Modeling and Control for Vision Based Rear Wheel Drive Robot And Solving Indoor SLAM Problem using LIDAR

Xianglong Lu

Chair: Dr. Armando Antonio Rodriguez
Dr. Spring Berman
Dr. Panagiotis Artemiadis

Arizona State University
July 19th 2016

Outline

IRA A. FULTON SCHOOLS OF engineering ARIZONA STATE UNIVERSITY

- Problem Statement & Contributions
- Hardware: Low Cost Self-Designed Robotic Vehicle
- Modeling & Control of Rear-Wheel Drive Robot
- Perform SLAM (Simultaneous localization and mapping)
- Demonstrations
- Summary and Directions for Future Research

Literature Survey: State of Field Use

1. Rear wheel drive robot TITO LTI model (Marino, et.al. 2007) – basis for both decoupled longitudinal and lateral plant

2. Vision based complete lateral model of RWD vehicle (Jana Kosecka, 1996) – vision based lateral dynamics and vision based outer loop design

3. Image processing algorithm in opencv2 (Bradski G, Kaehler A, 2008) – camera used to get directional information(8HZ, 320×240) or a USB camera (4.5Hz, 640×480)

4. ROS architecture and API (Morgan, et al. 2009) – basic introduction of the open source robot operation system I was using (ROS, Robot Operation System)

5. Hector Mapping, SLAM relies only on LIDAR scan data (Giorgio, et al. 2005) – EKF, Main algorithm implemented

6. Gmapping, SLAM relies on both odometry (encoder and IMU) and LIDAR scan data (SLAM for Dummies, Soren, et al.) – Extended Kalman Filter (EKF) is used to estimate the state of the robot from odometry data and landmark observation

Contributions

- General *FAME* architecture
- Self designed rear wheel drive multi-capability ground vehicle
- Modeling and control trade studies
- Inner loop $(v\ ,\omega)$ control
- Speed-directional outer loop (v, θ) control
- Planar $(x,\ y)$ Cartesian Stabilization
- Vision based outer loop (v, θ) control
- Line tracking performance study with:

  (1) Different cruise speed v_x

  (2) Different camera fixed look-ahead distance L

  (3) Different delay from vision subsystem T_d
- Manually remote controlled robot to perform indoor SLAM
- Autonomously line guided robot to perform indoor SLAM.

Motivation

Sensing / Monitoring

Foundations of Communications

Cooperative Planning & Control

Robots in the Market

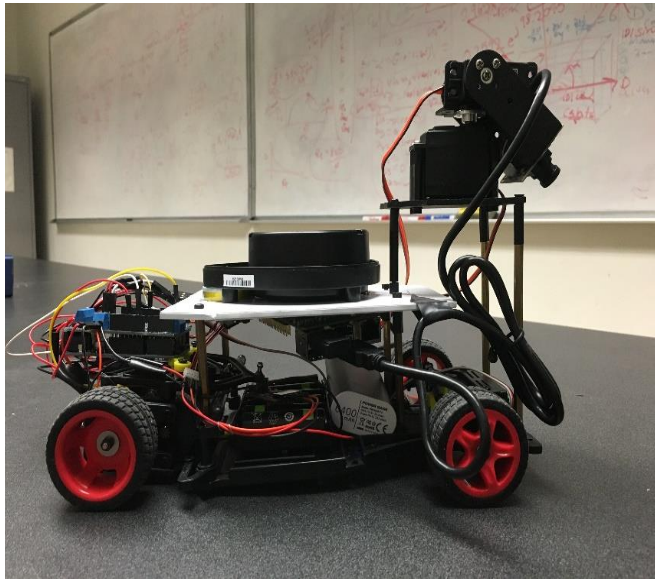
Pioneer 3 DX

- mapping
- teleoperation
- localization
- monitoring
- reconnaissance
- vision
- manipulation
- autonomous navigation
- multi-robot cooperation and other behaviors
- general robotics

$ 4000 **Pioneer 3 DX**

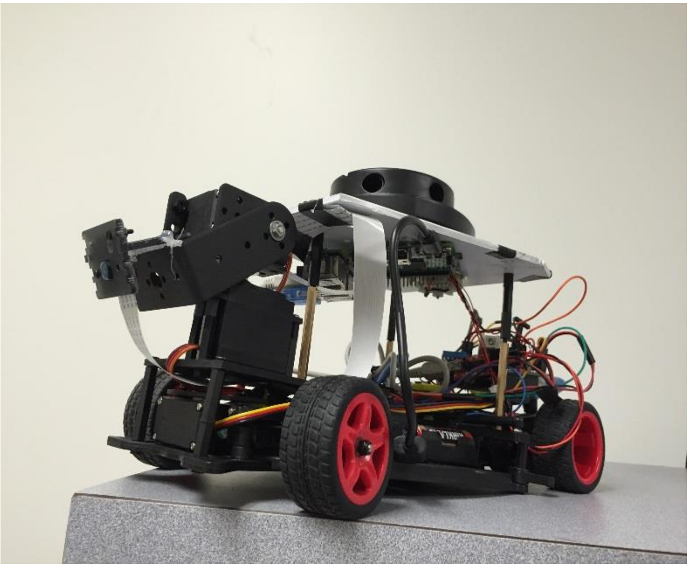
Powerful but Expensive

Robots (Different Styles and Modes)



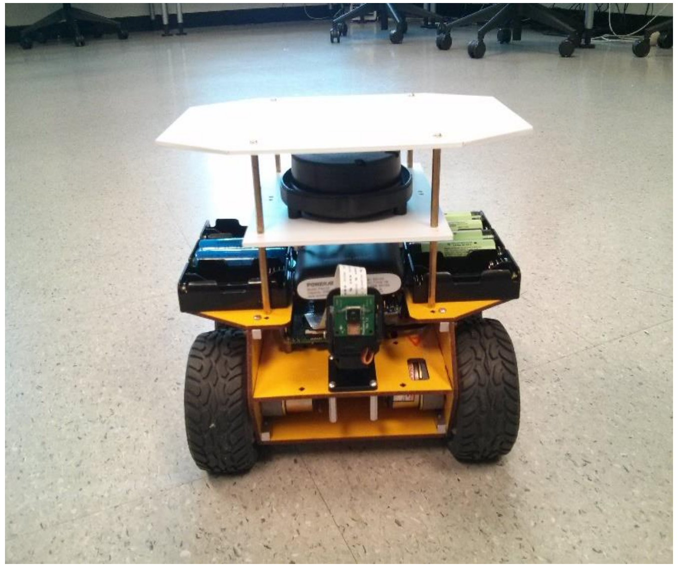
FreeSLAM Robot: Vision Mode

Rear Wheel Drive, UAV Tracking, Camera vision sensing, Depth sensors



FreeSLAM Robot: LIDAR Mode

High Accuracy LIDAR Sensing, Fixed Pan Servo, Less Speed for not Losing Landmarks

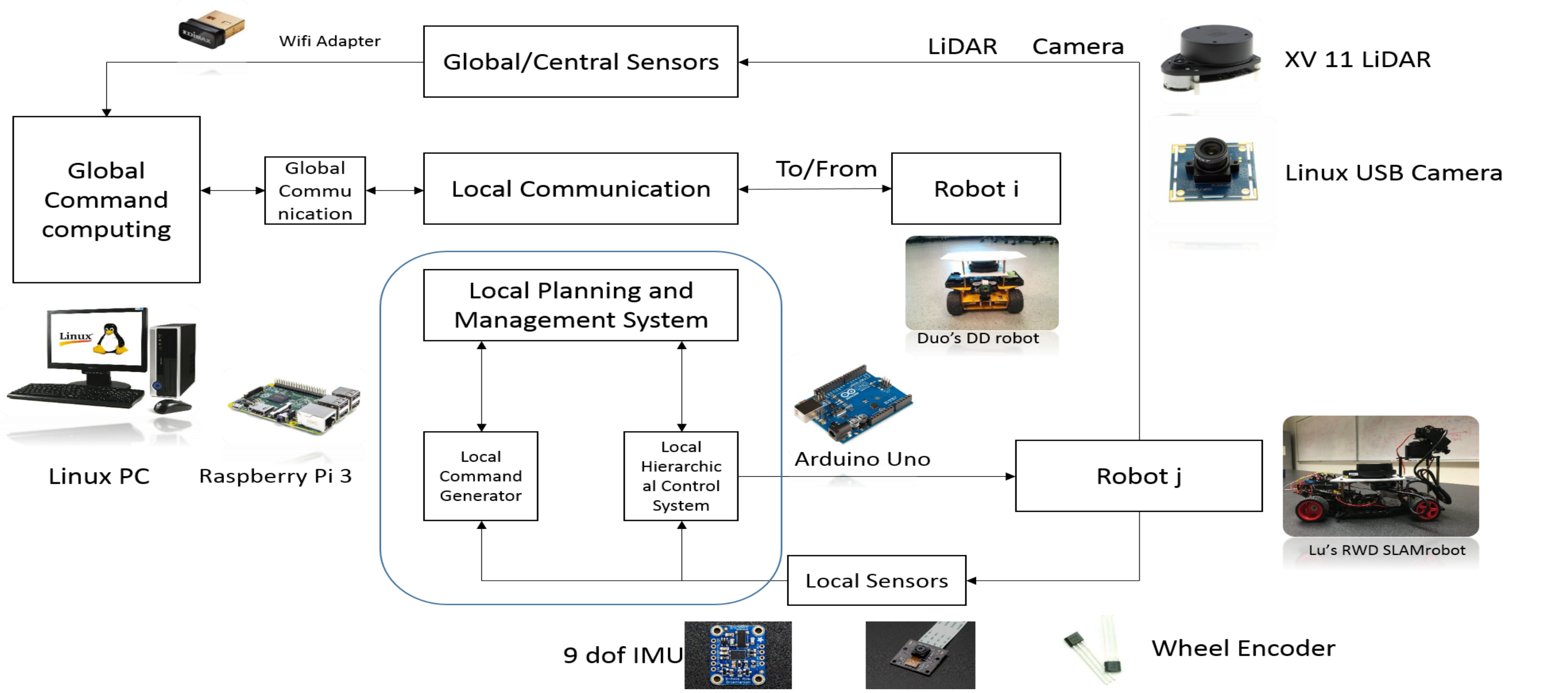


Duo Lv's Robot: Rigid Mode

Differential Drive, UAV landing, Less Speed, More Rigid, Easy Turning

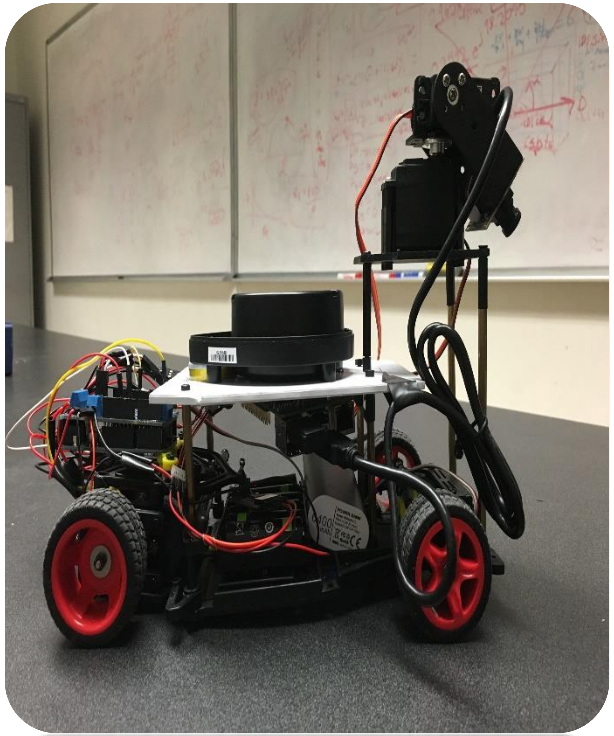
FAME Architecture

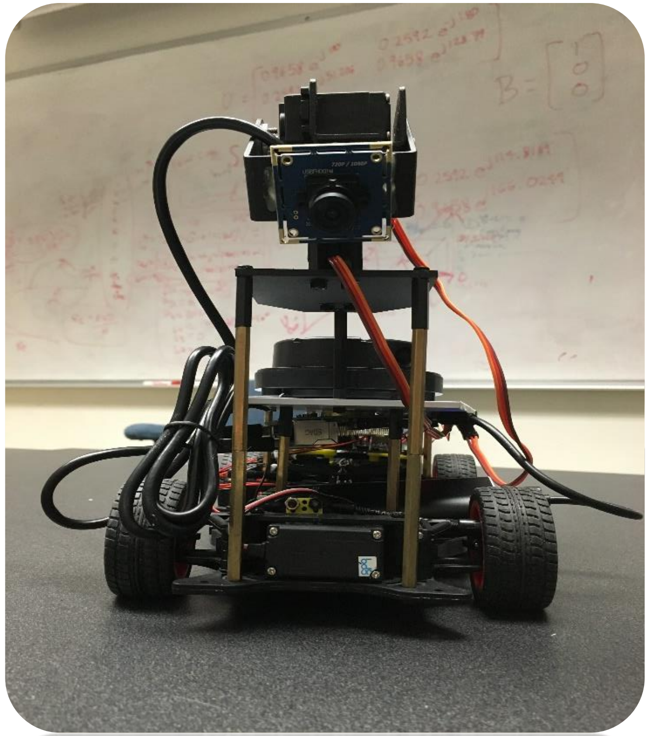
- *Flexible Autonomous Machines operating in an uncertain Environment*
- Candidate system-level architecture for a fleet of robotic vehicles



Hardware

Enhanced FreeSLAM Robot





Component	Price
Chassis and Motors	$180
Futaba S3003 Servo	$10
Arduino Uno	$25
Adafruit Motor Shield	$20
Raspberry Pi 3	$40
WiFi adapter	$25
Adafruit 9DOF IMU	$20
Pi camera	$20
Neato xv11 LIDAR	$80
5V external battery for Raspberry Pi	$20
Hitachi 18650 battery for motor	$30
Total Price	**$470**

Robot Nominal Parameter Values and Characteristics

Table 1.2: FreeSLAM Robot Nominal Parameter Values and Characteristics

Parameters	Definition	Nominal Values
m	Fully Loaded Mass	1.47kg
m_0	Mass (Not Loaded)	0.83kg
I	Moment of Inertia (Estimated using Cube)	0.0015kgm2
r	Wheel Radius	0.024m
d_w	Distance Btw 2 Rear Wheels	0.134m
L_a	Armature Inductance	0.2mH (neglected)
R_a	Armature Resistance	2.523Ω
K_b	Back EMF Constant	0.004V/(rad/sec)
K_t	Torque Constant	0.004Nm/A
v_{max}	Max. Observed Speed (Enhanced Vehicle)	5m/s
v_{max0}	Max. Observed Speed (Original vehicle)	7.2m/s
e_{amax}	Max. Motor Voltage	7.2V
a_{max}	Max. Accel. (Enhanced)	3.2m/sec2
$\omega_{wheelmax}$	Max. Angular Vel. (Enhanced)	208.3 rad/sec

Hardware Limitation

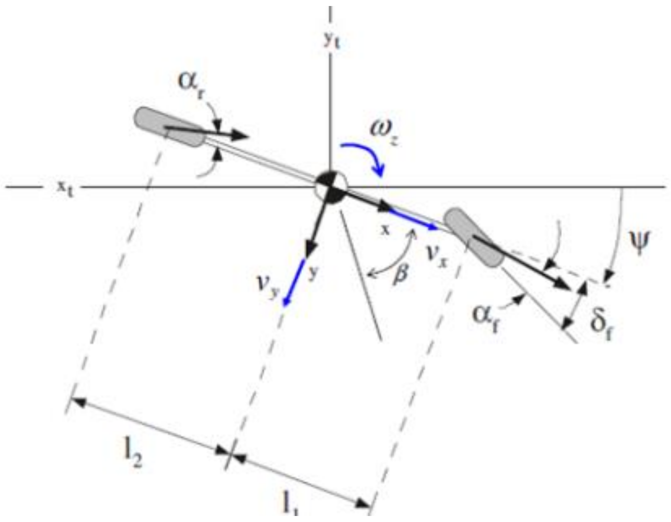
Sensors/Actuators/ Software	t (sec)	ω (rad/s)	Bandwidth Limitations (factor of 10 rule)
Arduino ZOH ½ sample delay	0.05	$\frac{2}{\Delta} = 40$	4 rad/s
Arduino DA/AD	0.1	60	6 rad/s
Image Processing	0.133	47.1	4.7 rad/s
Wheel Encoders	0.0131 v	479.4 v	4.79 v rad/s
BNO055 9 dof IMU	0.01	600	60 rad/s

Inner Loop Bandwidth is limited by 4 rad/s

Rear Wheel Drive Robot – State Space Representation
(Marino, et.al. 2007)

$$\begin{bmatrix} \dot{v}_x \\ \dot{v}_y \\ \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \frac{-2v_x c_a}{m} & 0 & 0 & 0 \\ 0 & -\frac{c_f+c_r}{mv_x} & 0 & -v_x + \frac{c_r l_r - c_f l_f}{mv_x} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{-l_f c_f + l_r c_r}{I v_x} & 0 & -\frac{l_f^2 c_f + l_r^2 c_r}{I v_x} \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \psi \\ \dot{\psi} \end{bmatrix} + \begin{bmatrix} \frac{1}{m} & 0 \\ 0 & \frac{c_f}{m} \\ 0 & 0 \\ 0 & \frac{l_f c_f}{I} \end{bmatrix} \begin{bmatrix} F \\ \delta_f \end{bmatrix} \quad y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ \psi \\ \dot{\psi} \end{bmatrix}$$

Decoupled
TITO LTI System



$$P_{long} = \frac{V_x}{F} = \left[\frac{0.6803}{(s+1.116)}\right]$$

➢ (Analysis in next slide)

Equilibrium cruise speed of $v_e = 0.1m/s$:

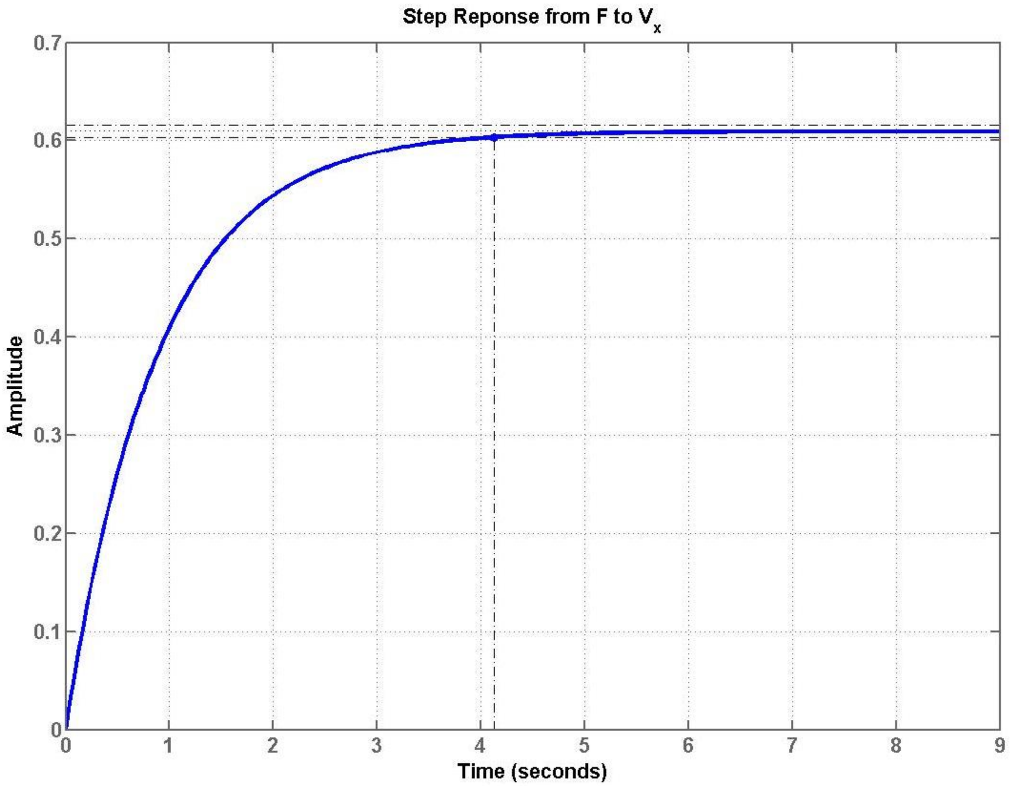
$$P_{Lateral} = \frac{\dot{\psi}}{\delta_f} = \frac{0.368(s+0.484)}{(s+1.007)(s+0.457)}$$

Why This Calculated Numerical Model is Not Quite Accurate

$$P_{long} = \frac{b}{s+a}$$

$$P_{long} = \frac{V_x}{F} = \left[\frac{0.6803}{(s+1.116)}\right]$$

- $t_s = \frac{5}{a}$ (1%)= 4.48s
- $\frac{y_{ss}}{e_{ss}} = \frac{b}{a} = \frac{0.6803}{1.116} = 0.61$



- a = 1.116
- b = 0.6803

$I = 0.0015 kg \cdot m^2$ (car is estimated as a cube)

$c_f = c_r = 0.0368\ N/rad$ (estimated wheel rotary stiffness)

Why model is not quite accurate:

- Inaccurate c_f, c_r and I
- Static friction

Robot Motor Parameter Estimations

DC Motor Transfer Function
(From input voltage to angular velocity)

$$\frac{\Omega(s)}{U_a(s)} = \frac{K_t}{L_a J s^2 + s(L_a B + R_a J) + K_e K_t + R_a B}$$

Known the DC motor model is RN 260-C

- $L_a = 0.2mH$ (Armature Inductance)
- R_a: Armature Resistance

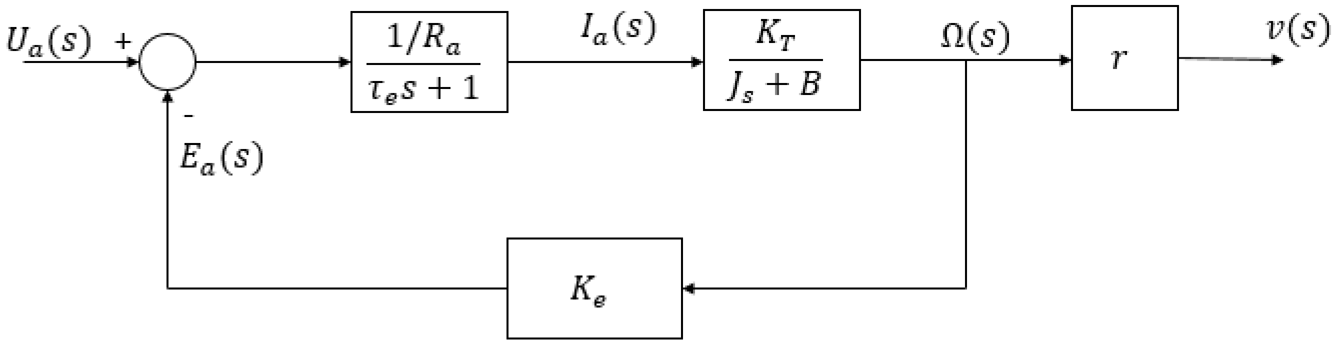
$$U_a = E_a + I_a R_a$$
$$P_1 = U_a I_a = 1.07\text{A} \times 4.5V = 4.815W$$
$$P_M = E_a I_a$$
$$R_a = \frac{P_1 - P_M}{{I_a}^2} = 2.523\Omega$$

Table 2.1: RN 260 Motor Dynamics

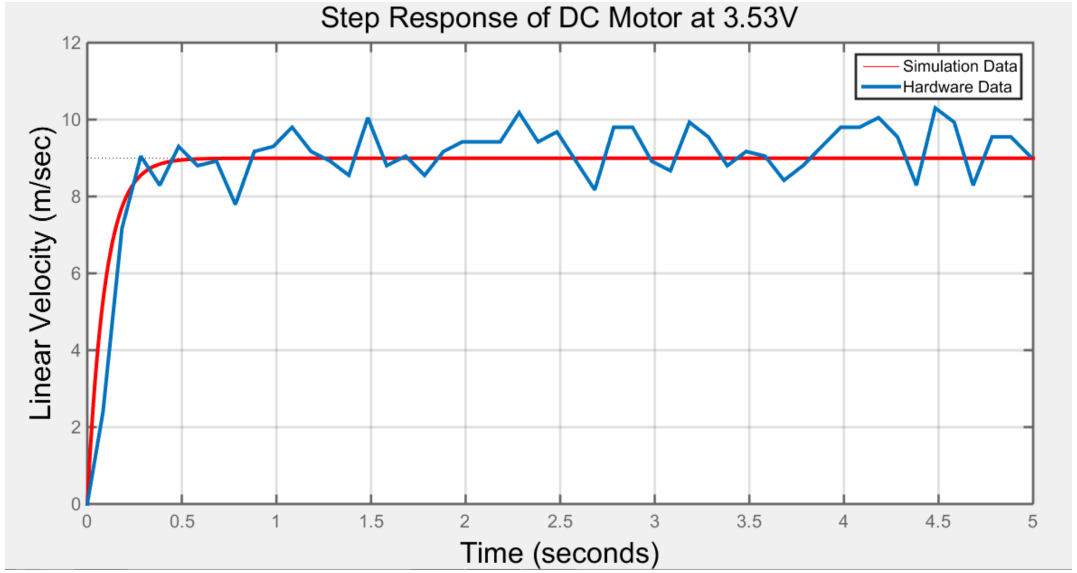
	Current (A)	Speed (rpm)	Torque (g*cm)	Voltage (V)
No Load	0.13	10000	0	4.5
Max Efficiency	0.51	7950	18	4.5
Max Output	1.07	5000	44	4.5
Stall	2	0	88	4.5

- K_t : motor torque constant
- K_e : motor back EMF constant
- J is moment of inertia of the motor shaft-load system
  $J = 2.96 \times 10^{-6}\ kg \cdot m^2$
- B is load-motor speed rotational damping constant
  $B = 4.3 \times 10^{-5}\ Nms$

DC Motor Dynamics



$$P_{motor} = \frac{v(s)}{e_a(s)} = \frac{27.1}{s + 10.64}$$

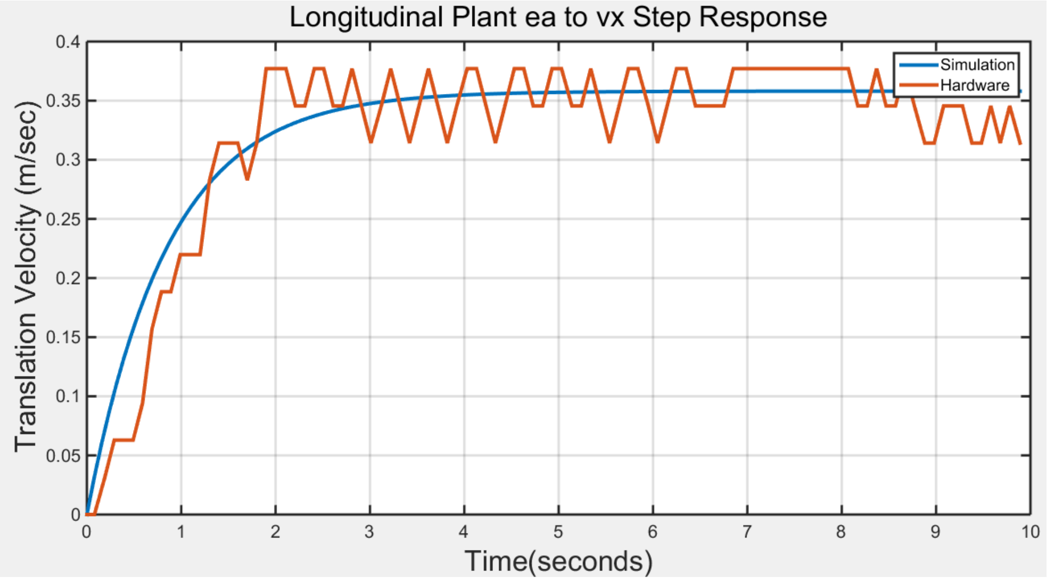


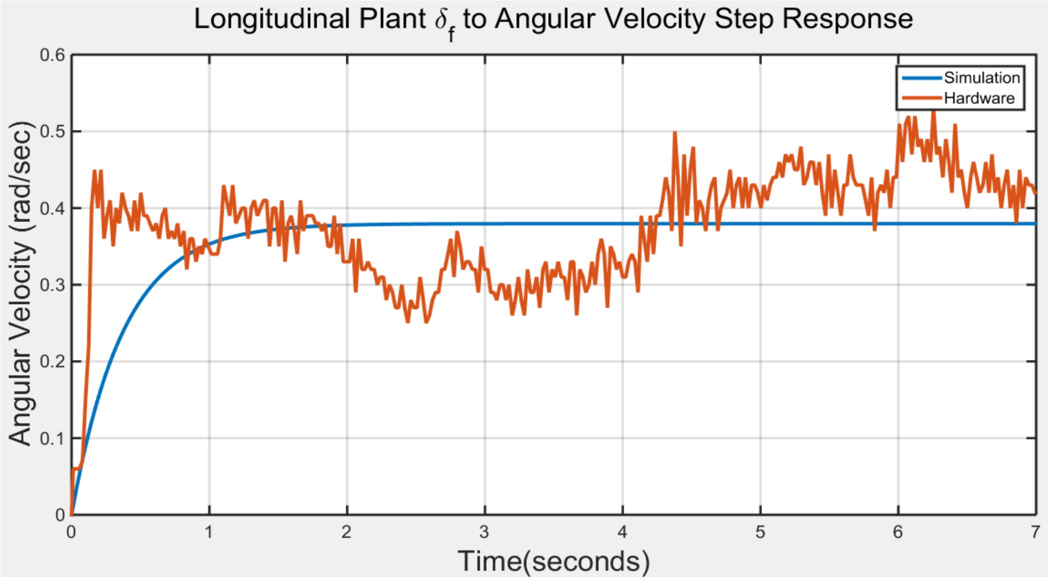
Step Response of DC Motor with

Motor input voltage is 3.53 V

- Step Response Ripple: 2.4 m/sec

On Ground Longitudinal and Lateral Model





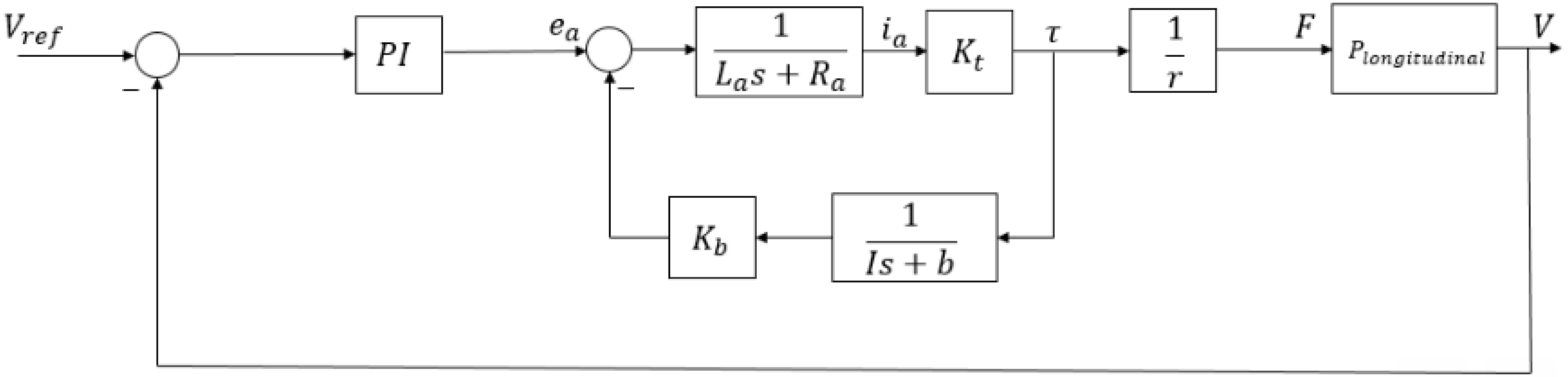
$$P_{long} = \frac{V_x}{e_a} = \frac{0.3274}{s + 1.176}$$

$$P_{lateral} = \frac{\dot{\psi}}{\delta_f} = \frac{2.892}{s + 2.659}$$

- Step Response Ripple: 0.06 m/sec
- Step Response Ripple: 0.27 rad/sec

Encoder is used to get linear velocity while IMU BON055 is used to get angular velocity information

Longitudinal Inner Loop PI Controller Design



PI controller: g = 11.68 z = 2.02

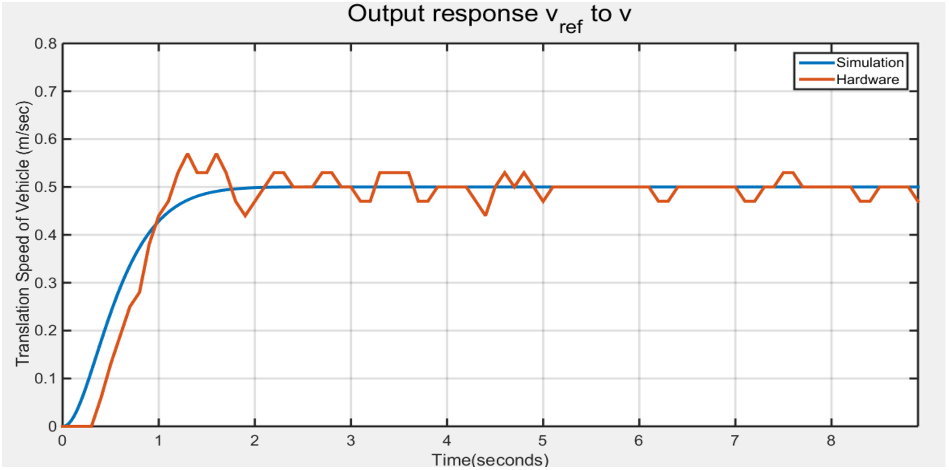
$$T_{ry} = WPK(1+PK)^{-1}$$

$$T_{ry} \xRightarrow{V_{ref}\ to\ V} \frac{7.716}{s^2+5s+7.716}$$

- Settling time t_s is set to 2 seconds
- Damping ratio ζ is set to 0.9

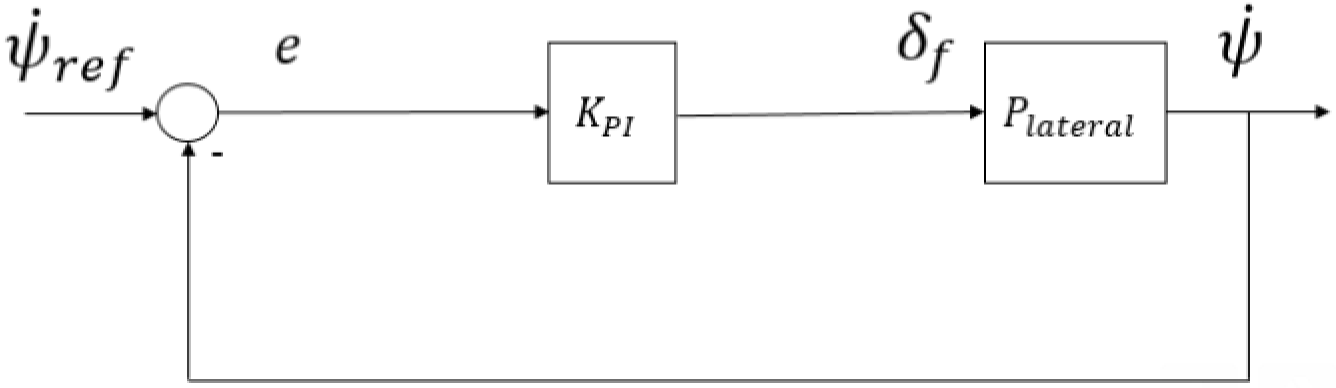
In this case

- ω_n is set to 2.78 rad/s
- Overshoot is 0.15%



Ripple: 0.06m/s

On Ground Lateral Inner Loop PI Controller Design



$\dot{\psi}_{ref}$ is desired angular velocity
δ_f is commanded front wheel steer angle

To design this PI controller

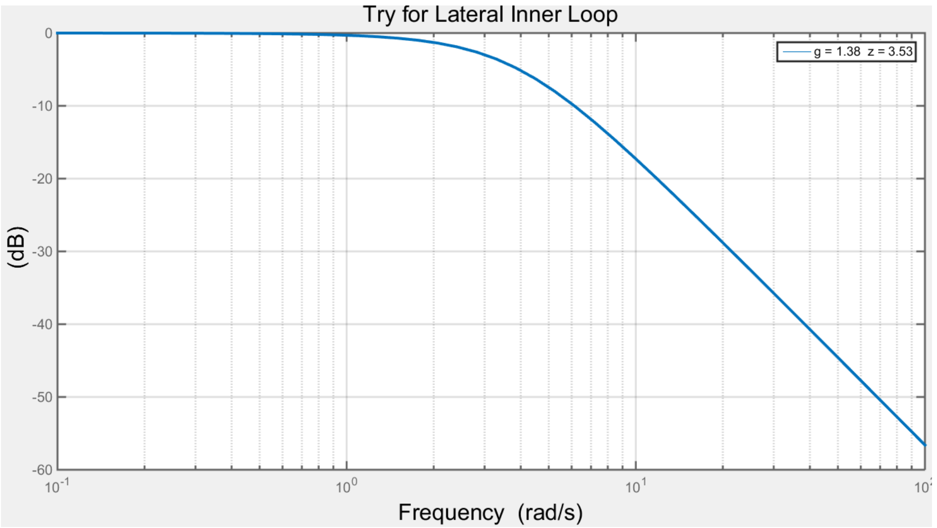
- Set settling time t_s to 1.5s
- Set damping ratio ζ to 0.886

In this case

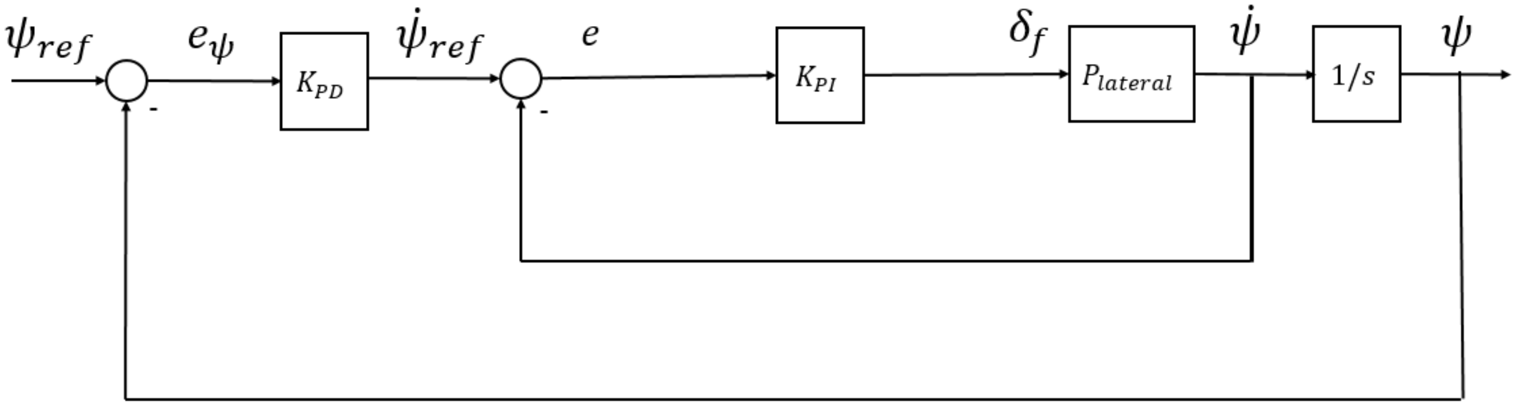
- ω_n is set to 3.8 rad/s
- Overshoot is set to 0.4%

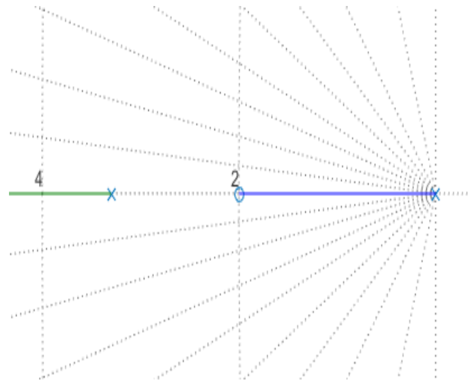
Then we have the PI controller: g = 1.38 z = 3.53

$$T_{ry} = WPK(1 + PK)^{-1} \qquad T_{ry} = \frac{14.8}{s^2 + 6.67s + 14.8}$$



Lateral Outer Loop PD Controller Design



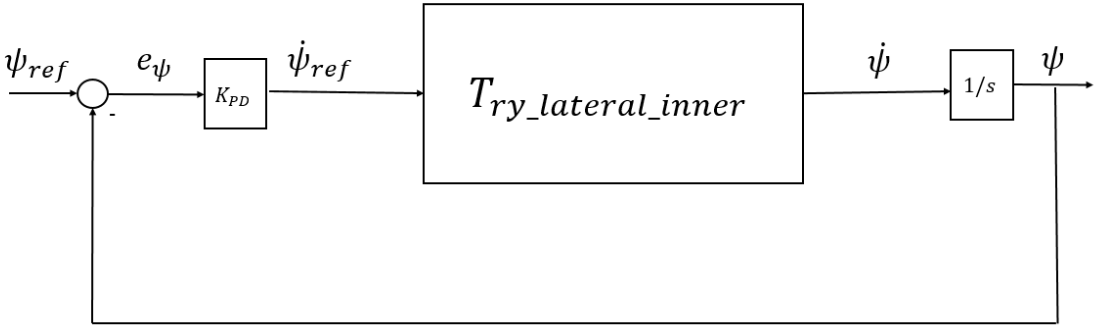


From system estimation aspect:

P_{outer} can be estimated as a

First order system with an integrator:

$$P_{outer} \approx \frac{3.3}{s(s+3.3)}$$



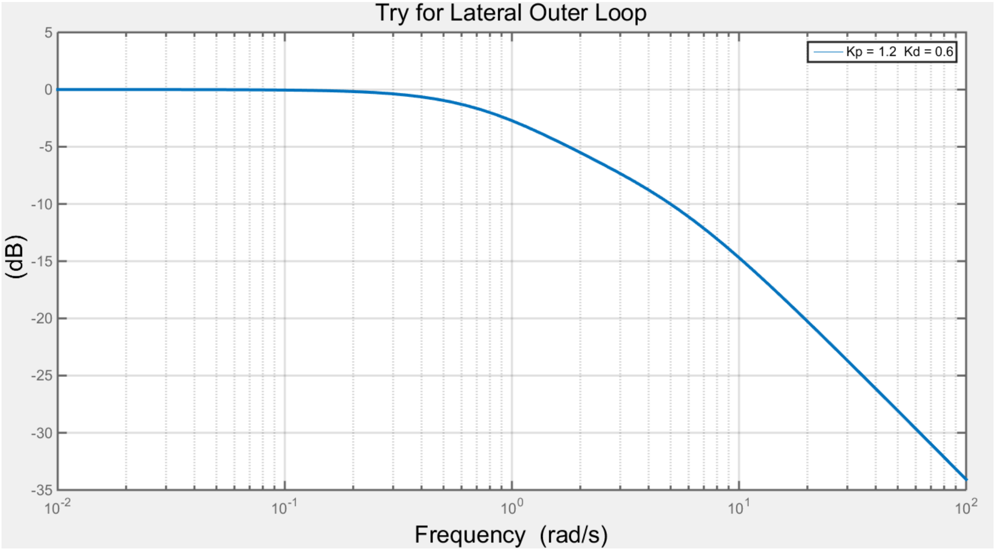
Using root locus method to design the PD controller:
(Put a zero at s = -2)

Kp = 1.2   Kd = 0.6   (g = 1.2 and z = 2)

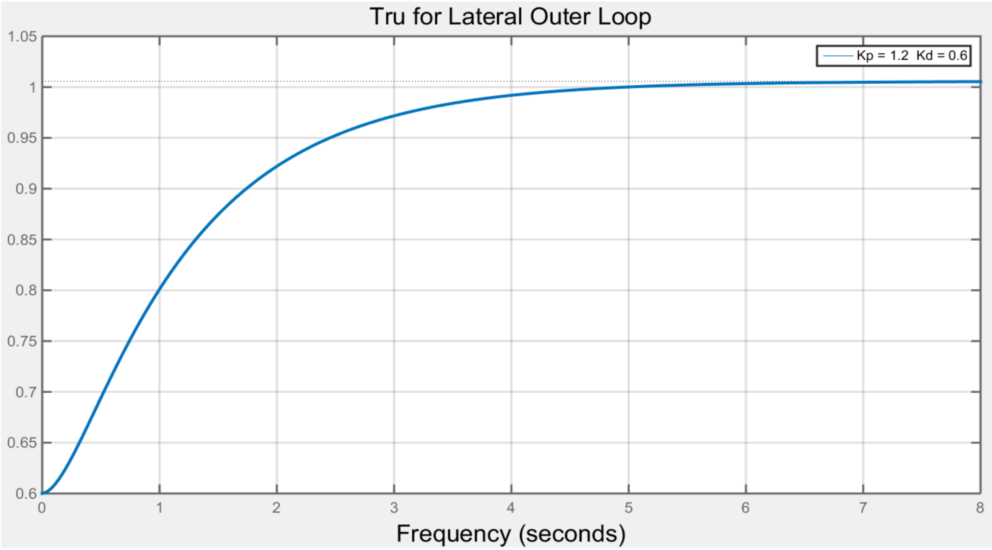
Lateral Outer Loop PD Controller Performance

$$T_{ry} = \frac{1.98(s+2)}{(s+0.9)(s+4.375)}$$

$$T_{ru} = \frac{0.6(s+3.3)(s+2)}{(s+0.9)(s+4.375)}$$

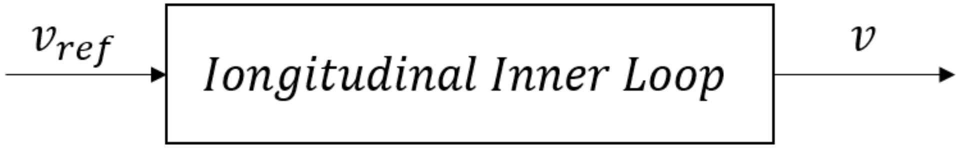


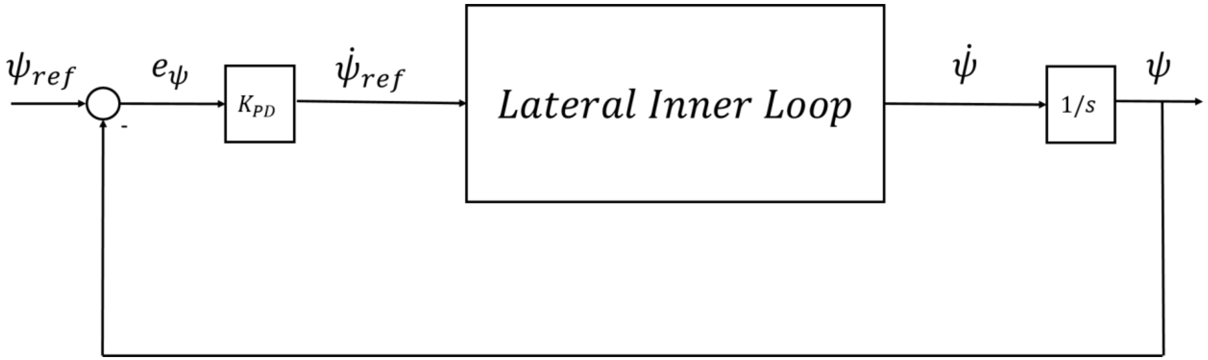
Bode Magnitude Plot for PD Outer Loop T_{ry}

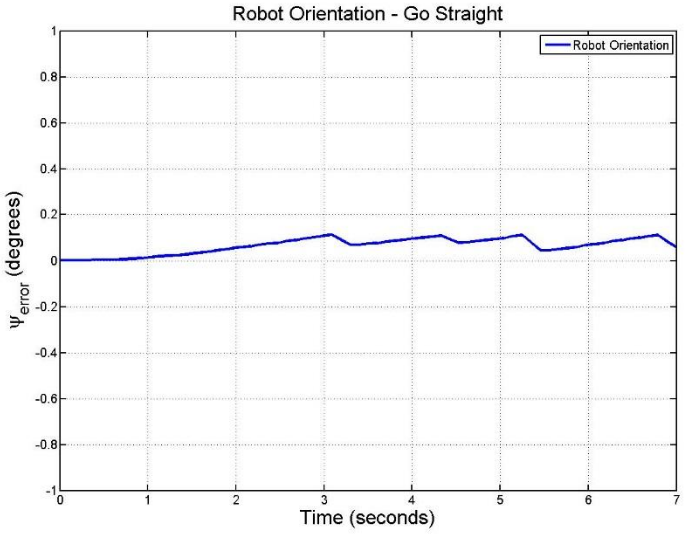


Step Response for Outer Loop T_{ru}

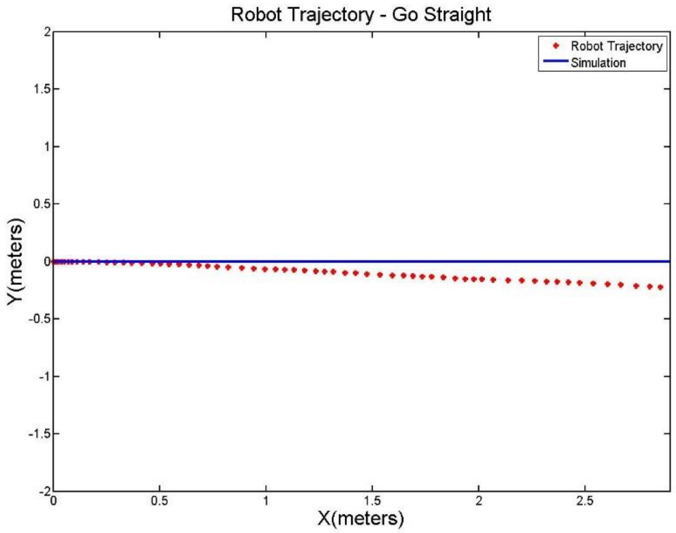
Going Along a Straight Line (v, θ Control)
(Dhaouadi, et al, 2013)







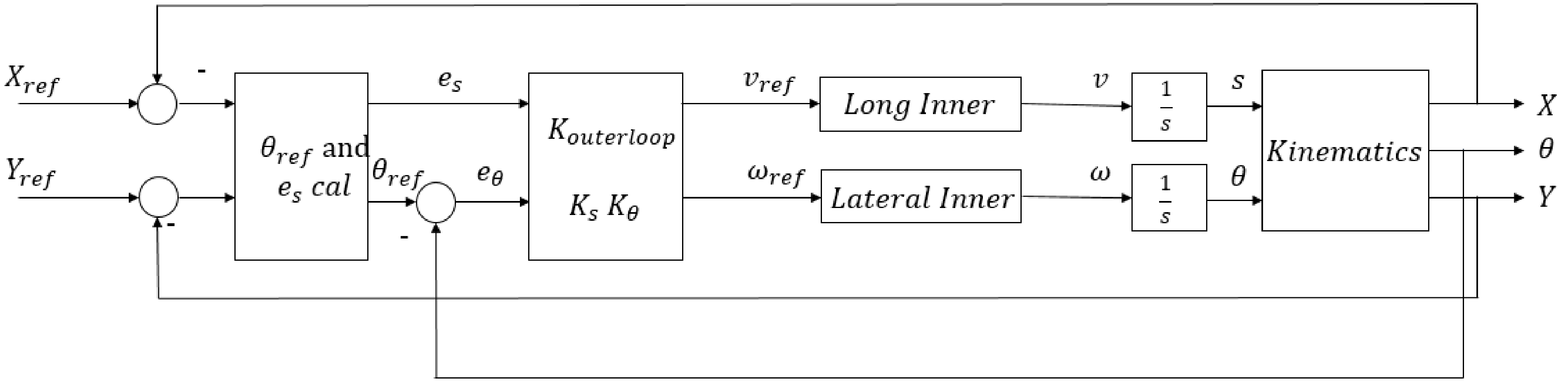
Orientation Angle Error (IMU)

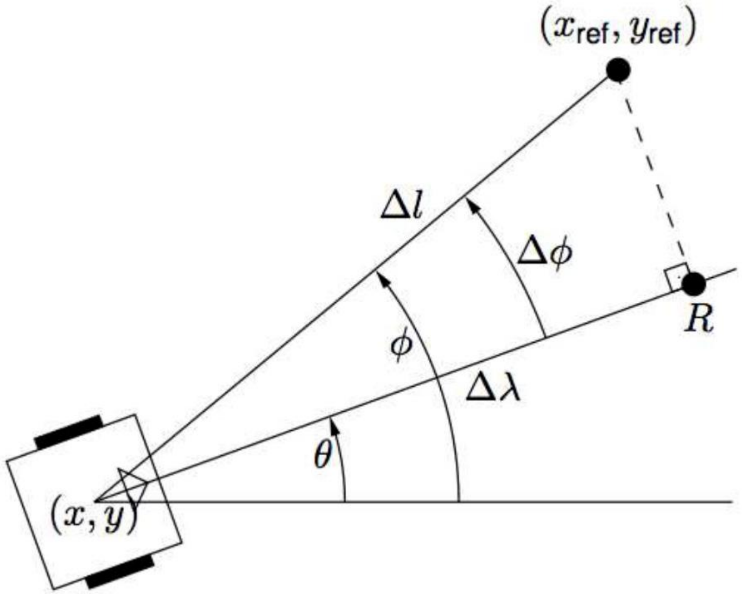


Trajectory (IMU and Encoder)

- Due to Dead Reckoning Error

Planar (x, y) Cartesian Stabilization – Algorithm
(Vieira, et.al. 2004)





Pointing angle:

$$\phi = \tan^{-1}\left(\frac{y_{ref} - y}{x_{ref} - x}\right)$$

$$e_\theta = \phi - \theta$$

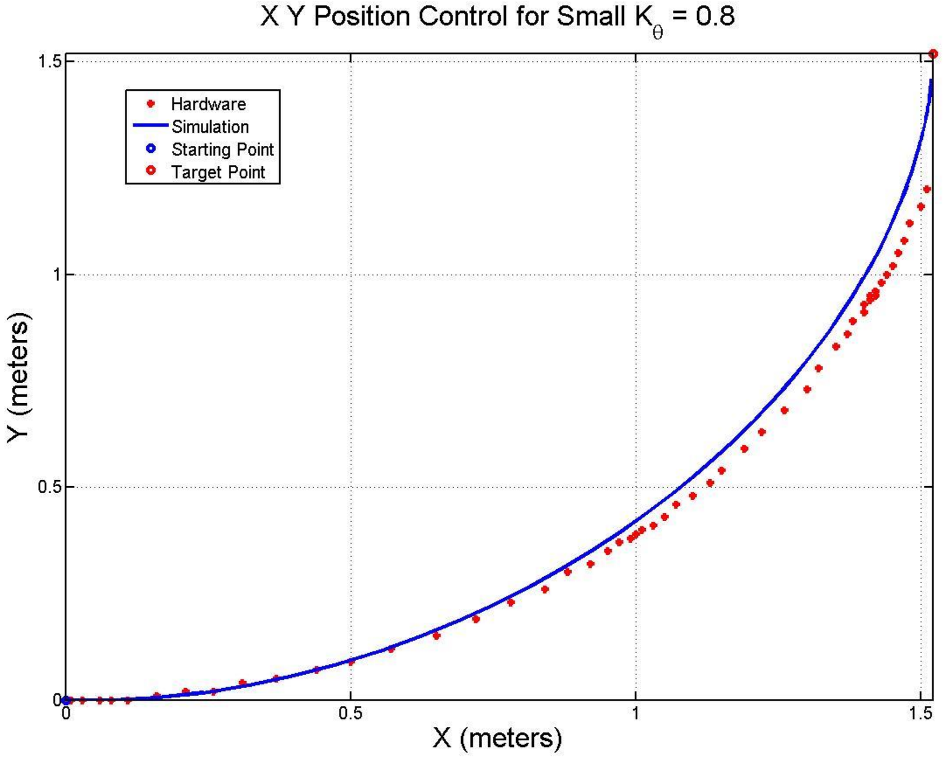
$$e_s = \Delta\lambda = \Delta l \cos \Delta\phi$$

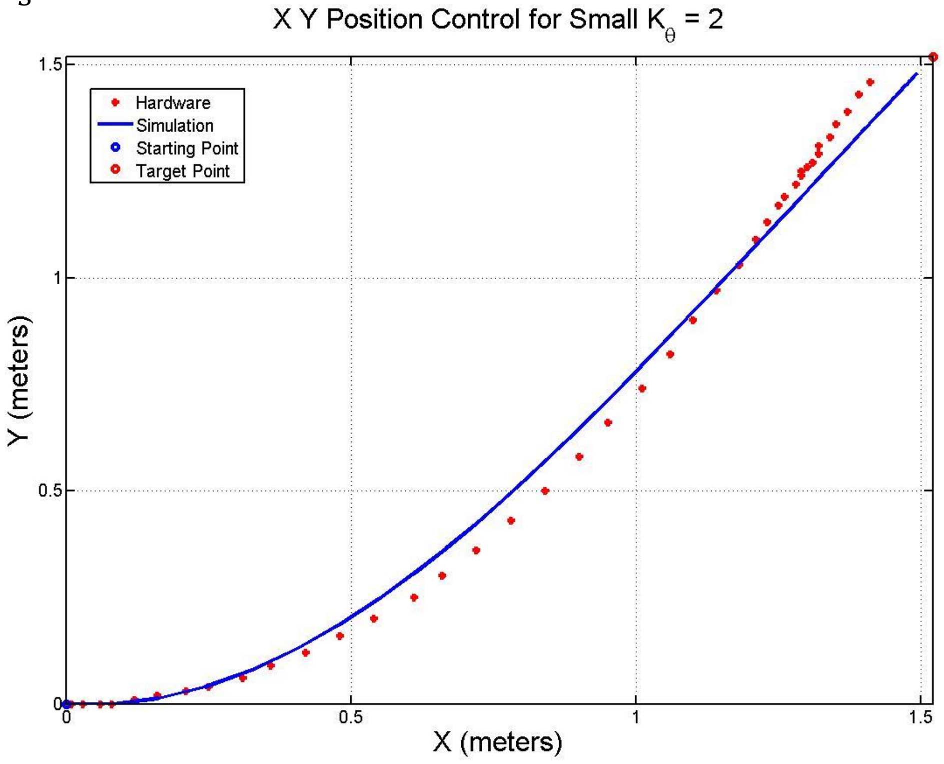
Outer Loop P controller, then send v_{ref} and ω_{ref} to inner loops:

$$v = k_s e_s \qquad \omega = k_\theta e_\theta$$

Planar (x, y) Cartesian Stabilization - Implementation

- Fixed K_s



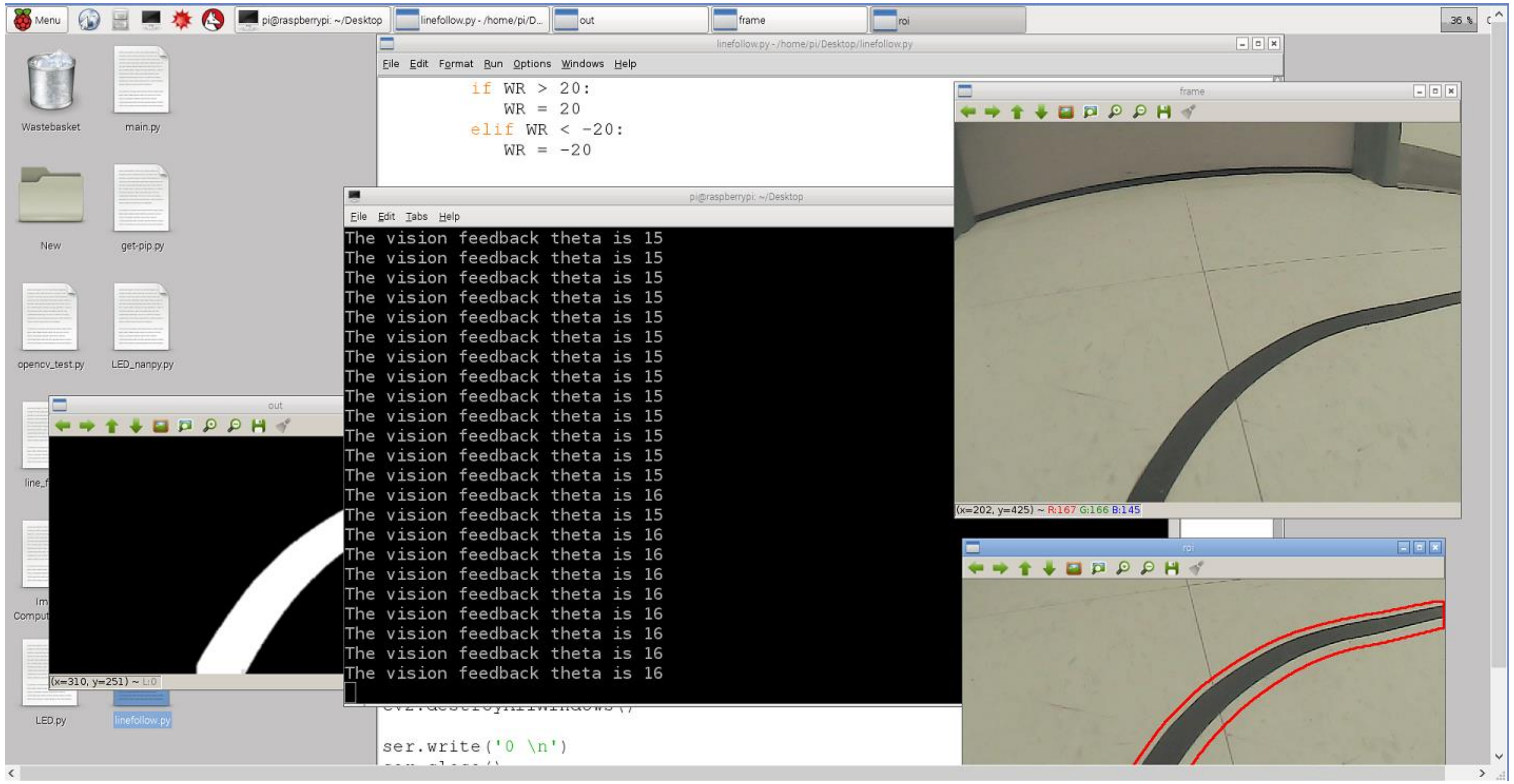


- Small K_θ ($K_\theta = 0.8$)
- Less directionally aggressive

- Large K_θ (K_θ = 2)
- Move more directly towards the target

Image Processing to Get Outer Loop ψ_{error}

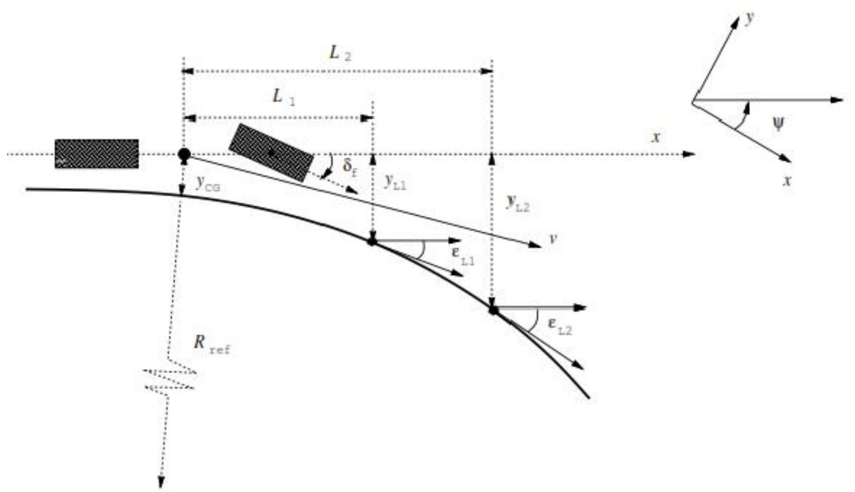
(Bradski G, Kaehler A, 2008)



- Vision subsystem offers ψ_{error} directly and send it to lower level controllers
- Outer loop frequency is limited by image processing process, which is 7.5Hz

Vision Subsystem Based Complete Model

$$\begin{bmatrix} \dot{v}_y \\ \ddot{\psi} \\ \dot{y}_L \\ \dot{\varepsilon}_L \end{bmatrix} = \begin{bmatrix} -\frac{c_f+c_r}{mv_x} & -v_x + \frac{c_r l_r - c_f l_f}{mv_x} & 0 & 0 \\ \frac{-l_f c_f + l_r c_r}{I_\psi v_x} & -\frac{l_f^2 c_f + l_r^2 c_r}{I_\psi v_x} & 0 & 0 \\ -1 & -L & 0 & v_x \\ 0 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \\ y_L \\ \varepsilon_L \end{bmatrix} + \begin{bmatrix} \frac{c_f}{m} \\ \frac{l_f c_f}{I_\psi} \\ 0 \\ 0 \end{bmatrix} \delta_f + \begin{bmatrix} 0 \\ 0 \\ 0 \\ v_x \end{bmatrix} K_L$$



$$y = \begin{bmatrix} -\frac{c_f+c_r}{mv_x} & \frac{c_r l_r - c_f l_f}{mv_x} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_y \\ \dot{\psi} \\ y_L \\ \varepsilon_L \end{bmatrix} + \begin{bmatrix} \frac{c_f}{m} \\ 0 \\ 0 \\ 0 \end{bmatrix} \delta_f$$

- K_L is Disturbance

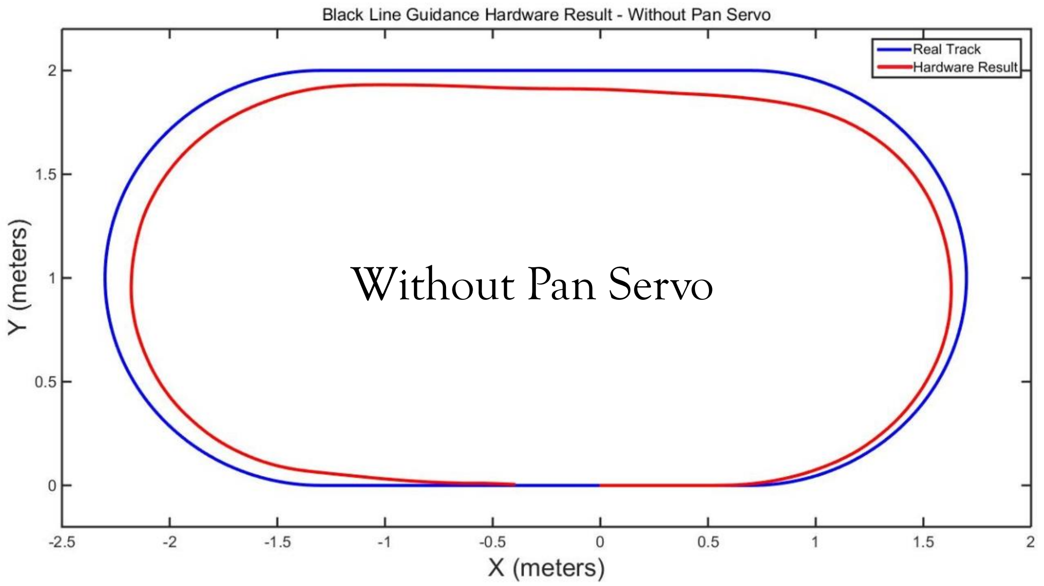
$$V_{1fs}(s) = \frac{y_L}{\delta_f} = \frac{0.06183s^2 + 0.04275s + 0.01781}{s^4 + 1.534s^3 + 1.228s^2}$$

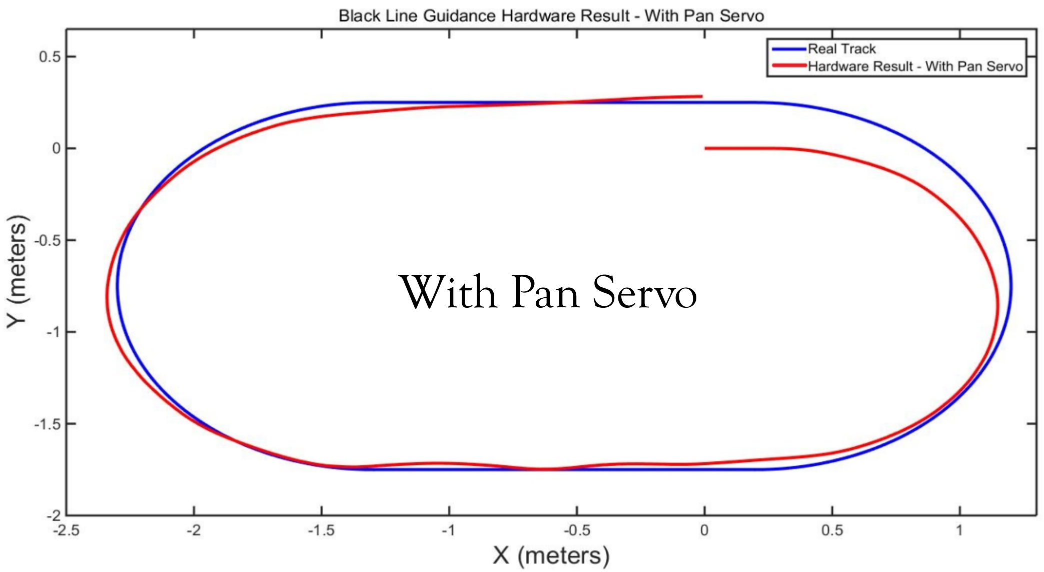
- y_L offset from the centerline at the look – ahead distance

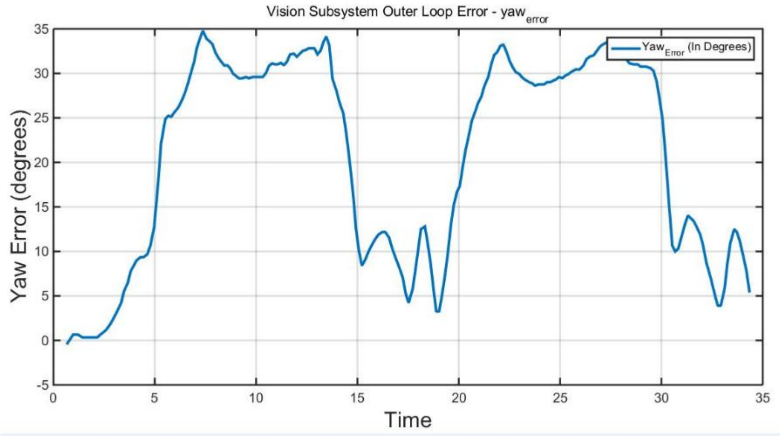
$$V_{2fs}(s) = \frac{\varepsilon_L}{\delta_f} = \frac{0.368s + 0.1781}{s^3 + 1.534^2 + 1.228s}$$

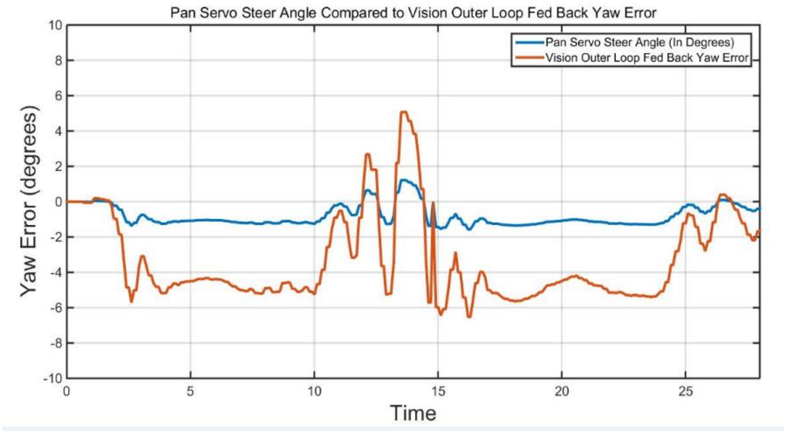
- ε_L angle between target to road and orientation of vehicle wrt the road
- L Look ahead distance at which the measurements are taken

Rear Wheel Drive Robot Finish Oval Track in Minimum Time With/Without Pan Servo

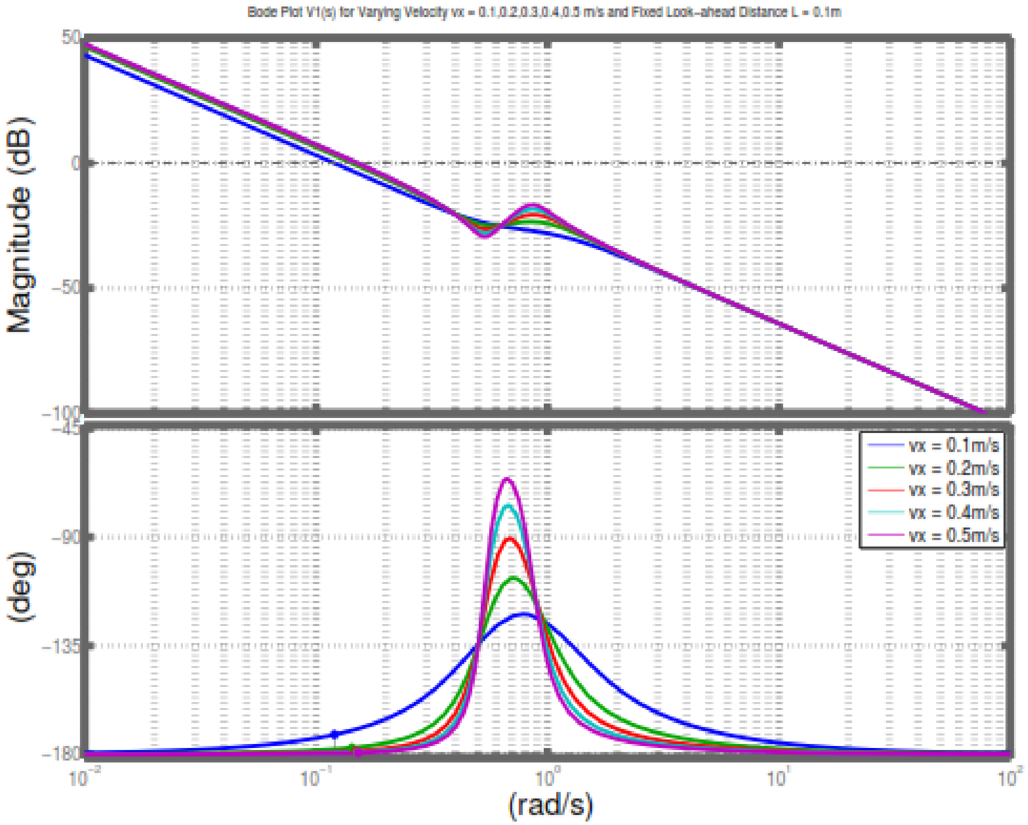


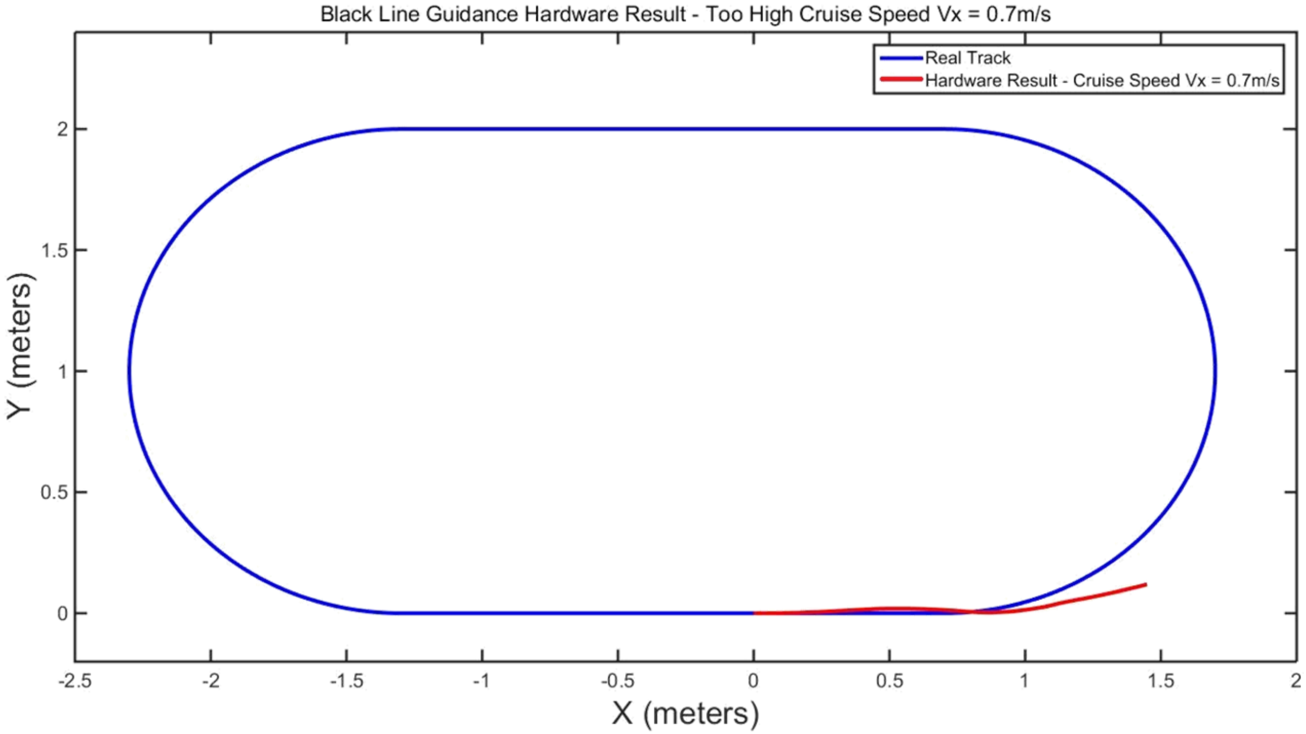






Track Following Performance with Different Cruise Speed V_x

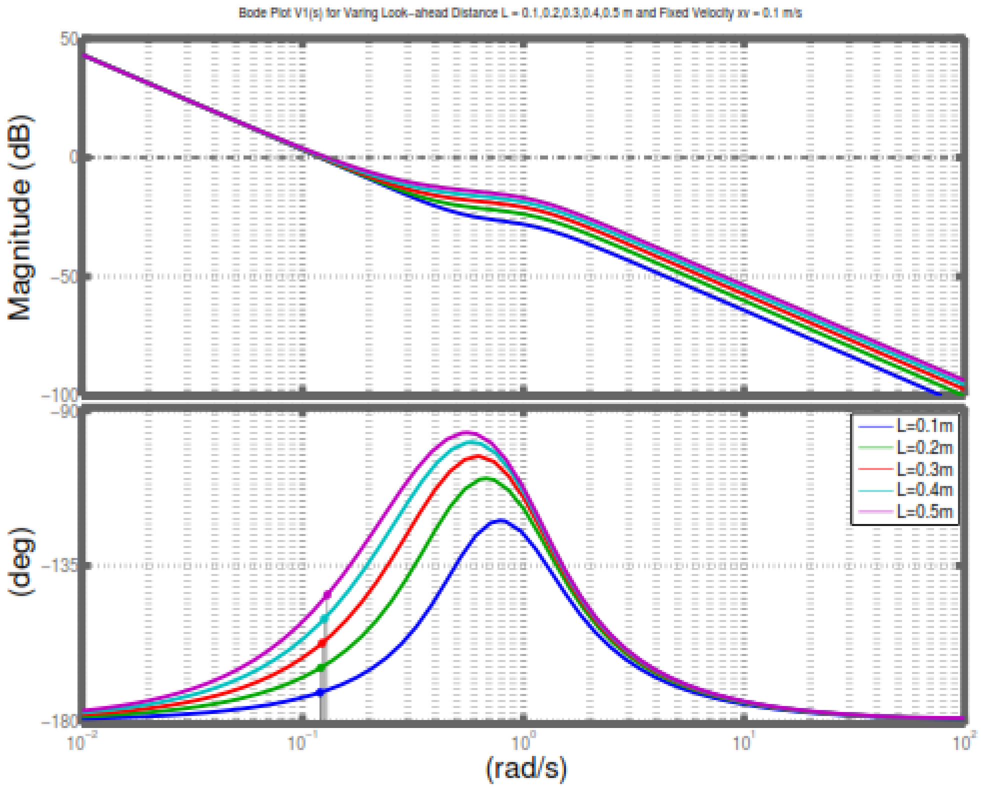


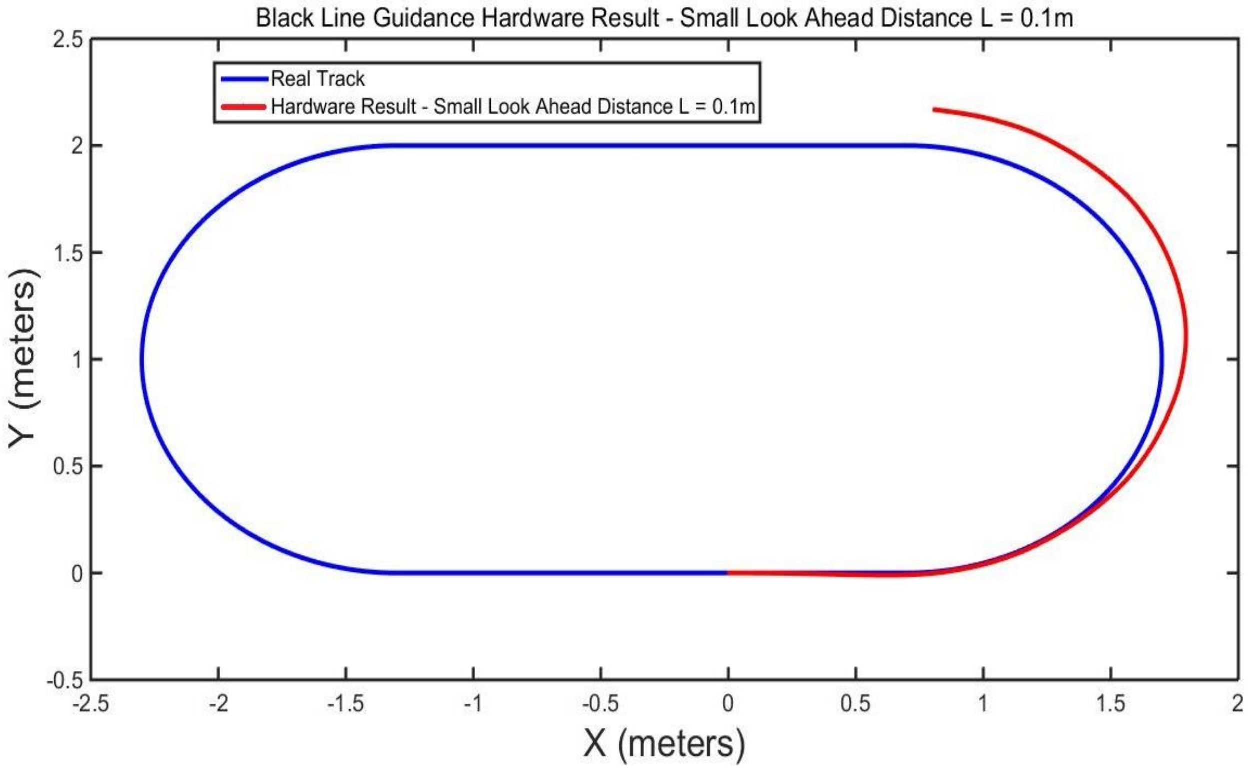


- When implementing a P controller: K = 1

- Phase Margin decreases as the robot cruise speed is increasing
- Hardware Result: Robot goes off the track with too high cruise speed ($v_x = 0.7m/s$)

Track Following Performance with Different Camera Look-Ahead Distance *L*





- ➢ When implementing a P controller: K = 1

- ➢ Phase Margin (PM) increases as the L is increasing.

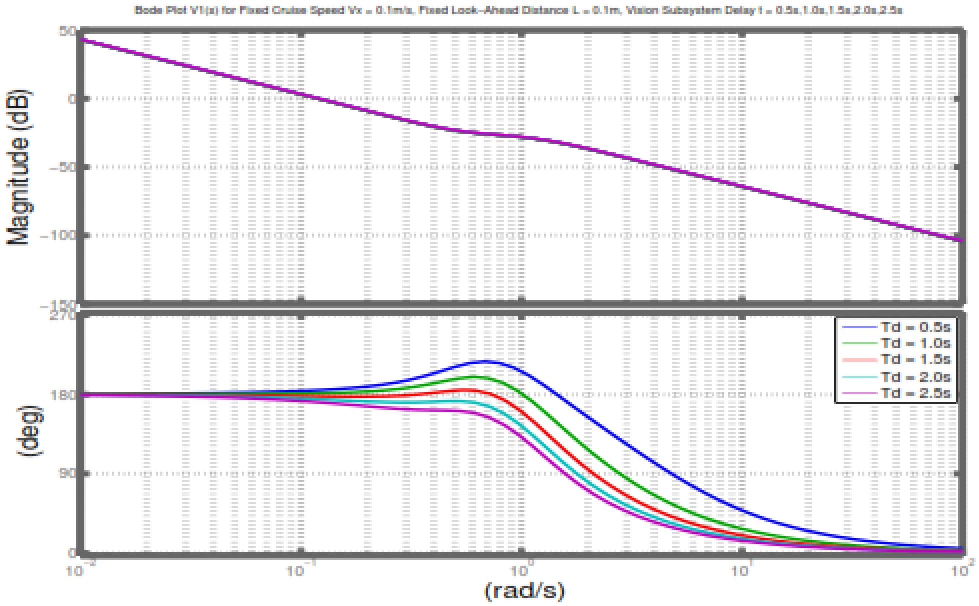
- ➢ Hardware Result: Robot goes off the track with too small camera look-ahead distance $L = 0.1m$

Track Following Performance with Different Vision subsystem delay T_d

According to Padé approximation:

$$D(s) = \frac{-0.5s + 2}{0.5s + 2}$$

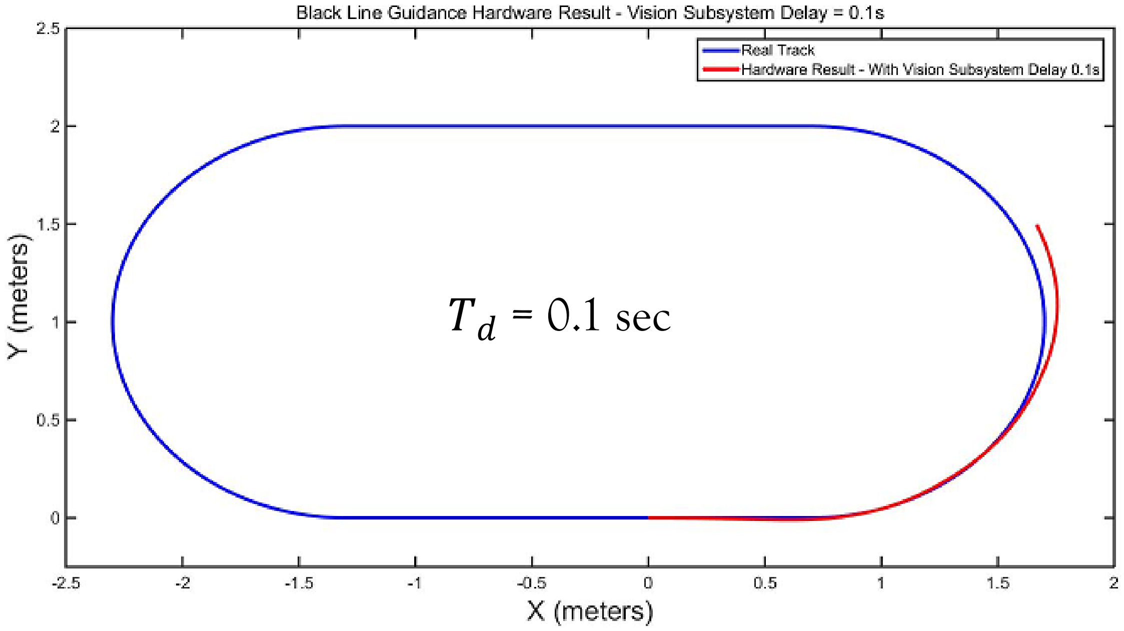
$$V_1(s)D(s) = \frac{y_L}{\delta_f} = \frac{-0.06183s^3 + 0.2046s^2 + 0.1532s + 0.07124}{s^5 + 5.534s^4 + 7.362s^3 + 4.912s^2}$$



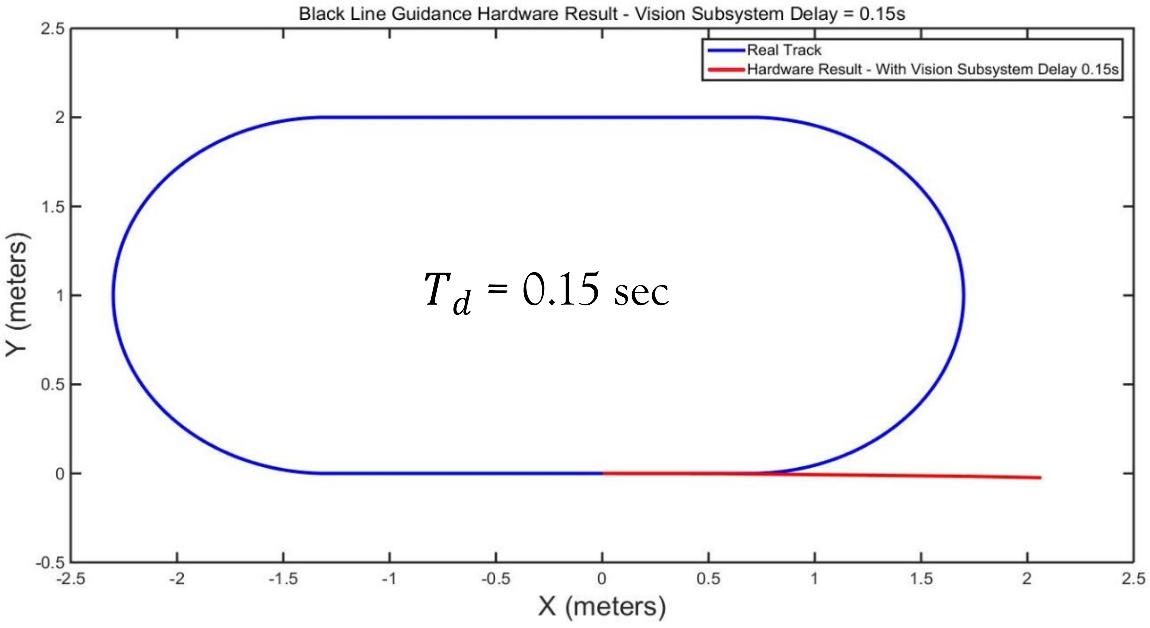
- With small vision subsystem delay T_d phase margin is very small
- With large vision subsystem delay T_d, L has a negative phase margin
- When implementing a P controller

K = 1

Track Following Performance with Different Vision subsystem delay T_d (Trajectory)



T_d = 0.1 sec



T_d = 0.15 sec

When we increases the delay from 0.1s to 0.15s

- Without vision subsystem delay (T_d = 0s), outer loop frequency is 7.52 Hz
- Without vision subsystem delay (T_d = 0.1s), outer loop frequency is 4.28Hz
- Without vision subsystem delay (T_d = 0.15s), outer loop frequency is 3.35Hz

LIDAR Hardware Description

- ➢ LIDAR I'm using:

XV 11 Hacked LIDAR

- ➢ A better LIDAR:

Hokuyo URG-04LX-UG01

- Price: $80
- Scan range: 0.2 to 6.0 meters
- Scan Frequency: 5.5 Hz
- Accuracy: ±80 mm
- Angular Resolution: 0.52°

- Price: $1115
- Scan range: 0.1 to 5.6 meters
- Scan Frequency: 10.0 Hz
- Accuracy: ±30mm
- Angular Resolution: 0.35°

SLAM Problem Definition

Given

- The robot's controls

$$u_{1:T} = \{u_1, u_2, u_3 \ldots, u_T\}$$

- Observations

$$z_{1:T} = \{z_1, z_2, z_3 \ldots, z_T\}$$

Wanted

- Map of the environment

$$m$$

- Path of the robot

$$x_{0:T} = \{x_0, x_1, x_2 \ldots, x_T\}$$

Estimate the robot's path and the map

$$p(x_{0:T}, m \mid z_{1:T}, u_{1:T})$$

distribution path map given observations controls

f -motion equation

u -control inputs

w -Input noise

g -observation equation

y -observation data

n -observation noise

Motion Model

- The motion model describes the relative motion of the robot

$$p(x_t \mid x_{t-1}, u_t)$$

distribution new pose given old pose control

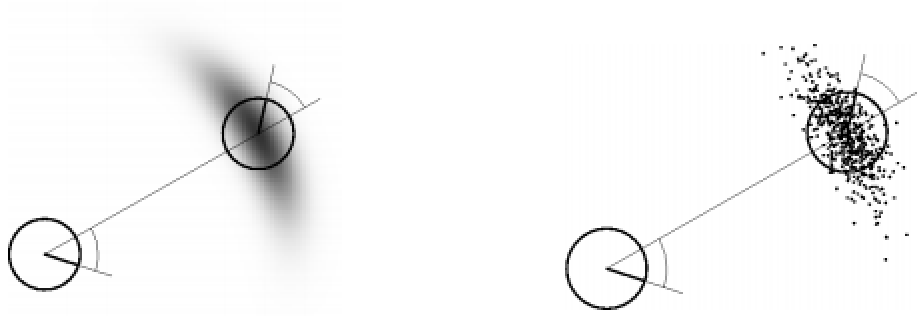
Pose: $x_k = [x, y, \psi]_k$

Motion equation f: $x_{k+1} = x_k + \Delta x_k + w_k$

Input Noise: w_k (Gaussian Noise)

- Gaussian model

- Non-Gaussian model



Non Gaussian Noise: Salt and Pepper Distribution

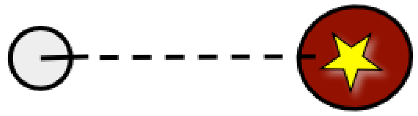
Observation Model

The observation or sensor model relates measurements with the robot's pose

$$p(z_t \mid x_t)$$

distribution observation given pose

Gaussian model



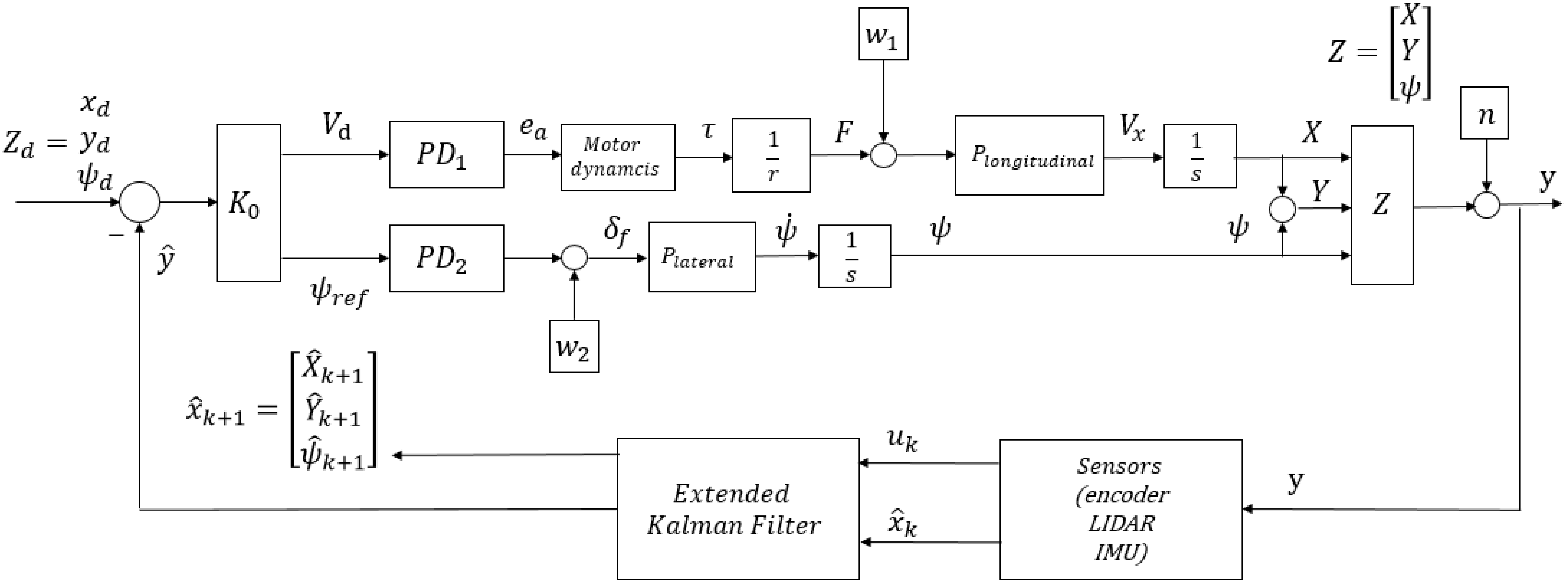
$L_k = [L_{k,x} , L_{k,y}]$ is a 2D landmark

Observation Equation g:

$$\begin{bmatrix} r \\ \theta \end{bmatrix}_k = \begin{bmatrix} \sqrt{\|x_k - L_k\|2} \\ tan^{-1} \frac{L_{k,y} - x_{k,y}}{L_{k,x} - x_{k,x}} \end{bmatrix} + n_k$$

- f and g are linearized around $\hat{x}_{k-1}$ *and* $\hat{x}_k$
- Then apply Kalman Filter

Block Diagram – Extended Kalman Filter (EKF)

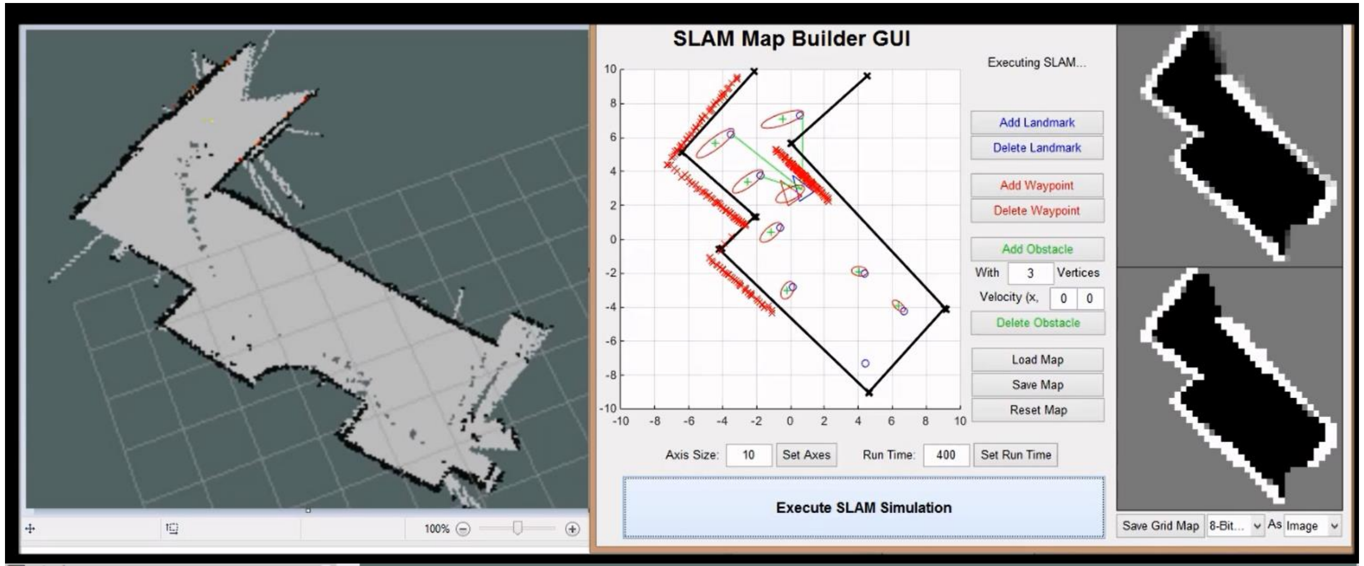


Pose Estimation for nonlinear system

Self Build Indoor Experiment Area (GWC 2nd Floor)

Robot has the ability to map a 26 m^2 environment in 38 seconds.

Simulation and Implementation Results for Mapping this Area

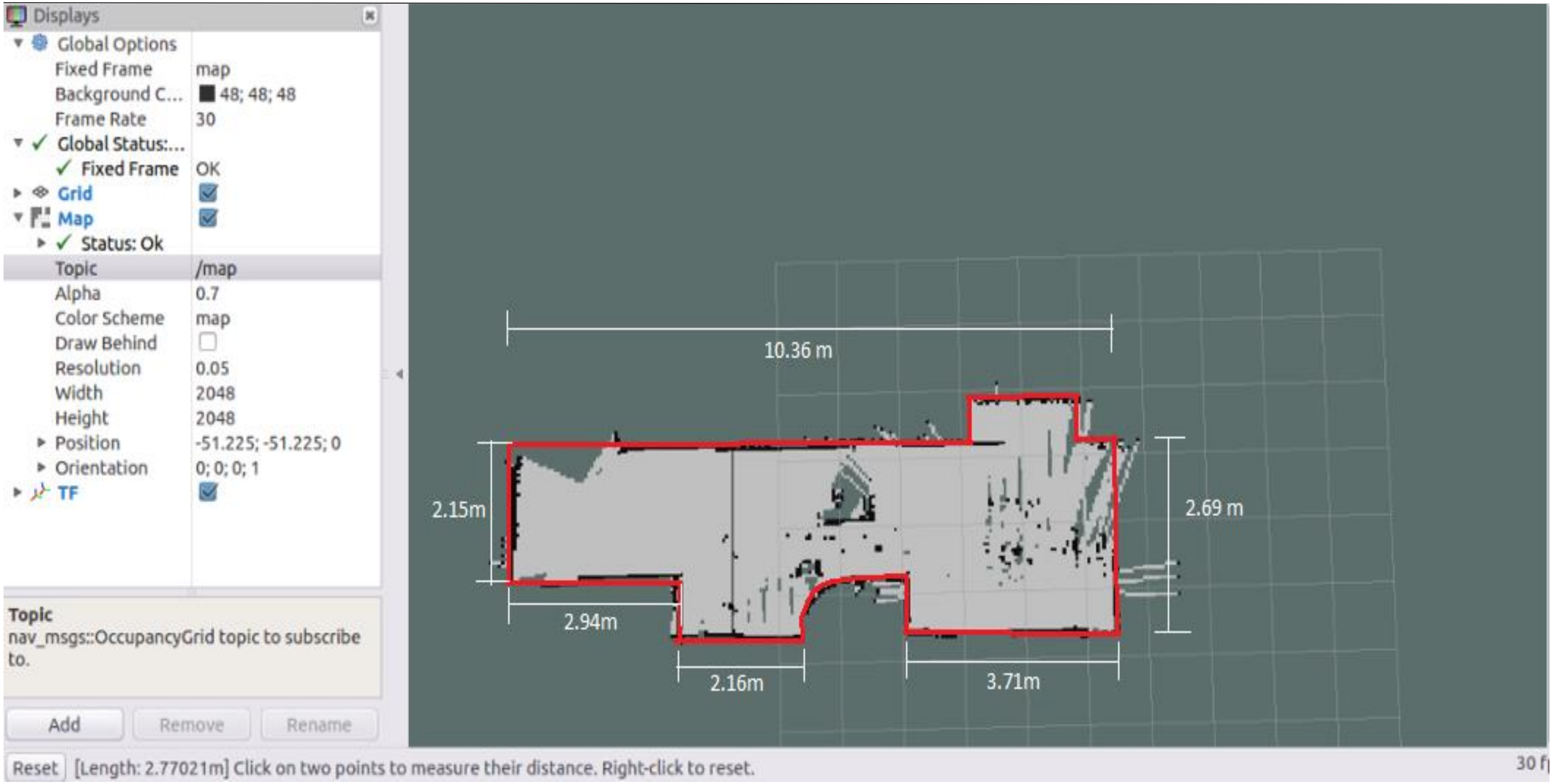


Simulation Result

Implementation Result

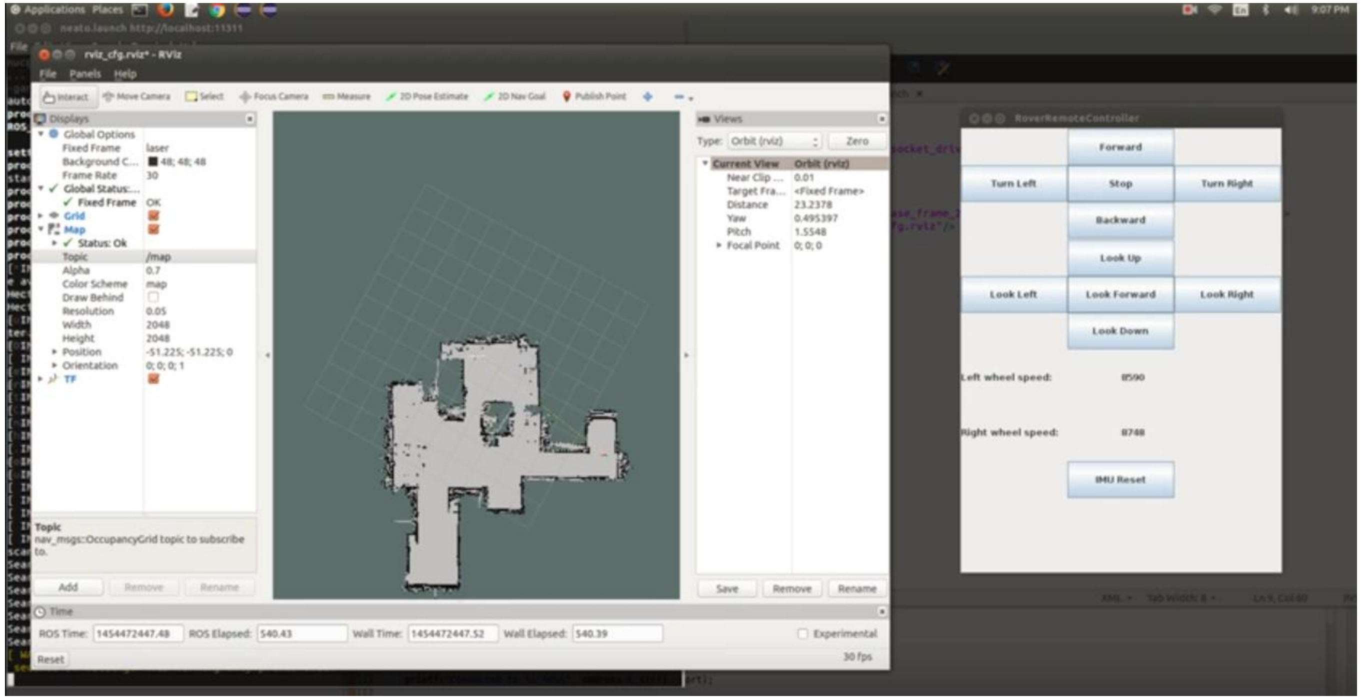
- horizontal accuracy : 5.40%
- vertical accuracy : 2.97%

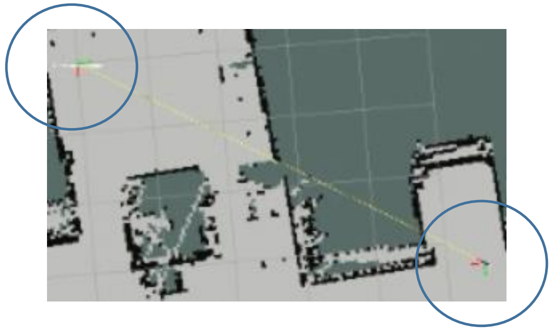
Comparison Between Real Floor Plan and Generated 2D Grid Map



Mapping Duo's house, robot was controlled manually by GUI pedals (on the right)

- Map the unknown environment
- Localize robot
- Real-time capable
- Saving GeoTiff maps





Real time position of the robot

Please see demo on Youtube:
https://www.youtube.com/watch?v=750z3U4tSAA

When Will Something Go Wrong (Turning too Fast)

➢ Map of CenterPoint Building Floor 4 Computer Science Lab and Hallway

➢ <u>Lack of scan frequency</u>

Future Works and Studies

- Localization Development of a lab-based localization system using a variety of technologies (e.g. USB cameras, depth sensors, LIDAR, ultrasonic, etc.).
- On-board Sensing Addition of multiple on board sensors; e.g. additional ultrasonic, depth sensors (Kinect), 3D LIDAR, GPS, cameras, etc.
- Advanced Image Processing Use of advanced image processing and optimization algorithms; e.g. Implementations of OpenCV and OpenGL and vision based mapping and localization.
- 3D unknown environment reconstruction. In this thesis, the 2D indoor unknown environment mapping was well discussed.
- Modelling and Control More accurate dynamic models and controls laws.
- Control-Centric Vehicle Design Understanding when simple control laws are possible and when complex control laws are essential.

Thank you

IRA A. FULTON SCHOOLS OF engineering ARIZONA STATE UNIVERSITY

Thank you

Any Questions?