TECHNICAL UNIVERSITY OF MUNICH

DEPARTMENT OF INFORMATICS

Master's Thesis in Informatics

Vehicle Localization and Tracking for Collision Avoidance System

Behtarin Ferdousi

TECHNICAL UNIVERSITY OF MUNICH

DEPARTMENT OF INFORMATICS

Master's Thesis in Informatics

Vehicle Localization and Tracking for Collision Avoidance System

Fahrzeuglokalisierung und -verfolgung für das Kollisionsvermeidungssystem

Author:	Behtarin Ferdousi
Supervisor:	Prof. Dr.-Ing. Matthias Althoff
Advisor:	Jagat Rath , Ph.D
Submission Date:	05.01.2020

I confirm that this master's thesis is my own work and I have documented all sources and material used.

Ich versichere, dass ich diese Master's Thesis selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Munich, 05.01.2020 Behtarin Ferdousi

Acknowledgments

Yet to be written

Abstract

With the current pace of development in autonomous vehicles, the demand for high-intelligent collision avoidance system is increasing. Due to noise in measurements from Lidar, GPS and radar sensors, researchers have utilized state estimation methods to converge measurements to the true state of the system. The purpose of this thesis is to review and implement different algorithms of set-based state estimation using zonotopes as domain representation on existing dataset of real traffic participants. Set-based methods are used in order to bound the true state of the system to a set, in contrast to an estimation from stochastic methods tolerating slight divergence from the true state, which is undesirable in autonomous vehicles to ensure safety. The algorithms are compared in terms of computation time, time to converge, tightness of bound and accuracy.

Contents

1 Introduction

There is a steep progress in research and development of autonomous vehicles. The race to the top of automobile industry, participated by companies like BMW, Tesla, Waymo/Google, requires fast devlopment and vigorous testing of novel technology. One of the many challenges of this field is to ensure collision avoidance. With no human behind wheels for Level 5 **[SAE2014]** cars, the vehicle must keep track of roads, surrounding traffic participants, like vehicles and pedestrians, in different circumstances including rain and fog, to ensure safety of its passengers. Current collision avoidance systems based on sensors, radar and camera will be overwhelmed with high computation demands for this purpose. Tolerating error in such system can cause accidents; such error in vehicles have already caused real-life accidents, including one resulting in death [1].

The Collision Avoidance System in car system consists of two parts: Sensing and Tracking, and Maneuver. The sensing and tracking part is done by sensors like radar, camera and GPS (Global Positioning System). With advancement in technologies in image processing, image analysis and object detection and the decline in the cost of camera sensors, the sensing and tracking is developing fast. Although cameras can classify vehicles, it cannot gurantee measurement in low-light environment (e.g. night) [**Hirz2018**]. On the other hand, radar gurantee robustness to weather with in exchange of high cost. Similarly, GPS has disturbances too which makes methods using combination of sensors to cover each others drawbacks to provide vehicle localization. After getting the data, the Maneuver is carried out in a way to avoid collision with the location found from the sensors. The probable location of the tracked vehicle, is thus also important to calculate a predicted trajectory. However, all the data to predict the vehicle's location is not measurable using just sensors and nevertheless the sensor data are not 100% accurate, and hence solely cannot be used to carry on maneuver to avoid collision.

Due to the lack of quality and availability of sensors, researchers have used state estimation algorithms to determine the state of the tracked vehicle. One of the widely applied technique is the Kalman Filter, that requires a probability distribution of perturbation in the measurements.

[1] https://www.theguardian.com/technology/2018/mar/19/uber-self-driving-car-kills-woman-arizona-tempe

On the other hand, there exists set-based state estimation which provides a set of possible states of the system, instead of a close estimation like Kalman filter. A high degree of accuracy and guarantee is demanding in the collision avoidance system, hence we chose to compare the set based state estimation algorithms for the scenario. A similar comparison can be found in [**Rath**].

Comparing different domain representation of the sets enclosing the possible state of the system, we chose zonotopes as opposed to ellipsoid and polytopes due to higher accuracy for a lower computation cost. Furthermore, zonotopes have gained fame for state estimation because of wrapping effect(i.e. not increasing in size in time due to accumulated noises) and Minkowski sum(i.e. sum of zonotopes is also a zonotope). We used CORA in Matlab for the functionalities in zonotope required for state estimation.

In order to utilize the state estimation algorithms, the foremost necessary step is to define the tracked vehicle in a linear model. Although there are complex models that can be used to represent a vehicle state [**Althoff**], not all can be used due to unaivailability of measurements like wheelbase, velocity, etc. unlikely to be acquired in run-time from tracked vehicle. Hence, the models used in this paper to compare are the simplest : Constant Velocity, Constant Acceleration and the Singer Acceleration Model.

The paper is organized as follows. Chapter 2 presents the vehicle localization problem suitable to be solved by state estimation algorithms. The following Chapter 3 discusses the zonotope-based state estimation algorithms to be compared. Chapter 4 and 5 give the evaluation of the algorithms. Finally, chapter 6 concludes with a summary and a discussion of possible future works.

2 Vehicle Localization : The Guaranteed Estimation Problem

Let us denote the state of the vehicle to be tracked at time k as x_k and the measured state as y_k. The equations to predict x_k from a previous step x_{k-1} and the mapping from measurement, y_k to state, x_k is shown in equation (2.1), where A,E,C and F are known matrices, w_k and v_k are process noise and measurement noise at time k, respectively.

$$\begin{aligned} x_{k+1} &= Ax_k + Ew_k \\ y_k &= Cx_k + Fv_k \end{aligned} \tag{2.1}$$

The state of the tracked vehicle can be represented using position, velocity and acceleration in x and y-direction. Different states can be estimated using different models, whereas the measured state of the vehicle is assumed to be position in x and y-direction for all models.

$$y = [s_x \quad s_y]^T$$

The process noise(w_k) bound is set as $[.1 \quad .1 \quad .4 \quad .4 \quad .1 \quad 0.1]^T$ and the measurement noise(v_k) bound is $[.1 \quad .1]^T$.

The state(x_k), the state transition matrix(A) and measurement matrix (C) differ across different models as discussed in the following section.

2.1 Vehicle Model

Three linear systems are implemented to compare the different algorithms for tracked vehicles. Although there exists highly precise vehicle models for ego vehicles, the simplest models are used here to represent the tracked vehicle since no precise model of tracked vehicles are available. In particular, physical dimensions like wheelbase or side-slip, cannot be measured directly. Another reason is that adding steering angle and yaw rate makes the system non-linear and hence does not suit all the algorithms presented. Hence, we chose to investigate the following models:

- **Constant Velocity Model**

- **Constant Acceleration Model**
- **Point Mass Model**

2.1.1 Constant Velocity Model

The vehicle is assumed to travel in constant velocity [**Schubert2008**]. The state of the system is:
$\begin{bmatrix} s_x & s_y & v_x & v_y \end{bmatrix}^T$
The state transition matrix (A in (2.1)) is derived to be:
$$\begin{bmatrix} 1 & 0 & \Delta T & 0 \\ 0 & 1 & 0 & \Delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
The measurement matrix(C in (2.1)) is:
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

2.1.2 Constant Acceleration Model

Although the Constant velocity Model is easy to implement, however it is dangerous to assume constant velocity. Acceleration model takes care of changing velocity and assumes constant acceleration [**Schubert2008**]. The state of the system is:
$\begin{bmatrix} s_x & s_y & v_x & v_y & a_x & a_y \end{bmatrix}^T$.
The state transition matrix (A in (2.1)) is derived to be:
$$\begin{bmatrix} 1 & 0 & \Delta T & 0 & \frac{1}{2}\Delta T^2 & 0 \\ 0 & 1 & 0 & \Delta T & 0 & \frac{1}{2}\Delta T^2 \\ 0 & 0 & 1 & 0 & \Delta T & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta T \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$
The measurement matrix(C in (2.1)) is:
$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

2.1.3 Point Mass Model

It is trivial to note that vehicles might have varying acceleration which is not satisfied in the previous models, which brings us to the Point Mass Model[**Althoff**] which is similar to the constant acceleration model, except that the acceleration can change upto a certain limit. This model treats the tracked vehicle as a point mass, ignoring wheel-base, slip-angle etc. of the tracked vehicle. The state transition matrix and measurement matrix are same as the constant acceleration model. The acceleration bounds are set as $11.5 m/s^2$ in both x and y-direction.

2.2 Zonotopes as set of state

Zonotopes are gaining fame in the set-based state estimation techniques [**Le2013**], due to its control of wrapping effect and that the Minkowski sum of the zonotope results in zonotopes.

Zonotope are represented by a center, denoted by p and generators, denoted by H. An m-zonotope in $I\!R^n$ can be defined as an affine transformation by H of an m-dimensional hypercube in $I\!R^n$ centered at p. The toolbox, CORA (COntinuous Reachability Analyzer) [**Althoff2018**], is used to construct and utilize zonotopes in the algorithms.

3 Zonotope-based guaranteed state estimation

State Estimation algorithms can be braodly classified into two types: Stochastic and Set-based algorithms. Stochastic state estimation algorithms assume that the uncertainties in the state of the system follow a known probability distributions. It is difficult to fulfill the assumption for such algorithms, however, Zorzi [1] proposed a family of Kalman filters that solves the minimax problem with an iterative probability distribution of the uncertainties.

Set-based algorithms, on the other hand, utilize geometrical sets as domain representation, like ellipsoid or zonotope, to bound the possible sets of state of the system. Zonotopes are better than ellipsoids due to the balance of accuracy and computational cost. Furthermore, zonotopes can control the wrapping effect [2], which is the term referred to the growth of the estimated state due to the propagated uncertainties in each iteration. In addition, sum of zonotopes is also a zonotope (Minkowski sum), which is a desirable property for the techniques.

Set-based algorithms can be further classified into segment intersection and interval observer. The former methods focus on intersecting the set of estimated state with the set of predicted state from the measurements. These methods try to minimize the bounds of the estimated state by using different properties, like volume and radius, of the geometric set. The interval observer methods, on the other hand, design observer to minimize the error on each time step. The following sections dig deeper on each of the aforementioned methods.

3.1 Segment Intersection

The predicted state of the system at a specific time and the previous state of the system are represented by zonotopes. The state estimated is the intersection of these zonotopes. Each algorithm tries to minimize the size of the intersected segment. Different properties of zonotopes, like F-radius, P-radius and volume, are considered to represent the size of the segment. The following sections list and elaborate the algorithm that focus and optimize different properties of zonotopes to minimize the segment.

3.1.1 Frobenius norm of generators

Frobenius norm of the generators of a zonotope is calculated using the formula (3.1) [**Alamo2005**].

$$||H||_F^2 = ||A + \lambda b^T||_F^2 \tag{3.1}$$

$$\lambda^* = \frac{-Ab}{b^T b} = \frac{HH^T c}{c^T HH^T c} + \sigma^2 \tag{3.2}$$

The λ that generates the minimum Frobenius norm of the generators of the intersected zonotope is calculated using the formula (3.2) for each iteration and the minimum zonotope is calculated.

3.1.2 Volume

The volume of a zonotope is calculated using the formula (3.3) [**Alamo2005**].

$$Vol(\hat{X}(\lambda)) = 2^n \sum_{i=1}^{N(n,r)} |1 - c^T\lambda||det(A_i)| + 2^n \sum_{i=1}^{N(n-1,r)} \sigma|det[B_i \quad v_i]||v_i^T\lambda| \tag{3.3}$$

3.1.3 P-radius

The P-radius of a zonotope can be calculated by the formula (3.4) where P is a positive definite matrix.

$$\max_{z\in Z}(||z-p||_P^2) = \max_{z\in Z}((z-p)^T P(z-p)) \tag{3.4}$$

3.2 Interval Observer

Interval Observers need to design observers to minimize the error in the estimation. For the system in (2.1), (3.5) defines the observer, where L is the observer gain to be designed. Although, accuracy is highly expected from these methods, the design of such observers are not very easy. The following section discusses about a method, which uses H-∞ observer design.

$$x_{k+1} = Ax_k + L(y_k - Cx_k) \tag{3.5}$$

3.2.1 H-∞ Observer

The interval observer, proposed in [**Tang2019**], designs the observer gain to minimize the estimation error in each step by using the observer gain as $L = P^{-1}Y$ with P, a positive deifinite matrix with dimension $n_x x n_x$, and Y, a matrix with dimension $n_x x n_y$, both solution to the optimization problem in (3.6).

$$\min_{\gamma_2} s.t. \text{ (3.7)} \tag{3.6}$$

$$\begin{bmatrix} I_{n_x} - P & * & * & * \\ 0 & -\gamma^2 I_{n_w} & * & * \\ 0 & 0 & -\gamma^2 I_{n_v} & * \\ PA - YC & PE & -YF & -P \end{bmatrix} \prec 0 \tag{3.7}$$

4 Evaluations

The INTERACTION Dataset [1] is used to compare the algorithms and models for tracking traffic participants. The dataset contains multiple scenarios in different locations, where each scenario consists of multiple traffic participants. Each traffic participant is identified by an ID for each scenario and each frame per 0.1s has a set of vehicles and their position. The x and y position of the vehicle is noted per time step. The initial state of the system is set using assignments (4.1).

$$\begin{aligned} x_0 &= zonotope([zeros(n), diag([1000; 1000; 10; 10; 10; 10])]) \\ w_k &= [0.1; 0.1; 0.4; 0.4; 0.1; 0.1] \\ v_k &= [0.1; 0.1] \end{aligned} \tag{4.1}$$

4.1 Computation Time

Table 4.1: Comparison of computation time (ms) for 10510 vehicles in an intersection in USA

Method	**Average Computation Time (ms)**
F-Radius	0.375
P-Radius	0.319
H-∞ approximation	0.147

Figure **??** shows that computation time for Volume Minimization rises exponentially making it futile in the state estimation for collision avoidance system. On the contrary, Table 4.1 shows that the computation time for the other methods are negligible compared to the frame rate, i.e 100ms. Furthermore, the interval observer using H-∞ has almost half the time required for the segment intersection methods, although the computation time does not consider the time for pre-computation for the techniques.

[1] https://interaction-dataset.com/

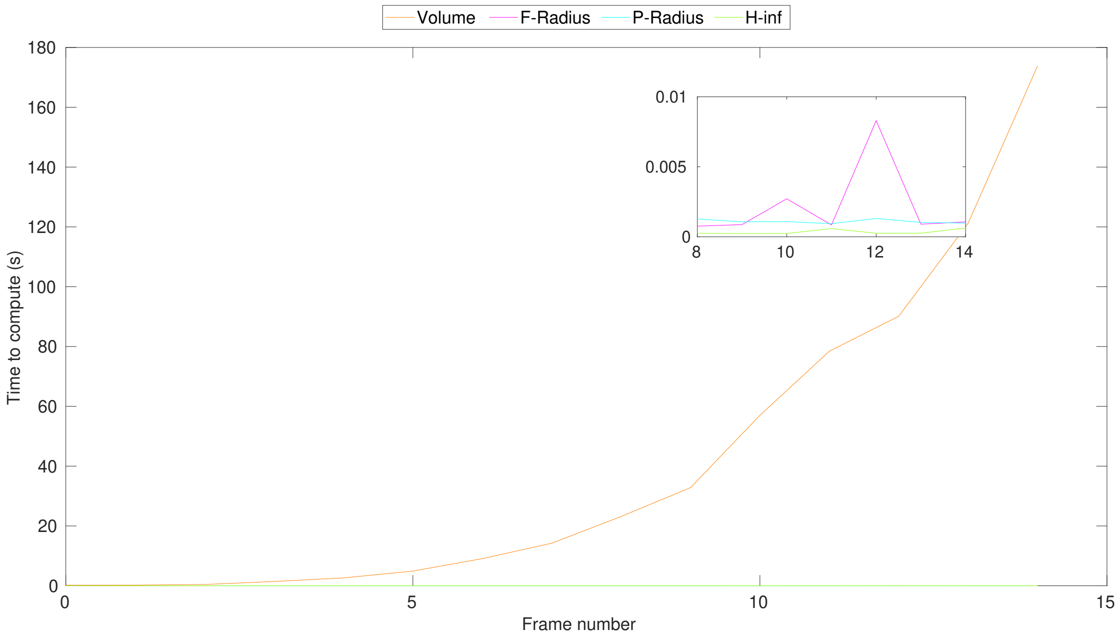

Figure 4.1: Computation time for each method

4.2 Time to Converge

Table 4.2: Comparison of average time(in ms) to converge for unmeasured state

Method	Constant Velocity	Constant Acceleration	Point Mass Model
F-Radius	20	40	12
P-Radius	22	35	25
H-∞ approximation	14	40	25

Table4.2 compares the time for each of the technique to converge unmeasured state(velocity for constant velocity, acceleration for the rest). Segment Intersection using F-Radius works the best using Point Mass Model, as it converges the fastest.

5.1 shows the estimated bounds for all the models on a vehicle with varying acceleration. Comparing the estimation of velocity, same bounds are obtained from constant acceleration and point mass model, however, the bounds of acceleration from the latter is better than the former. Hence, point mass model is used to compare the algorithms for estimating acceleration on the following sections.

4.3 Bounds

Table 4.3: Comparison of bounds of estimation

	Constant Velocity					
Method	x	y	v_x	v_y	a_x	a_y
F-Radius	.441	.441	5.686	5.686	-	-
P-Radius	.4606	.459	13.45	13.45	-	-
H-∞ approximation	.9867	.937	6.177	6.177	-	-
	Point Mass Model					
	x	y	v_x	v_y	a_x	a_y
F-Radius	.5713	.5075	8.461	8.461	15.79	15.78
P-Radius	.4598	.4523	16.39	16.39	16.43	16.18
H-∞ approximation	1.5	1.5	9.414	9.414	16.11	16.24

Bounds using constant velocity model is tighter compared to point mass model as seen from Table 4.3. The segment intersection using F-Radius has the tighter bounds compared to the other techniques. Interesting to note, the $H-\infty$ has much higher bounds in the initial time steps.

4.4 Accuracy

The boxplot using measurements in velocity using Point Mass Model for all the techniques are shown in Figure **??**. The error bound in lowest in H-∞, however, segment minimization using F-Radius has a lower mean error.

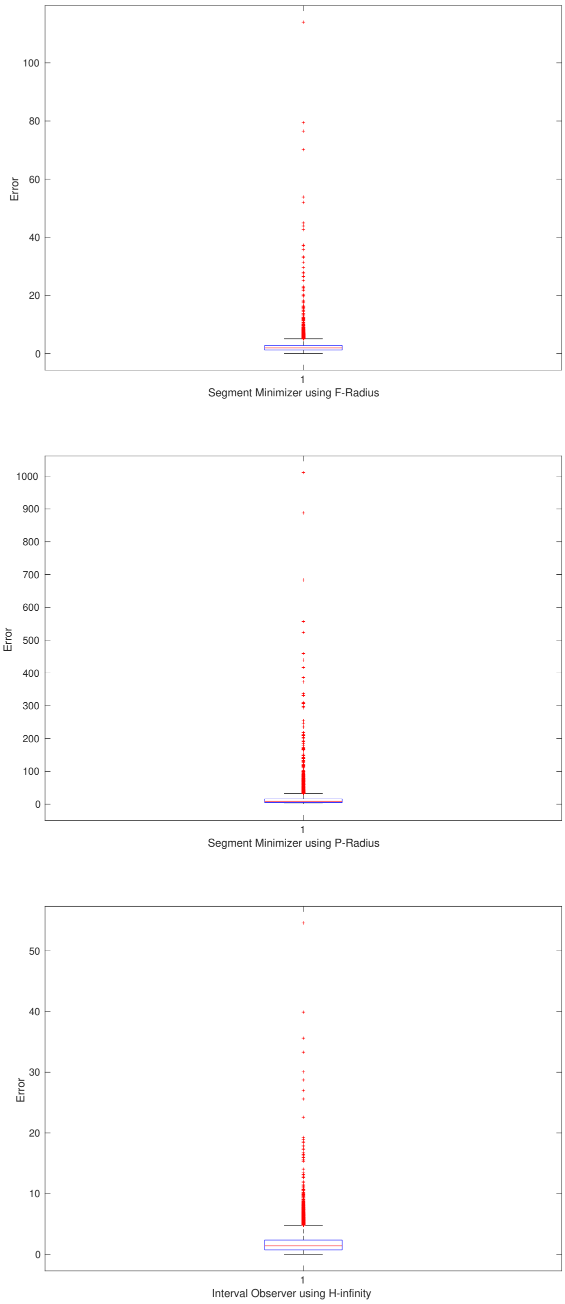

Figure 4.2: RMSE(Root Mean Squared Error)

5 Conclusion

A demand for intelligent collision avoidance system is timeless. To take load of sensors and hardware of a vehicle, state estimation algorithms can be used to track vehicle and estimate properties required for collision-free path prediction. On comparing multiple techniques using different models to represent the tracked vehicle, it can be concluded that the segment minimization using F-Radius is the most accurate. However, in terms of computation time, the H-∞ overrules other methods. Further developments can include implementing the technique in real vehicle system and use the estimation to track vehicles.

6 Extended Results

6.1 Set Estimation

6.1.1 Segment Minimization using F-Radius

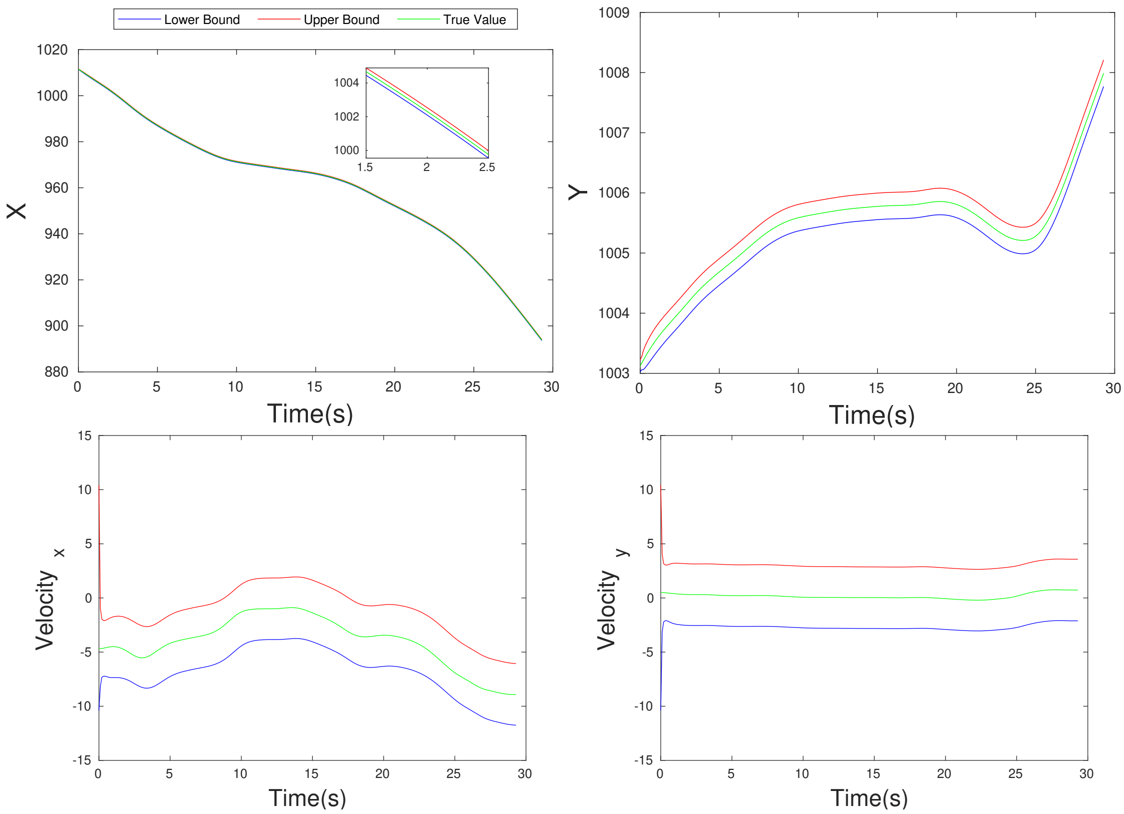

Figure 6.1: Estimation using Constant Velocity

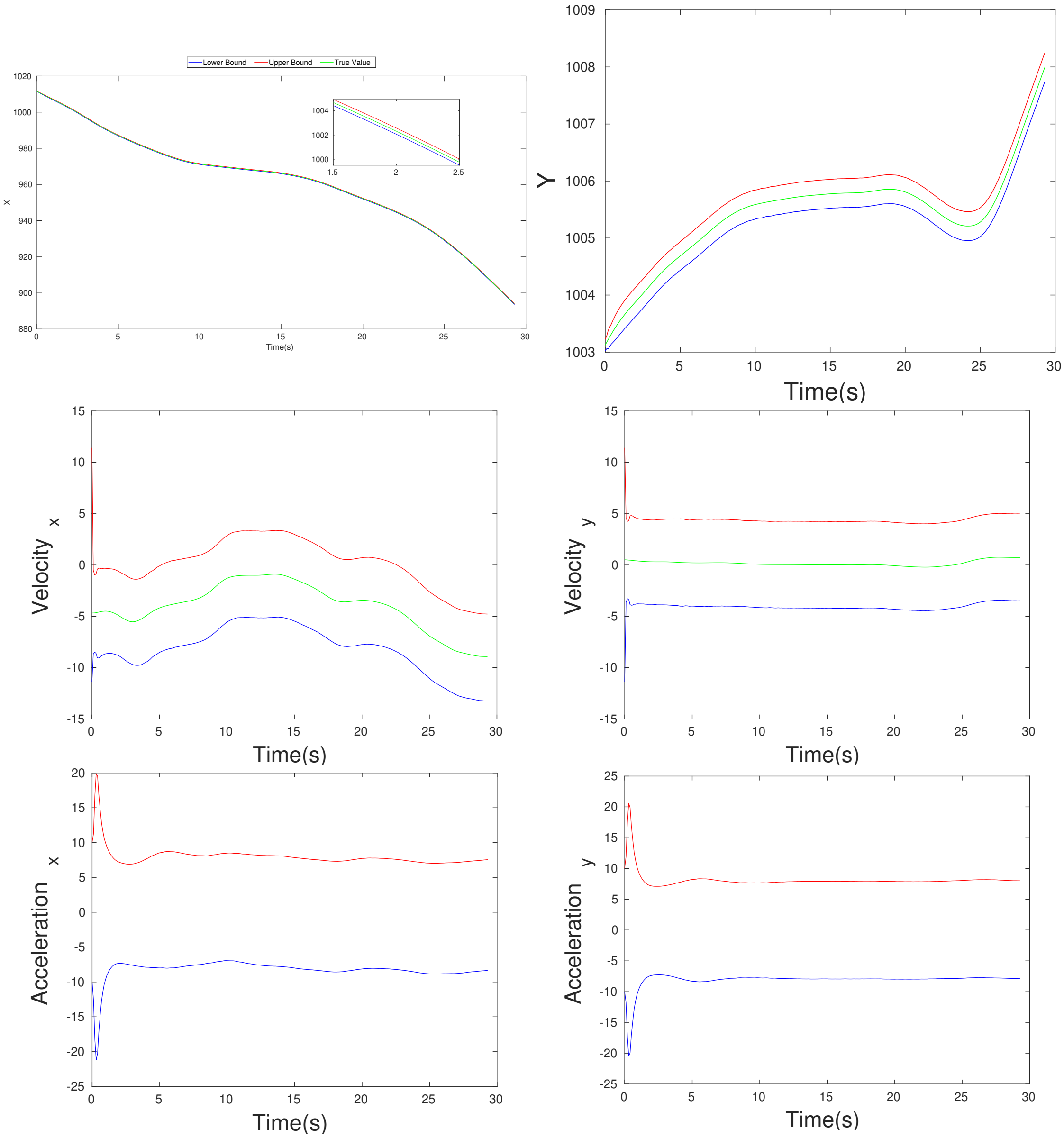

Figure 6.2: Estimation using Constant Acceleration

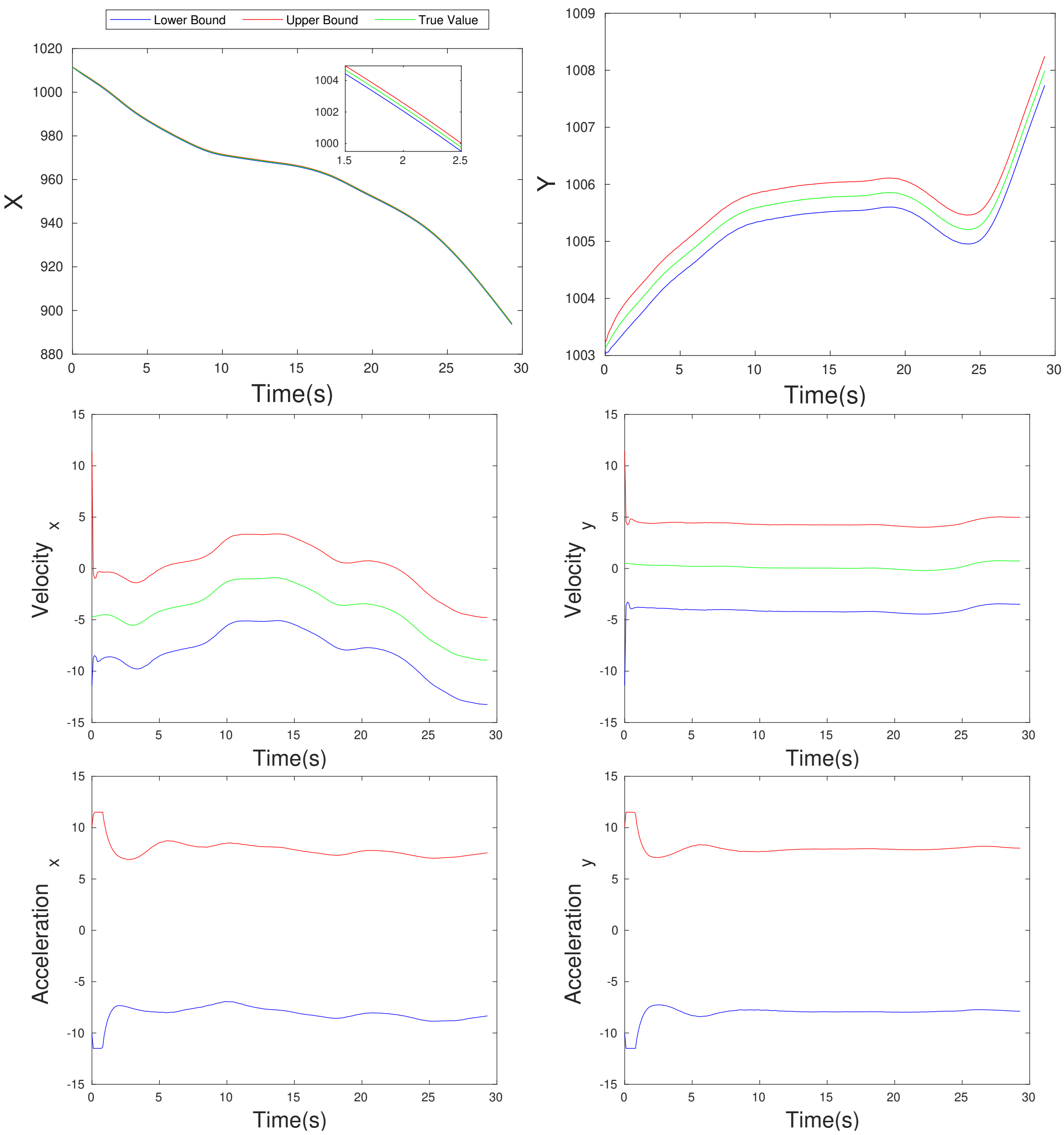

Figure 6.3: Estimation using Point Mass Model

6.1.2 Segment Minimization using P-Radius

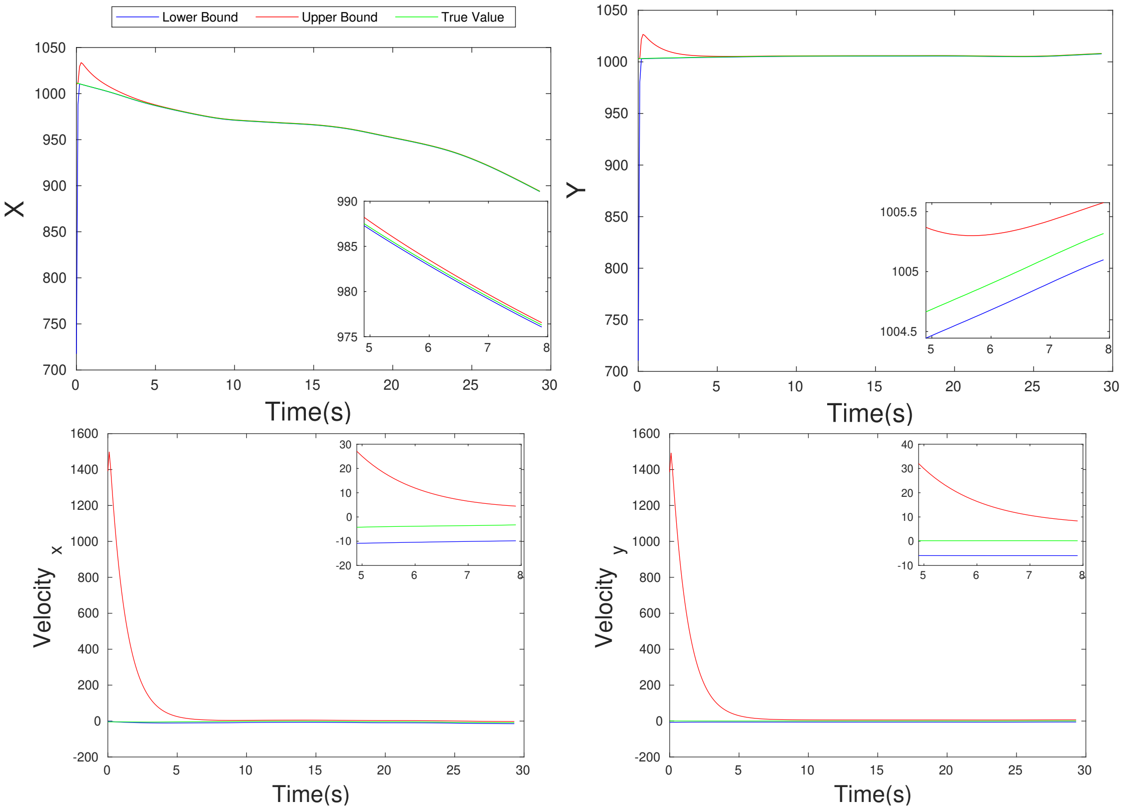

Figure 6.4: Estimation using Constant Velocity

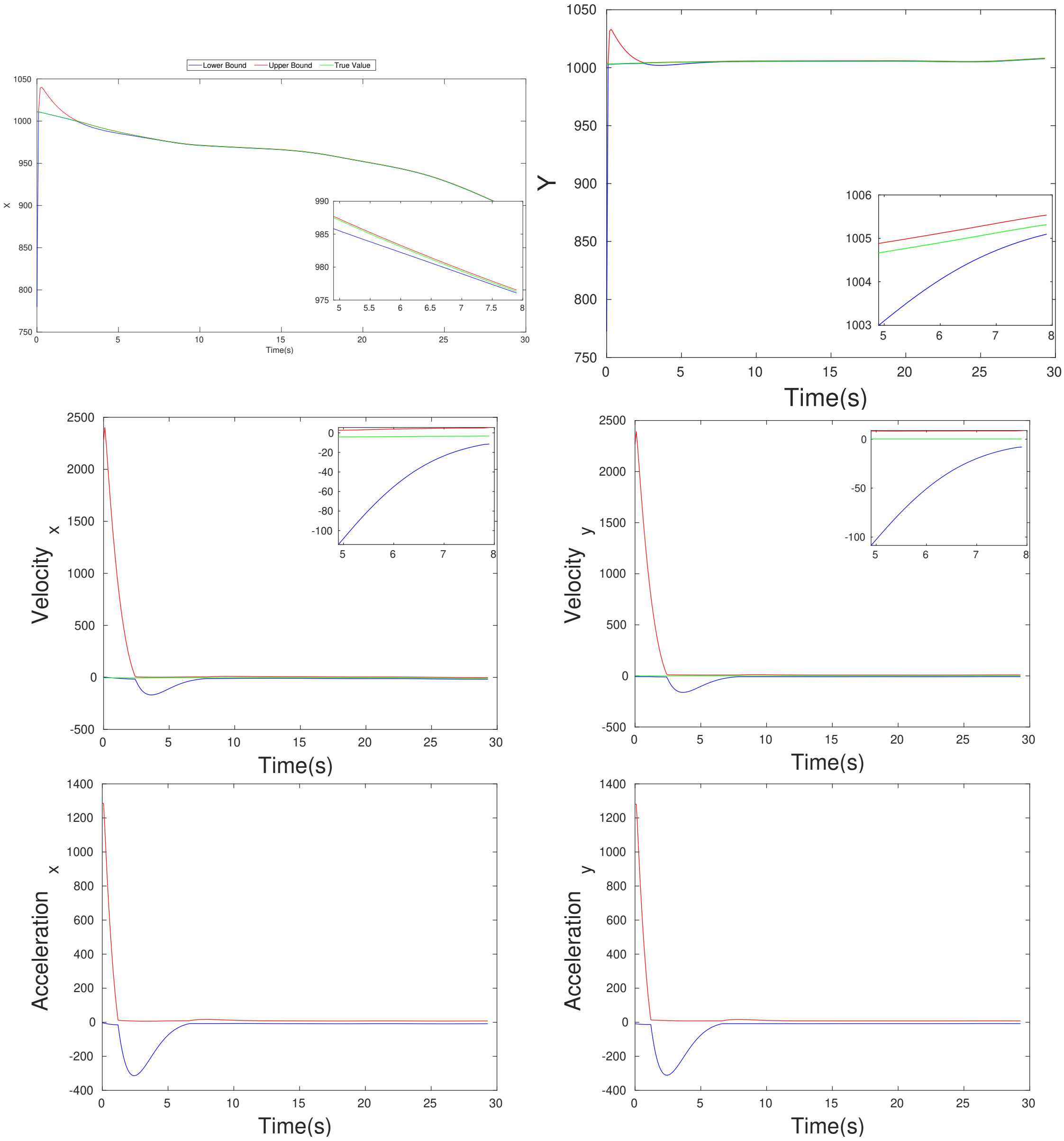

Figure 6.5: Estimation using Constant Acceleration

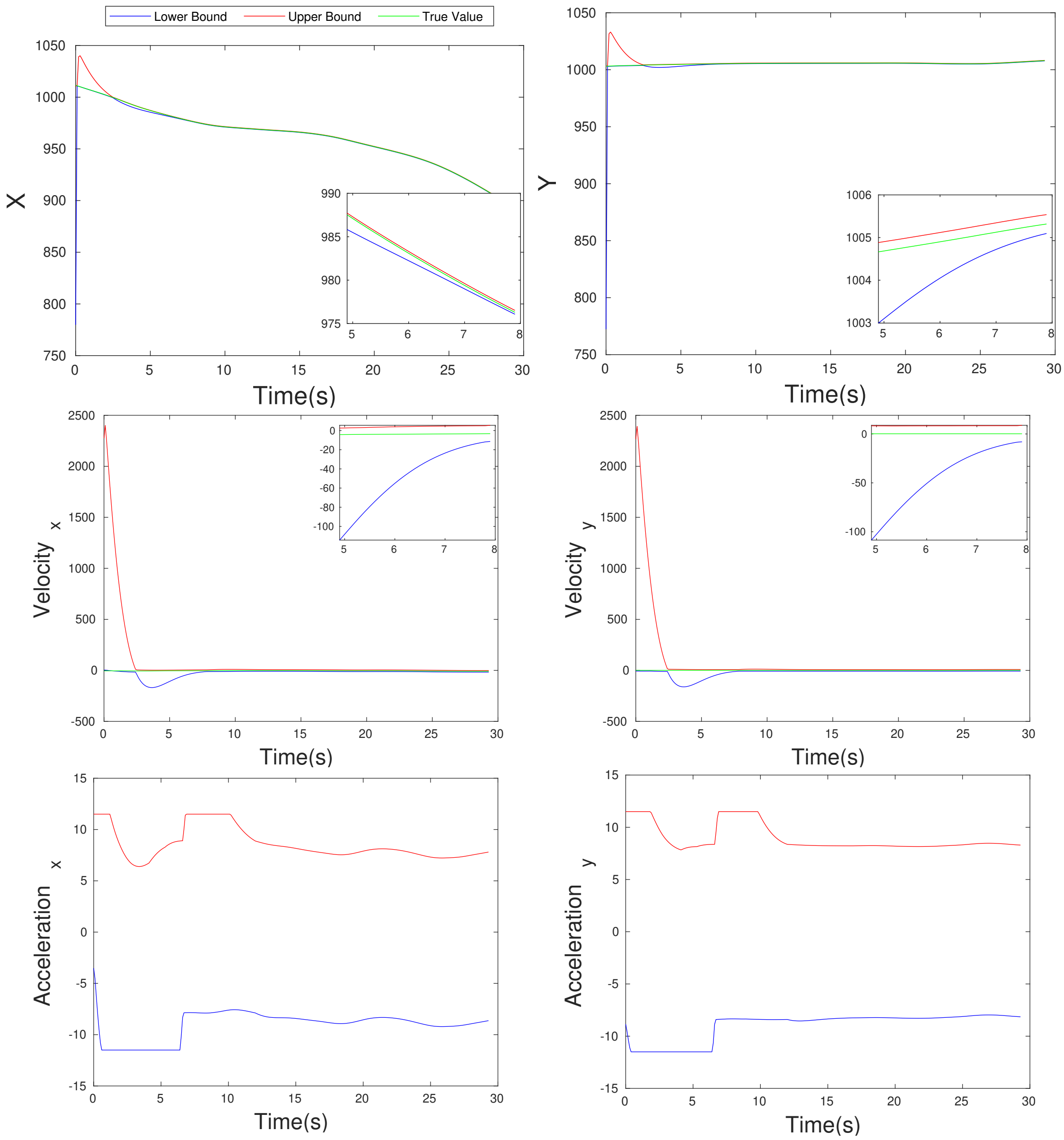

Figure 6.6: Estimation using Point Mass Model

6.1.3 Segment Minimization using H-∞

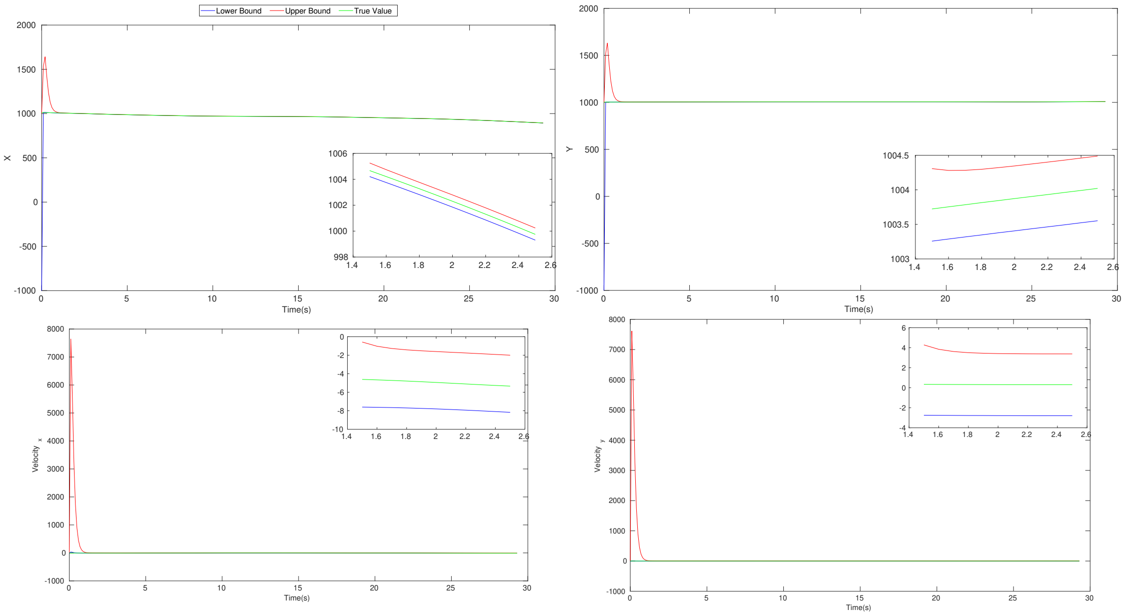

Figure 6.7: Estimation using Constant Velocity

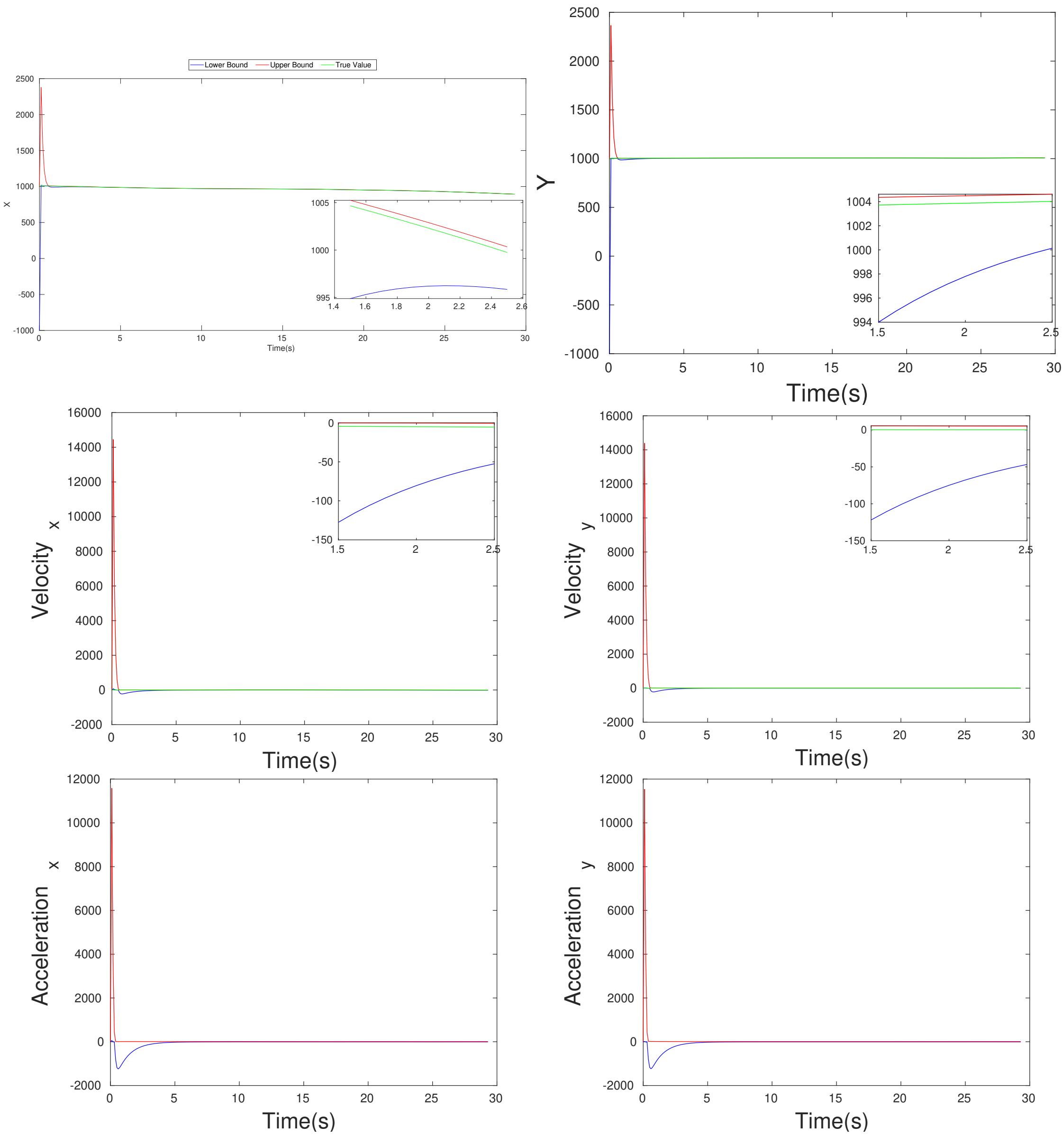

Figure 6.8: Estimation using Constant Acceleration

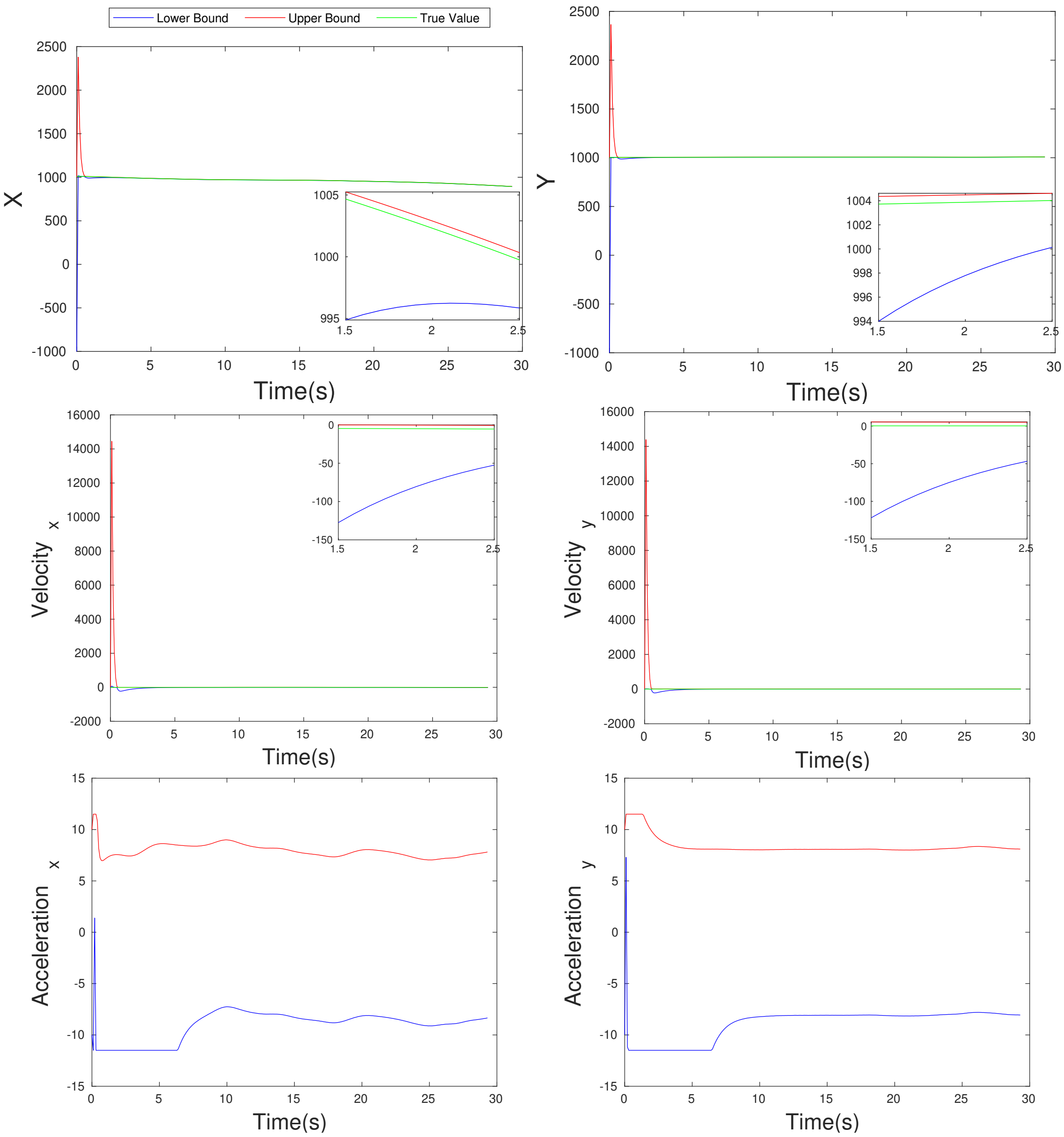

Figure 6.9: Estimation using Point Mass Model

6.2 Rate of Change of Bounds

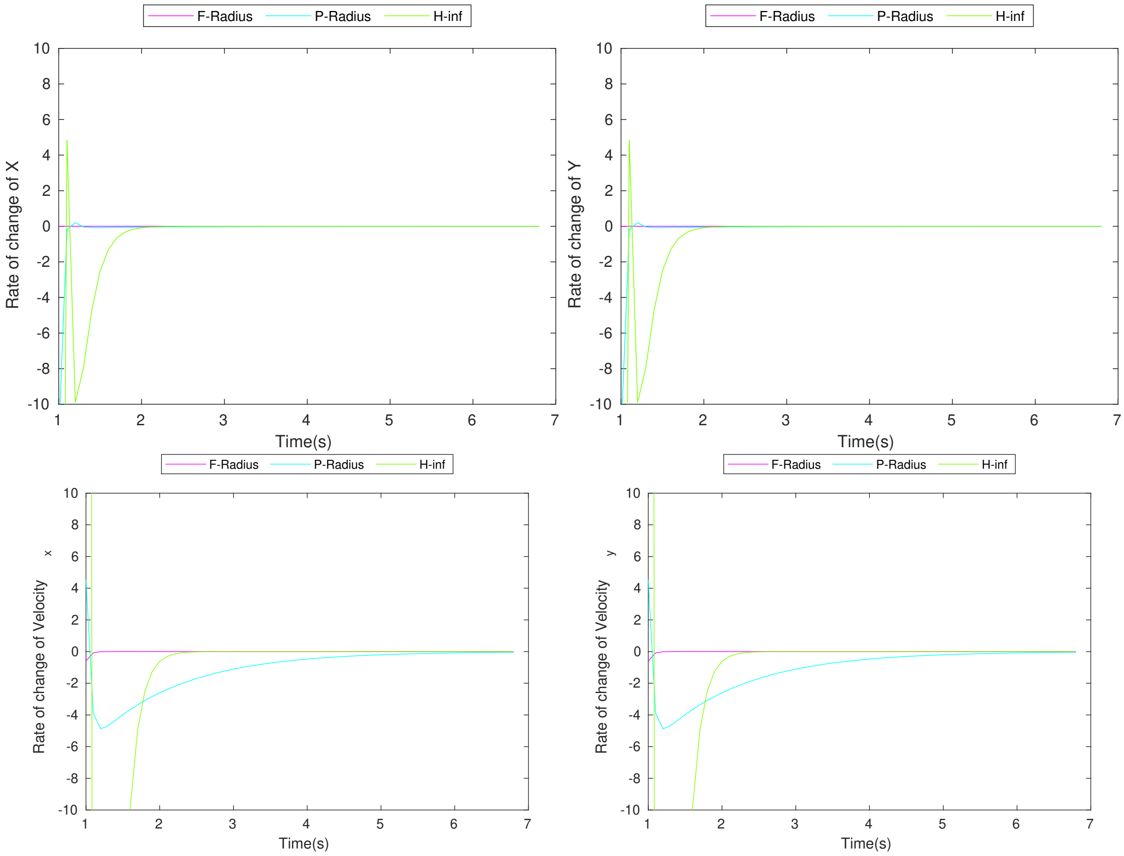

Figure 6.10: Rate of change of bounds using Constant Velocity Model

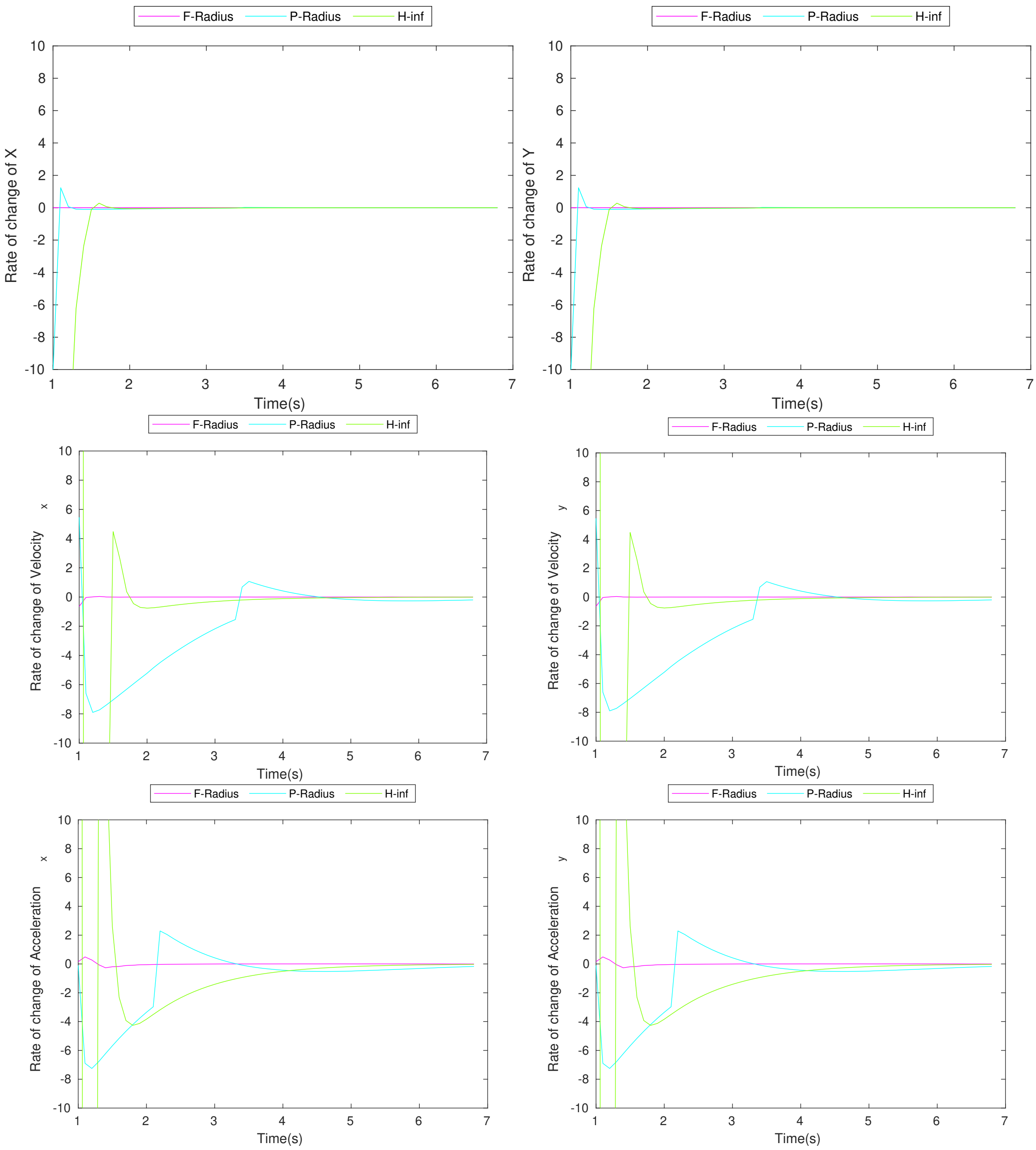

Figure 6.11: Rate of change of bounds using Constant Acceleration Model

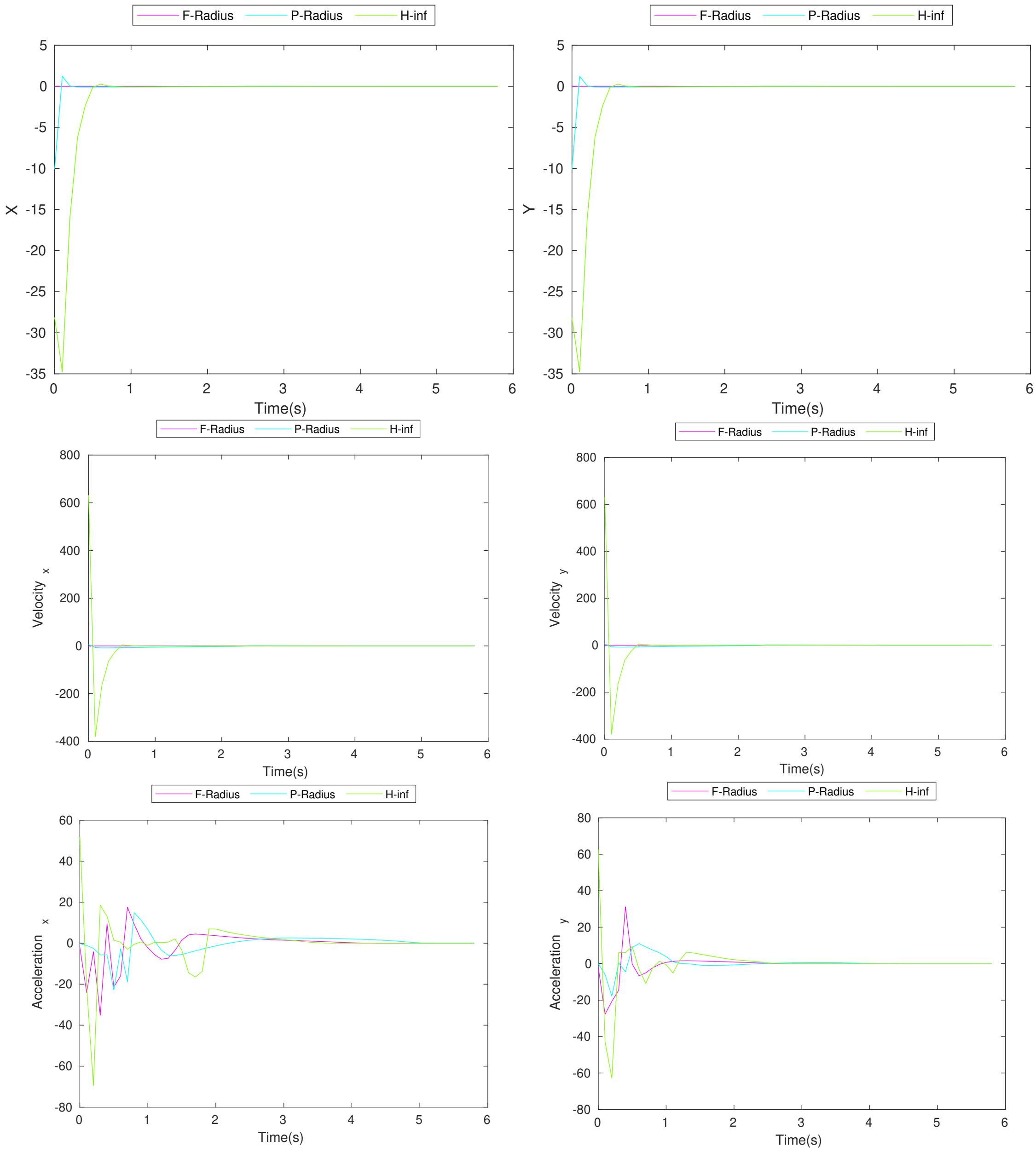

Figure 6.12: Rate of change of bounds using Point Mass Model

List of Figures

List of Tables

Bibliography

[1] M. Zorzi. "Robust Kalman filtering under model perturbations." In: *IEEE Transactions on Automatic Control* 62.6 (2017), pp. 2902–2907. ISSN: 15582523. arXiv: 1508.01906.

[2] W. Kühn. "Rigorously Computed Orbits of Dynamical Systems without the Wrapping Effect." In: *Computing (Vienna/New York)* 61.1 (1998), pp. 47–67. ISSN: 0010485X.