ENPM 667 Project #1

Technical report on a journal paper

Group Members:

Chang-Hong Chen, Hrushikesh Budhale

Selected Paper:

Spline-Based Motion Planning for Autonomous Guided Vehicle in a Dynamic Environment (Link)

Abstract

The paper presents a methods for solving constrained optimization problems on trajectory planning. The proposed method applies to many different types of systems including linear and nonlinear models. The method computes smooth motion trajectories that satisfies all kinematic and other constraints by spline parameterization. This methods can also be applied to environment with uncertainty and changes dynamically through online motion planning.

Introduction

Motion planning is an important problem for robots like autonomous guided vehicles (AGVs) to transport between station and station in modern industry. The AGVs have to move in a collision-free trajectory in every instant. These trajectories often have to be computed in ad-hoc methods where tracks are constant and velocities are highly limited. As a consequences, there is a growing interest in more flexible motion planning techniques, allowing AGVs with more freedom and robust behavior dealing with obstacles in dynamic environment.

There are mainly two different kinds of approaches when solving the planning problems, coupled and decoupled methods. Coupled motion planning solves the optimal motion trajectory and the required control inputs in the same step. Decoupled motion planning separates the trajectory planning and trajectory following in to two different stage.

- **Coupled:**

  Find the path and the control at the same time

  - pros: optimal
  - cons: slow, often has to simplify the environment or model

- **Decoupled:**

  First search for a path, then find a control method that guide the robot through the path, split the problem into path finding and path following

  - pros: fast
  - cons: sub-optimal, doesn't account kinematic constraint when finding path

  Example:

  - Path Finding: graph-based search (A*, D*), random sampling (RRT), probabilistic road maps (PRM), optimization-based (Trajopt, ITOMP)
  - Path Following: Model Predictive Control (MPC)

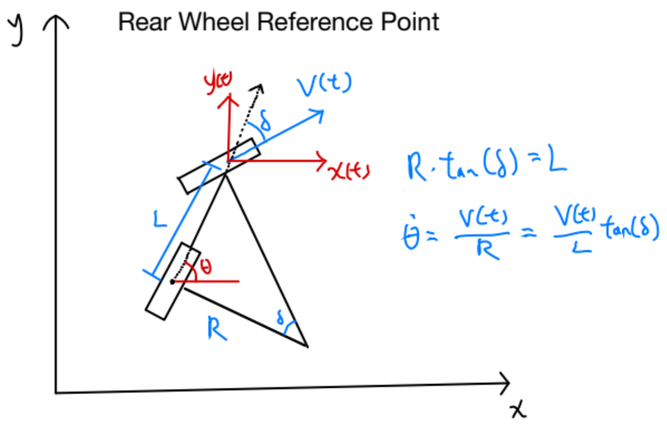
Most existing methods are decoupled methods because the computation speed of coupled motion planning methods are often too slow to use online. However, the decoupled method will return suboptimal results because of the separation of planning and control. Furthermore, the computed paths are often infeasible because of lacking the information of kinematic constraints in the path planning phase.

The paper provides a novel approach to the coupled motion planning methods. The proposed method uses spline parameterization methods to form an output trajectory that is guaranteed to satisfy all the input and state constraints on a differentially flat system. It simplifies the complex infinite constraints on any time along the trajectory into solvable finite constraints by using B-splines relaxations while guaranteeing the constraints will be held at all times.

Problem Formulation

The proposed method in the paper is to find the optimal trajectory along with the control input u(t) that brings the robot from its initial configuration q(0) to a final configuration q(T) while accounting all constraints and the robot kinematic along the way.

The paper uses a nonholonomic vehicle such as the bicycle model for example.



y
Rear Wheel Reference Point
y(t)
V(t)
δ
x(t)
L
θ
R
δ
x
$R \cdot \tan(\delta) = L$
$\dot{\theta} = \frac{V(t)}{R} = \frac{V(t)}{L} \tan(\delta)$

Fig. 1 Geometric representation for bicycle model

The kinematics of a bicycle model are descibed as follows

$$\dot{x}(t) = V(t) \cdot cos\theta(t)$$
$$\dot{y}(t) = V(t) \cdot sin\theta(t)$$
$$\dot{\theta} = \frac{V(t)}{L} \cdot tan\delta(t)$$

$\theta(t)$ is the orientation of the vehicle rear wheel, $\delta(t)$ is the steering angle, and L is the length of the wheel base.

The bicycle model is a nonholonomic model that has two input $u(t) = [V(t), \delta(t)]^{\mathrm{T}}$ and three states $q(t) = [x(t), y(t), \theta(t)]^{\mathrm{T}}$.

The boundary conditions of the trajectory are as follows

$$q(0) = q_0,\ q(T) = q_T$$
$$u(0) = u_0,\ u(T) = u_T$$

Combining with the input and kinodynamic constraints from time 0 to time T

$$\begin{gathered} V_{min} \leq V(t) \leq V_{max} \\ \dot{V}_{min} \leq \dot{V}(t) \leq \dot{V}_{max} \\ \delta_{min} \leq \delta(t) \leq \delta_{max} \\ \dot{\delta}_{min} \leq \dot{\delta}(t) \leq \dot{\delta}_{max} \\ \forall t \in [0, T] \end{gathered}$$

To account for moving obstacles, the paper uses a linear prediction model with the position p_k and velocity v_k of obstacles to predict their position over time.

$$p_k^{pred}(t) = p_k + t \cdot v_k$$

The collision constraints are expressed by the distance between the shape and position of the vehicle and obstacles

$$dist(veh(q(t)), obs_k(p_k^{pred}(t))) \geq \epsilon \quad \forall t \in [0, T]$$

where k=1....N and ϵ is a safety factor between the vehicle and the obstacle

All the constraint above can thus form a optimization control problem where the total time of the trajectory can be optimized. The problem is formulated using a dimensionless time $\tau = (t/T)$

$$\begin{aligned} \min_{q(.),u(.),T} \quad & T \\ s.t. \quad & q(0) = q_0, \quad q(1) = q_T \\ & u(0) = u_0, \quad u(1) = u_T \\ & V_{min} \leq V(\tau) \leq V_{max} \\ & \dot{V}_{min} \cdot T \leq \dot{V}(\tau) \leq \dot{V}_{max} \cdot T \\ & \delta_{min} \leq \delta(\tau) \leq \delta_{max} \\ & \dot{\delta}_{min} \cdot T \leq \dot{\delta}(\tau) \leq \dot{\delta}_{max} \cdot T \\ & dist(veh(q(\tau)), obs_k(p_k^{pred}(\tau))) \geq \epsilon \\ & k = 1...N, \ \forall \tau \in [0, 1] \end{aligned}$$

This is a infinite dimension problem where all constraints has to be obeyed along the time interval [0, T]. This is a optimization control problem that is hard to sovle due to the nonlinear and nonconvex constraints.

This paper transforms this infinite dimensional problem into a solvable finite dimensional problem by converting all the state constraints into polynomial function of the output and uses the properties of B-splines to construct controllable finite constraint due to the fact that this is a differentially flat system.

The following sections will be introducing the concepts of differentially flat systems and splines.

Mathematical Background

A. Differential Flatness

The problem of constrained optimal motion trajectory has been an challenging problem for a long time. In the 1990s, the concept of differential flatness arouse. Because the Differentially flat system encompass linear, controllable, and lots of nonlinear systems, it became more popular to deal with this kind of optimisation problem. Instead of complex system dynamic, the problem reduces to finding the best flat outputs that obeys the boundary conditions and the state and input constraints.

A system is differentially flat if we can find a set of outputs (equal in number to the number of inputs) such that all states and inputs can be determined from these outputs without integration.

More precisely, if the system has states $x \in \mathbb{R}^n$, and inputs $u \in \mathbb{R}^m$ then the system is flat if we can find outputs $y \in \mathbb{R}^m$of the form

$$y = \phi(x, u, \dot{u}, ..., u^{(p)})$$

such that

$$x = \psi(y, \dot{y}, ..., y^q)$$
$$u = \psi(y, \dot{y}, ..., y^q)$$

Differentially flat systems are useful when the explicit trajectory generation is required. We can obtain the corresponding input and states through the planned trajectory in output space because the input and states are function of outputs.

For example, in the case of the towed cable system, a resonable state space representation of the system consists of approximately 128 states. Traditional approaches to trajectory generation, such as optimal control, cannot be easiliy applied in this case. However, it follows from the fact that the system is differentially flat that the feasible trajectories of the system are completely characterized by the motion of the point at the bottom of the cable. By converting the input constraints on the system to constraints on the curvature and higher derivatives of the motion of the bottom of the cable, it is possible to compute efficient techniques for trajectory generation.

We can formulate the differential flat system below into an optimal control problem.
Consider a system governed by the differential equation

$$\dot{x} = f(x, u),\ x(0) = x_0$$

where $x \in \mathbb{R}^n$, and inputs $u \in \mathbb{R}^m$, we are interested in finding the control input $u(t),\ t \in [0, t_f]$ that steers the system from an initial state x_0, at $t = 0$ to a terminal state x_{t_f}, and that minimizes a performance criterion $g(x, u, t_f)$. The control law and the states must obey the constraints at all time.

$$h(x(t),\ u(t)) \geq 0, \forall t \in [0, t_f]$$

The constraint function are assumed to be polynomial in x and u.

Because this is a differential flat system, the output of the system can be in the form of the states and inputs.

$$y = \phi(x, u, \dot{u}, ..., u^{(p)})$$

such that

$$\begin{aligned} x &= \psi(y, \dot{y}, ..., y^q) \\ u &= \psi(y, \dot{y}, ..., y^q) \end{aligned}$$

Combining with the constraint above, we arrive at the following optimization problem:

$$\begin{aligned} \underset{y(.)}{min}\ \ & g(\psi_x(y, \dot{y}, ..., y^q),\ \psi_u(y, \dot{y}, ..., y^q), t_f) \\ s.t.\ \ & y^{(j)}(0) = y_0^{(j)}, j = 0, ..., q \\ & y^{(j)}(t_f) = y_{t_f}^{(j)}, j = 0, ..., q \\ & h(\psi_x(y, \dot{y}, ..., y^q), \psi_u(y, \dot{y}, ..., y^q)) \\ & \geq 0,\ \forall t \in [0, t_f] \end{aligned}$$

Another real example of differentially flatness system can be found in Appendix A.

Several methods have been proposed in the literature to guarantee constraint satisfaction at all time. Henrion and Lasserre (2006) propose a polynomial parameterization for the flat output, hereby transforming the constrained motion planning problem into a polynomial non-negativity problem

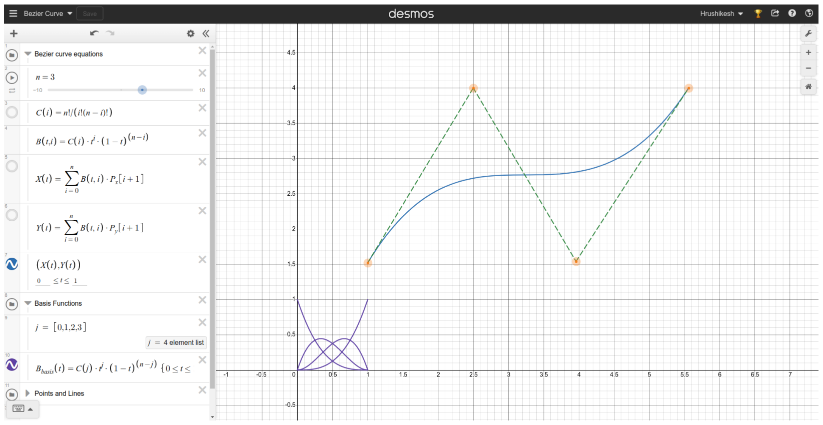
In this paper, the flat output is q(t) and the motion trajectory of this output is constructed by parameterization of spline curves. These curves can then be controlled by a finite conservative set of constraint points settled by the properties of B-splines.

B. Why Splines

A spline is a combination of multiple polynomial functions over different intervals. The coefficients c of these polynomials are chosen in such a way that the plot of these polynomials appear continuous even after taking their derivatives.

$$q(\tau) = \sum_{i=1}^{n} c_i^q \cdot B_i^q(\tau)$$

Where n and B_i are the degree and basis functions of the spline. A cubic spline for instance, is a combination of polynomials of degree 3. Where coefficients of these polynomials determine the shape of the spline. Alternatively, these splines can be considered as a weighted summation of basis functions. To demonstrate this we created an interactive demo in an online graphing tool.



$n = 3$

$C(i) = n!/(i!(n-i)!)$

$B(t,i) = C(i) \cdot t^i \cdot (1-t)^{(n-i)}$

$X(t) = \sum_{i=0}^{n} B(t,i) \cdot P_x[i+1]$

$Y(t) = \sum_{i=0}^{n} B(t,i) \cdot P_y[i+1]$

$(X(t), Y(t))$ $0 \leq t \leq 1$

$j = [0,1,2,3]$

$B_{basis}(t) = C(j) \cdot t^j \cdot (1-t)^{(n-j)} \{0 \leq t \leq$

Fig. 2 Demo application for representing Splines and it's basis functions in Desmos (online graphing tool)

Here it can be seen that the generated curve can be controlled using the 4 points around it. These four points are called control points and their position determines the coefficients of the polynomial.

This leads us to another important property: *convex hull*. That is, the spline formed by the control points is always bound to be contained in the convex hull of its control polygon. And this polygon corresponds to the piece-wise linear interpolation of the spline coefficients. Hence, the constraints on the spline can be enforced by imposing them on the spline coefficient.

$$a \leq c_i \leq b, \quad \forall i \in \{1 \ldots n\} \Rightarrow a \leq s(\tau) \leq b, \ \forall \tau \in [0,1]$$

Though this spline is controllable. it has few limitations which makes it less interesting for the trajectory optimisation purpose. One of them is that addition of every new control point increases the degree of the polynomial functions. This makes it difficult to calculate the solution with the increasing number of coefficients. Another major drawback of this spline is, introduction of new control points modifies the entire curve.

B-spline, which is a special type of spline, overcomes this limitation by selecting it's basis functions that span only over the part of region based in degree of spline. Thus the number of control points in a b-spline are not limited by the degree of that spline. This makes it suitable for generating trajectories with higher number of control points. To demonstrate this, we reproduced 3rd order b-spline with 6 control points, same as the one discussed in the paper. (Jupyter notebook)

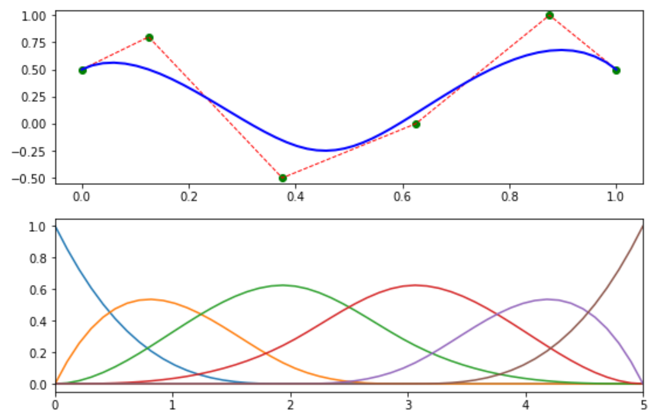


Fig. 3 B-Spline (in blue) and 6 control points (in green) along with it's basis functions (subplot 2)

Another property of b-spline which makes it unique for the trajectory optimisation problem it the property called *local control.* Unlike simple splines, which change the shape of the entire curve with the change in single control point, b-spline modifies only the curves around that point. This becomes possible because of the local span of the basis function forming a b-spline.

A new node (control point) can be inserted on the spline without modifying the shape of the spline. This property is important for planning trajectory, in the case when the trajectory is to be optimised at specific location without modifying the other remaining part of it. We demonstrate this by inserting a new control point at $t = 0.5$. To keep the spline intact adjoining 2 control points get new position and total number of control points become 7. (Jupyter Notebook)

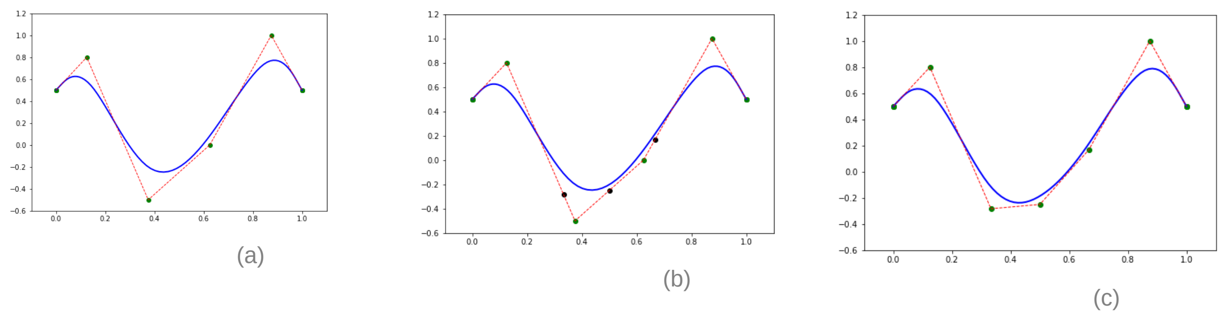


Fig. 4 Illustration of node insertion process. (a) Existing spline (b) Location of new node at $t = 0.5$ and updated position of adjacent 2 control points (c) Persistence of spline shape after node insertion.

C. Constraint based optimisation

As described above, the problem here is to find a trajectory that connects the initial and terminal configuration while satisfying other specified constraints at the endpoints (e.g., velocity and/or acceleration constraints). To demonstrate this we considered planning the trajectory for a point from one location to another, We assume that at time $t_0 = 0$ the point is at position $q_0 = 2m$ and at stand still, i.e. it has velocity V_0 of $0m/s$ and acceleration α_0 of $0m/s^2$.

Similarly we also define the constraints on the state at the end of period T, such that the point is at position $q_T = 10m$ and at stand still. These 6 constraints can be written as,

$$\begin{gathered} q_0 = 2m \\ V_0 = 0m/s \\ q_T = 10m \\ V_T = 0m/s \\ \alpha_T = 0m/s^2 \end{gathered}$$

Since we have six different constraints on the trajectory, we choose an equation of order 6, (degree 5) as a polynomial function of time t.

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

Using above equation and taking the appropriate number of derivatives we obtain following 6 equations,

$$\begin{aligned} q_0 &= a_0 + a_1 t_0 + a_2 t_0^2 + a_3 t_0^3 + a_4 t_0^4 + a_5 t_0^5 \\ V_0 &= a_1 + 2a_2 t_0 + 3a_3 t_0^2 + 4a_4 t_0^3 + 5a_5 t_0^4 \\ \alpha_0 &= 2a_2 + 6a_3 t_0 + 12a_4 t_0^2 + 20a_5 t_0^3 \\ q_T &= a_0 + a_1 t_T + a_2 t_T^2 + a_3 t_T^3 + a_4 t_T^4 + a_5 t_T^5 \\ V_T &= a_1 + 2a_2 t_T + 3a_3 t_T^2 + 4a_4 t_T^3 + 5a_5 t_T^4 \\ \alpha_T &= 2a_2 + 6a_3 t_T + 12a_4 t_T^2 + 20a_5 t_T^3 \end{aligned}$$

This system of equations can be combined into a single matrix equation.

$$\begin{bmatrix} 1 & t_0 & t_0^2 & t_0^3 & t_0^4 & t_0^5 \\ 0 & 1 & 2t_0 & 3t_0^2 & 4t_0^3 & 5t_0^4 \\ 0 & 0 & 2 & 6t_0 & 12t_0^2 & 20t_0^3 \\ 1 & t_T & t_T^2 & t_T^3 & t_T^4 & t_T^5 \\ 0 & 1 & 2t_T & 3t_T^2 & 4t_T^3 & 5t_T^4 \\ 0 & 0 & 2 & 6t_T & 12t_T^2 & 20t_T^3 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} = \begin{bmatrix} q_0 \\ V_0 \\ \alpha_0 \\ q_T \\ V_T \\ \alpha_T \end{bmatrix}$$

Writing above equation as, $Ma = b$
Where M is the coefficient matrix, $a = [a_0, a_1, a_2, a_3, a_4, a_5]^T$ is the vector of coefficients of the 6th order polynomial, and $b = [q_0, q_1, q_2, q_3, q_4, q_5]^T$ is the vector of constraints (initial and final position, velocity and acceleration). Solution to this equation can be written by taking the inverse of $M$and multiplying it with b such that, $a = M^{-1}b$

We solved these equations in our code. And obtained the following graphs for change of Position, Velocity and Acceleration over a period of 0 to 1.

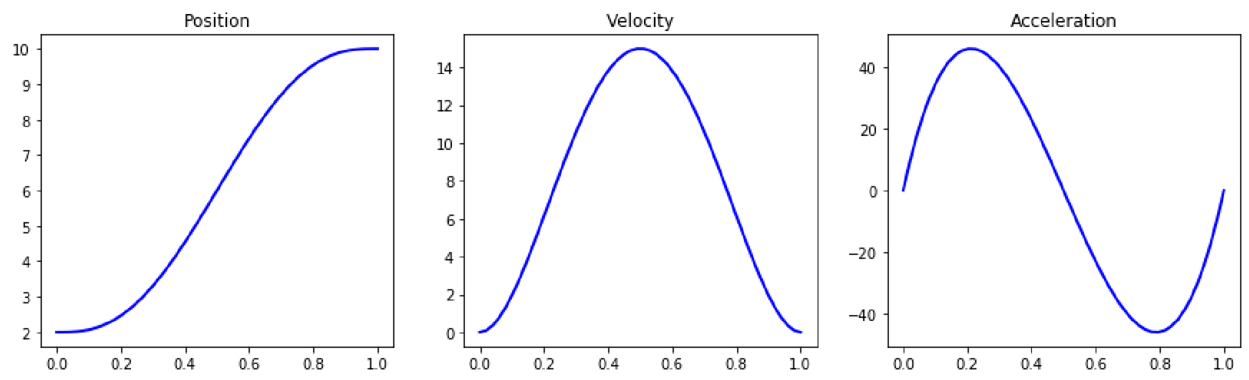


Fig. 5 Graph of Position, Velocity and Acceleration with respect to time for a point object

In the above problem we gave the value of time (1 sec) at which we want the point to reach certain value (10 m). But In real life we want to minimise the time, a point takes to reach there. And the limiting factor for the that time will be the maximum achievable velocity. Hence we can convert this problem into an optimisation problem by keeping the terminal time T as variable and adding constraint over maximum attainable velocity.

Following is a code snippet from our implementation of this problem in fitting_on_path_time.ipynb

```
opti.minimize( T )

opti.subject_to( T > 0.1 )          # Avoid T going in negative
opti.subject_to( f(0) == 2 )
opti.subject_to( df(0) == 0 )       # Initial conditions
opti.subject_to( ddf(0) == 0 )
opti.subject_to( f(T) == 10 )
opti.subject_to( df(T) == 0 )       # Terminal conditions
opti.subject_to( ddf(T) == 0 )

opti.subject_to( max_df(T) <= 5)    # Adding constraint on maximum achievable velocity
```

This change results in an optimum trajectory for going from initial point to terminal point. In the above example we reduced the maximum achievable velocity to 5 m/s. This gives optimum trajectory which takes 3 seconds to reach the terminal state.

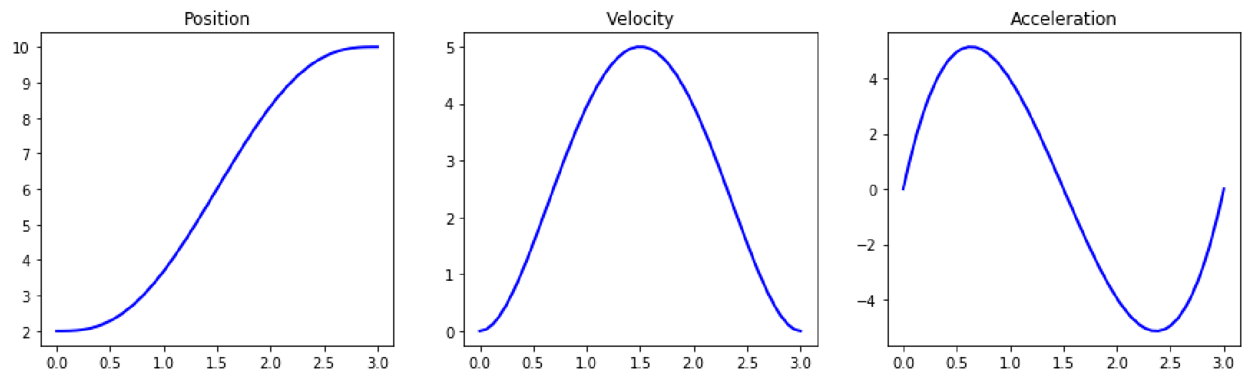


Fig. 6 Updated graphs after introducing 5 m/s limit on the maximum velocity. Takes 3 seconds to reach goal.

Spline-Based Motion Planning

A. Holonomic Vehicle

Consider the holonomic vehicle model with state $q = [x(\tau), y(\tau)]^T, \ \dot{q} = [\dot{x}(\tau), \dot{y}(\tau)]$

$q(\tau)$ is parameterised by the smooth spline space as explain above.

$$q(\tau) = \sum_{i=1}^{n} c_i^q \cdot B_i^q(\tau)$$

with $c_i^q \in \mathbb{R}^{n_q}$, where n_q denotes the dimension of q, in this case 2. Constraints of the velocity and acceleration can be used for finding the target spline coefficient c_i^q.

Orignial velocity constraints

$$V_{min} \leq \dot{q}(\tau) \leq V_{max}$$

The derivative of spline is also a spline curve, The derivative of $q(\tau)$ to t is as follows:

$$\dot{q}(\tau) = \sum_{i=1}^{n-1} c_i^{\dot{q}} \cdot \frac{1}{T} \cdot B_i^{\dot{q}}(\tau) \cdot$$

The $1/T$ comes from the derivative of τ

As mentions above, if we can assure that all the control points of the spline to be within the value of constraints, we can gaurantee the whole spline which lies inside the control polygon will be obeying the constraints.

$$V_{min} \leq c_i^{\dot{q}} \cdot \frac{1}{T} \leq V_{max}$$

$$V_{min} \cdot T \leq c_i^{\dot{q}} \leq V_{max} \cdot T$$

The above equation can garuntee the velocity along the velocity spline trajectory will be within maximum and minimum velocity.

After combining all constraints with initial and final conditions, the optimization problem becomes:

$$\begin{aligned} &\underset{c_i^q, T}{min} \quad T \\ &s.t. \ \ c_1^q = q_0, \ c_n^q = q_T \\ &\qquad \dot{c}_1^q = q_0, \ \dot{c}_n^q = q_T \\ &\qquad V_{min} \cdot T \leq c_i^{\dot{q}} \leq V_{max} \cdot T \\ &\qquad i = 1...n-1 \end{aligned}$$

We can then use a property of the spline derivative to establish relationship between $c_i^{\dot{q}}$ and c_i^q. The derivative of a spline is also a spline. And derived spline coefficients are functions of the orignial spline coefficients (see Appendix B).

After transforming all the conditions into function of c_i^q, optimization methods can be applied to find the best c_i^q to form the optimal spline trajectory.

B. Nonholonomic Vehicle

Nonholonomic Vehicle mentioned in the problem formation

$$\begin{gathered}\dot{x}(t) = V(t) \cdot cos\theta(t) \\ \dot{y}(t) = V(t) \cdot sin\theta(t) \\ \dot{\theta} = \frac{V(t)}{L} \cdot tan\delta(t)\end{gathered}$$

B-spline relaxations can only be applied on splines. All the constraint must be written as derivatives, anti derivatives, or polynomials of splines. Therefore, in this case, a nonlinear change of variables is adopted to transform all constraints into bounds on spline functions of τ.

Set two spline parameterisation function $r(\tau)$ and $\tilde{v}(\tau)$ as follows:

$$\begin{gathered}r(\tau) = tan\frac{\theta(\tau)}{2} \\ \tilde{v}(\tau) = \frac{V(\tau)}{1 + r(\tau)^2}\end{gathered}$$

The constraints of $V(\tau)$ can be described by the following spline function:

$$V(\tau) = \tilde{v}(\tau) \cdot (1 + r(\tau)^2)$$

The corresponding $x(\tau)$ and $y(\tau)$ are derived as follow:

$$\begin{gathered}x(t) = x_0 + \int_0^T V(t) \cdot cos\theta dt \\ y(t) = y_0 + \int_0^T V(t) \cdot sin\theta dt\end{gathered}$$

Using the half-angle formula we can obtain $sin\theta$ and $cos\theta$

$$sin\theta(\tau) = \frac{2 \cdot r(\tau)}{1 + r(\tau)^2}, \;\; cos\theta(\tau) = \frac{1 - r(\tau)^2}{1 + r(\tau)^2}$$

Transform t in the equation to dimensionless time τ

$$\tau = \frac{t}{T}$$
$$\Rightarrow t = \tau T$$
$$\Rightarrow dt = T d\tau$$

Plugging the above equations to $x(t)$ and $y(t)$ we get:

$$x(\tau) = x_0 + T\int_0^1 \tilde{v}(\tau)\cdot(1+r(\tau)^2)\cdot\frac{1-r(\tau)^2}{1+r(\tau)^2}d\tau$$
$$y(\tau) = y_0 + T\int_0^1 \tilde{v}(\tau)\cdot(1+r(\tau)^2)\cdot\frac{2\cdot r(\tau)}{1+r(\tau)^2}d\tau$$

The transformed states also become spline functions of $\tilde{v}(\tau)$ and $r(\tau)$.

$$x(\tau) = x_0 + T\int_0^1 \tilde{v}(\tau)\cdot(1-r(\tau)^2)d\tau$$
$$y(\tau) = y_0 + T\int_0^1 \tilde{v}(\tau)\cdot(2\cdot r(\tau))\tau$$

To include the constraints on the steering angle, we first derive $\dot{\theta}$:

$$\frac{d(sin\theta(\tau))}{dt} = cos\theta(\tau)\dot{\theta}(\tau) = \frac{2\dot{r}}{1+r(\tau)^2} - \frac{4r(\tau)^2\dot{r}(\tau)}{(1+r(\tau)^2)^2}$$

$$\begin{aligned}\dot{\theta}(\tau) &= \frac{2\dot{r}(\tau)}{1-r(\tau)^2} - \frac{4r(\tau)^2\dot{r}(\tau)}{(1+r(\tau)^2)(1-r(\tau)^2)} \\ &= \frac{2\dot{r}(\tau)-2\dot{r}(\tau)r(\tau)^2}{(1-r(\tau)^2)(1+r(\tau)^2)} \\ &= \frac{2\dot{r}(\tau)}{1+r(\tau)^2}\end{aligned}$$

Plugging $\dot{\theta}(\tau)$ back to the bicycle model we can derive $tan(\delta)$:

$$tan(\delta) = \frac{2\cdot\dot{r}(\tau)\cdot L}{\tilde{v}(\tau)\cdot(1+r(\tau)^2)^2}$$

Because the numerator and denominator of this equation is zero at $\tau = 0$, the constraint of initial steering angle δ should be calculated using l'Hopital's rule .

$$\lim_{t\to 0}\frac{f(t)}{g(t)} = \lim_{t\to 0}\frac{f'(t)}{g'(t)}$$

$$\lim_{\tau\to 0} tan(\delta) = \lim_{\tau\to 0}\frac{2\cdot\ddot{r}(\tau)\cdot L}{\dot{\tilde{v}}(\tau)\cdot(1+r(\tau)^2)^2 + 4\cdot\tilde{v}(\tau)\cdot(1+r(\tau)^2)\cdot r(\tau)\cdot\dot{r}(\tau)}$$

The relation ship of velocity of steering angle can be obtained through the time derivative of $tan(\delta)$

$$\frac{d\,tan(\delta)}{d\tau} = \frac{1}{cos(\delta)^2}\dot{\delta}$$
$$= \frac{2 \cdot \ddot{r}(\tau) \cdot L}{\tilde{v}(\tau) \cdot (1 + r(\tau)^2)^2} - \frac{2 \cdot r(\tau) \cdot L \cdot (\dot{\tilde{v}}(\tau) \cdot (1 + r(\tau)^2)^2 + 4 \cdot \tilde{v}(\tau) \cdot (1 + r(\tau)^2) \cdot r(\tau) \cdot \dot{r}(\tau))}{\tilde{v}(\tau)^2 \cdot (1 + r(\tau)^2)^4}$$

calculate $cos(\delta)$

$$cos(\delta)^2 = \frac{1}{1 + tan(\delta)^2} = \frac{1}{1 + \frac{4\dot{r}(\tau)^2 \cdot L^2}{\tilde{v}(\tau)^2 \cdot (1 + r(\tau)^2)^4}} = \frac{\tilde{v}(\tau)^2 \cdot (1 + r(\tau)^2)^4}{\tilde{v}(\tau)^2 \cdot (1 + r(\tau)^2)^4 + 4\dot{r}(\tau)^2 \cdot L^2}$$

Plugging $cos(\delta)$ back to the time derivative of $tan(\delta)$ to solve $\dot{\delta}$

$$\dot{\delta} = \frac{p(\tau) - 2 \cdot L \cdot \dot{r}(\tau) \cdot (q(\tau) + o(\tau))}{w(\tau)}$$
$$p(\tau) = 2 \cdot L \cdot \ddot{r}(\tau) \cdot \tilde{v}(\tau) \cdot (1 + r(\tau)^2)^2$$
$$q(\tau) = (4 \cdot r^3(\tau) \cdot \dot{r}(\tau) + 4 \cdot r(\tau) \cdot \dot{r}(\tau)) \cdot \tilde{v}(\tau)$$
$$o(\tau) = \dot{\tilde{v}} \cdot (1 + r^2(\tau))^2$$
$$w(\tau) = \tilde{v}(\tau)^2 \cdot (1 + r(\tau)^2)^4 + 4\dot{r}(\tau)^2 \cdot L^2$$

We believe that the T^2in $w(\tau)$ in the paper is a typo.

After transformation, all constraints of the bicycle kinematic models, including $V(\tau)$, $\dot{\delta}(\tau)$, and the state $q = [x, y, \delta]^T$, have become smooth functions of spline function $r(\tau)$ and $\tilde{v}(\tau)$. We can thus find optimal spline trajectory for $r(\tau)$ and $\tilde{v}(\tau)$ that guarantee following all the constraints.

C. Collision Avoidance Constraints

The constraints of collision avoidance express that the vehicle and the obstacle should remain a distance of a safety factor ϵ.

$$dist(veh(q(t)), obs_k(p_k^{pred}(t))) \geq \epsilon \quad \forall t \in [0, T]$$

This constraints can be described using hyperplane theorem (Appendix C). The theorem states that two non intersecting convex sets can always be separated by a hyperplane.

The parameters of the hyperplane are parameterised as spline curves over time.

$$a(\tau) = \sum_{i=1}^{n} c_i^a \cdot B_i^1(\tau),\ b(\tau) = \sum_{i=1}^{n} c_i^b \cdot B_i^b(\tau)$$

c_i^a and c_i^b are additional optimisation variables.

obstacle_hyperplane.gif

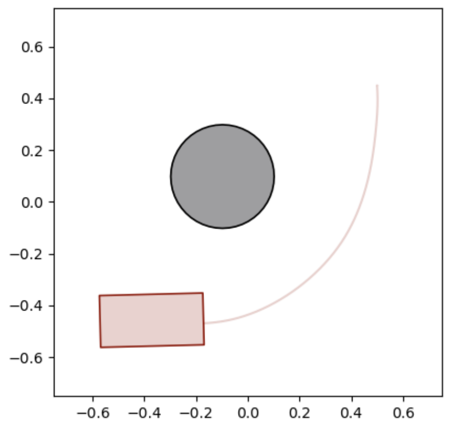


Fig. 7 Vehicle (Red) planning path while keeping 0.2 m distance from the obstacle.

Let $z_j^{veh}(\tau)$ be the vertices of the vehicle's shape in the local vehicle frame. The coordination $\hat{z}_j^{veh}(\tau)$ in the global frame can be computed by:

$$\hat{z}_j^{veh}(\tau) = [x(\tau),\ \ y(\tau)]\ \ +\ \ R(\tau)\cdot z_j^{veh}$$

$[x(\tau), y(\tau)]$ is the position of the vehicle, and $R(\tau)$ is the rotation matrix of vertics related to the vehicle base over time. $R(\tau)$ can be descrive using the previous equation of $sin\theta(\tau)$ and $cos\theta(\tau)$.

$$R(\theta) = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}$$

$$R(\tau) = \begin{bmatrix} \frac{1-r(\tau)^2}{1+r(\tau)^2} & -\frac{2\cdot r(\tau)}{1+r(\tau)^2} \\ \frac{2\cdot r(\tau)}{1+r(\tau)^2} & \frac{1-r(\tau)^2}{1+r(\tau)^2} \end{bmatrix}$$

The constrant of the hyperplane deduces as follow:

$$\begin{gathered} a(\tau)\cdot p^{pred}(\tau) - b(\tau) \geq d(\tau) - r^{obs}, \\ (1+r^2(\tau))\cdot(a(\tau)^T\cdot \hat{z}_j^{veh}(\tau) - b(\tau)) \leq 0, \\ \|x(\tau)\|_2 \leq 1, \\ j = 1...4,\ \forall\tau\in[0,1] \end{gathered}$$

$p^{pred}(\tau)$ gives the position of the circle center and r^{obs} represents its radius. The first line states that the obstacle has to be at least $d(\tau) - r^{obs}$ far to the left of the hyperplane. The $(1+r^2(\tau))$ in second line will cancel out the denominators in $R(\tau)$.
$d(\tau)$ are also a spline functions that can be considered as the distance between the obstacles and the vehicle. An extra term $\|d(\tau) - \epsilon\|$ can be added to the final objective. This will prevent $d(\tau)$ trying to

deviate from ϵ. The whole collision constraints will introduce new spline coefficient c_i^a, c_i^b, c_i^d for optimisation.

Experiments

To demonstrate the working of this method, we implemented following few examples using a python library called OMG-tools which was used by the authors of this paper.

A. Holonomic vehicle

In this example a circular holonomic vehicle moves on straight path in optimal time.

no_obstacle.gif

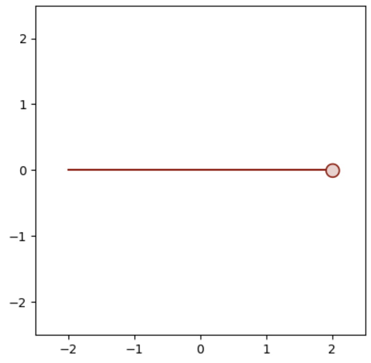


Fig. 8 Holonomic vehicle moving in straight path

B. Holonomic vehicle with static obstacle

A static obstacle with width $0.5m$ and height $3.0m$ was introduced in above example.

obstacle avoidance gif

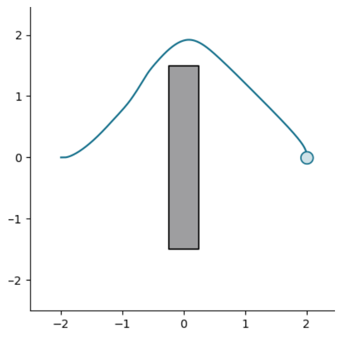


Fig. 9 Holonomic vehicle avoiding obstacle in the path

C. Non-holonomic vehicle

For demonstration of non-holonomic vehicles, we used a differential drive robot and limited it's linear and angular velocities to 0.5m/s and $\pm\pi/5$ rad/s.

non-holonomic trajectory planning gif

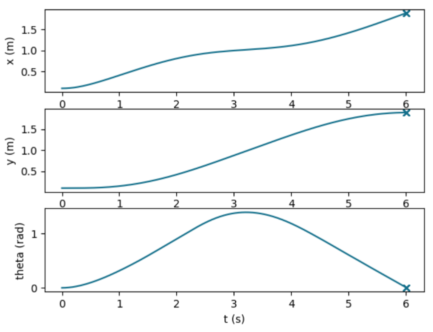


Fig. 10 Image showing graph of state Trajectory $q = [x, y, \theta]$

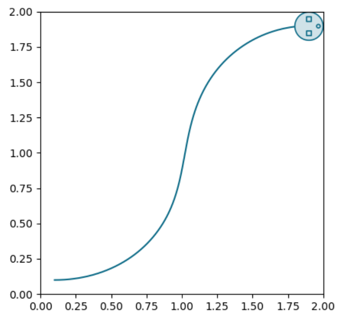


Fig. 11 Non holonomic trajectory of differential drive with time optimal path $q(t)$

D. Holonomic vehicle with dynamic collision avoidance

To recreate the demo of what the author has demonstrated in paper (Fig. 9 in paper) we created a similar simulation setup and started moving the obstacle when the vehicle is on the initial path. It can be seen that as soon as the moving obstacle gets registered in the system, the vehicle plans and executes new trajectory in the runtime.

dynamic obstacle avoidance gif

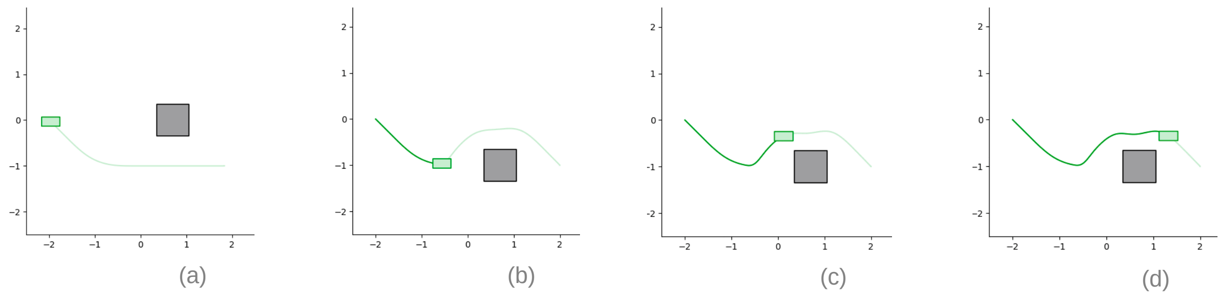


Fig. 12 Holonomic vehicle along with a moving obstacle. (a) t = 0.0 s. (b) t = 2.0 s. (c) t = 3.2 s. (d) t = 4.5 s

Conclusion

Spline-based motion planning provides an alternative coupled trajectory planning methods that consider all the constraints, including robot kinematic and collision avoidance at the planning stage while remaining fast and flexible. Trade-off between conservative and computation complexity can be made by setting knots or degree of basis function. It is suitable for solving robots with kinematic constraints in a dynamic environment.

Bibliography

[1] Michel Fliess, JeanLevine**,** Fliess, Michel, et al. "Flatness and defect of non-linear systems: introductory theory and examples." *International journal of control* 61.6 (1995): 1327-1361.

[2] Flatness wiki https://en.wikipedia.org/wiki/Flatness_(systems_theory)

[3] Differentially Flat System http://msl.cs.uiuc.edu/planning/node850.html

[4] Van Loock, Wannes, Goele Pipeleers, and Jan Swevers. "B-spline parameterized optimal motion trajectories for robotic systems with guaranteed constraint satisfaction." *Mechanical Sciences* 6.2 (2015): 163-171.

[5] Mercy, Tim, Wannes Van Loock, and Goele Pipeleers. "Real-time motion planning in the presence of moving obstacles." *2016 European Control Conference (ECC).* IEEE, 2016.

[6] B-Splines - JSXGraph Wiki. http://jsxgraph.uni-bayreuth.de/wiki/index.php/B-splines. Accessed 17 Nov. 2021. (B-Splines - JSXGraph Wiki)

[7] Kamermans, Mike “Pomax.” “A Primer on Bézier Curves.” Pomax.Github.Io, 13 June 2013, https://pomax.github.io/bezierinfo. (Kamermans)

[8] Wikimedia projects. “Bézier Curve - Wikipedia.” Wikipedia, the Free Encyclopedia, Wikimedia Foundation, Inc., 23 Nov. 2001, https://en.wikipedia.org/wiki/B%C3%A9zier_curve. (Wikimedia projects)

[9] Wikipedia. “B-Spline - Wikipedia.” Wikipedia, the Free Encyclopedia, Wikimedia Foundation, Inc., 25 Oct. 2001, https://en.wikipedia.org/wiki/B-spline. (Wikipedia)

[10] Mark W.Spong, Seth Hutchinson, M.Vidyasagar. "Robot Mmodeling and Control"

[11] Derivatives of a B-spline Curve https://pages.mtu.edu/~shene/COURSES/cs3621/NOTES/spline/B-spline/bspline-derv.html

[12] L'Hôpital's rule wiki https://en.wikipedia.org/wiki/L'Hôpital's_rule

[13] Deduce the recursive formula of B-splinne basis https://math.stackexchange.com/questions/1486833/how-to-deduce-the-recursive-derivative-formula-of-b-spline-basis

[14] Non-uniform rational B-spline wiki https://en.wikipedia.org/wiki/Non-uniform_rational_B-spline

[15] Optimal Motion Generation Tools https://github.com/meco-group/omg-tools

Appendix

A. Example of Differentially flat system

We can formulate the differential flat system below into an optimal control problem.

Consider the case of a flexible link manipulator

The dynamics of the manipulator are described by the equations

$$I_1\ddot{q}_1 + MgLsinq_1 + k(q_1 - q_2) = 0,$$
$$I_2\ddot{q}_2 - k(q1 - q2) = u$$

The system's state is given by $x = (q_1, \dot{q}_1, q2, \dot{q}_2)^T$.
The goal is to steer the system from the
initial state $x_0 = (0.8\ rad, 0\ rads^{-1}, -0.67\ rad, 0\ rads^-1)$
to the final state $x_{tf} = (-0.8\ rad, 0\ rads^{-1}, -0.67\ rad, 0\ rads^{-1})$
at tf = 5.35s with minimal deflection of the link, i.e.$g = \int_0^{t_f} q_1^2(t)dt$, while obeying the constraint on the joint positions

$$-\frac{\pi}{3} \le q_1 \le \frac{\pi}{3}, -\frac{\pi}{4} \le q_2 \le \frac{\pi}{4}, -\frac{\pi}{16} \le q_2 - q_1 \le \frac{\pi}{16}$$

The state vector is described by the flat output, y=q1 as:

$$x = \psi_x(y, \dot{y}, \ddot{y}, \dddot{y}) = (y,\ \dot{y},\ \frac{I_1}{k}\ddot{y} + \frac{MgL}{k}siny + y,\ \frac{I_1}{k}y^{(3)} + \frac{MgL}{k}\dot{y}siny + \dot{y})$$

The mapping ψ_x is not polynomial. Therefore, in order to use the proposed approach, the constraints must be approximated or manipulated into polynomial expressions. To this end, we search for polynomial lower and upper bounds such that

$$p(y)_{low} \le siny \le p(y)_{up},\ \ \forall y \in \left[-\frac{\pi}{3}, \frac{\pi}{3}\right]$$

The constraints on $q2$ and $q2 - q1$ can now be drvied as follow:

$$\frac{I_1}{k}\ddot{y} + \frac{MgL}{k}p(y)_{up} + y \le \frac{\pi}{4}, \qquad \frac{I_1}{k}\ddot{y} + \frac{MgL}{k}p(y)_{up} \le \frac{\pi}{16}$$
$$\frac{I_1}{k}\ddot{y} + \frac{MgL}{k}p(y)_{low} + y \ge -\frac{\pi}{4}, \qquad \frac{I_1}{k}\ddot{y} + \frac{MgL}{k}p(y)_{low} \ge -\frac{\pi}{16}$$

The whole optimal control problem can now be written as:

$$\begin{aligned}
&\underset{y(.)}{min} \quad \int_0^{t_f} y(t)^2(t)dt \\
s.t. \ & y(0) = 0.8, \ \ y(t_f) = -0.8 \\
& \dot{y}(0) = 0, \ \ \dot{y}(t_f) = 0 \\
& y^{(3)}(0) = 0, \ \ y^{(3)}(t_f) = 0 \\
& \frac{I_1}{k}\ddot{y} + \frac{MgL}{k}p(y)_{up} + y \le \frac{\pi}{4}, \\
& \frac{I_1}{k}\ddot{y} + \frac{MgL}{k}p(y)_{low} + y \ge -\frac{\pi}{4} \\
& \frac{I_1}{k}\ddot{y} + \frac{MgL}{k}p(y)_{up} \le \frac{\pi}{16}, \\
& \frac{I_1}{k}\ddot{y} + \frac{MgL}{k}p(y)_{low} \ge -\frac{\pi}{16} \\
& \qquad \forall t \in [0, t_f]
\end{aligned}$$

The flat output is now parameterized by a polynomial spline with 11 knots due to 11 constraints, including the optimization function.

B. Derivatives of a B-spline Curve

Suppose a Spline curve is defined as follows :

$$Q(t) = \sum_{i=1}^{n} c_i \cdot N_{i,p}(t)$$

c_i is the spline coefficient and the $N_{i,p}$ represents the ith basis spline function of degree p

$$N_{i,0}(x) := \begin{cases} 1 \ \text{ if } \ t_i \le t \le t_{i+1} \\ 0 \ \text{ otherwise} \end{cases}$$
$$N_{i,p}(t) = \frac{t - t_i}{t_{i+p} - t_i} N_{i,p-1}(t) + \frac{t_{i+p+1} - t}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(t)$$

The derivative of each of these basis function can be computed as follows:

$$\frac{d}{dt}N_{i,p}(t) = N'_{i,p}(t) = \frac{p}{t_{i+p} - t_i} N_{i,p-1}(t) - \frac{p}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(t)$$

The derivative of a basis function is actually the linear combination of another B-spline basis that is 1 order below, which is p-1 in this case.

Plugging back to the spline curve equation yields the following result:

$$\frac{d}{du}Q(t) = Q'(t) = \sum_{i=1}^{n-1} c'_i \cdot N_{i+1,p-1}(t)$$
$$c'_i = \frac{p}{t_{i+p+1} - t_{i+1}}(c_{i+1} - c_i)$$

C. Hyperplane Separation TheoremS

Let A and B be two disjoint nonempty convex subsets of $\mathbb{R}^n$.
Then there exist a nonzero vector v and a real number c such that
$\langle x, v\rangle \geq c$, *and* $\langle y, v\rangle \leq c$ for all x in A and y in B;
ie., the hyperplane $\langle \cdot, v\rangle = c$, v the normal vector, separates A and B

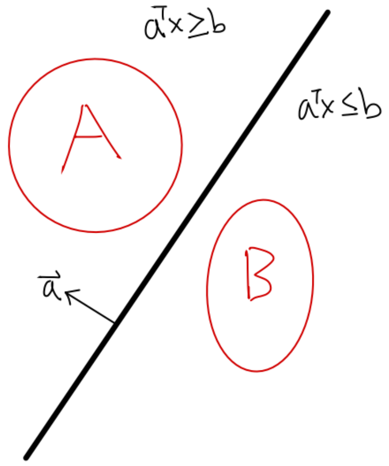


Fig. 13 Representation of hyper plane separating object A and B