corner. As discussed in Section 5.2.3, the VPT model can be used as a building block to construct a flow system. Therefore, enriching the building block library enables easier modelling of flow system of higher complexity.

Furthermore, the VPT model can be potentially extended to approximate not only the contaminant plume location, but also the contaminant plume shape as illustrated in Fig. 7.5. Finite number of particles resemble the boundary shape of the contaminant plume. Interaction force between the adjacent particles can be introduced to represent the surface tension, which counter reacts to the velocity difference between the two particles. It can be imagined that, when the interaction forces are all zeros, the situation reduces to the independent particle condition implied in this dissertation. When the interaction force is infinite, the contaminant plume becomes



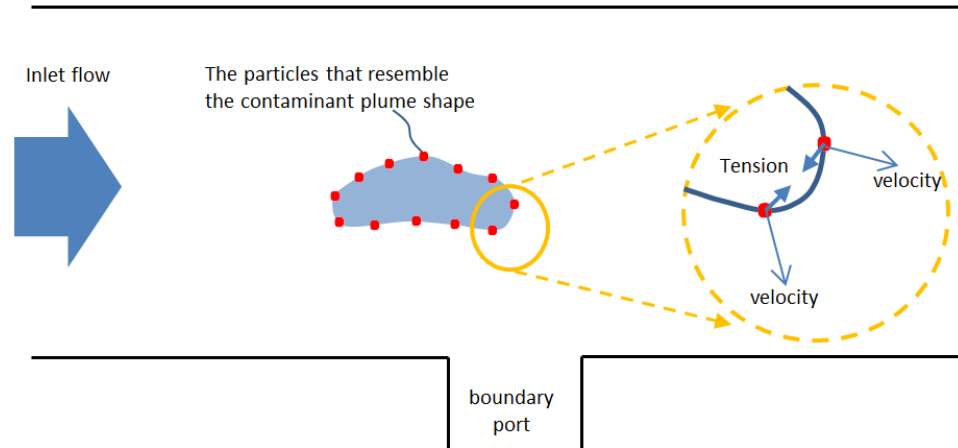


Figure 7.5: The VPT model that can potentially approximate the contaminant plume shape

a rigid body.

### 7.2.3 The Prototype System

The prototype system fabricated in this study is based on the simplified single boundary port control problem. Constructing a prototype system with multiple boundary ports and developing the corresponding control algorithm is potentially an attractive research topic. In addition, improvements can be made to the prototype. For example, a faster image processing algorithm, a smaller sized pipe that produce smooth fully developed laminar flow, more contaminant cloud injection spots and etc.

In conclusion, the outcomes of this study substantially advance the state of art of real-time flow control closer to real world applications. Many related research topics can be proposed based on the presented work.

### 7.2.4 Source Inversion Problem

In the contaminant source inversion problem, the measured contaminant solute location and/or concentration values are used to reconstruct the source of the contaminant. Contaminant source inversion problem is frequently discussed in the case

of accidental chemical release into the groundwater, atmosphere or a wall-bounded domain. The latter one covers the similar flow geometry discussed in this dissertation. Determining the initial status of the contaminant allows better understanding of the contaminant nature and better control action can be taken. Few examples for source inversion applications are summarized here. A method for determining the location of the source of an airborne chemical in a wall-bounded flow domain was presented by *Boggs et al.* (2006). A vast number of studies were conducted on groundwater contamination source identification (*Gorelick et al.* (1983), *Mahar and Datta* (1997), *Alapati et al.* (2000), *Yeh et al.* (2007), *Jha and Datta* (2012)). The source inversion technique can also be used for solving the atmospheric source characterization such as the wind speed/direction and contaminant source (*Bagtzoglou and Baun* (2005), *Chow et al.* (2008), *Addepalli et al.* (2011)).

The two mathematical models proposed in this dissertation can be potentially implemented in the source inversion problem, providing faster solving speed. Presented in Chapter V, the Algebraic Particle Tracing (APT) model is a simple algebraic relationship that relates the particle initial location, the boundary control and the particle final location. Therefore, the particle initial location can be computed directly given the particle final location. Also presented in Chapter V, the Velocity field Particle Tracking (VPT) model is in the form of Ordinary Differential Equations (ODEs). Therefore, the source inversion problem can be formulated by simulating the VPT model backward in time. This can be achieved by reversing the sign of any time related quantities in the VPT model equation, namely, adding negative sign to the equation right-hand side and reversing the boundary control wave form  $U(t)$  in time and magnitude.

## APPENDICES

## APPENDIX A

# Prototype Real-time Control Program Implemented via dSPACE and Matlab

This appendix documents the matlab and simulink program that was implemented during prototype experiments. The entire program is composed of four parts:

1. The dSPACE real-time system simulink program, which is loaded on the dSPACE real-time system.
2. The dSPACE control desk user interface penal, which is loaded on the host computing work station. The user interface allows the real-time communication between the dSPACE hardware board that connects to the prototype, and the computing work station which is responsible for intensive computing job such as solving the VPT model.
3. The camera interface simulink program which controls the live camera and pre-processes the received image data. This simulink program is located on the computing work station.
4. The main matlab program that communicates between the dSPACE real-time system and the host computing work station. This matlab program calls the camera simulink program, processes the image data, runs the optimal control algorithm and sends the boundary control strategy commands to the dSPACE real-time system.

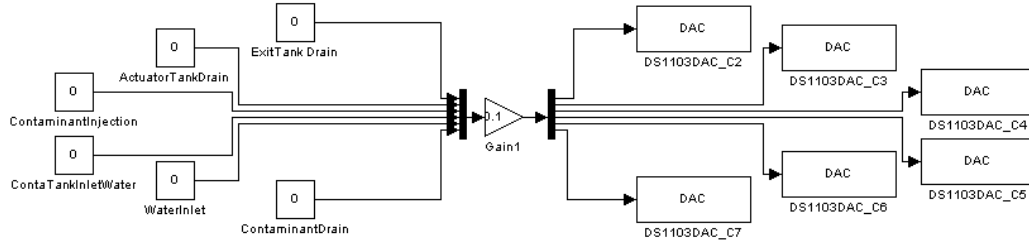


Figure A.1: dSPACE hardware loaded with the simulink program for valve control. Six solenoid air valves are installed on the prototype.

The software packages are:

matlab R2011b (32-bit) loaded with dSPACE(R) software for matlab(R) 7.13.0.564.

The dSPACE packages includes: RTI 6.8, RTIFPGA 2.2 and MLIB/MTRACE 4.7.3.

The dSPACE hardware is the DS1103 PPC Board.

### The dSPACE simulink program

The dSPACE simulink program is divided into three sections. The simulink program in the first section (Fig. A.1) controls the six solenoid air valves on the prototype system. The *DAC* simulink block is the interface that connects to the dSPACE hardware board analog output ports. The second section of the simulink program (Fig. A.2) receives data from the ultrasonic flow meter and the two pressure sensors. The third part (Fig. A.3) is the controller for controlling the boundary port flow rate delivered by the boundary pump. The boundary control profile is specified by the *ControlData* vector, which is computed by the optimal controller documented later in this appendix. The pump control signal is sent to the prototype pump controller through the *DS1103DAC\_C1* hardware output port.

### The dSPACE user interface

During experiments, the dSPACE user interface is loaded on the same PC (the computing work station) that processes the image processing and runs optimal controller. The real-time communication between the prototype and the controller is achieved in this way. The user interface is shown in Fig. A.4. The top six air valve switches

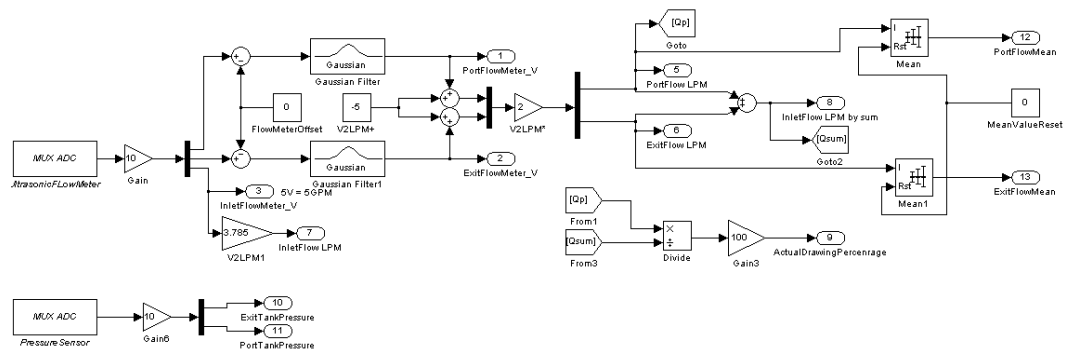


Figure A.2: dSPACE hardware loaded with the simulink program for receiving flow meter and pressure sensor data.

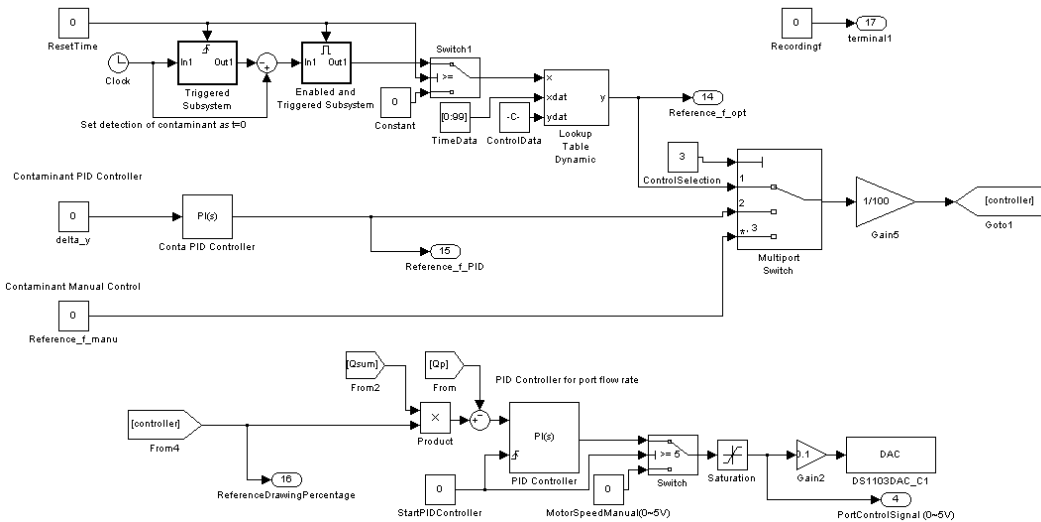


Figure A.3: dSPACE hardware loaded with the simulink program for controlling the water pump flow rate.

map to the DAC output ports shown in Fig. A.1. The *MechineTime* is the system time that will be synchronized between the dSPACE hardware board and the host PC. The system time is set to zero when the contaminant is first observed by the camera (Refer to the optimal control discussed in Section 6.1.2.3 and Fig. 6.6). The I/Os in the second frame map to the simulink problem in Fig. A.2 and display the measured values in real-time. The boundary port flow and the downstream exit flow are measured using high resolution ultrasonic flow meters. For incompressible fluid and rigid pipes, these two flow rates sum to the inlet flow. The extra rotary vane flow meter monitors the inlet flow to double check the measured value. The Flow Meter Offset input block tunes the measurement error that was caused by the analog signal voltage drop. The bottom frame includes the variables for controlling the boundary port flow. These variables are linked to the prototype system through the simulink problem shown in Fig. A.3. There are three different control options to choose from:

Option 1. The PID controller. The boundary port control magnitude  $U(t)$  is adjusted manually using the knob below the option selector. The PID controller adjusts the pump power based on the measured boundary port flow rate.

Option 2. The Manual controller. This option allows the user to control the power of the water pump directly using the slider positioned on the right of this frame.

Option 3. The Optimal controller. This option was the one selected when the real-time optimal controller was implemented. The boundary port flow rate follows the boundary control strategy profile, which is defined by the *ControlData* vector in Fig. A.3. This option and the three variables are set by the matlab program documented later in this appendix.

### **The camera interface simulink program**

The matlab image acquisition toolbox is required to link a live camera to the matlab workspace through a USB port. In this study, Microsoft LifeCam HD3000 was used. The simulink block diagram is shown in Fig. A.5. The image is split into RGB three

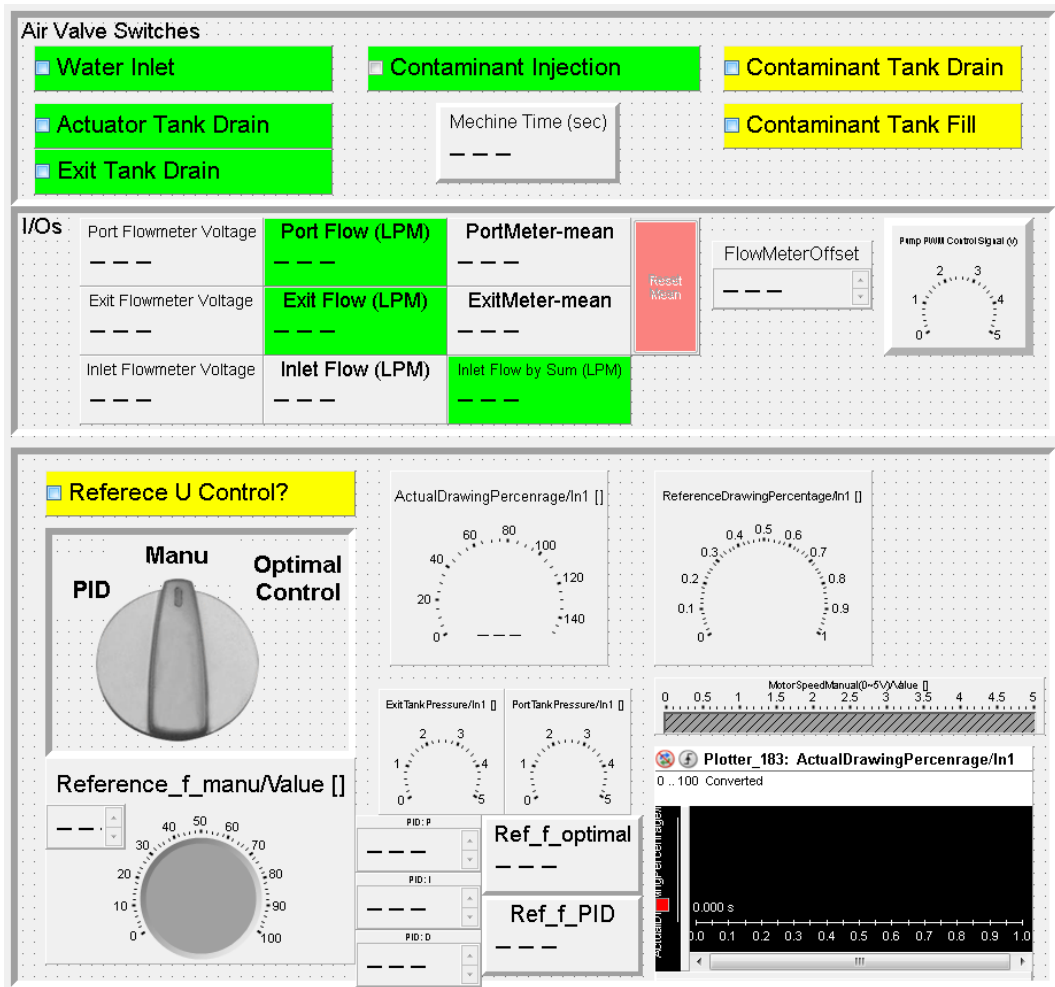


Figure A.4: The dSPACE user interface on the host PC.



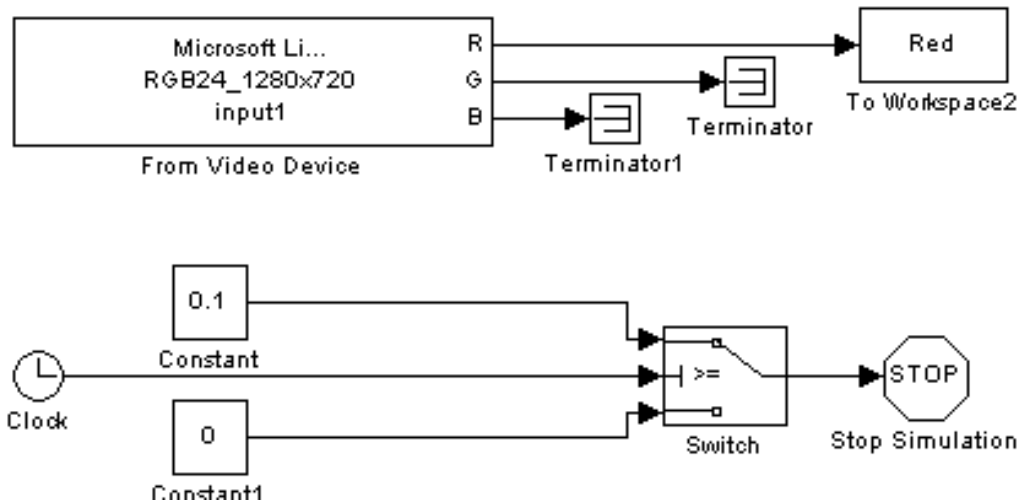


Figure A.5: The simulink program that links the live camera to the matlab environment.

color density. The red color density is used in this setup because the blue color cloud produced high contrast in red color component. This simulink program only runs for 0.1 sec. This time span was calibrated that 0.1 sec is about the minimum time

### The main real-time control matlab program

The main matlab program communicates between the dSPACE real-time system and the host computing work station. This matlab program is the main program that processes the camera image, runs the optimal controller and sends the boundary control strategy to the dSPACE real-time control system.

```

1 clear;
2 clc;
3 % run the camera simulink program and get background color
4 sim('CameraProgram.mdl') % include file path here
5 Back_Red = Red.signals.values;
6
7 % initialize intermidient variable
8 imsize = size(Back_Red);

```

```

9 temp2 = zeros(imsize);
10 % initialize contaminant cloud location
11 Xc = 1200; % the image pixel number
12 Yc = 1200; % the image pixel number
13 time = zeros(1,TestDataLength);
14 % initialize actual contaminant location X and Y
15 Xa = 0; % start of the monitored pipe region
16 Ya = 0; % the middle of the pipe
17
18 % Define iteration number
19 % the average image sampling rate is 1sec.
20 % therefore, length of 300 records about 5 minutes test.
21 TestDataLength = 300;
22
23 % define the pipe area to be monitored
24 window_pipe = [265 997 313 498]; % by pixel number
25 % initialize some image variable data type
26 window_p = uint8(temp2);
27 detectwindow = uint8(temp2);
28 window_p(window_pipe(3):window_pipe(4),window_pipe(1): ...
29         window_pipe(2)) = 1;
30 detectwindow(window_pipe(3):window_pipe(4), (window_pipe(2)-20): ...
31         window_pipe(2)) = 1;
32
33 % trim the background image
34 Back_Red = Back_Red.*window_p;
35
36 % initialize dSPACE-matlab interface
37 boards_info = mlib('GetBoardInfo');
38 mlibini
39 mlib('SelectBoard', 'ds1103')
40 timer_addr = mlib('GetTrcVar', 'Model Root/ResetTime/Value');
41 timer_state = 1;

```

```

42 mlib('Write', timer_addr, 'Data', timer_state)
43
44 % declare the address on dSPACE hardware board for the boundary ...
    control signal
45 % (The ControlData in the lookup table)
46 Control_var = {'Model Root/TimeData/Value'; 'Model ...
    Root/ControlData/Value'};
47 Control_addr = mlib('GetTrcVar', Control_var);
48 startcontrol = 10;
49 StartControl_addr = mlib('GetTrcVar', 'Model ...
    Root/StartPIDController/Value');
50 mlib('Write', timer_addr, 'Data', 0) % initialize timer
51
52 % Define the variable to select Control options
53 % (PID, Manual and Optimal control)
54 Control_selection = {'Model Root/ControlSelection/Value'};
55 Control_selection_addr = mlib('GetTrcVar', Control_selection);
56 Manual = 3;
57 ContaPID = 2;
58 OptimalControl = 1;
59
60
61 % declare dSPACE address for reading the measurement data
62 % the boundary control U:
63 DrawingP_var = {'Model Root/ActualDrawingPercentage/In1'};
64 DrawingP_addr = mlib('GetTrcVar', DrawingP_var);
65 DrawingP_data = zeros(1,TestDataLength);
66 % The downstream exit flow rate
67 ExitQ_var = {'Model Root/ExitFlow LPM/In1'};
68 ExitQ_addr = mlib('GetTrcVar', ExitQ_var);
69 ExitQ_data = zeros(1,TestDataLength);
70 % The boundary port flow rate
71 PortQ_var = {'Model Root/PortFlow LPM/In1'};

```

```

72 PortQ_addr = mlib('GetTrcVar', PortQ_var);
73 PortQ_data = zeros(1,TestDataLength);
74 % The accumulated exit volume of fluid
75 ExitV_var = {'Model Root/ActualDrawingPercentage/In1'};
76 ExitV_addr = mlib('GetTrcVar', ExitV_var);
77 ExitV_data = zeros(1,TestDataLength);
78 % The accumulated port volume of fluid
79 PortV_var = {'Model Root/ActualDrawingPercentage/In1'};
80 PortV_addr = mlib('GetTrcVar', PortV_var);
81 PortV_data = zeros(1,TestDataLength);
82 % the contaminant injection command
83 Cinj_var = {'Model Root/ContaminantInjection/Value'};
84 Cinj_addr = mlib('GetTrcVar', Cinj_var);
85
86 % The following is the feedback control algorithm
87 % if the optimal controller is to be implemented,
88 %insert the optimal control code here
89
90 % set feedback control info
91 T = [0:99]; % in this example, the boundary control resolution is ...
           1 sec.
92 % initialize the boundary control
93 ControlSig = zeros(1,100);
94 NoControl = zeros(1,100);
95 ExitQ_data_ini = mlib('Read', ExitQ_addr);
96 PortQ_data_ini = mlib('Read', PortQ_addr);
97 Qin = ExitQ_data_ini+PortQ_data_ini; % prototype inlet flow rate
98 beta = 2000 / Qin; % feedback gain is 2000.
99
100 % boundary port location in the image by pixel
101 X_port_p = 643;
102 portmountsize = 2; % inches
103 Xcoor_sensor = 998;

```

```

104 % the actual boundary port location in meter:
105 X_port = portmountsize/(675-611) * (Xcoor_sensor-X_port_p) * ...
        25.4/1000;
106
107 %(refer to Fig.\ref{Opl_Camera_Portxp})
108
109 ton_test = 17;
110 toff_test = 20;
111 dt = toff_test - ton_test;
112 % set boundary control level
113 % note that the dSPACE originally
114 % used percentage to represents the boundary control
115 U = 0.192*100;
116
117 % define the control vector for the dSPACE
118 ControlSig(floor(ton_test+1):floor(toff_test+1)) = U;
119
120 % initialize feedback parameters
121 fb = 0; % initialize the feedback iteration number
122 Yr = 0.03; % desired final location
123 datasave = 0; % initialize when to save feedback data
124 sendsig = 0;
125
126 % here starts the image capture:
127
128 % define image prcessing color sevel:
129 ColorLevel = 25;
130 % initialize and synchronize system time:
131 currenttime = datevec(now);
132 time(1) = currenttime(4)*60*60+currenttime(5)*60+currenttime(6);
133 time_test(1) = 0;
134
135 % inject the contaninant cloud for 0.07 sec

```

```

136 mlib('Write', Cinj_addr, 'Data', 5)
137 pause(0.07)
138 mlib('Write', Cinj_addr, 'Data', 0)
139
140 % In this example code, the feedback iterate is removed for ...
    simplicity
141 % for fb = 1:10
142
143 for i = 1:TestDataLength-1
144
145     % Get Flow data
146     DrawingP_data(i) = mlib('Read', DrawingP_addr);
147     ExitQ_data(i) = mlib('Read', ExitQ_addr);
148     PortQ_data(i) = mlib('Read', PortQ_addr);
149     ExitV_data(i) = mlib('Read', DrawingP_addr);
150     PortV_data(i) = mlib('Read', ExitV_addr);
151
152     % get image data and time
153     currenttime = datevec(now);
154     time(i+1) = ...
        currenttime(4)*60*60+currenttime(5)*60+currenttime(6);
155     time_test(i+1) = time(i+1)-time(1);
156     sim('CameraProgram.mdl')
157     Red = Red.signals.values.*window_p;
158
159     % compare the new image with the background color
160     ImDiff = abs(double(Red) - double(Back_Red));
161
162
163     % filter
164     temp = (ImDiff>=ColorLevel)*255;
165     nonzero = find(temp>0);
166     temp2 = zeros(imsi);

```

```

167     for ii = 1:length(nonzero)
168         temp2(nonzero(ii)) = ...
169             median([temp(nonzero(ii)-imsize(1)-1) ...
170                    temp(nonzero(ii)-1) ...
171                    temp(nonzero(ii)+imsize(1)-1) ...
172                    temp(nonzero(ii)-imsize(1)) ...
173                    temp(nonzero(ii)) ...
174                    temp(nonzero(ii)+imsize(1)) ...
175                    temp(nonzero(ii)-imsize(1)+1) ...
176                    temp(nonzero(ii)+1) ...
177                    temp(nonzero(ii)+imsize(1)+1) ...
178                    ]);
179     end
180
181     % erase the new contaminant cloud from the image
182     % if the contaminant from the last test is still in the pipe
183     if Xc(i) ≠ 1200
184         temp2(:, [(floor(Xc(i))+100):1280]) = 0;
185         temp2(:, [1:floor(Xc(i))-100]) = 0;
186     else % detect if new contaminant appear at inlet
187         injectdetect = temp2(window_pipe(3):window_pipe(4), ...
188                               (window_pipe(2)-70):window_pipe(2)
189                               );
190         detectSig = sum(sum(injectdetect>25));
191         if detectSig ≠ 0
192             temp2(:, (1:(window_pipe(2)-100))) = 0;
193         else
194             temp2 = temp2*0;
195         end
196     end
197
198     % find contaminant location:
199

```

```

200 nonzero2 = find(temp2≠0);
201 % initialize a intermediate variable X, Y
202 X = 0;
203 Y = 0;
204
205 for ii = 1: length(nonzero2)
206     Y(ii) = mod(nonzero2(ii), imsize(1));
207     X(ii) = (nonzero2(ii) - Y(ii))/imsize(1) + 1;
208 end
209
210 if isempty(nonzero2) == 1
211     Xc(i+1) = 1200;
212     Yc(i+1) = 1200;
213 else
214     Xc(i+1) = sum(X)/length(X);
215     Yc(i+1) = sum(Y)/length(Y);
216 end
217
218 % compute the actual contaminant location from pixel to meter
219 % note that the pipe is 4 inch in diameter for computing Ya
220 Xa(i+1)=(window_pipe(2)-Xc(i+1)) * portmountsize/(675-611) * ...
    0.0254;
221 Ya(i+1)=(Yc(i+1)-window_pipe(4)) * ...
    4/(window_pipe(3)-window_pipe(4))*0.0254;
222
223 % send pre defined control command and start control
224 % if Xc < 1160, which is different from the initialized ...
    value 1200,
225 % a contaminant cloud is observed.
226 if Xc(i+1) < 1160 && sendsig == 0
227     display(['At Step = ' num2str(i)])
228     display('Control signal sent')
229     display(['ton = ' num2str(ton_test)]);

```



```

230     display(['toff = ' num2str(toff_test)]);
231     display(['U = ' num2str(U)]);
232     mlib('Write', timer_addr, 'Data', timer_state)
233     mlib('Write', Control_addr(2), 'Data', ControlSig_OPLM);
234     mlib('Write', Control_selection_addr, 'Data', 1);
235     mlib('Write', StartControl_addr(1), 'Data', startcontrol);
236     sendsig = 1;
237     end
238
239     % if Xc value goes back to 1200, the contaminant is out ...
        of the monitored area
240     if Xc(i+1) == 1200 && sendsig == 1;
241         display(['At Step = ' num2str(i)])
242         display('Contaminant disappear')
243         tempyf = Yr(i);
244         tempxf = Xr(i);
245         if Xr(i) < 0.5;
246             display('Contaminant draw away')
247             tempyf = 0;
248         end
249
250         eval(['Yf' num2str(i) '=tempyf;' ])
251         eval(['Xf' num2str(i) '=tempxf;' ])
252         eval(['error' num2str(i) '=tempyf - Yfstar;' ])
253         display(['Error = ' num2str(tempyf - Yfstar)])
254
255         eval(['ton' num2str(i) '=ton_test;' ])
256         eval(['toff' num2str(i) '=toff_test;' ])
257         eval(['U' num2str(i) '=U;' ])
258         mlib('Write', timer_addr, 'Data', 0)
259         mlib('Write', Control_addr(2), 'Data', NoControl);
260         mlib('Write', StartControl_addr(1), 'Data', 0);
261         mlib('Write', Control_selection_addr, 'Data', 3);

```

```
262     display(['New ton = ' num2str(ton_test)])
263     display(['New toff = ' num2str(toff_test)])
264     display(['New U = ' num2str(U)])
265     sendsig = 0;
266     mlib('Write', Cinj_addr, 'Data', 5)
267     pause(0.07)
268     mlib('Write', Cinj_addr, 'Data', 0)
269     end
270 end
271
272 % end
```

## APPENDIX B

### VPT Model Matlab Code

This appendix documents the VPT model matlab function and the ode23 option setting for solving the VPT model.

#### The matlab ode23 solver that calls the VPT model

```
1 options = odeset('Events', @events_eliminate, 'RelTol', ...  
    1e-8, 'MaxStep', '1');  
2 [T,Y,TE,YE,IE] = ode23(@VPTModel, tspan, VPT.ic, options, ...  
    VPT.Parameter);
```

#### The matlab function: *events\_eliminate*

The *events\_eliminate* matlab function defines simulation pipe geometry region.

```
1 function [lookfor stop direction] = ...  
    events_eliminate(t,x,VPT.Parameter)  
2  
3 % the x(1) is the particle x coordinate solved by the VPT model.  
4 % the x(2) is the particle y coordinate solved by the VPT model.  
5 % the following four values defines the 4 pipe geometry boundary.
```

```

6
7 a = x(1)-0.9; % the total pipe length is 0.9 meter
8 b = x(2)+0.0508; % the pipe lower boundary, h = 0.0508 meter
9 c = x(2)-0.0508; % the pipe upper boundary, h = 0.0508 meter
10 d = x(1)+0.1; % this is an optional condition for the case if the ...
    particle moves backward in x.
11
12 lookfor = (a > 0 | b < 0 | c > 0 | d < 0) - 1;
13 % the particle is tagged as eliminated if any of the four ...
    condition is satisfied.
14
15 stop = 1; % terminate the ode23 solver
16 direction = 1; % get all the zeros

```

### The matlab function for the VPT model

The VPT model is formulated as 2 states ODEs.

```

1
2 function dx = VPTModel(t,x,VPT.Parameter)
3 % the VPT.Parameter stores all the parameters and boundary ...
    control singal variables
4
5 dx=zeros(2,1); % the VPT model has two states
6
7 % the three variables that defines the square wave boundary control
8 Uo   = VPT.Parameter.Uo;      % boundary control magnitude
9 ton  = VPT.Parameter.ton;     % control turn on time
10 toff = VPT.Parameter.toff;   % control turn off time
11
12 % the flow conditions and pipe geometry
13 Vu   = VPT.Parameter.Vu;     % upstream mean velocity
14 Qu   = VPT.Parameter.Qu;     % upstream volume flow rate ...

```

```

    [cm^2/s] for 2D and [cm^3/s] for 3D
15 xp    = VPT.Parameter.xp;      % boundary port location
16 w     = VPT.Parameter.w;      % boundary port width
17 D     = VPT.Parameter.D;      % pipe diameter, which is 2h
18 h     = D/2;
19
20 % the six VPT model parameters
21 a1    = VPT.Parameter.a1;
22 a2    = VPT.Parameter.a2;
23 a3    = VPT.Parameter.a3;
24 b1    = VPT.Parameter.b1;
25 b2    = VPT.Parameter.b2;
26 theta = VPT.Parameter.theta;
27
28 % the square wave boundary control
29 if t ≥ ton && t ≤ toff
30     U = Uo;
31 else
32     U = 0;
33 end
34 % for different boundary control wave form, different equation ...
    should be used
35 % for example, the sine wave boundary control should be written as
36 % U = Uo/2 - Uo/2*sin(t)
37 % for the example shown in Chapter 5
38
39 % the ordinary differential equations
40 v1 = 1.5*Vu*(1 - x(2).^2/h^2); % the upstream velocity profile
41 % for 3D pipe fully developed laminar flow, the v1 should be ...
    written as:
42 % v1 = 2*Vu*(1- x(2).^2/h^2);
43 % for a uniform velocity profile the v1 should be written as:
44 % v1 = Vu;

```

```

45
46 % the transision term for dx(1)
47 tau    = 0.5 - erf((xp-x(1))./(a1+a2*(x(2)+h))) + 0.5* ...
           erf((xp-x(1))/a3);
48
49 % the VPT model equations
50 dx(1) = v1.* (1 - tau*U);
51 dx(2) = - U* Qu *theta * (x(2)-h).^2 / (2*h)^2 / w / pi^0.5 * ...
           exp(- (x(1)-xp)^2/(b1*w^2 + b2*(x(2)+h)));

```

## APPENDIX C

### VPT Model Based Optimization Algorithm

This appendix documents the matlab program implementation of the optimization algorithm presented in Section 6.1.2 (Fig 6.5.)

```
1
2 % define system constants
3 h = 0.0508;          % pipe half height for 2D or the radius for the ...
                      3D pipe
4 D = 2*h;
5 w = 0.0127;         % boundary port width or diameter
6 xp = 0.51435;      % boundary port location from the ...
                      upstream sensor
7 Vu = 0.005;        % upstream flow mean velocity
8 Qu = Vu * D;       % upstream flow rate for 2-dimensional ...
                      geometry
9
10 % the six VPT model parameters
11 a1 = 0.001;
12 a2 = 0.5;
13 a3 = 0.037;
```

```

14 b1 = 1;
15 b2 = 0.01;
16 theta = 0.9571;
17
18 % the optimal controller set points
19 ControlObject.yini = 0;      % contaminant particle initial location
20 ControlObject.yr = -0.02;   % desired particle final location in ...
    the downstream
21
22 % Initialize the optimizer
23 Iter_limit = 100;           % maximum number of ...
    iterations
24
25 % Initialize the three boundary control variables - the variables ...
    to be optimized
26 Uo = zeros(Iter_limit+1,1); % boundary control magnitude
27 ton = zeros(Iter_limit+1,1); % boundary control turn on time
28 toff = zeros(Iter_limit+1,1); % boundary control turn off time
29 Uo(1) = 0.3;
30 ton(1) = 10;
31 toff(1) = 300;
32
33 Inputs = [Uo, ton, toff];
34 VPT.Parameter.Uo = Uo(1);
35 VPT.Parameter.ton = ton(1);
36 VPT.Parameter.toff = toff(1);
37
38 % prepare for the VPTmodel function
39 VPT.Parameter.Vu = Vu;
40 VPT.Parameter.Qu = Qu;
41 VPT.Parameter.xp = xp;
42 VPT.Parameter.D = D;
43 VPT.Parameter.w = w;

```



```

44 VPT.Parameter.a1 = a1;
45 VPT.Parameter.a2 = a2;
46 VPT.Parameter.a3 = a3;
47 VPT.Parameter.b1 = b1;
48 VPT.Parameter.b2 = b2;
49 VPT.Parameter.theta = theta;
50
51 % run the VPT model for the first time outside iteration loop
52 options = odeset('Events', @events_eliminate, 'RelTol', ...
    1e-8, 'MaxStep', 1);
53 [T, Y, TE, YE, IE] = ode23(@VPTmodel, 0:0.05:300, [0 ...
    ControlObject.yini], options, VPT.Parameter);
54 Yf_VPT(1) = Y(end, 2); % the particle final y location
55 Xf_VPT(1) = Y(end, 1); % the particle final x location
56 % if the final x location is near the boundary port region, the ...
    particle final y location is adjusted due to the ode23 ...
    resolution reason
57 if Xf_VPT(1) < VPT.Parameter.xp + 0.1
58     Yf_VPT(1) = -0.0508;
59 end
60
61 % record some other variables that are of interest
62 Yf_error = zeros(Opti_limit+1, 1); % the distant away from the ...
    set point y_r
63 Yf_error(1) = Yf_VPT(1) - ControlObject.Yf;
64 Vq = zeros(Opti_limit+1, 1); % removed amount of fluid
65 beta_iter = zeros(Opti_limit+1, 1); % the searching step size
66
67 % if the first searching point does not produce enough output
68 % the optimal set point cannot be achieved
69 if Yf_error(1) > 0
70     error('The control objective cannot be achieved')
71 end

```

```

72
73 % here starts the optimization loop
74 for i = 1:Opti_limit % maximum number of iteration
75
76     % the first temrination criteria epsilon 1
77     if abs(Y_error(i)) ≤ 0.0001
78         break
79     end
80     % the second temrination criteria epsilon 2
81     if beta_iter(i) ≤ 1
82         break
83     end
84
85     % the sub iteration loop
86
87     % defineing the local perturbation:
88     dVq = beta_iter(i) * Yf_error(i); % the local perturbation size
89     % three options for adjusting boundary control:
90     ton_temp = ton(i) - dVq/Uo(i); % postpone ton
91     toff_temp = toff(i) + dVq/Uo(i); % advancing toff
92     Uo_temp = Uo(i) + dVq/(toff(i)-ton(i)); % reduce control ...
93     % magnitude
94     Input_temp = [Uo(i), ton_temp, toff(i);... % option 1
95                 Uo(i), ton(i), toff_temp];... % option 2
96                 U_temp, ton(i), toff(i)]; % option 3
97     Yf_error_temp = [0;0;0]; % initialize the comparison matrix ...
98     % for the three options
99     % conducting the local perturbation parametric study:
100    for i_test = 1:3 % try three options
101        VPT.Parameter.Uo = Input_temp(i_test,1);
102        VPT.Parameter.ton = Input_temp(i_test,2);
103        VPT.Parameter.toff = Input_temp(i_test,3);
104        [T, Y, TE, YE, IE] = ode23(@VPTmodel, 0:0.05:300, [0 ...

```

```

        ControlObject.yini], options, VPT.Parameter);
103
104     Yf_VPT_temp(i_test) = Y(end,2);
105     Xf_VPT_temp(i_test) = Y(end,1);
106     if Xf_VPT_temp(i_test) < VPT.Parameter.xp + 0.1
107         Yf_VPT_temp(i_test) = -0.0508;
108     end
109     Yf_error_temp(i_test) = Yf_VPT_temp(i_test) - ...
        ControlObject.Yf;
110 end
111
112     % determine which option is the best and is the next ...
        searching point
113     if sum(sign(Yf_error_temp) == sign(Yf_error(1))) == 0 % the ...
        step size is too large?
114         beta_iter(i+1) = beta_iter(i)/2;
115         Uo(i+1) = Uo(i);
116         ton(i+1) = ton(i);
117         toff(i+1) = toff(i);
118         Yf_error(i+1) = Yf_error(i);
119         Yf_VPT(i+1) = Yf_VPT(i);
120         Xf_VPT(i+1) = Xf_VPT(i);
121         Vq(i+1) = Vq(i); % removed amount of fluid
122     else
123         beta_iter(i+1) = beta_iter(i);
124         [temp_a, temp_b] = min(Yf_error_temp); % find which ...
            option has larger error
125         Uo(i+1) = Input_temp(temp_b,1);
126         ton(i+1) = Input_temp(temp_b,2);
127         toff(i+1) = Input_temp(temp_b,3);
128         Yf_error(i+1) = Yf_error_temp(temp_b);
129         Yf_VPT(i+1) = Yf_VPT_temp(temp_b);
130         Xf_VPT(i+1) = Xf_VPT_temp(temp_b);

```

```
131         Vq(i+1) = VPT.Parameter.Qu*Uo(i+1)*(toff(i+1)-ton(i+1));
132     end
133 end
134
135 % remove redundant data points
136 Uo(Uo==0) = [];
137 ton(ton==0) = [];
138 toff(toff==0) = [];
139 Yf_error(Yf_error==0) = [];
140 beta_iter(beta_iter==0) = [];
141 Vq(Vq == 0) = [];
```

## APPENDIX D

### VPT Model Sensitivity Function

This appendix documents the mathematical derivation of the VPT model parameter sensitivities and associated matlab code.

#### Mathematical derivation of the sensitivity function

#### Matlab function for solving the parameter sensitivities

The sensitivities of the parameters are solved together with the mathematical model. Therefore, the sensitivity function is build into the VPT model matlab function.

```
1 function dx = VPT.Sensitivity(t,x,VPT.Parameter)
2
3 dx=zeros(20,1);
4 % 2 VPT model states
5 % 6 parameter sensitivities with respect to x
6 % 6 parameter sensitivities with respect to y
7 % 6 parameter sensitivities with respect to time
8
```

```

9 % the three variables that defines the square wave boundary control
10 Uo = VPT.Parameter.Uo; % boundary control magnitude
11 ton = VPT.Parameter.ton; % control turn on time
12 toff = VPT.Parameter.toff; % control turn off time
13
14 % the flow conditions and pipe geometry
15 Vu = VPT.Parameter.Vu; % upstream mean velocity
16 Qu = VPT.Parameter.Qu; % upstream volume flow rate ...
    [cm^2/s] for 2D and [cm^3/s] for 3D
17 xp = VPT.Parameter.xp; % boundary port location
18 w = VPT.Parameter.w; % boundary port width
19 D = VPT.Parameter.D; % pipe diameter, which is 2h
20 h = D/2;
21
22 % the six VPT model parameters
23 a1 = VPT.Parameter.a1;
24 a2 = VPT.Parameter.a2;
25 a3 = VPT.Parameter.a3;
26 b1 = VPT.Parameter.b1;
27 b2 = VPT.Parameter.b2;
28 theta = VPT.Parameter.theta;
29
30 % the square wave boundary control
31 if t ≥ ton && t ≤ toff
32     U = Uo;
33 else
34     U = 0;
35 end
36
37 v1 = 2*Vu*(1 - x(2).^2/h^2); % 3D pipe parabolic velocity profile
38 % v1 = Vu; % 3D pipe uniform velocity profile
39
40 c1 = Vu;

```

```

41 c2 = -U*D*Vu / (D^2 * w * pi^0.5 ); % equivalent 2D flow rate ...
    computed using 3D volume flow rate
42
43 g1 = (xp - x(1))./(a1 + a2*(x(2)+h));
44 g2 = (xp - x(1))./(a3);
45 g3 = -((x(1)-xp).^2)./(b2*(x(2)+h)+b1*w^2);
46
47 dvxx = -c1 .* U .* ...
    ( ...
48     2/pi^0.5 .* exp(-(g1.^2)) ./ (a1+a2.*(x(2)+h)) ...
49     - 1/pi^0.5 .* exp(-(g2.^2)) ./ a3 ...
50     );
51
52
53 dvxy = -c1 .* U .* ...
    ( ...
54     -2/pi^0.5 .* exp(-(g1.^2)) .* (x(1) - xp) ./ ...
55     ((a1+a2.*(x(2)+h)).^2) .* a2 ...
56     );
57
58
59 dvyx = c2.*theta.* ((x(2)-h).^2) .* exp(g3) .* ...
    (...
60     (-1) ./ (b1*w^2 + b2 .* (x(2)+h)) .* ...
61     (2.*(x(1) - xp)) ...
62     );
63
64
65 dvyy = c2.*theta* 2 .* (x(2)-h).*exp(g3) ...
66     + c2.*theta.*((x(2)-h).^2).*exp(g3) ...
67     .* ( ...
68     (x(1)-xp).^2./ ((b1*w^2 + b2 ...
69     .* (x(2)+h)).^2) .* b2 ...
70     );
71
72 dvxa1 = -c1 .* U .* ...

```

```

73         .* ( ...
74             -2./pi^0.5 .* exp(-(g1.^2)) ...
75                 .* (x(1)-xp) ./ ((a1+a2.*(x(2)+h)).^2) ...
76         );
77
78 dvxa2 = -c1 .* U ...
79         .* ( ...
80             -2./pi^0.5 .* exp(-(g1.^2)) ...
81                 .* (x(1)-xp) ./ ((a1+a2.*(x(2)+h)).^2) ...
82                 .* (x(2)+h) ...
83         );
84
85 dvxa3 = -c1 .* U ...
86         .* ( ...
87             1/pi^0.5 .* exp(-(g2.^2)) ...
88                 .* (x(1)-xp) ./ ((a3).^2) ...
89         );
90
91 dvxtheta = 0;
92 dvxb1 = 0;
93 dvxb2 = 0;
94
95 dvya1 = 0;
96 dvya2 = 0;
97 dvya3 = 0;
98
99 dvytheta = c2.*(x(2)-h).^2.*exp(g3);
100
101 dvyb1 = c2.*theta.*(x(2)-h).^2.*exp(g3) ...
102         .* ( ...
103             ((x(1)-xp).^2) ...
104                 ./ ((b1*w^2 + b2.*(x(2)+h)).^2) ...
105             .* w^2 ...

```



```

106         );
107 dvyb2 = c2.*theta.*(x(2)-h).^2).*exp(g3) ...
108         .* ( ...
109             ((x(1)-xp).^2) ...
110             ./ ((b1*w^2 + b2.*(x(2)+h)).^2) ...
111             .* (x(2)+h) ...
112         );
113
114
115 dx(1) = v1.*...
116         ( ...
117         (1- ...
118         ( ...
119         0.5 - ...
120         erf(g1) + ...
121         0.5*erf(g2) ...
122         ).*U ...
123         ) ...
124         );
125 dx(2) = c2.*theta*(h-x(2)).^2 .* exp( g3 );
126
127 % sensitivity with respect to x
128 % dxa1
129 dx(3) = dvxx .* x(3) + dvxy .* x(9) + dvxa1;
130 % dxa2
131 dx(4) = dvxx .* x(4) + dvxy .* x(10) + dvxa2;
132 % dxa3
133 dx(5) = dvxx .* x(5) + dvxy .* x(11) + dvxa3;
134 % dxb1
135 dx(6) = dvxx .* x(6) + dvxy .* x(12) + dvxb1;
136 % dxb2
137 dx(7) = dvxx .* x(7) + dvxy .* x(13) + dvxb2;
138 % dxtheta

```

```

139 dx(8) = dvxx .* x(8) + dvxy .* x(14) + dvxtheta;
140
141 % sensitivity with respect to y
142 % dya1
143 dx(9) = dvyx .* x(3) + dvyy .* x(9) + dya1;
144 % dya2
145 dx(10) = dvyx .* x(4) + dvyy .* x(10) + dya2;
146 % dya3
147 dx(11) = dvyx .* x(5) + dvyy .* x(11) + dya3;
148 % dyb1
149 dx(12) = dvyx .* x(6) + dvyy .* x(12) + dyb1;
150 % dyb2
151 dx(13) = dvyx .* x(7) + dvyy .* x(13) + dyb2;
152 % dytheta
153 dx(14) = dvyx .* x(8) + dvyy .* x(14) + dytheta;
154
155 % sensitivity with respect to t
156 dx(15) = dx(3) ./ dx(1) ;
157 dx(16) = dx(4) ./ dx(1) ;
158 dx(17) = dx(5) ./ dx(1) ;
159 dx(18) = dx(6) ./ dx(1) ;
160 dx(19) = dx(7) ./ dx(1) ;
161 dx(20) = dx(8) ./ dx(1) ;

```

## APPENDIX E

### VPT Model Parameter Adaptation

This appendix documents matlab code for the VPT model adaptation algorithm. According to the sensitivity analysis presented in Section 6.3, three of the VPT model parameters,  $b_1$ ,  $b_2$  and  $\theta$ , are included in the following parameter adaptation process.

```
1 lambda_ic = [1 1 1]; % normalize the three parameters
2 % least square optimization upper and lower bounds
3 lb = lambda_ic*0.5;
4 ub = lambda_ic *2;
5 % Invoke the least square optimizer
6 [x,resnorm] = lsqnonlin(@VPTLSError,lambda_ic);
7
8 % the VPTLSError function
9 function LSError = VPTLSError(lambda)
10
11 Xcoor_portmount = [611 675];
12 Xcoor_sensor = 998;
13 portmountsize = 2; % inches
14 SystemParameters.xp = ...
    portmountsize/(Xcoor_portmount(2)-Xcoor_portmount(1)) * ...
    (Xcoor_sensor-sum(Xcoor_portmount)/2) * 25.4/1000;
```

```

15
16 % set the VPT model parameter values
17 a1 = 0.001;
18 a2=0.5;
19 a3=0.037;
20 b1 = lambda(1) * 3.0968e-004;
21 b2 = lambda(2) * 0.016;
22 theta = lambda(3) * 1.23;
23
24 % define VPT model parameters
25 w = 0.0127;
26 D = 0.1016;
27 h = D/2;
28 VPT.Parameter.w = w;
29 VPT.Parameter.D = D;
30 VPT.Parameter.xp = SystemParameters.xp;
31 VPT.Parameter.Vu = 7*0.001/60/pi/0.0508^2; % the inlet mean flow ...
    rate for Qin = 7LPM
32 VPT.Parameter.Qu = VPT.Parameter.Vu*VPT.Parameter.D; % the ...
    equivalent 2D flow rate
33 VPT.Parameter.a1 = a1;
34 VPT.Parameter.a2 = a2;
35 VPT.Parameter.a3 = a3;
36 VPT.Parameter.b1 = b1;
37 VPT.Parameter.b2 = b2;
38 VPT.Parameter.theta = theta;
39
40 % The prototype experiments data
41 Uo = [0.3 0.2675 0.2409];
42 ton = [17.8291 17.8291 17.8291];
43 toff = [21.6513 21.6513 21.6513];
44 y-f = [-0.03546 -0.03398 -0.02980];
45

```

```

46 % solve the VPT model
47 options = odeset('Events', @events_eliminate, 'RelTol', 1e-8, ...
    'MaxStep', 1);
48 error = zeros(size(Uo));
49 for i = 1: length(Uo)
50     VPT.Parameter.Uo = Uo(i);
51     VPT.Parameter.ton = ton(i);
52     VPT.Parameter.toff = toff(i);
53     [T,Y,TE,YE,IE] = ode23(@OPL_3D_Final_Proto, 0:0.05:400, ...
        [0,0], options, VPT.Parameter);
54     error(i) = (Y(end,2)-y_f(i))^2; % compute the least ...
        square error
55     clearvars T Y
56 end
57
58 % compute the total least square error
59 LSError = sum(error);

```

## BIBLIOGRAPHY

## BIBLIOGRAPHY

- Aamo, O. M., M. Krstic, and T. R. Bewley (2003), Control of mixing by boundary feedback in 2d channel flow, *Automatica*, 39, 1597-1606.
- Abergel, F., and R. Temam (1990), On some control problems in fluid mechanics, *Theoretical and Computational Fluid Dynamics*, 1(6), 303–325.
- Acheson, D. (1990), *Elementary Fluid Dynamics*, Oxford Applied Mathematics and Computing Science Series, Clarendon Press.
- Addepalli, B., K. Sikorski, E. Pardyjak, and M. Zhdanov (2011), Source characterization of atmospheric releases using stochastic search and regularized gradient optimization, *Inverse Problems in Science and Engineering*, 19(8), 1097–1124.
- Alapati, S., Z. Kabala, et al. (2000), Recovering the release history of a groundwater contaminant using a non-linear least-squares method, *Hydrological processes*, 14(6), 1003–1016.
- Alvarez-Vázquez, L., A. Martínez, M. Vázquez-Méndez, and M. Vilar (2009), An application of optimal control theory to river pollution remediation, *Applied Numerical Mathematics*, 59(5), 845–858.
- Åström, K., and T. Hägglund (2006), *Advanced PID Control*, ISA-The Instrumentation, Systems, and Automation Society.
- Aström, K., and R. Murray (2010), *Feedback Systems: An Introduction for Scientists and Engineers*, 201 pp., Princeton University Press.
- Åström, K., and B. Wittenmark (2013), *Adaptive Control: Second Edition*, Dover Books on Electrical Engineering, Dover Publications.
- Åström, K. J., and K. Furuta (2000), Swinging up a pendulum by energy control, *Automatica*, 36(2), 287–295.
- Avriel, M. (2012), *Nonlinear Programming: Analysis and Methods*, Dover Books on Computer Science, Dover Publications.
- Babcock, D., C. Lee, B. Gupta, J. Kim, and R. Goodman (1996), Active drag reduction using neural networks, in *Neural Networks for Identification, Control, Robotics, and Signal/Image Processing, 1996. Proceedings., International Workshop on*, pp. 279–287, IEEE.

- Bagtzoglou, A. C., and S. A. Baun (2005), Near real-time atmospheric contamination source identification by an optimization-based inverse method, *Inverse Problems in Science and Engineering*, *13*(3), 241–259.
- Balogh, A., O. M. Aamo, and M. Krstic (2005), Optimal mixing enhancement in 3-d pipe flow, *IEEE Transactions on Control Systems Technology*, *13*, 27–41.
- Bastin, G., F. Jarachi, and I. M. Mareels (1999), Output deadbeat control of non-linear discrete-time systems with one-dimensional zero dynamics: global stability conditions, *IEEE Transactions on Automatic Control*, *44*(6), 1262–1266, doi:10.1109/9.769387.
- Becerra, V. M. (2008), Optimal control, *Scholarpedia*, *3*(1), 5354, revision 124632.
- Bennett, F. V., T. G. Price, and M. S. Center (1964), *Study of Powered-Descent Trajectories for Manned Lunar Landings*, National Aeronautics and Space Administration.
- Boggs, P. T., K. R. Long, S. B. Margolis, and P. A. Howard (2006), Rapid source inversion for chemical/biological attacks, part 1: The steady-state case, *SIAM J. on Optimization*, *17*(2), 430–458, doi:10.1137/040603036.
- Bretscher, O. (2013), *Linear Algebra with Applications*, Section 8.3 pp., Pearson Education.
- Bryson, A. (2002), *Applied Linear Optimal Control: Examples and Algorithms*, 107 pp., Cambridge University Press.
- Choi, H., R. Temam, P. Moin, and J. Kim (1993), Feedback control for unsteady flow and its application to the stochastic burgers equation, *Journal of Fluid Mechanics*, *253*, 509–543.
- Choi, H., P. Moin, and J. Kim (1994), Active turbulence control for drag reduction in wall-bounded flows, *Journal of Fluid Mechanics*, *262*, 75–110.
- Chow, F. K., B. Kosovic, and S. Chan (2008), Source inversion for contaminant plume dispersion in urban environments using building-resolving simulations, *Journal of applied meteorology and climatology*, *47*(6), 1553–1572.
- Deser, C., J. E. Walsh, and M. S. Timlin (2000), Arctic sea ice variability in the context of recent atmospheric circulation trends, *Journal of Climate*, *13*(3), 617–633.
- Evans, L. C. (2005), An introduction to mathematical optimal control theory, *Lecture Notes, University of California, Department of Mathematics, Berkeley*.
- Fattorini, H., and S. Sritharan (1992), Existence of optimal controls for viscous flow problems, *Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences*, *439*(1905), 81–102.



- Gorelick, S. M., B. Evans, and I. Remson (1983), Identifying sources of groundwater pollution: An optimization approach, *Water Resources Research*, 19(3), 779–790.
- Hammond, E., T. Bewley, and P. Moin (1998), Observed mechanisms for turbulence attenuation and enhancement in opposition-controlled wall-bounded flows, *Physics of Fluids (1994-present)*, 10(9), 2421–2423.
- Hayes, M. (2009), *Statistical Digital Signal Processing And Modeling*, Section 9.4 pp., Wiley India Pvt. Limited.
- Hellman, B., E. Gylfe, E. Grapengiesser, H. Dansk, and A. Salehi (2007), Insulin oscillations—clinically important rhythm. antidiabetics should increase the pulsative component of the insulin release, *Läkartidningen*, 104(32-33), 2236.
- Houpis, C., and G. Lamont (1992), *Digital control systems: theory, hardware, software*, MCGRAW HILL SERIES IN ELECTRICAL AND COMPUTER ENGINEERING, McGraw-Hill.
- Jha, M., and B. Datta (2012), Three-dimensional groundwater contamination source identification using adaptive simulated annealing, *Journal of Hydrologic Engineering*, 18(3), 307–317.
- Jianyong, L., and Z. Yu (1996), Digital active ear defender, in *Signal Processing, 1996., 3rd International Conference on*, vol. 2, pp. 1687–1690, IEEE.
- Kailath, T. (1980), *Linear Systems*, Information and System Sciences Series, Prentice-Hall.
- Kalman, R. E., and J. E. Bertram (1960), Control system analysis and design via the second method of lyapunov: Icontinuous-time systems, *Journal of Fluids Engineering*, 82(2), 371–393.
- Katopodes, N. D. (2009), Control of sudden releases in channel flow, *Fluid Dynamics Research*, 41(065002), 24.
- Khalil, H. K. (1995), *Nonlinear System*, second edition ed., 81–84 pp., Prentice Hall.
- Kirby, B. J. (2010), *Micro-and nanoscale fluid mechanics: transport in microfluidic devices*, Cambridge University Press.
- Lee, C., J. Kim, D. Babcock, and R. Goodman (1997), Application of neural networks to turbulence control for drag reduction, *Physics of Fluids (1994-present)*, 9(6), 1740–1747.
- Li, R., M. A. Henson, and M. J. Kurtz (2004), Selection of model parameters for off-line parameter estimation, *Control Systems Technology, IEEE Transactions on*, 12(3), 402–412.

- Lieu, T., C. Farhat, and M. Lesoinne (2006), Reduced-order fluid/structure modeling of a complete aircraft configuration, *Computer Methods in Applied Mechanics and Engineering*, 195(41 - 3), 5730 - 5742, doi: <http://dx.doi.org/10.1016/j.cma.2005.08.026>, John H. Argyris Memorial Issue. Part {II}.
- Lumley, J. L. (1967), The Structure of Inhomogeneous Turbulent Flows, in *Atmospheric turbulence and radio propagation*, edited by A. M. Yaglom and V. I. Tatarski, pp. 166-178, Nauka, Moscow.
- Mahar, P. S., and B. Datta (1997), Optimal monitoring network and ground-water-pollution source identification, *Journal of water resources planning and management*, 123(4), 199-207.
- Mazutis, L., J. Gilbert, W. L. Ung, D. A. Weitz, A. D. Griffiths, and J. A. Heyman (2013), Single-cell analysis and sorting using droplet-based microfluidics, *Nature protocols*, 8(5), 870-891.
- Meditch, J. S. (1964), On the problem of optimal thrust programming for a lunar soft landing, *Automatic Control, IEEE Transactions on*, 9(4), 477-484.
- Mirza, A., and S. Hussain (2000), Robust controller for nonlinear & unstable system: Inverted pendulum, *ADVANCES IN MODELLING AND ANALYSIS-C-*, 55(3/4), 49-I.
- Monaghan, J. J. (1992), Smoothed particle hydrodynamics, *Annual review of astronomy and astrophysics*, 30, 543-574.
- Piasecki, M., and N. D. Katopodes (1997), Control of contaminant release in rivers. i: Adjoint sensitivity analysis, *Journal of Hydraulic Engineering*, 123, 486-492.
- Post, S. (2011), *Applied and Computational Fluid Mechanics*, 227 pp., Jones and Bartlett Publishers.
- Shijie, X., and Z. Jianfeng (2007), A new strategy for lunar soft landing, *The Journal of the Astronautical Sciences*, 55(3), 373-387.
- Soros, G. (2008), *The new paradigm for financial markets: the credit crisis of 2008 and what it means*, Basic Books.
- Strogatz, S. (2014), *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering*, Studies in Nonlinearity, Westview Press.
- Teh, S.-Y., R. Lin, L.-H. Hung, and A. P. Lee (2008), Droplet microfluidics, *Lab on a Chip*, 8(2), 198-220.
- Williams, R., and D. Lawrence (2007), *Linear State-Space Control Systems*, John Wiley & Sons.

Wu, R., and N. D. Katopodes (2006), Control of chemical spills by boundary suction, in *Proceedings of the 2006 WSEAS/IASME International Conference on Fluid Mechanics*, pp. 109–114.

Yeh, H.-D., T.-H. Chang, and Y.-C. Lin (2007), Groundwater contaminant source identification by a hybrid heuristic approach, *Water Resources Research*, 43(9).