Sparse Feature-Based Visual SLAM
COMPGX04 Coursework 2

Thomas Luo

March 31, 2018

Contents

1 Preliminaries

The vehicle state is represented as a Pose3, which consists of 3 dimensional position and 3 axis rotation. However, we assume that the vehicle drives on a flat ground. Hence, the vehicle at time step 4 is

$$X_k = \begin{pmatrix} x_k \\ y_k \\ \psi_k \end{pmatrix}$$

The motion equation is given by the statement :

$$X_k = X_{k-1} + \begin{pmatrix} V_k \Delta T_k cos(\psi_{k-1} + 0.5\Delta T_k \phi_k) \\ V_k \Delta T_k sin(\psi_{k-1} + 0.5\Delta T_k \phi_k) \\ \Delta T_k \phi_k \end{pmatrix}$$

with ΔT_k the time interval between two successive steps, V_k the signed speed of the vehicle and ϕ_k the rate of heading change. Using a state-transition function $\mathbf{f}$, we can rewrite the motion as $x_k = \mathbf{f}_{u_k}(x_{k-1})$ with $u_k = \begin{pmatrix} V_k \\ \phi_k \end{pmatrix}$ the control input at time step k. Hence :

$$f_{\begin{pmatrix} V \\ \phi \end{pmatrix}} : (x, y, \psi) \mapsto \begin{pmatrix} x + V\Delta T cos(\psi + 0.5\Delta T \phi) \\ y + V\Delta T sin(\psi + 0.5\Delta T \phi) \\ \psi + \phi \Delta T \end{pmatrix}$$

for a given ΔT. From this expression, we can derive its Jacobian $\mathbf{J}$:

$$J = \begin{pmatrix} \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} & \frac{\partial f_1}{\partial \psi} \\ \frac{\partial f_2}{\partial x} & \frac{\partial f_2}{\partial y} & \frac{\partial f_2}{\partial \psi} \\ \frac{\partial f_3}{\partial x} & \frac{\partial f_3}{\partial y} & \frac{\partial f_3}{\partial \psi} \end{pmatrix}$$

with f_1, f_2 and f_3 the rows of f_u. Therefore, at time step k :

$$J_k = \begin{pmatrix} 1 & 0 & -V_k \Delta T_k sin(\psi_k + 0.5\phi_k \Delta T_k) \\ 0 & 1 & V_k \Delta T_k cos(\psi_k + 0.5\phi_k \Delta T_k) \\ 0 & 0 & 1 \end{pmatrix}$$

Besides, the measurement z_k at time step k is given by :

$$z_k = \begin{pmatrix} x_k + \tilde{x_k} \\ y_k + \tilde{y_k} \end{pmatrix}$$

, with $\tilde{x_k}$ and $\tilde{y_k}$ noise in the measurement of the position (x,y). This measurement can be expressed using a sensor measurement function $h = id$ such that

$$z_k = h(X_k) + \tilde{X_k}$$

with $\tilde{X_k} = \begin{pmatrix} \tilde{x_k} \\ \tilde{y_k} \end{pmatrix}$. Furthermore, its jacobian is

$$H = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

These notations are kept for all the report and for the code.

2 Part 1: Odometry and GPS Fusion

2.1 Odometry

In this subsection, we implement the odometry-based localization systems using an Extended Kalman-Filter (EKF) on the one hand, and iSLAM on the other hand.

The EKF predicting stage is given by[1] :

- $\hat{x_k} = f_{u_k}(x_{k-1})$
- $\hat{P_k} = J P_{k-1} J^T + Q$

with P_k the covariance matrix at time step k and Q the covariance matrix of measuring the vehicle control inputs.

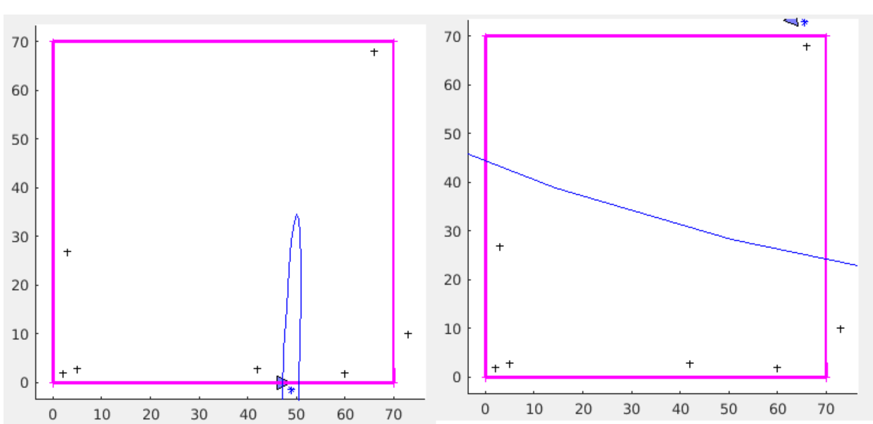

Figure 1: Prediction of the vehicle pose using EKF

Since we are only predicting the pose of the vehicle, we are adding uncertainty at each iteration. Indeed, we can see on figure 1 that the covariance grow larger at each new pose.

Then, we are asked to compute the relative transformation of the vehicle with its associated covariance. the relative transformation corresponds to the relative translation and relative rotation between two consecutive poses X_{k-1} and X_k. Besides,

$$X_k - X_{k-1} = \begin{pmatrix} V_k \Delta T_k cos(\psi_{k-1} + 0.5 \Delta T_k \phi_k) \\ V_k \Delta T_k sin(\psi_{k-1} + 0.5 \Delta T_k \phi_k) \\ \Delta T_k \phi_k \end{pmatrix}$$

Thus, its relative coordinate position is given by $\begin{pmatrix} V_k \Delta T_k cos(\psi_{k-1} + 0.5 \Delta T_k \phi_k) \\ V_k \Delta T_k sin(\psi_{k-1} + 0.5 \Delta T_k \phi_k) \end{pmatrix}$ and its relative orientation is given by $\Delta T_k \phi_k$. At each time step, it rotate by an angle $\Delta T_k \phi_k$ and move $V_k \Delta T_k$ forward. Therefore, the relative translation is $\vec{t_k} = \begin{pmatrix} V_k \Delta T_k \\ 0 \end{pmatrix}$ and the relative rotation is $\Delta T_k \phi_k$.

We also have to compute the covariance matrix of this transformation.

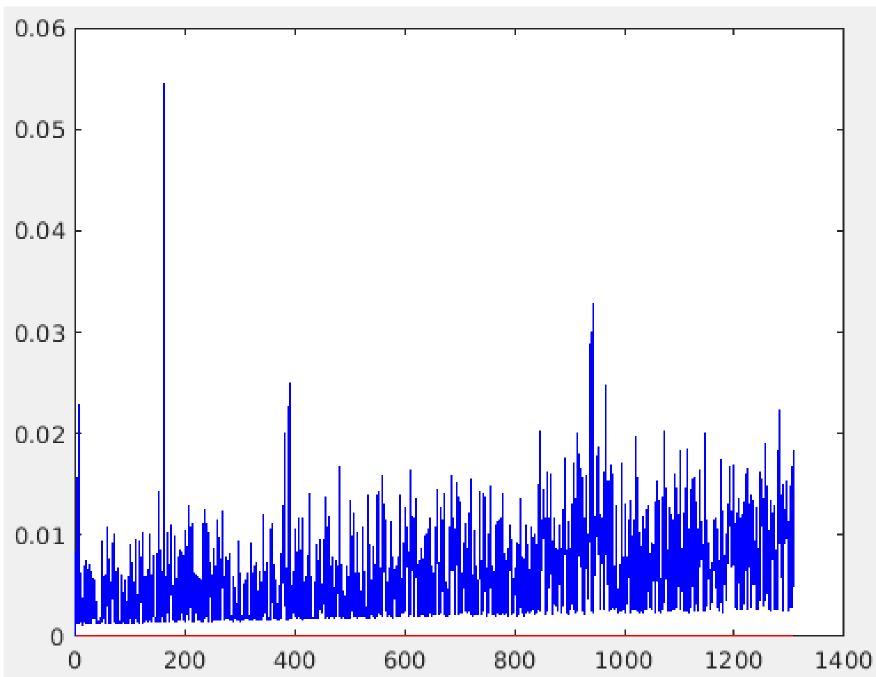

Figure 2: optimization time with GPS measurements

If we look at the optimization time, it globally increases with time. It grows because for each newly estimated state, the newly estimated covariance is "bigger" than the previous one. Since the uncertainty is larger at each iteration, the optimizer need more time to find the best solutions. For instance, if we apply a gradient descent method to a large covariance, the method might need more steps to find the minimum than if we apply it to a smaller covariance.

2.2 GPS

In this subsection, we integrate the the GPS measurements.
The EKF updating stage is given by[1] :

- $G_k = \hat{P}_k H_k^T (H_k P_k H_k^T + R)^{-1}$
- $x_k = \hat{x_k} + G_k(z_k - h(\hat{x_k}))$
- $P_k = (I - G_k H_k) \hat{P}_k$

with R the covariance matrix of measuring positions. One problem there is that the measurement does not provide any orientation informations. As a first approach, I use the orientation predicted by odometry. Then I can update the vehicle pose as if there were an orientation measurement.

Then, we also integrate the GPS measurement to GTSAM model.

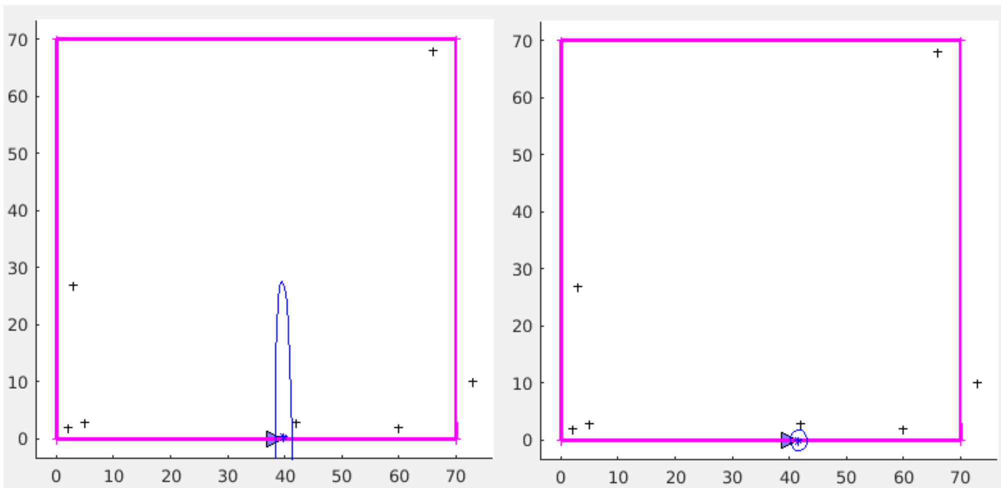

Figure 3: Covariance of the vehicle without GPS measurements, on the left, and with GPS measurement, on the right.

A GPS measurement is added every second. Using a GPS measurement helps to better estimate the successive pose of the vehicle : every second, a GPS measurement "recalibrate" the vehicle's position and, thus, reduce the error related to its estimated position. Thus, we prevent the covariance to grow to large.

If we compare to the previous subsection, we can see on the simulator Output that the "with GPS measurements" covariance is much smaller than the "without GPS measurements" covariance. Indeed, without GPS, the covariance is constantly increasing, which is not the case when we add GPS measurements (see figure 2).

3 Part 2: SLAM System

3.1 SLAM System Development

In this part, I integrate the range-bearing measurements using the landmark. Firstly, the vehicle detects the landmark. If the landmark is already registered, we just updates it with new measurements, else we initializes it using :

$$this.addNewVariable(landmarkKey, initialValue)$$

with

- **landmarkKey** the newly observed landmark.
- **initialValue** the initial coordinates of the landmark project in 2D.

.

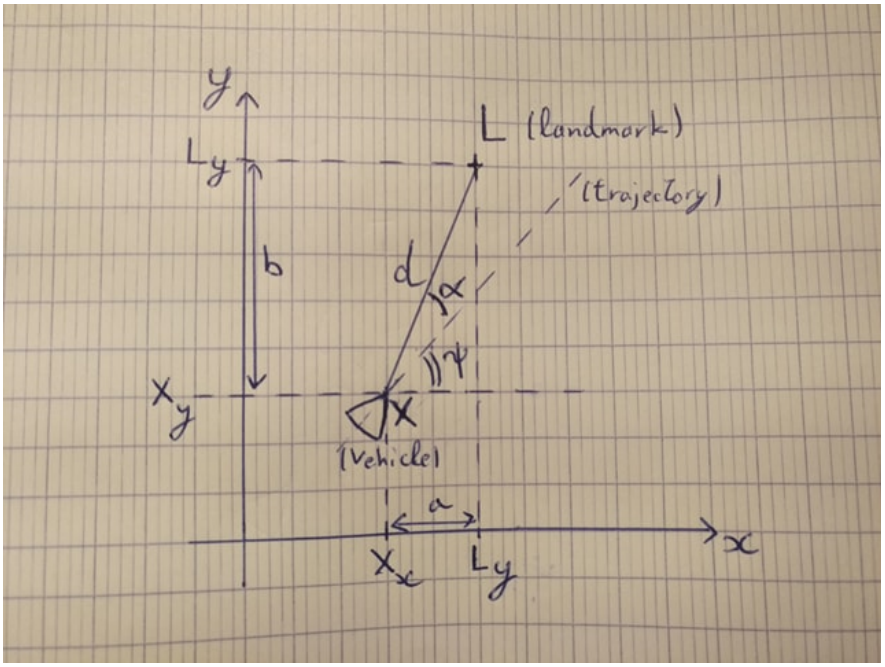

Figure 4: Drawing for determining the coordinate of the landmark L from the vehicle X, with d the range, α the azimuth and ψ the orientation of the vehicle

To determine the initial coordinates of the landmark, we can deduce it using the position of the vehicle, the azimuth to the landmark and the range. With a bit of trigonometry, we have (see figure 5) :

$$\begin{cases} L_x = X_x + a \\ L_y = X_y + b \end{cases}$$

Beside,

$$\begin{cases} a = dcos(\alpha + \psi) \\ b = dsin(\alpha + \psi) \end{cases}$$

$$\Rightarrow \begin{cases} L_x = X_x + dcos(\alpha + \psi) \\ L_y = X_y + dsin(\alpha + \psi) \end{cases}$$

To update it correctly, we have to construct a suitable observation factor using :

```
BearingRangeFactor2D(currentPoseKey, currentLandmarkKey, Rot2(angle), range,
    laserNoiseModel);
```

with

- **currentPoseKey** : the current pose of the vehicle (or current state in the posegraph).
- **currentLandmarkKey** : the current landmark that we are initializing/updating
- **angle** : the azimuth to the landmark
- **range** : the distance d of the vehicle from the landmark in the 2D plane
- **laserNoiseModel** : the measurement noise associated with the range-bearing sensor.

The SLAM system works well if there are enough landmarks. If there are not enough landmarks, or if their distribution in the space is too space, the vehicle's pose is poorly estimated. (see figure 6).

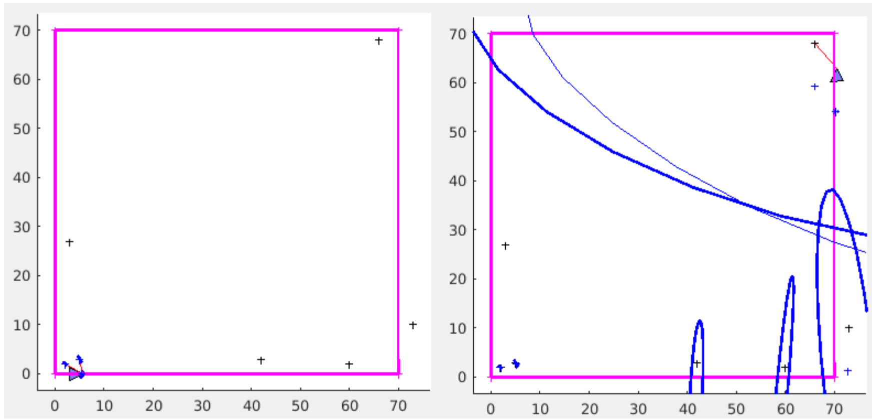

Figure 5: Vehicle pose and landmarks estimations at different time

On the left plot, we can see that the 2 first landmark are relatively close to the initial position of the vehicle. Thus, these landmark positions are well estimated and the pose of the vehicle is also well estimated. However, We can see on the right plot that the SLAm system produces poor result. This is due to the fact that between the second and the third landmark, there was a time interval during which the vehicle couldn't observe any landmark. That introduces large error in the vehicle pose estimation. Since we use the vehicle estimated position to estimate the position of the landmark, if the vehicle pose is poorly estimated, so are the landmark position. This is why the right plot displays poor results.

3.2 Comparison with Odometry and GPS

3.2.1 Odometry measurements

We'll first observe the effect of disabling/enabling odometry measurements

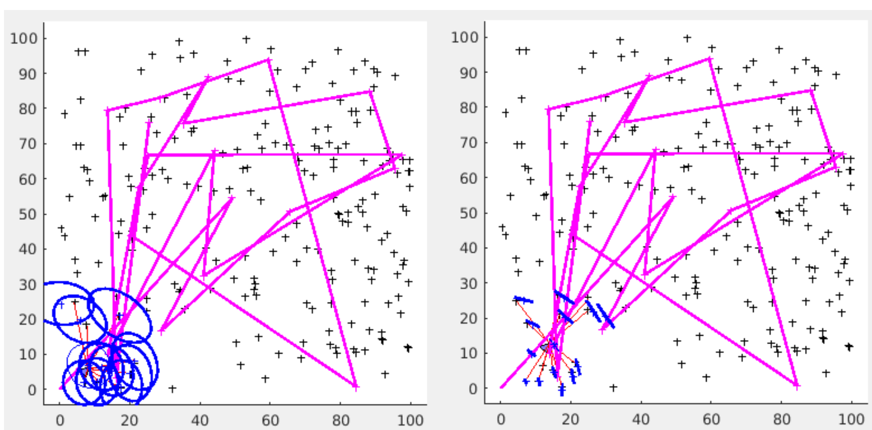

Figure 6: Vehicle pose and landmarks estimations without odometry measurements, on the left, and with odometry measurements, on the right

One major difference between the two plot is the covariance associated to the estimation of the landmark position. By disabling odometry measurements, we can observe that the covariance associated to each landmark become much larger. This is due to the fact that the landmarks' position are estimated from the vehicle pose : with odometry measurements, we have more informations about the vehicle. That reduces uncertainty about its estimated pose. Thus, we know more precisely the pose of the vehicle, which help us to better estimate the position of the landmarks. If we have more uncertainties about the vehicle pose, that implicitly introduces more uncertainties about the landmarks' position estimations.

We don't see it clearly but like the landmark, the covariance of the vehicle pose is also much larger without odometry measurements than with. This is closely related to what was said just above : with odometry measurement, the landmarks' position are better estimated than without odometry measurement. Since the landmarks' position are better estimated, we can better estimate the vehicle's pose, which reduce its associated covariance.

3.2.2 GPS Measurements

Adding GPS measurements helps to better estimate the lardmarks' position and, so, the vehicle's pose (see girue 8 just below).

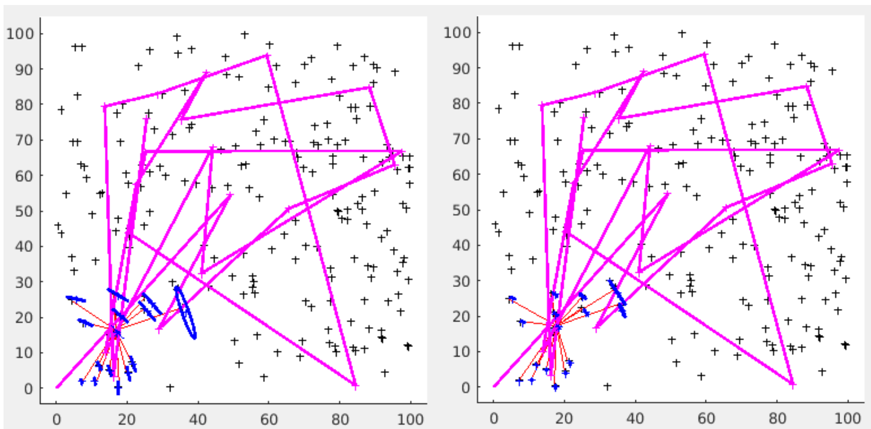

Figure 7: Vehicle pose and landmarks estimations without GPS measurements, on the left, and with GPS measurements, on the right

As explained in the previous subsubsection, adding GPS measurement improves the estimation of the vehicle's pose, which improves the estimations of the landmarks' position, which also improves the estimation of the vehicle's pose.
A GPS measurement is added every second, and we can see the effect it has on the covariance of the different objects (see figure 9) :

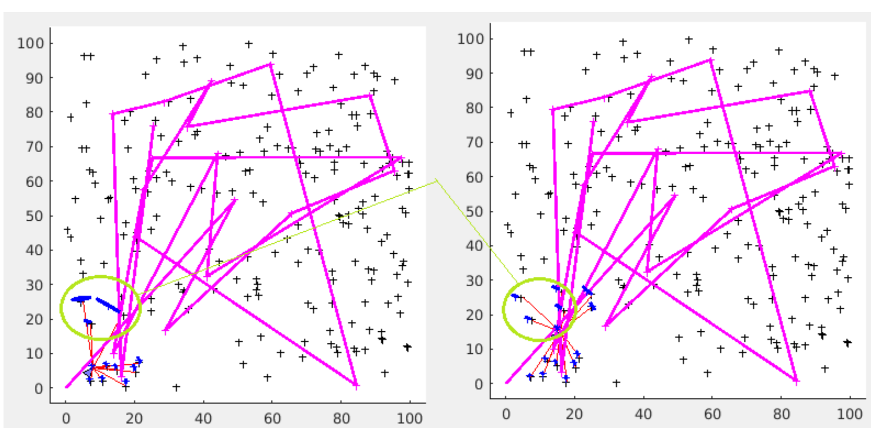

Figure 8: Vehicle pose and landmarks estimations before the addition of a GPS measurement (on the left) and after the addition of it (on the right)

We can see on the right plot that adding a GPS measurement greatly reduce the covariance associated to the landmarks' position

References

[1] Julier Simon *COMPGX04: Robotic Vision and Navigation Using a Kalman Filter for SLAM* . UCL, Lecture Slides.