1 OpenLoop Mean Controller

1.1 *OpenLoop Controller for One Robot*

We want to control position and velocity of the robots and deciding about force for reaching our desired position and velocity. So our input is going to be accelerator($F = ma$). If we show acceleration by a, velocity by v and position in x coordinate by P_x and in y coordinate by P_y, we have the following equations

$$\begin{bmatrix} \dot{P}_x = v_x \\ \dot{v}_x = a_x \end{bmatrix} \tag{1}$$

$$\begin{bmatrix} \dot{P}_y = v_y \\ \dot{v}_y = a_y \end{bmatrix} \tag{2}$$

The state-space representation of our OpenLoop controller is:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y &= Cx(t) + Du(t) \end{aligned} \tag{3}$$

where *x(t)* represents our states, *u(t)* is our input and *e(t)* represents noise in the system. We also have *y* as our output.
First we assume that we don't have noise in the system and we also have just one robot. As mentioned, *a* is our input and *x, y,* and their velocities are our states that we want to control.
We define our states as following:

$$x_1 = P_x \tag{4}$$

$$\dot{x}_1 = x_2 = \dot{P}_x = v_x \tag{5}$$

$$x_3 = P_y \tag{6}$$

$$\dot{x}_3 = x_4 = \dot{P}_y = v_y \tag{7}$$

So our state space representation is:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} a \tag{8}$$

We want to find number of states that we can control. We need to know rank of the controllability matrix *{B, AB,A^2B, ... , A^{n-1}B}*.

$$C = \{B, AB, A^2B, ..., A^{n-1}B\} \tag{9}$$

$$C = \left\{ \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \right\} \tag{10}$$

So we have 2 controllable states. (One corresponds x axis and corresponds y axis.)

1.2 *Controlling More than One Robot*

we know that we can control velocity of our robot, we want to see what happens if we had more than one robot? As we saw here, v_x is completely independent of v_y, so if we wanted to have n robots, we had n states in x axis, and n states in y axis, which were completely independent from each other. So assume we have n robots and want to control them in *x* axis:

$$\dot{P}_{x1} = v_{x1} \tag{11}$$

$$\dot{v}_{x1} = a_{x1} \tag{12}$$

$$\dot{P}_{x2} = v_{x2} \tag{13}$$

$$\dot{v}_{x2} = a_{x2} \tag{14}$$

$$\vdots \tag{15}$$

$$\dot{P}_{xn} = v_{xn} \tag{16}$$

$$\dot{v}_{xn} = a_{xn} \tag{17}$$

So our state-space representation will be:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ . \\ . \\ . \\ \dot{x}_{2n-1} \\ \dot{x}_{2n} \end{bmatrix} = \begin{bmatrix} 0 & 1 & . & . & . & 0 & 0 \\ 0 & 0 & . & . & . & 0 & 0 \\ 0 & 0 & 0 & 1 & . & 0 & 0 \\ 0 & 0 & 0 & 0 & . & 0 & 0 \\ . & . & . & . & . & . & . \\ 0 & 0 & . & . & . & 0 & 1 \\ 0 & 0 & . & . & . & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_{2n-1} \\ x_{2n} \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ . \\ . \\ . \\ 0 \\ 1 \end{bmatrix} a_x \tag{18}$$

If we had n robots, we had exactly the same symmetry of what we had for 1 robot. We can again control two states because:

$$C = \left\{ \begin{bmatrix} 0 \\ 1 \\ . \\ . \\ . \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ . \\ . \\ . \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ . \\ . \\ . \\ 0 \\ 0 \end{bmatrix}, \ldots \right\} \tag{19}$$

1.3 Controlling Mean Position

So for any number of robots if we give a global command to them, we have just two states to control them in each axis. So it is obvious that we can not control position of all the robots, but what states are controllable? To answer this question we create a sensor that calculates average position and average velocity of the robots:

$$\begin{bmatrix} \dot{\bar{x}}_p \\ \dot{\bar{x}}_v \end{bmatrix} = \frac{1}{n} \begin{bmatrix} 0 & 1 & 0 & 1 & \ldots & 0 & 1 \\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ . \\ . \\ . \\ x_{2n-1} \\ x_{2n} \end{bmatrix} + \frac{1}{n} \begin{bmatrix} 0 & 0 & 0 & 0 & \ldots & 0 & 0 \\ 0 & 1 & 0 & 1 & \ldots & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ . \\ . \\ . \\ 0 \\ 1 \end{bmatrix} a_x \tag{20}$$

Thus:

$$\begin{bmatrix} \dot{\bar{x}}_p \\ \dot{\bar{x}}_v \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \bar{x}_p \\ \bar{x}_v \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} a \tag{21}$$

We analyze C for y:

$$C = \left\{ \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix} \right\} \tag{22}$$

This matrix has rank two, and thus each state is controllable. These controllable states are average position and average velocity:

$$a = K \left[\begin{bmatrix} \dot{\bar{x}}_p \\ \dot{\bar{x}}_v \end{bmatrix} - \begin{bmatrix} x_{goal} \\ \dot{x}_{goal} \end{bmatrix} \right] \tag{23}$$

■

1.4 Applying Noise

Real systems, especially at the micro scale, are affected by unmodelled dynamics called Brownian noise. To model this equation (3) must be modified to following:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) + We(t) \\ y &= Cx(t) + Du(t) \end{aligned} \tag{24}$$

where e(t) is the error in the system.
After some time, gaussian distribution shapes the outline of the robots because of the brownian noise feature:

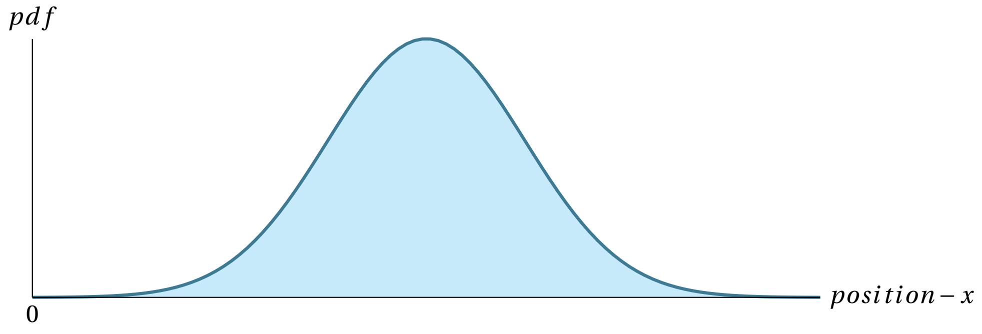
$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2 / 2\sigma^2} \tag{25}$$

where σ is standard deviation:

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \overline{x})^2} \tag{26}$$

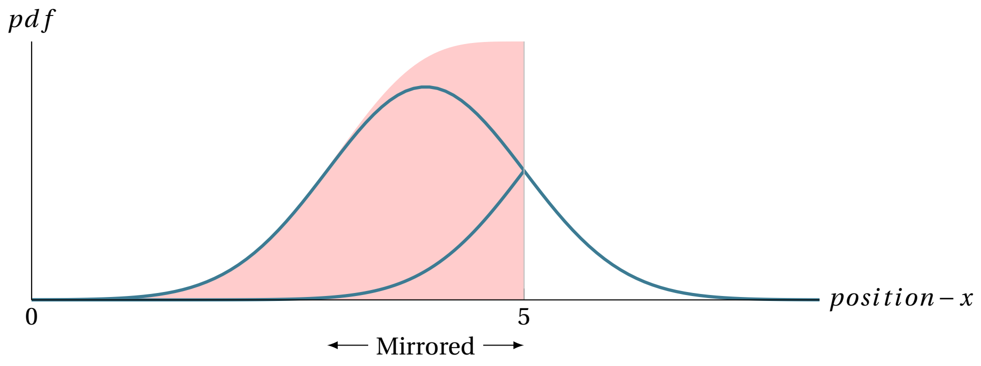
After a while, if we have lots of robots, we see the robots to make a bell shaped curve because of the Brownian noise. In we see the probability density function of a Gaussian distribution:

$$f(x) \sim N(\mu, \sigma) \tag{27}$$



Probability of the position of the robots, if we have Gaussian noise.

If robots face a wall, we expect to have the following shape:


pdf
$position - x$
Mirrored