System Model

- Model Including Noise:

$$\begin{cases} x(t+1) = x(t) + T_s \hat{u_f}(t) cos\theta(t) - T_s w_f(t) cos\theta(t) \\ y(t+1) = y(t) + T_s \hat{u_f}(t) sin\theta(t) - T_s w_f(t) sin\theta(t) \\ \theta(t+1) = \theta(t) + T_s \hat{u_a}(t) - T_s w_a(t) \end{cases} \tag{1}$$

- Measurements collected by the Lidar:

$$m_{\rho_j}(t) = \sqrt{(L_{j,x} - x(t))^2 + (L_{j,y} - y(t))^2} + v_{\rho_j}(t) \tag{2}$$

$$m_{\alpha_j}(t) = atan2\{L_{j,y} - y(t), L_{j,x} - x(t)\} - \theta(t) + v_{\alpha_j}(t) \tag{3}$$

State Definition

In the state we consider the robot pose and the coordinates x and y of Landmark, $n = 3 + 2Nland$.

$$z(t) = \begin{bmatrix} x(t) & y(t) & \theta(t) & L_{1,x} & L_{1,y} & ... & ... & L_{N_l,x} & L_{N_l,y} \end{bmatrix}^T \in \mathbb{R}^n \tag{4}$$

The coordinates of Landmarks are constant state, so $L_{i,x}(t+1) = L_{i,x}(t)$ and $L_{i,y}(t+1) = L_{i,y}(t)$.

Jacobian Matrices

We consider:

$$z(t+1) = p(z(t), \hat{u}(t), w(t)) : \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^d \to \mathbb{R}^n \tag{5}$$

$$y(t) = h(z(t)) + v(t) : \mathbb{R}^n \to \mathbb{R}^p \tag{6}$$

$$Z(t) = \frac{\partial p}{\partial z}\bigg|_{z=\hat{z}(t|t), u=\hat{u}(t), w=0} = \begin{pmatrix} 1 & 0 & T_s(w_f sin\theta(t) - \hat{u_f} sin\theta(t)) & \\ 0 & 1 & T_s(\hat{u_f} cos\theta(t) - w_f cos\theta(t)) & \mathbf{0}_{3\times 40} \\ 0 & 0 & 1 & \\ & & \mathbf{0}_{40\times 3} & I_{40\times 40} \end{pmatrix}\Bigg|_{z=\hat{z}(t|t), u=\hat{u}(t), w=0} \in \mathbb{R}^{n\times n} \tag{7}$$

$$G(t) = \frac{\partial p}{\partial w}\bigg|_{z=\hat{z}(t|t)} = \begin{bmatrix} -T_s cos\theta(t) & 0 \\ -T_s sin\theta(t) & 0 \\ 0 & -T_s \\ .. & .. \\ 0 & 0 \end{bmatrix}\Bigg|_{z=\hat{z}(t|t)} \in \mathbb{R}^{n\times d} \tag{8}$$

$$\begin{cases} dx = (L_{j,x} - x(t)) \\ dy = (L_{j,y} - y(t)) \end{cases} \tag{9}$$

$$H(t) = \frac{\partial h}{\partial z}\bigg|_{z=\hat{z}(t|t-1)} = \begin{bmatrix} \frac{-dx}{\sqrt{dx^2+dy^2}} & \frac{-dy}{\sqrt{dx^2+dy^2}} & 0 & 0 & .. & \frac{dx}{\sqrt{dx^2+dy^2}} & \frac{dy}{\sqrt{dx^2+dy^2}} & 0 & .. & 0 \\ \frac{dy}{dx^2+dy^2} & \frac{-dx}{dx^2+dy^2} & -1 & 0 & .. & \frac{-dy}{dx^2+dy^2} & \frac{dx}{dx^2+dy^2} & 0 & .. & 0 \end{bmatrix}\bigg|_{z=\hat{z}(t|t-1)} \in \mathbb{R}^{2\times n} \tag{10}$$

Initialization

$$z(0|-1) = \begin{bmatrix} z^u(0|-1) \\ z^l(0|-1) \end{bmatrix} = \begin{bmatrix} Pose(:,1) \\ \mathbf{0}_{2Nland} \end{bmatrix}$$

$$P(0|-1) = \begin{bmatrix} P^u(0|-1) & 0 \\ 0 & P^l(0|-1) \end{bmatrix}$$

$$P^u(0|-1) = \begin{bmatrix} 0.0001 & 0 & 0 \\ 0 & 0.0001 & 0 \\ 0 & 0 & 0.0001 \end{bmatrix}, \quad P^l(0|-1) = 100 \cdot I_{2Nland\times 2Nland}$$

The uncertainty on the robot's pose has been chosen very small since we have set the robot initial position as the real one based on the ground truth, so it has to trust a lot on it. The location of the landmarks has been set in the origin, choosing a very large uncertainty and observing that the maximum acquired distance is around 25 meters, we have decided to set the 3σ intervals around 30 meters. Unfortunately in this way each landmark is associated to a completely wrong position, for this reason, in case a landmark is seen for the first time, it is necessary to update the state with its first measured position.

Confidence intervals and ellipses

To verify the consistency of the estimates of the robot pose over time we used the 3σ confidence intervals while for the landmark positions the confidence ellipses, through the function below. As a spread indicator, the function use the eigenvalues of the covariance matrix relative to the $i-th$ landmark. The eigenvalues represent the spread in the direction of the eigenvectors, which are the variances under a rotated coordinate system. The actual radius of the ellipse are $\sqrt{\lambda_1}$ and $\sqrt{\lambda_2}$ for the two eigenvalues λ_1 and λ_2 of the scaled covariance matrix $s \cdot \Sigma$. The parameter s is the scaling factor of ellipse that we have to choose to met the desired confidence p [1]. In this case we have chosen $p = 0.99$.

[1] https://raw.org/math/how-to-plot-a-covariance-error-ellipse/

```
function DrawEllipse(Zest,Ptotal,p,istant,Landmarks)
        mu=[Zest(istant,2+2*q);Zest(istant,3+2*q)];
        Pel=reshape(Ptotal(end,:),[],size(Zest,2))';
        Sigma= Pel(2+2*q:3+2*q,2+2*q:3+2*q);
        s = -2 * log(1 - p);
        [V, D] = eig(Sigma * s);
        t = linspace(0, 2 * pi);
        a = (V * sqrt(D)) * [cos(t(:))'; sin(t(:))'];
        plot(a(1, :) + mu(1), a(2, :) + mu(2));
end
```

Results: SLAM-A

In Figure 1 e 3, we can observe the simulations respectively obtained with data set 1 and 2.
We have noticed during the two different simulations that in the first case the robot acquires more than one landmark at time and in the other case only one at each time, due to the lower radius of its laser, as consequence we have that the results obtained with the first dataset are more accurate than the second, due to the greater information contained.
In both of them are shown, on the left, the estimated trajectory's components, which are consistent with respect to the real ones. In the right side is represented the estimated landmarks' positions and the robot trajectory and orientation, in comparison with true data, and also in this case the estimate is sufficiently satisfying.
In Figure 2 e 4 are shown errors' plots, where we can observe that the estimation errors lie inside the confidence intervals, the same happens for the estimated landmarks' positions, that are inside the confidence ellipse. We can notice that the uncertainty over x component is small while over y component is bigger, along the first side. After the first corner, the first uncertainty increases and the second starts to decrease. The reason is the movement of robot along each side, because in the first side, it moves only along x axis and it is safer on the latter and less on y component while in the second side we have the opposite situation. The same considerations can be made about the variation in the dimensions of the confidence ellipses along the path.
Similar behaviors are observable along others parts until we don't achieve the last side where the robot see the first landmark of experiment. In this situation the uncertainties over components decrease rapidly thanks to the phenomena of "loop closure", associating the first landmarks, a correction takes place on everything else, improving the estimate and uncertainty, this situation is especially observable using the second data set.
Another thing that we can observe, looking the orange line in the figure, which represents the estimated trajectory and orientation obtained by odometry, is that the robot gets lost more when he turns respect when it goes straight this because of the grater uncertainty when it turns.

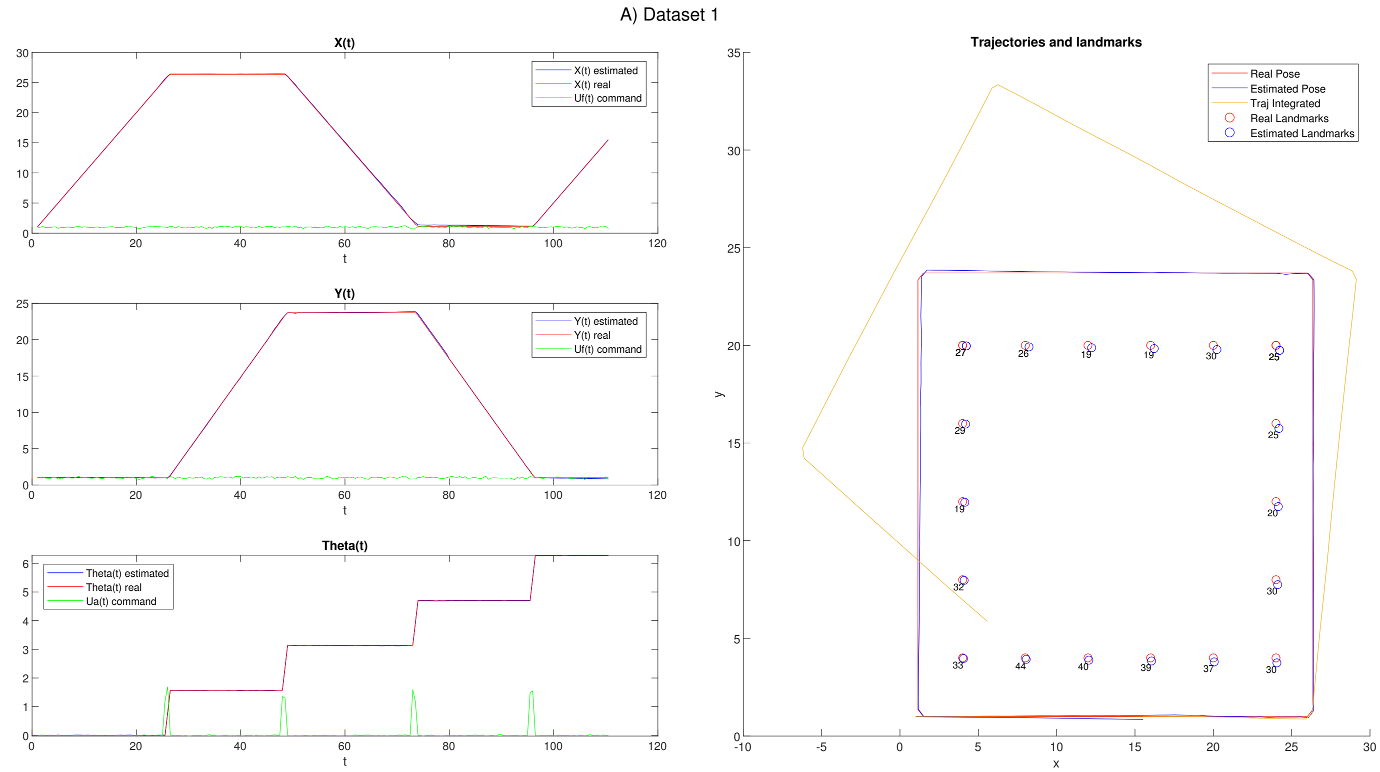

Figure 1: Trajectory and orientation [SLAM A : Data set 1]

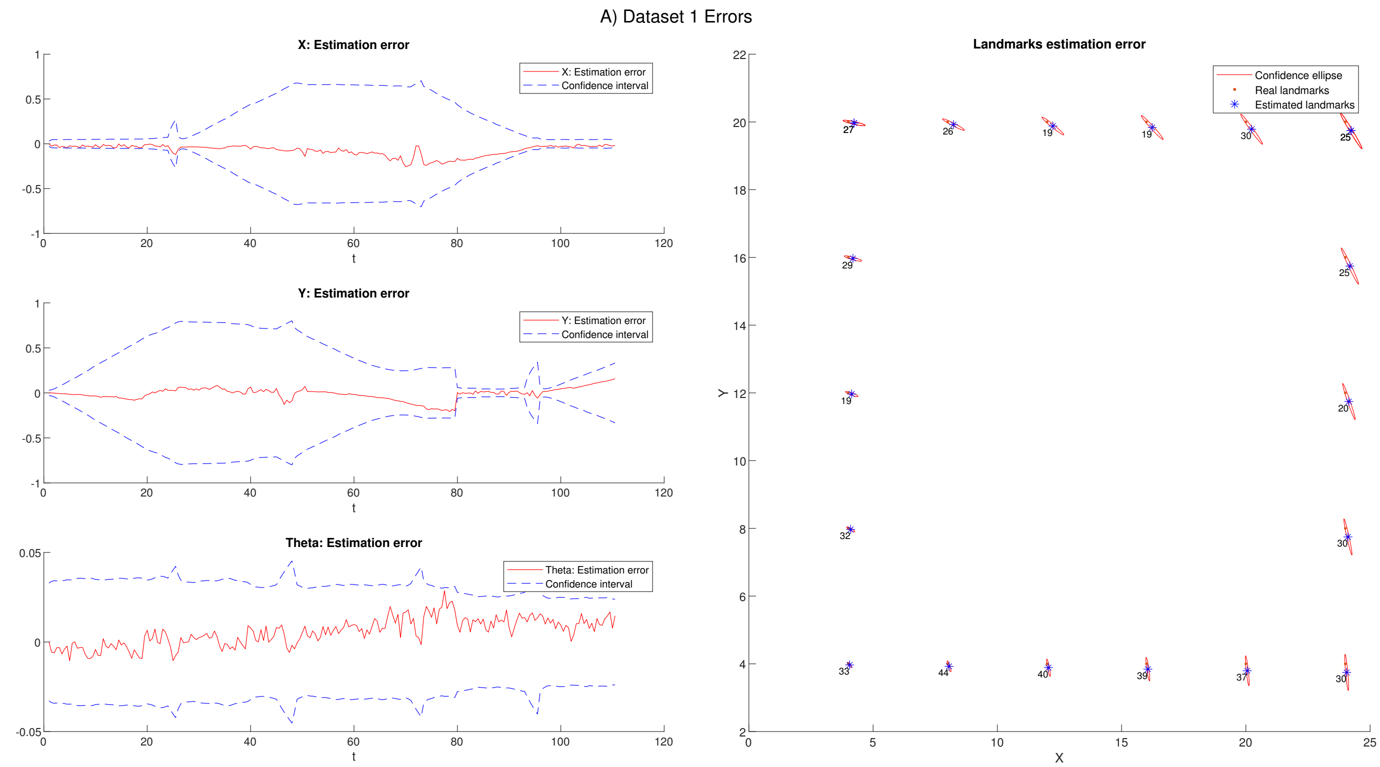

Figure 2: Confidence intervals and ellipses [SLAM A : Data set 1]

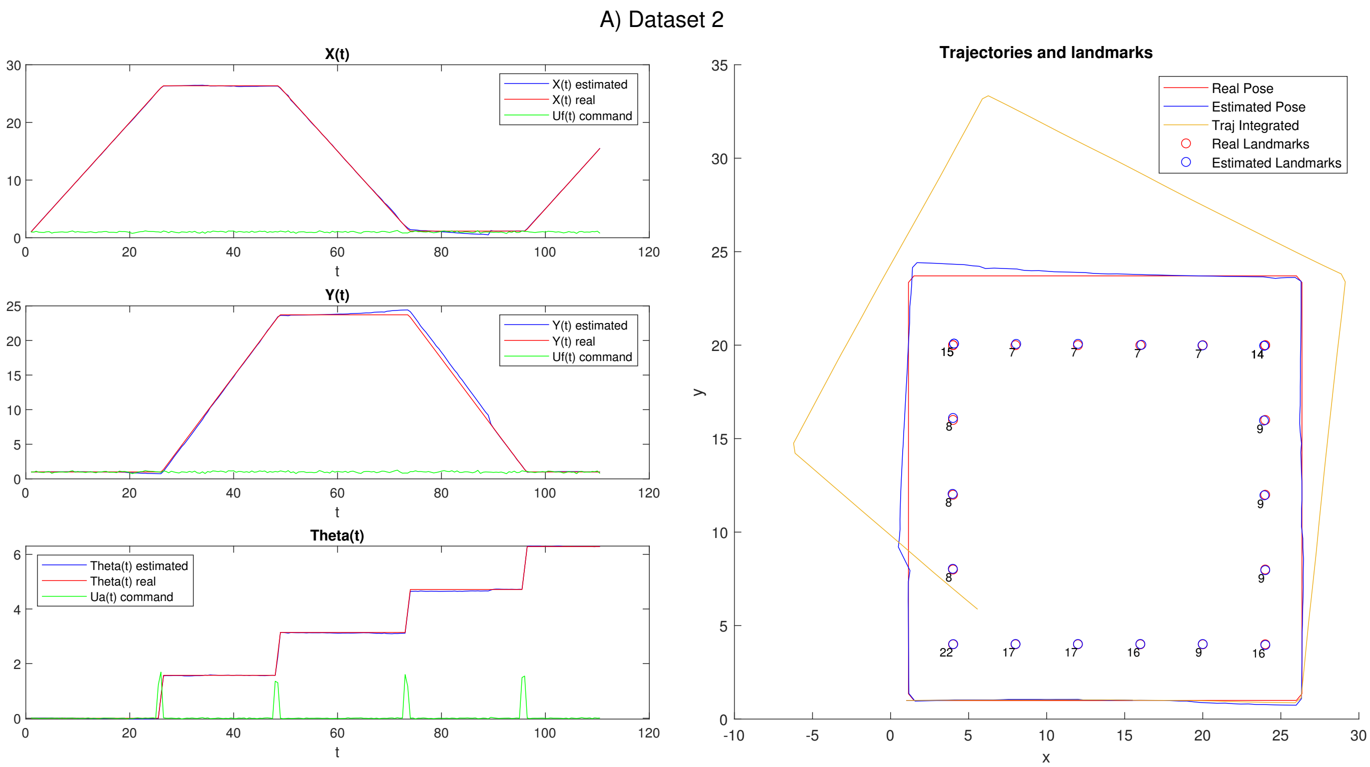

Figure 3: Trajectory and orientation [SLAM A : Data set 2]

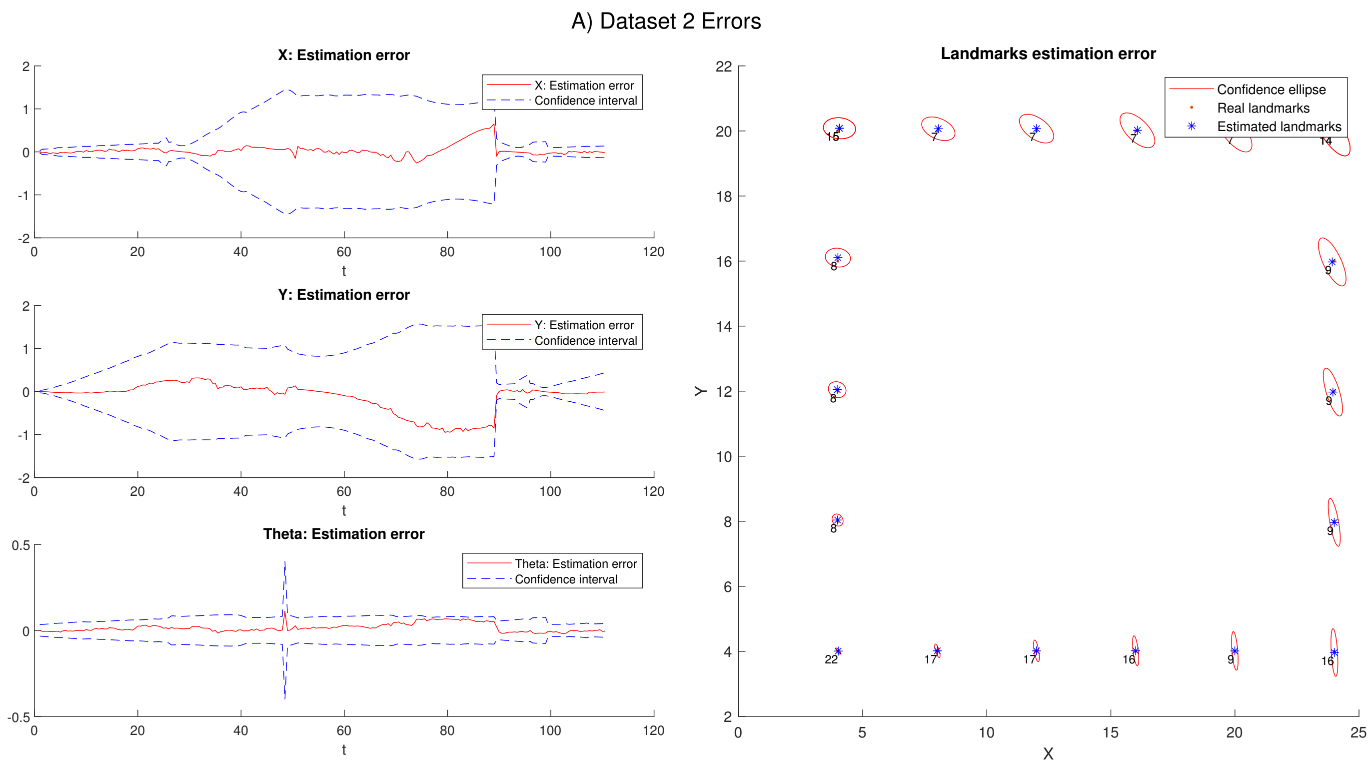

Figure 4: Confidence intervals and ellipses [SLAM A : Data set 2]

State Definition

The first big difference with respect to the previous case is that now the state is initialized only as a 3x1 vector with elements *x(t)*, *y(t)*, $\theta(t)$ and it grows in dimension at each iteration when it finds new landmarks by adding a 2x1 vector which contains their coordinates positions.
At the beginning we have: $z(t) = [x(t)\ y(t)\ \theta(t)]^T \in \mathbb{R}^3$ while at the end of the simulation, the state become:

$$z(t) = \begin{bmatrix} x(t) & y(t) & \theta(t) & L_{1,x} & L_{1,y} & \dots & \dots & L_{N_l,x} & L_{N_l,y} \end{bmatrix}^T \in \mathbb{R}^n \tag{11}$$

where $n = 3 + 2N$ and N represents the number of landmarks founded, placed according to the order in which the robot see them and not following the labels numbers as in SLAM A. Same happens for uncertainty of state. Initially it is a 3x3 matrix and grows by adding 2x2 matrices along the diagonal gradually. Same initial condition and uncertainty for robot's position are set as in challenge A, while for the landmarks' uncertainties, it has been chosen smaller ($\eta = 10$) with respect to the previous case since the first time each landmark is added to the state with its first measurement.

Mahalanobis Distance

In this case the landmarks aren't labelled at the beginning so when the robot sees one of them, it has to classify the latter to decide whether it is a new landmark or one that has previously seen. In the first case, the state has to been enlarge while in the second case only the correction step is applied to the previous state vector (in SLAM-A we have recognized directly a landmark through its label). The discriminant function used in this phase is the Mahalanobis Distance. In this case the two threshold are $\tau_1 = 5.9915$ and $\tau_2 = 13.8155$ that correspond respectively to the 95% and the 99,9% confidence levels of the χ^2 distribution with 2 degrees of freedom.

Results: SLAM-B

The Figure 5 and Figure 7, are referred respectively to Data set 1 and Data set 2, it is reported the behaviour of the estimated pose, the estimated robot's trajectory and the estimated landmarks positions, all compared with the real ones, and as in challenge A they are consistent and similar considerations can be done.
At first glance, in Figure 6 and in Figure 8, we can notice that using data set 1, we obtained smaller confidence ellipses and so more precision for the landmarks' positions. What we can say is that both the pose estimation errors and the estimated landmarks positions lies inside the 3σ confidence intervals and the 3σ confidence intervals respectively.
The correctness of the classification in this challenge is demonstrated from the fact that at the end of the simulation we have obtained 18 landmarks instead of 20 landmarks of challenge A. This happens because in

the previous case two equal landmarks were labelled two times with different numbers (landmark in position (24,20) and (4,20)) and so they were interpreted as different landmarks. In challenge B, since they aren't labelled, they are associated as the same, correctly.

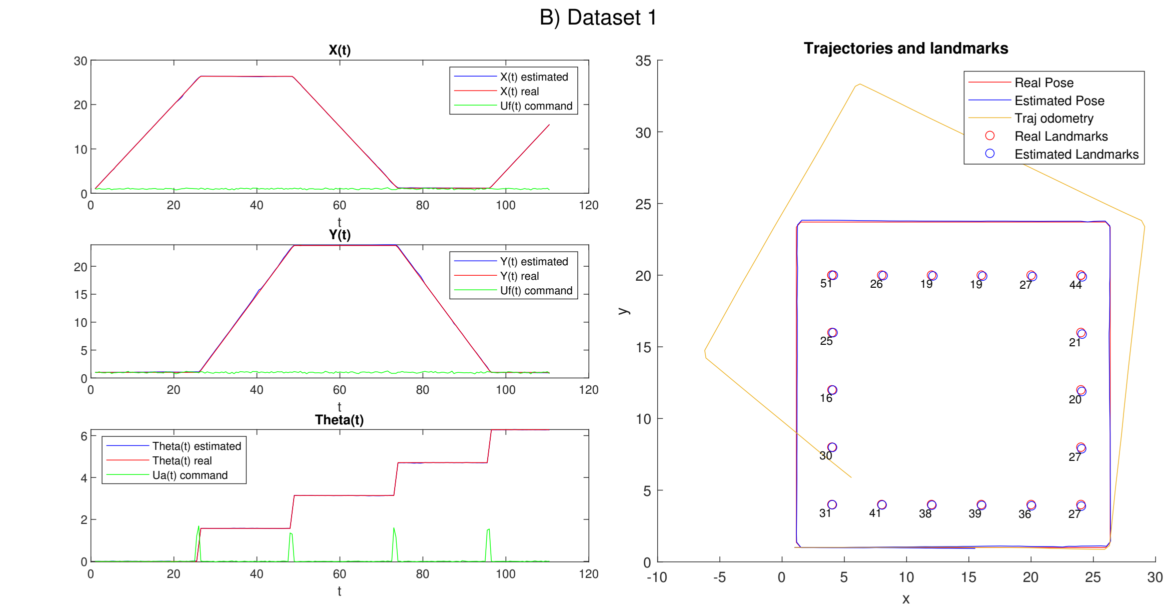

Figure 5: Trajectory and orientation [SLAM B : Data set 1]

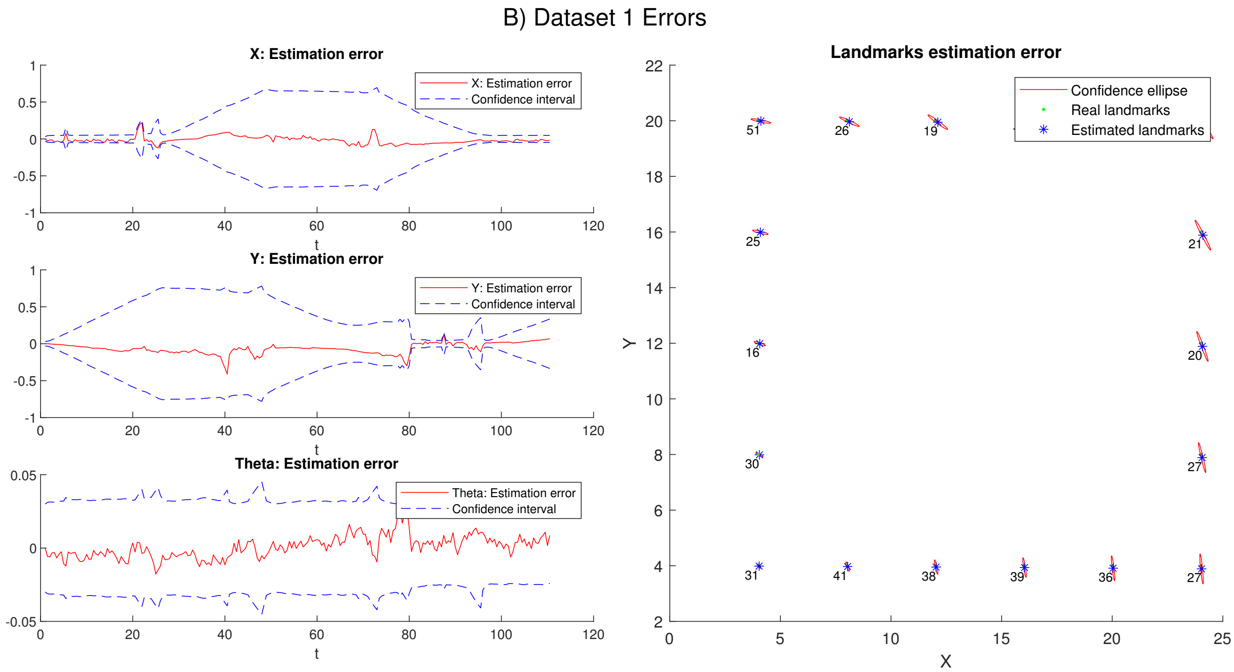

Figure 6: Confidence intervals and ellipses [SLAM B : Data set 1]

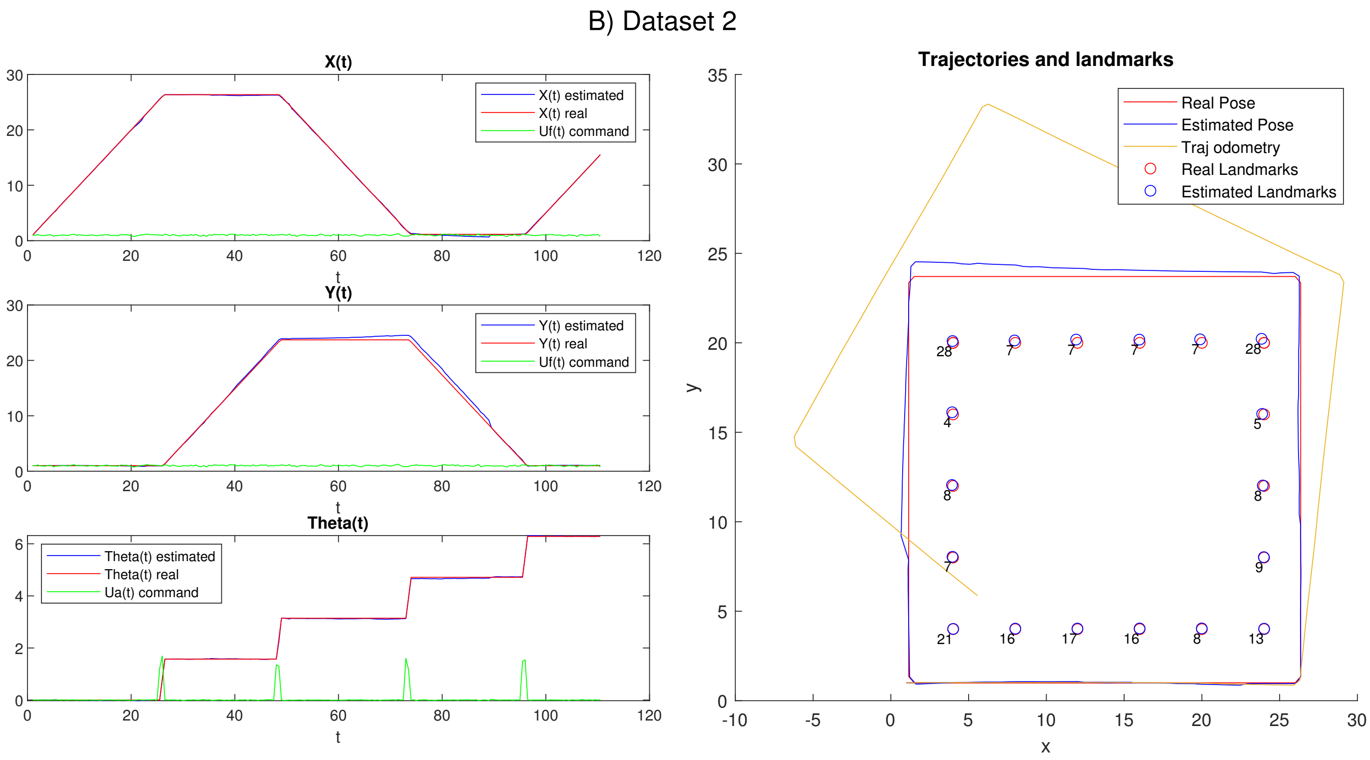

Figure 7: Trajectory and orientation [SLAM B : Data set 2]

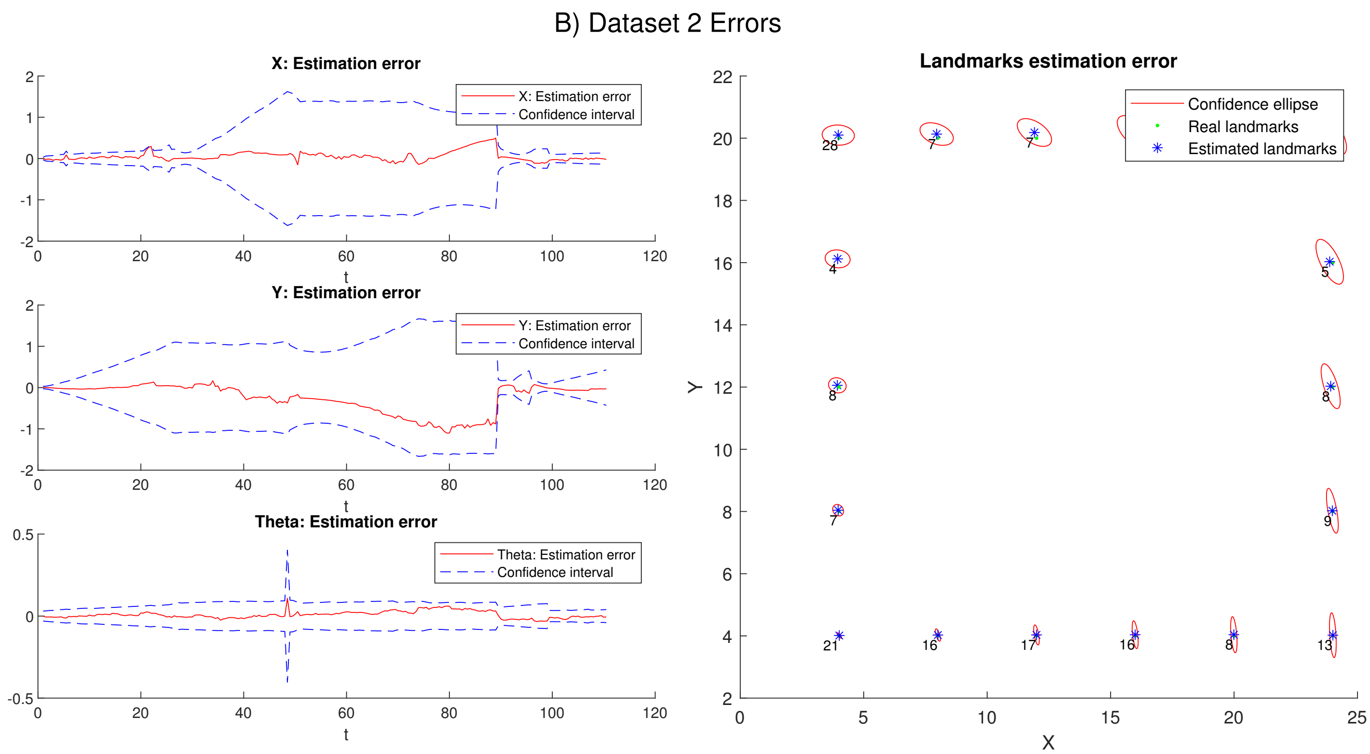

Figure 8: Confidence intervals and ellipses [SLAM B : Data set 2]

Corner Extraction function

The idea behind the corner extraction function is verify if a point is differentiable. Basically a corner is not differentiable while a point that belongs to a wall could have this property.

To detect the possible points that could be landmarks, first of all, we have extracted the local minima in the range of measured distances, using the Matlab function: `islocalmin('data', 'MinProminence', p)` where `p` is a new parameter that must be set. With the prominence we can decide how many minimums we can consider, selecting how deep these points should be, compared to those next.

Once we have got all the possible candidate at time instant t, first we have discarded the ones that are too far from the robot current position. Then we have computed for each of them the discriminant functions where the derivative is weighted by angles and distances of the point respect the current position. The idea is to trust more on points that are close and at an angle not extreme compared to the vision of the robot. At the end the tester is compared with a chosen threshold (new parameter) to separate the classes of points that belong to the wall and corners.

Parameters

In this challenge, we have to set the following parameters: τ_1 and τ_2 for the Mahalanobis distance, the classifier threshold, the prominence and the initial conditions of matrix P.

The threshold and prominence parameters has been chosen empirically, testing different values on each data set. The uncertainty for the robot's pose has been chosen a little bit higher respect the previous case (empirically), since in this way, we have obtained better results. Same initial conditions and uncertainties have been set as in the challenge B, with some tuning in this case. Also the thresholds for the classification have been obtained with the same method, where it has required a greater confidence to classify a new corner, and less confidence to associate the new to the old, this because the algorithm has difficulty to associate very close corners and this would require a more robust association through a physical distance constraint. The parameters have been fixed as follows:

Data Set 1	
λ	[0.002 0.002 0.00012]
η	15
τ_1	5.9 (94,77%)
τ_2	21.507 (100%)
Threshold	0.21
Prominence	0.111

Data Set 2	
λ	[0.002 0.002 0.00012]
η	15
τ_1	3.85 (85,51%)
τ_2	19.2 (99.99%)
Threshold	0.19
Prominence	0.113

Results: SLAM-C

The Figure 10 and Figure 12, are referred respectively to Data set 1 and Data set 2, it is reported the behaviour of the estimated pose, the estimated robot's trajectory and the estimated landmarks positions, all compared with the real ones, and as in the previous challenges they are consistent and similar considerations can be done. The loop closures are obtained in the fourth corridor and we can see its effect in the error plot.

At first glance, in Figure 11 and in Figure 13, we can notice that using data set 1, we obtained smaller confidence ellipses and so more precision for the landmarks' positions, since the first has less noise. What we can say is that both the pose estimation errors and the estimated landmarks positions lies inside the 3σ confidence intervals and the 3σ confidence intervals respectively.
In Figure 9, is shown the comparison between maps estimated and that obtained by odometry, as we can notice the first is similar to the true map, showing the effectiveness of the filter, a greater correction could be noticed on the date set 2.

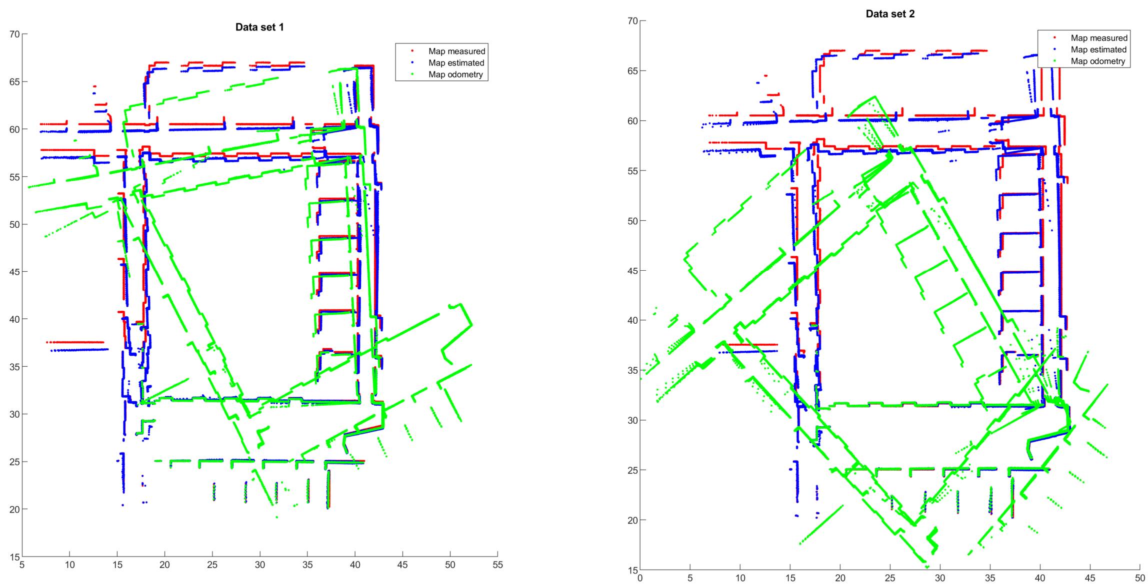

Figure 9: Map's comparison [SLAM C : Data set 1, Data set 2]

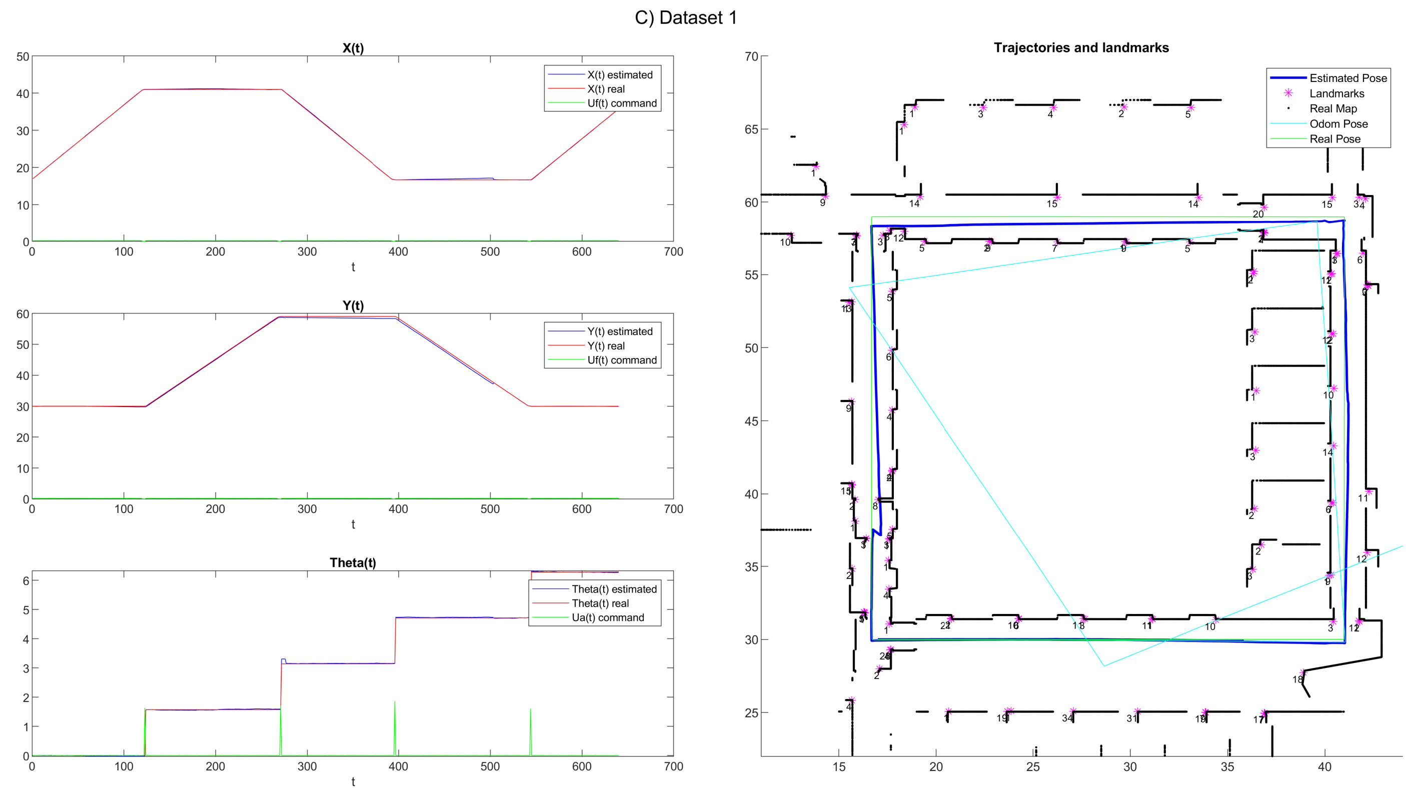

Figure 10: Trajectory and orientation [SLAM C : Data set 1]

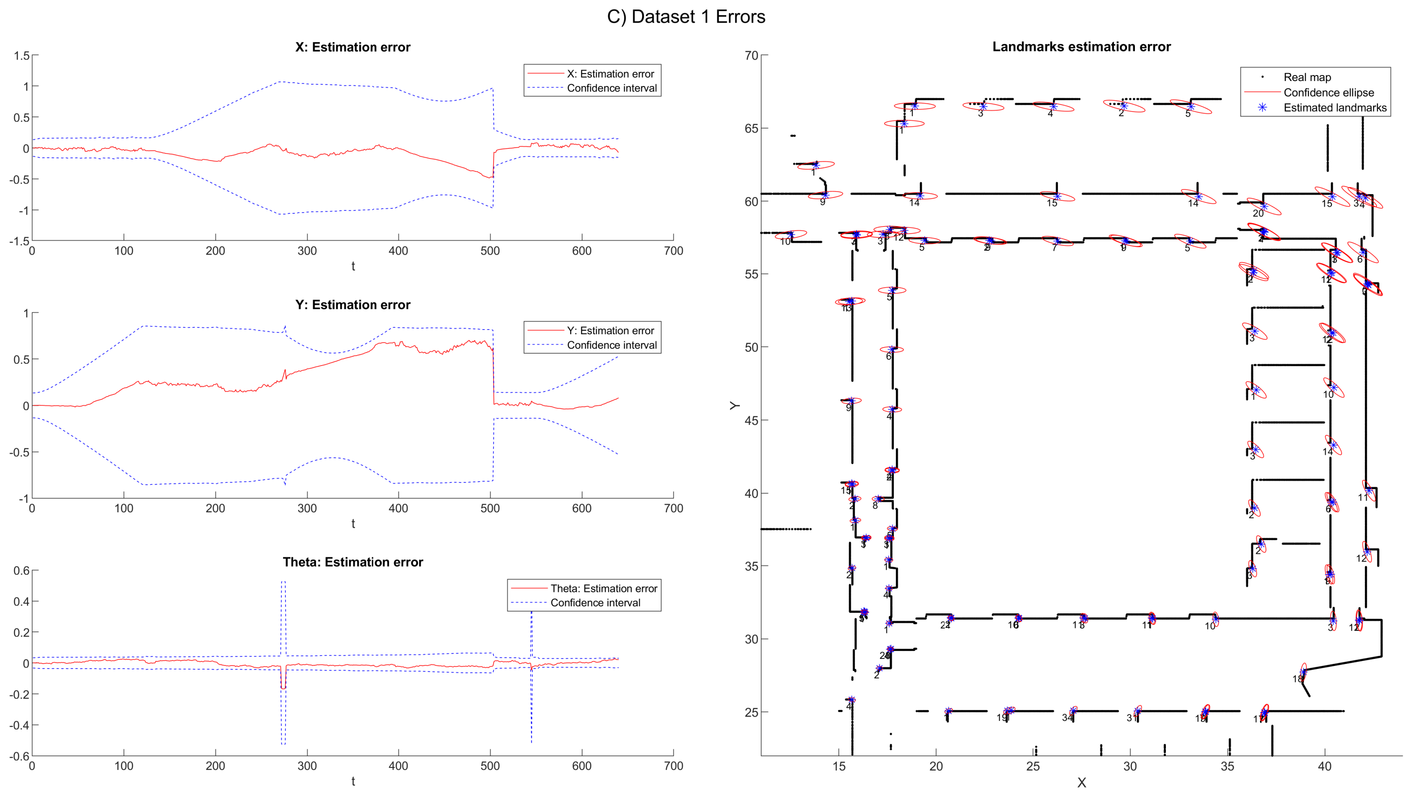

Figure 11: Confidence intervals and ellipses [SLAM C : Data set 1]

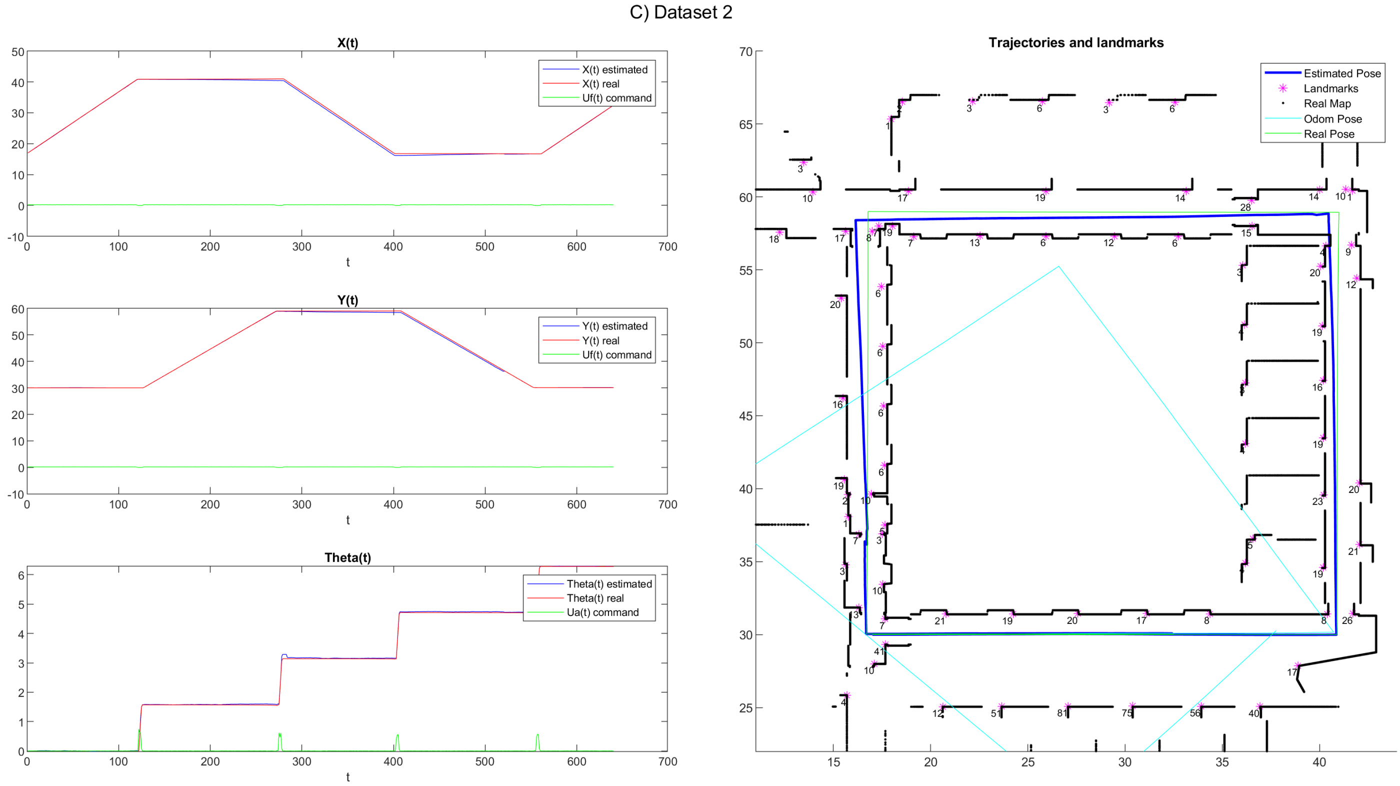

Figure 12: Trajectory and orientation [SLAM C : Data set 2]

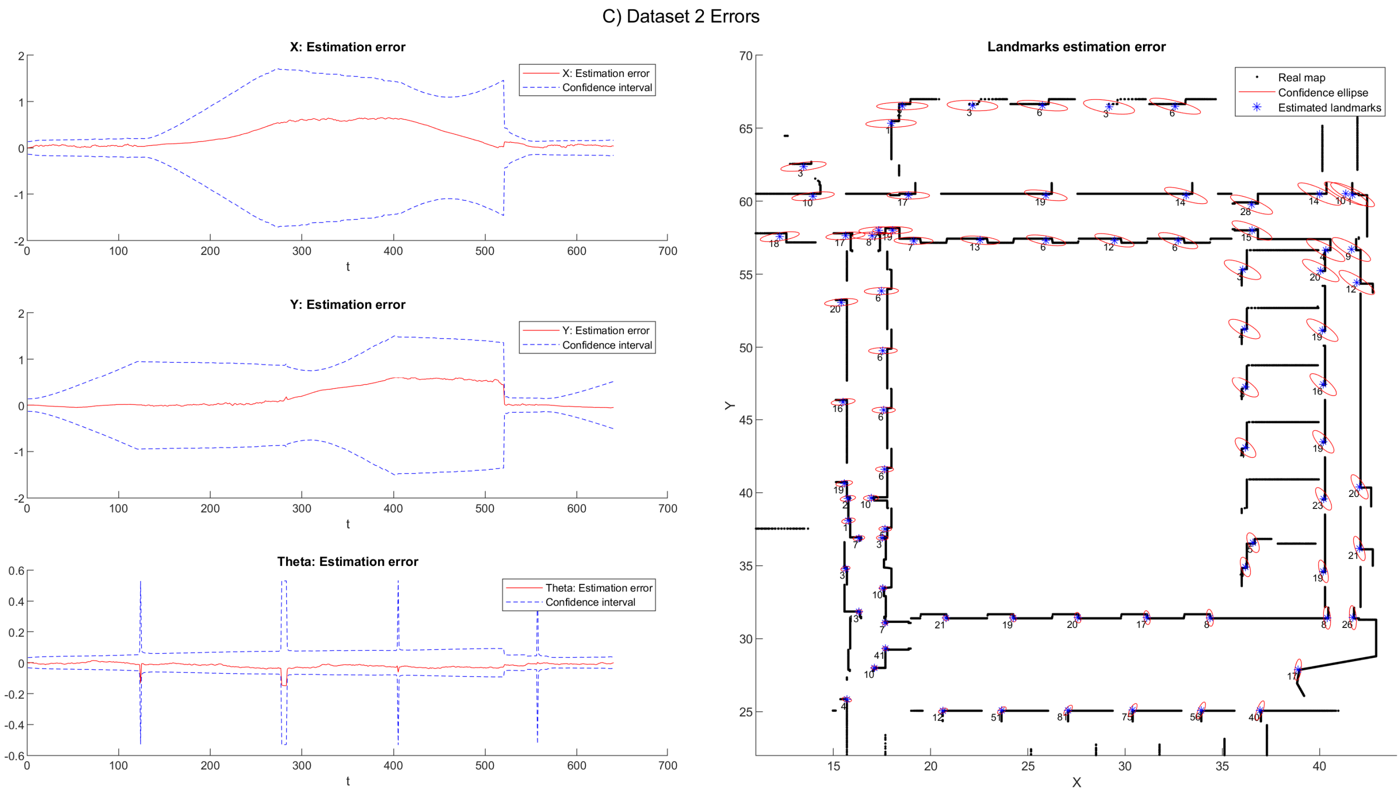

Figure 13: Confidence intervals and ellipses [SLAM C : Data set 2]

In this final challenge we didn't use simulated data as in the previous ones, but took data from a real-world experiment. They have been collected from tests performed with a Pioneer LX robot within the second floor of San Niccoló building. In particular, the challenge is divided into two exercises, but for each one we repeated the experiment twice; the first time we obtained a less accurate odometry, while in the second one we got very precise measurements since we tried to keep constant the linear velocity during linear along straight paths and turning the robot maintaining null the linear velocity. In both exercises the parameters to set are more than the previous case because we don't have the matrices Q, Q_{turn} and R, and the scalar w_{turn}. To choose them we based on the precision of the linear velocity U_f and the angular velocity U_a and from the lidar measurements. For the uncertainty on the measurements noise, we have chosen for the distance $3\sigma = 10\ cm$ and for the angle $3\sigma = 0.5\ grades$, indicative values that were then empirically tuned. The parameter η has been chosen greater in the first acquirement with respect to the second one due to the major uncertainty. In order to make our algorithm more robust for the classification we added an extra condition based on the physic distance set to 25cm, beyond the one yet present for the Mahalanobis distance.

Exercise 1 - 1 single side

The first exercise is based on walking along a single linear corridor and coming back to the initial pose. The parameters chosen for this exercise are in Table 1. As we can see in the left side of Figure 14 the linear velocity is oscillating during the first part and also while the robot is turning and it's quite constant when we come back. We took into account this big error interval when we have chosen the parameters, and this inaccuracy affects also the final trajectory estimate. With the second experiment instead, we obtained more precises velocities as it can be seen in the right side of Figure 14.

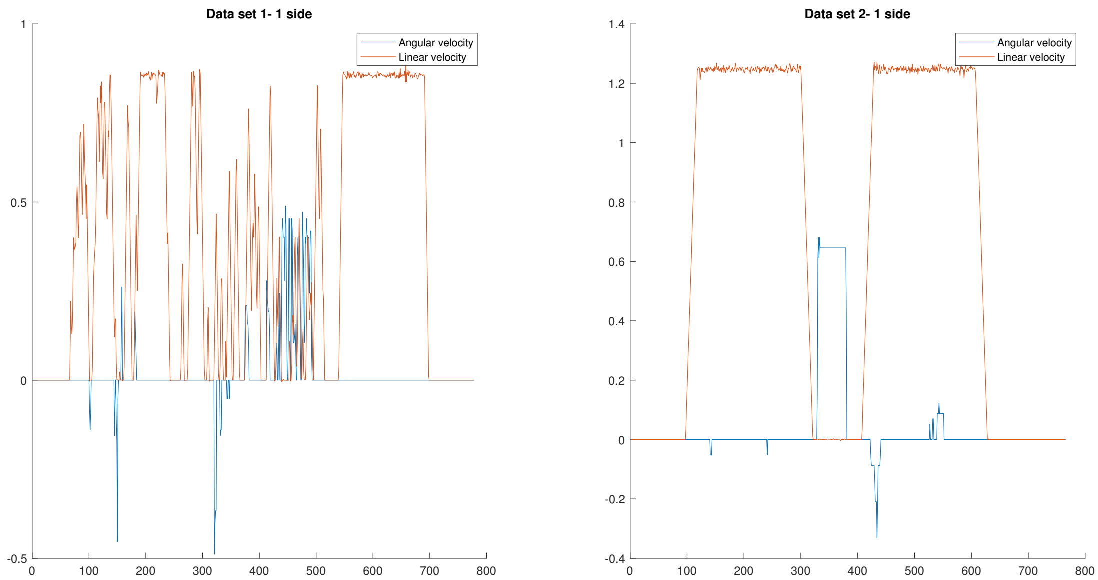

Figure 14: Comparison between velocities - 1 side

Once we did a very large number of simulations with different values of parameters, we conclude that these two combinations are the ones which provide the best results.

Parameters	**First acquirement**	**Second acquirement**
λ	[0.002 0.002 0.00055]	[0.2 0.2 0.0003]
η	180	10
σ	0.16	0.0075
w_{turn}	0.23	0.12
R	[0.01 0; 0 0.0015]	[0.095 0; 0 0.056];
Q	[σ^2 0; 0 0.1283]	[σ^2 0; 0 3.13e-5];
Q_{turn}	[σ^2 0; 0 0.0767]	[σ^2 0; 0 0.1];
τ_1	4 (86,47%)	4.5 (89,46%)
τ_2	24.107 (100%)	27.507 (100%)
Threshold	0.999	0.0335
Prominence	0.320	0.075

Table 1: Exercise 1 - 1 side - parameters

To sum up, we can notice that in the first case, despite the strong inaccuracy of the measurements, we managed to get a quite good estimation of the robot trajectory until it starts turning, while coming back it recognises few landmarks seen before, so the trajectory is very bad estimated, but it recovers almost the initial position. While in the second case, having a quite perfect odometry, surely we got a good estimation of the robot trajectory including the loop closure phenomenon. We have also checked the pose's trend and the amplitude of the confidence intervals. In the first acquirement they become tighter at the end of the trajectory due to the fact that the robot recognizes more or less its initial position; in the second acquirement instead, the intervals are quite constant thanks to the "perfect" odometry. For each experiment, we have compared the map generated by plotting the Lidar scans from the estimated pose and from odometry.

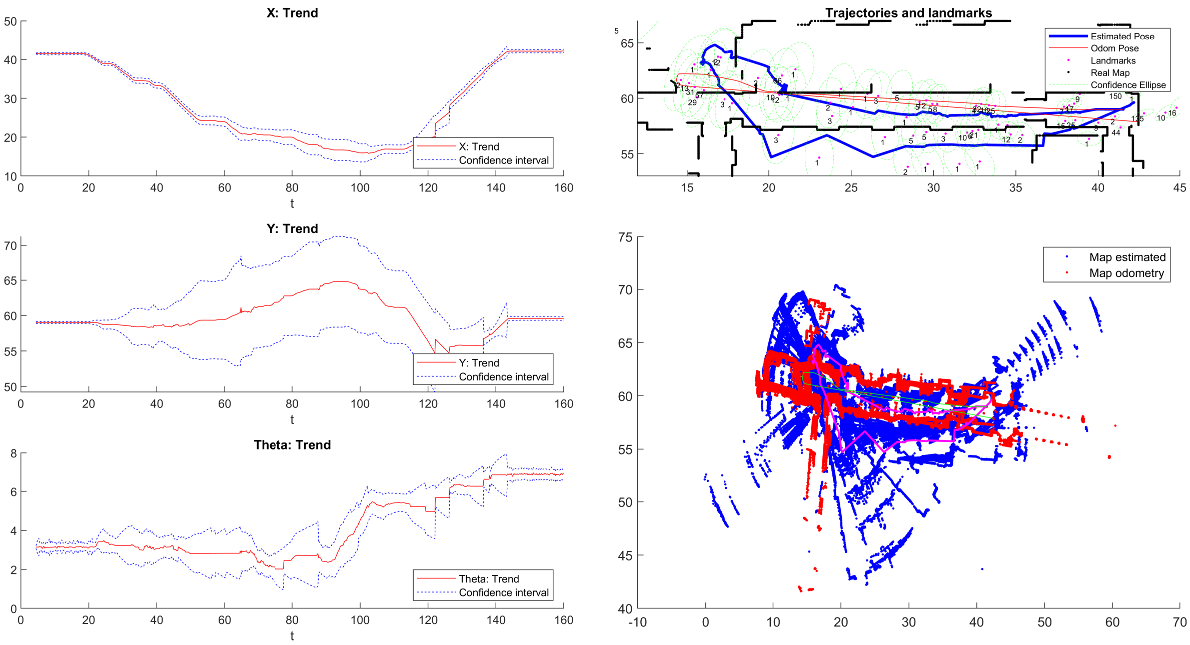

Figure 15: Trajectory and orientation [SLAM D : Data set 1 - 1 side]

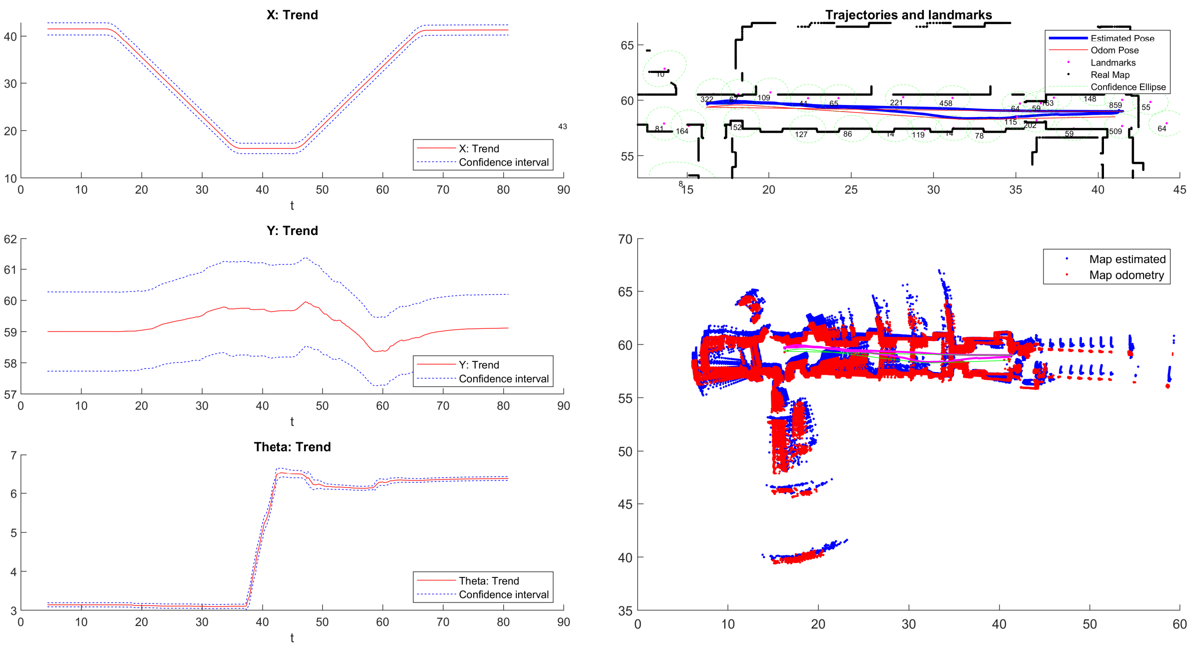

Figure 16: Trajectory and orientation [SLAM D : Data set 2 - 1 side]

Exercise 2 - 4 sides

In this second exercise the robot does a complete tour of the second floor of the San Niccoló building. As in the previous case, we repeated the experiment twice, under the same assumption. We can see in Figure 17 the linear and angular velocities. The parameters chosen for this exercise are in Table 2, and the result are showed in Figure 18 and in Figure 19. As in the Exercise 1 we can see that in the first acquirement case, even if we got a worst odometry, we are able to reconstruct more or less the robot trajectory, while in the second acquirement we notice that the estimated trajectory is more precise. However, the results are not as good as in the experiment with simulated data, this is due to the error introduced by the Lidar measurements and the corners recognition. This is underline by the fact that in the second acquirement even if the odometry is quite perfect, the trajectory is not well reconstructed. Regarding the confidence intervals, they are tigher in the second acquirement as before, but in both cases we can appreciate the loop closure phenomenon in fact in the last part the intervals become smaller, and also we can see it in the trajectory reconstruction .

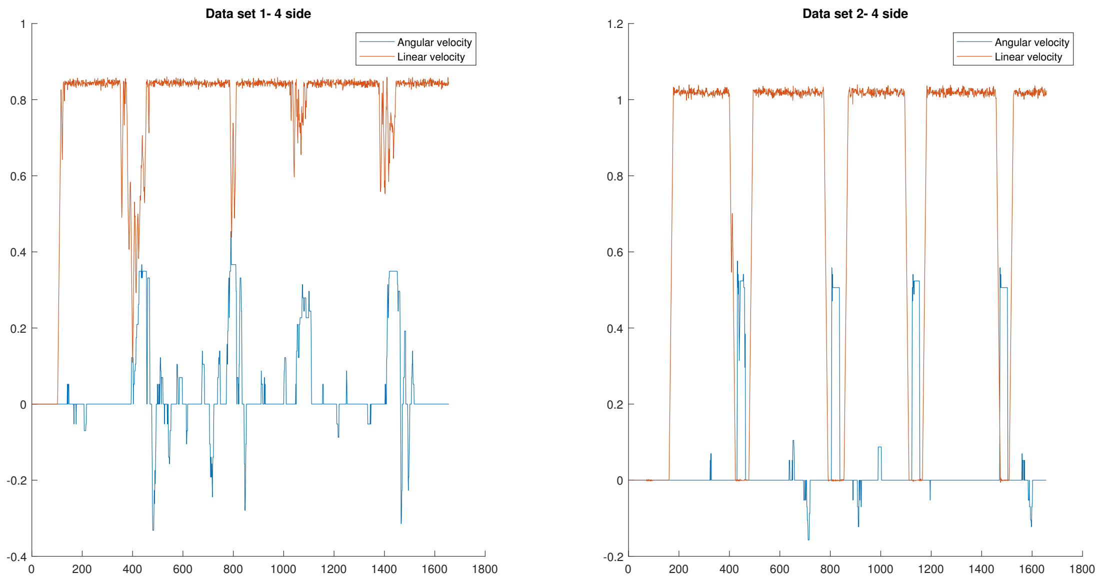

Figure 17: Comparison between velocities - 4 side

Parameters	First acquirement	Second acquirement
λ	[0.2 0.2 0.0002]	[0.01 0.01 0.0001]
η	90	10
σ	0.015	0.0067
w_{turn}	0.12	0.17
R	[0.01 0; 0 0.0015]	[0.01 0; 0 0.0015];
Q	[0.011^2 0; 0 3.13e-6]	[σ^2 0; 0 3.13e-6];
Q_{turn}	[σ^2 0; 0 0.031]	[σ^2 0; 0 0.045];
τ_1	2.5 (71,35%)	4.5 (89,46%)
τ_2	27.507 (100%)	22.507 (100%)
Threshold	0.45	0.78
Prominence	0.18	0.21

Table 2: Exercise 2 parameters

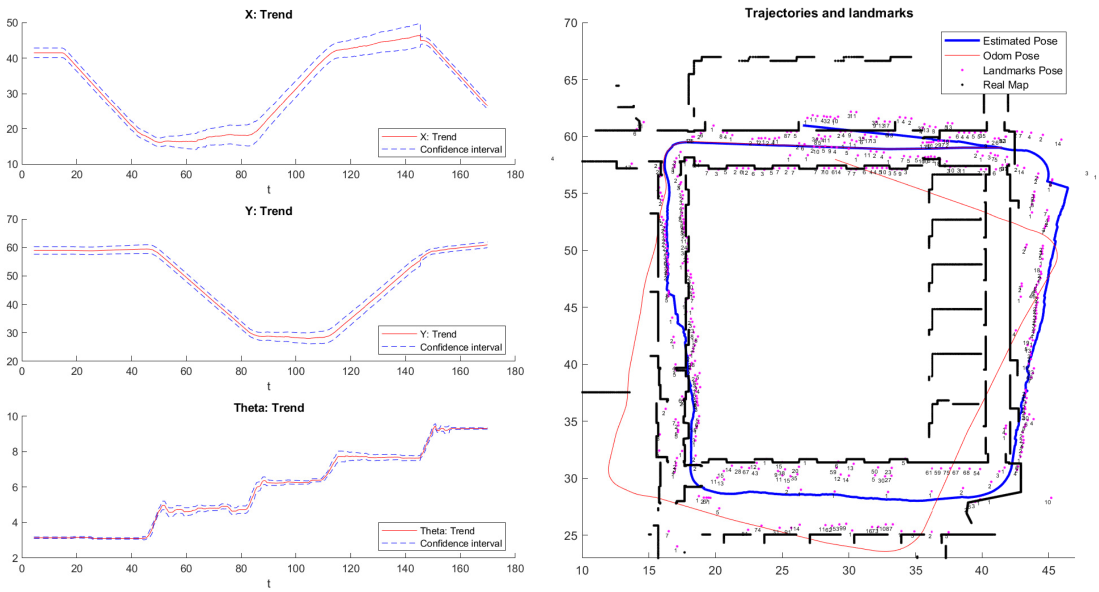

Figure 18: [SLAM D : Data set 1 - 4 sides]

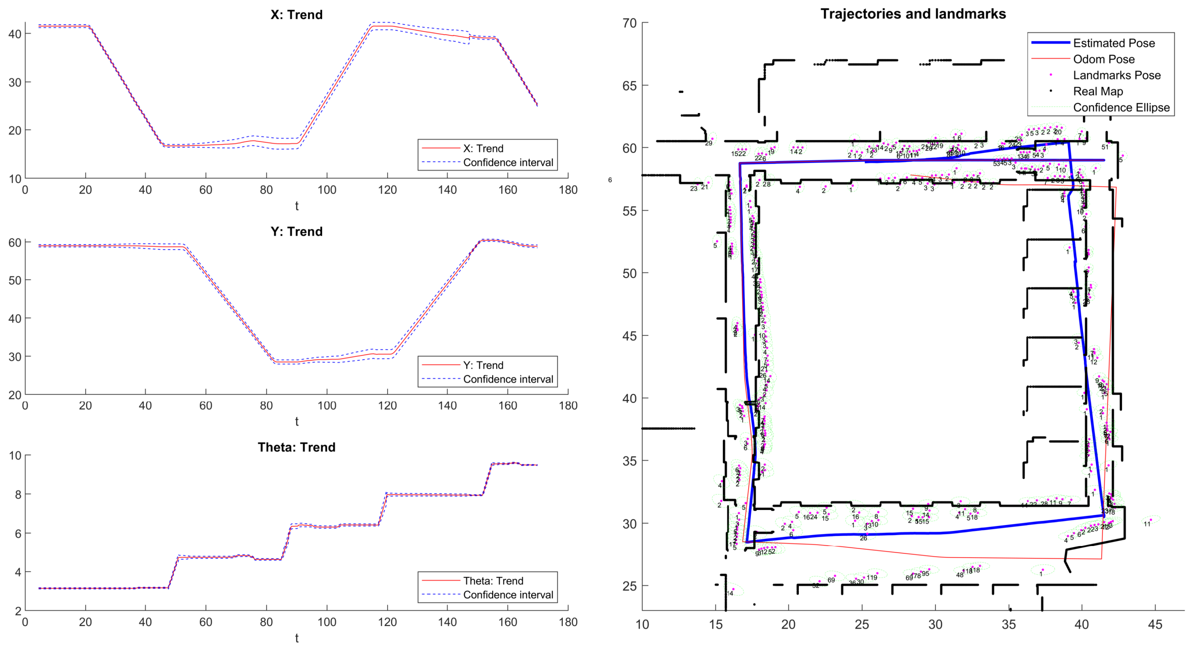

Figure 19: [SLAM D : Data set 2 - 4 sides]

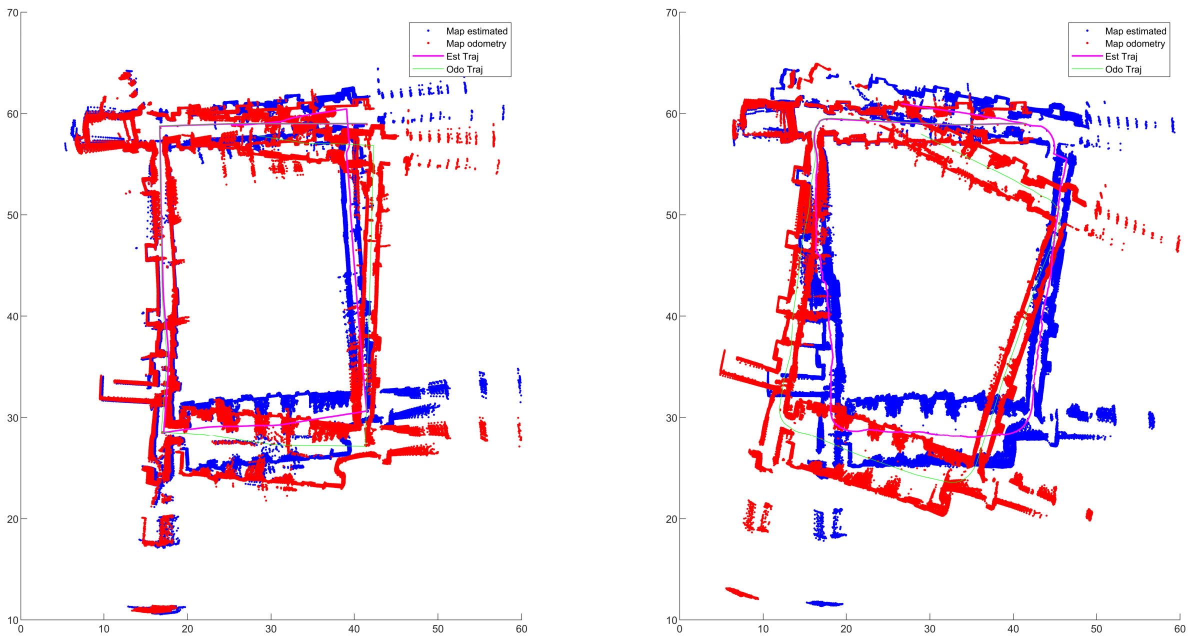

Figure 20: [SLAM D : First Dataset (right side), Second Dataset (left side) - 4 sides]

Conclusions

In conclusion, the project has demonstrated that the implementation of the Kalman filter for SLAM is an effective solution for simultaneous localization and mapping in unknown environments showing that using real-world data, the estimation tends to be less accurate compared to simulation data. However, some improvements can be done to obtain better results. For example, although in SLAM C the extraction of corners worked well, its application in SLAM D has shown its lack of strength and inefficiency going to capture too many points. More efficient extraction or other types such as segmentation or fitting line should be used to address the problem. Moreover, to avoid the extraction of spurious corners, an idea could be to insert unmatched features in a tentative list, adding them later to the state, only if these are seen a certain number of times in a given time frame, making them reliable measurements.