# SIMULTANEOUS LOCALIZATION AND MAPPING USING THE EXTENDED KALMAN FILTER

AGATA SZEREMETA

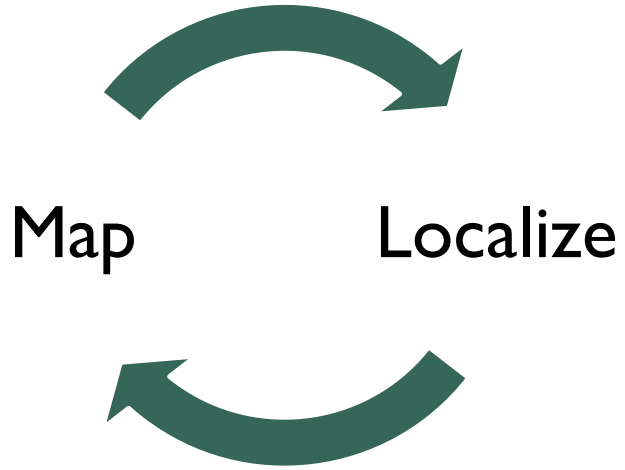
ESS 5430: FINAL PROJECT PRESENTATION

# EKF SLAM INTRODUCTION

# SIMULTANEOUS LOCALIZATION AND MAPPING (SLAM)

- SLAM is method of mapping an unknown environment, while simultaneously localizing the moving mapping sensor within it

- Problem requires information regarding the mapping agent's pose in order to map the environment while also requiring such a map to determine the robot's relative pose and trajectory
  - This "Chicken and Egg" problem makes the task difficult to complete

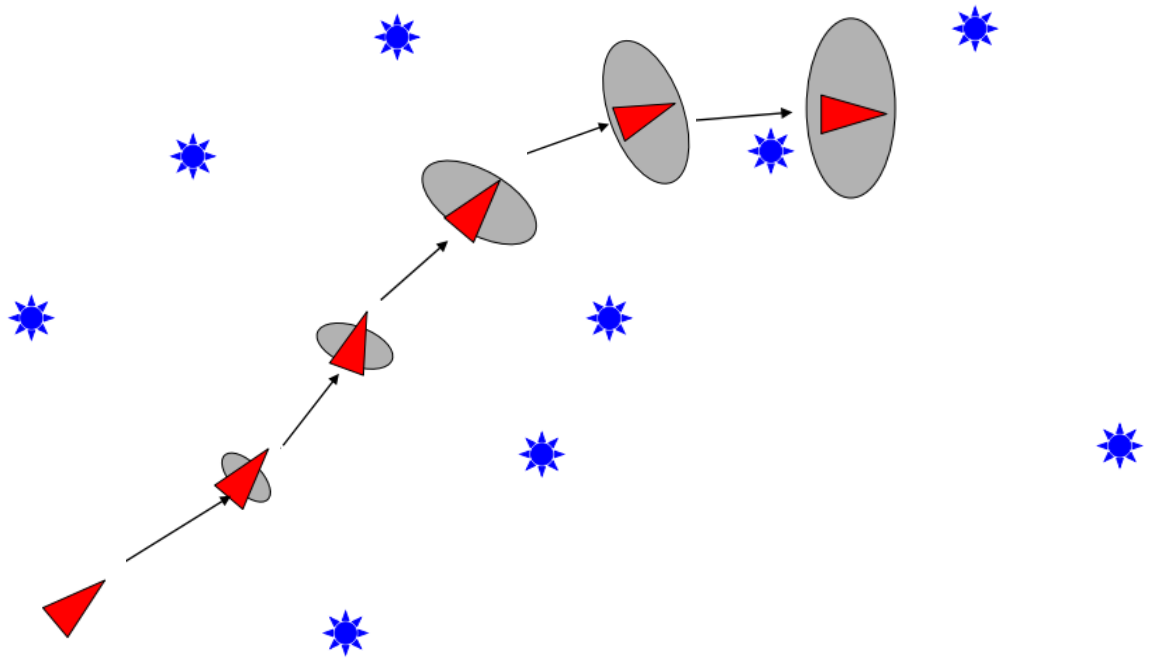


# INTRODUCTION: THE NEED FOR EKF BASED SLAM

- To understand the need for EKF in SLAM, consider a moving agent (robot) attempting to map an environment.
- The agent uses:
  - Auxiliary sensors (ex. cameras, LiDAR, RADAR) to detect features (to be mapped) in environment
  - Additional sensor data (ex. odometry from wheel encoders) to localize itself in environment
- Unfortunately both sensors are imperfect (as is case in real life)
  - Sensor data contains noise

# INTRODUCTION: THE NEED FOR EKF SLAM

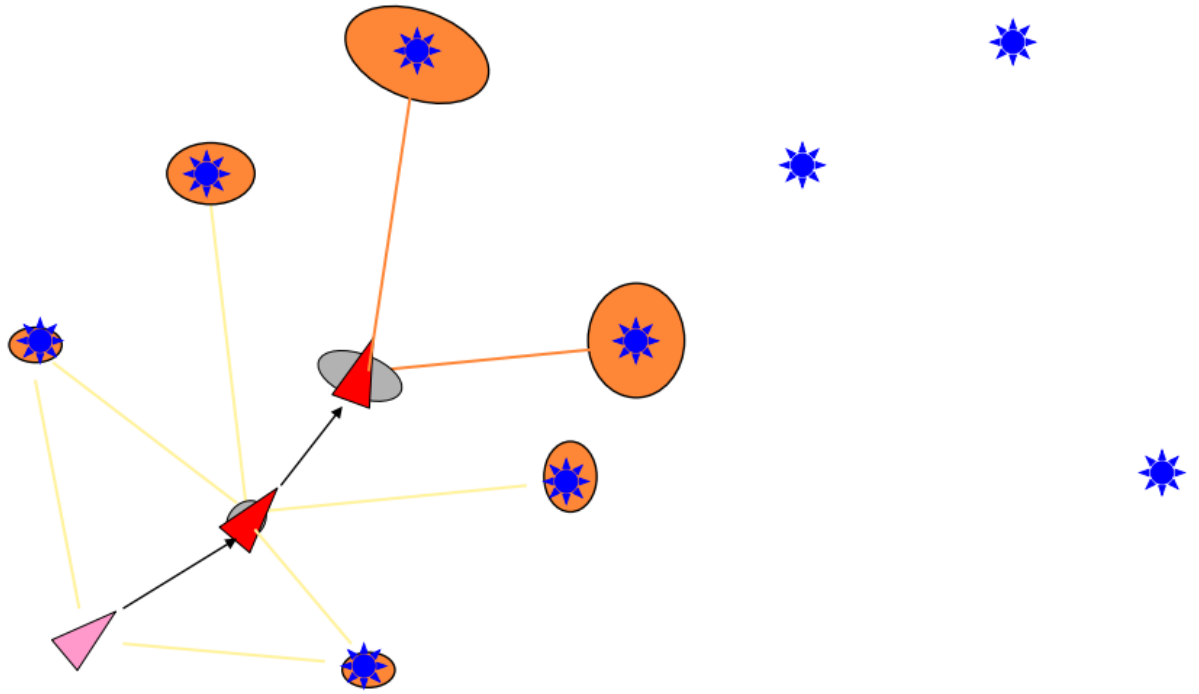
- By using only dead reckoning techniques, uncertainty errors in the odometry data propagate throughout agent's motion[1]



[1]

# INTRODUCTION: THE NEED FOR EKF SLAM

- These pose errors, in addition to the errors of the measuring sensor, also propagate into the uncertainty of the map generated from observing the landmarks



[1]

# INTRODUCTION: THE NEED FOR EKF SLAM

- To minimize the uncertainty of the agent pose and map, data fusion techniques can be applied on data collected through auxiliary sensors
  - Observations of landmarks can be used to improve the estimated pose of the agent and map

- Most common and significant technique: EKF SLAM[2]

# BACKGROUND: THE EXTENDED KALMAN FILTER (EKF)

- Kalman Filtering (KF) is an optimal method of estimating and filtering states in linear Gaussian systems[3]

- The Extended Kalman Filter (EKF) is an extension to the Kalman Filter which allows it to be applicable to cases where the state transitions and measurements are nonlinear.

# BACKGROUND: THE EXTENDED KALMAN FILTER (EKF)

- The general KF and EKF algorithm follow two main steps:

    1. State Prediction
    2. State Correction

- The state is first predicted from an expected control input
- It is then used to predict what a measurement to a landmark will be
- Finally, the disparity between this predicted measurement and the true measurement is used to compute a gain term, by which the prediction needs to be updated by to provide the correct state
- This updates both the localization and mapping aspects

# BACKGROUND: THE EKF ALGORITHM

1: **Algorithm Extended_Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**
2: $\bar{\mu}_t = g(u_t, \mu_{t-1})$
3: $\bar{\Sigma}_t = G_t \, \Sigma_{t-1} \, G_t^T + R_t$
4: $K_t = \bar{\Sigma}_t \, H_t^T (H_t \, \bar{\Sigma}_t \, H_t^T + Q_t)^{-1}$
5: $\mu_t = \bar{\mu}_t + K_t(z_t - h(\bar{\mu}_t))$
6: $\Sigma_t = (I - K_t \, H_t) \, \bar{\Sigma}_t$
7: return $\mu_t, \Sigma_t$

[2]

| | Kalman filter | EKF |
|---|---|---|
| state prediction (Line 2) | $A_t \, \mu_{t-1} + B_t \, u_t$ | $g(u_t, \mu_{t-1})$ |
| measurement prediction (Line 5) | $C_t \, \bar{\mu}_t$ | $h(\bar{\mu}_t)$ |

[2]

# BACKGROUND: EKF SLAM

- EKF and SLAM algorithms combine to:
  - Localize an agent, with *nonlinear* motion, in an environment by predicting its future position and adjusting this prediction based on the observations of features in the environment
  - Simultaneously, map features in the environment based on the agent's localization

Table 1: Comparison of Events in EKF and SLAM Algorithms[5]

| Event | Meaning in EKF Algorithm | Meaning in SLAM Algorithm |
|---|---|---|
| Agent Moves | EKF Prediction | Agent Motion |
| Sensor Detects New Landmark | State Augmentation | Landmark Initialization |
| Sensor Observes Known Landmark | EKF Correction | Map Correction |

# BACKGROUND: EKF SLAM IN ROBOTICS

- Often, EKF SLAM algorithm is used in robotics to localize a robot that[3]:
  - Moves in a 2D plane, using velocity based motion model as its control input
  - Performs Range-Bearing observations to point landmarks

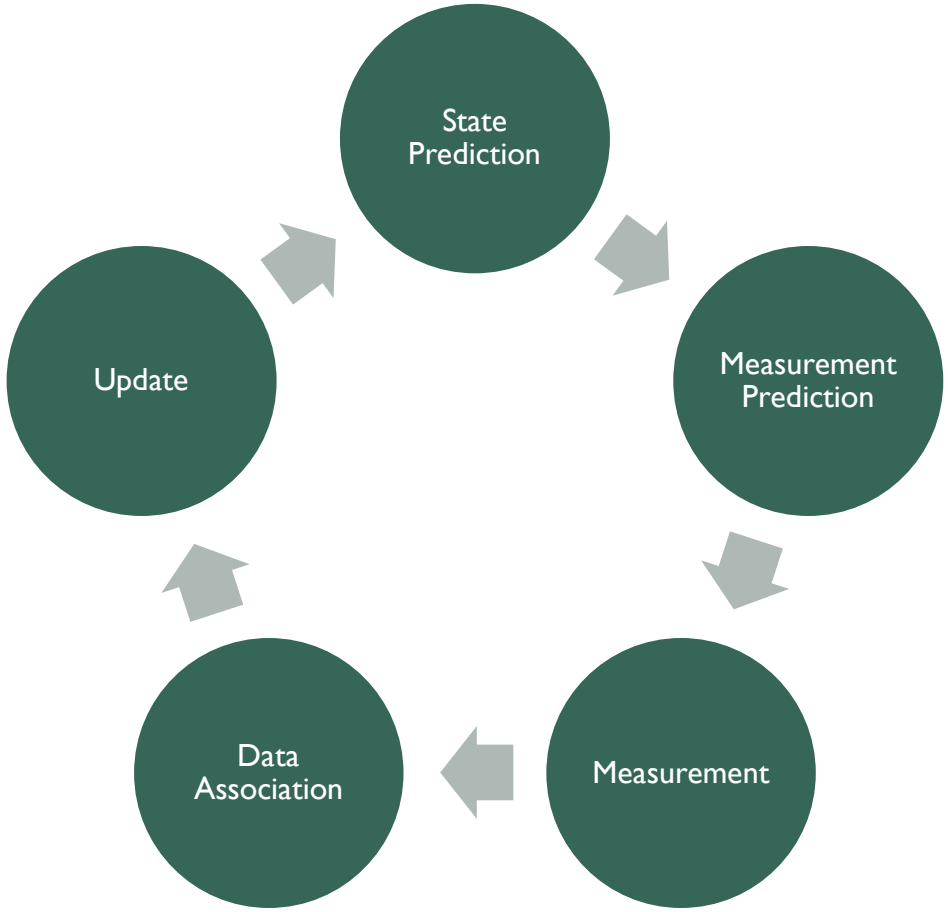
- The robot can:
  - Either know total number of landmarks in environment or not
  - Either know to which landmark it is observing or not

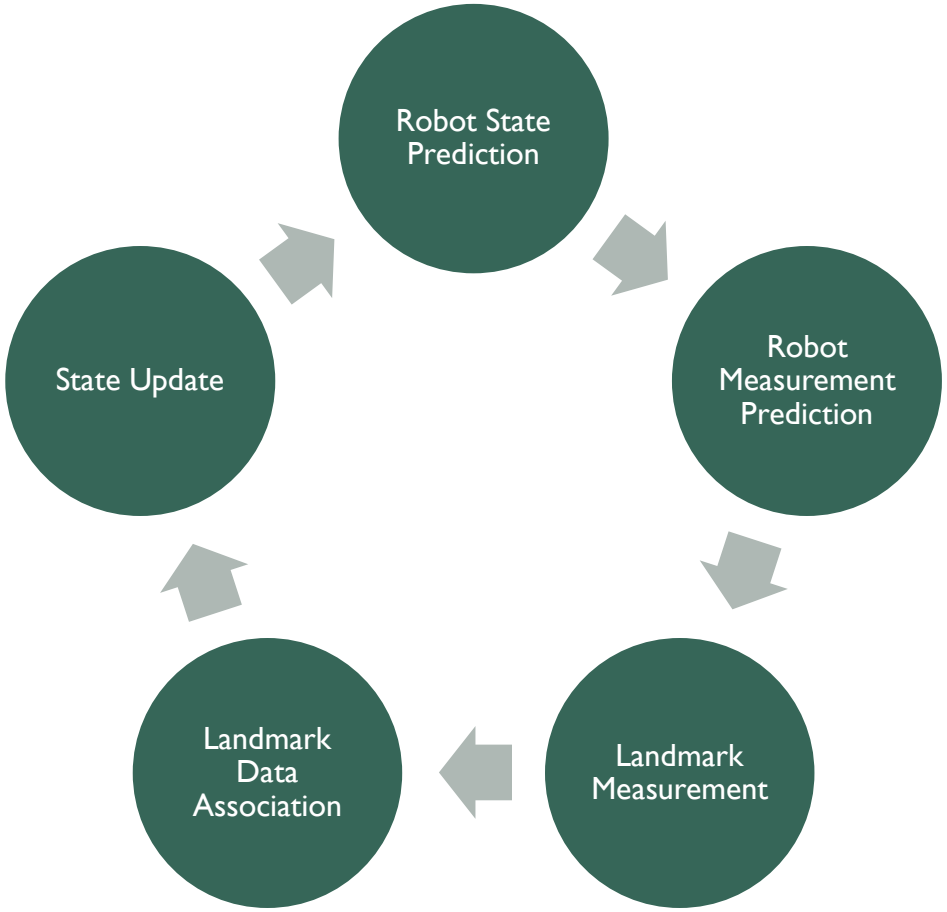
- Often, does not know either and must perform some sort of data association to determine whether is observing a *new* or *old* landmark[2]

# BACKGROUND: EKF SLAM WITH UNKNOWN DATA ASSOCIATION

- This unknown data association in EKF SLAM is considered the general EKF SLAM problem
- Because sensor does not know the landmark it is observing, needs to determine correspondence with previously mapped landmarks
  - If already observed landmark, use information to adjust estimated pose and map
  - If new landmark, augment it to the map and adjust pose accordingly
- Many correspondence techniques exist, however, commonly used in EKF SLAM is Maximum Likelihood

# BACKGROUND: SUMMARY OF EKF SLAM STEPS



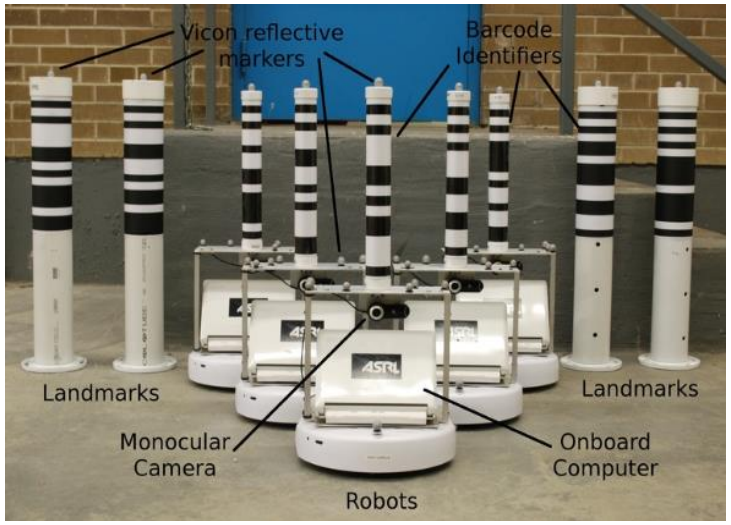
# EKF SLAM IMPLEMENTATION FOR PROJECT

# PROJECT: EKF SLAM IMPLEMENTATION

- An EKF SLAM algorithm was trialed on two systems:

    1. Data provided by University of Toronto Aerospace Studies (UTIAS) Autonomous Space Robotics Lab[4]
    2. Self-collected data using moving Microsoft Kinect RGB-Depth sensor

- Complete implementation using external data
- Incomplete implementation using Kinect data




[6]


[7]

# PROJECT: EXTERNAL DATA SYSTEM

- Part of dataset collection provided by Autonomous Space Robotics Lab for educational purposes
- Originally for Cooperative Multi-Robot Localization and Mapping
  - Individual robot data could be extracted from dataset
- Beneficial because provided all necessary data to implement EKF SLAM and assess results
- Downloadable at:

http://asrl.utias.utoronto.ca/datasets/mrclam/#Download

# PROJECT: EXTERNAL DATA SYSTEM

- The collection consisted of 9 datasets, which each included:
  - Odometry data of each of 5 robots
  - Range and bearing measurements from robots, found by processing images of onboard monocular camera
  - Accurate ground truth data for all robot poses and 15 landmarks in environment

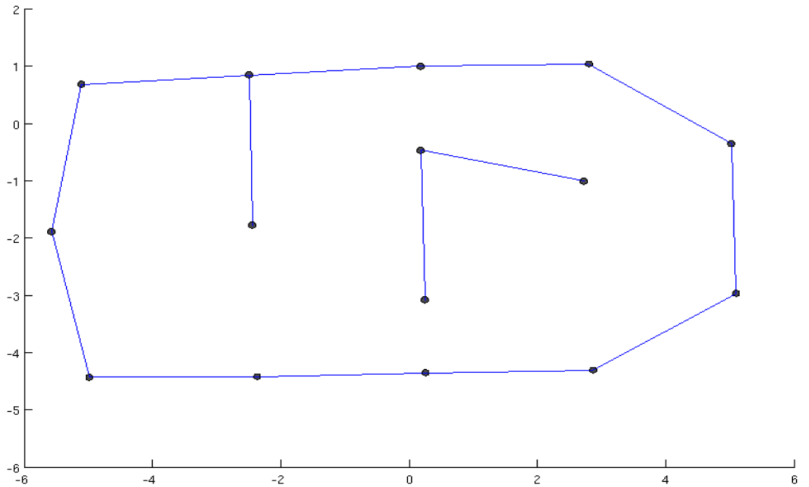
- Due to barcodes on landmarks, robots were aware of the landmark associated with each observation
  - This was ignored and EKF SLAM with unknown data association was still implemented

# PROJECT: EXTERNAL DATA SYSTEM

- Robot #1's observations in dataset #9 was chosen for initial study
  - Algorithm is expected to function on any robot number in any dataset

- Chose dataset #9 because of included occlusions:

[6]




[6]

# PROJECT: EKF SLAM PSEUDOCODE

State Prediction Steps

Observation Steps

Data Association Steps

State Update Steps

**Algorithm EKF_SLAM**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, N_{t-1})$:

$N_t = N_{t-1}$

$$F_x = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$$

$$\bar{\mu}_t = \mu_{t-1} + F_x^T \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} + \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t \Delta t) \\ \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t \Delta t) \\ \omega_t \Delta t \end{pmatrix}$$

$$G_t = I + F_x^T \begin{pmatrix} 0 & 0 & \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t \Delta t) \\ 0 & 0 & \frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t \Delta t) \\ 0 & 0 & 0 \end{pmatrix} F_x$$

$\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + F_x^T R_t F_x$

$$Q_t = \begin{pmatrix} \sigma_r & 0 & 0 \\ 0 & \sigma_\phi & 0 \\ 0 & 0 & \sigma_s \end{pmatrix}$$

for all observed features $z_t^i = (r_t^i \;\; \phi_t^i \;\; s_t^i)^T$ do

$$\begin{pmatrix} \bar{\mu}_{N_t+1,x} \\ \bar{\mu}_{N_t+1,y} \\ \bar{\mu}_{N_t+1,s} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{t,x} \\ \bar{\mu}_{t,y} \\ s_t^i \end{pmatrix} + r_t^i \begin{pmatrix} \cos(\phi_t^i + \bar{\mu}_{t,\theta}) \\ \sin(\phi_t^i + \bar{\mu}_{t,\theta}) \\ 0 \end{pmatrix}$$

for $k = 1$ to $N_t + 1$ do

$$\delta_k = \begin{pmatrix} \delta_{k,x} \\ \delta_{k,y} \end{pmatrix} = \begin{pmatrix} \bar{\mu}_{k,x} - \bar{\mu}_{t,x} \\ \bar{\mu}_{k,y} - \bar{\mu}_{t,y} \end{pmatrix}$$

$q_k = \delta_k^T \delta_k$

$$\hat{z}_t^k = \begin{pmatrix} \sqrt{q_k} \\ \text{atan2}(\delta_{k,y}, \delta_{k,x}) - \bar{\mu}_{t,\theta} \\ \bar{\mu}_{k,s} \end{pmatrix}$$

$$F_{x,k} = \begin{pmatrix} 1 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 1 & 0 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 1 & 0 \cdots 0 & 0 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 1 & 0 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 0 & 1 & 0 & 0 \cdots 0 \\ 0 & 0 & 0 & 0 \cdots 0 & 0 & 0 & 1 & 0 \cdots 0 \end{pmatrix}$$

$$H_t^k = \frac{1}{q_k} \begin{pmatrix} \sqrt{q_k}\delta_{k,x} & -\sqrt{q_k}\delta_{k,y} & 0 & -\sqrt{q_k}\delta_{k,x} & \sqrt{q_k}\delta_{k,y} & 0 \\ \delta_{k,y} & \delta_{k,x} & -1 & -\delta_{k,y} & -\delta_{k,x} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} F_{x,k}$$

$\Psi_k = H_t^k \, \bar{\Sigma}_t \, [H_t^k]^T + Q_t$

$\pi_k = (z_t^i - \hat{z}_t^k)^T \Psi_k^{-1} (z_t^i - \hat{z}_t^k)$

endfor

$\pi_{N_t+1} = \alpha$

$j(i) = \underset{k}{\text{argmin}} \; \pi_k$

$N_t = \max\{N_t, j(i)\}$

$K_t^i = \bar{\Sigma}_t \, [H_t^{j(i)}]^T \Psi_{j(i)}^{-1}$

endfor

$\mu_t = \bar{\mu}_t + \sum_i K_t^i \, (z_t^i - \hat{z}_t^{j(i)})$

$\Sigma_t = (I - \sum_i K_t^i \, H_t^{j(i)}) \, \bar{\Sigma}_t$

return $\mu_t, \Sigma_t$

[2]

# PROJECT: MODIFICATIONS TO PSEUDOCODE

- Unfortunately, exact errors for the motion model and measurement sensors were not provided
  - Their values were arbitrarily chosen
- Maximum likelihood correspondence was used with Mahalanobis Distance
  - For each observed landmark, the distance between that landmark and all previously observed was computed
  - If the found distance was greater than a defined threshold, then landmark was **potentially** new
  - To classify as entire new, the landmark also had to be more unique/different from its closest associated match

# IMPLEMENTATION RESULTS

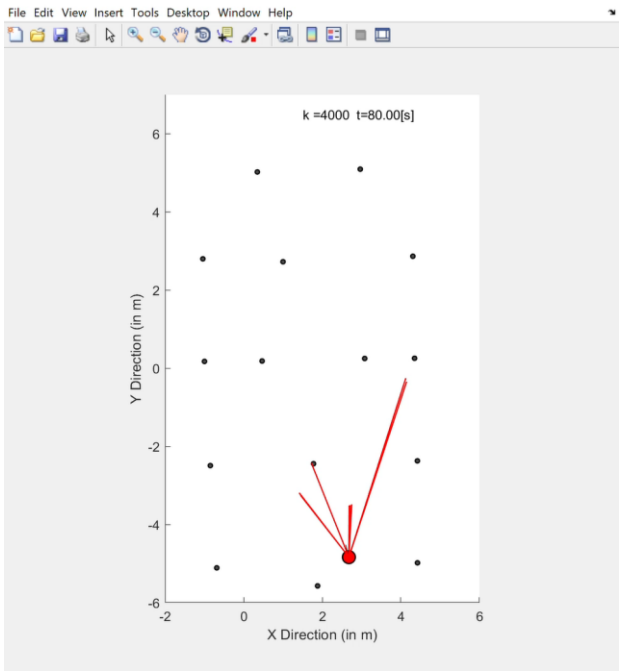
# PROJECT: RESULTS OF EKF SLAM ON EXTERNAL DATA

- Algorithm is able to estimate pose and landmark location from the given data
  - Unfortunately, trajectory does not appear to be correct (discussed in next slides)
- Run time: 64.8999 seconds or 1.0817 minutes
- Statistics on error between ground truth and estimated trajectories:

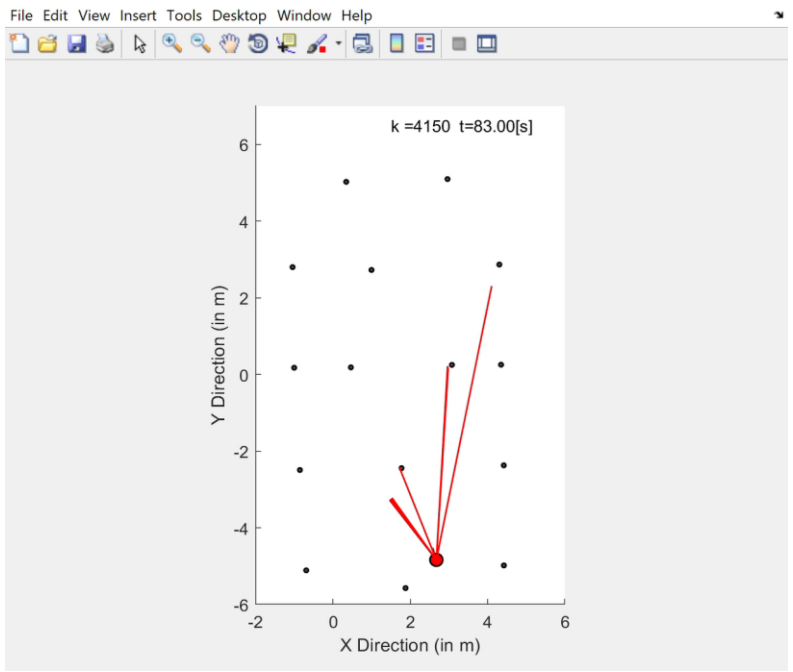
| Statistic | X Value (m) | Y Value (m) | Theta Value (rad) |
|---|---|---|---|
| Minimum | -16.2101 | -14.1675 | -6.2701 |
| Maximum | 2.9653 | 7.7197 | 6.2249 |
| Mean | -6.2890 | -2.6183 | -0.0790 |
| Standard Deviation | 4.8670 | 5.9465 | 2.4976 |

# PROJECT: RESULTS OF EKF SLAM ON EXTERNAL DATA

- Observe robot to be localized outside of barrier wall (which should have been physically impossible)
- Comparing to animation of ground truth trajectory, observe EKF SLAM estimated position to be more sporadic




Estimated Trajectory




Ground Truth Trajectory

# PROJECT: RESULTS OF EKF SLAM ON EXTERNAL DATA

- Possible explanation for errors:
  - In observation extraction code
  - Introduced when modifying given result animation script into function
  - Poor noise parameters (therefore, poor estimations and predictions)

  - **Possible errors in provided data**

# PROJECT: RESULTS OF EKF SLAM ON EXTERNAL DATA

- When running same code on different dataset (ex. Robot #4 of Dataset #1), have results closer to what is expected

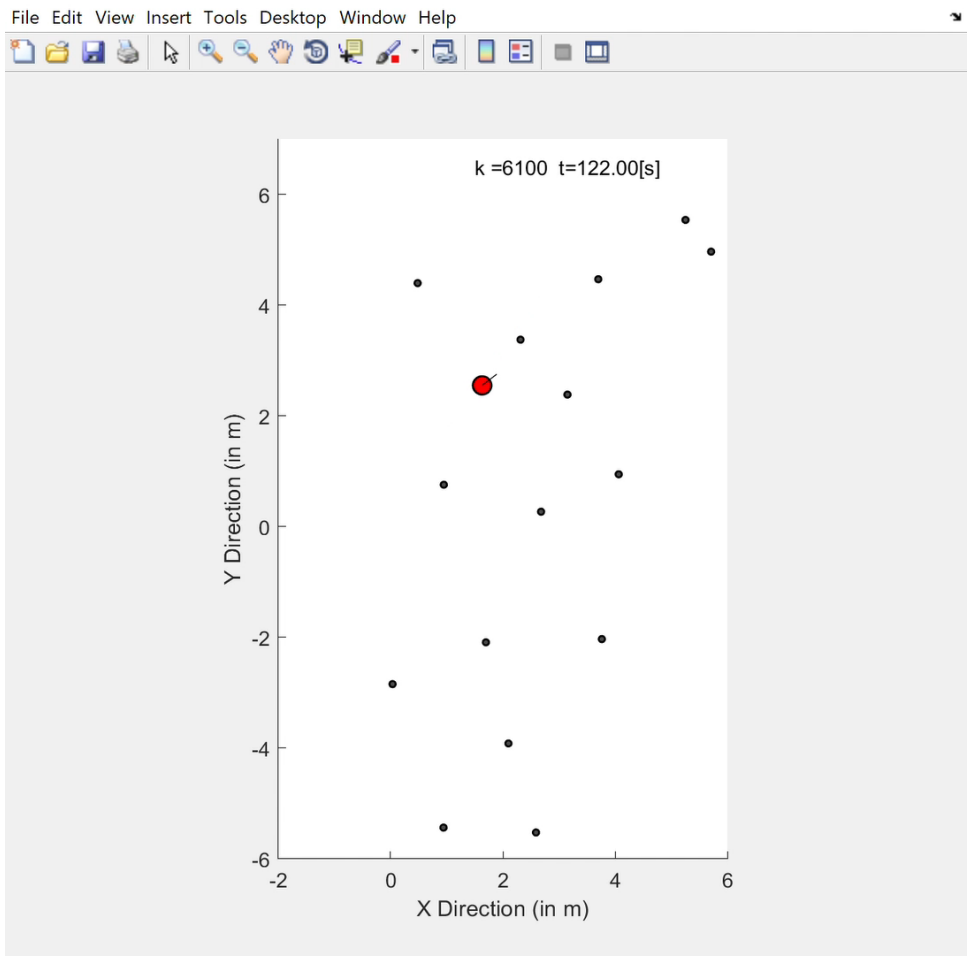
- Run time: 22.4804 seconds

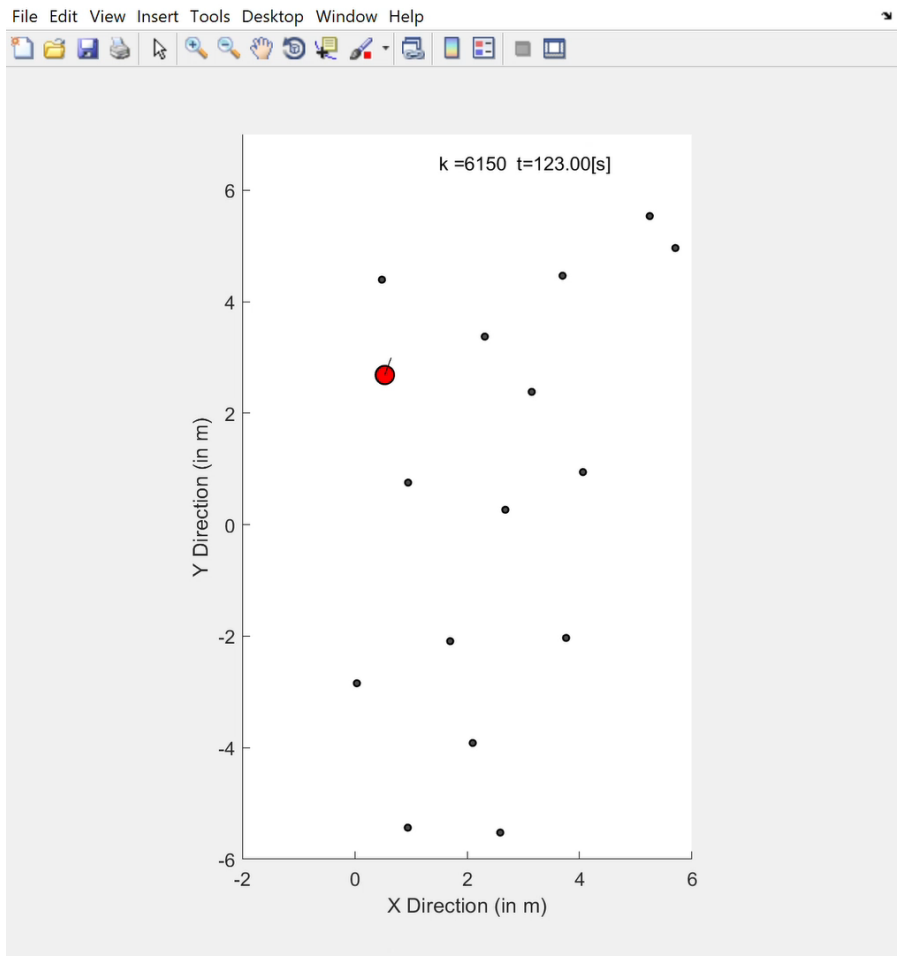
- Statistics on error between ground truth and estimated trajectories:

| Statistic | X Value (m) | Y Value (m) | Theta Value (rad) |
|---|---|---|---|
| Minimum | -3.8137 | -0.7301 | -6.2665 |
| Maximum | 1.0992 | 1.2084 | 6.1825 |
| Mean | -0.6431 | 0.3678 | 0.0083 |
| Standard Deviation | 1.1851 | 0.4531 | 0.8723 |

# PROJECT: RESULTS OF EKF SLAM ON EXTERNAL DATA

Estimated Trajectory

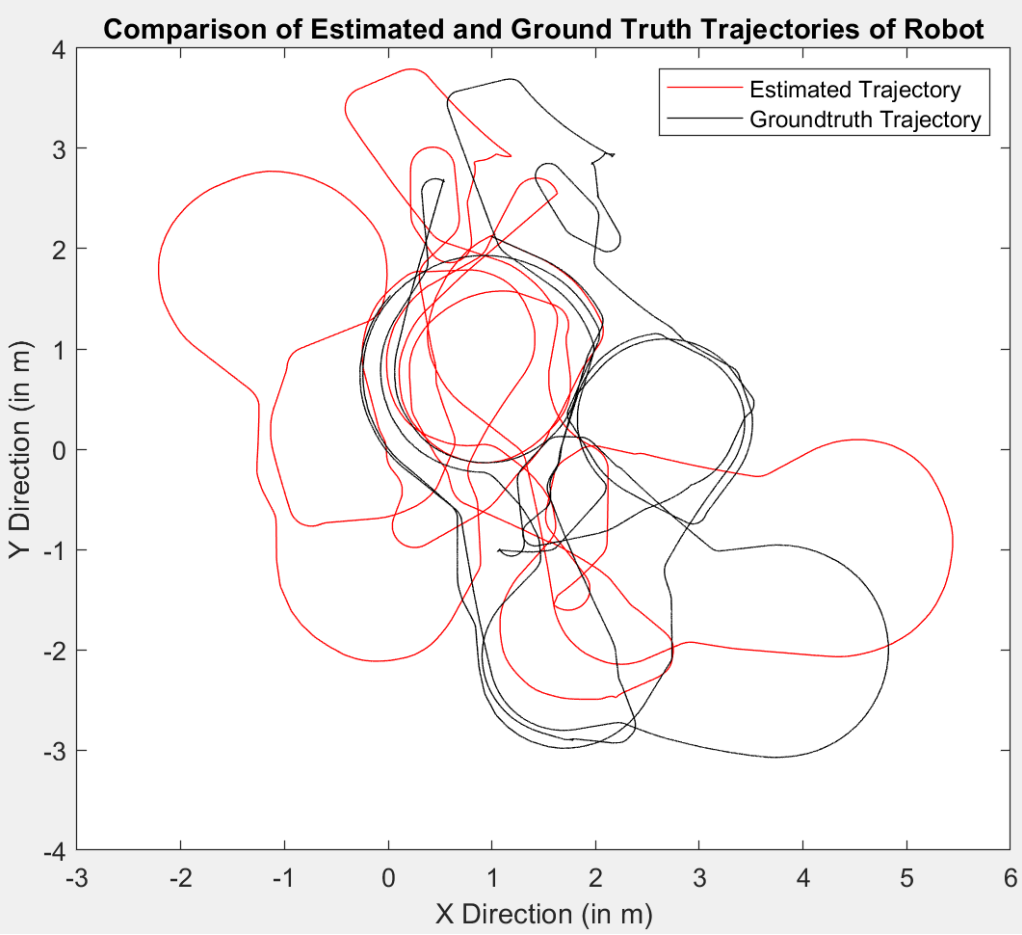



Ground Truth Trajectory

# PROJECT: RESULTS OF EKF SLAM ON EXTERNAL DATA

- Comparing ground truth to estimated still observe some noise
  - Ground truth trajectory shows patterns
  - Estimated still shows sporadic motion
- Can be indicative that more errors lie in code



# ADDITIONAL IMPLEMENTATIONS

# PROJECT: ADDITIONAL IMPLEMENTATIONS

- Attempted to implement EKF SLAM on data acquired through moving Kinect RGB-D sensor

- Two cases:
    1. Kinect moved in one direction, towards single landmark in horizontal plane
    2. Kinect moved in two directions in horizontal plane, locating artificial visual targets

- Unsuccessful implementation likely due to lack of correct odometry data and coordinate frames
- Software has possibility to function correctly with few modifications

# PROJECT: ADDITIONAL IMPLEMENTATIONS

- Kinect sensor provided RGB and Depth images of an environment
  - Goal was to identify position of target using RGB image and associated depth from camera using depth image

# PROJECT: ADDITIONAL IMPLEMENTATIONS

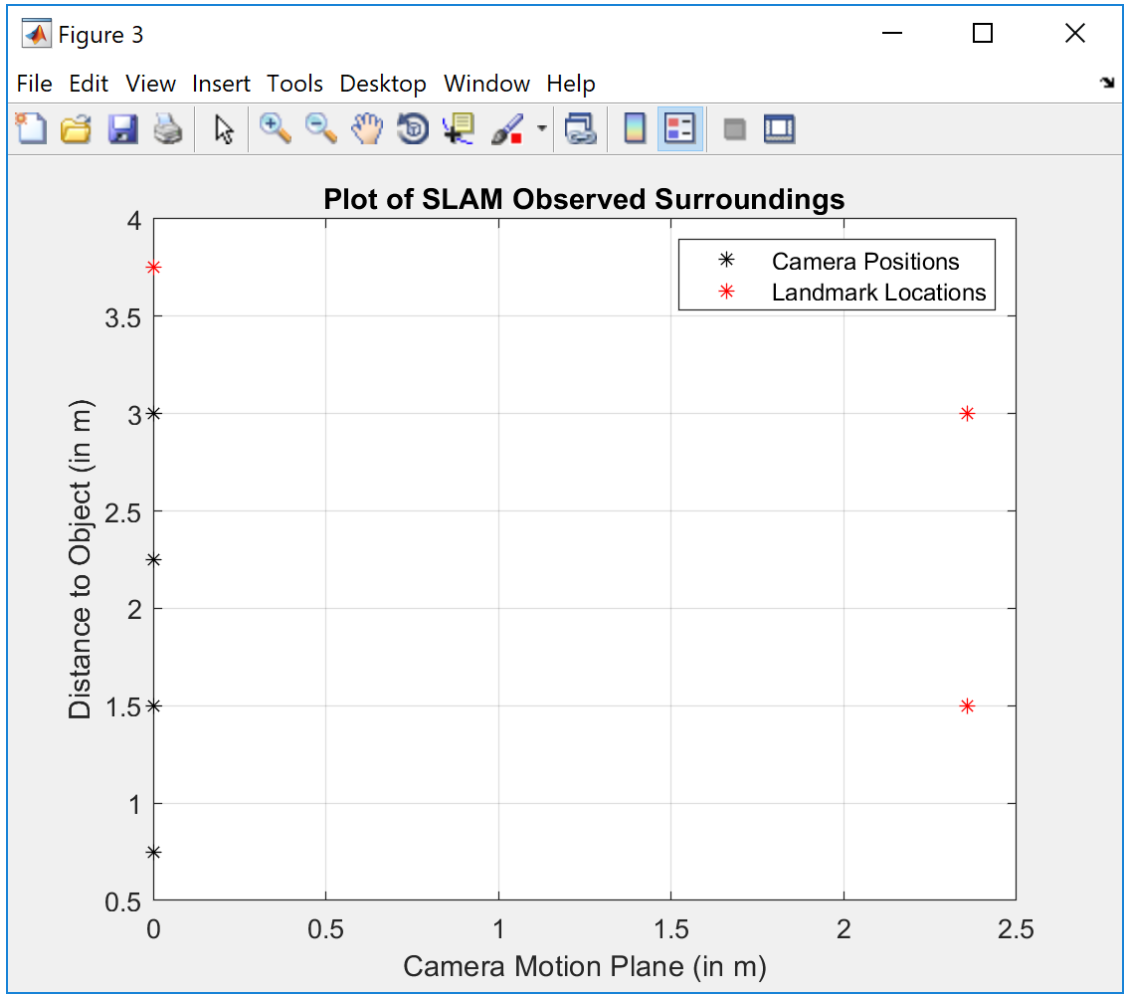
- First Case Steps:
  - Motion model/uncertainty determined by sampling my single step and finding its mean and deviation
  - Measurement (range) uncertainty determined by analyzing disparity between true depth to known point and measured
  - Kinect sensor moved on floor towards a wall, in a straight line by a step at a time
    - Assumed that the change in heading (angle) of the camera was 0
    - RGB and Depth images taken manually at each point
  - The principal point of the depth camera images was taken as landmark in each iteration

- Implementation:
  - Linear motion therefore applied Kalman Filter

# PROJECT: ADDITIONAL IMPLEMENTATIONS

- Results:

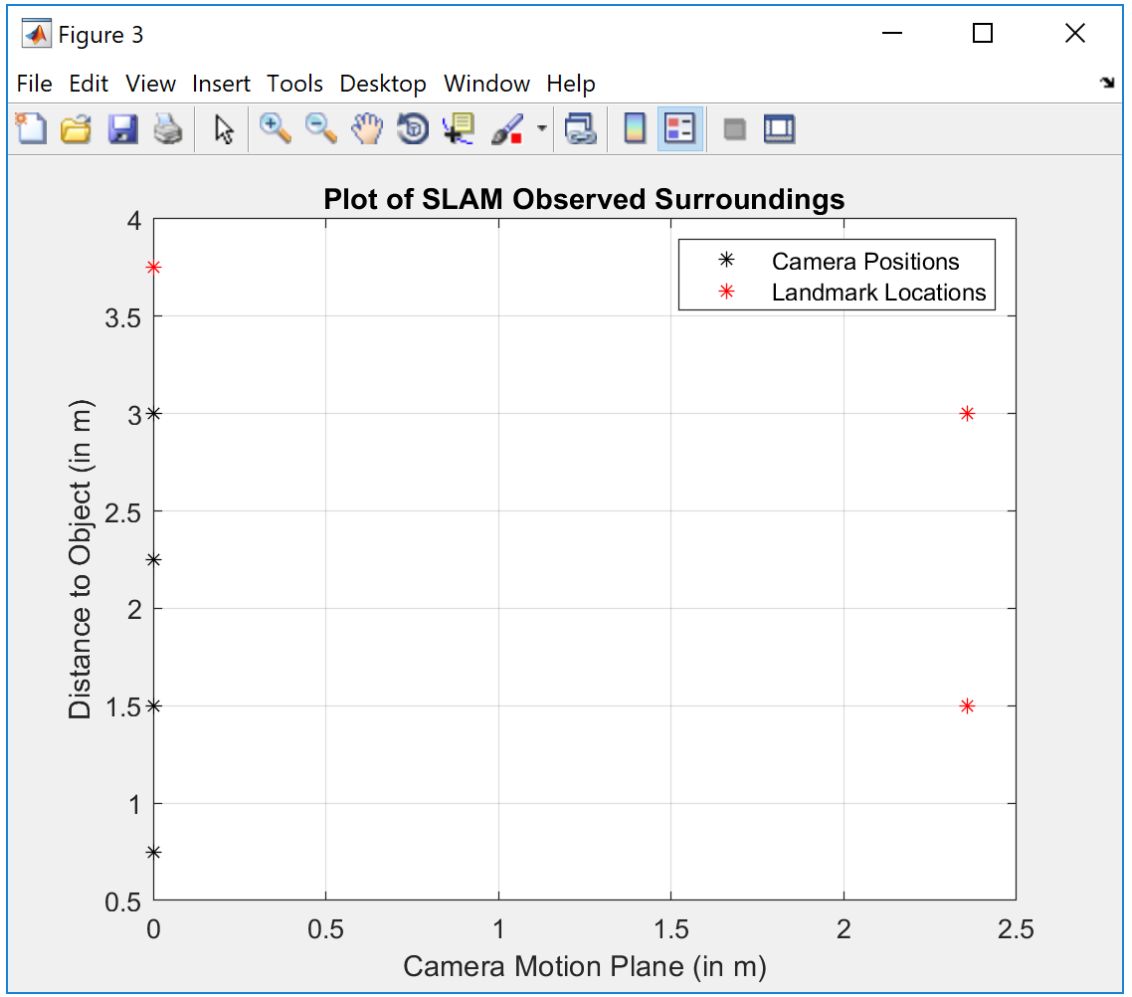
| X Position (m) | Y Position (m) |
|---|---|
| 0.75 | 0 |
| 1.499999 | 0 |
| 2.249999 | 0 |
| 2.999996 | 0 |
| 3.749996 | 0 |



# PROJECT: ADDITIONAL IMPLEMENTATIONS

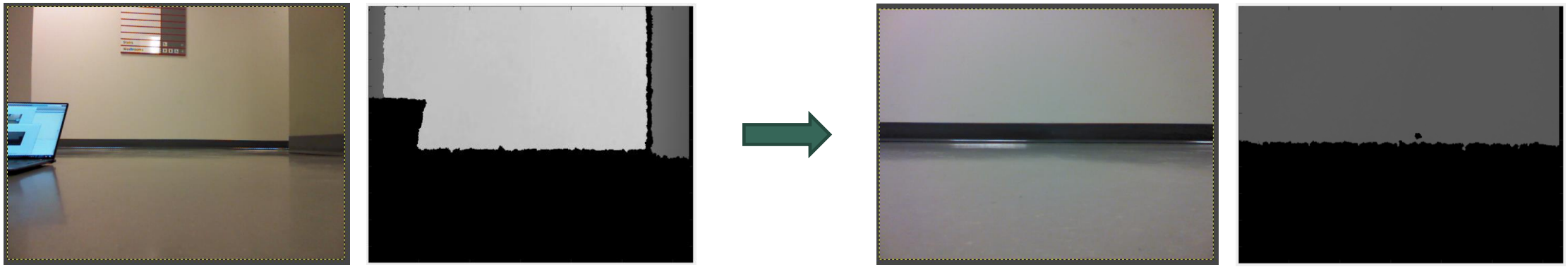
- Results:

| X Position (m) | Y Position (m) |
|---|---|
| 0.75 | 0 |
| 1.499999 | 0 |
| 2.249999 | 0 |
| 2.999996 | 0 |
| 3.749996 | 0 |



# PROJECT: ADDITIONAL IMPLEMENTATIONS

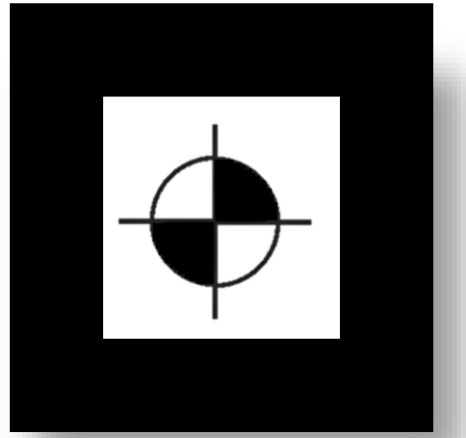
- Images likely source of error:

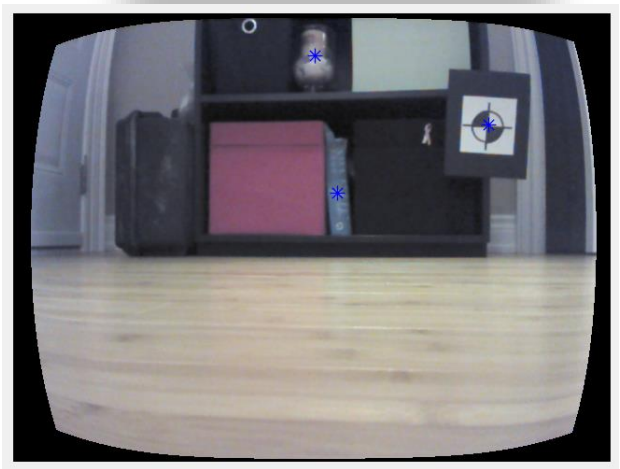


- Sensor unable to identify depth of features within approximately 1m and beyond 4m
  - Unable to provide depth to floor, which might be included in analysis of principal point region

# PROJECT: ADDITIONAL IMPLEMENTATIONS

- Seconds Case Steps:
  - Odometry determined by applying visual odometry methods
    - Involved determining relative camera locations and orientation from matching features in sequential images
  - Kinect sensor moved in environment containing identical targets which acted as landmarks
    - RGB and Depth images taken through manual trigger at each point
  - Measurements of range and bearing found through:
    - Centroid of targets identified in each RGB image
      - From this, angle of target to camera principal point was found (acted as bearing)
    - Depth to this target centroid was found (acted as range)





# PROJECT: ADDITIONAL IMPLEMENTATIONS

- Error arose from:
  - Differences in coordinate systems of camera (i.e. x,y - pixels) and real-world (i.e. depth - metres)
    - Because Kinect sensor RGB camera not truly a stereo system, could not acquire real-world coordinates of target centroids for map
    - Visual odometry also provided relative camera poses in previous camera coordinate frame (not real world)
    - This caused errors when using EKF SLAM algorithm, since it transformed matrices to the 2D real-world state space
  - Inability of visual odometry algorithm to determine relative pose without strong overlap between images
  - Poor target detection at oblique camera angles

# PROJECT: ADDITIONAL IMPLEMENTATIONS

- Additional comments:
  - How could odometry data help?
    - Would provide true real-world coordinate system motion
    - Reduces errors associated with the relative camera pose algorithm and the need for a scaling term
  - To account for differences in coordinate frames:
    - Could apply Structure From Motion (SFM) techniques and triangulate to find 3D coordinates of features[8]
      - Requires features to be observed 3 different frame (observed in 1st frame, re-observed and triangulated in 2nd frame, and used to calculate transformation in 3rd frame)

# FUTURE IMPROVEMENTS

# PROJECT: FUTURE IMPROVEMENTS TO ALGORITHM

- To improve processing capabilities of the software algorithm[2]:
  - Limit the number of landmarks considered in the data association step
  - Reduce probability of false data associations by making landmarks more unique and further apart
  - Remove/augment the system, accordingly, for landmarks that have not been observed recently

- To improve uncertainties in results of the software algorithm[2]:
  - Use a provisional landmark list to compensate for extreme outliers
    - Measurement errors can create fake landmarks and cause wrong corrections to be applied to the estimated pose
    - Can add landmark to map only if it's consistency observed and its uncertainty decreases

# GOOD SOURCES FOR FURTHER RESEARCH


1. Thrun, W. Burgard and D. Fox, Probabilistic Robotics, Draft ed., 2000.
2. C. Stachniss, "Robot Mapping: EKF SLAM," Albert-Ludwigs-Universitat Freiburg, Frieburg im Breisgau.
3. C. Stachniss EKF SLAM Video Series on Youtube
4. J. Sola, "Simultaneous Localization and Mapping with the Extended Kalman Filter," 2014.

# REFERENCES


[1] E. Olson, "L11. EKF SLAM: Part 1," University of Michigan, Ann Arbo.

[2] Thrun, W. Burgard and D. Fox, Probabilistic Robotics, Draft ed., 2000.

[3] C. Stachniss, "Robot Mapping: EKF SLAM," Albert-Ludwigs-Universitat Freiburg, Frieburg im Breisgau.

[4] J. Sola, "Simultaneous Localization and Mapping with the Extended Kalman Filter," 2014.

[5] Merriam-Webster Dictionary, "Maximum Likelihood," 2019. [Online]. Available: https://www.merriam-webster.com/dictionary/maximum%20likelihood. [Accessed 12 01 2019].

[6] K. Y. K. Leung, Y. Halpern, T. D. Barfoot and H. H. T. Liu, "The UTIAS Multi-Robot Cooperative Localization and Mapping Dataset," *International Journal of Robotics Research,* vol. 30, no. 8, pp. 969-974, July 2011.

[7] J. Callaham, "Microsoft to End Sales of Original Kinect for Windows Sensor in 2015," 30 12 2014. [Online]. Available: https://www.windowscentral.com/microsoft-end-sales-original-kinect-windows-sensor-2015. [Accessed 13 01 2019].

[8] K. Yousif, A. Bab-Hadiashar and R. Hoseinnezhad, "An Overview to Visual Odometry and Visual SLAM: Applications to Mobile Robotics," Melbourne, 2015.