MINGI JEONG
F00422M
Mingi.Jeong.GR@dartmouth.edu

Programming assignment -2
COSC69/169 – Robotics Perception – Winter 2020

1 Description

1.1 Introduction

This program was made to implement the codes of Rosbot's state estimate based on Kalman filter algorithm. Assuming that the robot is following the movement in 1-D system, the Kalman filter algorithm estimate a state using a saved rosbag file. Based on particular models for propagation(`cmd_vel` or `pose`) and measurement (lidar scan or camera depth), the program initiate the filter. For each launch file, you can also subscribe to topic "kalman_filter" in order to check published messages containing `PoseWithCovarianceStamed` type. The parameters for the Kalman filter are based on the initial belief and measurement errors. In addition, a user can check tf_echo between `odom_kf` ad `base_footprint`. The final results can be visualized using python scripts containing matplotlib.

1.2 How my program work

- Please refer to the readme on github `https://github.com/MingiJeong/kalman_filter_mg_cs169`
- rosbag file link `https://drive.google.com/open?id=1onVc88q--nnUFm7KvOSchqn1i-yoMVre`
- **Important:** screen -R core —– roscore
- **Important:** screen -R initializer —– rosrun kalman_filter_mg_cs169 initial_pose.py.
- After all configuration is done with download, for task 1, you can run "roslaunch kalman_filter_mg_cs169 cmd_estimate.launch" The calculation will be shown in 5 secs because I executed `cmd_vel` for moving a robot by establishing the serial connection.
- The launch file contains all the necessary parts including the requirement for graduate students. The user can check "rostopic echo kalman_filter" or tf echo.
- For task 2, you can launch each file which you are interested in.
  - Task 2-A, 2-C : pose_estimate.launch
  - Task 2-B, 2-F : cmd_estimate.launch
  - Task 2-D : cmd_estimate_camera.launch
  - Task 2-E : pose_estimate_camera.launch
- For task 3, you can run on python script "python yourownpath/plotter.py" and the plotted graphs are shown in pop-up as well as saved in a pdf format.

1.3 Design Decision

- Basically, this program assumes that the robot navigates in 1-D world even if the odom data is technically based on 2-D inputs (x,y) (actually, it contains 3-D, but the robot moves in 2-D on a plane; that's why I mentioned 2-D). Therefore, I used 1-D accumulated distance from starting point as of **"state"** we are interested in attaining by the Kalman filter.
- A system model using `cmd_vel` is based on the robot's kinematic model as per linear velocity along x-axis.
- On the other hand, a system model using `pose` is based on wheel odometry's data recorded on the bag file. Since it contains x, y coordinates together and both of them are changing while the robot is moving, the user should be careful of confusion by only calculating x difference.
- Hence, I used Euclidean distance between the current pose and the previous pose data. Since this is already inclusive of delta state data (in Kalman filter algorithm, it can be B, U part in the system model), I have only to interpolate it by making the propagation time synchronous with the measurement timestamp (lidar or camera).

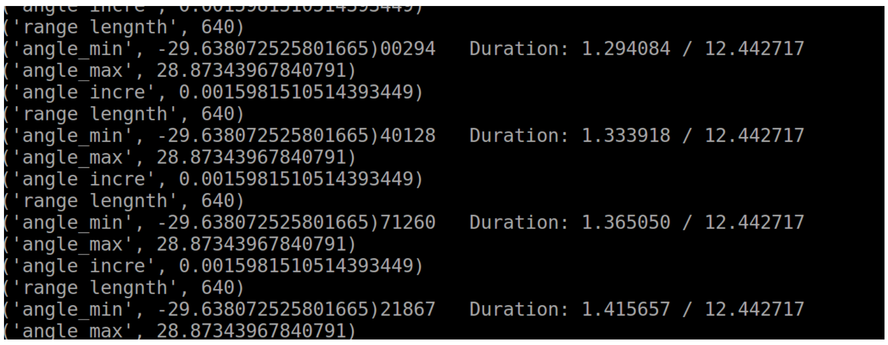
- The lidar scan data are from the straightforawrd measurement as of index 0. Some outliers when the lidar detects infinity are excluded for the update model. In the same way, the camera depth data are from the straightforwad as far as possible. Since it has a tiny difference between the left and right angle, I should find an index as the middle (324) as shown in Fig. 1. Then, as the data were very unstable (transformed into laser scan), I recalled nearby data +-10 index to input the minimum value among them.

- Also, I considered signal frequencies for processing this filter. Even if the user published or recorded topics in accordance with preset `Hz`, the reality does not work in that way. Thus, I measured time differences and interpolated as per the fundamental concept of the Kalman filter. That is, the propagation is adjusted depending on the measurement error (between the actual measurement and the expected measurement from the propagation).

- In summary, the output state estimate follows the timestamp of the measurement data. The data are stored in the user-defined directory as csv formats.

- After all programs are finished, the user can check the visualized plot. The rospy script in this package has not been fully tested because I kept encountering an error message regarding ”import yaml” which results from python version. Therefore, the user has only to run python script for plotting (There is no point of using matplotlib through ros particulary in this case).

- rosbag play follows the time when it was recorded. The initial time 0 is based on the first time `cmd_vel` was published. By doing so, it is efficient to compare paths and errors under the normalized time point. I used the function rospy.get_time() except for some parts where rospy.Time.now() is needed. This is because it makes easier for me to calculate the time differences in seconds.



```
('angle incre', 0.0015981510514393449)
('range lengnth', 640)
('angle_min', -29.638072525801665)00294   Duration: 1.294084 / 12.442717
('angle_max', 28.87343967840791)
('angle incre', 0.0015981510514393449)
('range lengnth', 640)
('angle_min', -29.638072525801665)40128   Duration: 1.333918 / 12.442717
('angle_max', 28.87343967840791)
('angle incre', 0.0015981510514393449)
('range lengnth', 640)
('angle_min', -29.638072525801665)71260   Duration: 1.365050 / 12.442717
('angle_max', 28.87343967840791)
('angle incre', 0.0015981510514393449)
('range lengnth', 640)
('angle_min', -29.638072525801665)21867   Duration: 1.415657 / 12.442717
('angle_max', 28.87343967840791)
```

Figure 1: Finding angles or camera depth

2 Evaluation

2.1 Performance

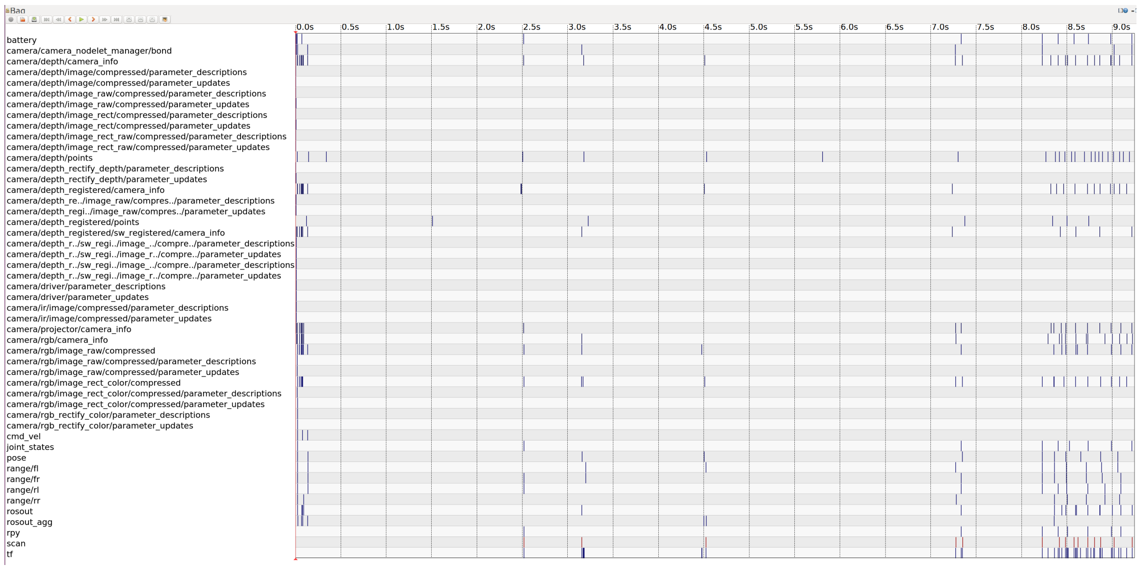
- As aforementioned, the parameters for the Kalman filter have been decided by kinematic model, system transition, sensor or system transition error and etc.

- For example, I used higher R value in camera than laser as I found out the transformed depth data are not that stable.

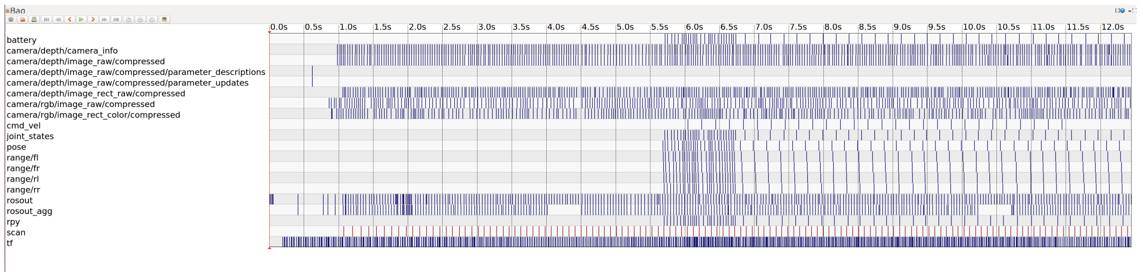
- At first, when I checked with rqt_bag from PA-1, I realized that some topics are not recorded correctly and many of them have only 1 messages. I assume that some camera-related topics are interfering a proper record; thus, I filtered out unnecessary topics and recorded again to meet the requirements of PA-2 as shown in Fig. 2.

- After tasks related with lidar scan were finished, I found out that topic `/camera/depth/image/compressed` is not enough for converting the compressed images into raw images. Therefore, I recorded data again particularly including topics such as `/camera/depth/camera_info`.

- The user can also check visualization of how it works by subscribing to the original bagfile’s topics such as pose; otherwise, it is also possible to monitor the movement from `odom_kf` and `initialpose` as shown in Fig. 3


Bag
battery
camera/camera_nodelet_manager/bond
camera/depth/camera_info
camera/depth/image/compressed/parameter_descriptions
camera/depth/image/compressed/parameter_updates
camera/depth/image_raw/compressed/parameter_descriptions
camera/depth/image_raw/compressed/parameter_updates
camera/depth/image_rect/compressed/parameter_descriptions
camera/depth/image_rect/compressed/parameter_updates
camera/depth/image_rect_raw/compressed/parameter_descriptions
camera/depth/image_rect_raw/compressed/parameter_updates
camera/depth/points
camera/depth_rectify_depth/parameter_descriptions
camera/depth_rectify_depth/parameter_updates
camera/depth_registered/camera_info
camera/depth_re../image_raw/compres../parameter_descriptions
camera/depth_regi../image_raw/compres../parameter_updates
camera/depth_registered/points
camera/depth_registered/sw_registered/camera_info
camera/depth_r../sw_regi../image_../compre../parameter_descriptions
camera/depth_r../sw_regi../image_r../compre../parameter_updates
camera/depth_r../sw_regi../image_../compre../parameter_descriptions
camera/depth_r../sw_regi../image_r../compre../parameter_updates
camera/driver/parameter_descriptions
camera/driver/parameter_updates
camera/ir/image/compressed/parameter_descriptions
camera/ir/image/compressed/parameter_updates
camera/projector/camera_info
camera/rgb/camera_info
camera/rgb/image_raw/compressed
camera/rgb/image_raw/compressed/parameter_descriptions
camera/rgb/image_raw/compressed/parameter_updates
camera/rgb/image_rect_color/compressed
camera/rgb/image_rect_color/compressed/parameter_descriptions
camera/rgb/image_rect_color/compressed/parameter_updates
camera/rgb_rectify_color/parameter_descriptions
camera/rgb_rectify_color/parameter_updates
cmd_vel
joint_states
pose
range/fl
range/fr
range/rl
range/rr
rosout
rosout_agg
rpy
scan
tf
0.0s 0.5s 1.0s 1.5s 2.0s 2.5s 3.0s 3.5s 4.0s 4.5s 5.0s 5.5s 6.0s 6.5s 7.0s 7.5s 8.0s 8.5s 9.0s

(a)


Bag
battery
camera/depth/camera_info
camera/depth/image_raw/compressed
camera/depth/image_raw/compressed/parameter_descriptions
camera/depth/image_raw/compressed/parameter_updates
camera/depth/image_rect_raw/compressed
camera/rgb/image_raw/compressed
camera/rgb/image_rect_color/compressed
cmd_vel
joint_states
pose
range/fl
range/fr
range/rl
range/rr
rosout
rosout_agg
rpy
scan
tf
0.0s 0.5s 1.0s 1.5s 2.0s 2.5s 3.0s 3.5s 4.0s 4.5s 5.0s 5.5s 6.0s 6.5s 7.0s 7.5s 8.0s 8.5s 9.0s 9.5s 10.0s 10.5s 11.0s 11.5s 12.0s

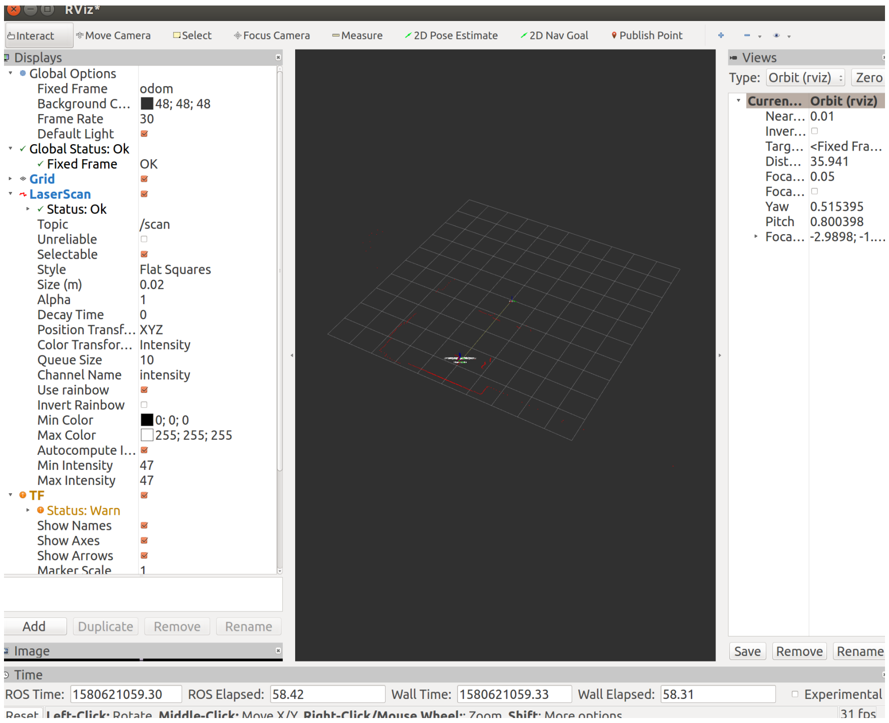
(b)

Figure 2: Comparison of collected data quality (a) original rosbag dropping necessary topics despite the robot's actual movement under 0.2 m/s (b) recollected rosbag after filtering out unnecessary topics

- I designed the initial pose randomly inputted to the system. The launch files subscribe to the initial pose while the initial pose node is running behind.
- For robustness of my program, several tests with different rosbag files were found to work correctly. Those bag files were shared in my google drive for your reference. `https://drive.google.com/open?id=1vMkeJstWFF-33-PqwsXU46wQfwEDaY1_`. The demo screen is as shown in Fig. 4

2.2 Result

- Results are as shown in and Fig. 5 and Fig. 6.
- The strategy using very sensitive time difference following the ideal model of Kalman filter was analyzed as per each state estimate process. When a laser was used as a main measurement source, it seems very reliable.
- On the contrary, the transformed camera depths have very unstable measurements and they generate unreliable state estimates. The depth image data themselves have many errors or fluctuations because I observed many ”NaN” or improper values(e.g. 0.4) given that the robot moves from 2 meter away from the wall to 1 meter away. The model based on the camera can be more improved by changing the parameter. For example, when I changed R value from 3 to 20 for the Kalman filter using depth image, it follows better than before. However, the last state estimate approaches almost double of the path that the robot actually travelled. This indicates that oscillation can be reduced by tuning parameters, but the accuracy of the estimate can be not that great [2]. Comprehensive investigation for future applications [1].
- For this reason, finding some good parameters and designing good system models looks essential to apply the Kalman filter algorithm.
- In conclusion, the model I implemented follows very well when it comes to the lidar scan and robustness compatible with several rosbag files without an error. In particular, the signal processing method based on actual signal timestamp difference and interpolation will be useful for my further researches.

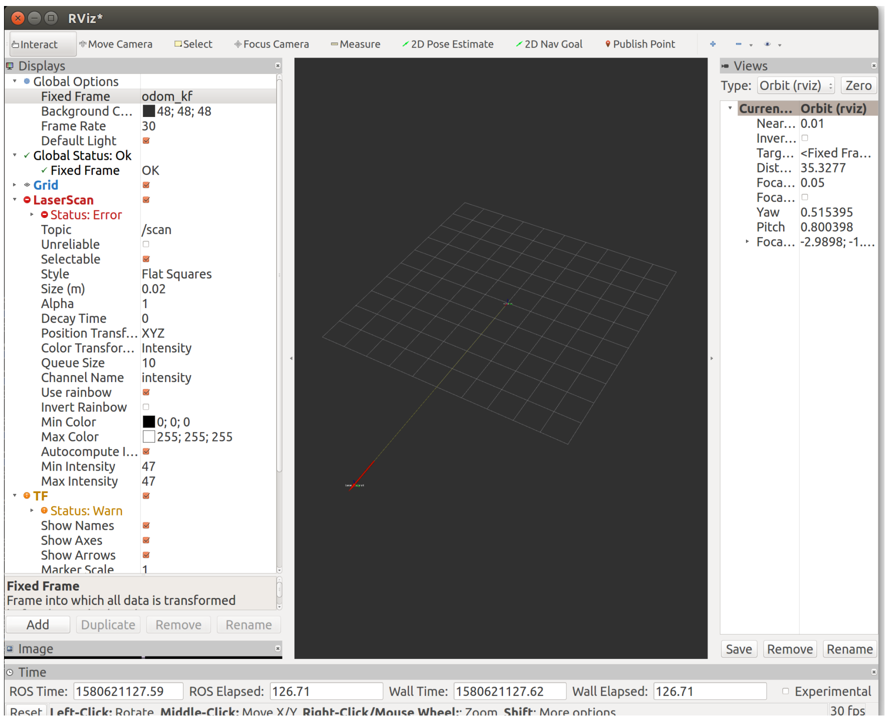


Global Options
Fixed Frame odom
Background C... 48; 48; 48
Frame Rate 30
Default Light
Global Status: Ok
Fixed Frame OK
Grid
LaserScan
Status: Ok
Topic /scan
Unreliable
Selectable
Style Flat Squares
Size (m) 0.02
Alpha 1
Decay Time 0
Position Transf... XYZ
Color Transfor... Intensity
Queue Size 10
Channel Name intensity
Use rainbow
Invert Rainbow
Min Color 0; 0; 0
Max Color 255; 255; 255
Autocompute I...
Min Intensity 47
Max Intensity 47
TF
Status: Warn
Show Names
Show Axes
Show Arrows
Marker Scale 1

Type: Orbit (rviz) Zero
Curren... Orbit (rviz)
Near... 0.01
Inver...
Targ... <Fixed Fra...
Dist... 35.941
Foca... 0.05
Foca...
Yaw 0.515395
Pitch 0.800398
Foca... -2.9898; -1....

ROS Time: 1580621059.30 ROS Elapsed: 58.42 Wall Time: 1580621059.33 Wall Elapsed: 58.31 Experimental
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options. 31 fps

(a)



Global Options
Fixed Frame odom_kf
Background C... 48; 48; 48
Frame Rate 30
Default Light
Global Status: Ok
Fixed Frame OK
Grid
LaserScan
Status: Error
Topic /scan
Unreliable
Selectable
Style Flat Squares
Size (m) 0.02
Alpha 1
Decay Time 0
Position Transf... XYZ
Color Transfor... Intensity
Queue Size 10
Channel Name intensity
Use rainbow
Invert Rainbow
Min Color 0; 0; 0
Max Color 255; 255; 255
Autocompute I...
Min Intensity 47
Max Intensity 47
TF
Status: Warn
Show Names
Show Axes
Show Arrows
Marker Scale 1

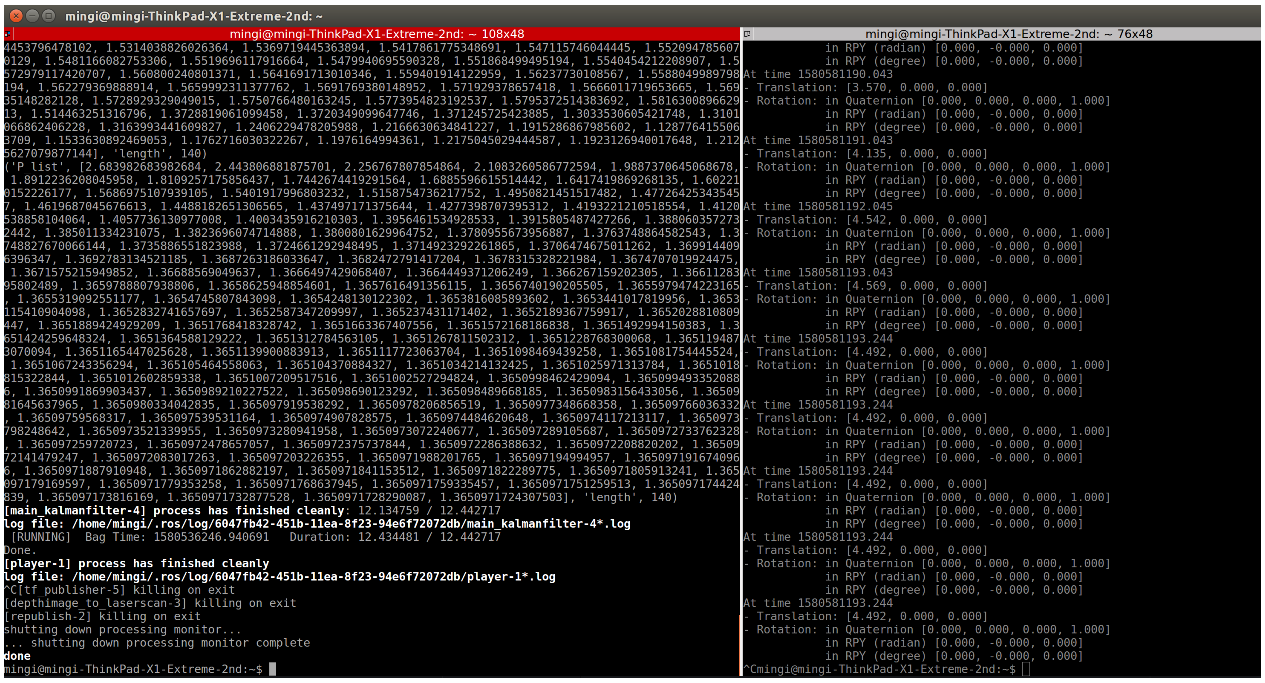
Fixed Frame
Frame into which all data is transformed

Type: Orbit (rviz) Zero
Curren... Orbit (rviz)
Near... 0.01
Inver...
Targ... <Fixed Fra...
Dist... 35.3277
Foca... 0.05
Foca...
Yaw 0.515395
Pitch 0.800398
Foca... -2.9898; -1....

ROS Time: 1580621127.59 ROS Elapsed: 126.71 Wall Time: 1580621127.62 Wall Elapsed: 126.71 Experimental
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options. 30 fps

(b)

Figure 3: RVIZ visualization (a) subscription to rosbag file's topics (b) subscription to published topic as of kalman filter (task1's question for graduate students)


mingi@mingi-ThinkPad-X1-Extreme-2nd: ~
mingi@mingi-ThinkPad-X1-Extreme-2nd: ~ 108x48
mingi@mingi-ThinkPad-X1-Extreme-2nd: ~ 76x48

```
4453796478102, 1.5314038826026364, 1.5369719445363894, 1.5417861775348691, 1.547115746044445, 1.552094785607
0129, 1.5481166082753306, 1.5519696117916664, 1.5479940695590328, 1.551868499495194, 1.5540454212208907, 1.5
572979117420707, 1.560800240801371, 1.5641691713010346, 1.559401914122959, 1.56237730108567, 1.5588049989798
194, 1.562279369888914, 1.5659992311377762, 1.5691769380148952, 1.571929378657418, 1.5666011719653665, 1.569
35148282128, 1.5728929329049015, 1.5750766480163245, 1.5773954823192537, 1.5795372514383692, 1.5816300896629
13, 1.514463251316796, 1.3728819061099458, 1.3720349099647746, 1.371245725423885, 1.3033530605421748, 1.3101
066862406228, 1.3163993441609827, 1.2406229478205988, 1.2166630634841227, 1.1915286867985602, 1.128776415506
3709, 1.1533630892469053, 1.1762716030322267, 1.1976164994361, 1.2175045029444587, 1.1923126940017648, 1.212
5627079877144], 'length', 140)
('P_list', [2.683982683982684, 2.443806881875701, 2.256767807854864, 2.1083260586772594, 1.9887370645068678,
 1.8912236208045958, 1.8109257175856437, 1.7442674419291564, 1.6885596615514442, 1.6417419869268135, 1.60221
0152226177, 1.5686975107939105, 1.5401917996803232, 1.5158754736217752, 1.4950821451517482, 1.47726425343545
7, 1.461968704567661 3, 1.4488182651306565, 1.437497171375644, 1.4277398707395312, 1.419322121051855 4, 1.4120
538858104064, 1.4057736130977008, 1.4003435916210303, 1.3956461534928533, 1.3915805487427266, 1.38806035727 3
2442, 1.385011334231075, 1.3823696074714888, 1.3800801629964752, 1.3780955673956887, 1.3763748864582543, 1.3
748827670066144, 1.3735886551823988, 1.3724661292948495, 1.3714923292261865, 1.3706474675011262, 1.369914409
6396347, 1.3692783134521185, 1.3687263186033647, 1.3682472791417204, 1.3678315328221984, 1.3674707019924475,
 1.3671575215949852, 1.36688569049637, 1.3666497429068407, 1.3664449371206249, 1.366267159202305, 1.36611283
95802489, 1.3659788807938806, 1.3658625948854601, 1.3657616491356115, 1.3656740190205505, 1.3655979474223165
, 1.3655319092551177, 1.3654745807843098, 1.3654248130122302, 1.3653816085893602, 1.3653441017819956, 1.3653
115410904098, 1.3652832741657697, 1.3652587347209997, 1.365237431171402, 1.3652189367759917, 1.3652028810809
447, 1.3651889424929209, 1.3651768418328742, 1.3651663367407556, 1.3651572168186838, 1.3651492994150383, 1.3
651424259648324, 1.3651364588129222, 1.3651312784563105, 1.3651267811502312, 1.3651228768300068, 1.365119487
3070094, 1.3651165447025628, 1.3651139900883913, 1.3651117723063704, 1.3651098469439258, 1.3651081754445524,
 1.3651067243356294, 1.365105464558063, 1.365104370884327, 1.3651034214132425, 1.3651025971313784, 1.3651018
815322844, 1.3651012602859338, 1.3651007209517516, 1.3651002527294824, 1.3650998462429094, 1.365099493352088
6, 1.3650991869903437, 1.3650989210227522, 1.365098690123292, 1.365098489668185, 1.3650983156433056, 1.36509
81645637965, 1.3650980334042835, 1.365097919538292, 1.3650978206856519, 1.3650977348668358, 1.3650976603633 2
, 1.36509759568317, 1.365097539531164, 1.3650974907828575, 1.3650974484620648, 1.3650974117213117, 1.3650973
798248642, 1.3650973521339955, 1.3650973280941958, 1.3650973072240677, 1.365097289105687, 1.3650972733762328
, 1.365097259720723, 1.3650972478657057, 1.3650972375737844, 1.3650972286388632, 1.3650972208820202, 1.36509
72141479247, 1.3650972083017263, 1.365097203226355, 1.3650971988201765, 1.365097194994957, 1.365097191674096
6, 1.3650971887910948, 1.3650971862882197, 1.3650971841153512, 1.3650971822289775, 1.3650971805913241, 1.365
097179169597, 1.3650971779353258, 1.3650971768637945, 1.3650971759335457, 1.3650971751259513, 1.365097174424
839, 1.365097173816169, 1.3650971732877528, 1.3650971728290087, 1.3650971724307503], 'length', 140)
[main_kalmanfilter-4] process has finished cleanly: 12.134759 / 12.442717
log file: /home/mingi/.ros/log/6047fb42-451b-11ea-8f23-94e6f72072db/main_kalmanfilter-4*.log
 [RUNNING]  Bag Time: 1580536246.940691   Duration: 12.434481 / 12.442717
Done.
[player-1] process has finished cleanly
log file: /home/mingi/.ros/log/6047fb42-451b-11ea-8f23-94e6f72072db/player-1*.log
^C[tf_publisher-5] killing on exit
[depthimage_to_laserscan-3] killing on exit
[republish-2] killing on exit
shutting down processing monitor...
... shutting down processing monitor complete
done
mingi@mingi-ThinkPad-X1-Extreme-2nd:~$
```

```
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581190.043
- Translation: [3.570, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581191.043
- Translation: [4.135, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581192.045
- Translation: [4.542, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581193.043
- Translation: [4.569, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581193.244
- Translation: [4.492, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581193.244
- Translation: [4.492, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581193.244
- Translation: [4.492, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581193.244
- Translation: [4.492, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
At time 1580581193.244
- Translation: [4.492, 0.000, 0.000]
- Rotation: in Quaternion [0.000, 0.000, 0.000, 1.000]
            in RPY (radian) [0.000, -0.000, 0.000]
            in RPY (degree) [0.000, -0.000, 0.000]
^Cmingi@mingi-ThinkPad-X1-Extreme-2nd:~$
```

Figure 4: Result of state estimates (left) and tf echo (right)

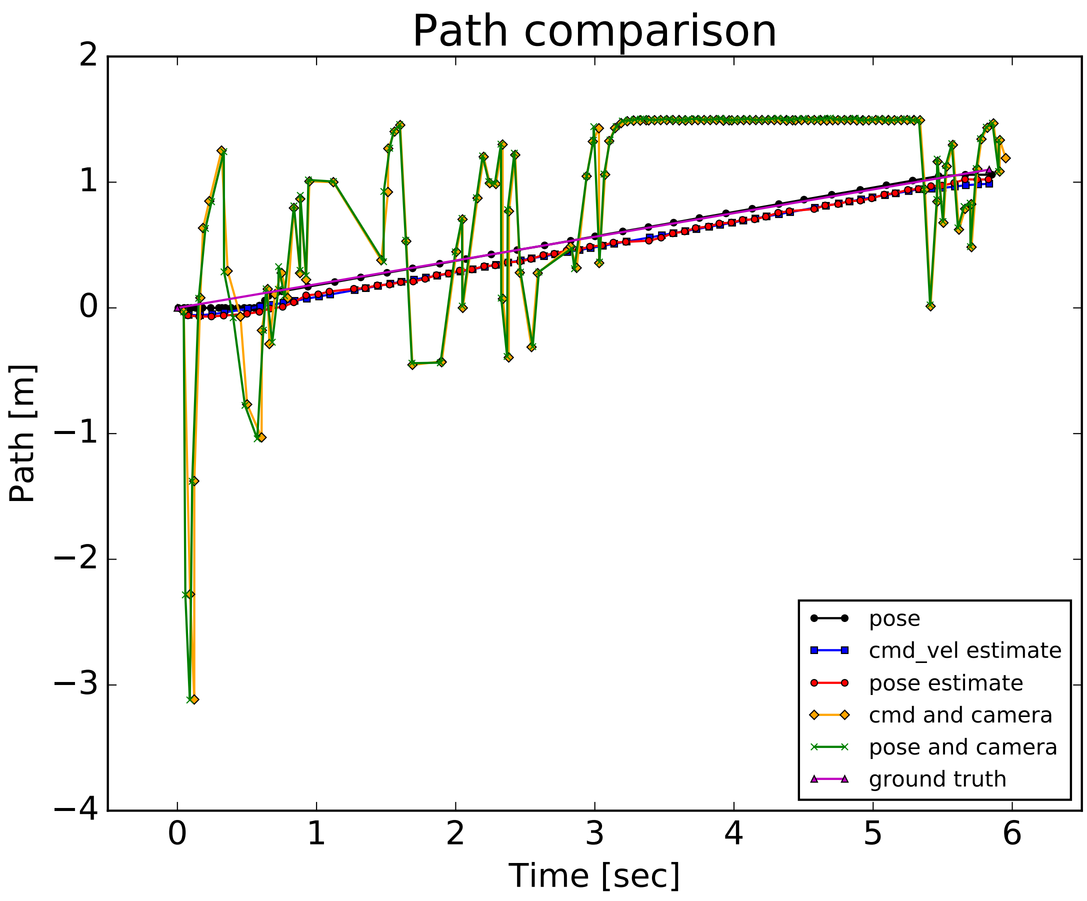


Figure 5: Task 2's result for path comparison as per each state estimate

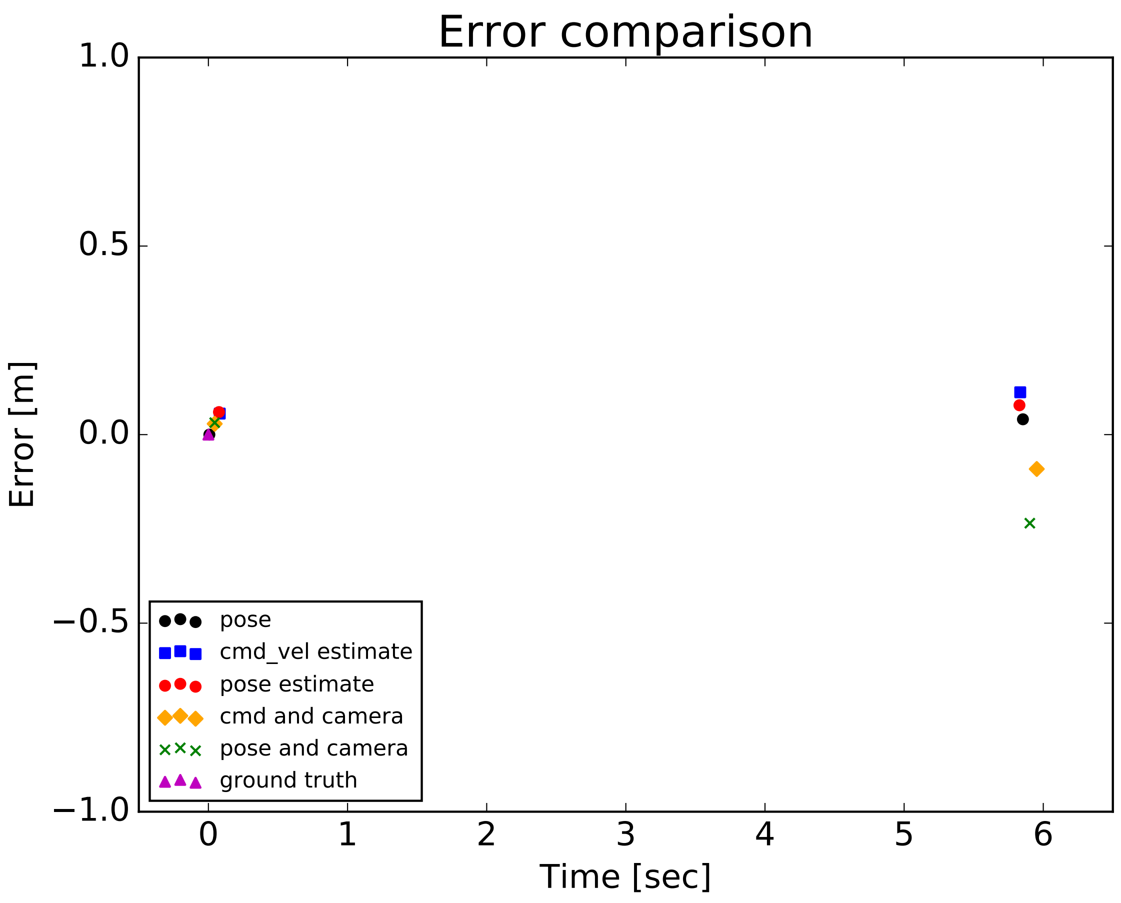


Figure 6: Task 3's result (start and end) for error comparison as per each state estimate

References

[1] Surrecio, A., Nunes, U., and Araujo, R. Fusion of odometry with magnetic sensors using kalman filters and augmented system models for mobile robot navigation. In *Proceedings of the IEEE International Symposium on Industrial Electronics, 2005. ISIE 2005.* (June 2005), vol. 4, pp. 1551–1556.

[2] Thrun, S. Probabilistic robotics. *Commun. ACM 45* (03 2002), 52–57.