SLAM2

ab1

March 18 ,2017

1 2D-LINEAR SLAM

The Odometry measurement :

odom = [$\mathrm{r}^t - r^{t-1}$]
$[\Delta_x, \Delta_y] = [{r_x}^t - {r_x}^{t-1}, {r_y}^t - {r_y}^{t-1}]$
Therefore,
$\mathrm{h}_o = [{r_x}^t - {r_x}^{t-1}, {r_y}^t - {r_y}^{t-1}]$
the x position of robot at time t : r_x^t

Landmark Measurement =
$\mathrm{h}_m = [l_x - r_x, l_y - r_y]$
Jacobians :

$$H_o = \begin{bmatrix} [-1, 0, 1, 0] \\ [0, -1, 0, 1] \end{bmatrix}$$

Landmark Jacobian : Differentiating $\mathrm{h}_m withrespecttor_x, r_y, l_x, l_y$

$$H_m = \begin{bmatrix} [-1, 0, 1, 0] \\ [0, -1, 0, 1] \end{bmatrix}$$

2 1,c,(iii) QR Decomosition

The "economy" size should be used because it converts the solution to square matrix and helps in decomposition/ factorization. Hence using economy, converts the matrix to square to help in factorization/decomposition.

3 1,d,(i) 2D linear dataset

The order of efficiency :
Chol2
QR2
Chol1

QR1
Pinv

Chol2 and QR2 are better because they are more sparse and easy to factorize / decompose. In the previous case also they were in the top 2.

Chol1 and QR1 are dense towards the end as can be seen from the figure.

Pinv is the most expensive task because the computations are very high as it does not use any optimization.

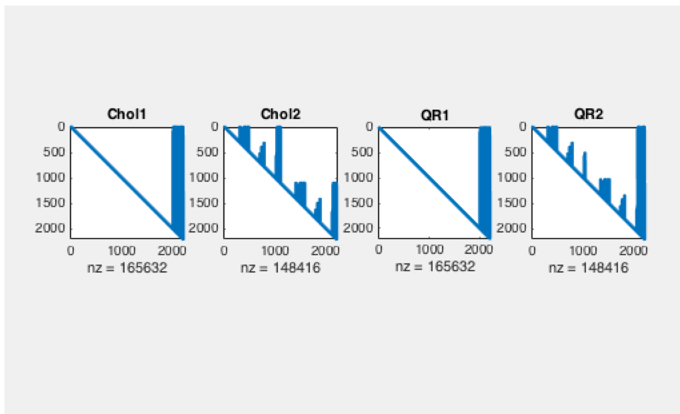

Figure 1: 2D LINEAR DATA SET

4 1,d,(ii) 2D linear loop dataset

The order of efficiency :

QR2
Chol2
Pinv
Chol1
QR1

The order is different from the 2D Linear data-set.

As it can be seen from the figure, QR1 and Chol1 become more dense than they were before and hence , their decomposition or factorization takes more time than Pinv

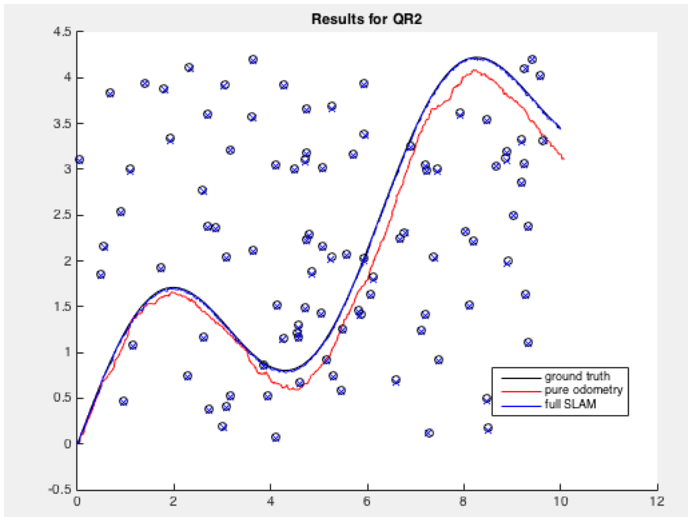

Figure 2: 2D LINEAR DATA SET

In the case of Linear , the QR1 and Chol1 were sparser and thus took less time than Pinv.

QR2 and Chol2 are stil in the first two. The QR2 becomes more efficient than Chol2 because it becomes slightly less dense

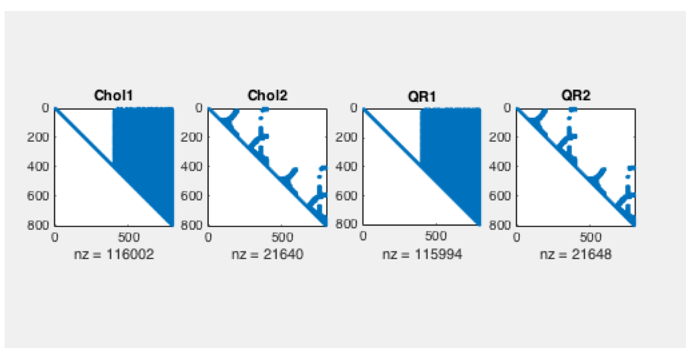

Figure 3: 2D LINEAR LOOP DATA SET

5 2(b) 2D NON LINEAR SLAM

Jacobian for non-linear landmark function :

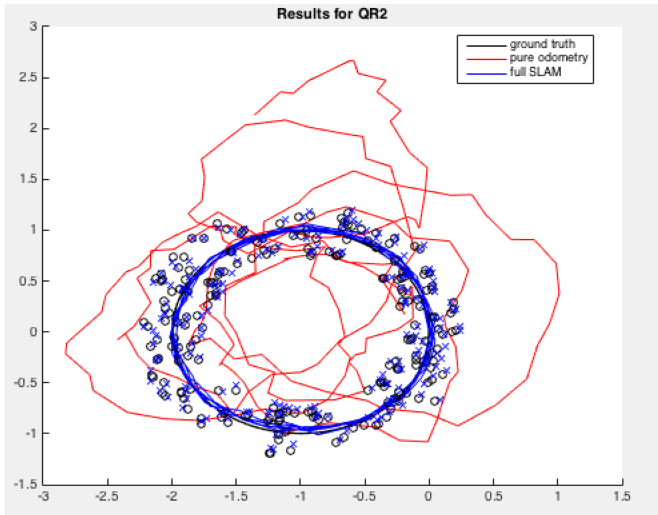

Figure 4: 2D LINEAR LOOP DATA SET

a = $((l_x - r_x)^2 + (l_y - r_y)^2)$

$$H_m = \begin{bmatrix} [(l_y - r_y)/a, (r_x - l_x)/a, (r_y - l_y)/a, (l_x - r_x)/a] \\ [(r_x - l_x)/(a^{0.5}), (r_y - l_y)/(a^{0.5}), (-r_x + l_x)/(a^{0.5}), (-r_y + l_y)/(a^{0.5})] \end{bmatrix}$$

6 2(d) 2D NON LINEAR SLAM

$\text{rmse}_o dom =$
0.0579

$\text{rmse}_s lam =$
0.0153

7 2(e) 2D NON LINEAR SLAM

We used incremental fashion for the results because it prevents the solution from being stuck at the local Minima and being closer to the solution during the start of the optimization at every time step.

If we implement the batch optimization at once , the solution might not converge at the minimum and might suffer from being stuck there.

Hence, it is better to start optimizing incrementally to converge at a global minimum toward the end.

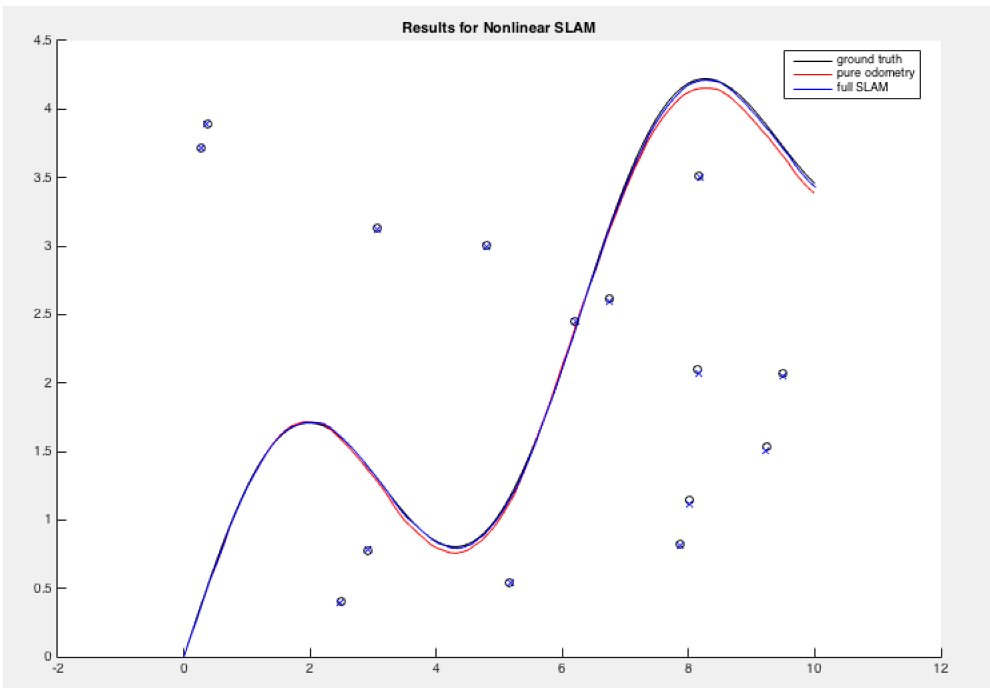

Figure 5: 2D LINEAR LOOP DATA SET