A ROS implementation of a 6-DoF EKF for indoor drone Visual SLAM

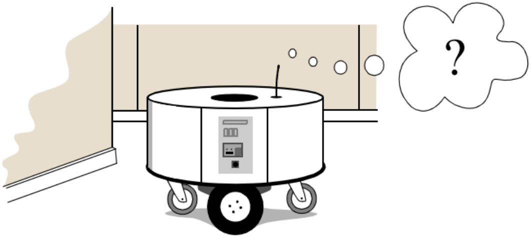
Diego Emanuel Avila

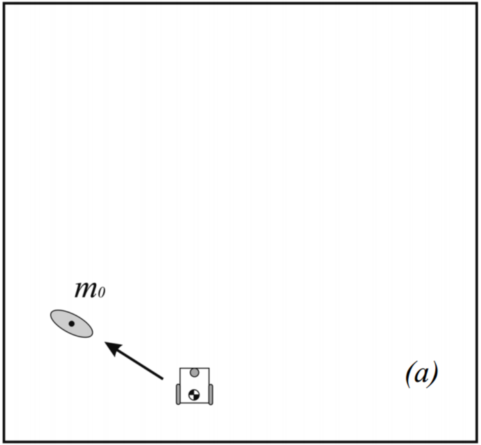
15 Dec. 2020

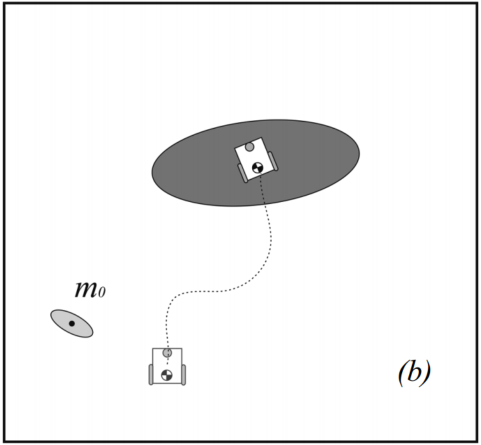
- Indoor environment
- Fully autonomous drone
- No GNSS nor Laser devices
- Inertial devices, range sensors, cameras and speed sensors are allowed
- Obstacles have at most 3mts height with passages of at least 1mt
- Landmarks: colored poles and QR markers
- 2-phase competition: inspection and path-follow

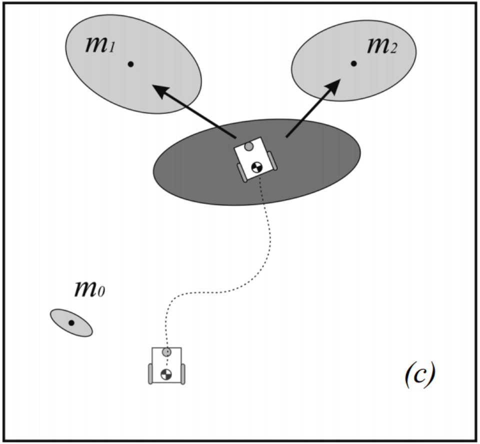
Simultaneous Localization & Mapping

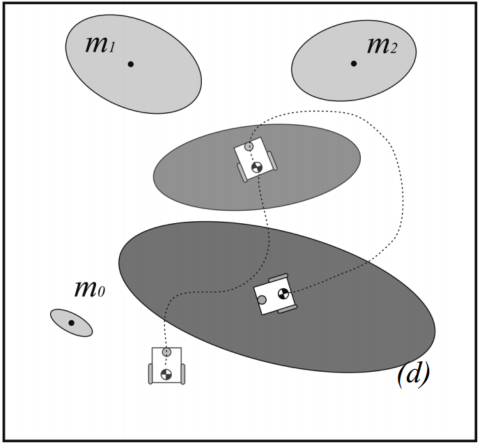
- Localization: where the robot is
- Mapping: build a map of the environment
- SLAM: localize itself while building a map

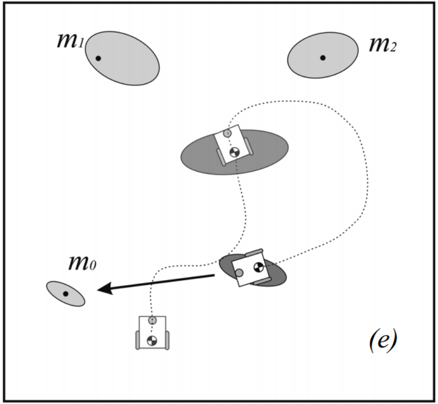



m_0
(a)


m_0
(b)


m_1
m_2
m_0
(c)


m_1
m_2
m_0
(d)


m_1
m_2
m_0
(e)

EKF-SLAM

- Motion and observation processes are usually not linear
- Motion model: $g\left(u_t, \mu_{t-1}\right)$
- Observation model: $h\left(\hat{\mu}_t\right)$
- EKF uses Taylor expansion to linearize the processes
- Builds feature-based maps
- The full-state (map) is
  $\mu = \begin{bmatrix} q_r & m_0 & \dots & m_{n-1} \end{bmatrix}^T$
- As in Kalman Filter, the process comprises 2 steps: prediction and correction

Characteristics

- PixHawk 4 Flight Controller
- 8 range finder
- 4 monocular cameras
- 2 stereo cameras: 1 pointing down, 1 pointing forward
- 1 optical-flow device
- Accelerometer + gyroscope
- Magnetometer
- Barometer

Control

- MAVROS node interfaces with the flight controller, and provides the control signal
- Control signal is composed by linear and angular velocities

$$u = \begin{bmatrix} v_x^b & v_y^b & v_z^b & \omega_x^b & \omega_y^b & \omega_z^b & \phi^b & \theta^b & \psi^b \end{bmatrix}^T$$

$$v^w = \mathbf{T} * \begin{bmatrix} v_x^b \\ v_y^b \\ v_z^b \end{bmatrix}$$

Motion model

- Given the linear velocities transformation v^w, the motion model can be summarized as follow

$$g\left(\mu_{t-1}, u_t\right) = \begin{cases} \begin{bmatrix} \mu_{t-1,x^w} \\ \mu_{t-1,y^w} \\ \mu_{t-1,z^w} \end{bmatrix} + \Delta t * v^w \\ \mu_{t-1,\psi} + \Delta t * \omega_z^b \end{cases}$$

- Noise in the motion process is assumed to be $\mathcal{N}\left(0, R_t\right)$

- 2 types of landmarks: Poles & Markers
  - Poles' observations is Range and Bearing
  - Markers' observations is the pose with respect to the camera reference frame
- Additionally, height correction is done using range sensor and 3D obstacle map
- Observation for landmark i is $z_i = h_i\left(x_t\right) + \mathcal{N}\left(0, Q_t\right)$

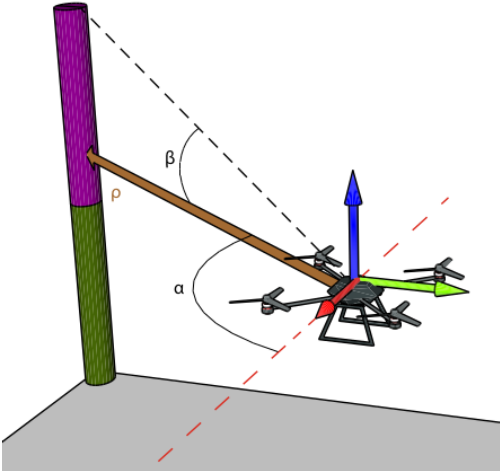
Poles

- Range and bearing information
  - ρ: distance
  - β: altitude
  - α: azimuth

Observation model:

$$\begin{bmatrix} p^b_{i,x} \\ p^b_{i,y} \\ p^b_{i,z} \end{bmatrix} = \boldsymbol{T}_{\boldsymbol{r}}^{-1} \begin{bmatrix} p^w_{i,x} \\ p^w_{i,y} \\ p^w_{i,z} \end{bmatrix}$$

$$h_i(\hat{\mu}_t) = \begin{bmatrix} p_{i,\rho} \\ p_{i,\alpha} \\ p_{i,\beta} \end{bmatrix} = \begin{bmatrix} \sqrt{p^2_{i,x^b} + p^2_{i,y^b}} \\ \text{atan2}\left(p^b_{i,y},\ p^b_{i,x}\right) \\ \text{atan2}\left(p^b_{i,z},\ p^b_{i,\rho}\right) \end{bmatrix}$$

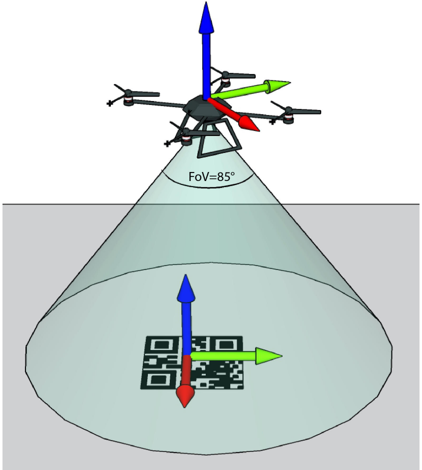

β
ρ
α

Markers

- Markers are unknown first, and added to the map once the drone sees them
- Observations is composed by its position with respect to the camera reference frame
- To add new markers to the state $h^{-1}(\hat{\mu}_t)$ should project the observation from the camera reference frame to the world reference frame

Observation model:

$$h_i(\hat{\mu}_t) = \begin{bmatrix} m^c_{i,x} \\ m^c_{i,y} \\ m^c_{i,z} \\ m^c_{i,\phi} \\ m^c_{i,\theta} \\ m^c_{i,\psi} \end{bmatrix} = (\boldsymbol{T_r} * \boldsymbol{T_c})^{-1} * \boldsymbol{T_m}$$


FoV=85°

Height

- The observations to correct the height are composed by the range sensor + Octomap measurement
- The height observation is $z = voxel_z + range_{distance}$
- Hence, the observation model is simply:

$$h\left(\hat{\mu}_t\right) = \hat{\mu}_z + bias$$

something

Normalized Estimation Error Squared

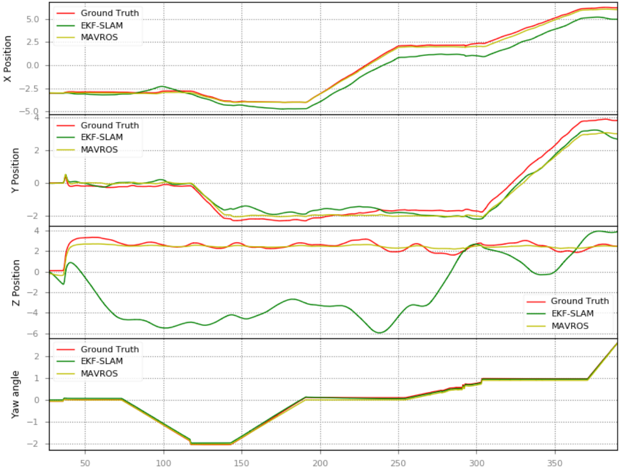
- Defined as:

$$NEES = (\boldsymbol{x} - \hat{\boldsymbol{x}})\, \Sigma^{-1}\, (\boldsymbol{x} - \hat{\boldsymbol{x}}) \leqslant \chi^2_{d,1-\alpha}$$

- Can be performed to check the consistency of the filter
- Consistency of the filter is maintained by using observations that satisfy the test

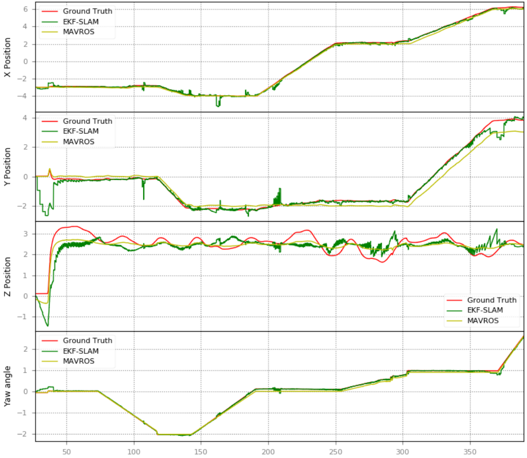
$$v^i = z^i_t - h^i\,(\hat{\mu}_t)$$
$$S = H^i_t \hat{\Sigma}_t H^{iT}_t + Q_t$$
$$D^2_i = v^i S^{-1} v^i \leqslant \chi^2_{d,1-\alpha}$$

- the importance of known poles and known and unknown markers in the localization and mapping process
- the accuracy of markers' pose estimation
- the importance of the NEES test to evaluate measurements and the filter's consistency



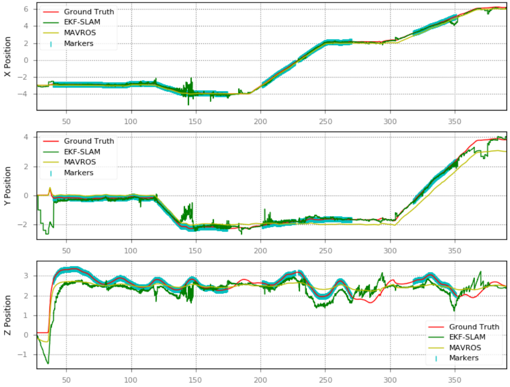
Real observation of poles

Experiment using *real observations* taken from the ROS bag.



SLAM with poles and markers

This experiment uses markers and poles to localize. It had the objective of understanding the importance of both poles and markers in the SLAM situation.



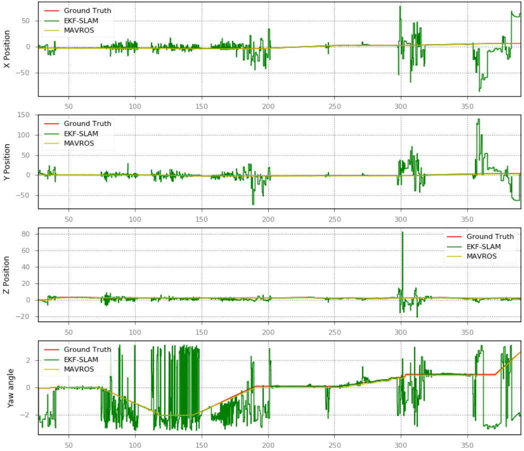
Distance between real and estimated poses.

Measures the distance between the true position of markers and the one estimated by the algorithm in a context of *real observations*, both for markers and poles.

Marker ID	Euclidean distance	ψ
0	0.076	0.034
1	0.186	0.052
4	0.119	0.055
5	0.155	0.013
Max.	0.186	0.055
Min.	0.076	0.013
Avg.	0.134	0.039

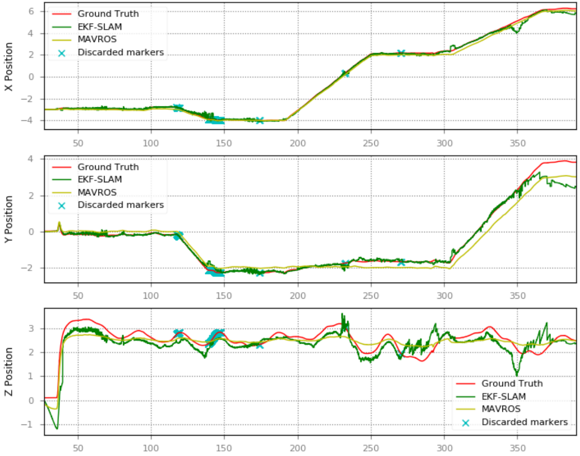
NEES test influence

What happen when no NEES test is used?



What about the χ^2_α threshold?

- So far, was used a threshold that corresponds to 95% of valid measurements. This means that we assume that the 95% of the observations are valid.
- This corresponds to $\alpha = 0.05$, but this value is the standard one.
- What happen with other confidence levels? for instance $\alpha = 0.9$



Conclusions

- The first 2 experiments showed the importance of poles and markers in the localization process.
- The markers' pose estimation in an SLAM situation is good enough for the competition context.
- NEES is fundamental for the consistency of the filter.
- Lower confidence of valid measurements implies discarding truly bad measurements, but on the other hand it can discard measurements that can be useful to correct the drone's pose.
- The value of α becomes a parameter of the system.

Future work

- Deploy and test the proposed implementation in the real drone.
- Fine-tune the noise covariance matrices.
- Extensive experimentation with Octomap and range sensor for height update.
- Camera self calibration procedure to improve the Z position estimation using poles and markers
- Compare the current algorithm with other algorithms like Error-State EKF-SLAM, or UKF-SLAM, and evaluate their performance with the current implementation as baseline

Thank you!

Questions?