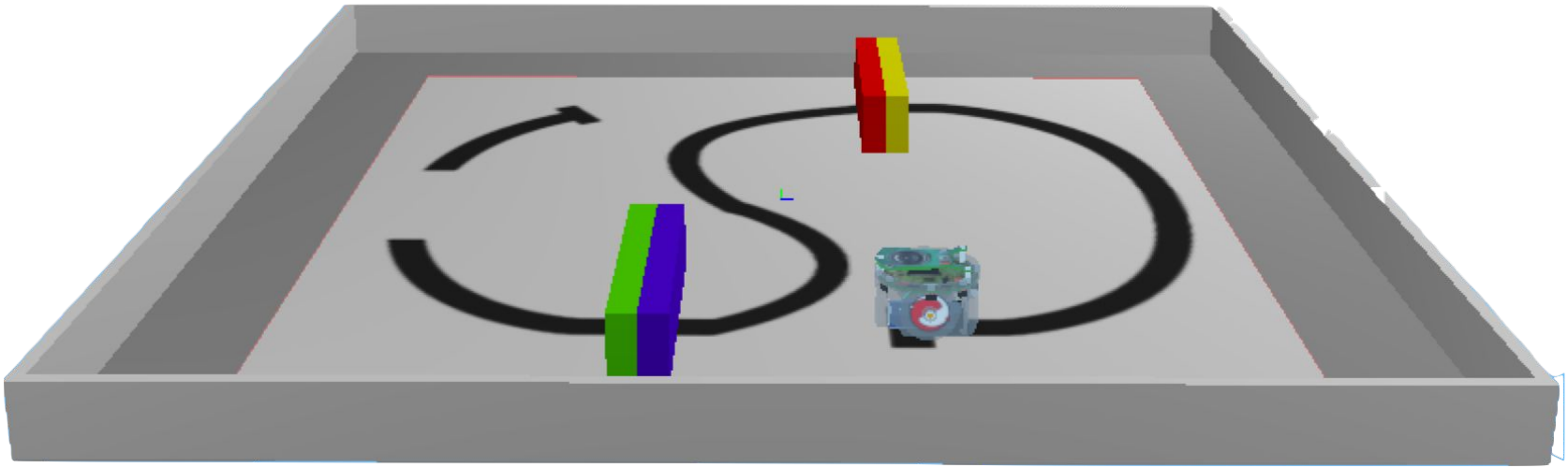
LINE FOLLOWING AND LOCALIZATION

Signals, Instruments and Systems
Course Project



BRENDOW Johanna, BREUILLOT Marie-Clémence. *Professor* : MARTINOLI Alcherio. *Supervisors* : Hugo Grall, Anwar Quraishi

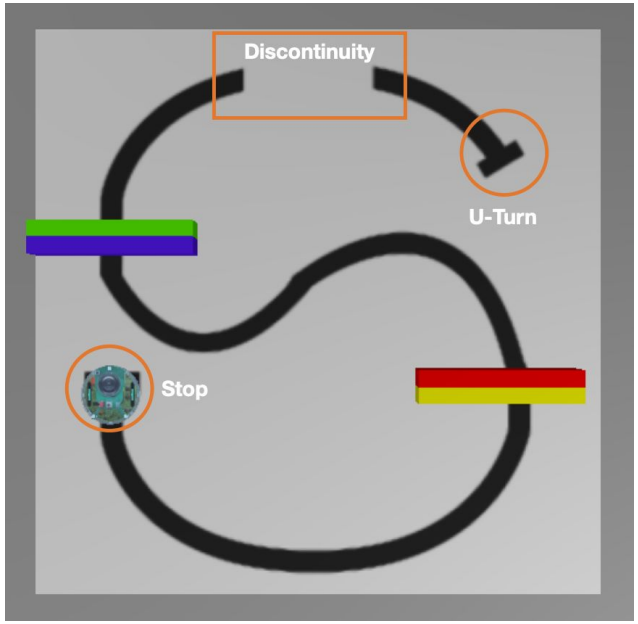
INTRODUCTION

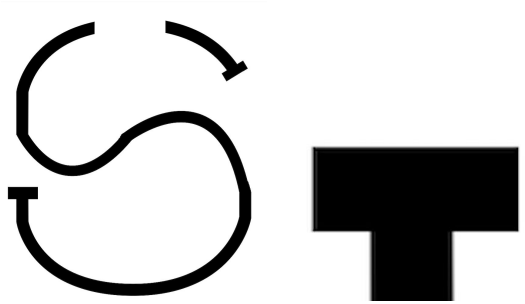
Main Goals

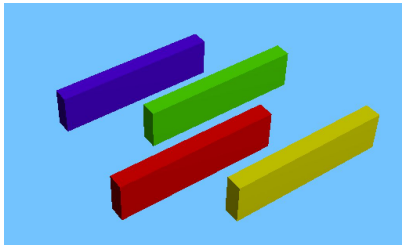
Line following
Obstacles avoidance
Localization - Odometry
Feature-based localization

Webots : C programming
Matlab

Main Set-Up







METHODS

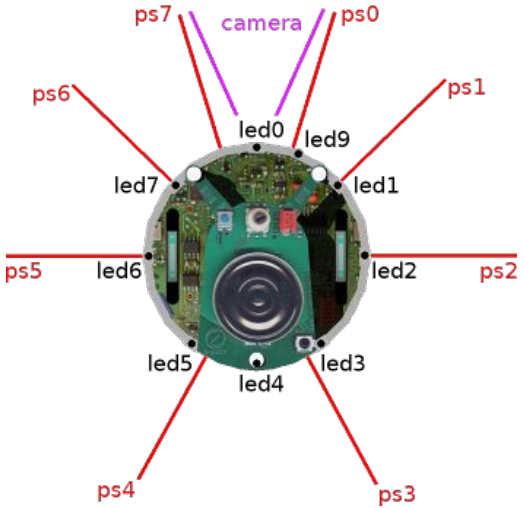
SENSORS

- Exteroceptive sensors :
- Camera
- Proximity IR sensors (8 sensors)

→ **line following and obstacle avoidance**

- Propioceptive sensors :
- Step counters
- Wheel encoders

→ **odometry**



ALGORITHM

1. **Check if obstacle nearby**

   → obstacles avoidance loop actioned

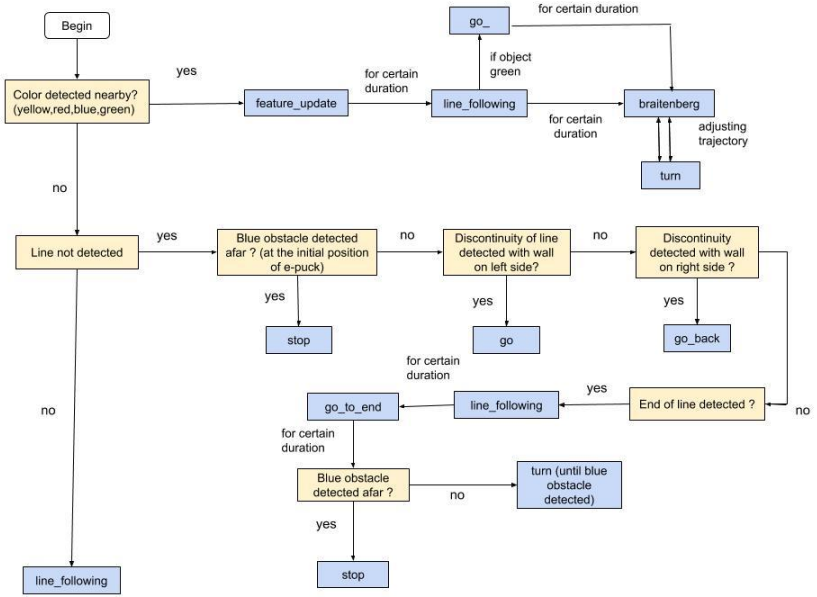
2. **Check if line not detected**

   → find if end of line or discontinuity

   using the camera

3. **Line following mode**

   → using the camera



FUNCTIONS

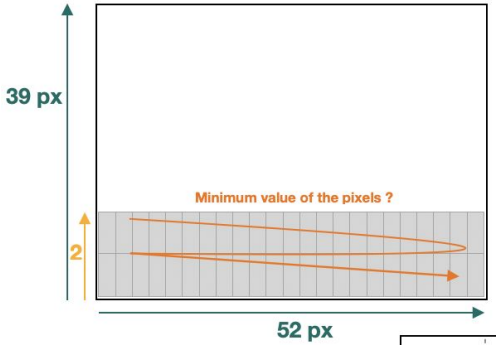
Principal Functions

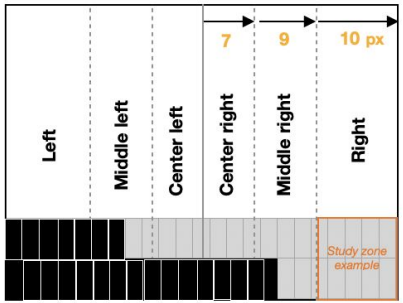
1. Line Following
2. Braitenberg : obstacles avoidance
3. Odometry : estimation of position
4. Feature Update + Kalman : improve estimation of position

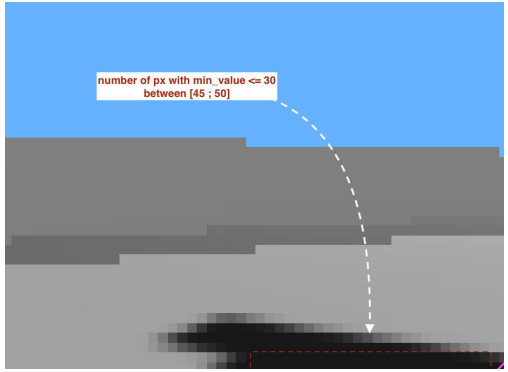
Auxiliary Functions

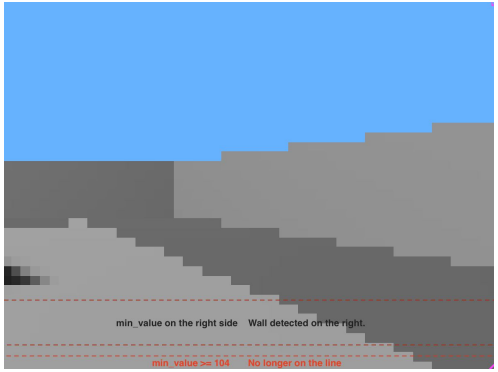
- Read camera image → get colors and gray levels
- Adjust the e-puck's trajectory + make the e-puck turn around and stop

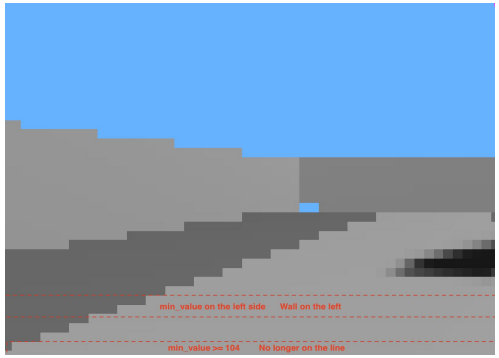
LINE FOLLOWING

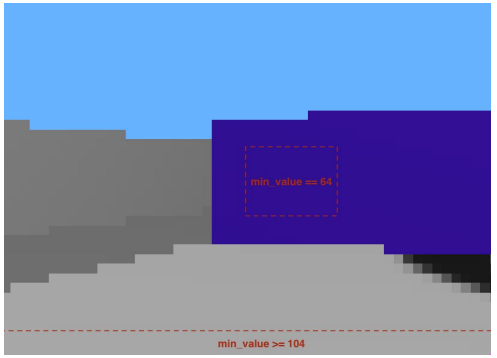












BRAITENBERG

(similar to lab)

- Use of the proximity IR sensors values
  → compute the speeds to avoid the obstacles

$$\text{speed}_{\text{left}} = \sum_{i=0}^{n} \alpha_{\text{left,i}} \left(1 - \frac{\text{ps\_value}_{\text{i}}}{\text{ps\_range}}\right)$$

$$\text{speed}_{\text{right}} = \sum_{i=0}^{n} \alpha_{\text{right,i}} \left(1 - \frac{\text{ps\_value}_{\text{i}}}{\text{ps\_range}}\right)$$

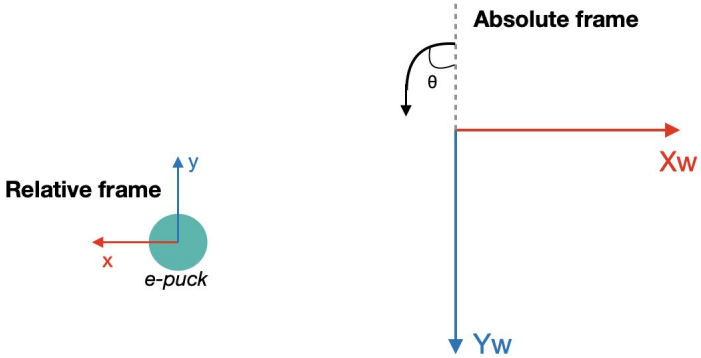
Braitenberg's coefficients (weights) for obstacle avoidance

Sensor	0	1	2	3	4	5	6	7
Left wheel	150	250	-25	-100	50	20	-125	-75
Right wheel	-75	-125	20	50	-100	-25	250	150

- Camera sees colored obstacle → Braitenberg

LOCALIZATION - ODOMETRY

Frames - Notations



Pose in the absolute frame = (Xw, Yw, θw) Pose in the relative frame = (x, y, θ)

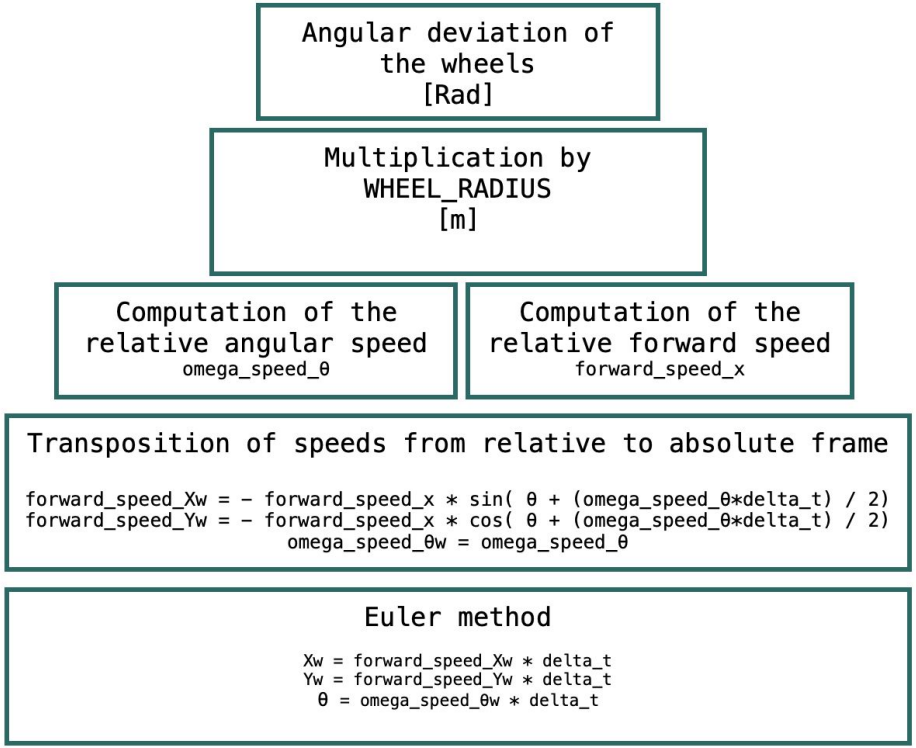
Initiale absolute pose : (-0.25, 0.10, π)

Model

$$\xi_I(T) = \xi_{I0} + \int_0^T \dot{\xi}_I dt = \xi_{I0} + \int_0^T R^{-1}(\theta)\dot{\xi}_R dt$$

ξ_{I0}	Initiale pose
$\xi_I(T)$	Absolute pose after a time T
$\dot{\xi}_R$	Relative speed
$\dot{\xi}_I$	Absolute speed

LOCALIZATION - ODOMETRY



FEATURE UPDATE

- Calculate position of e-puck using the objects
- **Principle :**
  - function IR_to_distance computes the distance to the obstacle
  - using informations on objects and an interpolation → get estimated position

	Red		Green		Blue		Yellow	
Distance [m]	IR [-]	std [-]	IR [-]	std [-]	IR [-]	std [-]	IR [-]	std [-]
0.08	30	23	-	-	-	-	-	-
0.07	33	28	-	-	-	-	29	21
0.06	66	28	-	-	-	-	51	24
0.05	109	28	-	-	-	-	84	25
0.04	212	40	27	23	30	23	156	23
0.03	452	54	49	25	40	26	305	31
0.02	1379	117	163	32	142	36	761	74
0.01	3096	211	969	133	690	84	2335	176

Obstacles	Position of center (x, z) [m]	Standard deviation of uncertainty in x [m]	Standard deviation of uncertainty in z [m]	Size (x,z) [m]
Green	-0.25, -0.1	0.005	0.005	0.2, 0.02
Blue	-0.25, -0.08	0.01	0.01	0.2, 0.02
Red	0.22,0.08	0.005	0.005	0.2, 0.02
Yellow	0.22, 0.1	0.02	0.02	0.2, 0.02

KALMAN FILTER

Inputs $\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$

Prediction

$$\overline{\mu}_t = A_t \mu_{t-1} + B_t u_t$$
$$\overline{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$$

Correction or update

$$K_t = \overline{\Sigma}_t C_t^T (C_t \overline{\Sigma}_t C_t^T + Q_t)^{-1}$$
$$\mu_t = \overline{\mu}_t + K_t (z_t - C_t \overline{\mu}_t)$$
$$\Sigma_t = (I - K_t C_t) \overline{\Sigma}_t$$

Outputs μ_t, Σ_t

RESULTS

SIMULATION VIDEO

(accelerated 8x)

SUCCESS RATE

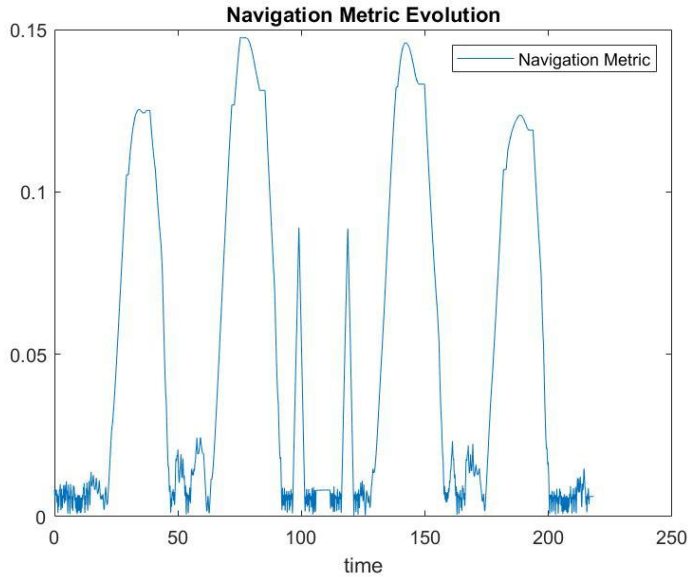
A simulation is considered successful if the e-puck accomplishes all the tasks.

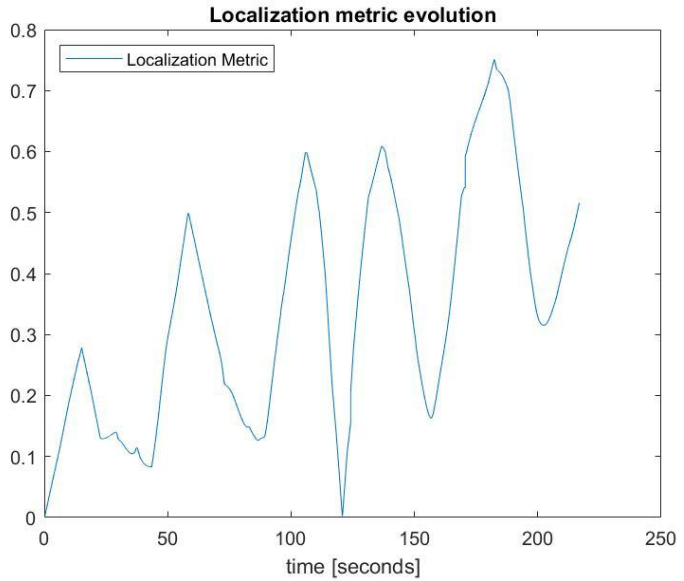
RandomSeed	Success Rate
0	100 %
-1	60 %

Problematic areas

- green obstacle
- line discontinuity

METRICS





Units = [m]

Metric evolution or Error evolution

METRICS

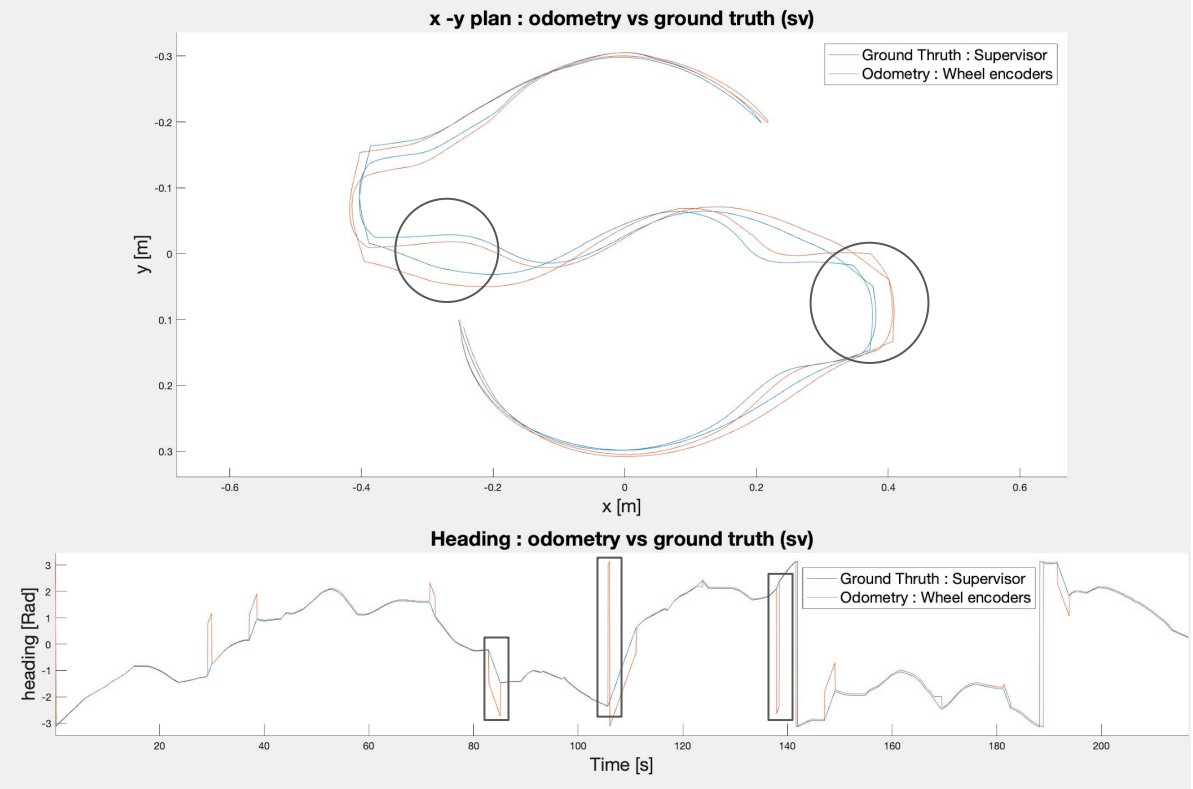
Average Metrics

Navigation metric	Time metric	Localization metric
0.0588	219.248	1.689

Units for navigation and localization = [m]

Units for time metric = [s]

ODOMETRY RESULT



Localization uncertainties

Deterministic errors :

Reduced by changes in

- WHEEL_RADIUS
- WHEEL_AXIS

Non-deterministic errors

Use of odometry model :

- Cumulative pose error
- Incrementally increase

POSITION ESTIMATION USING FEATURES

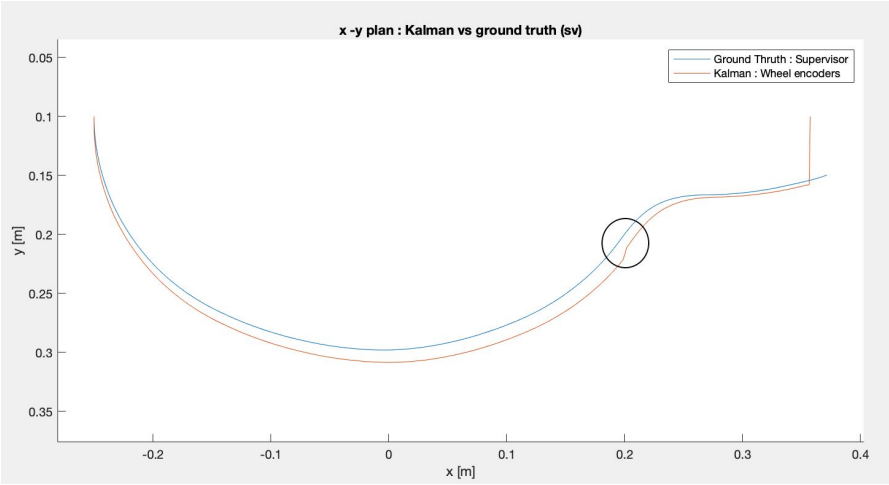
Using the feature_update function

object seen	x error	y error
yellow	0.0512	0.0695
blue	0.0726	0.0403
green	0.103	0.117
red	0.0804	0.0604

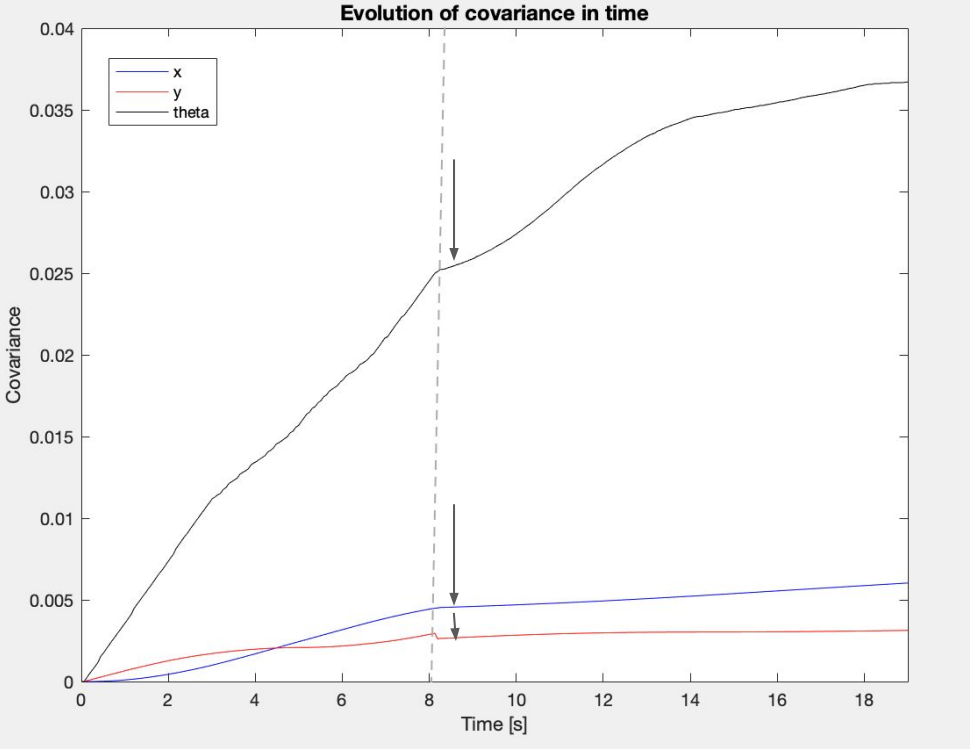
(results over one successful simulation : values did not change much between simulations)

KALMAN RESULT

Slightly better pose estimation



Solution to reduce remaining error =
Filter IR-values to better approximate the distance



CONCLUSION

CONCLUSION

- E-puck performs complete task most of the time
- **Improvements :**

  - Adjust the code to reduce sensitivity to noise
  - Completely implement Kalman → better odometry
  - Make the simulation work in other worlds