2-D vehicle control

Bohumír Sládek

March 11, 1995

1 Problem

Find the control law to move the truck from given initial state to the desired state (fig. 1).

Initial state: position (in rectangular coordinates) x_0, y_0, speed (in polar coordinates) v_0, α_0. Current state t_0. Other state variables (position derivatives of higher order) are neglected.

Desired final state: position (in rectangular coordinates) x_1, y_1, speed (in polar coordinates) v_1, α_1.

Free parameters: Time t_1 when the final state is reached and trajectory shape are arbitrary.

2 Theory of trajectory planning

The biggest problem in trajectory planning is caused by different number of degrees of freedom of the physical system and the approximating curve.

The vector of system initial state has 5 components $[x_0, y_0, v_0, alpha_0, t_0]$ (fig. 1). The required state has 4 components $([x_1, y_1, v_1, alpha_1], t_1$ is arbitrary. Consequently the trajectory planning problem has 9 degrees of freedom while most 2-D curve parametrizations has even number of parameters.

There are two ways how to match the problem with a solution:

1. Use a curve with higher number of parameters and fix free ones. The vehicle follows this trajectory exactly.
2. Use a curve with lower number of parameters fitting exactly final state and approximatively initial state. The vehicle uses a controller to minimize the difference between actual and fitted trajectory.

None of these two methods is superior to the other and both will be implemented in practice.

To test the approach where number of parameters is higher then the problem dimension, the polynomial trajectory was chosen. The trajectory is described by a set of equations parametrized by t.

$$\begin{aligned} x(t) &= a_x t^3 + b_x t^2 + c_x t + d_x \\ y(t) &= a_y t^3 + b_y t^2 + c_y t + d_y . \end{aligned} \tag{1}$$

such that

$$\begin{aligned} x(t_0) &= x_0 \qquad y(t_0) = y_0 \\ \dot{x}(t_0) &= \dot{x}_0 \qquad \dot{y}(t_0) = \dot{y}_0 \\ x(t_1) &= x_1 \qquad y(t_1) = y_1 \\ \dot{x}(t_1) &= \dot{x}_1 \qquad \dot{y}(t_1) = \dot{y}_1 . \end{aligned} \tag{2}$$

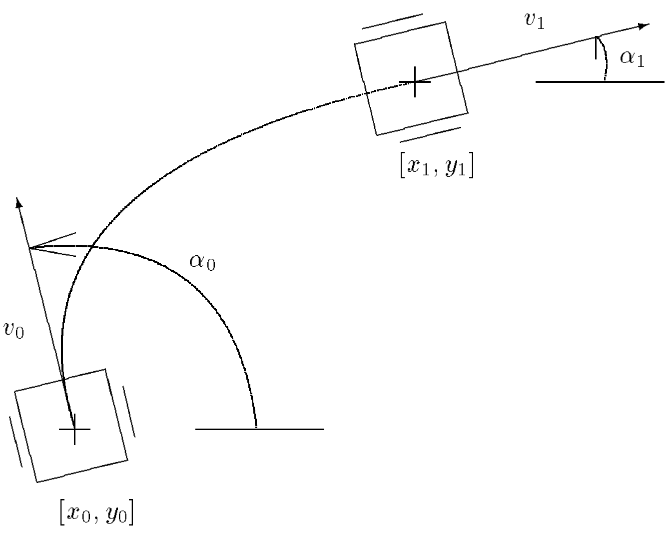

Figure 1: Problem definition

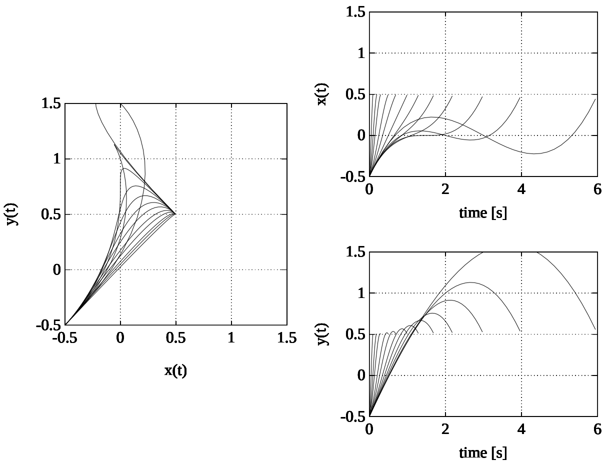

Figure 2: Polynomial trajectory parametrized by t_1.

The figure 2) shows, that the trajectory is highly sensitive with respect to the trajectory final time t_1. Bezier curves are in its mathematical nature also polynomials of 3rd degree but thank to its more natural representation it is easier to find a universal setting for the free parameter (figure 3).

To test the approach where number of trajectory parameters is lower then the dimension of the problem, the circular trajectory was chosen.

This method uses a circle defined by points $[x_t, y_t]$, $[x_1, y_1]$ and the vector $[v_1, alpha_1]$ as a desired trajectory. The controller acts to minimize the difference between the actual angle α_t and the desired angle $\bar{\alpha}_t$ along the trajectory.

Circular trajectory is described explicitly as

$$x(t)^2 + y(t)^2 = r^2. \tag{3}$$

such that

$$x(t_0) \quad = \quad x_0 \qquad y(t_0) = y_0 \tag{4}$$

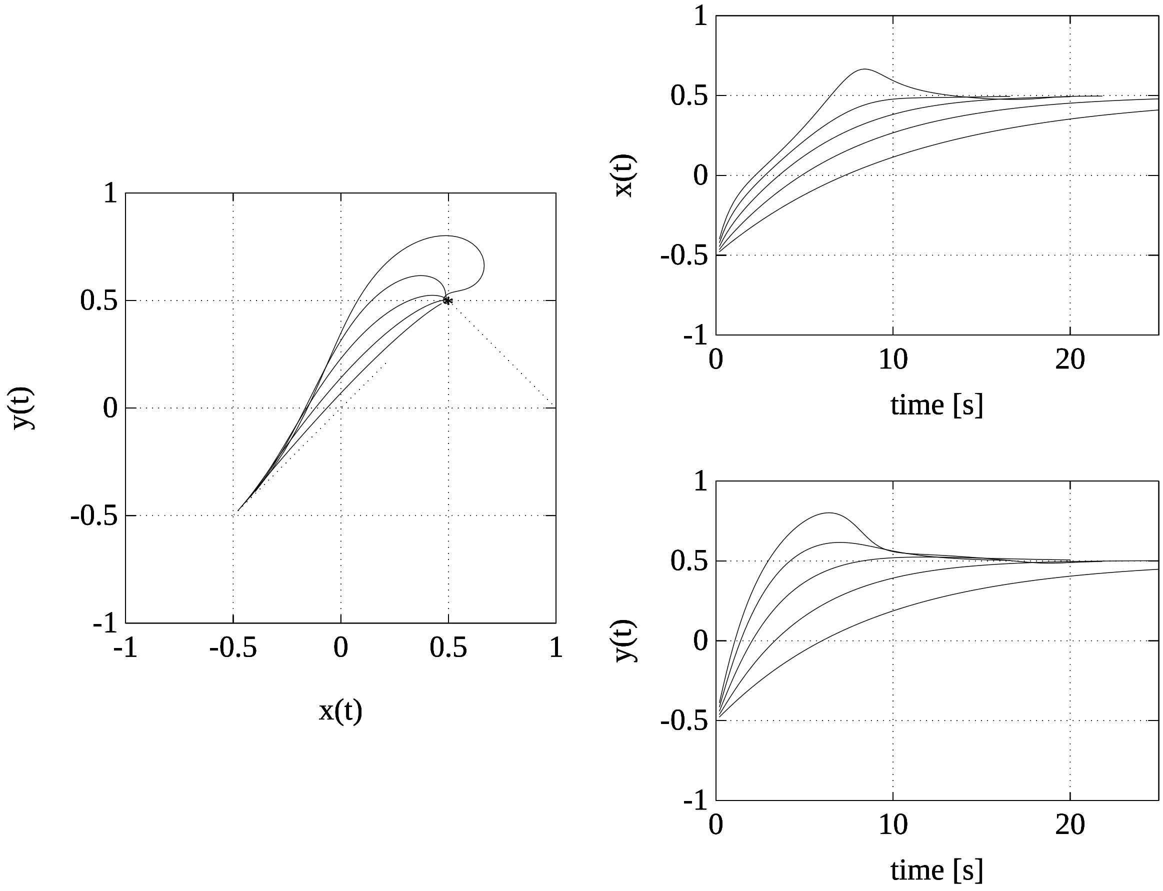

Figure 3: Polynomial trajectory (Bezier splines) parametrized by B_2, B_3.

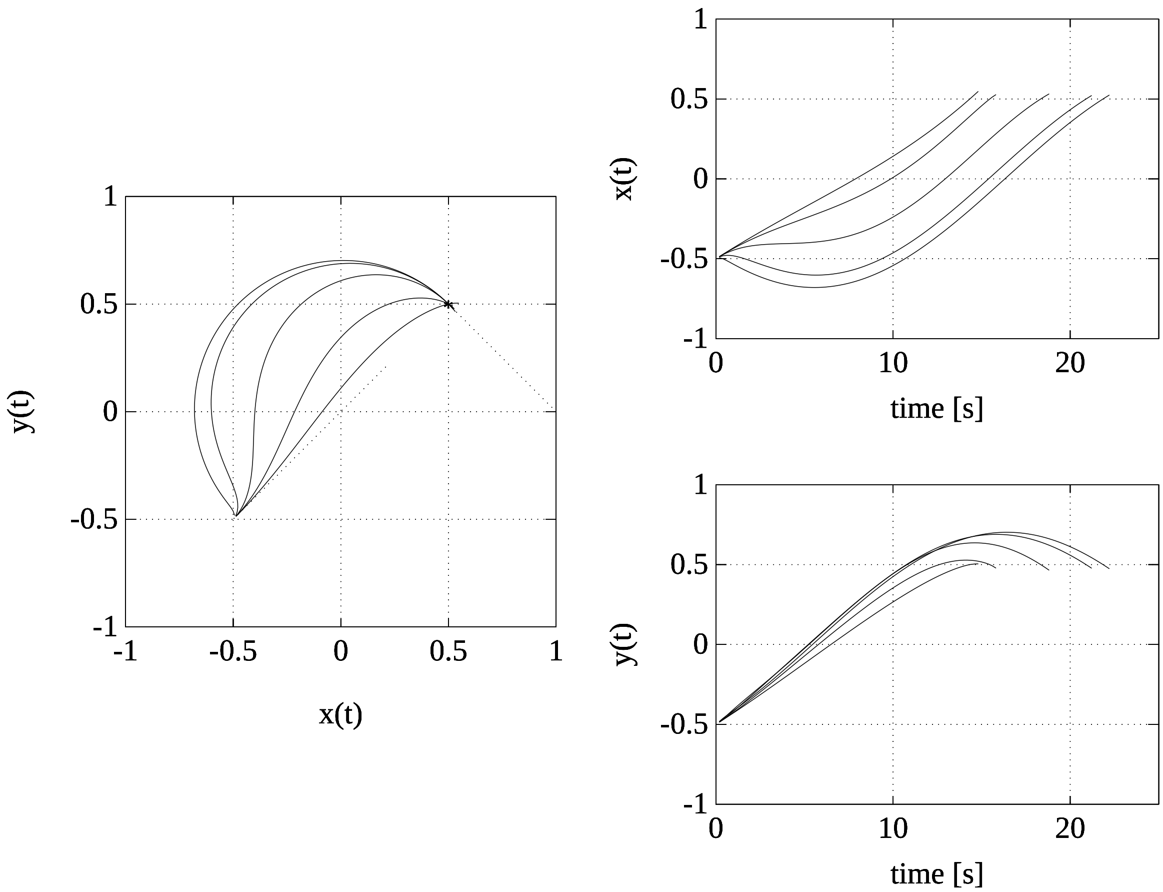

Figure 4: Circular trajectory parametrized by $K_{p\omega}$.

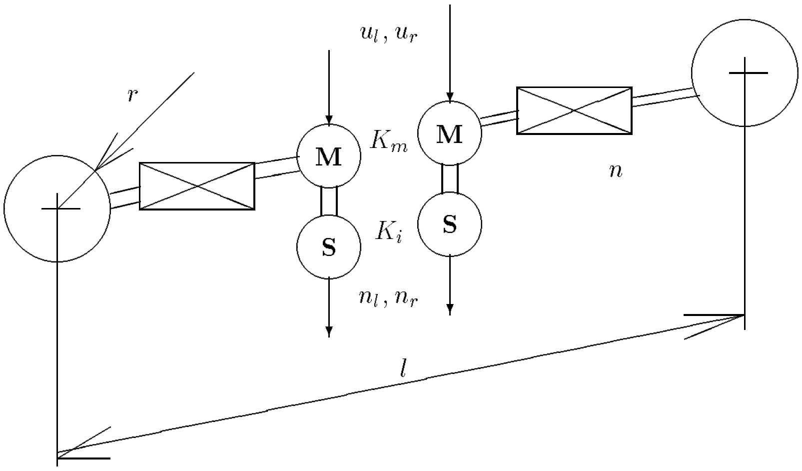

Figure 5: Mechanical vehicle model

$$\begin{aligned} x(t_1) &= x_1 \qquad y(t_1) = y_1 \\ \dot{x}(t_1) &= \dot{x}_1 \qquad \dot{y}(t_1) = \dot{y}_1. \end{aligned}$$

Behavior of this controller is superior to the polynomial one (compare figures 2. and 4.) and therefore it was chosen for the final implementation.

3 Vehicle model

The mechanical scheme of the vehicle is shown at the figure (5). Its numerical values of model parameters can be found in the table 3.

To simplify further computation these parameters were aggregated:

Sensor constants $K_{i,x}$, $K_{i,\alpha}$:

$$\begin{aligned} x_k &= x_{k-1} + K_{i,x} n_r + n_l / over 2 \cos(\alpha) \\ y_k &= y_{k-1} + K_{i,x} n_r + n_l / over 2 \sin(\alpha) \\ \alpha_k &= \alpha_{k-1} + K_{i,\alpha}(n_r - n_l). \end{aligned} \tag{5}$$

System constants $K_{s,v}$, $K_{s,\omega}$:

$$\begin{aligned} v &= K_{s,v} u_r + u_l / over 2 \\ \omega &= K_{s,\omega}(u_r - u_l). \end{aligned} \tag{6}$$

$$\tag{7}$$

Parameter	Value
Tyre radius	$r = 72.5$ [mm]
Motor constants	$K_m = 1/0.017$ [rot/min/V]
Transmission ratio	$n = 1/25$ [-]
Distance of tyres	$l = 0.45$ [m]
Sensor sensitivity	$K_s = 96$ [1/rot]

Table 1: Model parameters

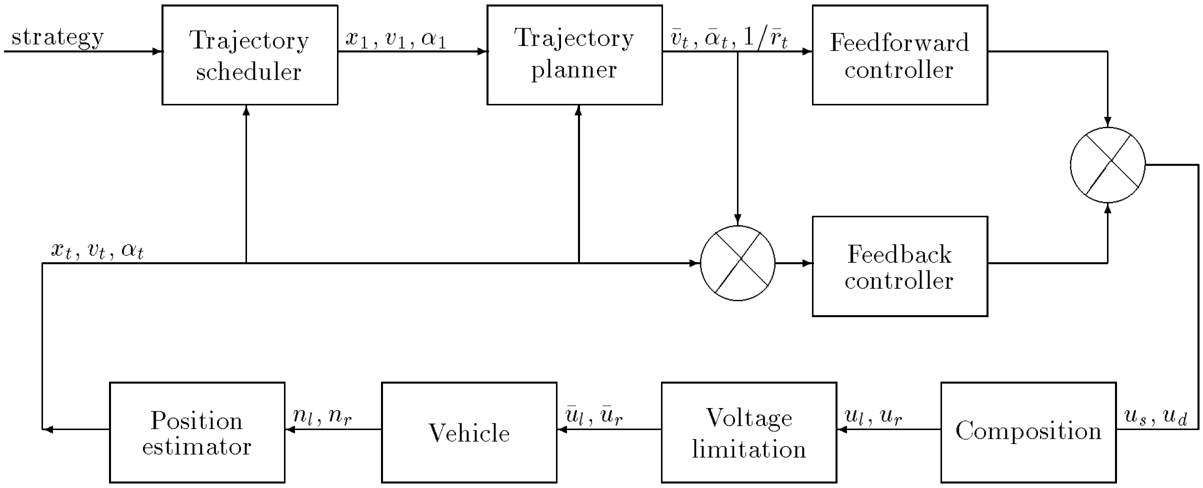

Figure 6: Controller structure

Controller constants $K_{p,v}$, $K_{p,\omega}$ (entered by user):

$$
\begin{aligned}
u_s &= K_{p,v}(\bar{v}_t - v_t) \\
u_d &= K_{p,\omega}(\bar{\alpha}_t - \alpha_t)
\end{aligned} \tag{8}
$$

where:

x, y	= position
α	= angle
v	= velocity
ω	= angular velocity
u_l, u_r	= voltage at the left resp. right motor
n_l, n_r	= number of impulses of the left. resp. right IRC sensor

$$
\begin{aligned}
K_{i,x} &= n\frac{1}{K_i}2\pi r \\
K_{i,\alpha} &= n\frac{1}{K_i}\frac{2\pi r}{l} \\
K_{s,v} &= nK_m 2\pi r \\
K_{s,\omega} &= nK_m\frac{2\pi r}{l}
\end{aligned} \tag{9}
$$

4 Circular arc trajectory

The controller structure is shown at the figure 6. where:

x_1, v_1, α_1	= target coordinates
x_t, v_t, α_t	= actual coordinates
$\bar{v}_t, \bar{\alpha}_t$	= desired actual coordinates
u_l, u_r	= voltage for left resp. right motor
$\bar{u}_l, \bar{u}_r$	= motor voltage before limitation
n_l, n_r	= number of impulses from left resp. right IRC sensor.

4.1 Scheduler

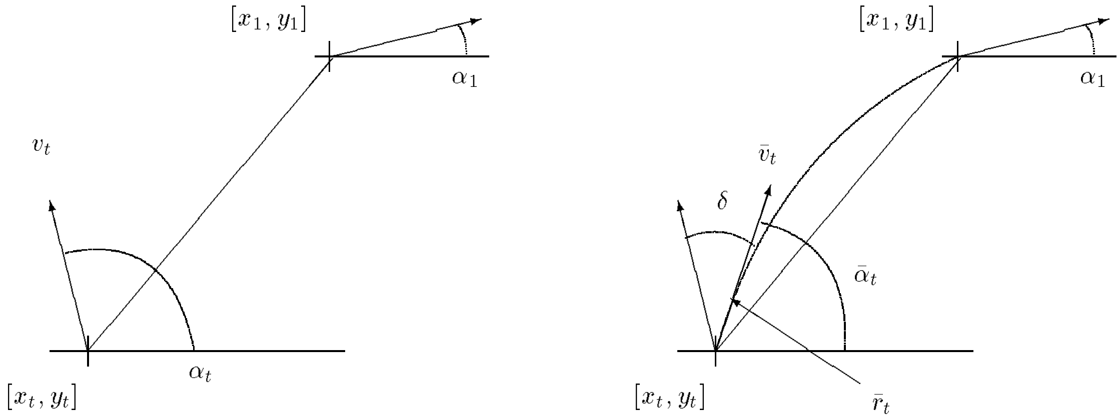

Figure 7: Trajectory computation I.

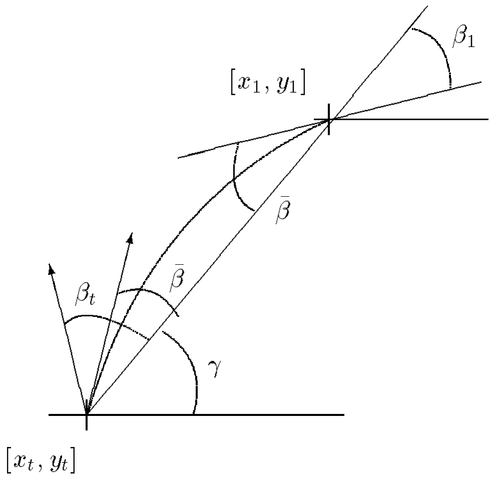

Figure 8: Trajectory computation II.

Converts strategy of vehicle motion into coordinates of next target point. In the example it is simply a vector of target coordinates x, y, v, α and a simple logic switching to next point when the previous one was reached.

4.2 Planner

Converts actual position x_t, y_t and target coordinates x_1, y_1, v_1, α_1 into desired parameters of the vehicle trajectory $\bar{v}, \bar{\alpha}, 1/\bar{r}$ at the actual time.

$$
\begin{aligned}
\beta_1 &= \gamma - \alpha_1 \\
\bar{\beta} &= \beta_1 \\
\bar{\alpha} &= \bar{\beta} + \gamma \qquad (10) \\
\bar{v} &= v_1 \qquad (11) \\
\bar{r} &= -\frac{\sqrt{(x_1 - x_t)^2 + (y_1 - y_t)^2}}{2} \frac{1}{\sin(\beta_1)} \qquad (12)
\end{aligned}
$$

4.3 Feedback controller

Converts the difference between actual (v, α) and desired $(\bar{v}, \bar{\alpha})$ trajectory parameters into motor voltages. Simple P controllers are used.

$$\begin{aligned} u_{d1} &= K_{p,\omega} \frac{1}{K_{s,\omega}} (\bar{\alpha} - \alpha) \\ u_{s1} &= K_{p,v} \frac{1}{K_{s,v}} (\bar{v} - v). \end{aligned} \tag{13}$$

4.4 Feedforward controller

Removes the steady state error for nominal vehicle parameters.

$$\begin{aligned} u_{d2} &= \frac{1}{K_{s,\omega}} \frac{\bar{v}}{\bar{r}} \\ u_{s2} &= \frac{1}{K_{s,v}} \bar{v}. \end{aligned} \tag{14}$$

4.5 Conversion

Converts compound voltages u_s, u_d into individual ones u_l, u_r by solving a linear set of equations:

$$\begin{aligned} u_s &= \frac{u_l + u_r}{2} \\ u_d &= u_r - u_l. \end{aligned} \tag{15}$$

4.6 Limiter

Prevents the motor voltage to exceed its maximum value. Both u_l and u_r must be decreased by the same coefficient $k \leq 1$ to prevent unplanned changes of vehicle direction.

$$\begin{aligned} k &= \min(1, \max(u_l/u_m, u_r/u_m)) \\ \bar{u}_l &= u_l/k \\ \bar{u}_r &= u_r/k \end{aligned} \tag{16}$$

4.7 System

This block contains motors, IRC sensors and their counters. Motor dynamics is influenced by vehicle load.

4.8 Position estimation

Converts old position x_k, y_k and number of impulses n_l, n_r into new vehicle positionx_{k+1}, y_{k+1} and velocity v_{k+1}, α_{k+1}.

$$\begin{aligned} x_k &= x_{k-1} + K_{i,x} \frac{n_r + n_l}{2} \cos \alpha \\ y_k &= y_{k-1} + K_{i,x} \frac{n_r + n_l}{2} \sin \alpha \\ \alpha_k &= \alpha_{k-1} + K_{i,\alpha}(n_r - n_l) \end{aligned} \tag{17}$$

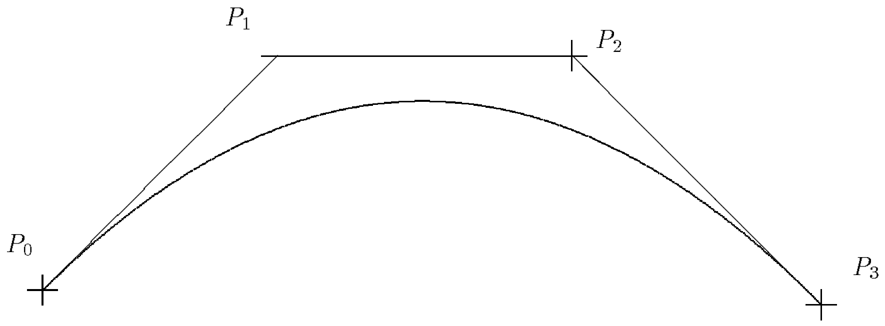

Figure 9: Bezier curve

4.9 Gain scheduling

P controller with fixed gain results in the azimuth branch results in vehicle behavior with fixed time constant. This means that the trajectory between identical target points will have different shape for different vehicle speed and for different trajectory scaling.

To obtain invariance with respect to vehicle speed, controller gain must be scheduled by the formula:

$$\bar{K}_{p,\omega} = K_{p,\omega} \frac{v}{v_{ref}}$$

To obtain invariance with respect to changes in x, y scale, controller gain must be scheduled by the factor:

$$\bar{K}_{p,\omega} = K_{p,\omega} \frac{l_{ref}}{\sqrt{(x_1 - x_t)^2 + (y_1 - y_t)^2}}.$$

4.10 Increasing computation speed

Angles α, β, γ can be represented in two ways:

- as the angle in degrees or radians.
- as a couple $[\sin(\phi), \cos(\phi)]$.

The former approach is more intuitive, leads to a straightforward implementation but requires computation of a goniometric function. The later one gives faster computations.

5 Bezier curve trajectory

The controller based on Bezier curves has the structure identical with the controller, based on circular arcs (fig. 7). All changes are limited to the Planner block - other blocks remain unchanged.

5.1 Theory of Bezier curves

The analytical expression for Bezier curve is

$$B(t) = (1-t)^3 P_0 + t(1-t)^2 P_1 + 3t^2(1-t) P_2 t^3 P_3. \tag{18}$$

This ban be also shortened as

$$B(t) = \sum_{1}^{4} P_1 B_i \tag{19}$$

where:
$P_1 \dots P_4$ = coordinates of control points (figure 5.1)
$B_1 \dots B_4$ = Bernstein polynomials
In the 2-d case, each point P_i is represented by a pair $[x_i, y_i]$. All operations must be performed for each component.

To obtain curvature along the bezier curve, needed for computation of the differential voltage u_d, first and second derivative must be computed.

$$\begin{aligned} \dot{B}(t) &= -3(1-t)^2 P_0 + 3(1-t)(1-3t)P_1 + 3t(2-3t)P_2 + 3t^2 P_3 & (20) \\ \ddot{B}(t) &= 6(1-t)P_0 + 6(2-3t)P_1 + 6(1-2t)P_2 + 6tP_3 & (21) \end{aligned}$$

First and second derivatives at the begin and end of the curve are obtained substitutions $t = 0$ resp. $t = 1$:

$$\begin{aligned} \dot{B}(0) &= 3(P_1 - P_0) \qquad \ddot{B}(0) = 6(P_2 - P_1 + P_0) & (22) \\ \dot{B}(1) &= 3(P_2 - P_1) \qquad \ddot{B}(1) = 6(P_3 - 2P_2 + P_1) & (23) \end{aligned}$$

5.2 Planner

Desired angle $\bar{\alpha}$ is set directly to α_t - this method uses feedback only in vehicle speed. Trajectory shape is controlled in feedforward branch only.

Desired vehicle velocity is computed in the same way as in the case of circular arc trajectory, e.g. $\bar{v} = v_1$.

Control points for the Bezier curve are set by actual state and final state:

$$\begin{aligned} P_0 &= [x_0, y_0] & (24) \\ P_1 &= [x_0 + dl\cos\alpha_0 y_0 + dl\sin\alpha_0] & (25) \\ P_2 &= [x_1 - dl\cos\alpha_1 y_1 - dl\sin\alpha_1] & (26) \\ P_3 &= [x_1, y_1] & (27) \end{aligned}$$

The distance dl was set by a rule of thumb to

$$dl = \frac{1}{3}\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}.$$

This setting leads to smooth trajectory with small overshoots. Constants smaller, then 1/5 tend to a piecewise linear approximation, those bigger then 1/2 tend to be unstable (fig.3).

Note, that although at any moment the planned trajectory is a bezier curve, the resulting trajectory has nothing to do with it. This effect is caused by the fact, that at any moment, only the first step is applied and another curve is computed.

Inverse curve radius $1/\bar{r}$ at the time $t = 0$ will be obtained by substitution of $\dot{x}, \dot{y}, \dot{y}, \dot{y}$ from (22 into the general expression for radius of curvature:

$$\frac{1}{r} = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{\sqrt{\dot{x}^2 + \dot{y}^2}^3}$$

6 Conclusion

Both algorithms for vehicle control passed through extensive tests by computer simulation. Both of them proved its robustness and performance. The algorithm based on Bezier curves seems to be faster and needs only one controller parameter, the algorithm based on circular arcs seems to be more robust with respect to changes in vehicle parameters and also more flexible.

In both cases it is necessary to accept limitations done by the task itself rather then by the control algorithm:

1) The couple DC motor/IRC angle sensor works well only in certain range of motor speeds. In the area of very low speeds, its operation is not quite smooth and may even become unstable.

2) If changes in azimuth are big, it is hard to predict the trajectory shape. It is recommended to keep azimuth changes under $3/4\ \pi$.