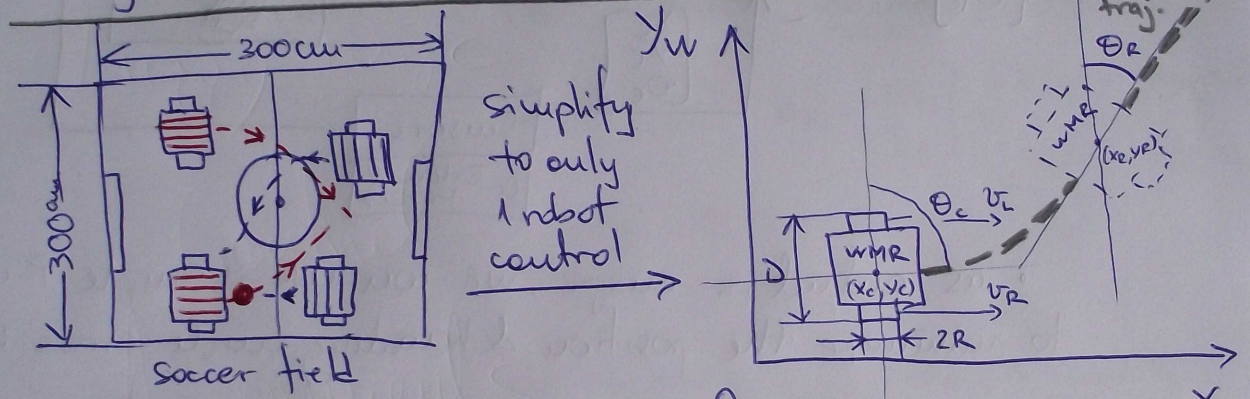
Mobile robot Fuzzy Logic Controller for Robot Soccer

Scenario:

⊕ in robot soccer each team of 2 robots has a strategy to score in opponent's team post

⊕ robots' movement is described by reference trajectories on the field which must be tracked by robots in a certain time (trajectory tracking)



- trajectory tracking ↓ robot should track a parametrized curve in a given time with minimal error in distance & angle

Robot kinematics

$$\begin{cases} v_{WMR} = \dfrac{(v_L + v_R)}{2}; \\ \omega_{WMR} = \dfrac{(v_L - v_R)}{D}; \end{cases} \qquad \begin{cases} \dot{x}_c = v_{WMR} \cdot \cos\theta_c; \\ \dot{y}_c = v_{WMR} \cdot \sin\theta_c; \\ \dot{\theta}_c = \omega_{WMR}; \end{cases}$$

The problem reduces in minimizing the mismatch between the current pose of the robot $[x_c, y_c, \theta_c]^T$ and the reference (given by the curve) $[x_R, y_R, \theta_R]^T$.

The errors used are given as:

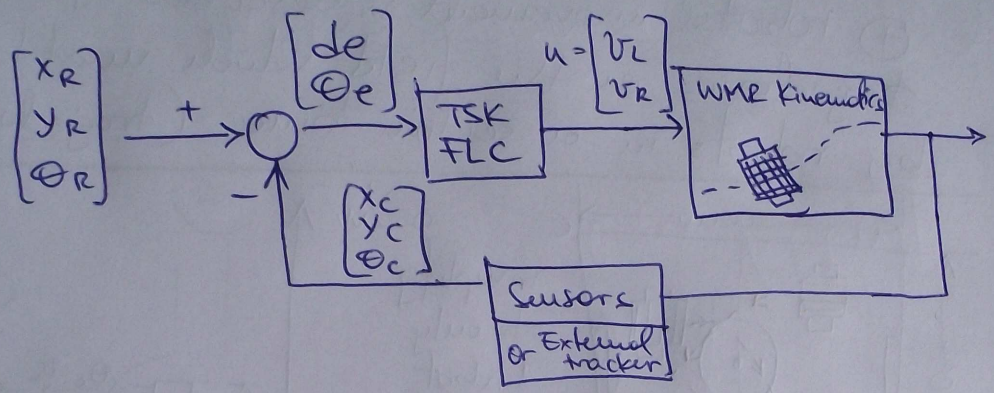
$$\begin{cases} \text{dist. error} \begin{cases} d_e = \sqrt{(x_R - x_c)^2 + (y_R - y_c)^2}; \end{cases} \\ \text{angle error.} \begin{cases} \theta_e = \theta_R - \theta_c; \end{cases} \end{cases}$$

2 Proposals for problemsolving:

① TSK (Takagi-Sugeno-Kang) Fuzzy Controller

Problem

- no need for defuzzification, output is already crisp (real-world value) and ready to be sent to robot actuators (e.g. voltage / velocity reference)
- control loop architecture:



- as output is crisp, we can use a simple P-controller to minimize the position & heading angle errors:

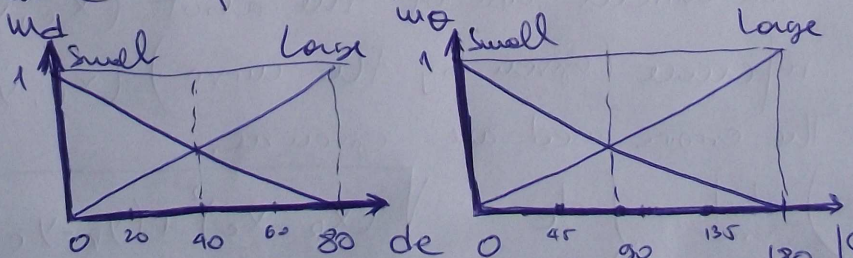
$$u = \begin{bmatrix} v_L \\ v_R \end{bmatrix} = \begin{bmatrix} K_d \cdot de + K_\theta \cdot \theta_e \\ K_d \cdot de - K_\theta \cdot \theta_e \end{bmatrix}$$

Fuzzy Controller Setup:

① Fuzzification:

- only the 2 inputs $[de, \theta_e]^T$ have membership values (are fuzzified)

e.g.:



sample mfs:

$$\mu_d^{small}(de) = \min\left(\max\left(0, \frac{de - 80}{-160}\right), 1\right)$$

$$\mu_d^{large}(de) = 1 - \mu_d^{small}(de)$$

$$\mu_\theta^{small}(\theta_e) = \frac{-|\theta_e| + 180}{180}$$

$$\mu_\theta^{large}(\theta_e) = 1 - \mu_\theta^{small}(\theta_e)$$

(2) Fuzzy rules:

- as we chose only 2 membership functions for each of the inputs we have 4 rules

e.g:

(R1): if de is small and $|\theta_e|$ is small

then $u_1 = \begin{bmatrix} v_{L_1} \\ v_{R_1} \end{bmatrix} = \begin{bmatrix} K_{d_1} \cdot de + K_{\theta_1} \cdot \theta_e \\ K_{d_1} \cdot de - K_{\theta_1} \cdot \theta_e \end{bmatrix}$

where e.g. $\begin{cases} K_{d_1} = 0.250; \\ K_{\theta_1} = 0.125; \end{cases}$

(R2): if de is small and $|\theta_e|$ is large

then $u_2 = \begin{bmatrix} v_{L_2} \\ v_{R_2} \end{bmatrix} = \begin{bmatrix} K_{d_2} de + K_{\theta_2} \cdot \theta_e \\ K_{d_2} \cdot de - K_{\theta_2} \cdot \theta_e \end{bmatrix}$

where e.g. $\begin{cases} K_{d_2} = 0.250; \\ K_{\theta_2} = 0.250; \end{cases}$

(R3): if de is large and $|\theta_e|$ is small

then $u_3 = \begin{bmatrix} v_{L_3} \\ v_{R_3} \end{bmatrix} = \begin{bmatrix} K_{d_3} \cdot d_3 + K_{\theta_3} \cdot \theta_3 \\ K_{d_3} \cdot d_3 - K_{\theta_3} \cdot \theta_3 \end{bmatrix}$

where e.g. $\begin{cases} K_{d_3} = 0.80; \\ K_{\theta_3} = 0.125; \end{cases}$

(R4): if de is large and $|\theta_e|$ is large

then $u_4 = \begin{bmatrix} v_{L_4} \\ v_{R_4} \end{bmatrix} = \begin{bmatrix} K_{d_4} \cdot de + K_{\theta_4} \cdot \theta_4 \\ K_{d_4} \cdot de - K_{\theta_4} \cdot \theta_4 \end{bmatrix}$

where e.g. $\begin{cases} K_{d_4} = 0.800; \\ K_{\theta_4} = 0.250; \end{cases}$

3. Output computation

- the output is just a weighted average of the individual rule outputs (only true for TSK)

$$u = \frac{\sum_{i=1}^{4} w_i \cdot u_i}{\sum_{i=1}^{4} w_i}$$

where w_i are the membership degrees computed using prod operation (as we have the and between the antecedents we can use min or prod when aggregating) such that:

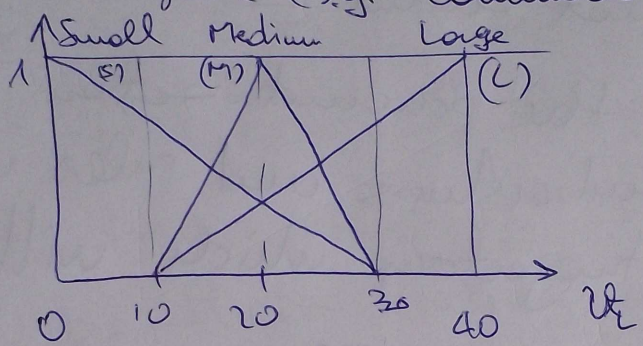
$$\begin{cases} w_1 = \mu_d^{\text{Small}} \cdot \mu_\theta^{\text{Small}} \\ w_2 = \mu_d^{\text{Small}} \cdot \mu_\theta^{\text{Large}} \\ w_3 = \mu_d^{\text{Large}} \cdot \mu_\theta^{\text{Small}} \\ w_4 = \mu_d^{\text{Large}} \cdot \mu_\theta^{\text{Large}} \end{cases}$$

2nd Problem

Mamdani Fuzzy Controller

- the difference between this and the first problem is that the output is now fuzzy (has a membership degree) and must be defuzzified considering both components $\begin{bmatrix} v_L \\ v_R \end{bmatrix}$

e.g.: add membership for output signal (e.g. command for robot)



similar for v_R

Rules become:

R1: if $\dot{e}$ is small and $|e|$ is small then v_L is S & v_R is S

R2: if $\dot{e}$ is small and $|e|$ is large then v_L is M & v_R is L

R3: if $\dot{e}$ is large and $|e|$ is small then v_L is L & v_R is M

R4: if $\dot{e}$ is large and $|e|$ is large then v_L is L & v_R is L

The defuzzification will happen using Center-Of-Gravity method:

$$\begin{cases} COG_{v_L} = \dfrac{\sum_{x=0}^{40} \mu_{v_L}(x) \cdot x}{\sum_{x=0}^{40} \mu_{v_L}(x)} \\ COG_{v_R} = \dfrac{\sum_{x=20}^{40} \mu_{v_R}(x) \cdot x}{\sum_{x=20}^{40} \mu_{v_R}(x)} \end{cases}$$

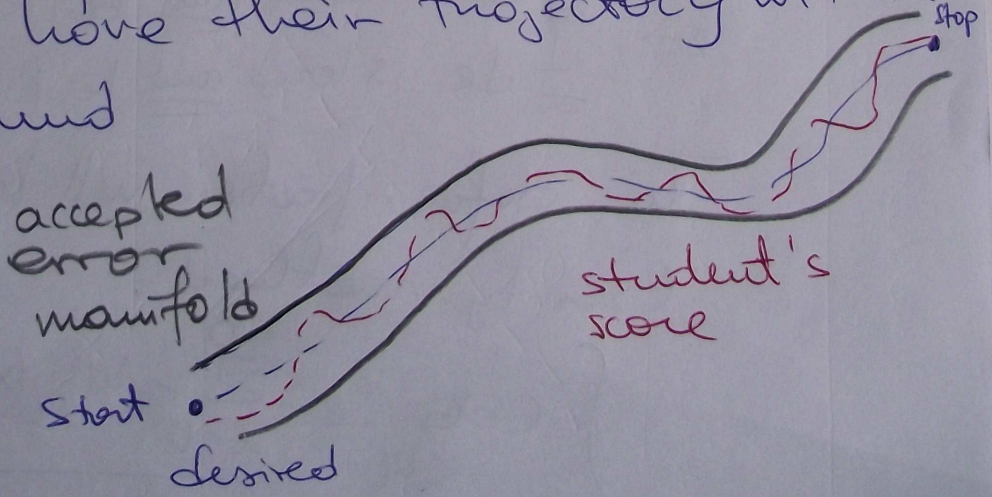
Students' tasks:

⊕ choose proper membership functions (e.g. shape and number for given bounds of inputs and/or output)

⊕ build the rules as the inputs (de, Θ_e) are given and we mention that output is a P-controller $\left(u=\begin{bmatrix} v_1 \\ v_n \end{bmatrix}\right)$

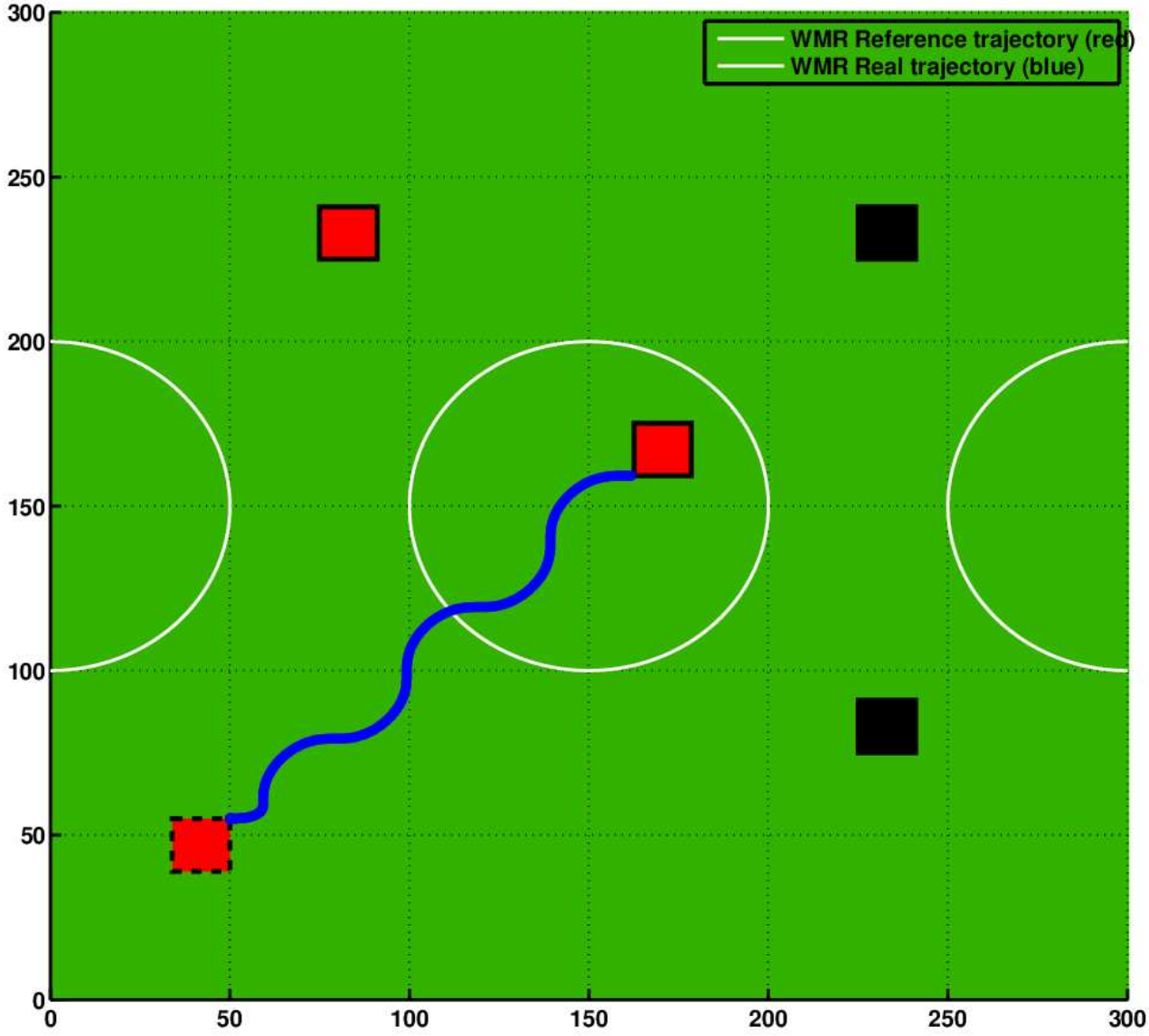
⊕ the parametrization of their memberships and rules will provide a trajectory which will try to approach the reference.

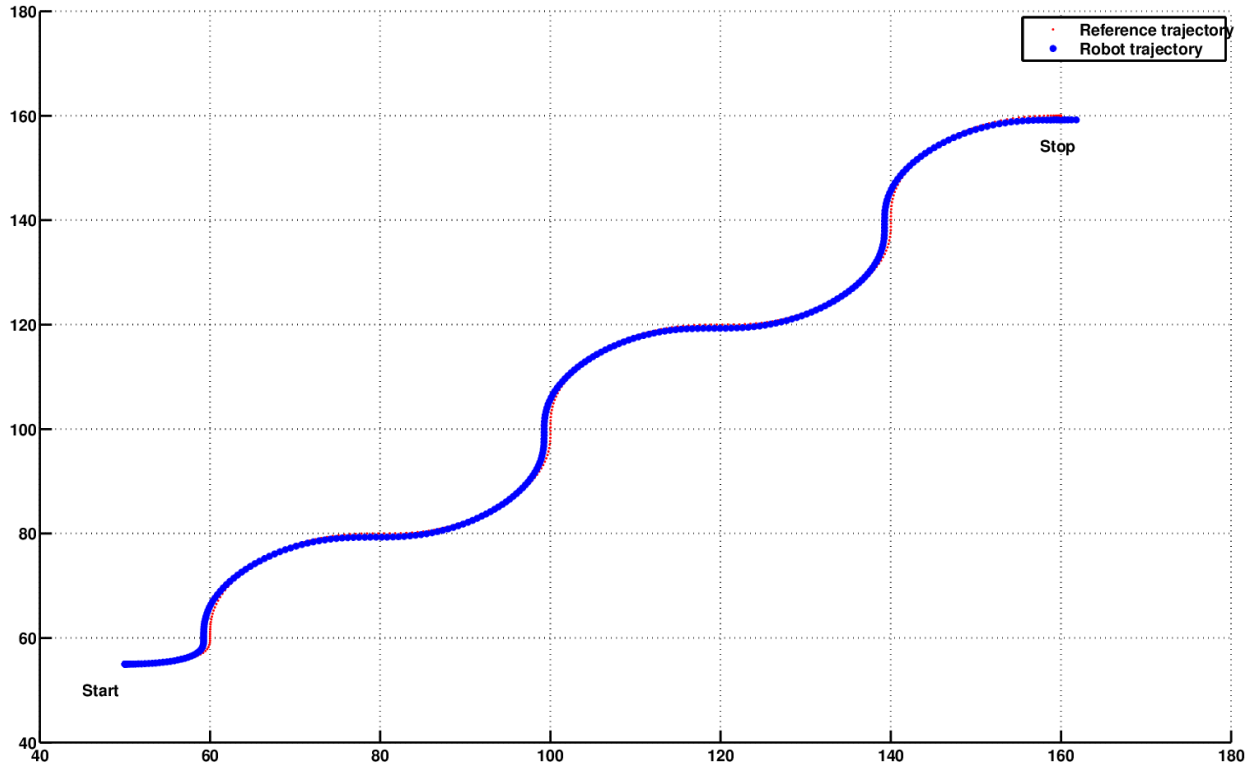
- We set a bound of accepted error on the reference and scorers will need to have their trajectory within the bound

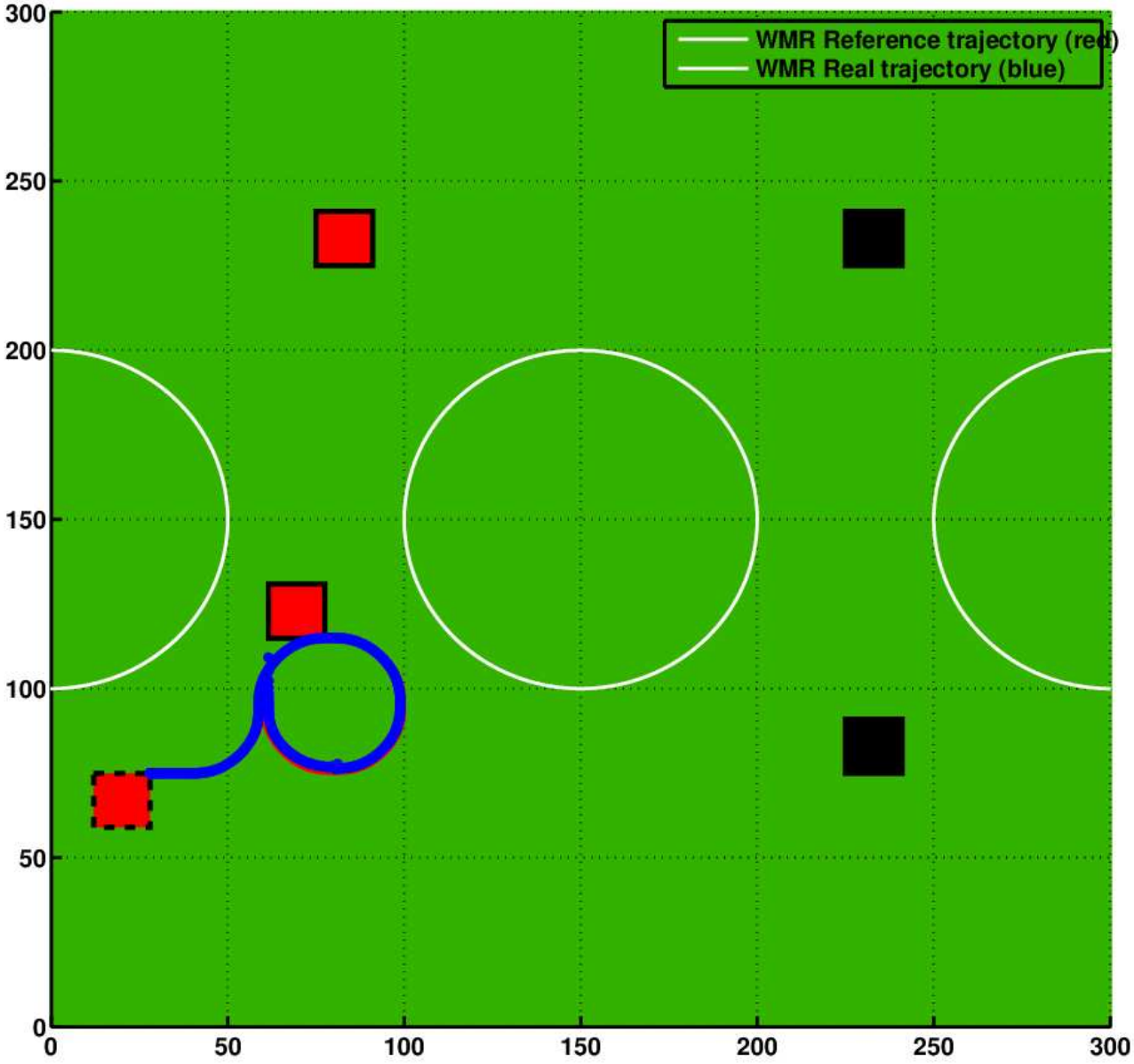


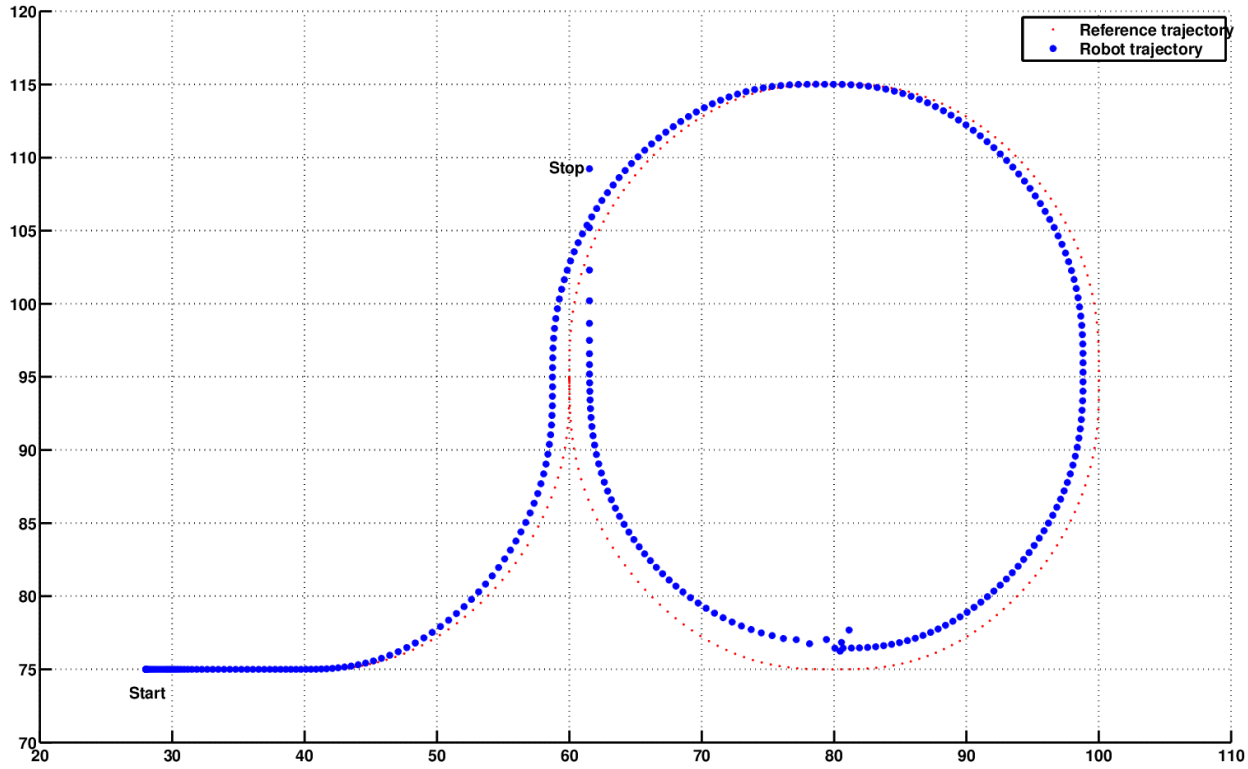
What we provide as support code:

- source file with all the needed tools:
  - 2/3 membership function types
  - 1 defuzzification method
  - robot kinematic model in the control loop.
  - rule processing depending on their linking elements (e.g. and, or)
  - parametrized curves as reference trajectories


WMR Reference trajectory (red)
WMR Real trajectory (blue)
300
250
200
150
100
50
0
0
50
100
150
200
250
300


Reference trajectory
Robot trajectory
Start
Stop
180
160
140
120
100
80
60
40
40
60
80
100
120
140
160
180


WMR Reference trajectory (red)
WMR Real trajectory (blue)
300
250
200
150
100
50
0
0
50
100
150
200
250
300


Reference trajectory
Robot trajectory
Stop
Start
120
115
110
105
100
95
90
85
80
75
70
20
30
40
50
60
70
80
90
100
110