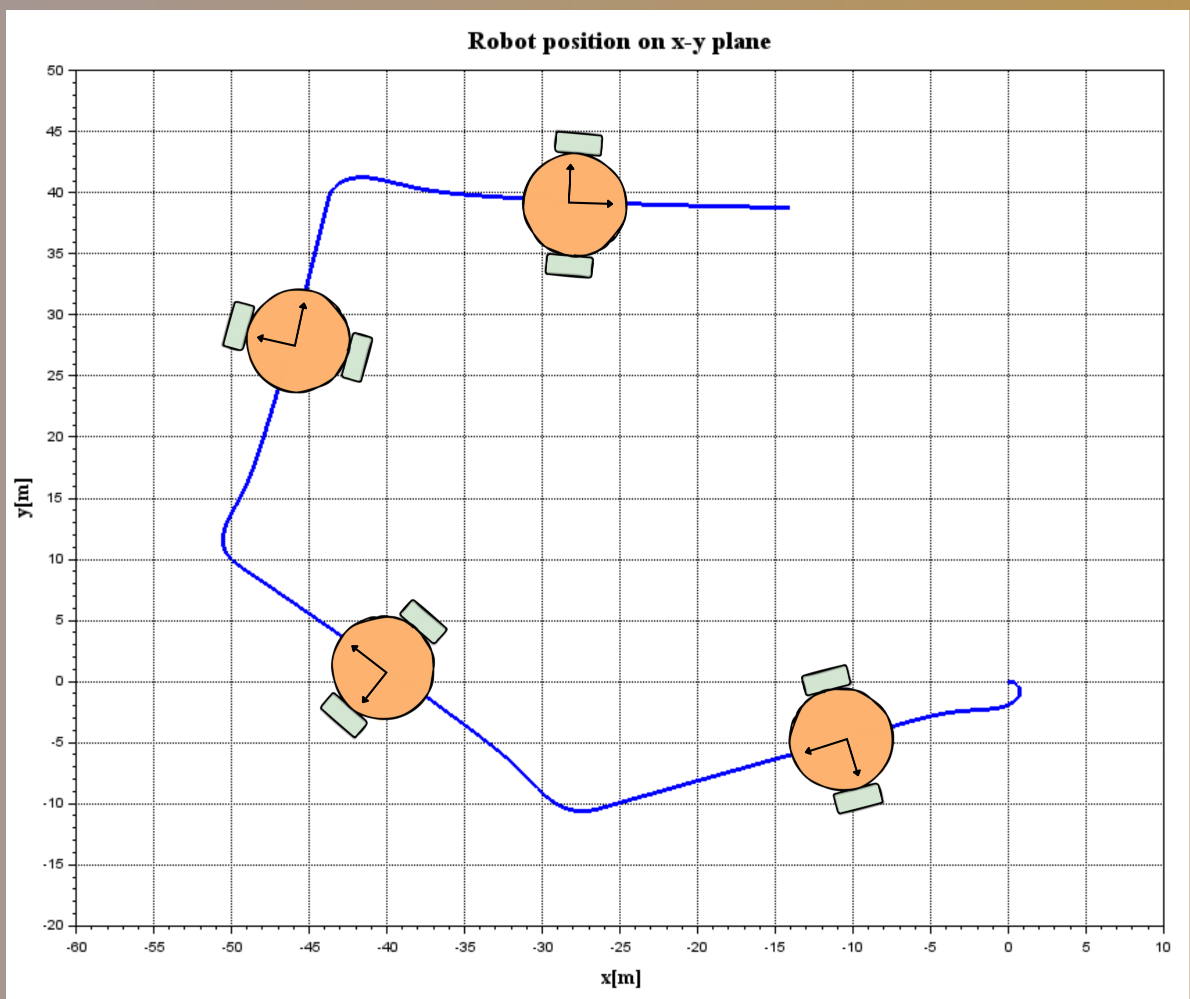
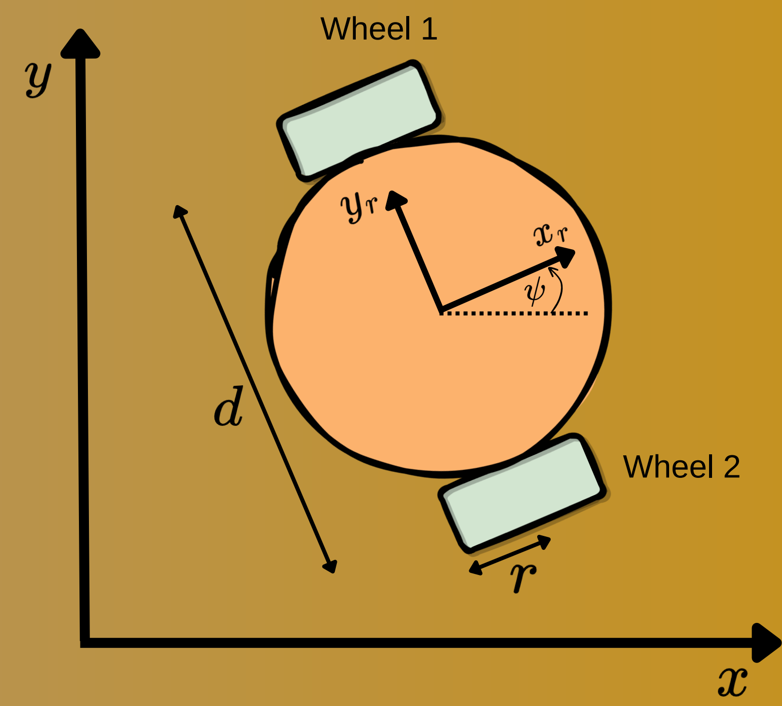
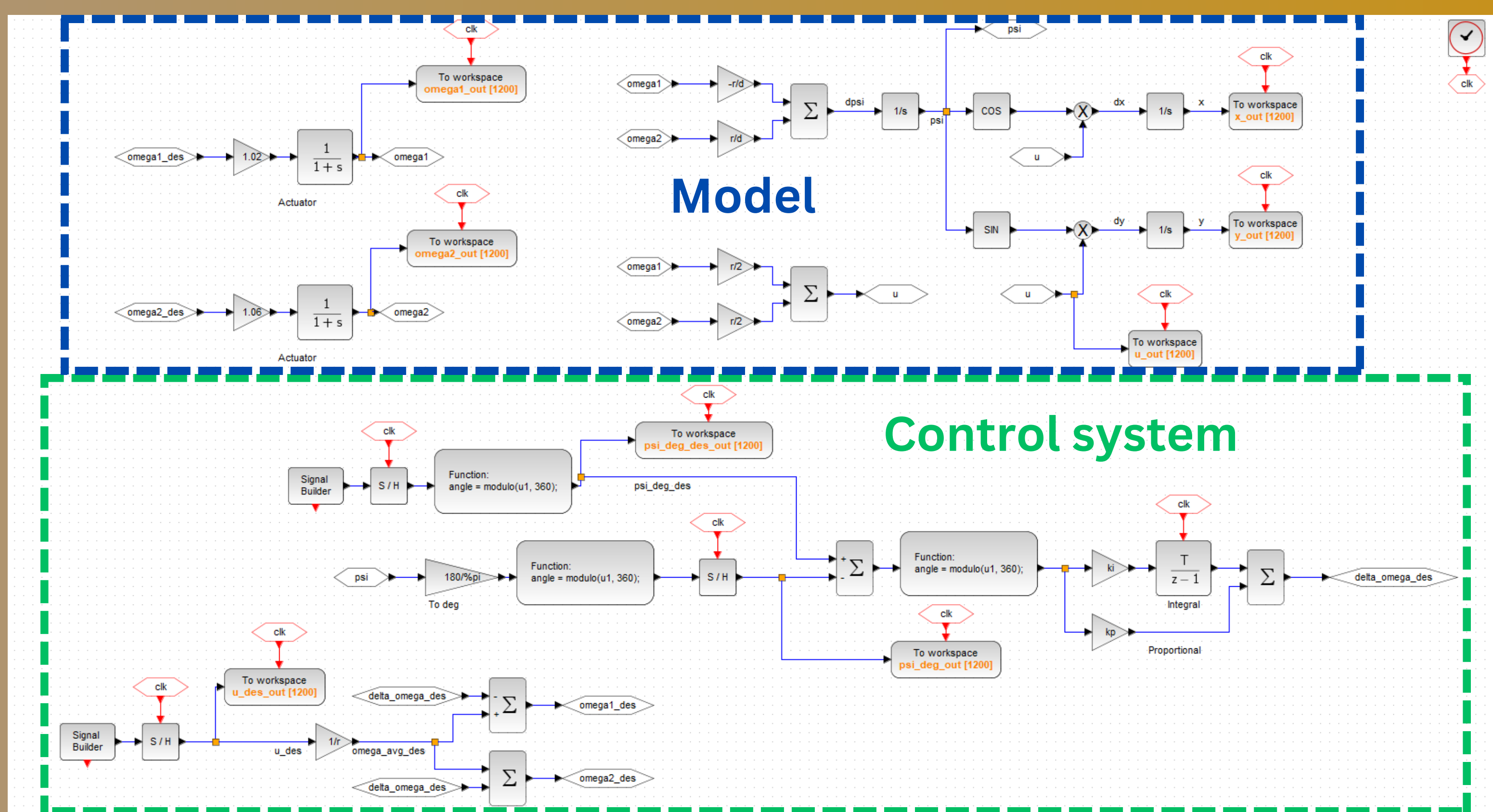
Wheeled robot control



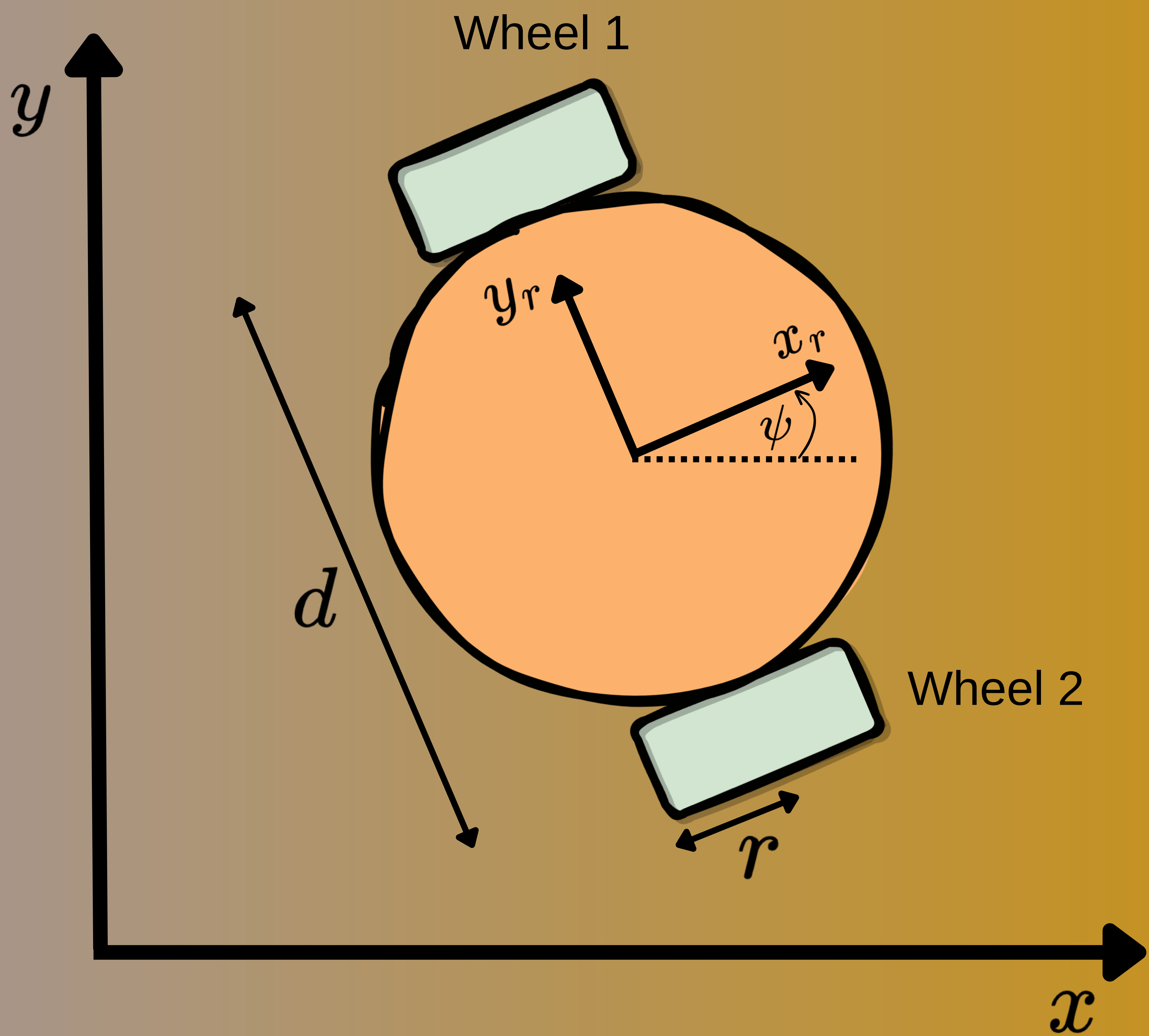




Simone Bertoni

Code
https://github.com/simorxb/Wheeled-robot-control

Robot description



Simone Bertoni

Differential equations of motion

Assuming a kinematic model where the two wheels can only move along x_r when they are spinning (i.e. they don't slip) and calling ω_1 and ω_2 the angular speed and u_1 and u_2 the linear speed of respectively wheel 1 and wheel 2, we have:

$$u_1 = \omega_1 r$$
$$u_2 = \omega_2 r$$

Let u and v be the linear speed of the centre of mass of the robot along x_r and y_r, then:

$$u = \omega_1 \frac{r}{2} + \omega_2 \frac{r}{2}$$
$$v = 0$$

And finally the differential equation of motion, where the state variables are $[x, y, \psi]$:

$$\dot{x} = u\cos(\psi)$$
$$\dot{y} = u\sin(\psi)$$
$$\dot{\psi} = \omega_2 \frac{r}{d} - \omega_1 \frac{r}{d}$$

To make the model more realistic we assume that each wheel's speed controller responds as a first order with transfer function $\frac{1}{s+1}$ and has an error factor respectively of 1.02 and 1.06.

Simone Bertoni

Control system

The control system assumes 2 setpoints (that could be from a user or from a path planner):

$\psi_{deg_{des}}$: yaw angle in degrees
u_{des}: linear speed in m/s

To control the linear speed we assume that we have no access to the actual speed measurement, therefore we simply use the knowledge of the system:

$$\omega_{avg_{des}} = \frac{u_{des}}{r}$$

To control the yaw angle we use a PI controller that outputs $\Delta\omega_{des}$.

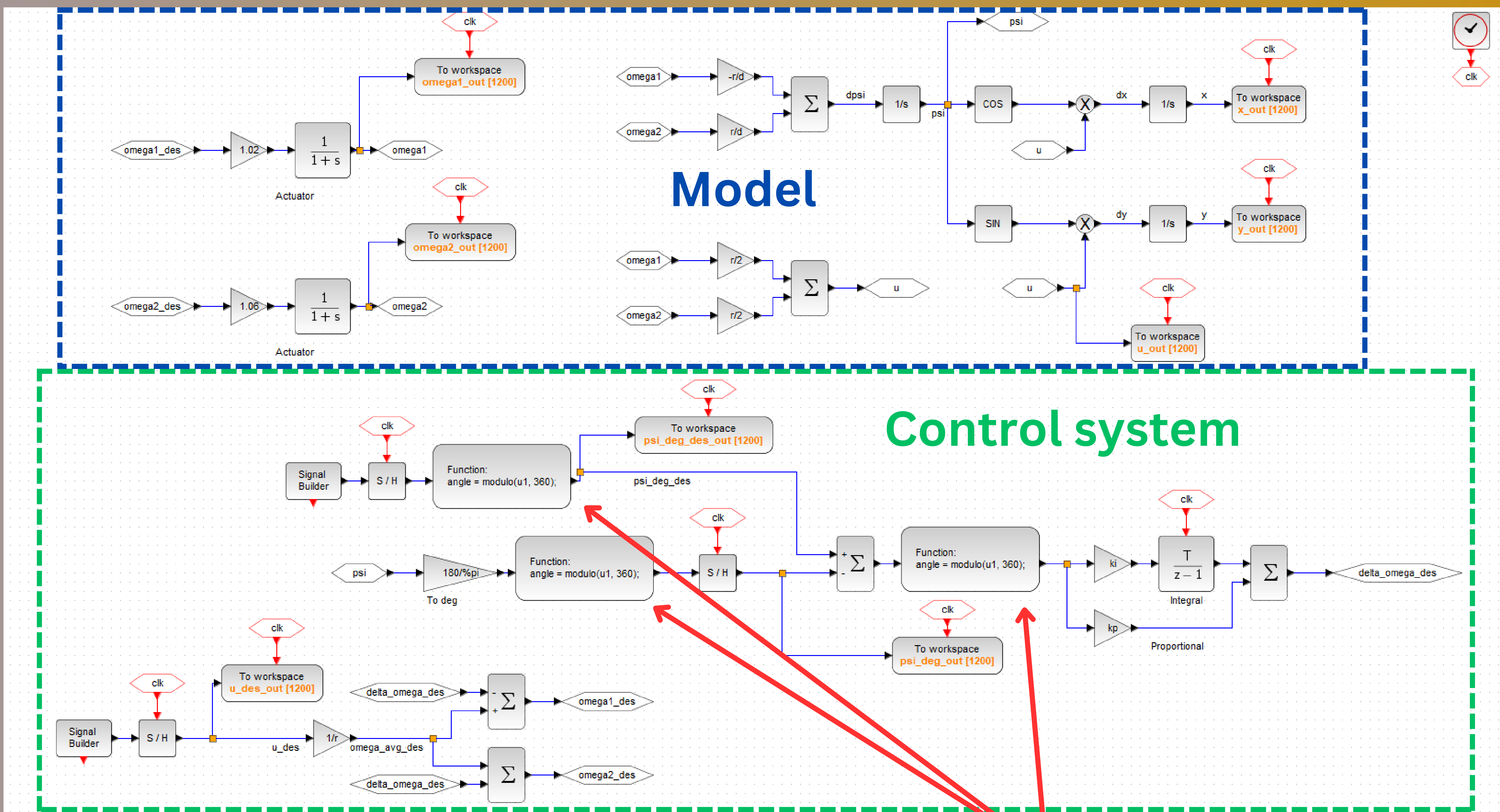
Then we have:

$$\omega_{1_{des}} = \omega_{avg_{des}} - \Delta\omega_{des}$$

$$\omega_{2_{des}} = \omega_{avg_{des}} + \Delta\omega_{des}$$

Simone Bertoni

Xcos model

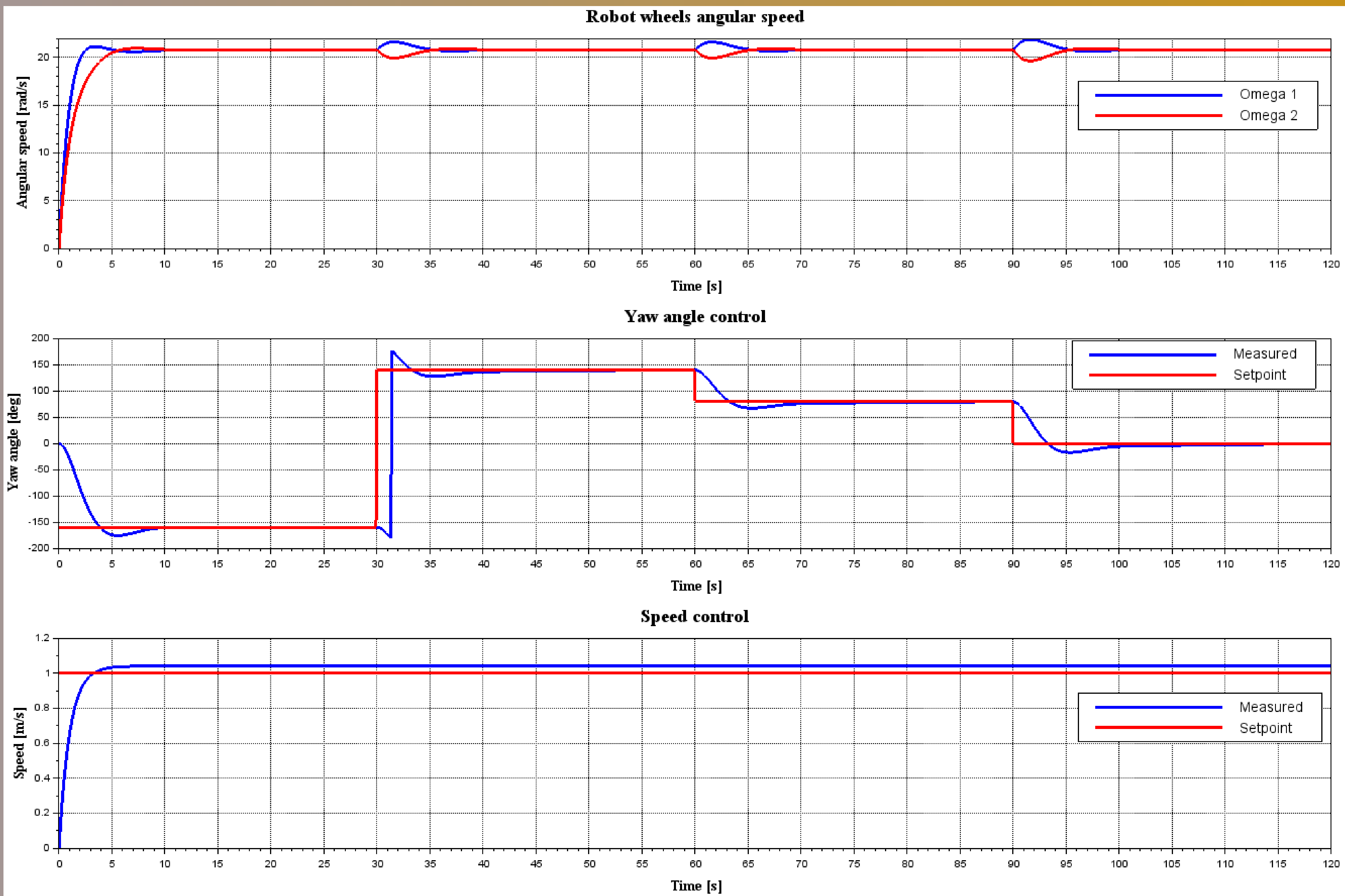



```
angle = modulo(u1, 360);

if angle > 180
        y1 = angle - 360;
elseif angle < -180
        y1 = angle + 360;
else
        y1 = angle;
end
```

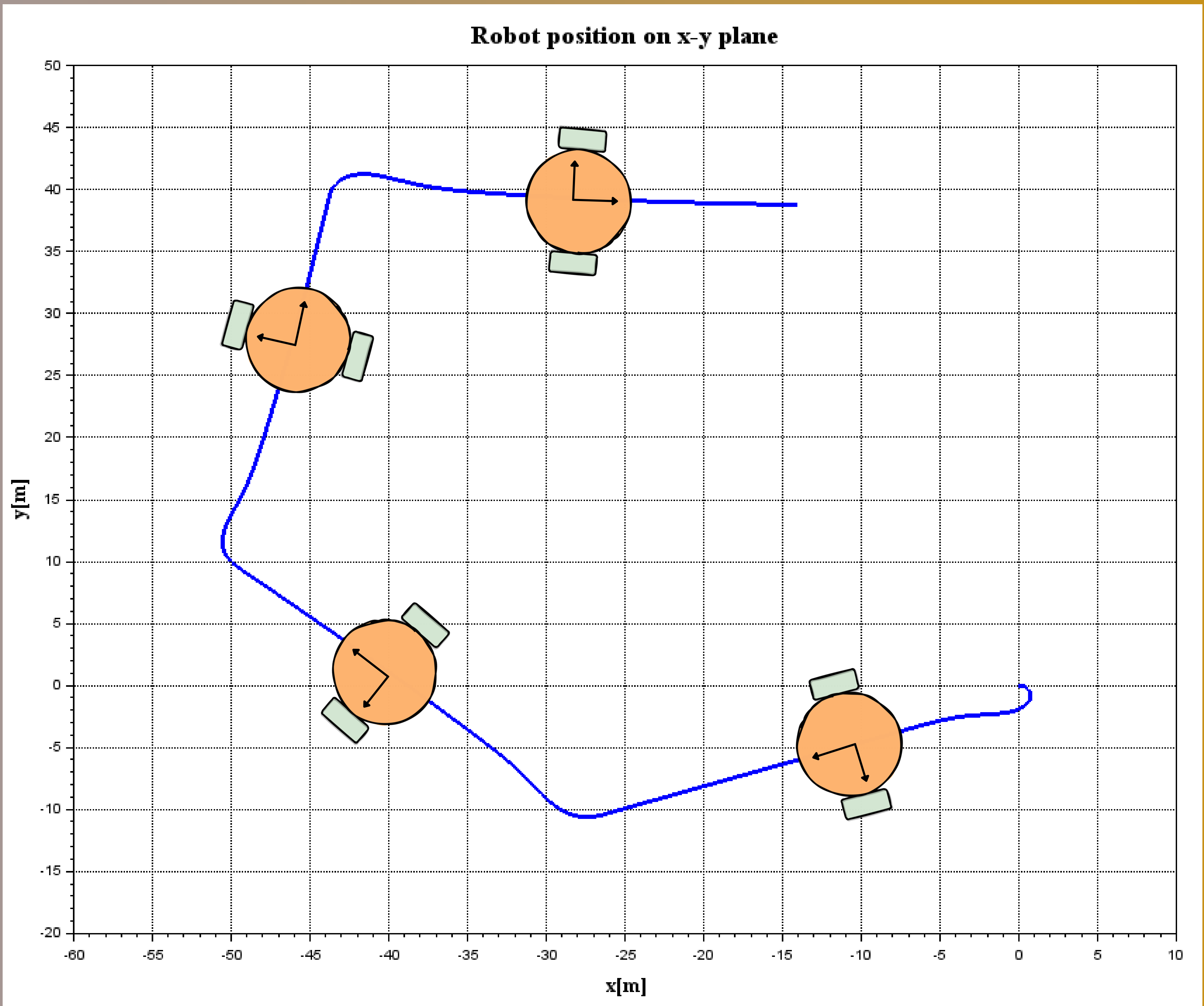
Simone Bertoni

Simulation – control system



Simone Bertoni

Simulation – robot position



Simone Bertoni