Robot Control

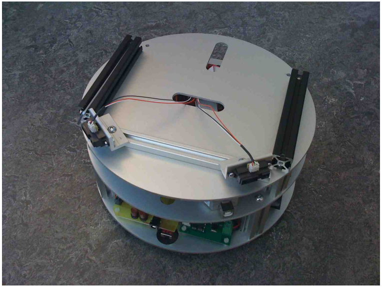
Ulf Holmberg

September 25, 2009

Robot Control

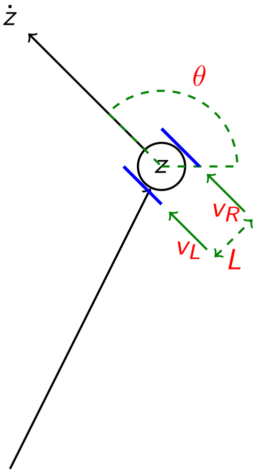
Robot Control

Components



- Main processor for control of robot
- Local processor with speed control of wheels
- PC (matlab) for analysis of camera picture
- Wireless radio connection PC-robot

Dynamical model



$$\begin{cases} \dot{\theta} = u \\ \dot{z} = ve^{i\theta} \end{cases}$$

- θ moving direction of robot
- z complex position of robot
- u angular velocity
- v speed

Control of wheel velocity

$$\begin{matrix} v_R = v + \frac{L}{2}u \\ v_L = v - \frac{L}{2}u \end{matrix} \Rightarrow \dot{\theta} = (v_R - v_L)/L = u$$

Discretized model

Approximate discretized model

$$\begin{aligned} \Delta\theta(k) &= \theta(k) - \theta(k-1) = u(k) \\ \Delta z(k) &= z(k) - z(k-1) = v(k)e^{i\theta(k)} \end{aligned}$$

where k is present sampling instant and $k-1$ previous

Robot Control

Calibration of velocity

Wheel speed setpoints $w_R = w_L = w$: [0,255]
Calibration model

$$w = k_v v + 128$$

Choose w and measure v [pixels/sample]

$$\rightarrow k_v = (w - 128)/v$$

Calibration of angular velocity

Turn around $v = 0$

$$
\begin{aligned}
v_R &= v + \tfrac{L}{2}u = \tfrac{L}{2}u \\
v_L &= v - \tfrac{L}{2}u = -\tfrac{L}{2}u \\
\Delta\theta &= (v_R - v_L)/L = u
\end{aligned}
$$

Choose motor setpoints and count samples T for n revolutions

$$
\begin{array}{l} w_R = 128 + w \\ w_L = 128 - w \end{array} \rightarrow \sum_{k=1}^{T} \Delta\theta(k) = Tu = 2\pi n \Rightarrow u = \frac{2\pi}{T} n
$$

Then $w = k_v v_R = k_v \frac{L}{2} u$

$$
\rightarrow L = \frac{2w}{k_v u} = \frac{wT}{k_v \pi n}
$$

Alternative calibration strategy

Turn around the robot and measure L from camera [pixels]
Perform turning experiment as before n revolutions
Calculate scale factor k_v from

$$k_v = \frac{wT}{L\pi n}$$

Conversion from control variables to wheel setpoints

- Control variables u and v for robot motion control design
- Wheel setpoints w_R and w_L (sent to local controller)

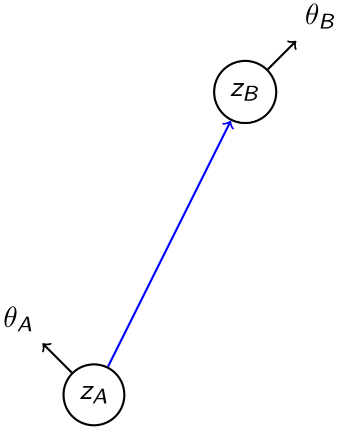
$$v_R = v + \frac{L}{2}u$$
$$v_L = v - \frac{L}{2}u$$
$$w_R = k_v \cdot v_R + 128$$
$$w_L = k_v \cdot v_L + 128$$

where k_v and L are found experimentally

Robot Control

Line path

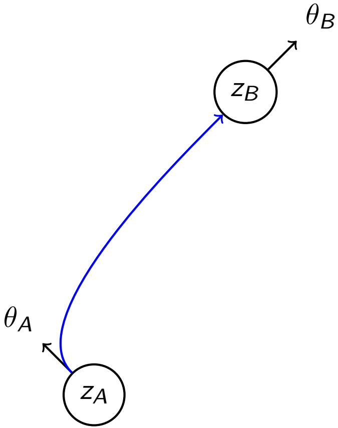
Objective: go from position A: (z_A, θ_A) to B: (z_B, θ_B)



- Start by turning to right direction ($v = 0,\ u \neq 0$)
- Go straight from A to B ($v \neq 0,\ u = 0$)
- End by turning to right direction ($v = 0,\ u \neq 0$)

Smooth path

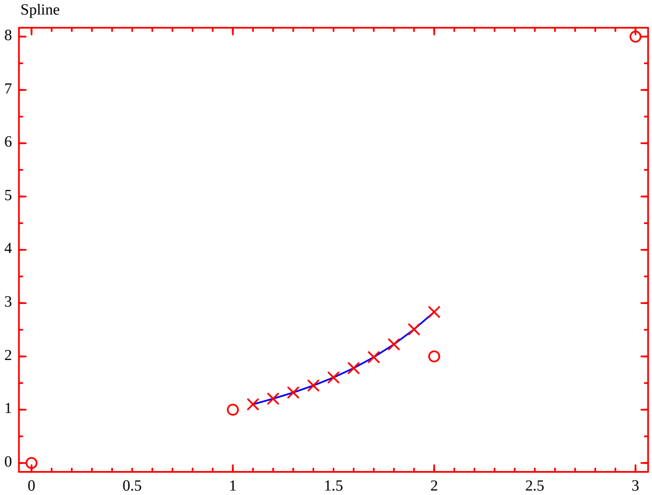
Objective: go from position A: (z_A, θ_A) to B: (z_B, θ_B)



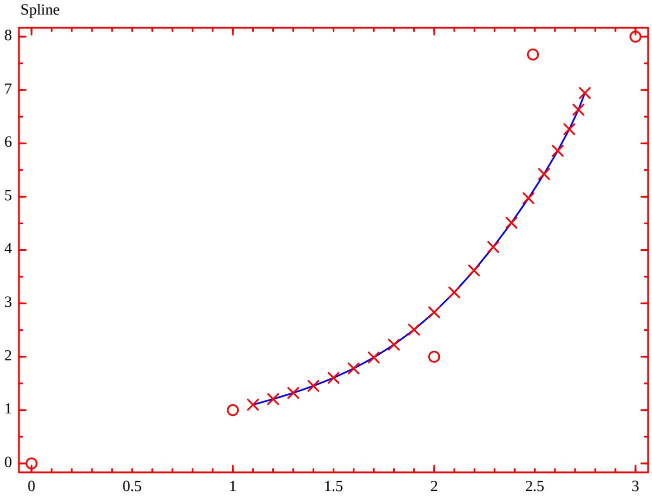
- Move in present direction from A
- End up at B with correct direction

Robot Control

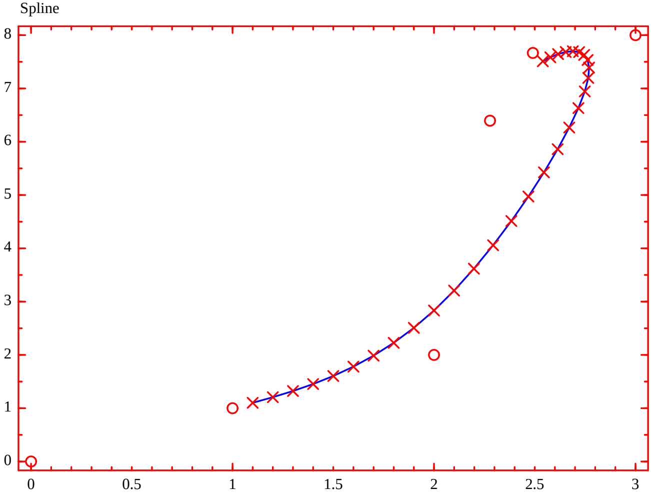
Spline



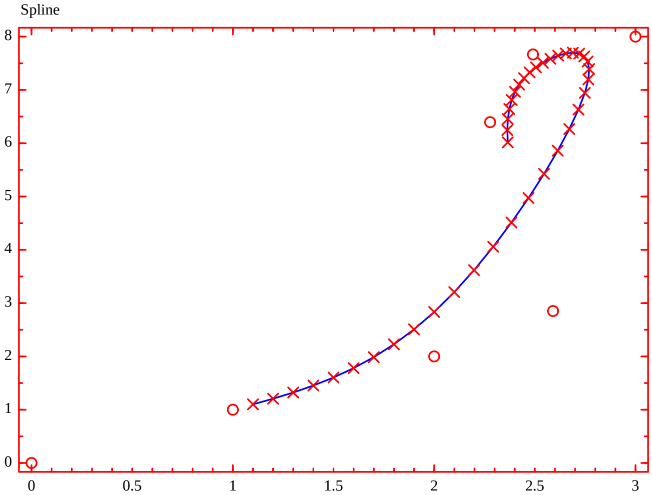
Spline



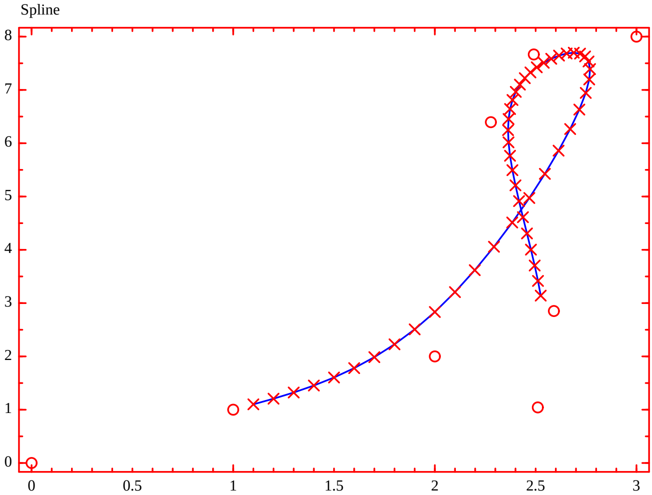
Spline



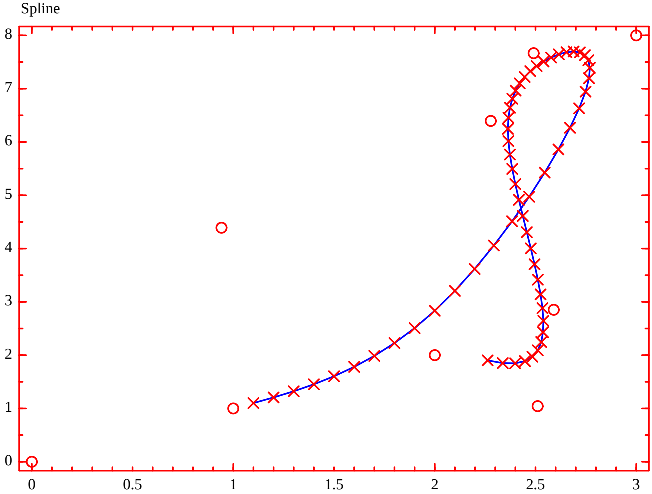
Spline



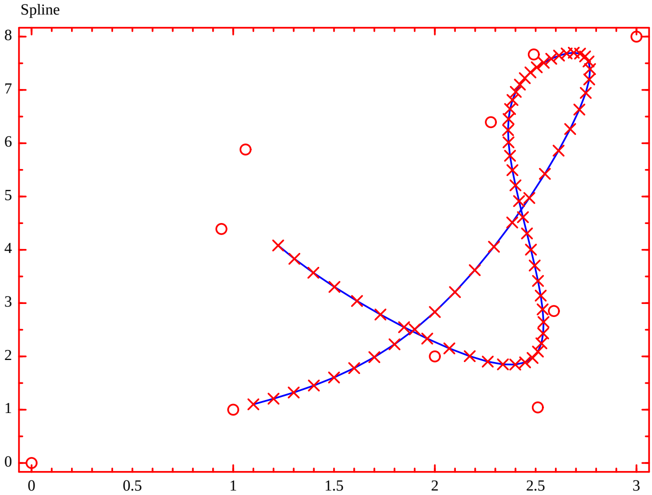
Spline



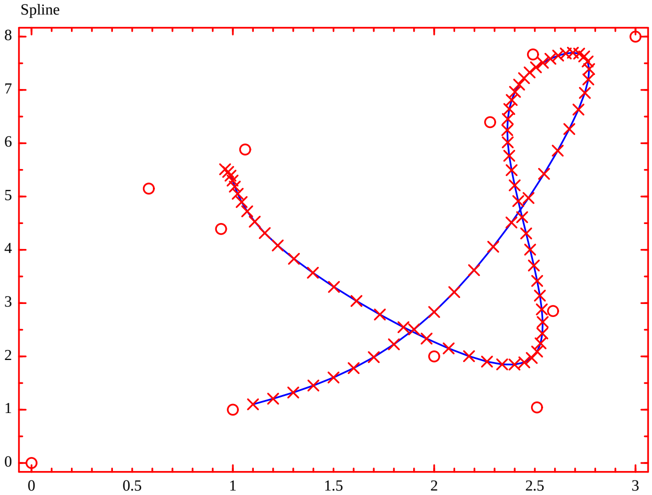
Spline



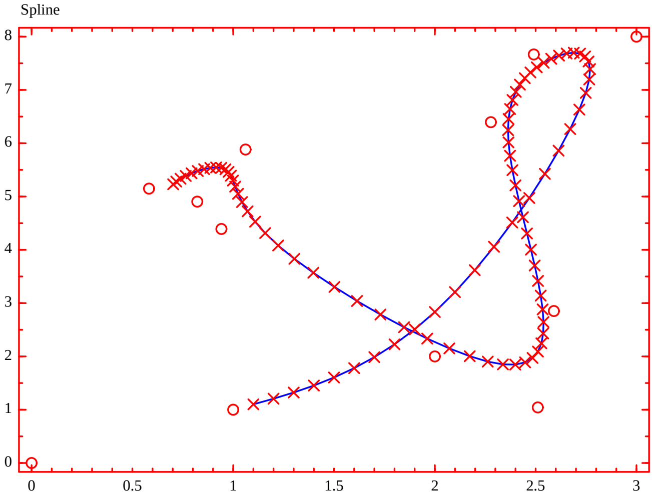
Spline



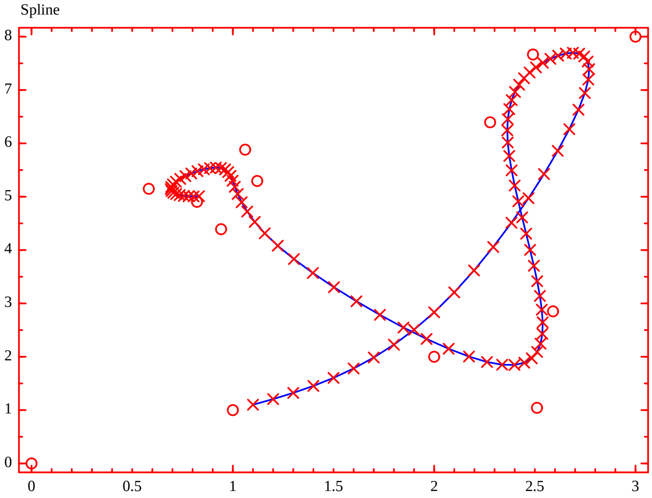
Spline



Spline



Spline



Uniform cubic B-splines

Most commonly used form of B-spline
Successive use of 4 control points $\mathbf{p_i} = (x, y)$

$$\mathbf{S_i}(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_{i-1} \\ \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{p}_{i+2} \end{bmatrix}, t \in [0, 1]$$

Uniform cubic B-splines

Most commonly used form of B-spline
Successive use of 4 control points $\mathbf{p_i} = (x, y)$

$$\mathbf{S_i}(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{p}_{i-1} \\ \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{p}_{i+2} \end{bmatrix}, t \in [0, 1]$$

Sysquake or Matlab

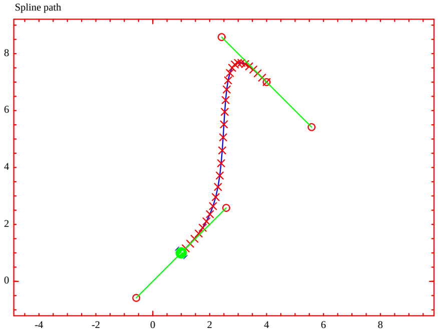
```
function S = bspline(x, y, n)
  if nargin == 2, n=10; end;
  P = [x', y'];
  t = 1/n:1/n:1;
  tL = length(t);
  for i=1:length(x)-3
      for j=1:tL,
         S((i-1)*tL + j,:) = [t(j)^3 t(j)^2 t(j) 1]*1/6*[-1 3 -3 1;3 -6 3 0;-3 0 3 0;1 4 1 0]*P(i:i+3,:);
      end
  end
```

A simple spline

- A: Align 3 control points
- B: Align 3 control points

Control point split $r = |z_B - z_A|/3$

$$p_1 = z_A - re^{i\theta_A} \qquad p_4 = z_B - re^{i\theta_B}$$
$$p_2 = z_A \qquad p_5 = z_B$$
$$p_3 = z_A + re^{i\theta_A} \qquad p_6 = z_B + re^{i\theta_B}$$

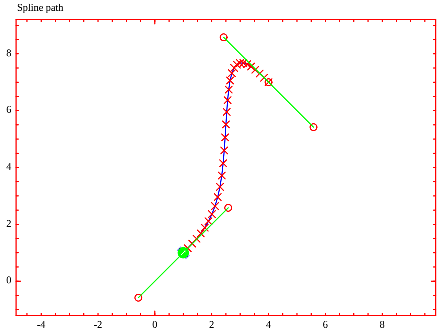


A simple spline

- A: Align 3 control points
- B: Align 3 control points

Control point split $r = |z_B - z_A|/3$

$$p_1 = z_A - re^{i\theta_A} \qquad p_4 = z_B - re^{i\theta_B}$$
$$p_2 = z_A \qquad p_5 = z_B$$
$$p_3 = z_A + re^{i\theta_A} \qquad p_6 = z_B + re^{i\theta_B}$$



Sysquake or Matlab

```
function (z, x, y) = newspline(zA, thA, zB, thB, n)
   r = abs(zB-zA)/3;  % approximate distance between control points
   Z = [zA - r*exp(i*thA), zA,  zA +  r*exp(i*thA), zB -  r*exp(i*thB), zB, zB +  r*exp(i*thB)];
   x = real(Z);
   y = imag(Z);
   z=bspline(x,y,n);
```

Robot Control

Inverse dynamics

Robot model $(u, v) \longrightarrow (\Delta\theta, \Delta z)$

$$\Delta\theta(k) = \theta(k) - \theta(k-1) = u(k)$$
$$\Delta z(k) = z(k) - z(k-1) = v(k)e^{i\theta(k)}$$

Inverse dynamics

Robot model $(u, v) \longrightarrow (\Delta\theta, \Delta z)$

$$\Delta\theta(k) = \theta(k) - \theta(k-1) = u(k)$$
$$\Delta z(k) = z(k) - z(k-1) = v(k)e^{i\theta(k)}$$

Inverse model $(\Delta\theta, \Delta z) \longrightarrow (u, v)$

$$u(k) = \Delta\theta(k)$$
$$v(k) = \Delta z(k)e^{-i\theta(k)} = |\Delta z(k)| \quad (v > 0)$$

Inverse dynamics

Robot model $(u, v) \to (\Delta\theta, \Delta z)$

$$\Delta\theta(k) = \theta(k) - \theta(k-1) = u(k)$$
$$\Delta z(k) = z(k) - z(k-1) = v(k)e^{i\theta(k)}$$

Inverse model $(\Delta\theta, \Delta z) \to (u, v)$

$$u(k) = \Delta\theta(k)$$
$$v(k) = \Delta z(k)e^{-i\theta(k)} = |\Delta z(k)| \quad (v > 0)$$

If $\Delta z(k) \neq 0$: $\Delta z \to (u, v)$

$$u(k) = \Delta\theta(k) = \angle\Delta z(k) - \angle\Delta z(k-1)$$

Tracking any trajectory

Desired trajectory $A \rightarrow B$

$$A = \{z_A, \theta_A\} = \{z(0), \theta(0)\}, \ldots, \{z(N), \theta(N)\} = \{z_B, \theta_B\} = B$$

Control

$$\begin{aligned} u(k) &= \Delta\theta(k) \in (-\pi, \pi) \\ v(k) &= |\Delta z(k)| \end{aligned} \qquad k = 1, \ldots, N$$

Special case: line tracking

Turn during k_1 samples

$$u(k) = \Delta\theta(k) = [\angle(z_B - z_A) - \theta_A]/k_1 \qquad k = 1, \ldots, k_1$$
$$v(k) = 0$$

Special case: line tracking

Turn during k_1 samples

$$\begin{aligned} u(k) &= \Delta\theta(k) = [\angle(z_B - z_A) - \theta_A]/k_1 \\ v(k) &= 0 \end{aligned} \qquad k = 1, \ldots, k_1$$

Move straight during k_2 samples

$$\begin{aligned} u(k) &= 0 \\ v(k) &= |\Delta z(k)| = |z_B - z_A|/k_2 \end{aligned} \qquad k = k_1 + 1, \ldots, k_1 + k_2$$

Special case: line tracking

Turn during k_1 samples

$$u(k) = \Delta\theta(k) = [\angle(z_B - z_A) - \theta_A]/k_1 \qquad k = 1, \ldots, k_1$$
$$v(k) = 0$$

Move straight during k_2 samples

$$u(k) = 0 \qquad k = k_1 + 1, \ldots, k_1 + k_2$$
$$v(k) = |\Delta z(k)| = |z_B - z_A|/k_2$$

Turn during k_3 samples

$$u(k) = \Delta\theta(k) = [\theta_B - \angle(z_B - z_A)]/k_3 \qquad k = k_1+k_2+1, \ldots, k_1+k_2+k_3$$
$$v(k) = 0$$

Spline tracking

Desired spline trajectory $A \to B$

$$A : z_A = z(0), z(1), z(2), \ldots, z(k), \ldots, z(N) = z_B : B$$

Calculate

$$\begin{aligned} \Delta z(k) &= z(k) - z(k-1) \\ \theta(k) &= \angle \Delta z(k) \\ \theta(0) &= \theta_A \\ \Delta\theta(k) &= \theta(k) - \theta(k-1) \end{aligned} \qquad k = 1, \ldots, N$$

Spline tracking

Desired spline trajectory $A \to B$

$$A : z_A = z(0), z(1), z(2), \ldots, z(k), \ldots, z(N) = z_B : B$$

Calculate

$$\begin{aligned} \Delta z(k) &= z(k) - z(k-1) \\ \theta(k) &= \angle \Delta z(k) \\ \theta(0) &= \theta_A \\ \Delta\theta(k) &= \theta(k) - \theta(k-1) \end{aligned} \qquad k = 1, \ldots, N$$

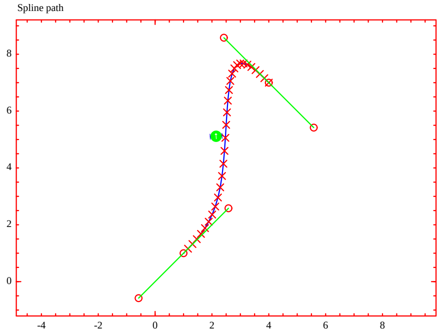
Control

$$\begin{aligned} u(k) &= \Delta\theta(k) \in (-\pi, \pi) \\ v(k) &= |\Delta z(k)| \end{aligned} \qquad k = 1, \ldots, N$$

Robot Control

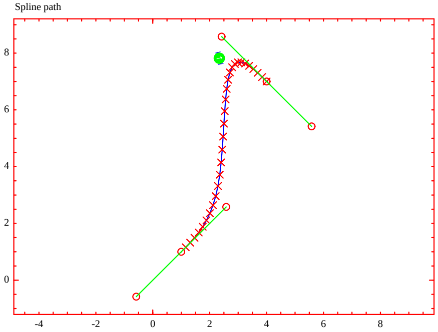
Modeling error

- Desired trajectory $\hat{z}(k)$
- Control design
  $\hat{z}(k) \longrightarrow (u(k), v(k))$
- Robot dynamics
  $(u(k), v(k)) \longrightarrow z(k)$
- Perfect model $z(k) = \hat{z}(k)$
- Model error $z(k) \neq \hat{z}(k)$



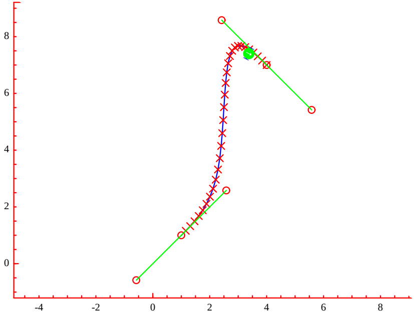
Modeling error

- Desired trajectory $\hat{z}(k)$
- Control design $\hat{z}(k) \longrightarrow (u(k), v(k))$
- Robot dynamics $(u(k), v(k)) \longrightarrow z(k)$
- Perfect model $z(k) = \hat{z}(k)$
- Model error $z(k) \neq \hat{z}(k)$



Modeling error

- Desired trajectory $\hat{z}(k)$
- Control design $\hat{z}(k) \longrightarrow (u(k), v(k))$
- Robot dynamics $(u(k), v(k)) \longrightarrow z(k)$
- Perfect model $z(k) = \hat{z}(k)$
- Model error $z(k) \neq \hat{z}(k)$



Use position measurements to update spline

Choose number of samples: $N = 3n$ to go from A to B

Spline design

$\hat{z}(k)$ from A to B with n grid points
$\hat{z} \rightarrow (u(k), v(k)),\ k = 1, \ldots, 3n$

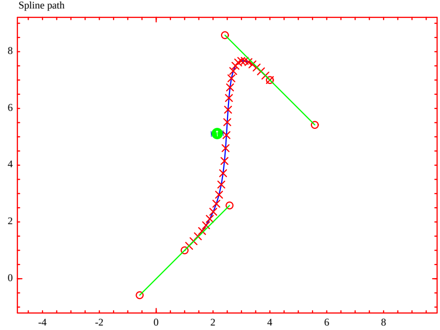
Feedforward control

While $N > 0$ and no new measurement, Execute $(u(k), v(k))$ and $N = N - 1,\ k = k + 1$

Feedback control

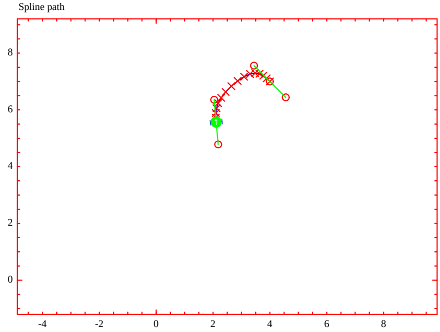
Measured position where $3|N$, set $n = N/3$ Update present position to A, Go to Spline design

Feedback with spline update



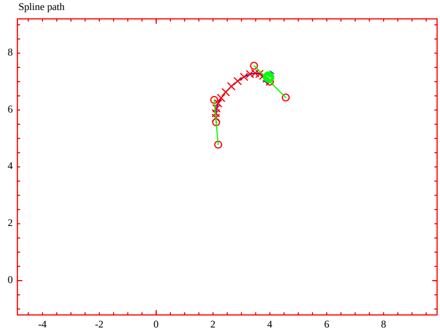
- Feedforward control

Feedback with spline update



- Feedforward control
- Update spline by present position

Feedback with spline update



- Feedforward control
- Update spline by present position
- Feedforward control

Examination Task

Prerequisites: Learn how to

- set sampling time
- output control w_R and w_L to local processor
- input camera picture to Matlab

Task: To be examined

- Calibrate robot: k_v and L
- Execute control to move robot from A to B