Data Assimilation of Mobile Sensors in Hydrological Models

A Master Thesis
Presented by

Mr. Affan

In Fullfilment
of the Requirements for the Degree of
Master of Science in Electrical Engineering

Supervisor: Abubakr Muhammad (LUMS)
Co-supervisor: Basit Shafiq(LUMS)
External-supervisor: Hasan Arshad Nasir(NUST)

Syed Babar Ali School of Science and Engineering
Lahore University of Management Sciences
May 2019

Acknowledgment

In the name of God, the most kind and most merciful. I am thankful to my family, who supported me throughout this work all the time. I dedicate this work to my late parents, May their souls rest in peace.

The research work has been led by the Center for Water Informatics & Technology (WIT) and Cloud Computing Research Lab (CCRL), LUMS, Pakistan. It is partially sponsored by DAAD grant DyMaSH: "Dynamic Mapping and Sampling for High Resolution Hydrology"; and partially by Higher Education Commission (HEC), Pakistan under the project grant, HEC NRPU Grant Project number 2305.

I am grateful to have the guidance and support of Dr. Abubakr Muhammad, Dr. Basit Shafiq and Dr. Hasan Arshad Nasir. To encourage me to give my full effort and dedication to the work. I am thankful for their discussions on the critical and important issues. The knowledge provided by them in regards of my work is immense and very fruitful. They encouraged me to have innovate and new ideas regarding my field and work.

I am very thankful to Allah Almighty, who bestowed his blessings over me and helped me through all my ups and downs. May He keep His blessings over us to provide our services for the goodness of humanity.

This Master thesis has been examined by a Committee of the Department of Electrical Engineering as follows:

Dr. Abubakr Muhammad ....
Thesis Supervisor
Associate Professor of Electrical Engineering

Dr. Basit Shafiq ....
Thesis Co-Supervisor
Associate Professor of Computer Science

Dr. Hasan Arshad Nasir ....
Thesis External-Supervisor
Assistant Professor of Electrical Engineering

Abstract

In this research work, the estimation of the spatio-temporal variation of water bodies for state variables, velocity (m/s) and water surface elevation (m) for unsteady flows in open channels has been investigated. For data assimilation, two methods are studies, one is using conventional way to incorporate velocity data in system model and other one is incorporating GPS locations in to augmented model, the GPS locations are obtained from mobile sensors such as Lagrangian sensors, which have the ability to float passively in water bodies. One-dimensional Saint-Venant equations are used for a system model linearized by a Taylor series expansion. To obtain a discrete-time state-space model, the coupled PDEs are discretized by Lax diffusive method in time and space. For state estimation of the open channel, a Kalman filter is set up with suitable filtering parameters for the channel's model. In this research studies, the augmented system model is developed to incorporate the GPS location. For data assimilation in augmented model, the state dependent interacting multiple model (SD-IMM) is implemented as system model is time varying. Eulerian (fixed) sensors present at the head and tail of the canal provide the minimally required boundary conditions to run the model. The trajectory of float is also estimated using water velocity profiles as well as GPS locations. The system is simulated using HEC-RAS simulation software. The estimated states are compared with actual values. The system is also tested in real environment.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Water is one the most valuable resource in nature, which is used in household for cooking, washing and other daily life application as well as in industry for almost every kind of production site either it is steel industry or leather industry. As the need for water is increasing in the world, the issues related to water are also increasing at an unimaginable rate. As the population of the world grows and societies shift towards urbanization, the demand for water is increasing for agriculture and domestic usage. On the other hand, due to the development of industrial society, the water bodies has become a major sink for industrial waste. The major motivation behind this research is the dumping of untreated industrial waste into water bodies. According to the United Nations, the amount of 1500 Km^3 untreated wastewater is produced each year in the world, which is six times the total water in the river all across the world. In recent years, drinking water has become a major issue in Pakistan. The water availability per capita in 1951 was 5000 m^3 which decreased to 1200 m^3 in 2000. The estimated water availability per capita in 2025 is 659 m^3[8]. The 20 to 40 percent of all disease is due to contaminated drinking water in Pakistan[3], which results in an income loss of Rs. 25-58 billion annually and these figures are increasing tremendously. One of the major reason behind the contaminated drinking water is the lack of monitoring of industrial wastewater, which is being dumped into water bodies. One such location is the hydyara drain in Lahore, Pakistan, which is sink to a large number of industries, which are constructed along it. The current water and health issues can be resolved by using proper equipment and scientific approach. To solve issues of contaminated water, the first step is to monitor and understand the behavior of contamination transportation and its effects, which requires knowledge of hydrodynamic parameters of water bodies. This research study is a contribution to monitoring the hydrodynamic parameters such as water level and water velocity, which can be further used to visualize the water flows and sediment transport in the water bodies.

1.2 Related Work

State estimation is an important part of the study for the control of water bodies. The efficient use of water requires improved methods of monitoring water bodies at very high spatio-temporal scales. In the recent past, monitoring techniques have shifted from manual observation to the use of ICT powered sensor networks [15] as part of the wider hydro-informatics revolution [12]. The data is typically collected at specific locations using in-situ sensors at high frequencies, which enables the modeling and study of the temporal phenomenon at unprecedented scales. Similarly, such telemetry systems have also enabled a revolution in water management, allowing the enforcement of governance principles such as equity, transparency and water rights in complex river basins [20]. The data is typically assimilated in dynamical models, allowing higher temporal scales and more accurate estimates than what raw measurements can promise [6]. However, many scientific applications and water governance issues demand access to high-resolution data not only at a specific location but also at multiple spatial points within a water body. Examples include the monitoring of waste disposal in drains, unauthorized water diversion from irrigation channels and contamination spread in wide rivers.

In the present era, new technologies have opened the possibility of exploiting mobility as an enabler to expand the coverage of in-situ sensors. In a hydrological setting, a very important class of mobile sensors are the so-called Lagrangian sensors that can float passively in water bodies and provide information on their position using GPS. These sensors are cost efficient as they provide wide coverage exploiting the natural movement of water and can perform simultaneous measurement of multiple variables related to water quality and quantity [1]. Another type of mobile sensing which is relevant to the framework used in this work (but not used here as a case study) is social sensing, in which the mobile sensor is a human that can observe water-related variables such as the height of water at different locations and share the values by using social media as the communication medium [11]. The common challenge with the use of mobile sensors is that these sensors provide asynchronous data at different steps in time and space. The water dynamics of water bodies including water level and water velocity can be estimated using data assimilation of sensor data in the 1-D or 2-D hydrological models. However, there is always some error in the mathematical model due to uncertainty in model parameters and selection of boundary conditions [10]. The boundary conditions are obtained by using the so-called Eulerian sensors deployed at the location of gates, which are fixed sensors deployed at fixed locations in the water body of interest. The data assimilation can be done using multiple techniques related to optimization and statistical models of uncertainty. The researchers working in the field of control methods for water bodies has estimated the hydrodynamic variables by using data assimilation of average velocity data, which is obtained by using basic relation $S = Vt$ based on GPS location from Lagrangian sensors in hydrological models [13], The computation of average velocity is the pre-processing on sensor data. In the field of estimation, the pre-processing on sensor data which alters the type of data is not encouraged. In literature, the state estimation of hydrodynamic parameters is also done using quadratic programming, for which cost function is defined [17]. To incorporate the GPS sensor measurement in the model, the modified system is required as the location of the Lagrangian sensor also becomes a state of the system. In the research study [9], an augmented system model is proposed, in which the sensor motion model is augmented with the hydrological model. The augmented system model requires improved data assimilation method, as the system becomes time-dependent. In this research study, the state dependent interacting multiple models (SD-IMM) inspired by the motion of ballistic missiles along the Kalman filter is used. In SD-IMM, all the possible models are executed at the same time with appropriate probabilities.

1.3 Summary

In this study, our focus is on understanding dynamic flow conditions in open channels similar to irrigation canals [5]. For the mathematical model, the Saint-Venant partial differential equations are linearized by using appropriate linearized technique around the steady-state values by using backwater curve steady state equations. The Taylor series expansion is used for the linearization of one-dimensional Saint-Venant equation, despite the fact that water bodies are nonlinear by nature, a linearized mathematical model is expected to work efficiently for the setup of this research study. As Saint-Venant equations are coupled partial differential equations [5], so for discretization the Lax diffusive method is used in space and time with suitable time step and spatial step. The discretized system is converted into a state space model [16]. The Lagrangian sensor motion model is developed for one-dimensional motion, which is augmented with Saint-Venant state space model. For the measurement model, the vector of the three outputs is considered, which consist of Lagrangian sensor position, water elevation of first and last cell obtained from Eulerian sensors mounted at both ends. Position measurements are provided by a Lagrangian sensor (GPS powered float) released from upstream. As the system is time-dependent, the state dependent interacting multiple model technique along the Kalman filter is used. In this research study, states estimation of hydrodynamic parameters is done by two ways, (1) using augmented model along SD-IMM with GPS measurement (2) using Saint-Vaint model along simple Kalman filter with average velocity as a measurement.

Chapter 2

System Model

2.1 Saint-Venant Equations

The water bodies can be mathematically explained by Saint-Venant equations [5]. These equations can be derived by using the law of conversation of mass and the law of conservation of momentum. For one dimensional flow as shown in Fig. (2-1), let us assume that a control volume of fluid in a water body has Q the rate of

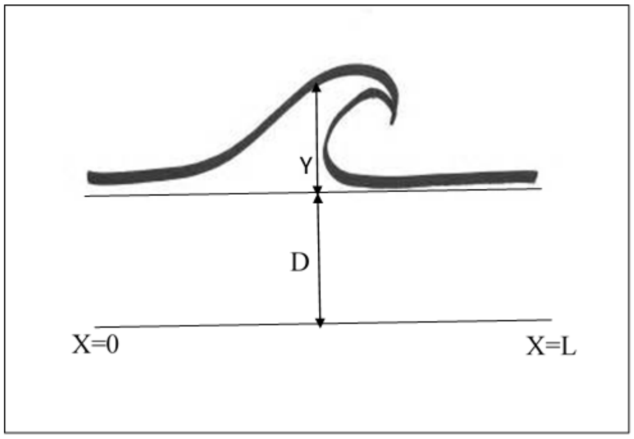

Figure 2-1: control volume fluid in a reach

discharge(m^3/s), V the flow velocity (m/s), Y the water level (m), D the depth of channel (m), x the length of reach (m), t the time (s), B the extensive property (Mass or Momentum of water) and β the intensive property. The x is measured positive along downward stream and negative along upward stream. Let assume the control volume in a section of water body as shown in Fig. (2-2). First by law of conservation

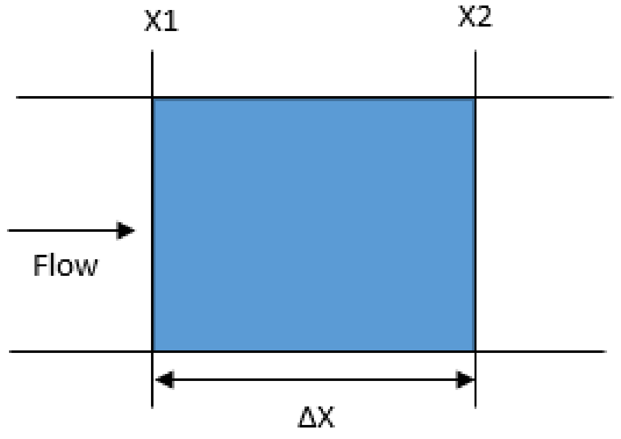

Figure 2-2: Control volume fluid in a reach

of mass, let the extensive property $B = M$, where M is the mass:

$$\frac{\partial M}{\partial t} = 0. \tag{2.1}$$

By Reynolds transport theorem:

$$\frac{\partial M}{\partial t} = \frac{\partial}{\partial t}\int_{x_1}^{x_2} \beta\rho\partial V + (\beta\rho Q)_{out} - (\beta\rho Q)_{in} \tag{2.2}$$

Where A is the flow area of fluid within the control volume and ρ is the mass density of water. The inflow is considered positive and outflow as negative. The system is considered to be closed contour or no sink without any lateral inflow. As intensive property $\beta = \frac{\Delta m}{\Delta m} = 1$, the Equation (2.2) can be written as:

$$\frac{\partial M}{\partial t} = \frac{\partial}{\partial t}\int_{x_1}^{x_2} \rho\partial v + (\rho Q)_{out} - (\rho Q)_{in} \tag{2.3}$$

$$\frac{\partial M}{\partial t} = \frac{\partial}{\partial t}\int_{x_1}^{x_2} \rho A\partial x + \rho Q_2 - \rho Q_1 - \rho q_l(x_2 - x_1) \tag{2.4}$$

Where q_l is the lateral inflow. For the in-compressible fluid the ρ is constant for all constant cross sections.

$$\frac{\partial M}{\partial t} = \frac{\partial}{\partial t}\int_{x_1}^{x_2} A\partial x + Q_2 - Q_1 - q_l(x_2 - x_1) \tag{2.5}$$

if $q_l = 0$, then mass is conserved and if it is not zero, it act as sink or source. Let lateral inflow $q_l = 0$,

$$\frac{\partial M}{\partial t} = \frac{\partial}{\partial t}\int_{x_1}^{x_2} A\partial x + Q_2 - Q_1 \tag{2.6}$$

As the Equation (2.6) is in integral form, we have to convert it into differential form. The differential form can be obtained by using Leibnitz rule, which is shown in Equation (2.7).

$$\frac{\partial}{\partial t}\int_{f_1(t)}^{f_2(t)} F(x,t)\partial x = \int_{f_1(t)}^{f_2(t)} \frac{\partial}{\partial x}F(x,t)\partial x + F(f_2(t),t)\frac{\partial f_2}{\partial t} - F(f_1(t),t)\frac{\partial f_1}{\partial t} \tag{2.7}$$

Now driving the differential form of the system. By Equation (2.6) and by Leibnitz rule:

$$\int_{x_1}^{x_2} \frac{\partial A}{\partial t}\partial x + (Q_2 - Q_1) = 0 \tag{2.8}$$

$$(x_2 - x_1)\frac{\partial A}{\partial t} + Q_2 - Q_1 = 0 \tag{2.9}$$

$$\frac{\partial A}{\partial t} + \frac{Q_2 - Q_1}{(x_2 - x_1)} = 0 \tag{2.10}$$

$$\frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} = 0 \tag{2.11}$$

The above equation is one-dimensional Saint-Venant equation. Now converting this equation into water level and velocity form. As the water level and velocity are the state variables, which are being observed in this research work. The flow of fluid in a reach length can be written in terms of flow area and velocity of fluid as follow:

$$Q = AV \tag{2.12}$$

For a uniform channel, the change in flow area ∂A can be written as follows:

$$\partial A = w\partial Y \tag{2.13}$$

Where w is the free surface width.

$$\frac{\partial A}{\partial x} = w\frac{\partial Y}{\partial x} \tag{2.14}$$

So equation (2.11) can be written as follows:

$$w\frac{\partial Y}{\partial t} + \frac{\partial (AV)}{\partial x} = 0 \tag{2.15}$$

$$w\frac{\partial Y}{\partial t} + Vw\frac{\partial A}{\partial x} + A\frac{\partial V}{\partial x} = 0 \tag{2.16}$$

$$\frac{\partial Y}{\partial t} + V\frac{\partial Y}{\partial x} + \frac{A}{w}\frac{\partial V}{\partial x} = 0 \tag{2.17}$$

Now substitute $D = \frac{A}{w}$ in Equation (2.17), where D is the channel depth:

$$\frac{\partial Y}{\partial t} + V\frac{\partial (Y)}{\partial x} + D\frac{\partial V}{\partial x} = 0 \tag{2.18}$$

This equation is the first equation of model used in this study based on law of conservation of mass. Now deriving the equation for law of conservation of momentum. Let extensive property $B =$ momentum of water$= mV$ and intensive property $\beta = V\frac{\Delta m}{\Delta m} = V$

By Newton's law:

$$\frac{\partial B}{\partial t} = \sum F \tag{2.19}$$

Again by Reynolds transport theorem:

$$\frac{\partial B}{\partial t} = \sum F = \frac{\partial}{\partial t}\int_{x_1}^{x_2} \rho A V \partial x + V_2 \rho A_2 V_2 - V_1 \rho A_1 V_1 - V_x \rho q_l (x_2 - x_1) \tag{2.20}$$

Let $q_l = 0$,

$$\sum F = \frac{\partial}{\partial t}\int_{x_1}^{x_2} \rho Q \partial x + V_2 \rho Q_2 - V_1 \rho Q_1 \tag{2.21}$$

By using Leibnitz rule,

$$\sum F = \int_{x_1}^{x_2} \frac{\rho \partial Q}{\partial t}\partial x + V_2 \rho Q_2 - V_1 \rho Q_1 \tag{2.22}$$

By solving the integral,

$$\sum F = (x_2 - x_1)\frac{\rho \partial Q}{\partial t} + \rho V_2 Q_2 - \rho V_1 Q_1 \tag{2.23}$$

$$\sum F = (x_2 - x_1)\frac{\rho \partial Q}{\partial t} + \rho (V_2 Q_2 - V_1 Q_1) \tag{2.24}$$

$$\frac{\sum F}{(x_2 - x_1)} = \frac{\rho \partial Q}{\partial t} + \rho \frac{(V_2 Q_2 - V_1 Q_1)}{(x_2 - x_1)} \tag{2.25}$$

$$\frac{\sum F}{\partial x} = \rho \frac{\partial Q}{\partial t} + \rho \frac{\partial (QV)}{\partial x} \tag{2.26}$$

$$\frac{\sum F}{\rho \partial x} = \frac{\partial Q}{\partial t} + \frac{\partial (QV)}{\partial x} \tag{2.27}$$

For simplicity, assume that the shear stress on flow surface due to wind and the effect of corollas acceleration is negligible. These are valid assumptions. The pressure

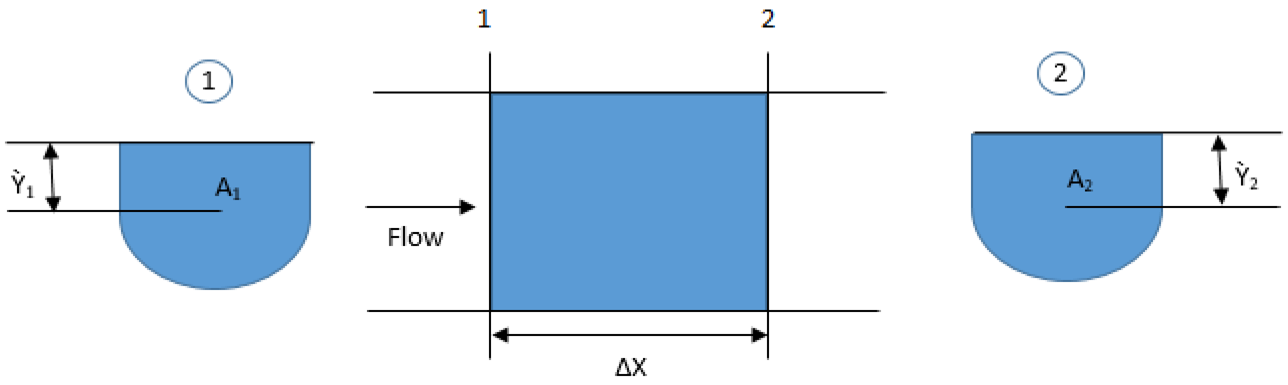

Figure 2-3: Upstream cross section (1) and downstream cross section (2) with centroid depth of $\bar{y}_1$ and $\bar{y}_2$

force acting on the upstream for the centroid depth $\bar{y}_1$ for A_1 is as follows:

$$F_1 = \rho g A_1 \bar{y}_1 \tag{2.28}$$

Where g is the gravitation force which is $9.8 m/s^2$ similarly the pressure force acting on the downstream end is:

$$F_2 = \rho g A_2 \bar{y}_2 \tag{2.29}$$

The force due to the weight of water in the direction of x can be written as:

$$F_3 = \rho g \int_{x_1}^{x_2} A S_b \partial x \tag{2.30}$$

S_b is the channel bed slop. The frictional force or the shear force between channel sides, bottom and water is:

$$F_4 = \rho g \int_{x_1}^{x_2} A S_f \partial x \tag{2.31}$$

Where S_f is the frictional slope which can be calculated as follows:

$$S_f = CV \frac{|V|^{m-1}}{R^p} \tag{2.32}$$

The C, R and p depends on the channel structure.
Now the sum of all all forces can be written as:

$$\sum F = F_1 + F_2 + F_3 + F_4 \tag{2.33}$$

$$\sum F = \rho g A_1 \bar{y}_1 - \rho g A_2 \bar{y}_2 + \rho g \int_{x_1}^{x_2} A S_b \partial x - \rho g \int_{x_1}^{x_2} A S_f \partial x \tag{2.34}$$

$$\sum F = \rho g (A_1 \bar{y}_1 - A_2 \bar{y}_2) + \rho g \int_{x_1}^{x_2} A (S_b - S_f) \partial x \tag{2.35}$$

$$\sum F = \rho g(A_1\bar{y}_1 - A_2\bar{y}_2) + \rho g(x_2 - x_1)A(S_b - S_f) \tag{2.36}$$

$$\frac{\sum F}{\rho(x_2 - x_1)} = \frac{g(A_1\bar{y}_1 - A_2\bar{y}_2)}{(x_2 - x_1)} + gA(S_b - S_f) \tag{2.37}$$

By re-arranging and applying central limit theorem the Equation (2.37) can be written as:

$$\frac{\sum F}{\rho \partial x} = \frac{-g\partial(A\bar{y})}{\partial x} + gA(S_o - S_f) \tag{2.38}$$

Now by Equation (2.27) and Equation (2.38):

$$\frac{\partial Q}{\partial t} + \frac{\partial(QV)}{\partial x} = -g\frac{\partial A\bar{y}}{\partial x} + gA(S_b - S_f) \tag{2.39}$$

$$\frac{\partial Q}{\partial t} + \frac{\partial(QV + gA\bar{y})}{\partial x} = gA(S_b - S_f) \tag{2.40}$$

As,

$$\frac{\partial gA\bar{y}}{\partial x} = g\frac{\partial A\bar{y}}{\partial y}\frac{\partial y}{\partial x} = gA\frac{\partial y}{\partial x} \tag{2.41}$$

Using above expression in Equation (2.40),

$$\frac{\partial Q}{\partial t} + \frac{\partial(QV)}{\partial x} + gA\frac{\partial y}{\partial x} = gA(S_b - S_f) \tag{2.42}$$

$$\frac{\partial AV}{\partial t} + \frac{\partial(QV)}{\partial x} + gA\frac{\partial y}{\partial x} = gA(S_b - S_f) \tag{2.43}$$

$$V\frac{\partial A}{\partial t} + A\frac{\partial V}{\partial t} + V\frac{\partial (Q)}{\partial x} + Q\frac{\partial (V)}{\partial x} + gA\frac{\partial y}{\partial x} = gA(S_b - S_f) \tag{2.44}$$

$$Vw\frac{\partial Y}{\partial t} + A\frac{\partial V}{\partial t} + V\frac{\partial (AV)}{\partial x} + AV\frac{\partial (V)}{\partial x} + gA\frac{\partial Y}{\partial x} = gA(S_b - S_f) \tag{2.45}$$

$$Vw\frac{\partial Y}{\partial t} + A\frac{\partial V}{\partial t} + VA\frac{\partial (V)}{\partial x} + VV\frac{\partial (A)}{\partial x} + AV\frac{\partial (V)}{\partial x} + gA\frac{\partial Y}{\partial x} = gA(S_b - S_f) \tag{2.46}$$

$$Vw\frac{\partial Y}{\partial t} + A\frac{\partial V}{\partial t} + VA\frac{\partial (V)}{\partial x} + VVw\frac{\partial (Y)}{\partial x} + AV\frac{\partial (V)}{\partial x} + gA\frac{\partial Y}{\partial x} = gA(S_b - S_f) \tag{2.47}$$

$$V(w\frac{\partial Y}{\partial t} + A\frac{\partial V}{\partial x} + wV\frac{\partial Y}{\partial x}) + A(\frac{\partial V}{\partial t} + V\frac{\partial V}{\partial x} + g\frac{\partial Y}{\partial x}) = gA(S_b - S_f) \tag{2.48}$$

$$V(w\frac{\partial Y}{\partial t} + A\frac{\partial V}{\partial x} + wV\frac{\partial Y}{\partial x}) + A(\frac{\partial V}{\partial t} + V\frac{\partial V}{\partial x} + g\frac{\partial Y}{\partial x} - g(S_b - S_f)) = 0 \tag{2.49}$$

The first term will be equal to zero due to law of conservation of mass.

$$A(\frac{\partial V}{\partial t} + V\frac{\partial V}{\partial x} + g\frac{\partial Y}{\partial x} - g(S_b - S_f)) = 0 \tag{2.50}$$

$$\frac{\partial V}{\partial t} + V\frac{\partial V}{\partial x} + g\frac{\partial Y}{\partial x} - g(S_b - S_f) = 0 \tag{2.51}$$

This is the Saint-Venant equation for conservation of momentum of fluid. The Equations (2.51) and (2.18) are known as Saint-Venant equations. These equations are non-linear and continuous. To implement the data assimilation, the state space models including linearization and discretization is discussed in next section.

2.2 State Space Model for Hydrological Systems

2.2.1 Linearization

The one dimensional Saint-Venant equation in the form of Equations (2.16) and (2.51) are nonlinear. For linearization, multiple techniques are available which can be implemented. The most prominent technique is the Taylor series expansion as shown in Equation (2.52).

$$f(x+\bar{x}) = f(x) + \sum_{n=1}^{\infty} \frac{(\bar{x})^n}{n} \frac{\partial^n f}{\partial x^n} \tag{2.52}$$

One of the important points is that in some of the techniques only perturbations are considered as output. However, in the control system, the perturbation must be added in steady states to analyze the characteristics of the non-linear system. For linearization, the steady-state values are required, which are obtained by using backwater curve steady state equations written as follows:

$$\frac{\mathrm{d}\bar{V}(x)}{\mathrm{d}x} = -\frac{\bar{V}(x)}{\bar{Y}(x)}\frac{\mathrm{d}\bar{Y}(x)}{\mathrm{d}x} - \frac{\bar{V}(x)}{w(x)}\frac{\mathrm{d}w(x)}{\mathrm{d}x}, \tag{2.53}$$

$$\frac{\mathrm{d}\bar{Y}(x)}{\mathrm{d}x} = \frac{S_b - S_f}{1 - F(x)^2}. \tag{2.54}$$

For this research study, the first order perturbation is considered, neglecting higher order terms of Taylor series. The water level and water velocity for first order perturbation is given as follows:

$$Y = \bar{Y} + y \tag{2.55}$$

$$V = \bar{V} + v \tag{2.56}$$

The linearized form of the Saint-Venant equation is as follows:

$$\frac{\partial y}{\partial t} + \bar{Y}(x)\frac{\partial v}{\partial x} + \bar{V}(x)\frac{\partial y}{\partial x} + \alpha(x)v + \beta(x)y = 0 \tag{2.57}$$

$$\frac{\partial v}{\partial t} + \bar{V}(x)\frac{\partial v}{\partial x} + g\frac{\partial y}{\partial x} + \gamma(x)v + \eta(x)y = 0 \tag{2.58}$$

The α, β, γ and η can be written as:

$$\alpha(x) = \frac{\mathrm{d}\bar{Y}}{\mathrm{d}x} + \frac{\bar{Y}}{w}\frac{\mathrm{d}\bar{w}}{\mathrm{d}x}, \tag{2.59}$$

$$\beta(x) = -\frac{\bar{V}}{Y}\frac{\mathrm{d}\bar{Y}}{\mathrm{d}x} - \frac{\bar{V}(x)}{w(x)}\frac{\mathrm{d}\bar{w}}{\mathrm{d}x}, \tag{2.60}$$

$$\gamma(x) = 2gm^2\frac{\left|\bar{V}\right|}{\bar{Y}^{4/3}} - \frac{\bar{V}}{\bar{Y}}\frac{\mathrm{d}\bar{Y}}{\mathrm{d}x} - \frac{\bar{V}(x)}{w(x)}\frac{\mathrm{d}w(x)}{\mathrm{d}x}, \tag{2.61}$$

$$\eta(x) = -\frac{4}{3}gm^2\frac{\bar{V}\left|\bar{V}\right|}{\bar{Y}^{7/3}}. \tag{2.62}$$

2.2.2 Discretization

The linearized system of equations is discretized in space as well as time. There are multiple techniques are available for discretization, most prominent techniques are grid discretization by finite difference method and Lax diffusive method. The finite difference method can be used with forward, central or backward Euler method. For this research work, Lax diffusive method is used. The Equations (2.63) and (2.64) shows the working of Lax diffusive method.

$$\frac{\partial f}{\partial t} = \frac{f_i^{k+1} - \frac{1}{2}\left(f_{i+1}^k + f_{i-1}^k\right)}{\Delta t} \tag{2.63}$$

$$\frac{\partial f}{\partial x} = \frac{f_i^{k+1} - f_{i-1}^k}{2\Delta t} \tag{2.64}$$

The discretized form of these system equations is shown in Equation (2.65) and (2.66) as follows:

$$\begin{aligned} y_i^{k+1} = y_{i+1}^k &\left[\frac{1}{2} - \frac{\Delta t}{4\Delta x}(\bar{v}_{i+1} + \bar{v}_{i-1}) - \frac{\Delta t}{2}\beta_{i+1}\right] \\ +y_{i-1}^k &\left[\frac{1}{2} + \frac{\Delta t}{4\Delta x}(\bar{v}_{i+1} + \bar{v}_{i-1}) - \frac{\Delta t}{2}\beta_{i-1}\right] \\ +v_{i+1}^k &\left[-\frac{\Delta t}{4\Delta x}(\bar{y}_{i+1} + \bar{y}_{i-1}) - \frac{\Delta t}{2}\alpha_{i+1}\right] \\ +v_{i-1}^k &\left[\frac{\Delta t}{4\Delta x}(\bar{y}_{i+1} + \bar{y}_{i-1}) - \frac{\Delta t}{2}\alpha_{i-1}\right], \end{aligned} \tag{2.65}$$

$$\begin{aligned} v_i^{k+1} = v_{i+1}^k &\left[\frac{1}{2} - \frac{\Delta t}{4\Delta x}(\bar{v}_{i-1} + \bar{v}_{i+1}) - \frac{\Delta t}{2}\gamma_{i+1}\right] \\ +v_{i-1}^k &\left[\frac{1}{2} + \frac{\Delta t}{4\Delta x}\bar{(}v_{i+1} - \bar{v}_{i-1}) - \frac{\Delta t}{2}\gamma_{i-1}\right] \\ +y_{i+1}^k &\left[-g\frac{\Delta t}{2\Delta x} - \frac{\Delta t}{2}\eta_{i+1}\right] \\ +y_{i-1}^k &\left[g\frac{\Delta t}{2\Delta x} - \frac{\Delta t}{2}\eta_{i-1}\right]. \end{aligned} \tag{2.66}$$

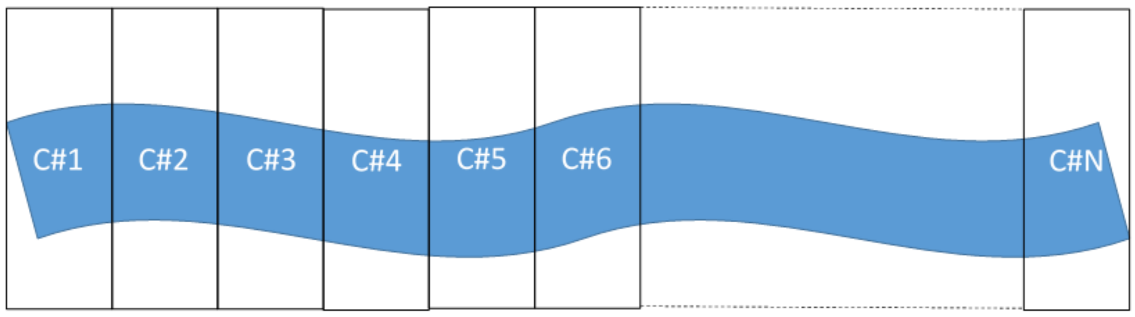

Figure 2-4: Discretization of water body in cells

The physical interpretation of discretization is shown in Fig. (2-4). For the stability of the system, the Courant-Friedrich-Lewy (CFL) condition must be satisfied. The condition is as follows,

$$\frac{\Delta t}{\Delta x}\left|V\right| \leq 1. \tag{2.67}$$

2.2.3 Boundary Conditions

The boundary conditions for the water elevation at both ends of channel are given as follows:

$$y_1^k = h_1^k, \tag{2.68}$$

$$y_N^k = h_N^k. \tag{2.69}$$

The boundary conditions for water velocities are calculated by using overshot-gate/weir equation [19] as shown in Equations (2.70) and (2.71).

$$v(x=0,t) = \frac{0.6\sqrt{g}T}{y_1 w}(y_1 - p)^{3/2}, \tag{2.70}$$

$$v(x=L,t) = \frac{0.6\sqrt{g}w}{y_N w}(y_N - p)^{3/2}, \tag{2.71}$$

where T is the gate width, w channel width, y_1 water level in first cell and y_N water level in last cell. y is approximated at $t = k\Delta t$, v at $x = \Delta x$ and N is the number of cells. The undershot gate equations are as follows:

$$v(t) = C_d\sqrt{2gy_u} \tag{2.72}$$

$$C_d = \frac{C_c}{\sqrt{1 + C_c(p/y_u)}} \tag{2.73}$$

$$C_c = y_d/p, \tag{2.74}$$

where y_u is the upstream water level of the undershot gate and y_d is the downstream water level of the gate. These boundary values are obtained by using Eulerian sensors at both ends. The state space model is derived with y and v as state variables [16]. The general state space model is given by:

$$x(k+1) = Ax(k) + Bu(k) + W(k), \tag{2.75}$$

$$y(k) = C(k)x(k) + Du(k) + r(k). \tag{2.76}$$

The state vector for the above model is as follows:

$$x(k) = \left[v_2^k \dots v_{N-1}^k y_2^k \dots y_{N-1}^k\right]^T. \tag{2.77}$$

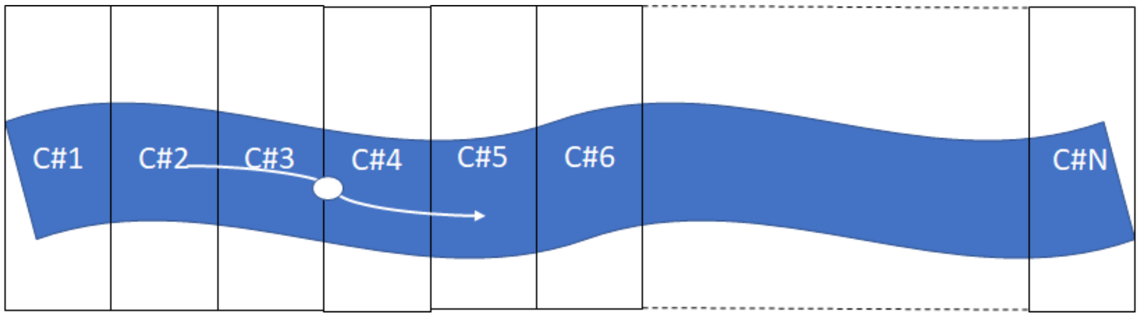

Figure 2-5: Movement of Lagrangian sensor water body across cells

The input vector u is given by:

$$u(k) = \begin{bmatrix} v_1^k \\ v_N^k \\ y_1^k \\ y_N^k \end{bmatrix}. \tag{2.78}$$

The boundary conditions are used as input to the state space model. The process noise $W(k)$ is modeled by i.i.d. Gaussian random variables. The dimensions of matrix A is $(2N-4)\times(2N-4)$ while dimension of B is $(2N-4)\times(4)$. The details of these matrices are omitted for brevity but can be derived easily from the discretization scheme above.

2.3 Lagrangian Sensor Motion Model

The Lagrangian sensor has the capability to float passively along the water body. These sensors provides GPS location at each spatial step and temporal step. The motion of Lagrangian sensor is based on the velocity of water as shown in Fig. (2-5). The water velocity varies in each cell of water body, which effects the motion of Lagrangian sensor. Let pos be the position of sensor, v velocity of water in a particular cell i, Δt the time step for sensor measurement. The motion model for

Lagrangian sensor can be written as follows,

$$pos(k) = pos(k-1) + v_i(k)\Delta t. \tag{2.79}$$

2.4 Augmented Model

As the Lagrangian sensors provide only the GPS location, it is common practice to compute the velocity of Lagrangian sensor by $S = V\Delta t$ relation. As it is not an appropriate way to assimilate sensor data into system model. In this way, sensor data is pre-processed which act as another estimator. In order to assimilate the GPS location directly to system model and preserve the application of Lagragian sensor, an augmented model is required. For this purpose, the Lagrangian sensor motion model is developed in the previous section. As discussed above that states space model consist of only water level and water velocities, now the position of Lagrangian sensor is also included in state vector. The state space model defined in Equation (2.75) is modified as follows,

$$x(k+1) = \begin{bmatrix} 1 & 1(i)t \\ 0 & A \end{bmatrix} x(k) + Bu(k) \tag{2.80}$$

The state vector for modified model is as follows:

$$x(k) = \begin{bmatrix} pos(k) v_2^k \dots v_{N-1}^k y_2^k \dots y_{N-1}^k \end{bmatrix}^T. \tag{2.81}$$

The augmented input vector u is given by:

$$u(k) = \begin{bmatrix} v_1^k \\ v_N^k \\ y_1^k \\ y_N^k \end{bmatrix}. \tag{2.82}$$

The state transition matrix and input transition matrix defined in Equation (2.80) and (2.81) are time dependent.

2.5 Measurement Model

For measurement model, the model described in equations (2.83), (2.84) and (2.85) of three outputs is considered.

$$Y_1^k = h_2^k, \tag{2.83}$$

$$Y_2^k = v_n^k, \tag{2.84}$$

$$Y_3^k = h_{N-1}^k, \tag{2.85}$$

Where n is the number of cell in which the Lagrangian sensor is moving. C is $3\times(2N-4)$ matrix in which the first and last row of the matrix will remain the same because water level at 2nd and N-1 point is being measured by Eulerian sensors at fixed locations. The middle row will change as sensor moves along the channel, providing the velocity of the each cell.

Chapter 3

Sensors

3.1 Lagrangian Sensors

The mobile sensors used in our work are called Lagrangian sensors, also popularly known as drifters or floats. These sensors passively float with water velocity in water body. Lagrangian sensors are usually equipped with multiple modalities of sensing such as temperature, pH, salinity, turbidity and other physico-chemical parameters. An important component of Lagrangian sensors is GPS to provide position at each step. Sensors for our work are inspired by drifters developed by our group and reported in [1], where we have deployed such sensors to observe water quality data in canals and rivers. A photograph of our sensor deployed in a canal in Pakistan is show in Fig. (3-1). The GPS measurements can be used to estimate the velocity of the drifter and thereby of a steady channel. However, due to the inherent spatial variation in fluid flow due to variation in channel geometry and the occasional deviation of the sensor from forward movements, a simple method such as averaging does not yield good results in unsteady flows, when the channels are unstructured or when the measurements may be intermittent.

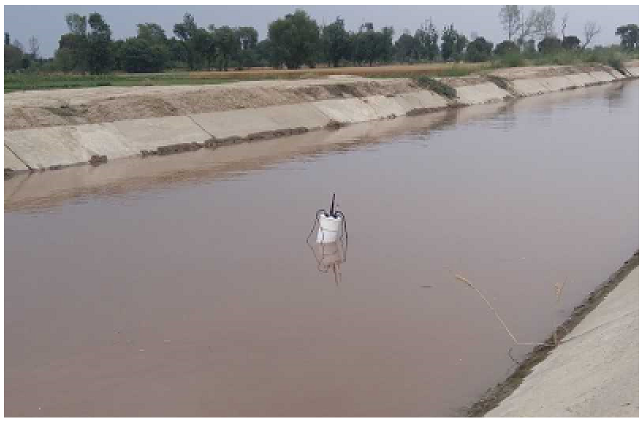

Figure 3-1: Lagrangian sensor equipped with GPS floating in a canal.

3.2 Eulerian Sensors

The static sensors in this work are Eulerian and also inspired from sensors developed and deployed by our group [14]. These sensors measure water height using ultrasonic or radar based ranging. Eulerian sensors provide data at regular time intervals at a specific location. From a modeling perspective, the sensor model is time-invariant and does not provide directly measured data away from its location. The accuracy of these sensors is usually high and are not prone to the type of errors present in Lagrangian sensors. An field setting of such a sensor providing water height and derived flow is shown in Fig. (3-2).

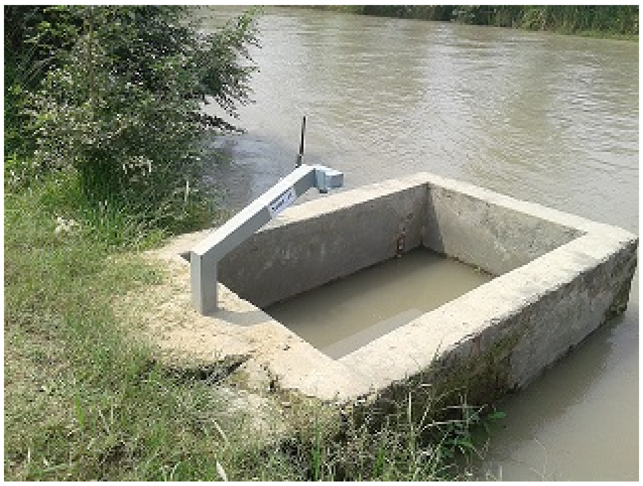

Figure 3-2: Eulerian sensor providing the data of water level in canal.

Chapter 4

Data Assimilation

Data assimilation is the process of estimation of the unknown states or some of the unknown states of the system by combining the measurements received from sensors and model[18]. Whereas, numerical solutions of mathematical models may not provide the exact solution due to uncertainty in modeling but covers the whole area of observation. By combining the model and measurements from sensor leads to better estimation of the values. There are many data assimilation techniques being practiced. Some of the techniques are variational data assimilation techniques, filtering methods, statistical assimilation and Newtonian relaxation. The most common used data assimilation technique is Kalman filter.

4.1 Kalman filter

Kalman filter has been considered as one of optimal filter for the purposes of tracking, estimation and smoothing. The Kalman filter is based on minimum mean square error technique[2]. The Kalman filter mainly includes three steps: prediction, Kalman gain calculation and update. The Equations (4.1) and (4.2) show the prediction of state vector and error covariance matrix.

$$x(k+1) = Ax(k) + Bu(k), \tag{4.1}$$

$$P^{'}(k+1) = AP(k)A^{T} + Q, \tag{4.2}$$

The equation (4.3) shows the computation of Kalman gain.

$$K(k) = P^{'}(k)H^{T}(HP^{'}(k)H^{T} + R)^{-1}. \tag{4.3}$$

The equations (4.4) and (4.5) show the process to update state vector and error covariance matrix.

$$x(k) = x^{'}(k) + K(k)(z(k) - Cx^{'}(k)), \tag{4.4}$$

$$P(k) = (I - K(k)H)P^{'}(k). \tag{4.5}$$

The Q is the covariance matrix of model noise as follows:

$$Q = E[w(k)w(k)^{T}], \tag{4.6}$$

where w is the model noise. The R is the covariance matrix of sensor noise, which can be written as follows:

$$R = E[r(k)r(k)^{T}], \tag{4.7}$$

where r is the measurement or sensor noise. The P is the covariance matrix of error e, which can be written as:

$$P = E[e(k)e^{T}(k)]. \tag{4.8}$$

4.1.1 Filter Performance Analysis

The performance of filter used in data assimilation is a critical point for the validation of results. As it is possible that visually cyber system is tracking the actual physical system but it is the states are not correct. To validate the performance of filter, some method is required. We can't measure the performance by the state error measurement because actual states are not available. So the performance of filter is measured by the output error, which is computed during filtering known as innovation in Kalman filter. There are two ways to check the performance of Kalman filter, which are:

- To check that error is consistent with its covariance by verifying that the magnitude of the innovation is bounded by $\pm\sqrt{2S_k}$.
- In order to test the unbiasness we calculate the normalised innovation squared q_{k+1} for k trials of Kalman filter. The q_{k+1} is computed as follows:

  $$q_{k+1} = e_{k+1} S_{k+1}^{-1} e_{k+1} \tag{4.9}$$

  where S is the innovation covariance matrix computed during Kalman filter execution.
- Perform auto-correlation at error vector.

4.2 State Dependent Interacting Multiple Models (SD-IMM)

The approach to use multiple filter in parallel is practised in the area of target tracking[7], in which target does not follow a straight line motion model [4]. So the tracking of target is difficult using single motion model. In this research study, the Lagrangian sensor is referred as target. In order to get good estimate of the target maneuver states, run multiple models in parallel and assign probabilities to each model as shown in Fig. (4-1).

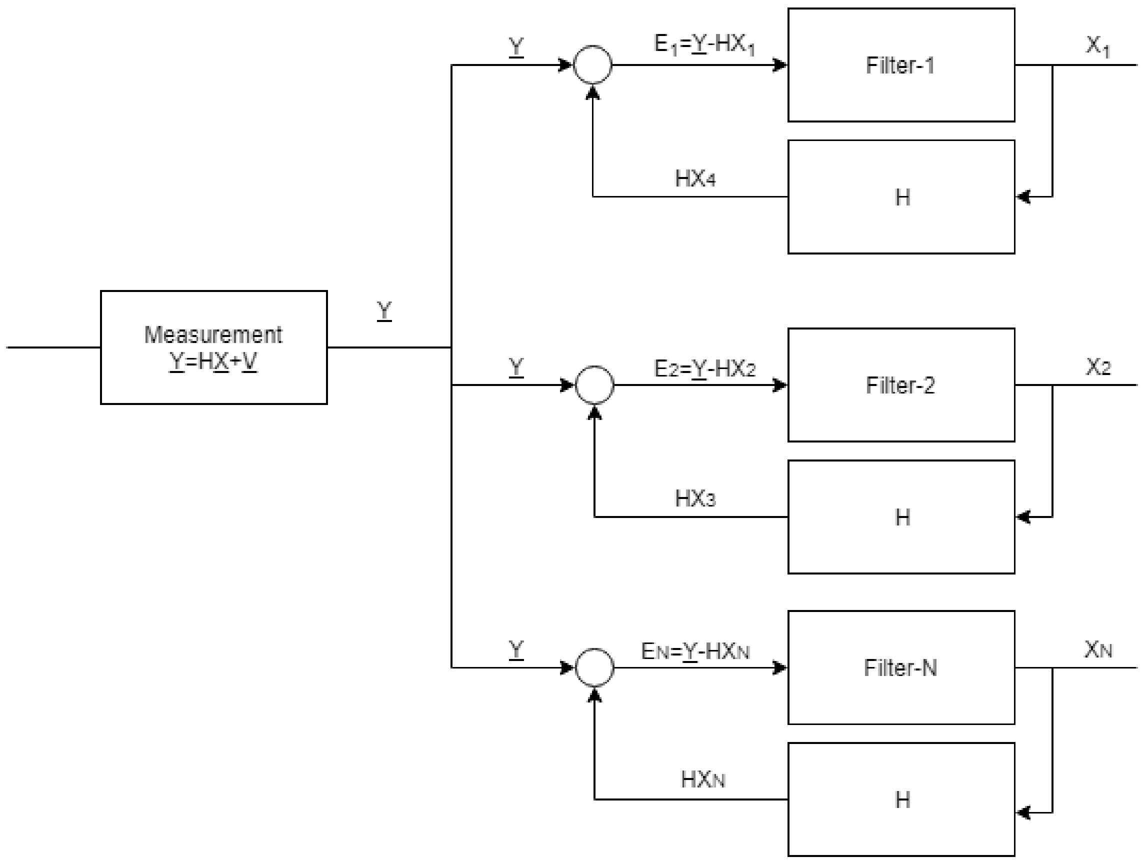

Figure 4-1: N parallel models filter bank

These probabilities are based on the likelihood of the models to be executed at a particular time based on Bayes's rule and residuals. These probabilities are updated with time. This process is also known as Markove chain. The output of these parallel models is weighted with probabilities. To implement the Markove chain, at each iteration, there is a transition probability matrix P_{ij} that the Lagrangian sensor has moved from ith cell to jth cell, so jth model will have more probability. The number of models depend on the number of cells. The matrix P_{ij} is defined as follows:

$$P_{ij} = \begin{bmatrix} P_{1,1} & P_{1,2} & \dots & P_{1,N} \\ P_{2,1} & P_{2,2} & \dots & P_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ P_{N,1} & P_{N,1} & \dots & P_{N,N} \end{bmatrix} \tag{4.10}$$

The matrix P_{ij} is defined perior of the iterations. The sum of probabilities of all models in each row should be equal to 1.

$$\sum_{j=1}^{N} P_{ij} = 1 \tag{4.11}$$

The flow chart in Fig. (4-2) shows the complete procedure for clear understanding of IMM technique.

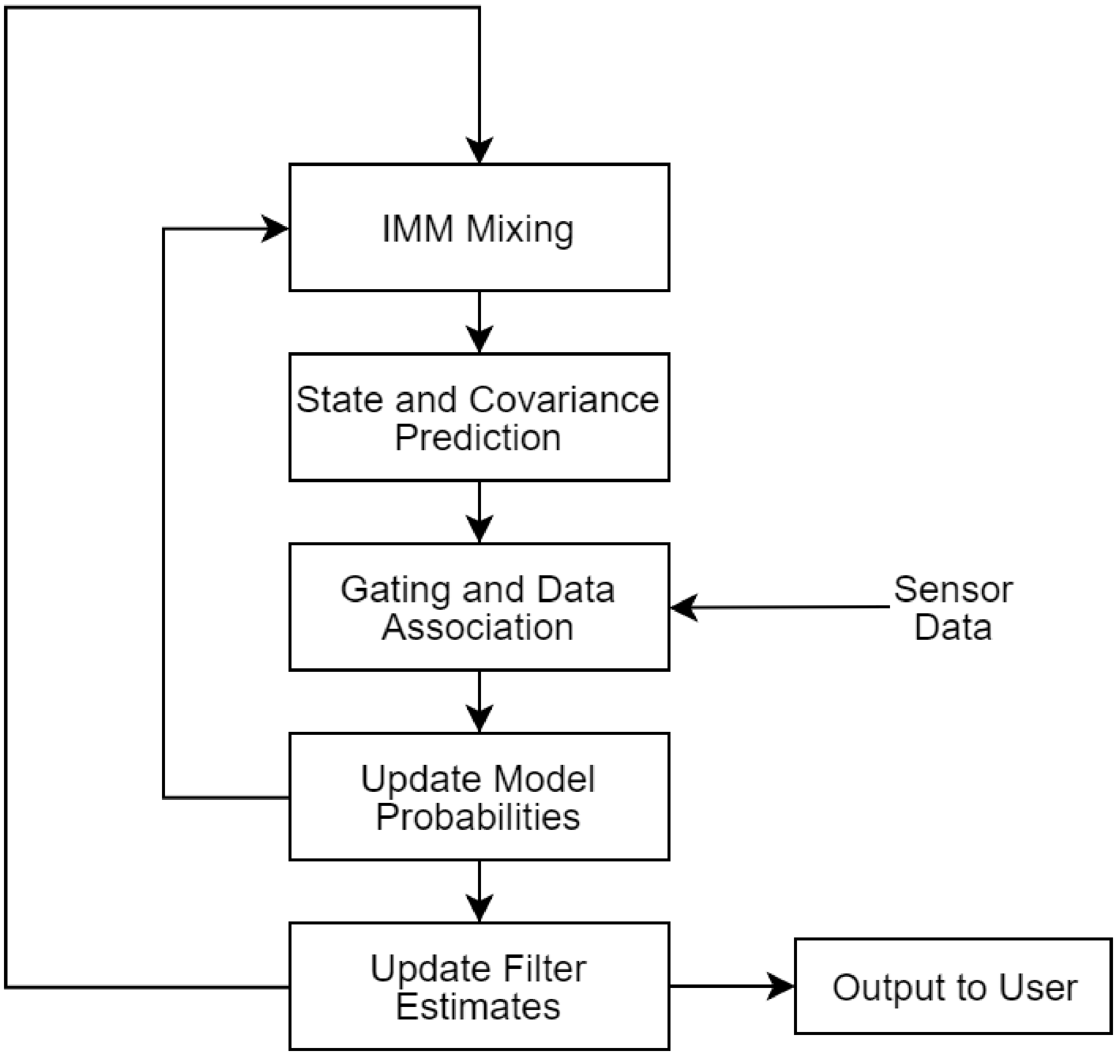
IMM Mixing
State and Covariance Prediction
Gating and Data Association
Sensor Data
Update Model Probabilities
Update Filter Estimates
Output to User

Figure 4-2: IMM flowchart

4.2.1 IMM Mixing

IMM mixing is the process to track Kalman filter estimated states and covariance matrix according to the transition probabilities that the Lagrangian sensor makes a transition. To understand the IMM mixing process, following terms are necessary to understand.

$\hat{x}_i(k-1|k-1)$ = Estimated states from Kalman filter at $k-1$ by model i

$P_i(k-1|k-1)$ = Covariance matrix from Kalman filter at $k-1$ by model i

$\mu_i(k-1)$ =Probability that target is in model i

$\mu_{ij}(k-1)$ =Transition probability that target made transition from ith model to jth model

The μ_{ij} is calculated as follows:

$$\mu_{ij}(k-1) = P_{ij}\mu_i(k-1)/C_j(k-1) \tag{4.12}$$

The C_j is the probability of jth model after transition, which is calculated as follows:

$$C_j(k-1) = \sum_{i=1}^{N} P_{ij}\mu_i(k-1) \tag{4.13}$$

After the computation of transition probabilities, the mixing process produces new filtered state estimates and covariance matrices, which are calculated as follows:

$$x_j^0(k-1|k-1) = \sum_{i=1}^{N} \mu_{ij}(k-1)\hat{x}_i(k-1|k-1) \tag{4.14}$$

$$P_j^0(k-1|k-1) = \sum_{i=1}^{N} \mu_{ij}(k-1)[P_i(k-1|k-1) + DP_{ij}(k-1)] \tag{4.15}$$

The DP_{ij} is the increment term, which is calculated as follows:

$$DP_{ij}(k-1|k-1) = Dx_{ij}(Dx_{ij})^T \tag{4.16}$$

$$Dx_{ij}(k-1) = \hat{x}_i(k-1|k-1) - \hat{x}^0_j(k-1|k-1) \tag{4.17}$$

The next step is prediction of states and covaraince matrix by using system model for time step k.

4.2.2 Gating and Data Association

As the result of prediction step, each track will have estimated state vector and covariance matrix for time step k. The next step is the gating and data association. In this process the estimates are multiplied and add at for each model. There are two alternatives as follows:

- Predict and combine
- Combine and predict

The gating and data association works fine in both above mentioned ways. The equations used in gating and data association are as follows:

$$\hat{x}(k|k-1) = \sum_{j=1}^{N} C_j(k-1)\hat{x}_j(k|k-1) \tag{4.18}$$

$$\hat{P}(k|k-1) = \sum_{j=1}^{N} C_j(k-1)\hat{P}_j(k|k-1) \tag{4.19}$$

The gating and data association will lead to the observation to track assignment. In most of application of IMM, the sensor data is not assigned to all models while in this study sensor data is assigned to all models as Lagragian can be anywhere in the channel so it is important to assign the sensor data to all of the models.

4.2.3 Model Probabilities Calculation

The updated model probabilities are calculated using Baye's rule as follows:

$$\mu_i(k) = \Lambda_i(k)C_i(k-1)/C \tag{4.20}$$

where $\Lambda_i(k)$ is the likelihood function for the measurement of ith model and C is the normalizing constant, which are calculated as follows:

$$C = \sum_{j=1}^{N} \Lambda_j(k)C_j(k-1) \tag{4.21}$$

$$\Lambda_i(k) = \frac{exp[-d^2(k)/2]}{\sqrt{(2\pi)^M \left|S_i(k)\right|}} \tag{4.22}$$

where d^2 is the distance of an observation to track assignment, which is calculated as follows:

$$d^2 = \tilde{y}^T S^{-1} \tilde{y} \tag{4.23}$$

The $\tilde{y}$ is the innovation between sensor output and model output, the S is the innovation covariance matrix. The following Kalman filter equations are used to calculate these quantities.

$$\tilde{y} = y(k) - H\hat{x}(k|k-1) \tag{4.24}$$

$$S(k|k-1) = HP(k|k-1)H^T + R \tag{4.25}$$

Chapter 5

Simulations

5.0.1 Simulation Scenario

The channel of length 2640 m is simulated for this research work. The channel is discretized into 11 equal cells. The Eulerian sensors are considered at both ends to provide the water elevation at regular time interval. The single drifter is considered for this study, which moves along the channel and provide the data about water velocity at each time and spatial step. The simulation scenario used for this study is shown in Fig. (5-1). The constant surface width of 5 m is considered for the rectangular channel.

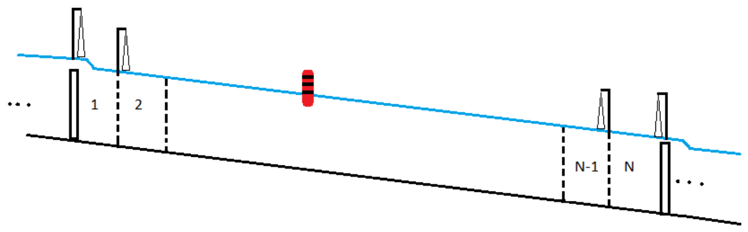

Figure 5-1: Simulation setup for a channel of 2640 meter length with 5 meter wide cross-section.

5.0.2 Model Simulation

The state space model of the system described in section-II is simulated in MATLAB. The parameters used in this simulation are given in table (5.1). The steady state

Table 5.1: Simulation parameters for system model.

Parameters	Values
Channel depth (m)	3
Channel width (m)	5
Channel length (m)	2640
No. of cells	11
Cell step (m)	240
Time step (s)	60

values for linearized state space system model, which are calculated by backwater curve steady state equations, are shown in Fig. (5-2). The model simulation results are

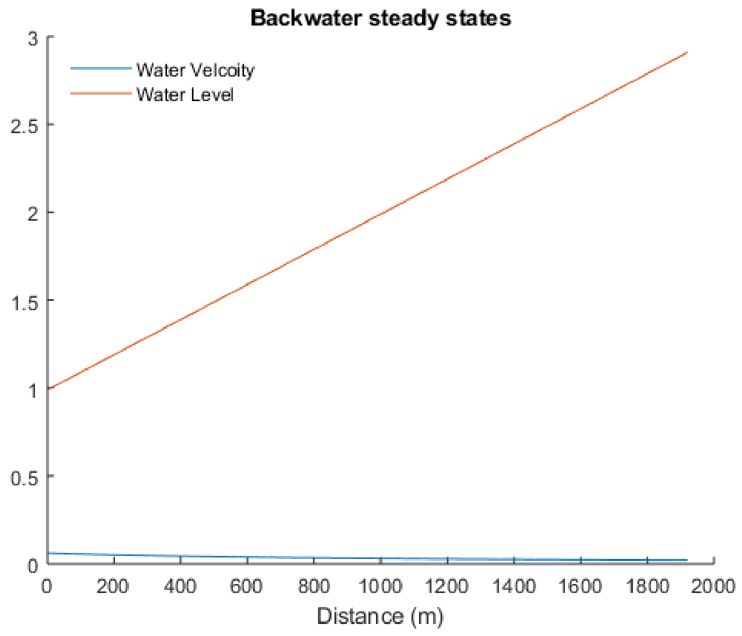

Figure 5-2: The steady state values for Saint-Venant model, calculated by backwater curve steady state equations.

shown in Fig. (5-3). The model results from 20 minutes to 50 minutes are important as during this time period, the water level in the first cell increases showing the increase in water flow at upstream end, showing the physical phenomenon of opening a gate.

The water elevation in the channel remains constant till increase in water elevation in

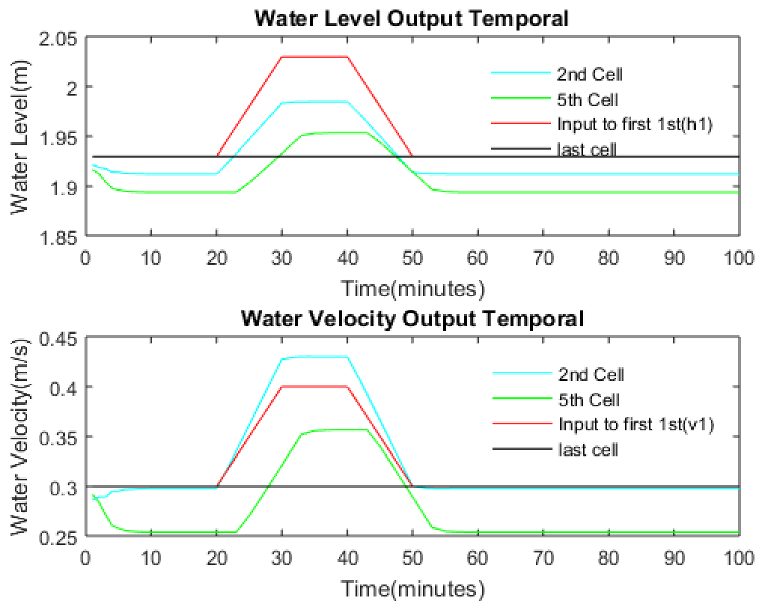

Figure 5-3: The states of water level and velocity 1st cell, 2nd cell, 5th cell and last cell.

1st cell at 20 minutes of simulation after that the wave starts propagating in channel and increases water level of other cells as well. The water elevation in all cells gets to the minimum value again at 50 minutes. The velocity of channel starts increasing at 20 minutes of simulation as well. The decrease in velocity values shows the backwater effect or natural slow velocity of water in the channel. As the water velocity and water level in last cell is constant as boundary conditions are constant in last cell.

5.0.3 HEC-RAS Simulation

For data assimilation, the system values are generated by HEC-RAS simulation software. HEC-RAS is the River Analysis Software by Hydrological Engineering Center, USA. This software simulates the hydraulic systems close to the real environment and solves 1-D/2-D Saint-Venant equations according to user given specifications. The cross section of simulated channel in HEC-RAS is showed in Fig. (5-4). The parameters used for HEC-RAS simulation are given in table. (5.2). The system in HEC-RAS is simulated for unsteady flow. For the upstream cross-section, a stage hydrograph is

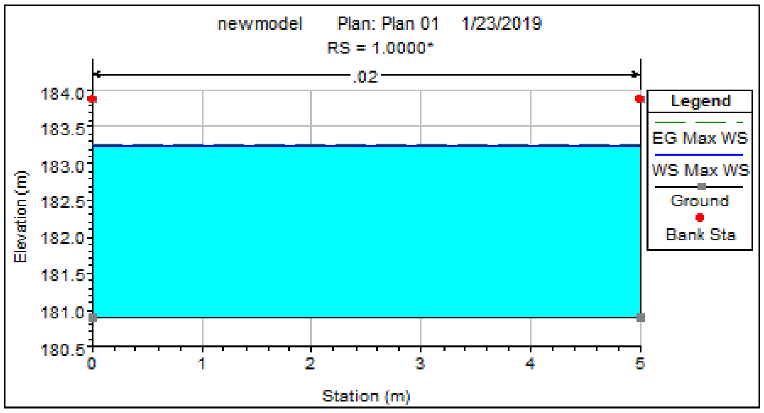

Figure 5-4: Cross-section of a channel with rectangular geometry in HEC-RAS.

used as boundary condition as shown in Fig. (5-5). The normal depth is considered for downstream end as boundary condition. The normal depth as boundary condition is suitable for an open end channel. Under normal depth boundary conditions, the water level at boundaries is calculated on the basis of flow at each temporal and spatial step by using manning equation. For the better data generation the MATLAB model results are compared with HEC-RAS generated results. The comparison is shown in Fig. (5-6) The results of water level and water velocity in HEC-RAS are similar to MATLAB results.

Table 5.2: Simulation parameters for system model in HEC-RAS.

Parameters	Values
Channel depth (m)	3
Channel width (m)	5
Channel edges height from sea level (m)	183.88
Channel bed height from sea level (m)	180.88
Channel length (m)	2640
No. of cells	11
Cell step (m)	240
Time step (s)	60
Simulation time (minutes)	100

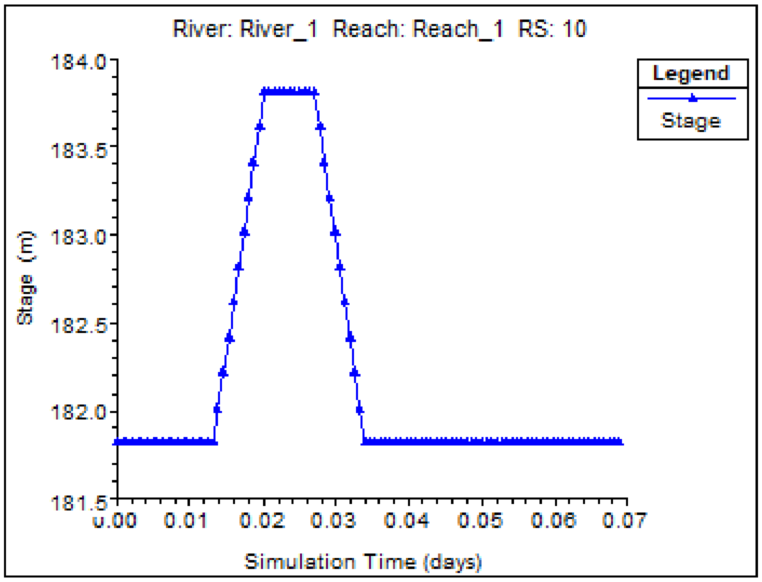

Figure 5-5: Stage hydrograph as upstream boundary condition in HEC-RAS for 100 minutes.

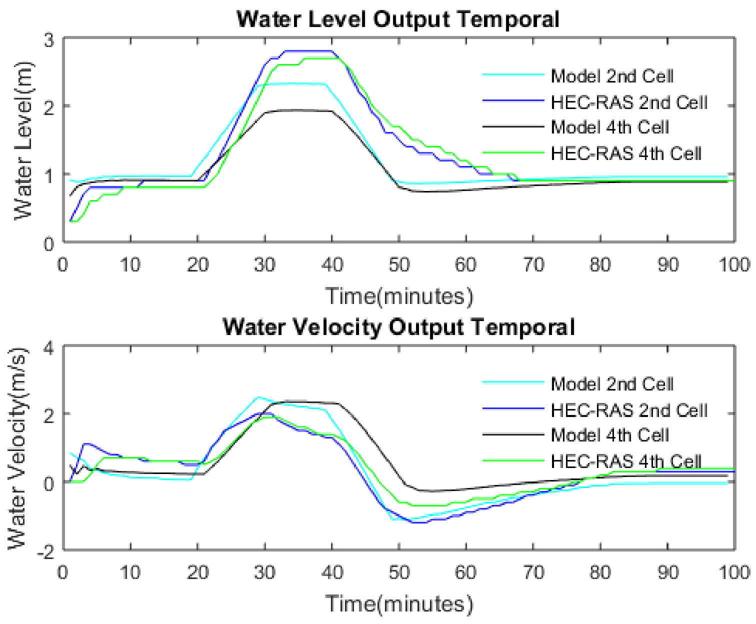

Figure 5-6: Comparison between MATLAB and HEC-RAS data for water elevation and velocity.

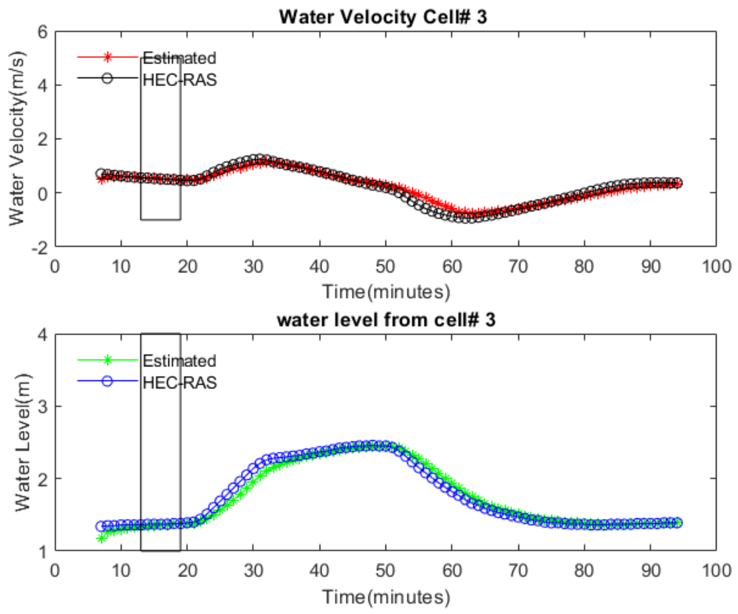

Figure 5-7: The estimated states of water level and water velocity in cell number 3 along with the true values from HEC-RAS.

5.0.4 State Estimation using Velocity Data

For the task of data assimilation, the Kalman filter is used which is explained in chapter-IV. The Kalman filter is performed for simulation time of 100 minutes as Lagrangian sensors covers the 2270 meter in 100 minutes along the flow of water, during this time the drifter covers all 10 cells. The values of velocity from each cell are incorporated in the model. The Fig. (5-7) shows estimated states of cell number 3 along the comparison with actual HEC-RAS values. The estimated states of cell number 6 along the comparison with actual HEC-RAS values are shown in Fig. (5-8). The estimated water level and water velocity in cell number 9 with comparison to HEC-RAS data is shown in Fig. (5-9).

In results, the rectangular box shows the movement of Lagrangian sensor. The most critical part of a model for data assimilation is the boundary conditions. The boundary conditions of velocity and water elevation are shown in Fig. (5-18) The Kalman filter estimated the sates with significant low error for both the velocity and water elevation. The path followed by Lagrangian sensor is also calculated by using

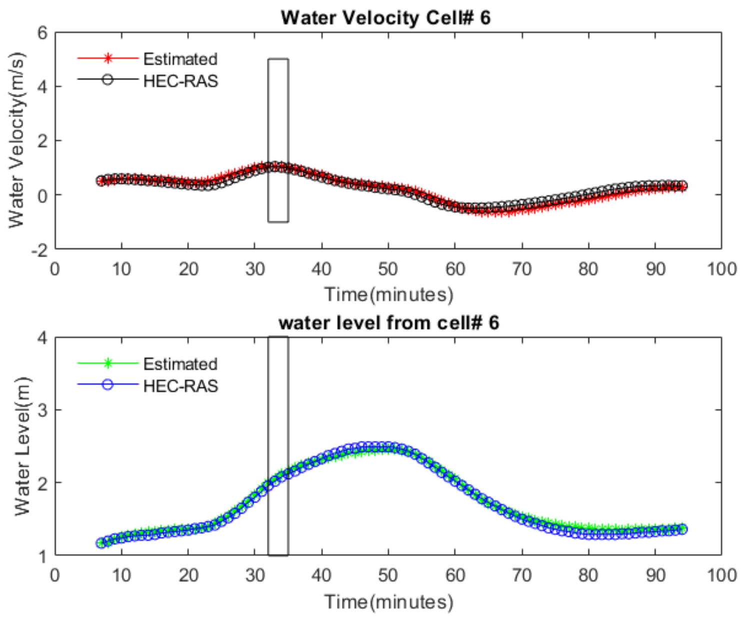

Figure 5-8: The estimated states of water level and water velocity in cell number 6 along with the true values from HEC-RAS.

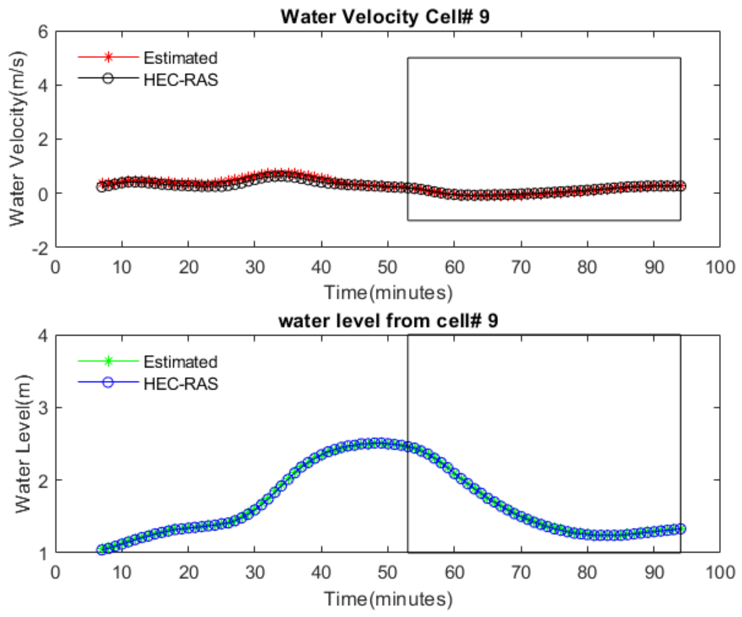

Figure 5-9: The estimated states of water level and water velocity in cell number 9 along with the true values from HEC-RAS.

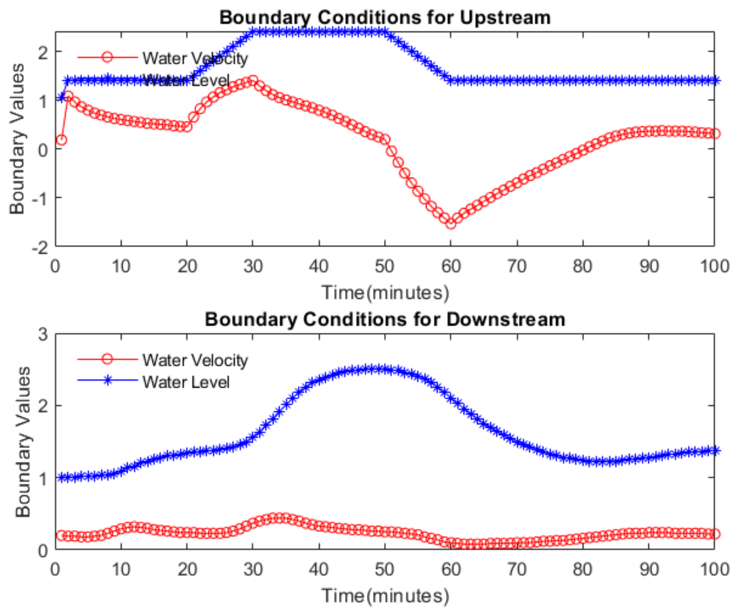

Figure 5-10: The boundary conditions of 1st and last cell for data assimilation.

information of velocity from HEC-RAS and appropriate time step. The path followed by Lagrangian sensor is shown in Fig. (5-11). To analyze the performance of the Kalman filter in this scenario, the Chi-Squared error analysis and auto-correlation in error is performed. The performance is shown in Fig. (5-12) and (5-13). The chi-squared analysis a significant low error values. It is visible that the chi squared error increases as the input water flow is increased which is the phenomenon of gates opening. The auto-correlation error analysis shows the correlation between error at each time step with other time steps. It is prominent that the error correlation is increased during the gate opening, which results the unsteady behavior of hydrological systems.

5.0.5 State estimation using Position Data

For the task of data assimilation using position data, the state dependent interacting multiple model(SD-IMM) along Kalman filter is used which is explained in chapter-IV. The data assimilation is performed for simulation time of 100 minutes as Lagrangian sensors covers the 2270 meter in 100 minutes along the flow of water, during this time

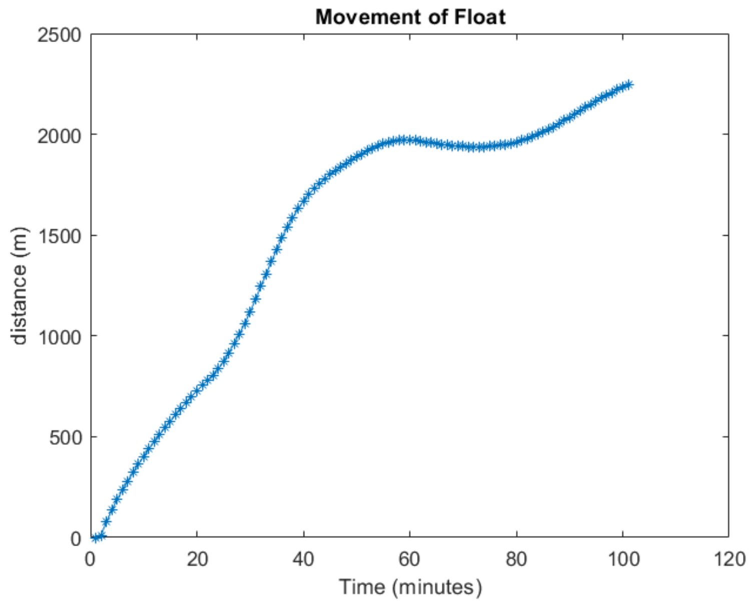

Figure 5-11: The path followed by Lagrangian sensor (Float).

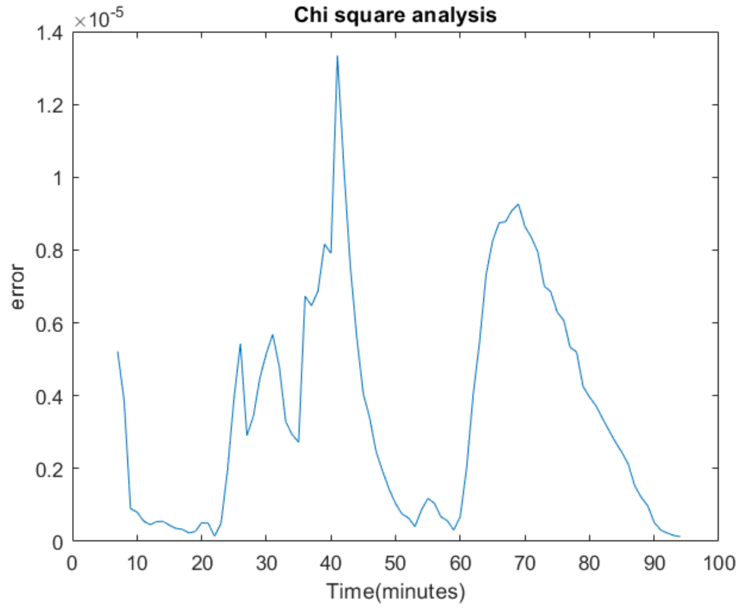

Figure 5-12: The Chi-Squared analysis for the state estimation using Kalman filter

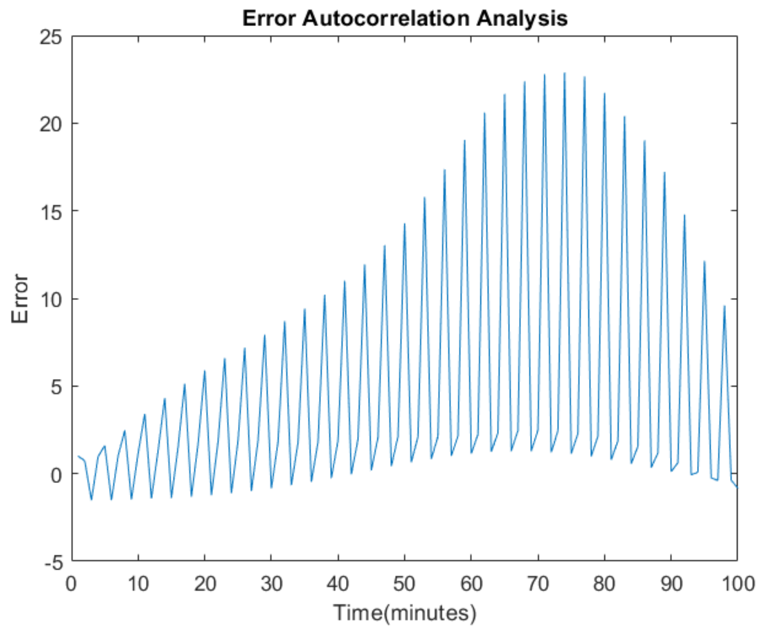

Figure 5-13: The Auto-correlation error analysis for the state estimation using Kalman filter

the drifter covers all 10 cells. The values of velocity from each cell are incorporated in the model. The Fig. (**??**) shows estimated states of cell number 2 along the comparison with actual HEC-RAS values. The estimated states of cell number 5 along the comparison with actual HEC-RAS values are shown in Fig. (5-15). The estimated water level and water velocity in cell number 6 with comparison to HEC-RAS data is shown in Fig. (5-17).

The estimated water level and water velocity in cell number 9 with comparison to HEC-RAS data is shown in Fig. (**??**).

In results, the rectangular box shows the movement of Lagrangian sensor. The most critical part of a model for data assimilation is the boundary conditions. The boundary conditions of velocity and water elevation are shown in Fig. (5-18) The Kalman filter estimated the sates with significant low error for both the velocity and water elevation. The path followed by Lagrangian sensor is also estimated. The path followed by Lagrangian sensor is shown in Fig. (5-19). To analyze the performance of the Kalman filter in this scenario, the minimum mean square error(MMSE) error

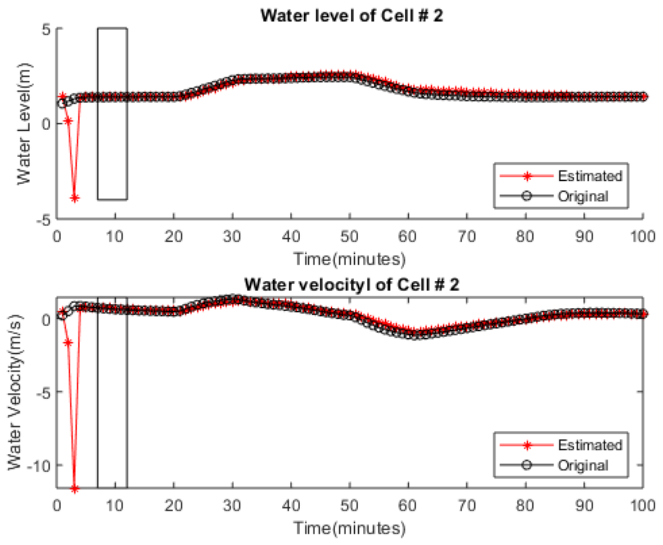

Figure 5-14: The estimated states of water level and water velocity in cell number 2 along with the true values from HEC-RAS.

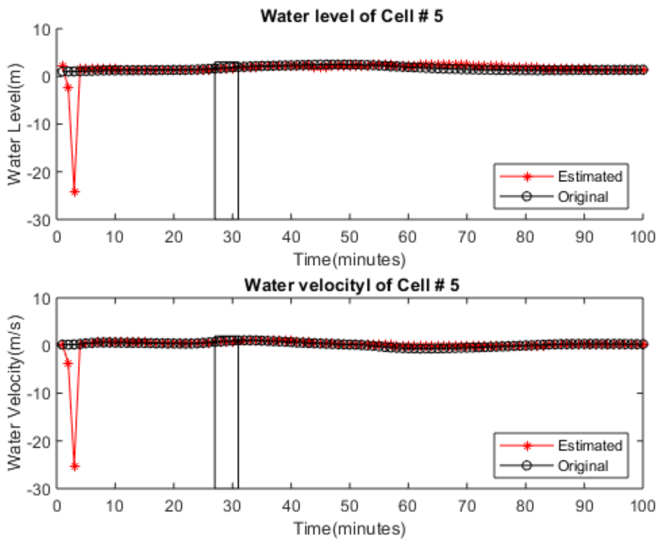

Figure 5-15: The estimated states of water level and water velocity in cell number 5 along with the true values from HEC-RAS.

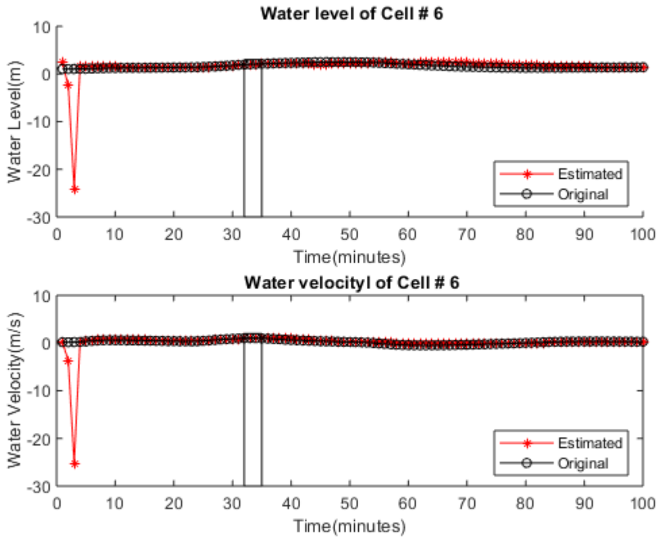

Figure 5-16: The estimated states of water level and water velocity in cell number 6 along with the true values from HEC-RAS.

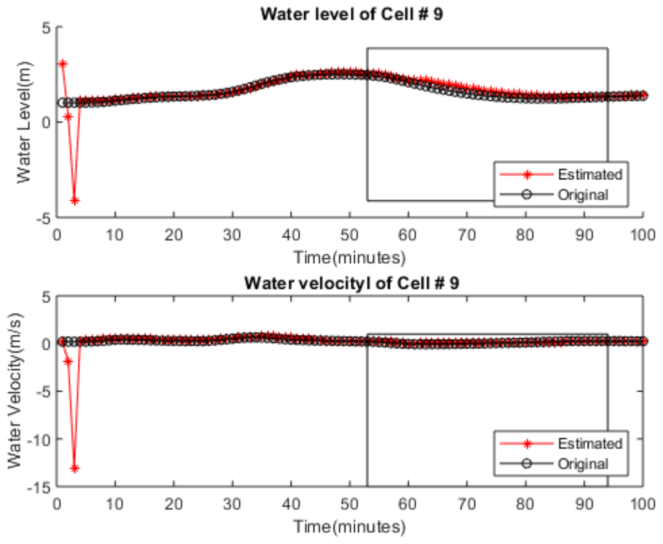

Figure 5-17: The estimated states of water level and water velocity in cell number 9 along with the true values from HEC-RAS.

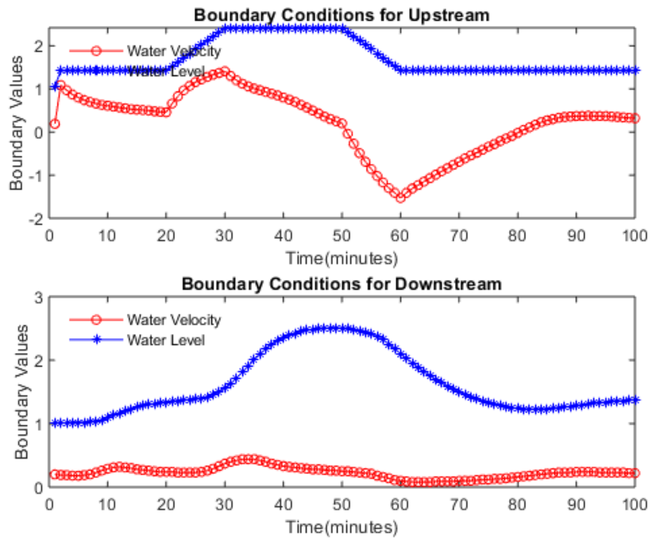

Figure 5-18: The boundary conditions of 1st and last cell for data assimilation.

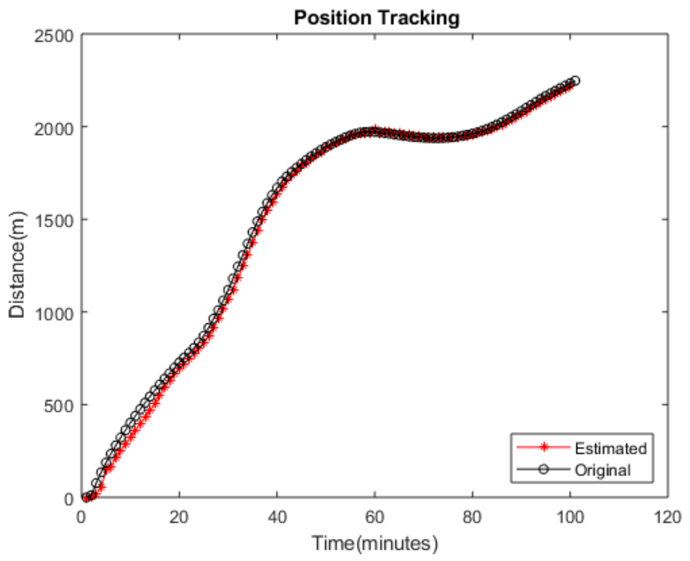

Figure 5-19: The path followed by Lagrangian sensor (Float).

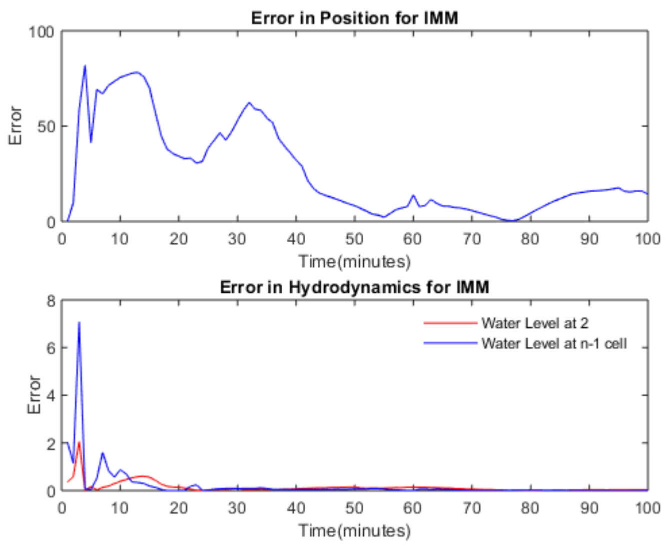

Figure 5-20: The error analysis for the state estimation using Kalman filter

analysis is performed. The performance is shown in Fig. (5-20). The error analysis a significant low error values. The MMSE analysis shows high error in position at start but drops to low values as time passes.

Chapter 6

Experimental Testing

6.1 Experimental Setup

The experiments are conducted at Raiwind-1 canal at Bedian distributary, Lahore, Pakistan as shown in Fig (6-1). The experimental section of the canal is 3 Km in

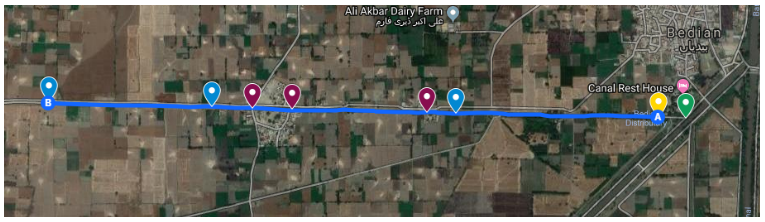

Figure 6-1: The experimental canal site of Raiwind-1 canal with length of 3 Km at Bedian distributary, Lahore, Pakistan.

length. The canal has smooth and paved structure. The cross sections are smooth and constant throughout the canal as shown in Fig (6-2). The canal width is approximately around 13-15 feet and it has depth around 3-5 feet. The experimental scenario is shown in Fig (6-3). The canal has an undershot gate at start which links it to the main canal as shown in Fig (6-4). At the end of the experimental site, an virtual over shot gate is assumed which is completely opened. For the calculations of water level and water velocity due to undershot gate at boundary condition at start of canal, two static sonar sensors are mounted, one is at main MBL canal before the gate and other is after the gate mounted at start of Raiwind-1 canal as shown in

Figure 6-2: The experimental Raiwind- 1 canal view with uniform and paved structure.

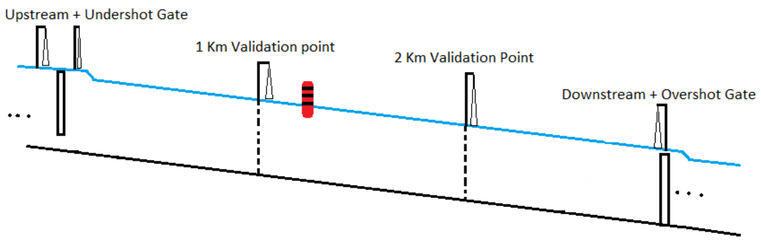

Figure 6-3: The experimental scenario showing sonar sensor, mobile sensor and gates at both ends.

Fig (6-5). For the boundary condition values at the end an static sensor is mounted at the end of experimental canal for overshot gate. For the validation of estimated values, two middle point validation static sensors are also mounted 1 Km apart from each-other as shown in Fig (6-6).

6.1.1 State estimation using Position Data

For the task of data assimilation using position data, the state dependent interacting multiple model(SD-IMM) along Kalman filter is used which is explained in chapter-IV. The experiment duration is around 90 mints. The sampling time of static sonar sensors is around 1 minute with transmission time to the server is around 2 minutes. The gate at start of the canal is opened to maximum height after 20 minutes of experiments and it remained open for around 30 minutes. After 50 minutes of

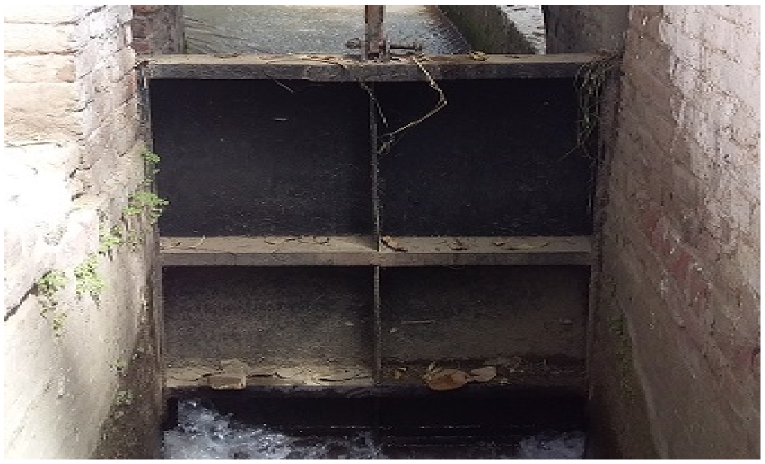
Figure 6-4: The undershot gate at the start of canal to control the water flow in canal and provide boundary condition values.

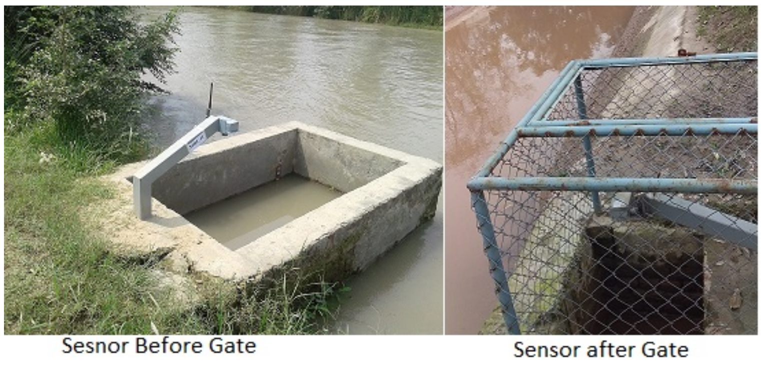

Figure 6-5: The static water level sonar sensors upstream and downstream end of the undershot gate at the start of canal.

experiment, the gate was closed to the original position. The release of mobile sensor is shown in Fig (6-7). One of the major challenges was the asynchronous arrival of data. The sonar sensor provided the values at specific time interval but GPS values from float were arriving at random time intervals as shown in the Fig (6-8). One of the other challenge during the experiment was the missing values of GPS as till first 31 minutes, GPS had transmission issues as GPS antenna was inside the mobile sensor case and also the tree canopy above the canal. In the light of this challenge, the data assimilation is only provided when GPS data is arrived else only prediction step is performed. The sonar sensor data is shown in Fig (6-9). In data assimilation only GPS data of mobile sensor, water level data from start and end point is used

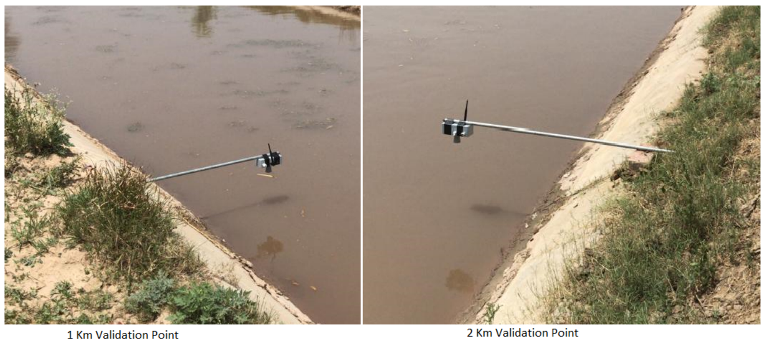

Figure 6-6: The static water level sonar sensors at 1 Km and 2 Km from sensor mounted at downstream of undershot gate.

Figure 6-7: The release of mobile sensor into the water body at upstream end of experimental canal

other two middle point sensor data is used for validation of estimated values. The input to the system model consist of water level and water velocity from gates at both ends. The output sensor vector consist of only start and end point sensor. The experimental site is divided into 12 cells of length 270 meters each. The estimated values for the 1 Km validation point is shown in Fig (6-10). The data assimilation algorithm estimated the values with significant low error even the GPS had highest error till 31 minutes of experiment. The movement of mobile sensor by the 1 Km validation point is shown in Fig (6-11). The estimated values for the 2 Km validation point is shown in Fig (6-12). The movement of mobile sensor by the 2 Km validation point is shown in Fig (6-13). The mobile sensor arrived at the end of experimental site within duration of 90 mints as shown in Fig (6-14). The estimated trajectory of mobile sensor with the comparison to the actual trajectory obtained by GPS co-ordinates is shown in Fig (6-15). The straight line in the Fig (6-15) shows that no GPS data was received due to the issue with mobile sensor and tree canopy over the

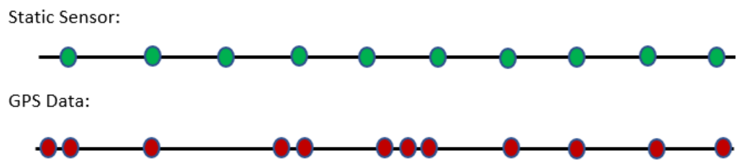

Figure 6-8: The graphical representation of arrival of asynchronous sensor data during experiment

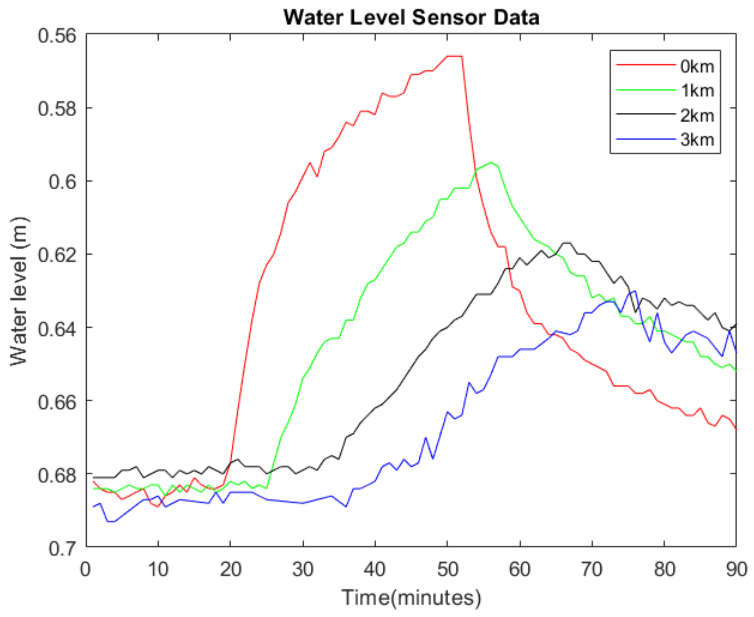

Figure 6-9: The water level data from static sensor at downstream of undershot gate, 1 Km validation point, 2 Km validation point and 3Km downstream end of the experimental site

canal. these issues were fixed within the time span of 10 minutes on the run. As GPS reading were received at the varying rates of sampling so the time scale is little bit different from sonar sensors but the in total it contains values of 90 minutes experiment. At some time of around 65 mints the water wave with high velocity and water level due to gate opening hits the mobile sensor and increase the velocity of mobile sensor. To validate the estimated values, the error analysis is performed at both validation points for water levels and mobile sensor position as shown in Fig (6-16). The error in estimated position is high till 31 minutes of experiments due to the GPS issue in mobile sensor, after that the error is relatively low. The error in estimated

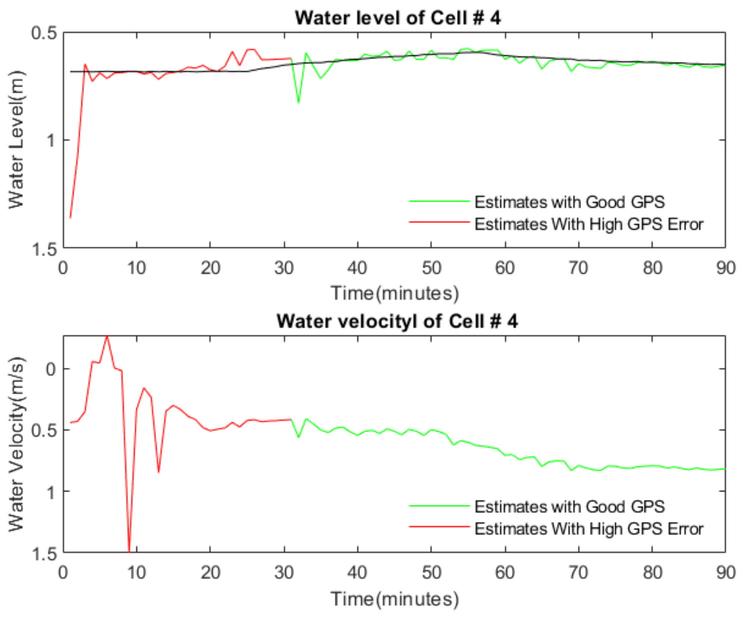

Figure 6-10: The estimated water level with comparison to sensor data from 1 Km validation point at experimental site.

in water level at both validation points is significant low which shows that the data assimilation algorithm performed very well.

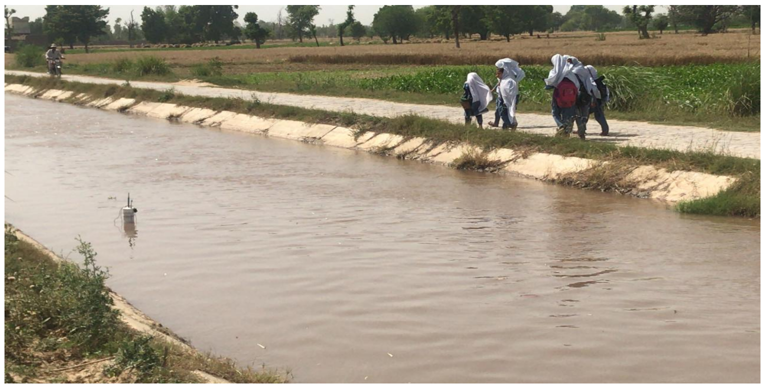
Figure 6-11: The passing of mobile sensor by 1 Km validation point at experimental site.

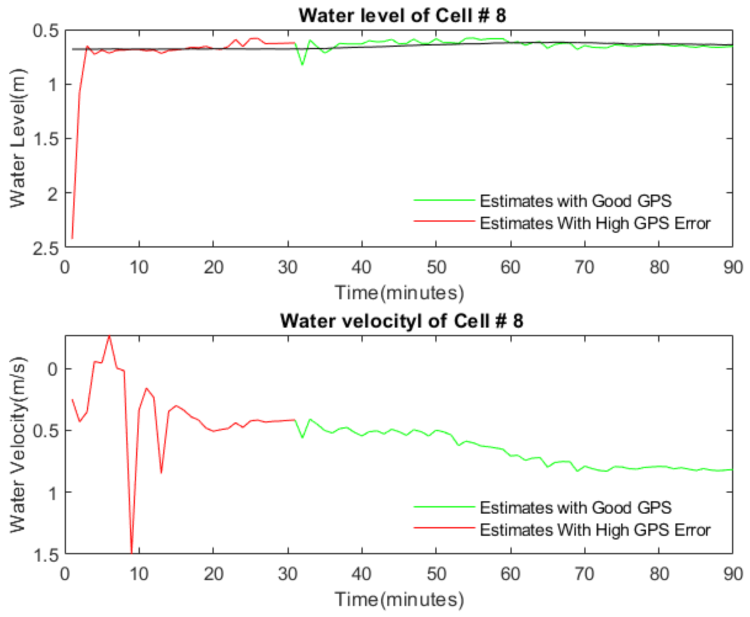

Figure 6-12: The estimated water level with comparison to sensor data from 2 Km validation point at experimental site.

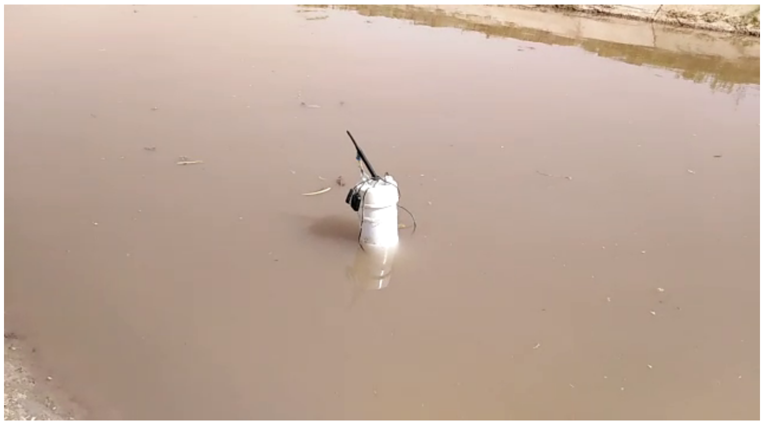

Figure 6-13: The passing of mobile sensor by 2 Km validation point at experimental site.

Figure 6-14: The arrival of mobile sensor at the end point of experimental site after floating passively into water for 3 Km.

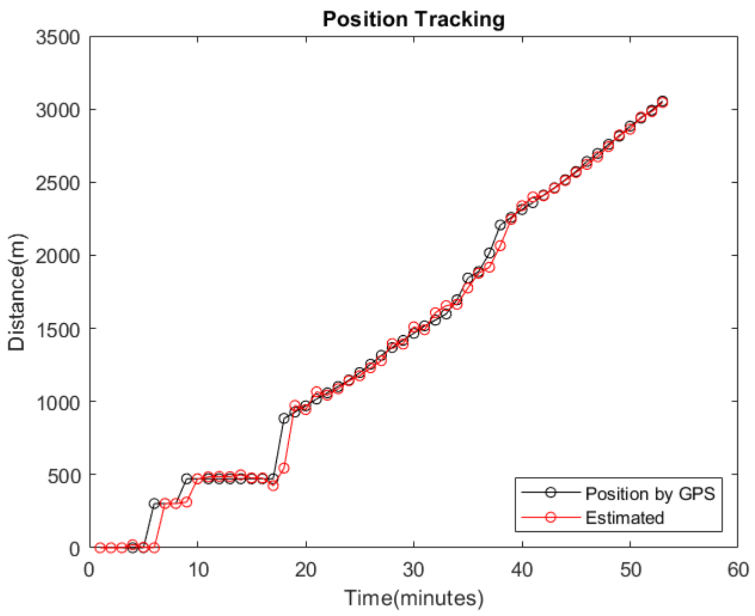

Figure 6-15: The estimated trajectory of mobile sensor with the comparison to the actual trajectory.

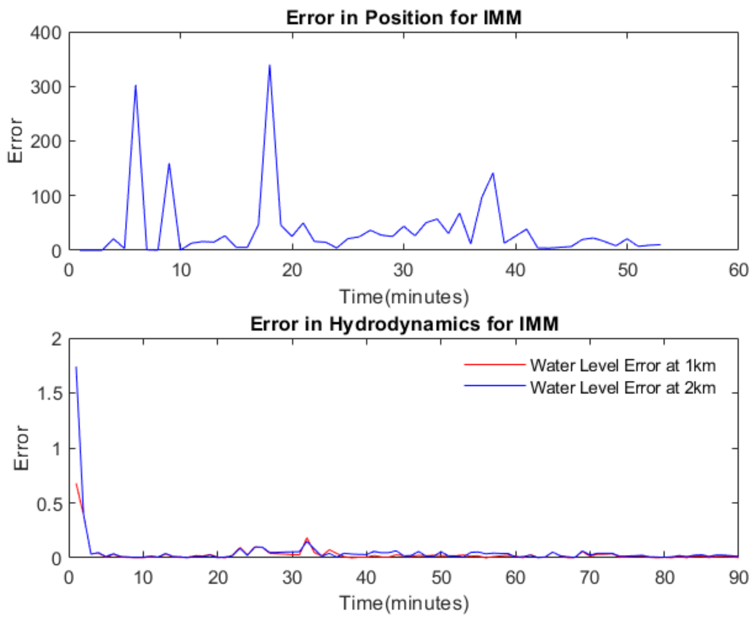

Figure 6-16: The error analysis of estimated water level at both validation points and estimated position of mobile sensor.

Chapter 7

Conclusion

In this research work, states of water bodies are estimated by using mobile sensors. Mobile sensors are good passive floating source for cost effective sensing in water bodies which provide only GPS locations along the channel. The state estimation of water bodies by linearized one dimensional Saint-Venant equations using mobile sensor data is simulated successfully in MATLAB. The system is simulated in the HEC-RAS for rectangular cross sections. The estimated states from Kalman filter are also compared with actual data generated from HEC-RAS. The states estimation is done by using velocity data as well as position data. For position data, the state dependent interacting multiple models (SD-IMM) is used. The data assimilation algorithm is tested in real world at Raiwind-1 canal. Low cost mobile sensors can provide good estimation results by using simplified models for the purposes of tracking of unauthorized activity in channel, irrigation system and track the flow of contamination in channel. For future work, this successful simulated data assimilation method can also be implemented for social sensors to estimate the hydrodynamics in urban areas and for flood mapping.

Bibliography

[1] Zahoor Ahmad, Rubab Khalid, and Abubakr Muhammad. Spatially distributed water quality monitoring using floating sensors. In *IECON 2018-44th Annual Conference of the IEEE Industrial Electronics Society*, pages 2833–2838. IEEE, 2018.

[2] Brian DO Anderson and John B Moore. Optimal filtering. *Englewood Cliffs*, 21:22–95, 1979.

[3] Azizullah Azizullah, Muhammad Nasir Khan Khattak, Peter Richter, and Donat-Peter Häder. Water pollution in pakistan and its impact on public health—a review. *Environment international*, 37(2):479–497, 2011.

[4] Samuel Blackman and Robert Popoli. Design and analysis of modern tracking systems (artech house radar library). *Artech house*, 1999.

[5] M Hanif Chaudhry. *Open-channel flow.* Springer Science & Business Media, 2007.

[6] Geir Evensen. *Data assimilation: the ensemble Kalman filter.* Springer Science & Business Media, 2009.

[7] Michael E Farmer, Rein-Lien Hsu, and Anil K Jain. Interacting multiple model (imm) kalman filters for robust high speed human motion tracking. In *Object recognition supported by user interaction for service robots*, volume 2, pages 20–23. IEEE, 2002.

[8] Azra Jabeen, Xisheng Huang, and Muhammad Aamir. The challenges of water pollution, threat to public health, flaws of water laws and policies in pakistan. *Journal of Water Resource and Protection*, 7(17):1516, 2015.

[9] Leonid Kuznetsov, Kayo Ide, and CKRT Jones. A method for assimilation of lagrangian data. *Monthly Weather Review*, 131(10):2247–2260, 2003.

[10] Xavier Litrico and Vincent Fromion. Boundary control of linearized saint-venant equations oscillating modes. *Automatica*, 42(6):967–972, 2006.

[11] Maurizio Mazzoleni. *Improving flood prediction assimilating uncertain crowd-sourced data into hydrologic and hydraulic models.* CRC Press, 2017.

[12] Abubakr Muhammad. Managing river basins with thinking machines. In *Norbert Wiener in the 21st Century (21CW), 2016 IEEE Conference on*, pages 1–6. IEEE, 2016.

[13] Mohammad Rafiee, Qingfang Wu, and Alexandre M Bayen. Kalman filter based estimation of flow states in open channels using lagrangian sensing. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 8266–8271. IEEE, 2009.

[14] Waqas Riaz, Zahoor Ahmad, and Abubakr Muhammad. A smart metering approach towards measuring flows in small irrigation outlets. *Procedia Engineering*, 154:236–242, 2016.

[15] Michael Rode, Andrew J Wade, Matthew J Cohen, Robert T Hensley, Michael J Bowes, James W Kirchner, George B Arhonditsis, Phil Jordan, Brian Kronvang, Sarah J Halliday, et al. Sensors in the stream: the high-frequency wave of the present, 2016.

[16] Romuald Szymkiewicz. *Numerical modeling in open channel hydraulics*, volume 83. Springer Science & Business Media, 2010.

[17] Andrew Tinka, Issam Strub, Qingfang Wu, and Alexandre M Bayen. Quadratic programming based data assimilation with passive drifting sensors for shallow water flows. *International Journal of Control*, 83(8):1686–1700, 2010.

[18] Martin Verlaan. *Efficient Kalman Filtering Algorithms for Hydrodynamic Models*. PhD thesis, 01 1998.

[19] BT Wahlin and John A Replogle. *Flow measurement using an overshot gate*. UMA Engineering, 1994.

[20] James L Wescoat, Afreen Siddiqi, and Abubakr Muhammad. Socio-hydrology of channel flows in complex river basins: Rivers, canals, and distributaries in punjab, pakistan. *Water Resources Research*, 54(1):464–479, 2018.

Chapter 8

MATLAB: Codes

8.1 Data Assimilation Algorithms

8.1.1 Saint Venant State Space Model

```
function [A,B,A_size,N,hbar,vbar]=matrix2(L,t,del_x,Vo,Ho)
yo = [Vo; Ho];
[x,y]=steady_values(L,yo);
N=ceil(L/del_x);
pick=1;
size_y=size(y);
for k=1:1:N
hbar(k)=y(pick,2);
vbar(k)=y(pick,1);
pick=pick+del_x-1;
end
m=0.02;
g=9.8;           %Gravitaional Force
%Length of Each Grid Point
del_t=t;         %Time Step
A_size=(N*2)-4;
B_size=4;
A=zeros(A_size,A_size);
```

```
alpha(1)=(hbar(2)-hbar(1))/del_x;
alpha(N)=(hbar(N)-hbar(N-1))/(del_x);
beta(1)=-(vbar(1)/hbar(1))*((hbar(2)-hbar(1))/(del_x));
beta(N)=-(vbar(N)/hbar(N))*((hbar(N)-hbar(N-1))/(del_x));
gema(1)=2*g*(m^2)*(vbar(1)/(hbar(1)^(4/3)))+(vbar(1)/hbar(1))*((hbar(2)-hbar(1))/(del_x
gema(N)=2*g*(m^2)*(vbar(N)/(hbar(N)^(4/3)))+(vbar(N)/hbar(N))*((hbar(N)-hbar(N-1))/(del
eta(1)=-(4/3)*g*(m^2)*((vbar(1)*abs(vbar(1)))/(hbar(1)^(7/3)));
eta(N)=-(4/3)*g*(m^2)*((vbar(N)*abs(vbar(N)))/(hbar(N)^(7/3)));
w=2;
for i=1:N-2
alpha(i)=(hbar(w+1)-hbar(w-1))/(2*del_x);
beta(i)=(-vbar(w)/hbar(w))*((hbar(w+1)-hbar(w-1))/(2*del_x));
gema(i)=2*g*(m^2)*(vbar(w)/(hbar(w)^(4/3)))-(-vbar(w)/hbar(w))*((hbar(w+1)-hbar(w-1))/(
eta(i)=-(4/3)*g*(m^2)*((vbar(w)*abs(vbar(w)))/(hbar(w)^(7/3)));
w=w+1;
end
vshift=1;
hshift=1;
A(1,:)=[0 ...
    (1/2)-(del_t/(4*del_x))*(vbar(3)+vbar(1))-((del_t/2)*gema(3)) ...
    zeros(1,N-3) (-g*(del_t/(2*del_x))-(del_t/2)*eta(3)) zeros(1,N-4)];
A(A_size/2,:)=[zeros(1,N-4) ...
    (1/2)+(del_t/(4*del_x))*(vbar(N)-vbar(N-2))-(del_t/2)*gema(N-2) ...
    zeros(1,N-3) (g*(del_t/(2*del_x))-(del_t/2)*eta(N)) 0];
w=3;
for i=2:1:(A_size/2)-1
A(i,:)=[(1/2)+(del_t/(4*del_x))*(vbar(w+1)-vbar(w-1))-(del_t/2)*gema(w-1) ...
    0 ...
    (1/2)-(del_t/(4*del_x))*(vbar(w+1)+vbar(w-1))-(del_t/2)*gema(w+1) ...
    zeros(1,N-5) (g*(del_t/(2*del_x))-(del_t/2)*eta(w-1)) 0 ...
    (-g*(del_t/(2*del_x))-(del_t/2)*eta(w+1)) zeros(1,N-5)];
if i>2
A(i,:)=circshift(A(i,:),[vshift,1]);
vshift=vshift+1;
end
w=w+1;
```

```
end
A((A_size/2)+1,:)=[0 ...
    (-del_t/(4*del_x))*(hbar(3)+hbar(1))-(del_t/2)*alpha(3) ...
    zeros(1,N-3) ...
    ((1/2)-(del_t/(4*del_x)*(vbar(3)+vbar(1)))-(del_t/2)*beta(3)) ...
    zeros(1,N-4)];
hrow=(A_size/2)+2;
for i=3:1:(A_size/2)
A(hrow,:)=[(del_t/(4*del_x))*(hbar(i+1)+hbar(i-1))-(del_t/2)*alpha(i-1) ...
    0 (-del_t/(4*del_x))*(hbar(i+1)+hbar(i-1))-(del_t/2)*alpha(i+1) ...
    zeros(1,N-5) ...
    ((1/2)+(del_t/(4*del_x)*(vbar(i+1)+vbar(i-1)))-(del_t/2)*beta(i-1)) ...
    0 ...
    ((1/2)-(del_t/(4*del_x)*(vbar(i+1)+vbar(i-1)))-(del_t/2)*beta(i+1)) ...
    zeros(1,N-5)];
if hrow>(A_size/2)+2
A(hrow,:)=circshift(A(hrow,:),[hshift, 1]);
hshift=hshift+1;
end
hrow=hrow+1;
end
A(A_size,:)=[zeros(1,N-4) ...
    ((del_t/(4*del_x))*(hbar(N)+hbar(N-2)))-((del_t/2)*alpha(N-2)) ...
    zeros(1,N-3) ...
    (1/2)+((del_t/(4*del_x)*(vbar(N)+vbar(N-2))))-((del_t/2)*beta(N-2)) ...
    0];
B=zeros(A_size,B_size);
 B(1,:)=[(1/2)+(del_t/(4*del_x))*(vbar(3)-vbar(1))-(del_t/2)*gema(1) ...
     0 (g*(del_t/(2*del_x)))-(del_t/2)*eta(1) 0];
 B(A_size/2,:)=[0 ...
     (1/2)-((del_t/(4*del_x))*(vbar(N)+vbar(N-2)))-((del_t/2)*gema(N)) ...
     0 (-g*(del_t/(2*del_x))-(del_t/2)*eta(N))];
 B((A_size/2)+1,:)=[((del_t/(4*del_x))*(hbar(3)+hbar(1)))-((del_t/2)*alpha(1)) ...
     0 (1/2)+(del_t/(4*del_x))*(vbar(3)+vbar(1))-(del_t/2)*beta(1) 0];
 B(A_size,:)=[0 ...
     ((-del_t/(4*del_x))*(hbar(N)+hbar(N-2)))-((del_t/2)*alpha(N)) 0 ...
    (1/2)−((del_t/(4*del_x))*(vbar(N)+vbar(N−2)))−((del_t/2)*beta(N))];
```

8.1.2 Algorithm for Position Data Assimilation

```
 function []=Aug_DA_Exp()
 close all
 dt_f=0;
 %Reading GPS data for Float
fileID = fopen('position2.txt');
C = textscan(fileID,'%f');
fclose(fileID);
whos C;
pos=[C{1}];
%Reading Sonar sensor data at 1km range.
fileID = fopen('1KM_rangedata.txt');
C = textscan(fileID,'%f');
fclose(fileID);
whos C;
h1km=flipud([C{1}]);
h1km=h1km./1000; %Converting to meters from milimeters
%Reading sonar sensor data at 2km
fileID = fopen('2KM_rangedata.txt');
C = textscan(fileID,'%f');
fclose(fileID);
whos C;
h2km=flipud([C{1}]);
[h2km]=interpolation(h2km);
h2km=h2km./1000;
del_x=270; %Cell size
[y2,y3,v1,v3]=boundary_cond(); %Computing boundary conditions.
h1km=h1km+0.2;
h2km=h2km+0.22;
y3=y3+0.2;
[pos1,steps]=dis(del_x); %Cumputing Euclidian distance and number of ...
    measurmentss within each cell
```

```
%Definig parameters to compute matrix A and B
Pos=pos;
time=length(pos);
L=pos(end); %Channel Length in meters
t=1; %time step between two cell for discretization
%Initial values for steady state back water curve
Vo=v1(1);
Ho=y2(1);
% Defining system
[A,B,A_size,n,hbar,vbar]=matrix2(L,t,del_x,Vo,Ho);
len_A=length(A) ;
%Defining System matrix H
H(1,:)=[0 zeros(1,(A_size/2)) 1 zeros(1,(A_size/2)-1)];
H(2,:)=[1 zeros(1,(A_size))];
H(3,:)=[zeros(1,(A_size)) 1];
%Initial states
X=[pos(1) vbar(2:end-1) hbar(2:end-1)]';
%Defining state transition matrix
Pt=n;
Pmatrix=conv2(eye(Pt),[0.1 0.8 0.1],'same');
Pmatrix(1,2)=0.2;
Pmatrix(n,n-1)=0.2;
% Definig ui, probability that target is in the state i as computed just
% after the data is received
ui=[0.75 0.15 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01 0.01]';

%Conditional Probability given that the tragte is in state j that the
%transition occured from state i
uij=zeros(Pt,Pt);
%Defining Kalman filter coveriance matrixes
Q=eye(A_size+1);
P=eye(A_size+1);
R=eye(3);
%Defining multistep Q (cont. time). For discrete time multiply by ...
    dominant
%dt term
```

```
for i=1:A_size+1
    if i==1
        Q(i,i)=0.015 %Position  ¬25% of baseline 0.5 m/s due to air ...
            gusts (0.015)
    elseif i>1&&i<((A_size/2)+1)
        Q(i,i)=0.01  %Velocity ¬25% of baseline 0.5 m/s due to ...
            modeling errors (0.01)
    else
        Q(i,i)=0.0001  %Water level (+/− 1 cm error) (0.0001)
    end
end
%Defining multistep P
for i=1:A_size+1
    if i==1
        P(i,i)=30 %Position
    elseif i>1&&i≤((A_size/2)+1)
        P(i,i)=30  %Velocity
    else
        P(i,i)=30  %Water level
    end
end
for i=1:3
    if i==1
        R(i,i)=0.0001 %Start water level
    elseif i==2
        R(i,i)=0.001   %Position
    else
        R(i,i)=0.0001  %Water level
    end
end
%Creating n state vectors
for i=1:n
    Xj{i}=X;
end
sta=X;
% Defining IMM
```

```
for k=1:time      %loop on time.
    output(:,k)=H*sta;
    State(:,k)=sta;
    %Sensor output vector for Kalman Filter
    z=[y2(k) Pos(k) y3(k)];
    %Input vector for System
    u=[v1(k) v3(k) y2(k) y3(k)];
    Pro(:,k)=ui;
    %IMM Mixing
    %Computing Cj
    C=Pmatrix*ui;
    %Computing uij
    for i=1:n
        for j=1:n
            uij(i,j)=(Pmatrix(i,j)*ui(i))/C(j);
        end
    end
    for cell=1:n
        %Computing Xj
        for i=1:n
            Xtemp{i}=uij(i,cell)*Xj{i};
        end
         Xj{cell}=0;
        for i=1:n
            Xj{cell}=Xj{cell}+Xtemp{i};
        end
        %Computing Pj
        for i=1:n
            Ptemp{i}=uij(i,cell)*(P+((Xj{i}-Xj{cell})*(Xj{i}-Xj{cell})'));
        end
        Pj{cell}=0;
        for i=1:n
            Pj{cell}=Pj{cell}+Ptemp{i};
        end
        %Defining Augmented models
        if cell==1
```

```
            A_aug=[1 zeros(1,(len_A));
                  zeros(len_A,1) A];
            B_aug=[60 zeros(1,3); B];
        elseif cell==n
            A_aug=[1 zeros(1,(len_A));
                  zeros(len_A,1) A];
            B_aug=[0 60 zeros(1,2); B] ;
        else
            A_aug=[1 zeros(1,cell-2) 60 zeros(1,(len_A)-(cell)+1);
                  zeros(len_A,1) A];
            B_aug=[zeros(1,4); B];
        end
        %Applying Kalman filter

        [Xj{cell}, Pi{cell}] = predict(Xj{cell}, Pj{cell}, A_aug, Q, ...
           B_aug, u);         %State Prediction
        if Pos(k)≠-1  %Condition to cehck if GPS data is available ...
           or not
        [nu{cell}, S{cell}] = innovation(Xj{cell}, Pi{cell}, z, H, ...
           R);                %Computing Innovaton/error
        [Xj{cell}, Pi{cell},c_out] = innovation_update(Xj{cell}, ...
           Pi{cell}, nu{cell}, S{cell}, H);  %Updating states by ...
           using kalman gain and innovation
        %computing statistical distance of an obervation-to-track ...
           assignment
        d(cell)=nu{cell}'*inv(S{cell})*nu{cell};
        sigma(cell)=exp(-d(cell)/2)/sqrt(((2*pi)^3)*det(S{cell}));
        end
    end
    %Updating Probabilities
    if Pos(k)≠-1
    Ctemp=0;
    for i=1:n
        Ctemp=Ctemp+(sigma(i)*C(i));
    end
    for i=1:n
```

```
            ui(i)=(sigma(i)*C(i))/Ctemp;
        end
        end
        P=0;
        %Combining State vector and Covariance matrix from all models
        for i=1:n
            St_temp{i}=C(i)*Xj{i};
            P_temp{i}=C(i)*Pi{cell};
            %Pi_temp2{i}=C(i)*Pi{i};
            P=P+P_temp{i};
        end
         sigma_heat(:,k)=sigma;
        cel2mat=cell2mat(St_temp);
        sta=sum(cel2mat,2);
    end
    %Plotting
    for cell=1:n-2
         figure(cell) %Plotting estimated values of current cell
         subplot(2,1,1)
         %hold on
         plot(State((A_size/2)+cell+1,:),'g','MarkerSize',5)
         hold on
         plot(State((A_size/2)+cell+1,1:31),'r','MarkerSize',5)
         hold off
         if cell+1==4
         hold on
         plot(h1km,'k','MarkerSize',5)
         hold off
         end
         if cell+1==8
             hold on
             plot(h2km,'k','MarkerSize',5)
             hold off
         end
         %hold on
         str = sprintf(' Water level of Cell # %d', cell+1);
         title(str)
         xlabel('Time(minutes)')
         ylabel('Water Level(m)')
         xlim auto
         ylim auto
         %ax.YDir = 'reverse'
         legend('Estimates with Good GPS','Estimates With High GPS ...
            Error','Location','southeast')
         legend('boxoff')
         ax=gca;
         Ylim=get(ax,'YLim');
         ax.YDir = 'reverse';
         hold off
         subplot(2,1,2)
         plot(State((cell)+1,:),'g-','MarkerSize',5)
         hold on
         plot(State(cell+1,1:31),'r','MarkerSize',5)
         %plot(vel((1:time),cell+1),'k-o','MarkerSize',5)
         hold off
         str = sprintf(' Water velocityl of Cell # %d', cell+1);
         title(str)
         xlabel('Time(minutes)')
         ylabel('Water Velocity(m/s)')
         xlim auto
         ylim auto
         legend('Estimates with Good GPS','Estimates With High GPS ...
            Error','Location','southeast')
         legend('boxoff')
         ax=gca;
         ax.YDir = 'reverse';
         Ylim=get(ax,'YLim');
         hold off
    end
    affa=1;
    figure(cell+1)
    for i=1:1:length(pos)
```

```
    if Pos(i)≠-1
    position(affa)=State(1,i);
    affa=affa+1;
    end
end
plot(pos1,'k-o','MarkerSize',5)
hold on
plot(position,'r-o','MarkerSize',5)
str = sprintf('Position Tracking');
title(str)
xlabel('Time(minutes)')
ylabel('Distance(m)')
xlim auto
ylim auto
ax=gca;
legend('Position by GPS','Estimated','Location','southeast')
figure(cell+2)
subplot(2,1,1)
plot(v1,'r-o','MarkerSize',5)
hold on
plot(y2,'b-*','MarkerSize',5)
hold off
str = sprintf('Boundary Conditions for Upstream');
title(str)
xlim auto
ylim auto
ax=gca;
ax.YDir = 'reverse'
xlabel('Time(minutes)')
ylabel('Boundary Values')
legend('Water Velocity','Water Level','Location','northwest')
legend('boxoff')
subplot(2,1,2)
plot(v3,'r-o','MarkerSize',5)
hold on
plot(y3,'b-*','MarkerSize',5)
```

```
hold off
str = sprintf('Boundary Conditions for Downstream');
title(str)
xlim auto
ylim auto
ax=gca;
ax.YDir = 'reverse'
xlabel('Time(minutes)')
xlim auto
ylim auto
ylabel('Boundary Values')
legend('Water Velocity','Water Level','Location','northwest')
legend('boxoff')
s_pro=Pro;
figure(cell+3)
h = heatmap([1:time],[1:n],Pro);
str = sprintf('Heatmap for Probabilities of Models in IMM');
title(str)
xlabel('Time(minutes)')
ylabel('Model')
figure(cell+4)
h = heatmap([1:time],[1:n],sigma_heat);
str = sprintf('Heatmap for sigma of Models in IMM');
title(str)
xlabel('Time(minutes)')
ylabel('Model')
figure(cell+5)
%SENSOR DATA Plotting
plot(y2,'r')
hold on
plot(h1km,'g')
hold on
plot(h2km,'k')
hold on
plot(y3,'b')
ax=gca;
```

```
ax.YDir = 'reverse'
title('Water Level Sensor Data')
xlabel('Time(minutes)')
xlim auto
ylim auto
ylabel('Water level (m)')
legend('0km','1km','2km','3km')
%Computing Error
 Error_pos=abs(pos1-position)
 Error_level2=abs(State((A_size/2)+3+1,:)'-h1km)
 Error_leveln=abs(State((A_size/2)+7+1,:)'-h2km)

figure(cell+5)
subplot(2,1,1)
plot(Error_pos,'b')
str = sprintf('Error in Position for IMM');
title(str)
xlabel('Time(minutes)')
ylabel('Error')
subplot(2,1,2)
plot(Error_level2,'r')
hold on
plot(Error_leveln,'b')
hold off
%plot(output(2,:),'b')
str = sprintf('Error in Hydrodynamics for IMM');
title(str)
xlabel('Time(minutes)')
ylabel('Error')
legend('Water Level Error at 1km','Water Level Error at 2km')
legend('boxoff')
%Defining Kalman Filter Functions
function [Xpred, Ppred]=predict(x,P,F,Q,B,u)
Xpred=F*x+B*u';
Ppred=F*P*F'+Q;
end
```

```
function [nu,S]=innovation(Xpred,Ppred,z,H,R)
nu=z'-(H*Xpred);
S=R'+(H*Ppred*H');
end
function [Xnew,Pnew,c_out]=innovation_update(Xpred,Ppred,nu,s,H)
K=Ppred*(H'*inv(s));
Xnew=Xpred+(K*nu);
c_out=H*Xnew;
Pnew=Ppred-(K*s*K');
end
 end
```

8.1.3 Algorithm for Velocity Data Assimilation

```
function []=new_model_KF2()
%Physical System output /Sesnor output
no_out=100;
fileID = fopen('ifac.txt');
C = textscan(fileID,'%f %f %f %f %f');
fclose(fileID);
whos C
out=[C{1} C{2} C{3} C{4}] ;
len=length(out);
vel= reshape(out(:,4),[no_out,11]);
level=reshape(out(:,3),[no_out,11]);
vel=vel
level=level
j=1;
Vo=vel(1,2);
Ho=level(1,2);
del_x=240;
Pos=0;
dis=del_x;
%Calculating Position of float
for i=1:no_out
```

```

Pos(i+1)=Pos(i)+vel(i,j)*60  %Computing Position
if Pos(i+1)≥dis
    dis=dis+del_x
    steps(j)=i
    j=j+1
    i=i
end
if Pos(i+1)>2400
    steps(j)=i
    break
end
if i+1>no_out
    steps(j)=i
    break
end
end
%Defining System
L=dis %Channel Length in meters
t=1 ;%time step between two cell for discretization
[A,B,A_size,N]=matrix2(L,t,del_x,Vo,Ho)
j=0;
A_size=A_size
len_A=length(A)
  n=N   %No. of cells in river channel
%Defining coverience matrixes
r=10;
q=30;
p=10;
Q=q*eye(A_size);
P=p*eye(A_size);
R=r*eye(3);
%code varaibles
cy_state=[];
cy_output=[];
p_st=1;
```

```
st=1;
H=[];
%initial states for cyber system
ini_v=vel(2,2);
ini_h=level(2,2);
ini_pos=0
X = [repmat(ini_v,1,A_size/2) repmat(ini_h,1,A_size/2)];
x_cy=X';
%Defining System matrix C
H(1,:)=[zeros(1,(A_size/2)) 1 zeros(1,(A_size/2)-1)];
H(2,:)=[1 zeros(1,(A_size)-1)];
H(3,:)=[zeros(1,(A_size)-1) 1];
st=steps(1)+1
loop=steps(2)-steps(1)
State=[]
s=1;
for cell=1:n-2
    u=[];
    z=[];
    se=steps(cell+1)
    z=[level((st:se),2) vel((st:se),cell+1) level((st:se),n-1)];
    u=[vel((st:se),1) vel((st:se),n) level((st:se),1) level((st:se),n)];
       state=[];
       output=[];
    for k=1:1:loop
        state(:,k)=x_cy;           %Saving states at each time step
        [xpred, Ppred] = predict(x_cy, P, A, Q,B,u(k,:));        ...
           %State Prediction
        [nu, S] = innovation(xpred, Ppred, z(k,:), H, R); ...
                            %Computing Innovaton/error
        [x_cy, P,c_out] = innovation_update(xpred, Ppred, nu, S, H); ...
            %Updating states by using kalman gain and innovation
        cy_st=x_cy;
        state(:,k)=x_cy;
        State(:,s)=x_cy
        q(:,s) = nu'*inv(S)*nu
        error(:,s)=nu
        s=s+1;
        output(:,k)=c_out;
        cy_out=c_out;
    end
    H(2,:)=circshift(H(2,:),[1, 1]);
    if cell+1>n-1
        break
    end
    st=steps(cell+1)+1
    loop=steps(cell+2)-steps(cell+1)
    j=cell
    A_aug=[1 zeros(1,j) t zeros(1,(len_A-1)-j);
         zeros(len_A,1) A]
end
st=steps(1)+1;
   for cell=1:n-2
    figure(cell) %Plotting estimated values of current cell
    subplot(2,1,1)
    se=steps(cell+1);
    plot([steps(1)+1:steps(end-1)],State(cell,:),'r-*','MarkerSize',5)
    hold on
    plot([steps(1)+1:steps(end-1)],vel([steps(1)+1:steps(end-1)],cell+1),'k-o','MarkerS
    hold on
    loop=se-st;
    rectangle('Position',[st -1 loop 6])
    hold off
    str = sprintf('Water Velocity Cell# %d ',cell+1);
    title(str)
    xlabel('Time(minutes)')
    ylabel('Water Velocity(m/s)')
    xlim auto
    ylim auto
    legend('Estimated', 'HEC-RAS' ,'Location','northwest')
    legend('boxoff')
    subplot(2,1,2)
    plot([steps(1)+1:steps(end−1)],State(cell+(A_size/2),:),'g−*','MarkerSize',5)
    hold on
    plot([steps(1)+1:steps(end−1)],level([steps(1)+1:steps(end−1)],cell+1),'b−o','Marke
    hold on
    rectangle('Position',[st 1 loop 3]);
    hold off
    str = sprintf('water level from cell# %d',cell+1);
    title(str)
    xlabel('Time(minutes)')
    xlim auto
    ylim auto
    ylabel('Water Level(m)')
    legend('Estimated','HEC−RAS','Location','northwest')
    legend('boxoff')
    st=steps(cell+1)+1;
   end
   figure(cell+1)
    plot(Pos,'−*','MarkerSize',5)
    xlim auto
    ylim auto
    xlabel('Time (minutes)')
    ylabel('distance (m)')
    title('Movement of Float')
figure(cell+2)
subplot(2,1,1)
plot(vel(:,1),'r−o','MarkerSize',5)
hold on
plot(level(:,1),'b−*','MarkerSize',5)
hold off
str = sprintf('Boundary Conditions for Upstream');
title(str)
xlim auto
ylim auto
xlabel('Time(minutes)')
ylabel('Boundary Values')
legend('Water Velocity','Water Level','Location','northwest')
legend('boxoff')
subplot(2,1,2)
plot(vel(:,n),'r-o','MarkerSize',5)
hold on
plot(level(:,n),'b-*','MarkerSize',5)
hold off
str = sprintf('Boundary Conditions for Downstream');
title(str)
xlim auto
ylim auto
xlabel('Time(minutes)')
xlim auto
ylim auto
ylabel('Boundary Values')
legend('Water Velocity','Water Level','Location','northwest')
legend('boxoff')
for i=1:3
    error_cov(i,:)=xcorr(error(i,:))
end
sumq=sum(q);
figure(cell+3)
plot([steps(1)+1:steps(end-1)],q/steps(end))
str = sprintf('Chi square analysis');
title(str)
xlim auto
ylim auto
xlabel('Time(minutes)')
xlim auto
ylim auto
ylabel('error')
figure(cell+4)
plot((error_cov(steps(end):2*steps(end)-1))/(error_cov(steps(end))))
str = sprintf('Error Autocorrelation Analysis');
title(str)
xlim auto
ylim auto
```

```matlab
xlabel('Time(minutes)')
xlim auto
ylim auto
ylabel('Error')
%%Kalman filter functions
function [Xpred, Ppred]=predict(x,P,F,Q,B,z1)
Xpred=F*x+B*z1';
Ppred=F*P*F'+Q;
end
function [nu,S]=innovation(Xpred,Ppred,z,H,R)
nu=z'-(H*Xpred);
S=R'+(H*Ppred*H');
end
function [Xnew,Pnew,c_out]=innovation_update(Xpred,Ppred,nu,s,H)
K=Ppred*(H'*inv(s));
Xnew=Xpred+(K*nu);
c_out=H*Xnew;
Pnew=Ppred-(K*s*K');
end
end
```