Final Year Project

School of Mechanical and Aerospace Engineering

Vision and Odometry Localization in an Indoor Environment

TAN YOU LIANG U1420457B

NANYANG TECHNOLOGICAL UNIVERSITY

MAE
Mechanical and Aerospace Engineering Club

CONTENT

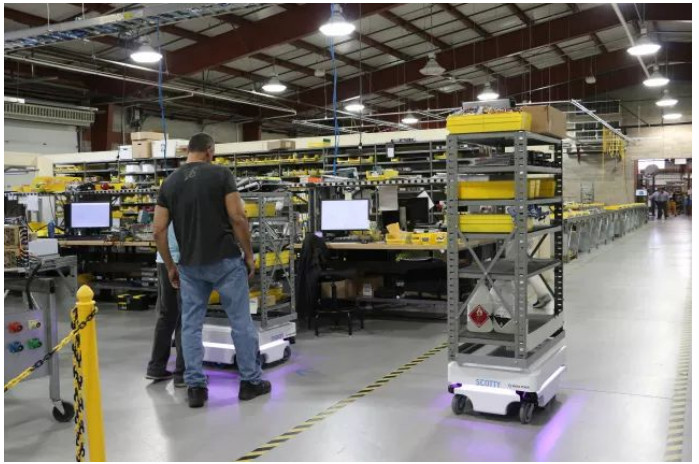
- Introduction
- Literature Review
- Phase 1 Implementation: *Vision Pose Estimation*
- Phase 2 Implementation: *IMU and System Integration*
- Phase 3 Implementation: *Encoder and Fusion Enhancement*
- Discussions
- Conclusion

INTRODUCTION

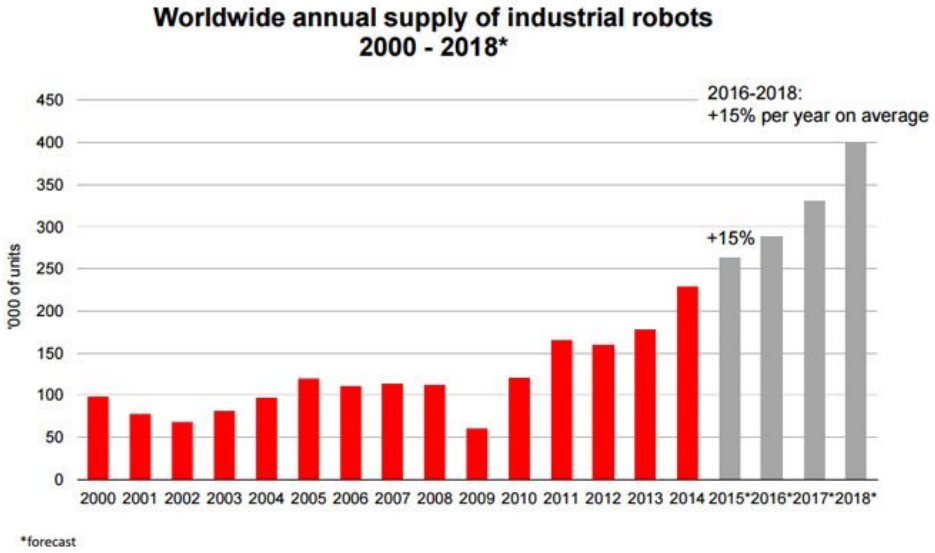
Automation in Manufacturing and Logistics

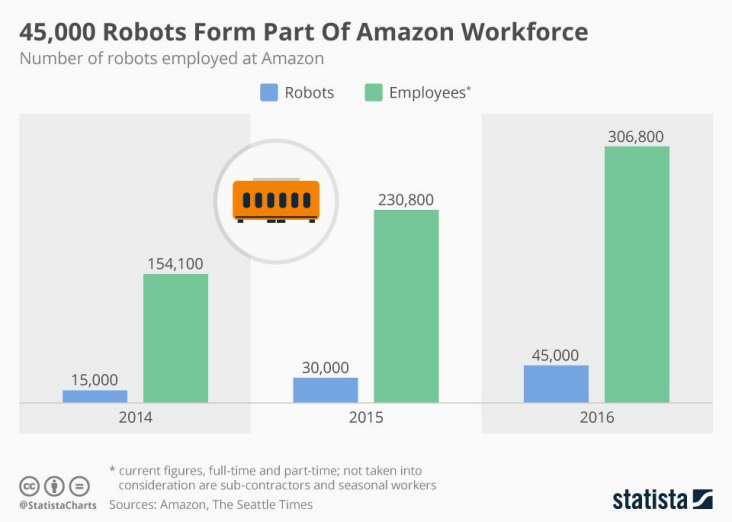
Gross Market Sales in Automation:

Increase x15 times in 10 years time



Statistics


Worldwide annual supply of industrial robots
2000 - 2018*
'000 of units
450
400
350
300
250
200
150
100
50
0
2016-2018:
+15% per year on average
+15%
2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013 2014 2015* 2016* 2017* 2018*
*forecast

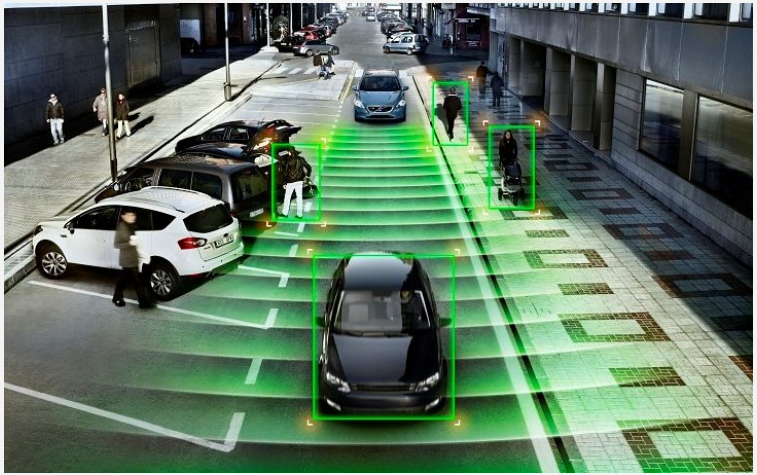

45,000 Robots Form Part Of Amazon Workforce
Number of robots employed at Amazon
Robots
Employees*
15,000
154,100
30,000
230,800
45,000
306,800
2014
2015
2016
* current figures, full-time and part-time; not taken into consideration are sub-contractors and seasonal workers
@StatistaCharts
Sources: Amazon, The Seattle Times
statista

Automation system

- Perception System
  - Environmental Perception
  - **Localization**
- Planning System
  - Mission Planning
  - Behavioral Planning
  - Motion Planning
- Control System
  - Path Tracking
  - Trajectory Tracking

Localization

Capability of localize oneself in an environment



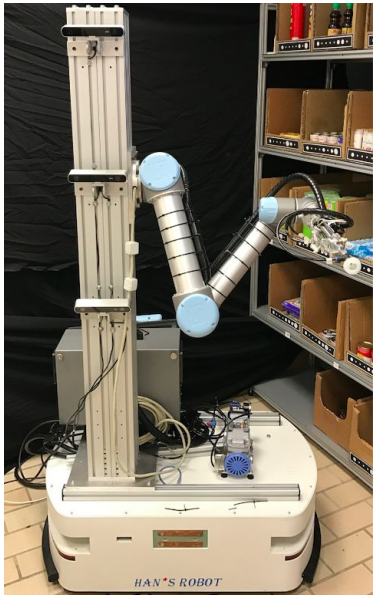
Problem Statement

Localization of mobile picking robot in an enclosed warehouse Environment

Methodology?

Cost effectiveness?

Accuracy?
(error < 15cm)



Precision?

Blindspot?

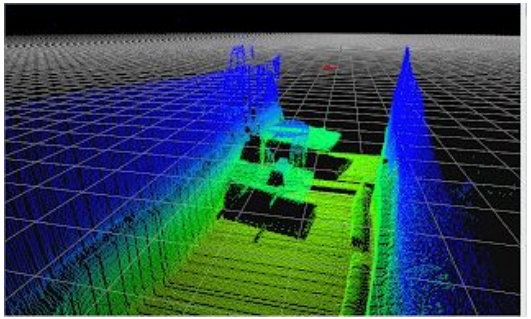
Latency?

Scope and Objective 

Design a ground-up localization system for a mobile robot working in an indoor warehouse environment. An accuracy of +-15cm need to be achieved.

Current Indoor localization Methods

Lidar Sensing



Floor Markers

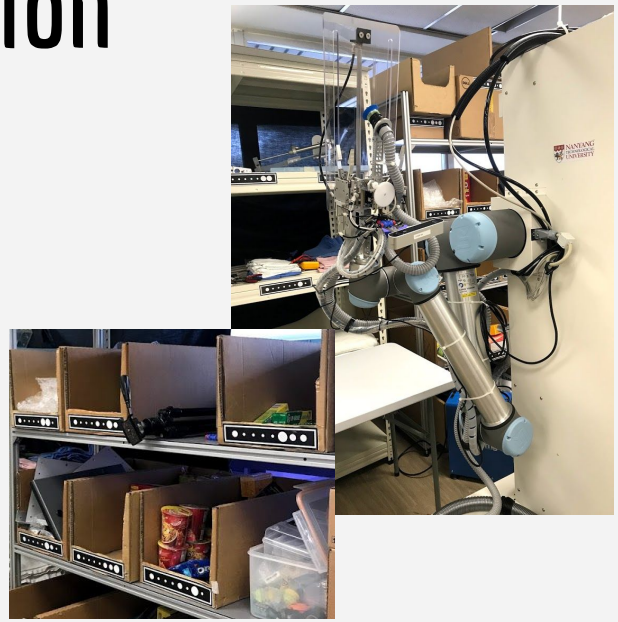
EM wave Emitter

Proposed Solution

Vision and Odometry Localization

Design an accurate localization system for indoor warehouse navigation by using camera and odometry localization, while sensor fusion will be incorporated.

Hardware: Camera, IMU, Encoder



General Advantages

Cost Effectiveness

- Leverage on existing info markers as localization target
- Using existing cameras which compliment with additional IMU

Easy Implementation

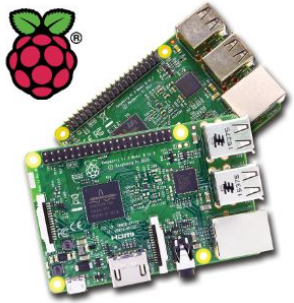
Simple implementation since no additional installation needed for the warehouse

Performance & Redundancy

- Effective in extracting useful info via marker detection in fast changing environment
- Less computing power
- Sensor fusion helps in improving accuracy and redundancy
- On-board localization

Tools

Hardware:
Raspberry pi, cameras, 9dof IMU, encoders



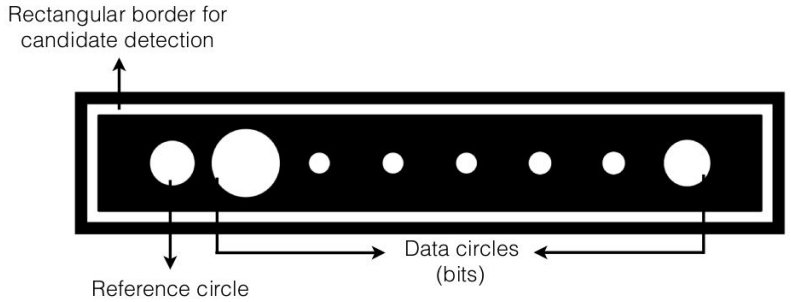
Software:
Python and C++: OpenCV, ROS, RVIZ, RTIMU

Research Papers:
Homography transformation, openCV optimization,
Kalman filter for sensor fusion, Odometry localization

LITERATURE REVIEW

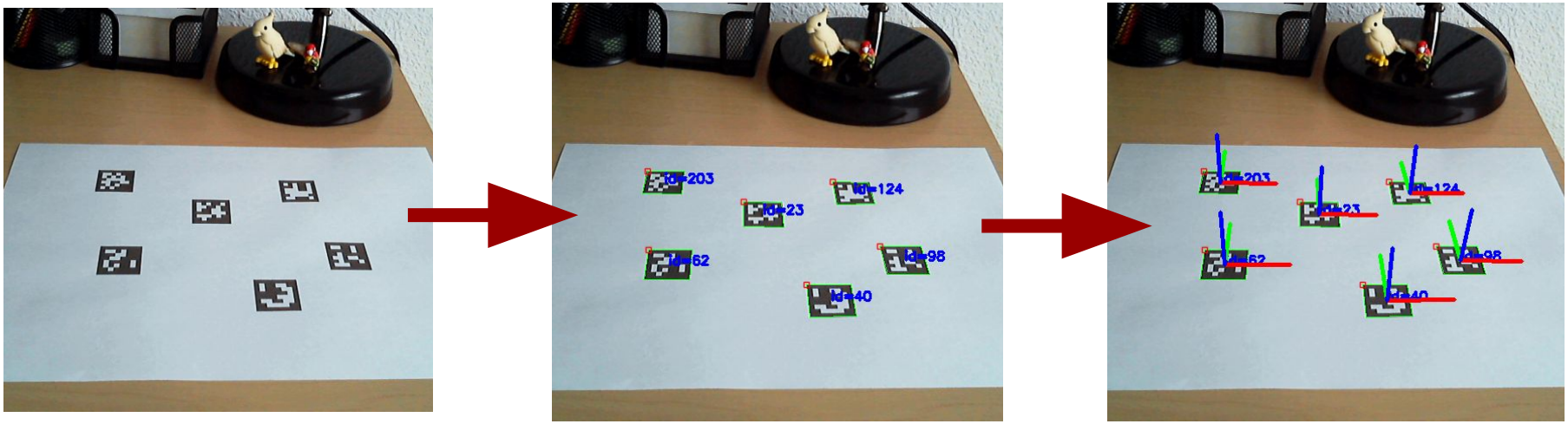
VITag

- Custom Designed Visual Marker
- Function as ID for all products on shelves
- Distinct ID can be used for localization
- Feature extraction & frame filtering
- VITag Extraction & Pose estimation



ArUCO Markers

Commonly used in Computer Vision Application (e.g position tracking)



Marker Extraction & Identification Process

Pose Estimation

Homography Transformation

Obtain Rotational and Translation Matrix (6dof) of planar Marker from 2 coordination system

$$\mathrm{b} = \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \sim \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Calibration Matrix, K

Intrinsic camera parameters

Extrinsic camera parameters

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \sim \underbrace{\begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}}_{\text{Homography } \mathbf{H}} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$x' = K \begin{bmatrix} R & | & 0 \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix} = KRX \quad (5)$$

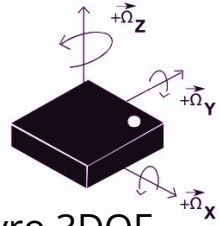
$$x = K \begin{bmatrix} I & | & 0 \end{bmatrix} \begin{bmatrix} X \\ 1 \end{bmatrix} = KX \quad (6)$$

By compare equations 5&6, $x' = KRK^{-1}x = H\,x$

Homography, $H = KRK^{-1}$

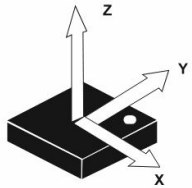
IMU (Inertial Measuring Unit)

IMU SENSOR



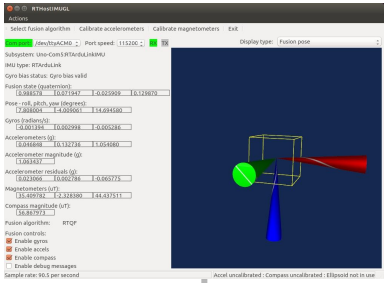
Gyro 3DOF

Magnetometer 3DOF



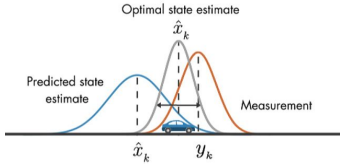
Accelerometer 3DOF

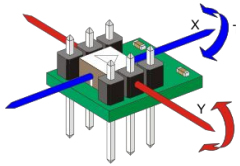
➔ Raw pitch yaw
➔ Acceleration on xyz axis



Odometry Displacement, Velocity and Heading

Kalman Filter (sensor fusion)





IMU Odometry

Encoder Odometry

Time Update
(prediction)

1 Project the state ahead

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$$

2 Project the error covariance ahead

$$P_k^- = AP_{k-1}A^T + Q$$

Measurement Update
(correction)

1 Compute the **Kalman Gain**

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$

2 Update the estimate via z_k

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$

3 Update the error covariance

$$P_k = (I - K_k H)P_k^-$$

Initial estimates at **k = 0**

The outputs at **k** will be the input for **k+1**

Vision localization

Output of sensor fusion:

{ X, Y, YAW }

2D and 3D Transformation

$[X, Y] = R_{(2 \times 2)} [x, y]$

$[X, Y, Z] = R_{(3 \times 3)} [x, y, z]$

$$R_z(\alpha) = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}. \qquad R_y(\beta) = \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix}. \qquad R_x(\gamma) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}.$$

$$R(\alpha,\beta,\gamma) = R_z(\alpha)\, R_y(\beta)\, R_x(\gamma) = \begin{pmatrix} \cos\alpha\cos\beta & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\alpha\cos\beta & \sin\alpha\sin\beta\sin\gamma + \cos\alpha\cos\gamma & \sin\alpha\sin\beta\cos\gamma - \cos\alpha\sin\gamma \\ -\sin\beta & \cos\beta\sin\gamma & \cos\beta\cos\gamma \end{pmatrix}.$$

PHASE 1 IMPLEMENTATION:

VISION POSE ESTIMATION

What Phase 1 is about ?

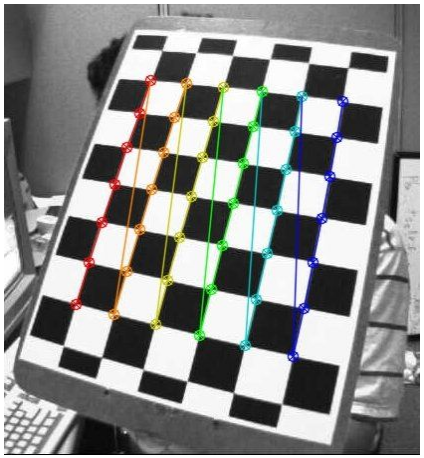
- Developed a Vision Based Localization Tool by using OpenCV Library
- Reference to ArUCO detection and Pose Estimation
- Obtain Camera Pose Estimation by VITag marker detection
- Primary FYP target

LOOK AND DETECT THE MARKER.

THEN, IDENTIFY WHERE IS THE CAMERA

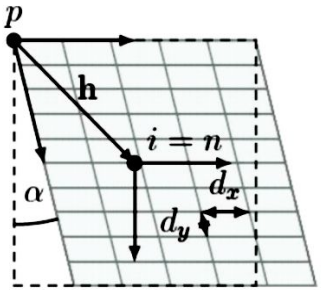
Camera Calibration

- OpenCV ChessBoard Camera Calibration
- Obtain Camera Intrinsic and Distortion Matrix



Chessboard

$$\begin{bmatrix} x_p \\ y_p \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s_\alpha & h_x \\ 0 & f_y & h_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix}$$

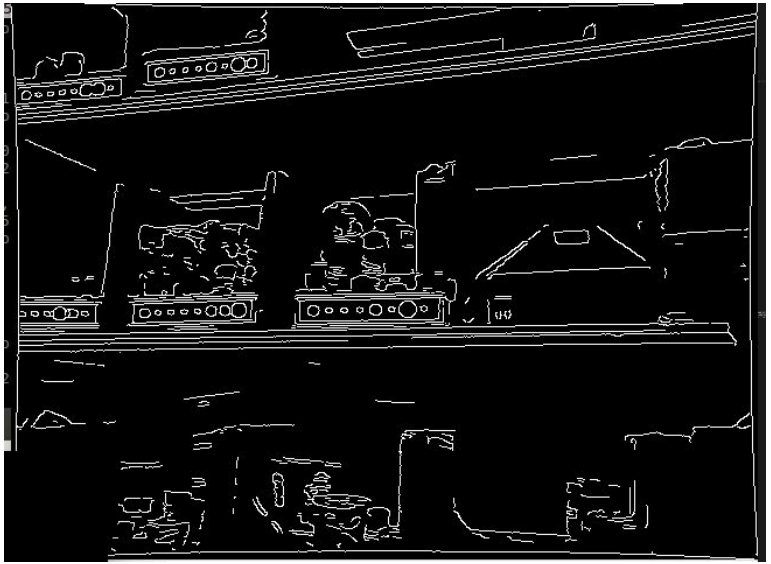


$$\begin{bmatrix} x_d \\ y_d \end{bmatrix} = \underbrace{(1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6)}_{\text{radial distortion}} \begin{bmatrix} x_n \\ y_n \end{bmatrix}$$

Initial Filtering

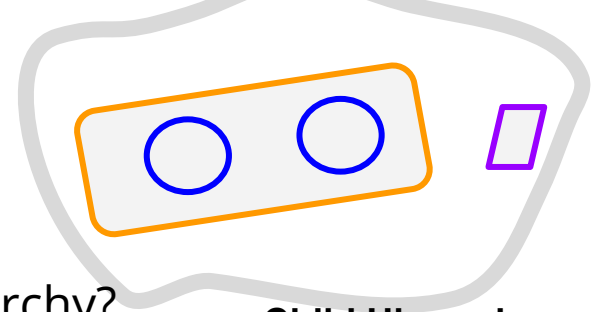
- Undistort frame, Erode and edge Detection

Raw Frame



Filtered Frame

Contour Filtering



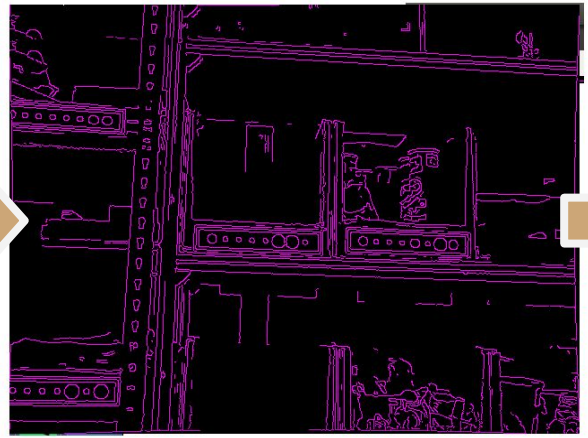
How to find VITag border contour according to hierarchy?

- Set threshold for contour size
- Find ‘youngest child’s parent’ (orange)
- eliminate outer filtered contours in contours

Child Hierarchy

Grey: orange
Orange: Blue
Blue: -1

Raw Frame



Frame will all contours

Frame with filtered contours

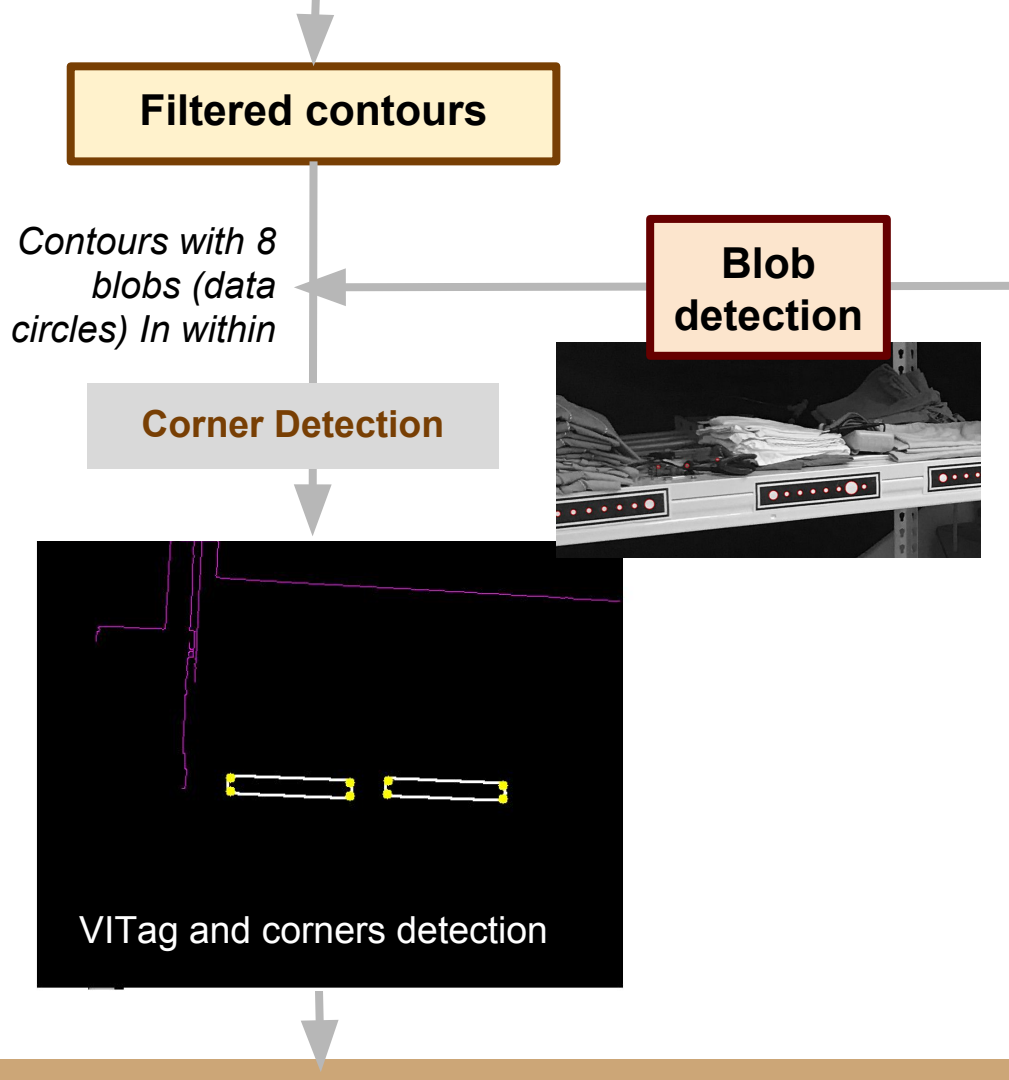
VITag Detection

Find VITag contour

- Blob detection
- Find all number of blobs in each contour
- Identify target VITag contour

Find 4 extreme corners of contour

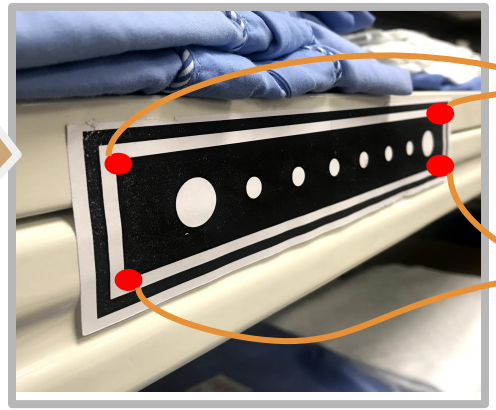
- Corner Detection
- Shi Tomasi Corner Detection



Pose Estimation

- Use VITag 4 corner coordinates to match Target coordinates
- Perform Homography transformation to Obtain Pose Estimation

Detected corner Coordinates



Homography Matrix, ***H***

Target corner coordinates

OpenCV Function: *SolvePnP()*

$$Homography,\ H = KRK^{-1}$$

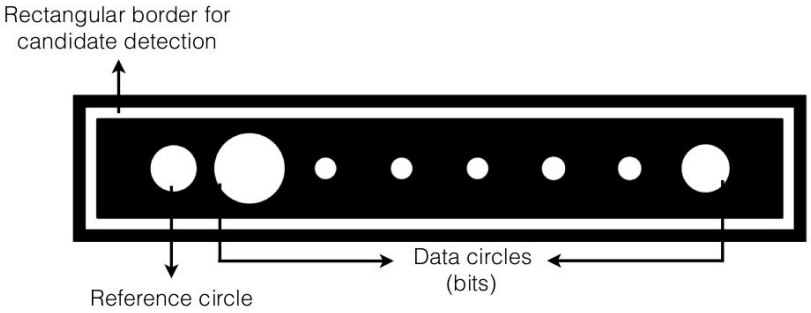
$$x' = K\begin{bmatrix} R & | & T \end{bmatrix}\begin{bmatrix} X \\ 1 \end{bmatrix}$$

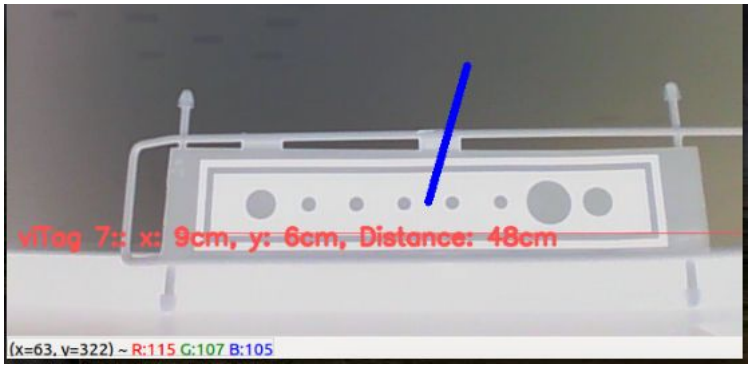
OUTPUT

Translation {x, y, z}
Rotation {R, P, Y}

ID Recognition

- Homography transformation, Obtained planar view of detected marker
- Data Circle: Set size threshold by reference to “reference data circle”
- Find ternary form of data circles according to size (0000021)
- Convert to decimal (7)





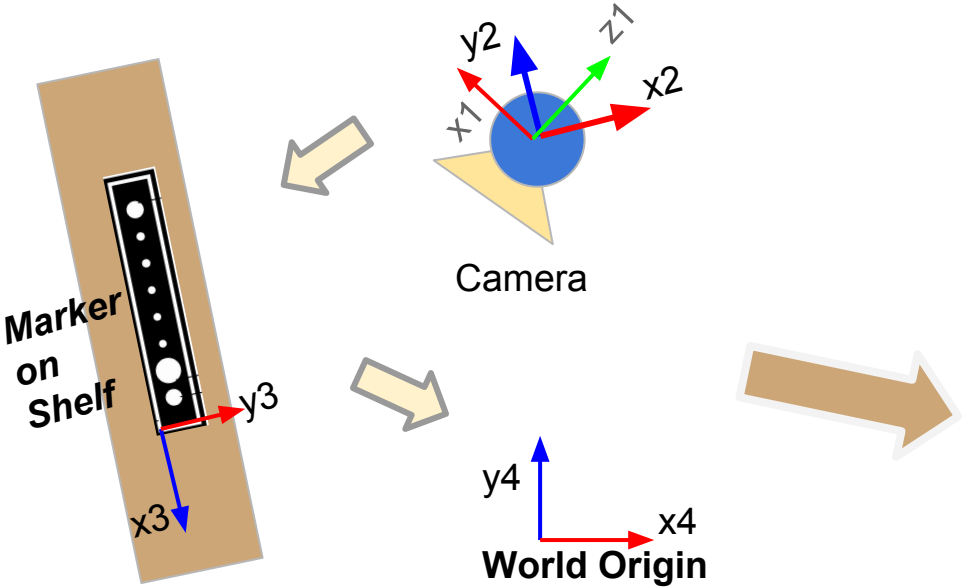
2D Frame Transformation

Transform pose estimation output of x1z1 (viewing frame) to x4y4 origin frame.

```yaml
# VITAG POSITION YAML
description:
- yaml storing x, y coor and yaw of each marker
- all respective to origin
- Date: 14 March 2018
- Version: 1.0

markers:
 1 : {x: 10, y: 30, yaw: 1.571}
 2 : {x: 40, y: 30, yaw: 1.571}
 3 : {x: 25, y: 60, yaw: 0.785}
 7 : {x: 80, y: 70, yaw: 0.785}
 14 : {x: 135, y: 70, yaw: 0.785}
 22 : {x: 150, y: 80,, yaw: 0.785}
 33 : {x: 180  y: 90   yaw: 0.785}
```

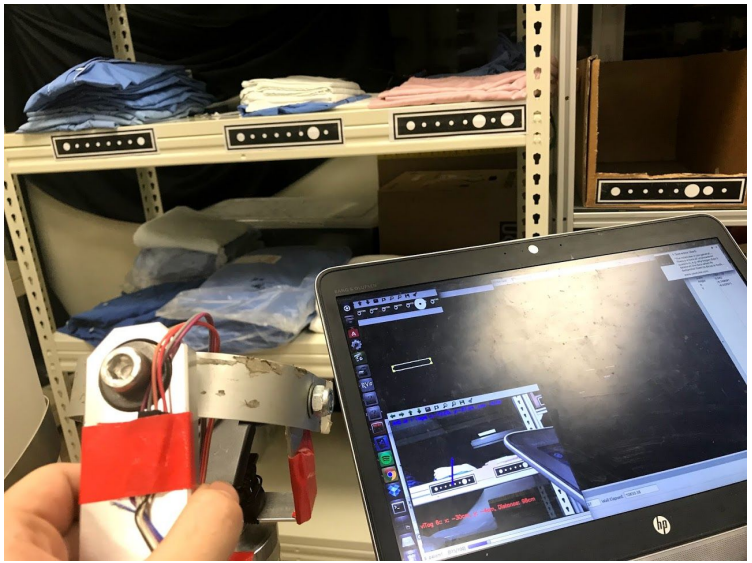
Saved VITag Position yaml file: used for determine marker's position from the world origin



Final pose estimation output:
{ X, Y, YAW }

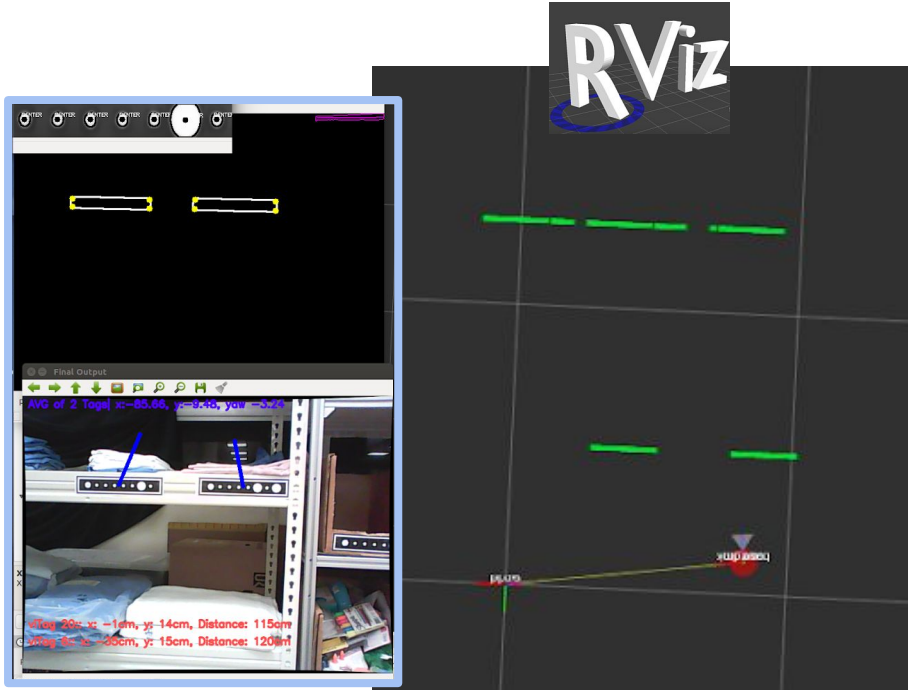
Experiment Setup with Stabilizer

2-axis stabilized gimbal: remove roll and pitch angle of the camera

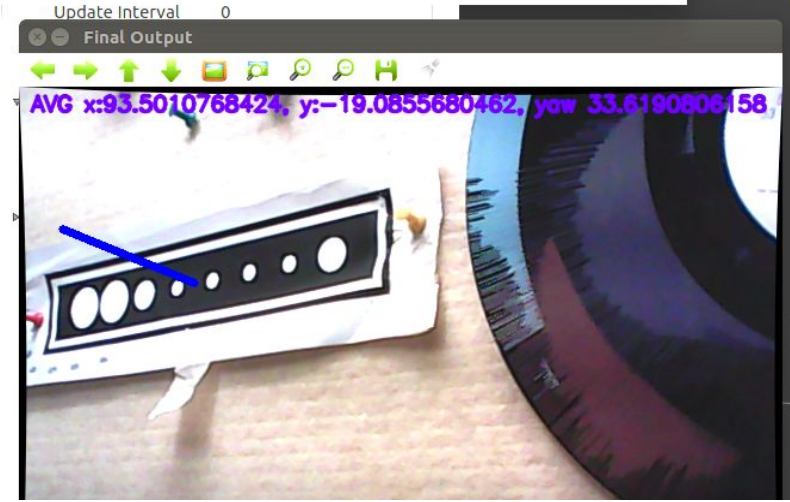


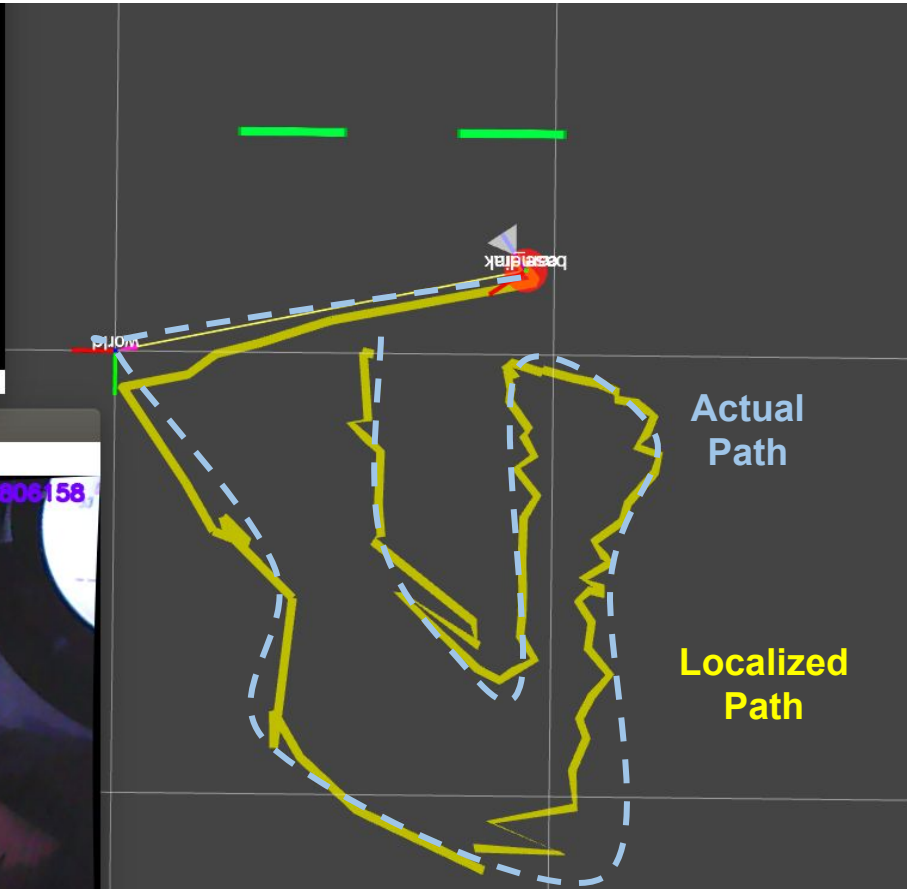
ROS and Rviz

- ROS (Robot Operating System)
  - Communication between nodes
  - ROS Rviz Package
- Rviz Package is used for visualization purpose
- markers.py
  - Tracking
  - VITag Model



Performance Result





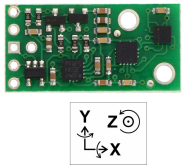
Thoughts and Evaluation

- Accomplished my FYP objective
- Huge range of improvements and add-on to improve performance
- Problems
  - Low frequency, 8 hz
  - Inconsistency of output pose estimation (huge error while further away)
  - Blind spot
  - Discrete and non-continous
- Enhance performance by incorporate supplementary sensors

https://photos.app.goo.gl/F5SZVryuGmjZ5AfD3

PHASE 2 IMPLEMENTATION:

IMU AND SYSTEM INTEGRATION



What Phase 2 is about?

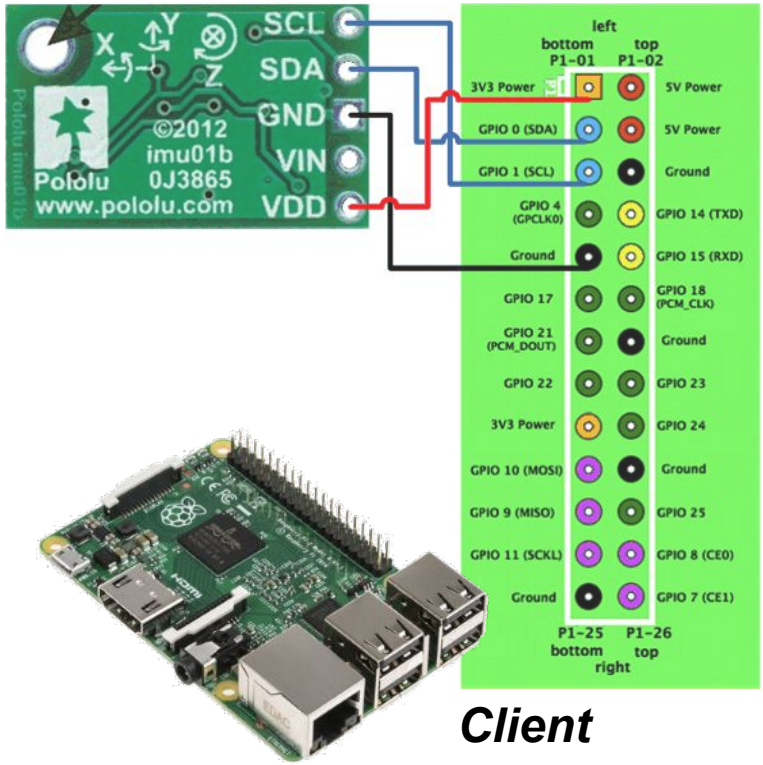
- Incorporate IMU to eliminate blind spot and discrete sensing
- Integration of IMU to enhance the Pose Estimation Result
- Integration of Tools and communicating framework
- Implementation of Kalman Filter

INTRODUCE IMU, THEN USE FUSE IMU AND CAM READING TOGETHER

Setup of RaspBerry Pi

- Raspberry pi is used to read the reading of the IMU via its GPIO pins
- SPI Communication
- Send signal to server via socket communicate (wifi network)
- Signal with 6dof and 1 timestamp

IMU



server

Acceleration(3), rotation (3), timestamp

Client

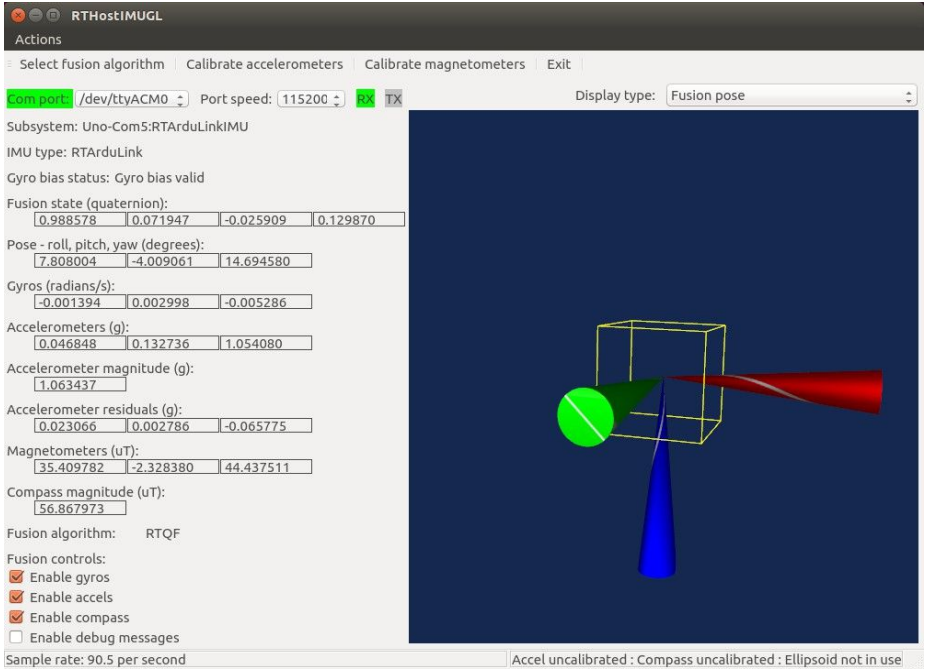
IMU Calibration

3 sensors calibration:

- Magnetometer
  - find max and min
- Accelerometer
  - Normalize to (-1 to 1)
- Gyro Sensor
  - (respective to magnetometer)

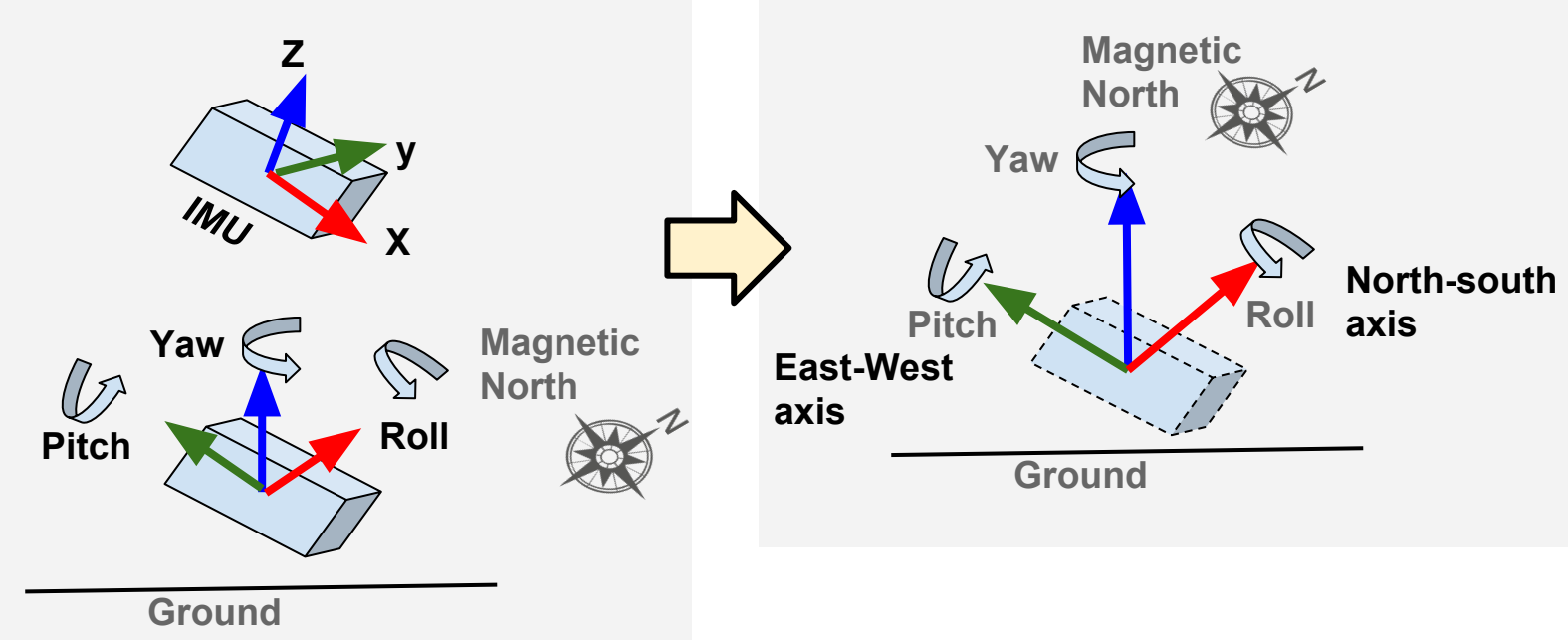
OutPut Values:

{accel_x, accel_y, accel_z,
Roll, Pitch, Yaw}



3D Frame Transformation

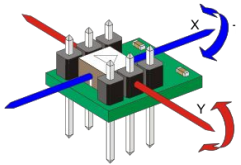
Convert to Magnetic North reference Frame (show 3D matrix)



Sensor Fusion

$$\hat{x}_k = \begin{bmatrix} Position \\ velocity \end{bmatrix}, \quad P_k = \begin{bmatrix} \Sigma_{pp} & \Sigma_{pv} \\ \Sigma_{vp} & \Sigma_{vv} \end{bmatrix}$$

output



a_k = Acceleration from encoder from t to t+1

Time Update

State Estimate: $x_k = A_k x_{k-1} + B_k a_k$

Estimated Covariance Matrix: $P_k = A_k P_{k-1} A_k T + Q_k$

$$A = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix}, \quad \hat{x}_k = \begin{bmatrix} Position \\ velocity \end{bmatrix}, \quad P_k = \begin{bmatrix} \Sigma_{pp} & \Sigma_{pv} \\ \Sigma_{vp} & \Sigma_{vv} \end{bmatrix}$$

$$z_k = \begin{bmatrix} Position \\ velocity \end{bmatrix}$$

Measurement Update

Kalman Gain: $K' = P_k H_k \hat{} T (H_k P_k H_k \hat{} T + R_k) \hat{} -1$

State Estimate: $x'_k = x_k + K' (z_k - H_k x'_k)$

Estimated Covariance Matrix: $P'_k = P_k (1 + K' H_k)$

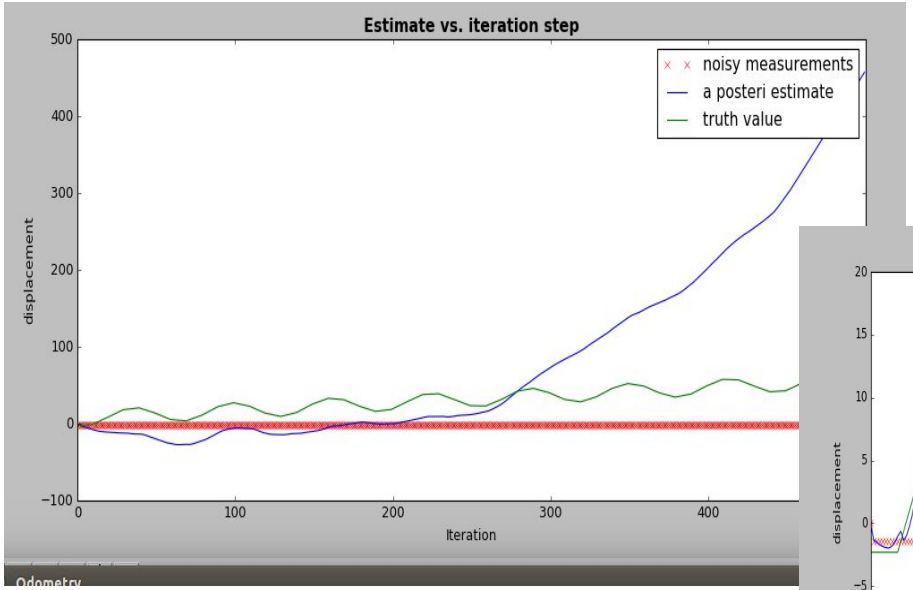
Kalman Filter simulation

Legend

***Red**: Measurement Update (camera)*

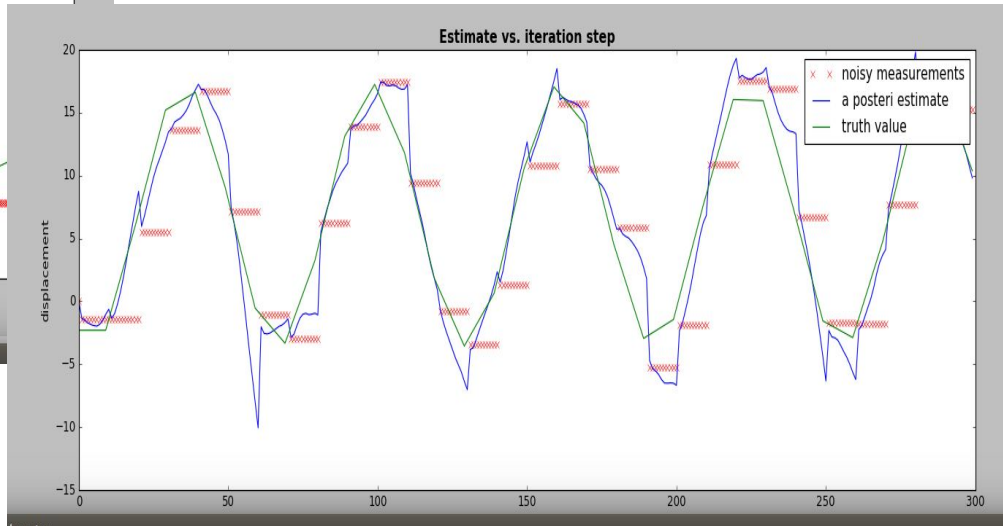
***Green**: Ground-Truth*

***Blue**: Fusion Output*

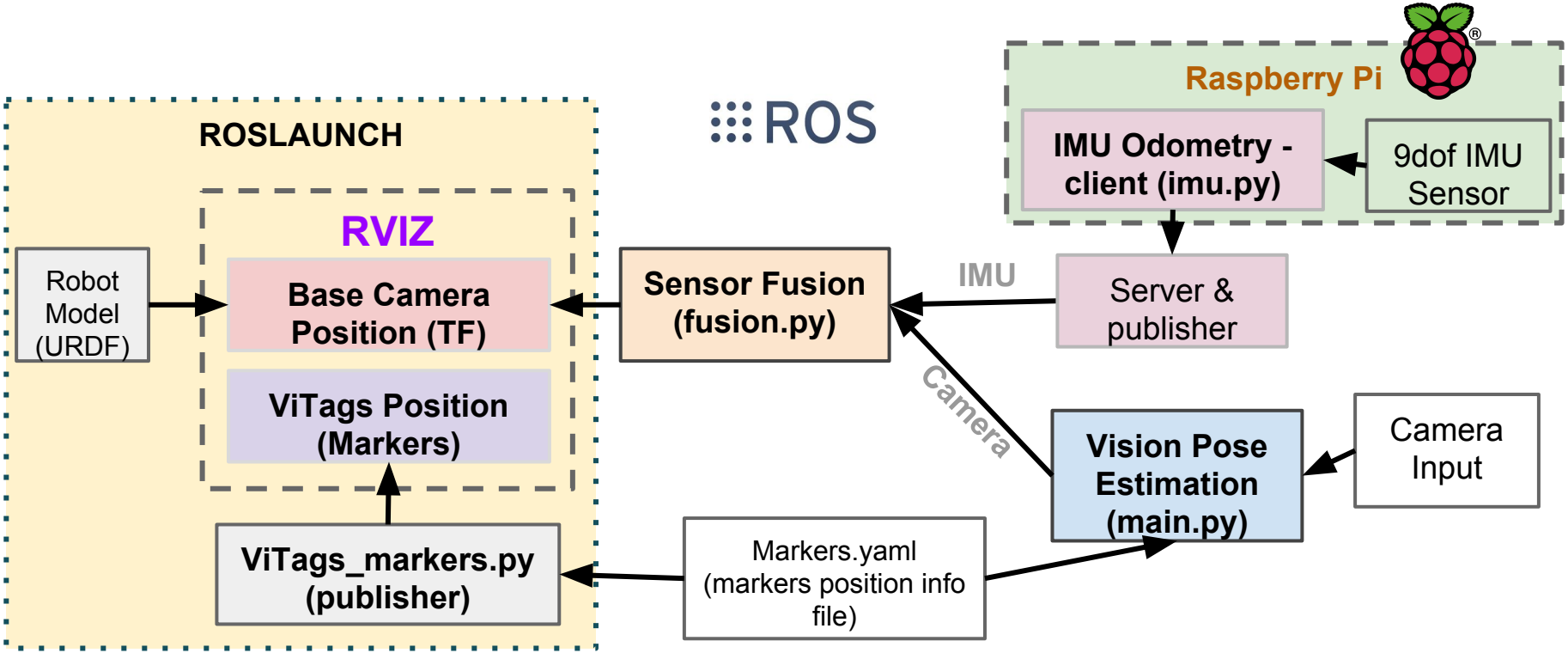


Drifting of output
(without measurement update)

Measurement Update along interval

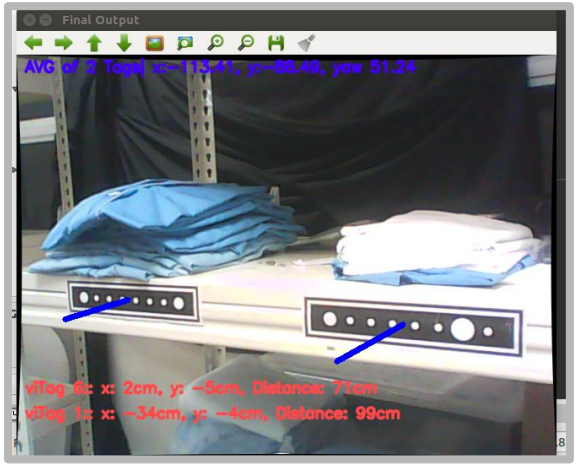


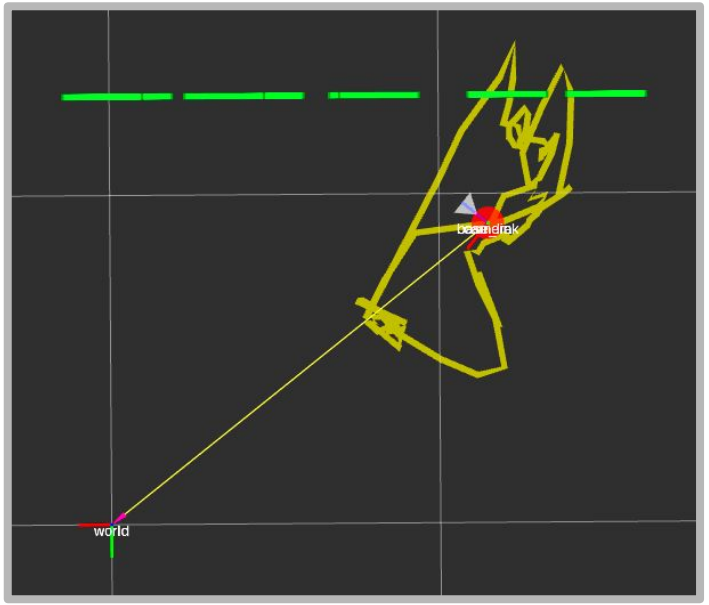
Communication FrameWork



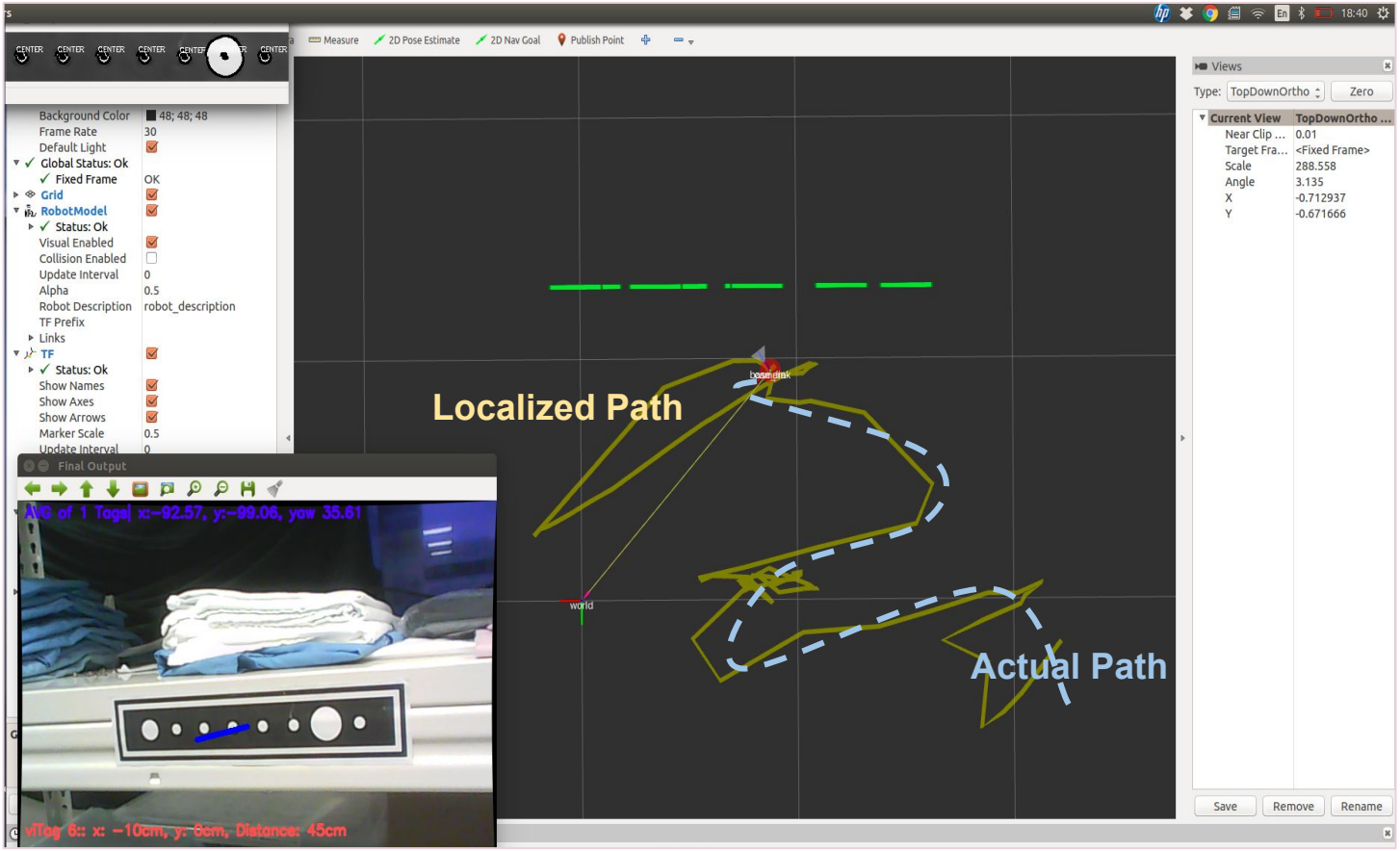
Drifting Problems

- Interval between camera frame was solved
- Severe Drifting during absence measurement update (camera)

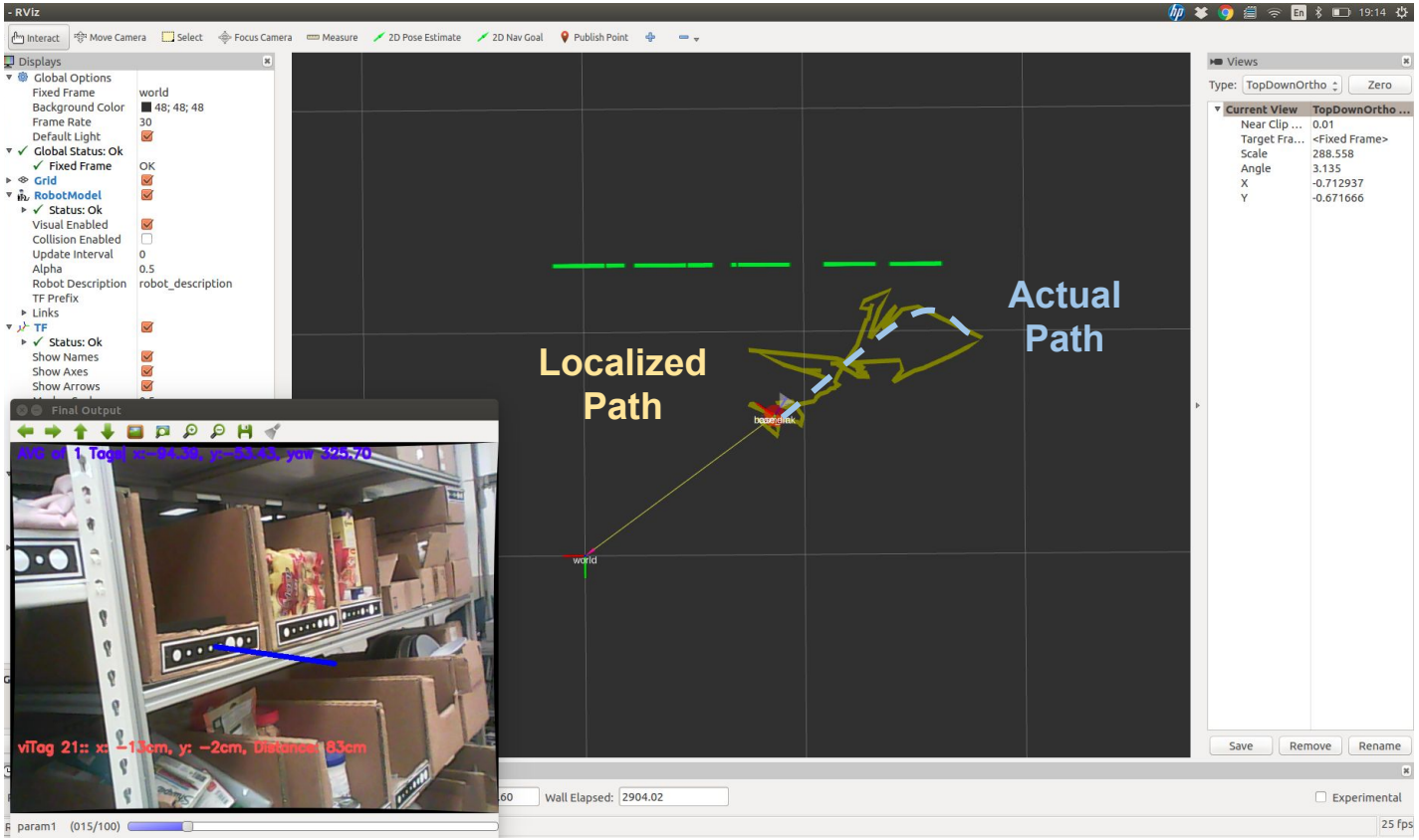




Drifting Result



Drifting Result



Thoughts and Evaluation

- Accelerometer is not reliable for relatively low acceleration value and short range tracking
- Slightlest tilt of gimbal will affect the acceleration output (due to gravity)
- IMU outputs highly reliable Rotation Matrix (absolute value)

PHASE 3 IMPLEMENTATION:

ENCODER AND FUSION ENHANCEMENT

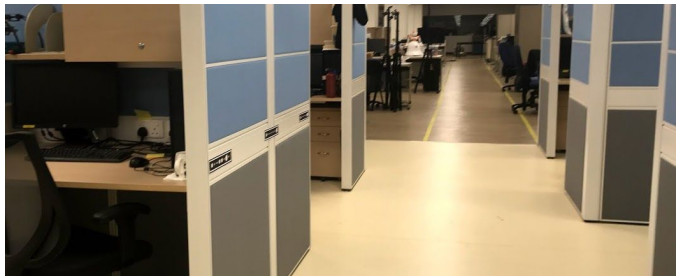
What is Phase 3 About?

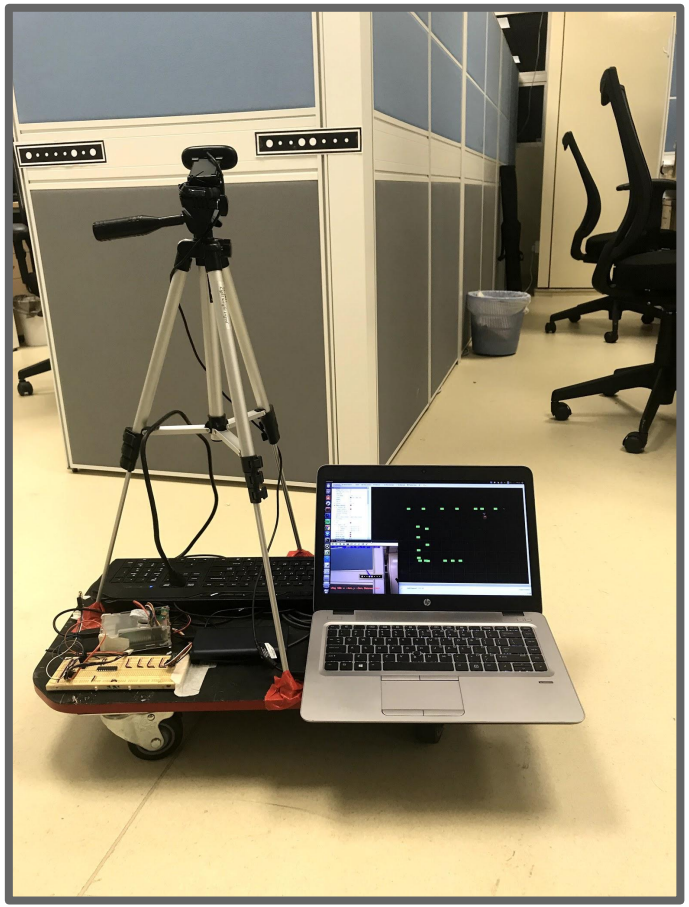
- Integrate X-Y encoders
- Fully fuse 3 type of sensor inputs in Sensor Fusion
- Utilize each sensors leading advantage
- Performance enhancement
- Redesign the experiment setup
- Proper transformation of all reference frames
- Conduct a complete localization testing and experiment

ADD ENCODERS; AND SET UP PROPER EXPERIMENT

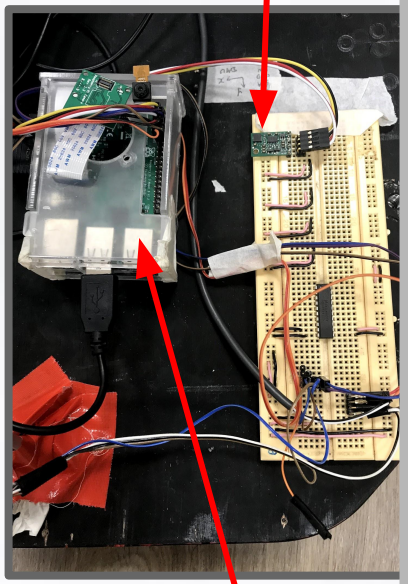
Complete Experiment Setup

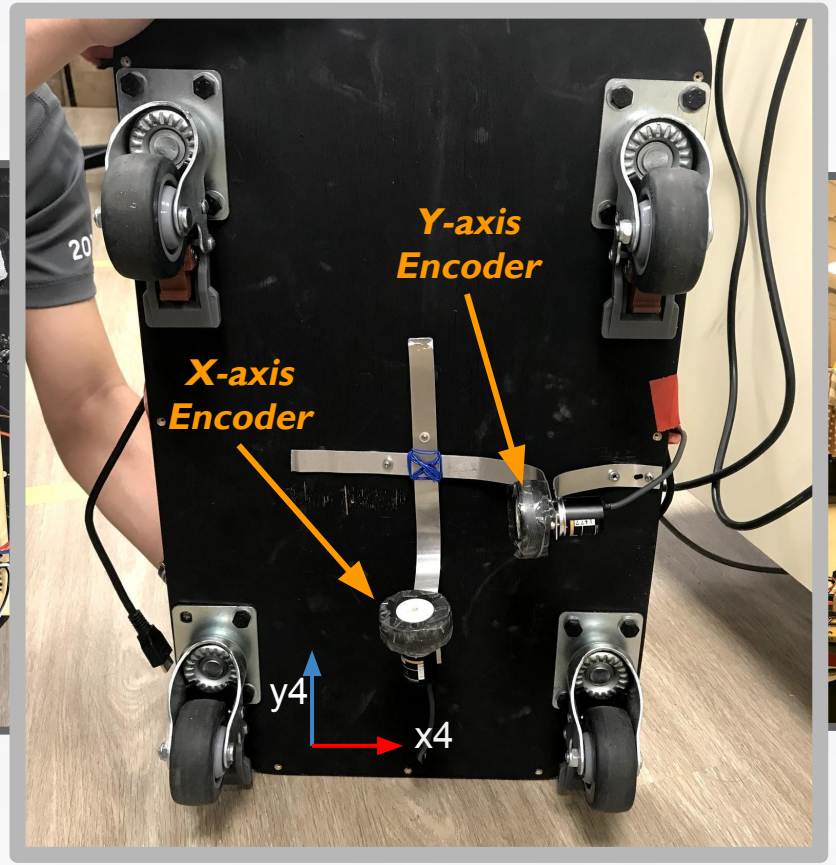
- Deploy a moving platform
- Enable a stabilized imu, camera
- Attach 2 axis encoders below to measure encoder odometry

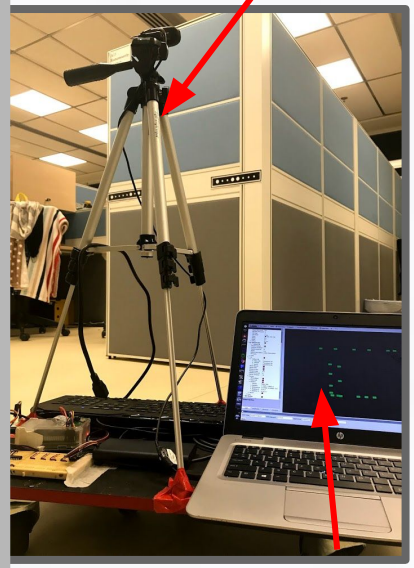




Set Up

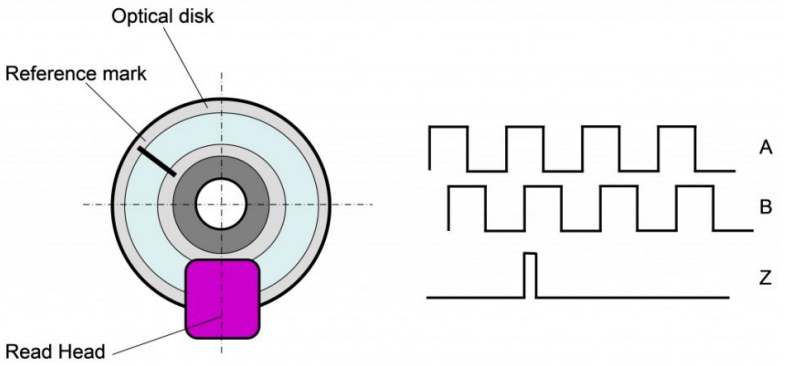

IMU
Raspberry Pi


Y-axis Encoder
X-axis Encoder
y4
x4


Camera Stand
Real-time Viewing with laptop

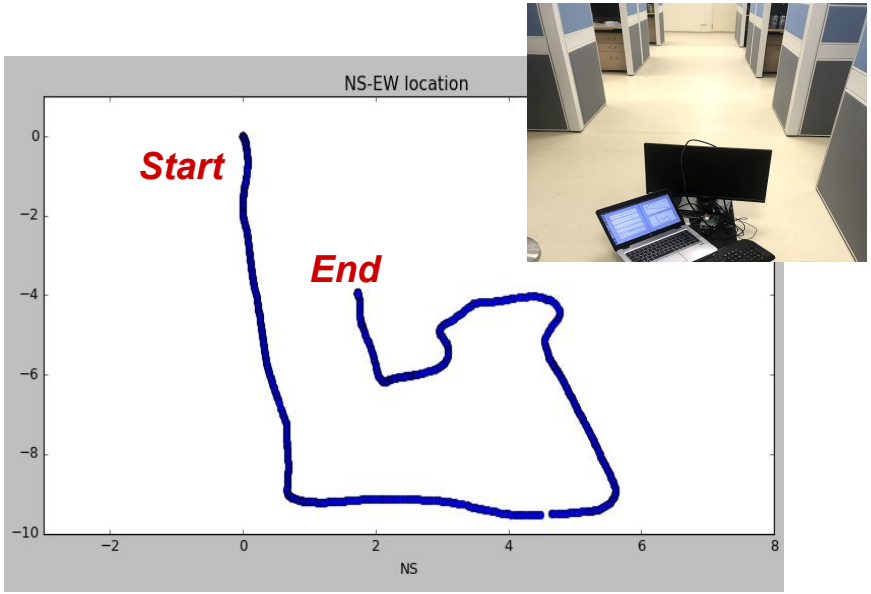
Encoder Installation

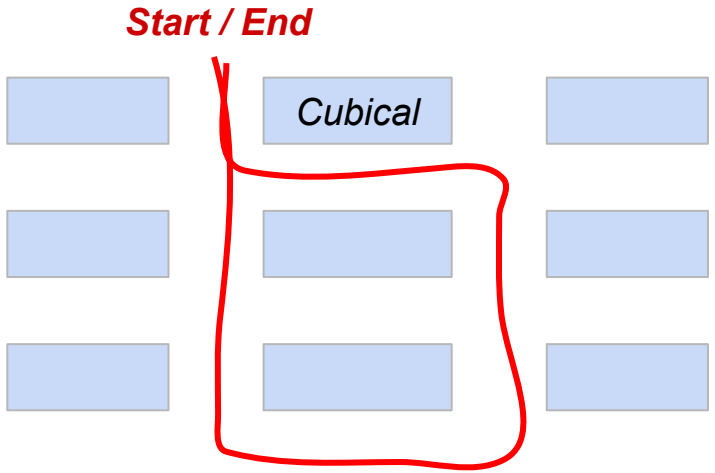
- Incremental encoder, identify rotating direction by its pulses difference
- Dual-axis, and measure displacement of moving platform
- Eliminate noisy input from acceleration; Unlike IMU, encoder can determine the velocity value



Encoder Odometry

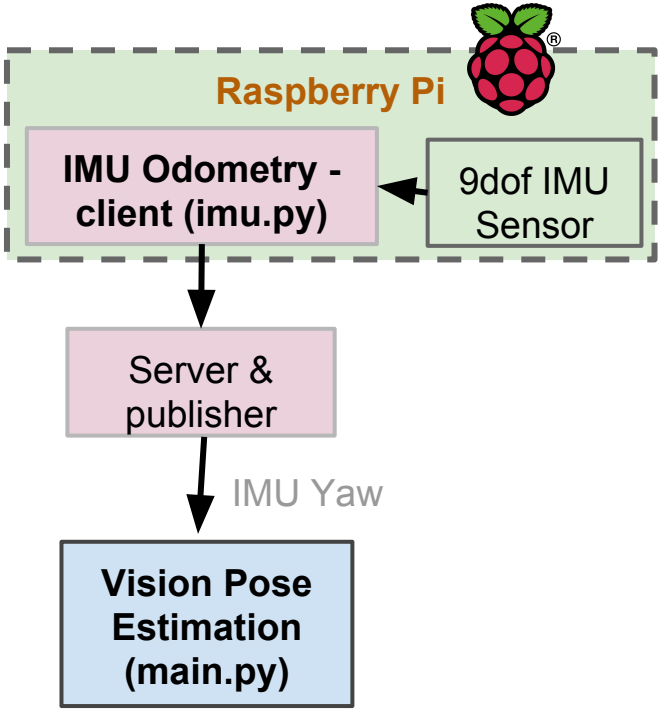
Experiment shown the drifting effect of 2-axis encoder odometry





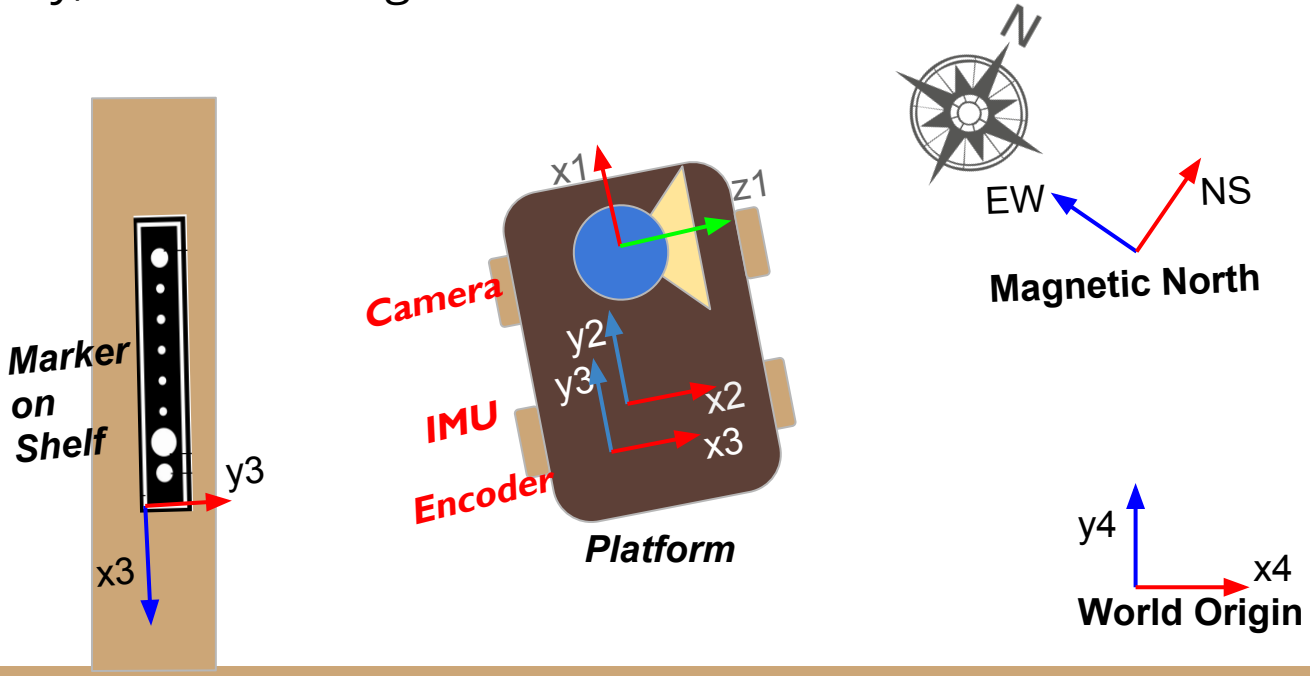
IMU Yaw angle to Vision Node

- Homography transformation inheres small deviation on rotation matrix (*yaw angle)
- Amplified along 2d transformation
- IMU provides absolute rotation matrix
- Yaw angle from IMU respect to magnetic north is provided to Vision Sensing Node



2D Transformation of Frame

2D transformation will transform all frames to the absolute Magnetic North frame, lastly, the world origin frame.



Reconstruct Sensor Fusion

D_k = Displacement from encoder from t to t+1

Z_k = absolute position from Camera

x_k = absolute position

P_k = Variance

Time Update

State Estimate: $x_k = x_{k-1} + D_k$

Estimated Covariance Matrix: $P_k = A_k P_{k-1} A_k^T + Q_k$

Measurement Update

Kalman Gain: $K' = P_k H_k^T (H_k P_k H_k^T + R_k)^{-1}$

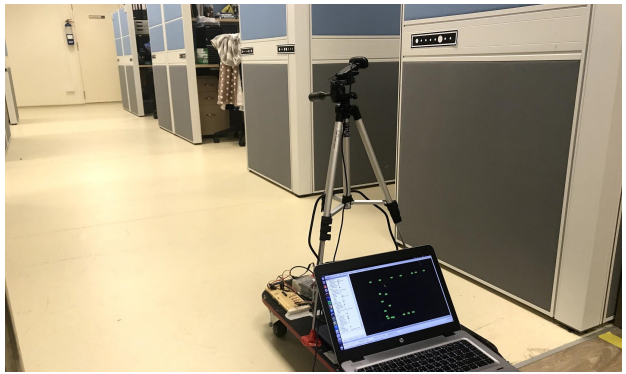
State Estimate: $x'_k = x_k + K' (z_k - H_k x'_k)$

Estimated Covariance Matrix: $P'_k = P_k (1 + K' H_k)$

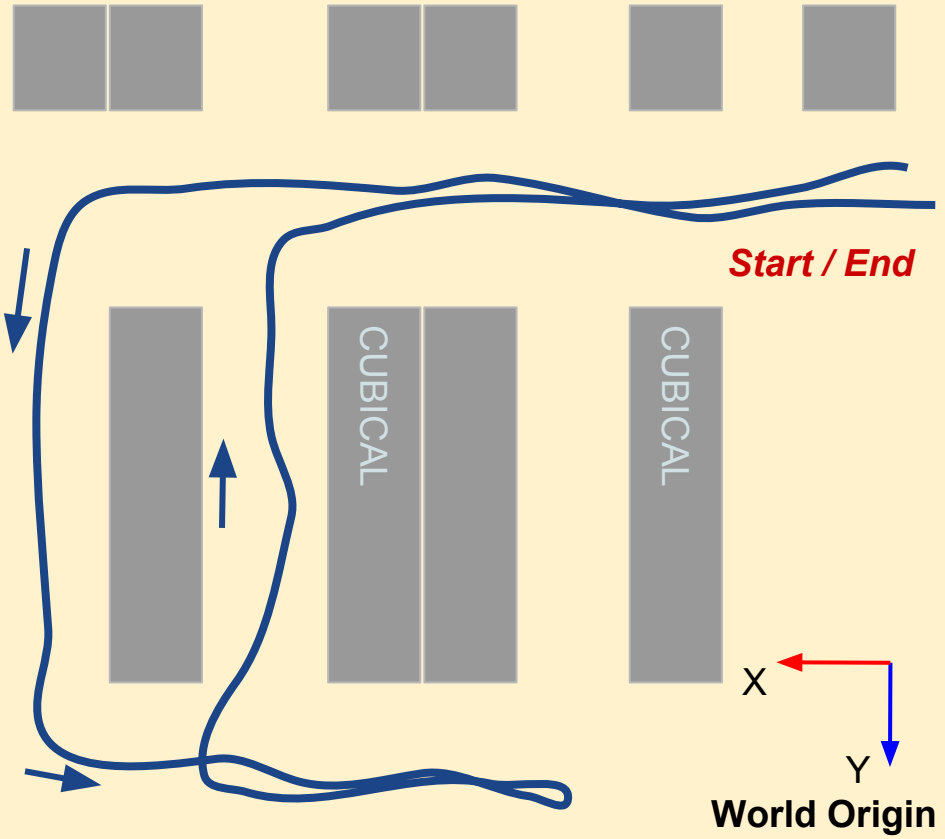
output

Experiment

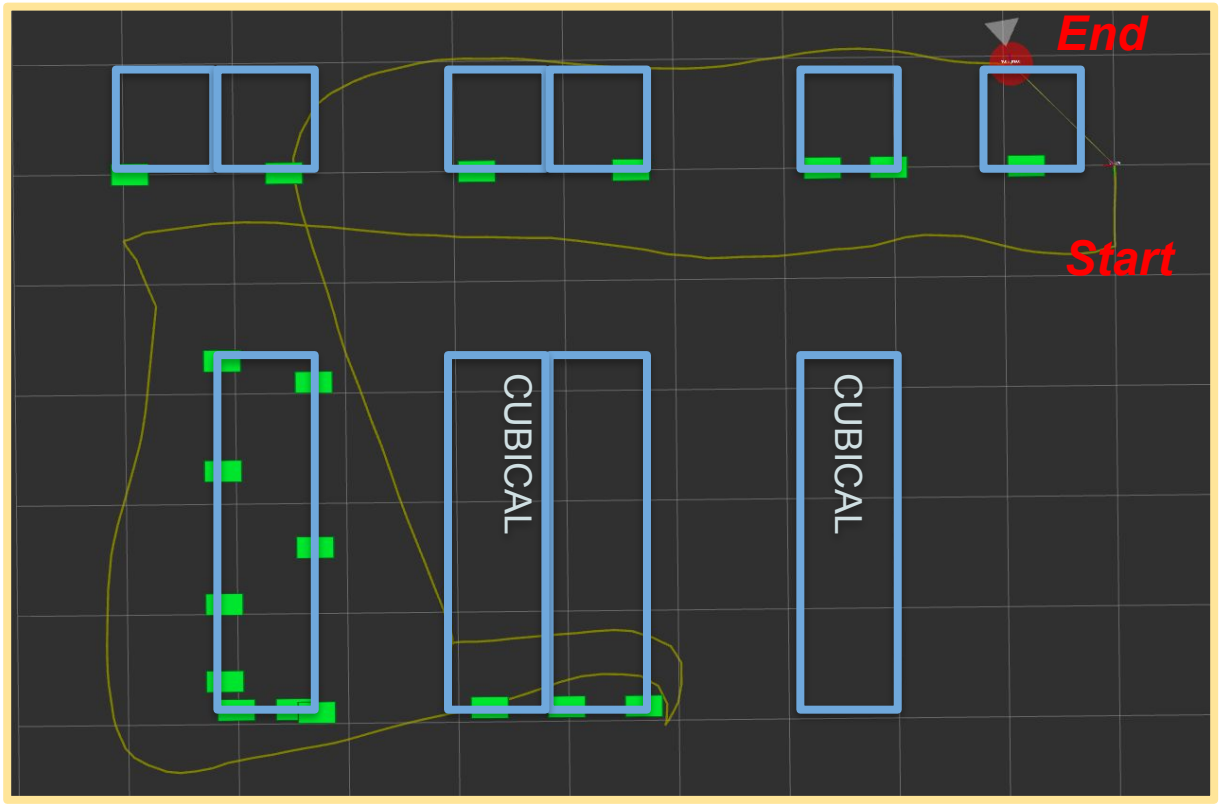
Platform manually being pushed around the corridors, according to the testing path. Measured paths are recorded and compared.



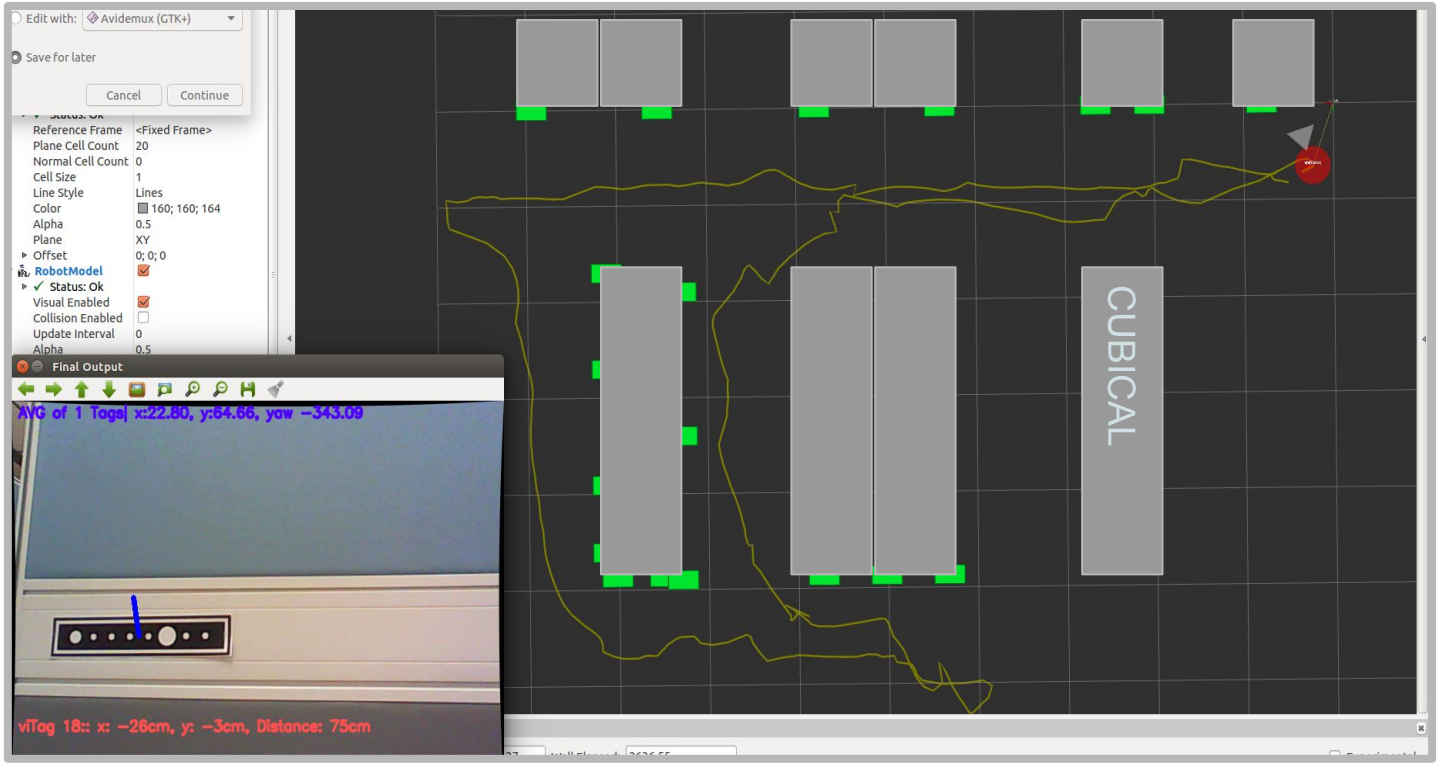
Map with actual testing path



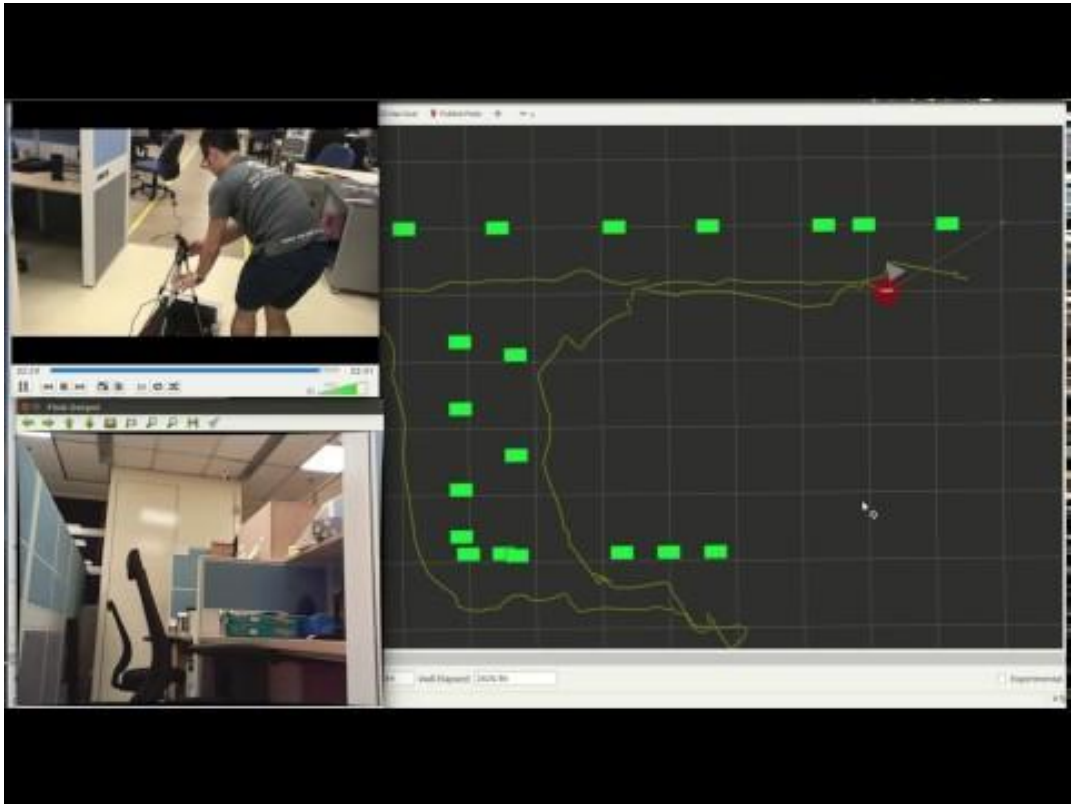
Performance ::: *Pure Encoder Odometry*



Performance ::: *Vision + Encoder Odometry + IMU*

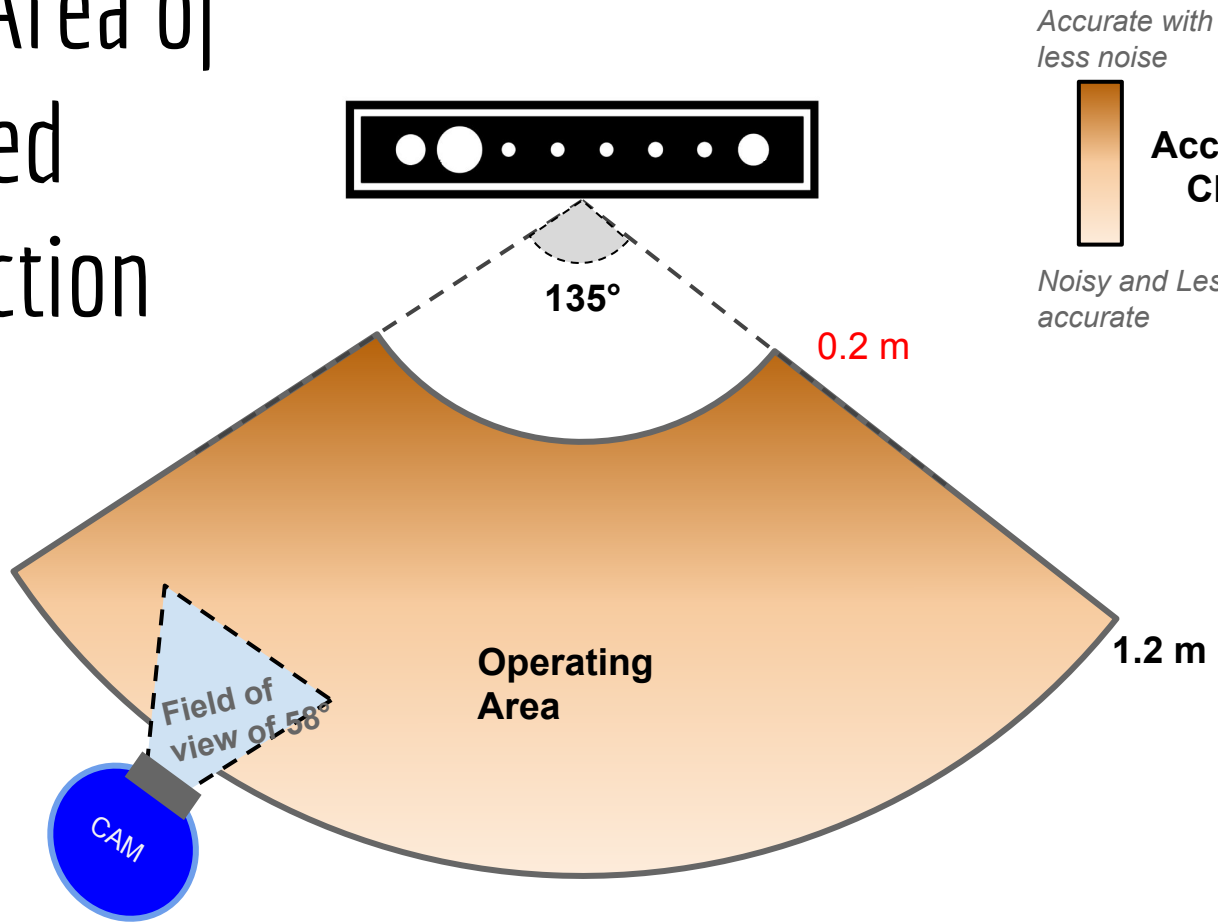


VIDEO ::: *Vision + Encoder Odometry + IMU*



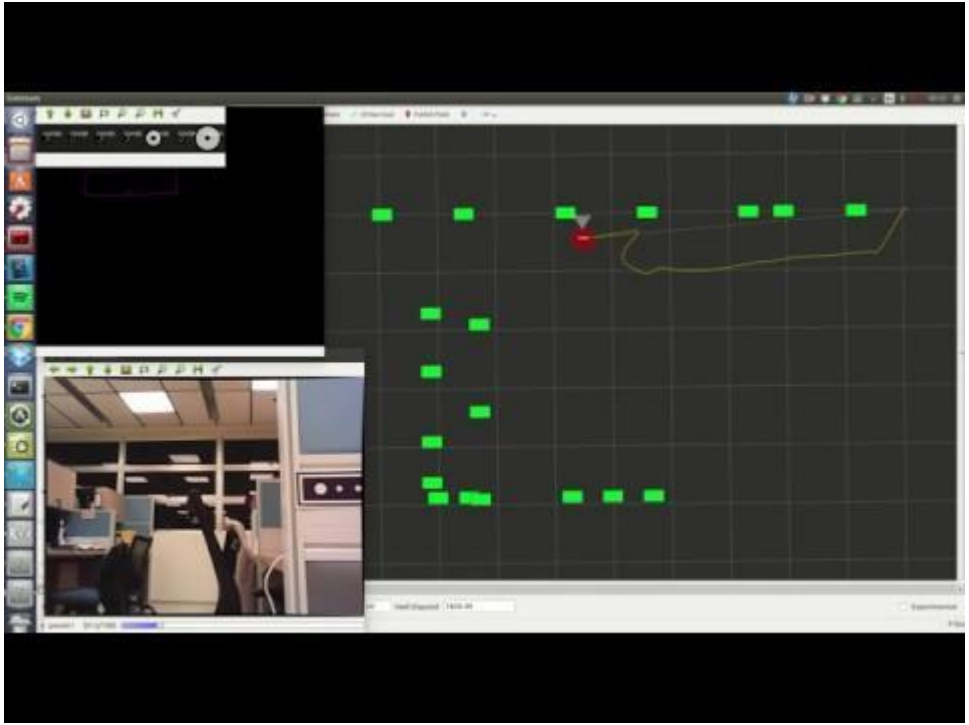
DISCUSSIONS

Operating Area of Vision-based VITag Detection



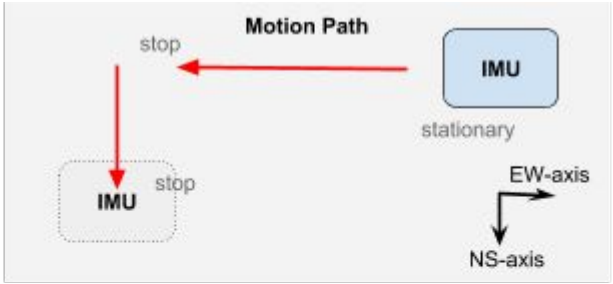
VIDEO ::: *Pure Vision*

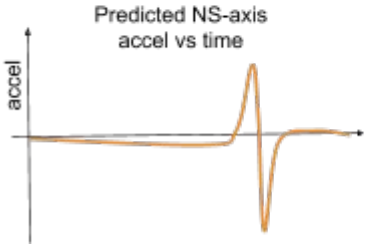
Discrete and non-continuous

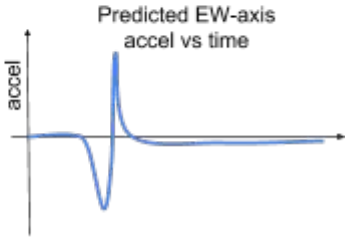


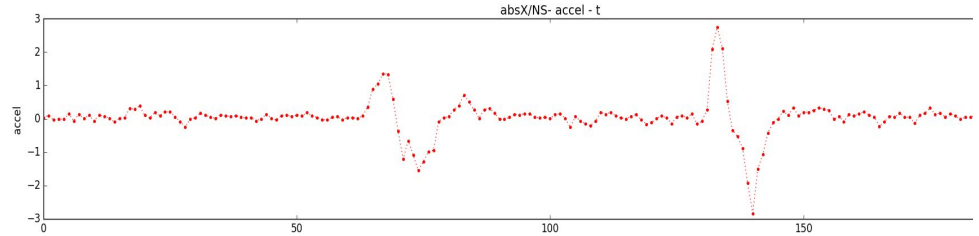
Challenge - Noisy IMU

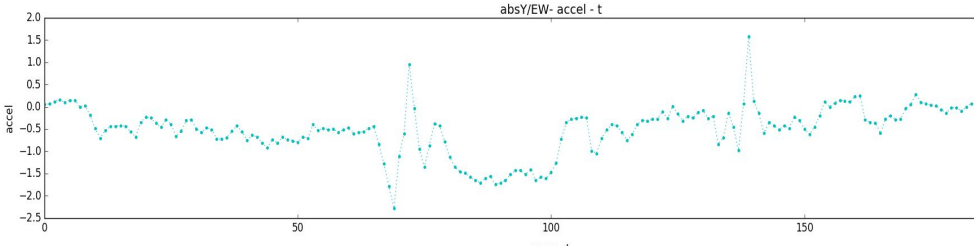
- Noisy with the presence of gravity









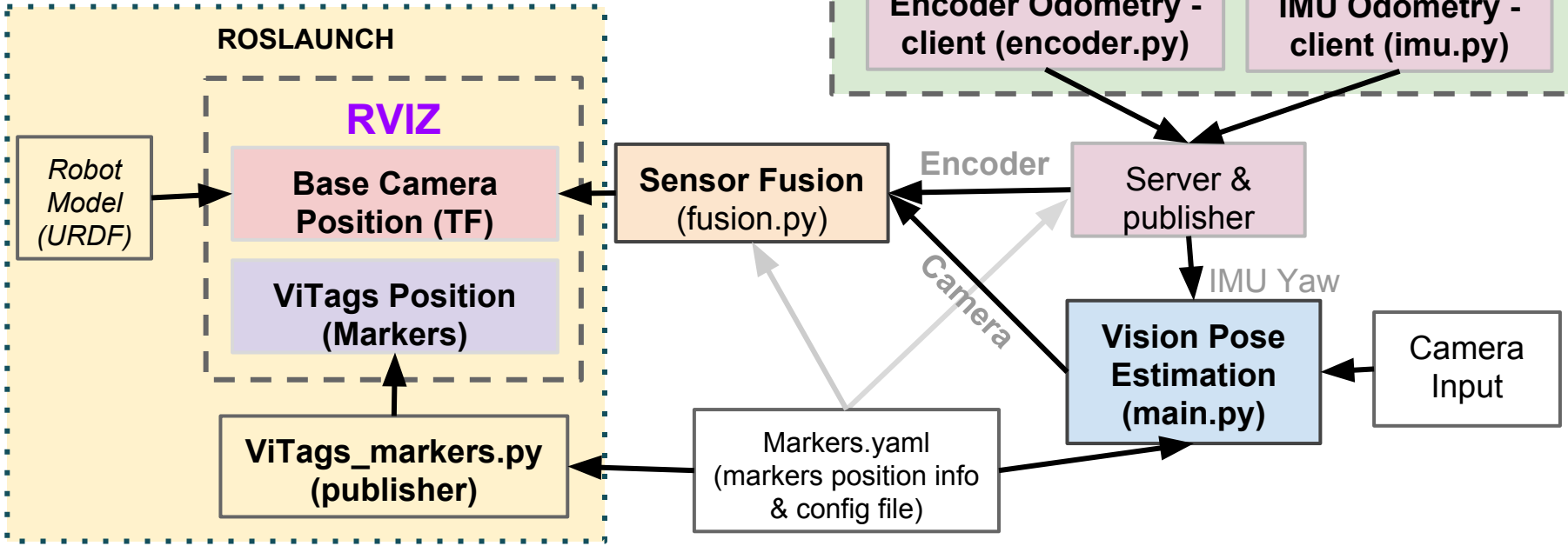


Overall Testing Summary

1. **IMU Odometry:** Severe drifting, not accurate
2. **Vision:** Discrete and non-continuous, imprecise yaw angle
3. **Encoder Odometry + IMU:** Stable, odometry drift
4. **Vision + IMU Odometry:** Being affected by IMU, shifting during vision interval
5. **Vision + Encoder + IMU:**
   - ➢ **Best Among all;**
   - ➢ **Stable and accurate;**
   - ➢ **Within precision of +- 15cm if have markers in every 2.5m interval**

Finalized Process Architecture

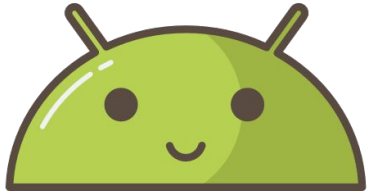
ROS



CONCLUSION

CONCLUSION

- Vision and odometry Localization is useful for wide range of navigation system.
- Manage to produce a working prototype with reliable and accurate result (within +-15cm)
- Improvements:
  - Proper configuration of encoders tracking
  - Equip with multiple wide angle HD cameras
  - Optimization of vision sensing (improve sampling rate)
  - Test on Robot



THANK YOU

NANYANG TECHNOLOGICAL UNIVERSITY

MAE
Mechanical and Aerospace Engineering Club