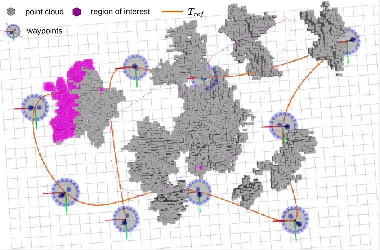
Motion Planning for Autonomous Vehicles

Curve Fitting

Geesara Kulathunga

March 25, 2023



Curve Fitting

Contents

The least-squares method can be used to fit *nth* order fitting

- Consider a set $(x_i, y_i), \, , i = 0, \ldots, m-1$

The least-squares method can be used to fit *nth* order fitting

- Consider a set (x_i, y_i), , $i = 0, \ldots, m-1$
- Define *nth* order fitting curve

$$a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$$

N DEGREE POLYNOMIAL FITTING

The least-squares method can be used to fit *nth* order fitting

- Consider a set $(x_i, y_i), , i = 0, ..., m-1$
- Define *nth* order fitting curve

$$a_0 + a_1x + a_2x^2 + ... + a_nx^n$$

- Convert to mean square error

$$J(x, y) = \Sigma_{i=0}^{m-1}(a_0 + a_1x_i + a_2x_i^2 + ... + a_nx_i^n - y_i)^2$$

The least-squares method can be used to fit *nth* order fitting

- Consider a set $(x_i, y_i), \, , i = 0, \ldots, m-1$
- Define *nth* order fitting curve

$$a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n$$

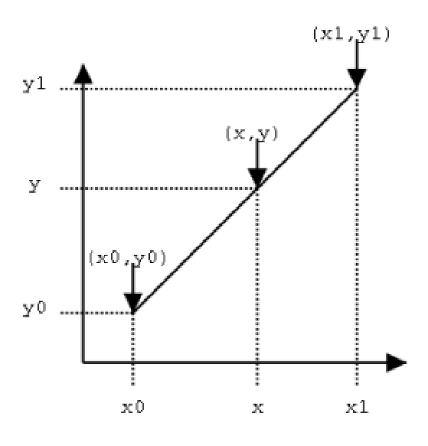
- Convert to mean square error

$$J(x, y) = \Sigma_{i=0}^{m-1} (a_0 + a_1 x_i + a_2 x_i^2 + \ldots + a_n x_i^n - y_i)^2$$

- Solve it

$$A^\top A \begin{bmatrix} a_0 \\ a_1 \\ \ldots \\ a_n \end{bmatrix} = A^\top \begin{bmatrix} y_1 \\ y_2 \\ \ldots \\ y_m \end{bmatrix}, A = \begin{bmatrix} 1 & x_1 & \ldots & x_1^n \\ 1 & x_2 & \ldots & x_2^n \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_m & \ldots & x_m^n \end{bmatrix}$$

Linear interpolation



$$f(x) = f(x_0) + (x - x_0)\frac{f(x_0) - f(x_1)}{x_0 - x_1} \tag{1}$$

A **solution** of the **Euler-Lagrange** equation is called an **extremal** (minimum or maximum) of the **functional**. If Lagrangian $L(x, \dot{x})$ depends only on first-order derivatives, a second-order equation of motion can be found where only two boundary conditions are required, e.g., the **position** of the vehicle at an initial and final time. Such a **condition fixes the endpoint**. However, if Lagrangian $L(x, \dot{x}, \ddot{x})$ depends on second-order derivatives, a fourth-order equation of motion can be found. Hence, it requires four boundary conditions and **fixing the velocity** (as well as the **position**) at the **initial** and **final** time. Euler-Lagrange equations for a Lagrangian $L(x, \dot{x}, \ddot{x}, \dots)$ are given by

$$\frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) + \cdots + (-1)^n \frac{d^n}{dt^n}\left(\frac{\partial L}{\partial x^{(n)}}\right) = 0. \tag{2}$$

Euler–Lagrange equation

Consider the AGV moves between the two positions within the time interval T.

$$J = \int_0^T L dt \tag{3}$$

where T is the motion duration and the performance index is L, the minimum jerk trajectory for unconstrained point-to-point movement, is

$$L = \left(\frac{\partial^3 x}{\partial t^3}\right)^2 + \left(\frac{\partial^3 y}{\partial t^3}\right)^2 \tag{4}$$

where, x and y indicates the position components. The objective is to deduce the local path minimizing the cost function J.

Jerk is **the time derivation** of **acceleration**. Jerk is the way to define comfortness mathematically (or suppressing vibration effects or sudden acceleration change). Additionally, the first and second derivatives are continuous, so **continuous velocity and curvature are satisfied**.

Minimum jerk trajectory (MJT) generation

- To solve this, Euler–Lagrange equation can be utilized.

$$
\begin{aligned}
\frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) + \cdots + (-1)^n \frac{d^n}{dt^n}\left(\frac{\partial L}{\partial x^{(n)}}\right) = 0, \\
\frac{\partial L}{\partial y} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{y}}\right) + \cdots + (-1)^n \frac{d^n}{dt^n}\left(\frac{\partial L}{\partial y^{(n)}}\right) = 0.
\end{aligned} \tag{5}
$$

[1].https://courses.shadmehrlab.org/Shortcourse/minimumjerk.pdf

Minimum jerk trajectory (MJT) generation

- To solve this, Euler–Lagrange equation can be utilized.

$$
\begin{aligned}
\frac{\partial L}{\partial x} - \frac{d}{dt}\Big(\frac{\partial L}{\partial \dot{x}}\Big) + \cdots + (-1)^n \frac{d^n}{dt^n}\Big(\frac{\partial L}{\partial x^{(n)}}\Big) = 0, \\
\frac{\partial L}{\partial y} - \frac{d}{dt}\Big(\frac{\partial L}{\partial \dot{y}}\Big) + \cdots + (-1)^n \frac{d^n}{dt^n}\Big(\frac{\partial L}{\partial y^{(n)}}\Big) = 0.
\end{aligned}
\tag{5}
$$

- Since jerk has to be minimized, $\frac{d}{dt}\Big(\frac{\partial \dddot{x}^2}{\partial \dddot{x}}\Big) = 0$ and $\frac{d}{dt}\Big(\frac{\partial \dddot{y}^2}{\partial \dddot{y}}\Big) = 0$ must be satisfied[1]. Hence,

$$
\frac{d^6 x}{dt^6} = 0, \quad \frac{d^6 y}{dt^6} = 0 \tag{6}
$$

[1].https://courses.shadmehrlab.org/Shortcourse/minimumjerk.pdf

- To solve this, Euler–Lagrange equation can be utilized.

$$\begin{aligned}\frac{\partial L}{\partial x} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) + \cdots + (-1)^n \frac{d^n}{dt^n}\left(\frac{\partial L}{\partial x^{(n)}}\right) = 0, \\ \frac{\partial L}{\partial y} - \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{y}}\right) + \cdots + (-1)^n \frac{d^n}{dt^n}\left(\frac{\partial L}{\partial y^{(n)}}\right) = 0.\end{aligned} \tag{5}$$

- Since jerk has to be minimized, $\frac{d}{dt}\left(\frac{\partial \dddot{x}^2}{\partial \ddot{x}}\right) = 0$ and $\frac{d}{dt}\left(\frac{\partial \dddot{y}^2}{\partial \ddot{y}}\right) = 0$ must be satisfied[1]. Hence,

$$\frac{d^6 x}{dt^6} = 0, \quad \frac{d^6 y}{dt^6} = 0 \tag{6}$$

- Therefore, $x(t)$ and $y(t)$ must having the 5th order polynomial as follows:

$$\begin{aligned} x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \\ y(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5 \end{aligned} \tag{7}$$

[1].https://courses.shadmehrlab.org/Shortcourse/minimumjerk.pdf

A polynomial of degree five defines a quintic function.

$$\begin{aligned} x(t) &= a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \\ y(t) &= b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5 \end{aligned} \tag{8}$$

Consider the initial condition $x_0, y_0, \dot{x}_0, \dot{y}_0, \ddot{x}_0, \ddot{y}_0$ at $t = 0$ and final condition $x_f, y_f, \dot{x}_f, \dot{y}_f, \ddot{x}_f, \ddot{y}_f$ at $t = T$ are given.

Quintic polynomial curve fitting **decouples** the along **x** and **y** directions, however, **position, velocity acceleration and jerk** are solved by **coupling**.

Hence,

$$a_0 = x_0, \quad a_1 = \dot{x}_0, \quad a_2 = \ddot{x}_0/2$$
$$A = \begin{bmatrix} t^3 & t^4 & t^5 \\ 3t^2 & 4t^3 & 5t^4 \\ 6t & 12t^2 & 20t^3 \end{bmatrix}, \quad b = \begin{bmatrix} x_f - a_0 - a_1 - a_2 t^2 \\ \dot{x}_f - a_1 - 2a_2 t \\ \ddot{x}_f - 2a_2 \end{bmatrix} \tag{9}$$
$$\begin{bmatrix} a_3 \\ a_4 \\ a_5 \end{bmatrix} = A^{-1} b$$

similar way $b_0, \ldots, b_5$ can be calculated. The higher-order derivatives can be estimated as follows:

$$\begin{aligned} x(t) &= a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \\ \dot{x}(t) &= a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4 \\ \ddot{x}(t) &= 2a_2 + 6a_3 t + 12a_4 t^2 + 20a_5 t^3 \end{aligned} \tag{10}$$

- Given a set of points: $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^2$, to define a Lagrange polynomial, it is required to define a set of cardinal functions: $l_1, l_2, .., l_n \in \mathbb{P}^n$ such that

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{11}$$

for $\forall\, i \in [0, \ldots, n]$. Term δ_{ij} is called Kronecker's delta.

- Given a set of points: $(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n) \in \mathbb{R}^2$, to define a Lagrange polynomial, it is required to define a set of cardinal functions: $l_1, l_2, .., l_n \in \mathbb{P}^n$ such that

$$l_i(x_j) = \delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases} \tag{11}$$

  for $\forall\, i \in [0, \ldots, n]$. Term δ_{ij} is called Kronecker's delta.
- Term $\mathbb{P}^n$, denoted polynomial of nth order.

$$\begin{aligned} l_i(x) &= \prod_{j=0, j\neq i}^{n} \left(\frac{x - x_j}{x_i - x_j} \right) \\ &= \frac{x - x_0}{x_i - x_0} \cdot \frac{x - x_1}{x_i - x_1} \cdots \frac{x - x_n}{x_i - x_n} \end{aligned} \tag{12}$$

Both conditions: $l_i(x_i) = 1$ and $l_i(x_k) = 0$, $i \neq k$ can be verified.

- Therefore, the Lagrange form of a polynomial interpolation can be defined as

$$P_n(x) = \Sigma_{i=0}^{n} l_i(x) \cdot y_i \tag{13}$$

- Therefore, the Lagrange form of a polynomial interpolation can be defined as

$$P_n(x) = \Sigma_{i=0}^{n} l_i(x) \cdot y_i \tag{13}$$

- With that interpolation property is expressed as

$$P_n(x_j) = \Sigma_{i=0}^{n} l_i(x_j) \cdot y_i = y_j \tag{14}$$

Lagrange polynomials

- Formulate a second-order Lagrange polynomial that goes through these points: $x, y = (0, 1), (2/3, 0.5), (1, 0)$.

- Formulate a second-order Lagrange polynomial that goes through these points: $x, y = (0,1), (2/3, 0.5), (1,0)$.

- Define cardinal functions $l_0(x), l_1(x), l_2(x)$, afterwards Lagrange polynomial can be determined as

$$p_2(x) = l_0(x)y_0 + l_1(x)y_1 + l_2(x)y_2 \tag{15}$$

- Formulate a second-order Lagrange polynomial that goes through these points: $x, y = (0, 1), (2/3, 0.5), (1, 0)$.

- Define cardinal functions $l_0(x), l_1(x), l_2(x)$, afterwards Lagrange polynomial can be determined as

$$p_2(x) = l_0(x)y_0 + l_1(x)y_1 + l_2(x)y_2 \tag{15}$$

- The main **disadvantage** of the **Lagrange polynomial** is that **adding or removing a new point, it has to recompute all the** $l_i's$

Lagrange first order interpolation and second order interpolation

- Lagrange first-order interpolation

$$
\begin{aligned}
f(x) &= f(x_0) + (x - x_0)\frac{f(x_0) - f(x_1)}{x_0 - x_1} \\
&= \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1)
\end{aligned} \tag{16}
$$

Lagrange first order interpolation and second order interpolation

- Lagrange first-order interpolation

$$
\begin{aligned}
f(x) &= f(x_0) + (x - x_0)\frac{f(x_0) - f(x_1)}{x_0 - x_1} \\
&= \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1)
\end{aligned}
\tag{16}
$$

- Lagrange second-order interpolation

$$
\begin{aligned}
f(x) = f(x_0) + (x - x_0)\frac{f(x_0) - f(x_1)}{x_0 - x_1} &+ (x - x_0)(x - x_1)\frac{f(x_0, x_1) - f(x_1, x_2)}{x_0 - x_1} \\
= \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} f(x_0) &+ \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} f(x_1) \\
+ \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} f(x_2)
\end{aligned}
\tag{17}
$$

$$f(x) = f(x_0)\delta_0(x) + f(x_1)\delta_1(x) + \ldots + f(x_n)\delta_n(x) \tag{18}$$

where $\delta_i(x)$ can be determined as

$$\delta_i(x) = \prod_{j=0, j\neq i}^{n} \left(\frac{x - x_j}{x_i - x_j} \right) \tag{19}$$

Some example: `https://polympc.readthedocs.io/en/latest/ocp.html`

Variation of calculus to spline fitting

Given

$$
\begin{aligned}
\min_{f:[0,1]\to\mathbb{R}} \quad & \int_0^1 \left[f^{(2)}(t)\right]^2 dt \\
\text{s.t.} \quad & f(0)=a,\ f^{(1)}(0)=c \\
& f(1)=b,\ f^{(1)}(1)=d
\end{aligned}
\tag{20}
$$

Variation of calculus to spline fitting

- The objective

$$J(f) = \int_0^1 \left[f^{(2)}(t) \right]^2 dt$$

Variation of calculus to spline fitting

- The objective

$$J(f) = \int_0^1 \left[f^{(2)}(t) \right]^2 dt$$

- After considering the perturbation

$$J(f, p) = \int_0^1 \left[f^{(2)}(t) + h p^{(2)}(t) \right]^2 dt$$

Variation of calculus to spline fitting

- The objective

$$J(f) = \int_0^1 \left[f^{(2)}(t)\right]^2 dt$$

- After considering the perturbation

$$J(f,p) = \int_0^1 \left[f^{(2)}(t) + hp^{(2)}(t)\right]^2 dt$$

- Gateaux derivative

$$\begin{aligned} dJ(f,p) &= \frac{d}{dh}\left(\int_0^1 \left[f^{(2)}(t) + hp^{(2)}(t)\right]^2 dt\right)_{h=0} \\ &= \int_0^1 2\left[f^{(2)}(t) + hp^{(2)}(t)\right] p^{(2)}(t)|_{h=0} dt \\ &= \int_0^1 2f^{(2)}(t)p^{(2)}(t)dt \end{aligned}$$

Taking integration by parts

$$
\begin{aligned}
dJ(f,p) =& [2f^{(2)}(t)p^{(1)}(t)]_0^1 - [2f^{(3)}(t)p(t)]_0^1 + \int_0^1 2f^{(4)}(t)p(t)dt \\
=& \int_0^1 2f^{(4)}(t)p(t)dt
\end{aligned}
$$

If $f(t)$ is optimal for the considered constraint problem, then $dJ(f,p) = 0$ as long as , $f(t) + hp(t)$ is feasible for small $h \Rightarrow p(0) = p(1) = p^{(1)}(0) = p^{(1)}(1) = 0$. The function $p(t)$ can have infinitely many forms. Therefore, to obtain $dJ(f,p) = 0$, $f^{(4)}(t) = 0$. Hence, $f(t) = a^0 + a^1 t + a^2 t^2 + a^3 t^3$.

Only consider sub-interval without considering the **whole polynomial** as formulated in Lagrange nth order interpolation. Let $S(t)$ be interpolated function through a given set of points $(t_i, y_i)_{i=0}^{n}$. The ordered set $t_0 < t_1 < \ldots < t_n$ is called knots vector. Hence, $S(t)$ contains a set of piece-wise polynomials

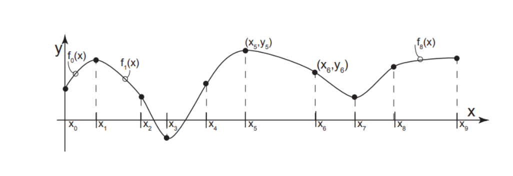
$$S(t) = \begin{cases} S_0(t), & t_0 \le t \le t_1 \\ S_1(t), & t_1 \le t \le t_2 \\ \vdots & \\ S_{n-1}(t) & t_{n-1} \le t \le t_n \end{cases} \tag{21}$$

$S(t)$ is a polynomial of degree k, if and only if $S(t)$ is k-1 times continuous differentiable

$$S_{i-1}(t_i) = S_i(t_i), S'_{i-1}(t_i) = S'_i(t_i), \ldots S_{i-1}^{(k-1)}(t_i) = S_i^{(k-1)}(t_i), \tag{22}$$

When n equals **1 linear** Spline, equals **2 quadratic** Spline, and equals **3 cubic** spline

Spline: piece-wise interpolation



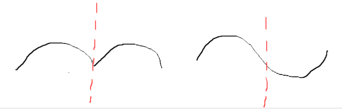
In general, $f(x_i) = a_i + b_i x + c_i x^2 + d_i x^3$, is the function which depicts the curve in between i^{th} and $i+1^{th}$ **control points**[1]. Hence, each curve represents by a cubic polynomial, with four coefficients for each. **How many parameters are to be solved?**

[1]. `https://people.cs.clemson.edu/~dhouse/courses/405/notes/splines.pdf`

Spline: piece-wise interpolation

Each segment pass through its control points
$f_i(x) = y_i,\ f_i(x_{i+1}) = y_{i+1}$

Consecutive segments should have the **same slop** and **same curvature** where they **join together** $f_i^{'}(x_{i+1}) == f_{i+1}^{'}(x_{i+1})$,
$f_i^{''}(x_{i+1}) = f_{i+1}^{''}(x_{i+1})$



How many parameters are to be solved?

Piece-wise linear interpolation, i.e., straight-line. The constraints are

$$
\begin{aligned}
S_0(t_0) &= y_0 \\
S_{i-1}(t_i) = S_i(t_i) = y_i, \quad i = 1, 2, \ldots, n-1, \quad \Rightarrow S_i(t) &= y_i + \frac{y_{i+1} - y_i}{t_{i+1} - t_i}(t - t_i) \\
S_{n-1}(t_n) &= y_n
\end{aligned}
\tag{23}
$$

Given ordered set $(t_i, y_i)_{i=0}^{n}$, cubic spline can be defined as

$$S(t) = S_i(t) \quad for \quad t_i \le t \le t_{i+1} \tag{24}$$

where $S_i(t) = d_i(t-t_i)^3 + c_i(t-t_i)^2 + b_i(t-t_i) + a_i, i = 0, 1, ..., n-1$. Thus, the total number of unknown 4n. However, the following constraints must be satisfied $S(t)$ is a polynomial of degree k=3, if and only if $S(t)$ is k-1 times continuous differentiable

$$\begin{aligned} S_i(t_i) = y_i, S_i(t_{i+1}) = y_{i+1}, i = 0, 1, \ldots, n-1 &\quad \Rightarrow 2 \cdot n \text{ equations} \\ S_i'(t_{i+1}) = S_{i+1}'(t_{i+1}), i = 0, 1, \ldots, n-2 &\quad \Rightarrow n-1 \text{ equations} \\ S_i^{(2)}(t_{i+1}) = S_{i+1}^{(2)}(t_{i+1}), i = 0, 1, \ldots, n-2 &\quad \Rightarrow n-1 \text{ equations} \\ S_0^{(2)}(t_0) = 0,\ S_{n-1}^{(2)}(t_n) = 0 &\quad \Rightarrow 2 \text{ equations} \end{aligned} \tag{25}$$

- Consider $z_i = S^{(2)}(t_i), \quad i = 1, 2, \dots, n-1, \quad z_0 = z_n = 0$. Since $S^{(2)}$ are linear functions, $S^{(2)}$ can be formulated in the Lagrange form

$$\begin{aligned} S_i^{(2)}(t) &= \frac{z_{i+1}}{t_{i+1} - t_i}(t - t_i) - \frac{z_i}{t_{i+1} - t_i}(t - t_{i+1}) \\ &= \frac{z_{i+1}}{h_i}(t - t_i) - \frac{z_i}{h_i}(t - t_{i+1}), \end{aligned} \tag{26}$$

  where term $h_i = t_{i+1} - t_i$.

- Consider $z_i = S^{(2)}(t_i), \quad i = 1, 2, \dots, n-1, \quad z_0 = z_n = 0$. Since $S^{(2)}$ are linear functions, $S^{(2)}$ can be formulated in the Lagrange form

$$\begin{aligned} S_i^{(2)}(t) &= \frac{z_{i+1}}{t_{i+1} - t_i}(t - t_i) - \frac{z_i}{t_{i+1} - t_i}(t - t_{i+1}) \\ &= \frac{z_{i+1}}{h_i}(t - t_i) - \frac{z_i}{h_i}(t - t_{i+1}), \end{aligned} \tag{26}$$

where term $h_i = t_{i+1} - t_i$.

- After integration, terms $S_i'(t)$ and $S_i(t)$ can be derived as follows:

$$\begin{aligned} S_i'(t) &= \frac{z_{i+1}}{2h_i}(t - t_i)^2 - \frac{z_i}{2h_i}(t - t_{i+1})^2 + C_i - D_i \\ S_i(t) &= \frac{z_{i+1}}{6h_i}(t - t_i)^3 - \frac{z_i}{6h_i}(t - t_{i+1})^3 + C_i(t - t_i) - D_i(t - t_{i+1}) \end{aligned} \tag{27}$$

- Considering interpolating properties

$$S_i(t_i) = y_i, \Rightarrow y_i = -\frac{z_i}{6h_i}(-h_i)^3 - D_i(-h_i) \Rightarrow D_i = \frac{y_i}{h_i} - \frac{h_i}{6}z_i$$

$$S_i(t+1) = y_{i+1}, \Rightarrow y_{i+1} = \frac{z_{i+1}}{6h_i}(-h_i)^3 + C_i(-h_i) \Rightarrow C_i = \frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}$$

$$\Rightarrow y_{i+1} = a_{i+1} = a_i + b_i h_i + c_i h_i^2 + d_i h_i^3 \tag{28}$$

- Considering interpolating properties

$$S_i(t_i) = y_i, \Rightarrow y_i = -\frac{z_i}{6h_i}(-h_i)^3 - D_i(-h_i) \Rightarrow D_i = \frac{y_i}{h_i} - \frac{h_i}{6}z_i$$

$$S_i(t+1) = y_{i+1}, \Rightarrow y_{i+1} = \frac{z_{i+1}}{6h_i}(-h_i)^3 + C_i(-h_i) \Rightarrow C_i = \frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1}$$

$$\Rightarrow y_{i+1} = a_{i+1} = a_i + b_i h_i + c_i h_i^2 + d_i h_i^3 \tag{28}$$

- Since D and C are known,

$$S_i(t) = \frac{z_{i+1}}{6h_i}(t-t_i)^3 - \frac{z_i}{6h_i}(t-t_{i+1})^3 + (\frac{y_{i+1}}{h_i} - \frac{h_i}{6}z_{i+1})(t-t_i) - (\frac{y_i}{h_i} - \frac{h_i}{6}z_i)(t-t_{i+1}) \tag{29}$$

$$S_i'(t) = \frac{z_{i+1}}{2h_i}(t-t_i)^2 - \frac{z_i}{2h_i}(t-t_{i+1})^2 + \frac{y_{i+1}-y_i}{h_i} - \frac{z_{i+1}-z_i}{6}h_i$$

- Continuity of $S^{'}(t)$ requires $S'_{i-1}(t_i) = S'_i(t_i), i = 1, \ldots, n-1$,

$$
\begin{aligned}
S'_i(t_i) &= -\frac{z_i}{2h_i}(-h_i)^2 + \underbrace{\frac{y_{i+1} - y_i}{h_i}}_{e_i} - \frac{z_{i+1} - z_i}{6} h_i \\
&= -\frac{1}{6} h_i z_{i+1} - \frac{1}{3} h_i z_i + e_i \qquad (30) \\
S'_{i-1}(t_i) &= \frac{1}{6} h_{i-1} z_{i-1} + \frac{1}{3} h_{i-1} z_i + e_{i-1}
\end{aligned}
$$

- Continuity of $S^{'}(t)$ requires $S'_{i-1}(t_i) = S'_i(t_i), i = 1, \ldots, n-1$,

$$
\begin{aligned}
S'_i(t_i) &= -\frac{z_i}{2h_i}(-h_i)^2 + \underbrace{\frac{y_{i+1} - y_i}{h_i}}_{e_i} - \frac{z_{i+1} - z_i}{6} h_i \\
&= -\frac{1}{6} h_i z_{i+1} - \frac{1}{3} h_i z_i + e_i \\
S'_{i-1}(t_i) &= \frac{1}{6} h_{i-1} z_{i-1} + \frac{1}{3} h_{i-1} z_i + e_{i-1}
\end{aligned} \tag{30}
$$

- Also, $S'_i(t_{i+1}) = S'_{i+1}(t_{i+1})$ and $S^{(2)}_i(t_{i+1}) = S^{(2)}_{i+1}(t_{i+1}), i = 0, \ldots, n-2$,

$$
\begin{aligned}
&\Rightarrow b_{i+1} = b_i + 2c_i h_i + 3d_i h_i^2 \\
&\Rightarrow c_{i+1} = 2c_i + 6d_i h_i
\end{aligned} \tag{31}
$$

- After setting them equal to each other,

$$\begin{cases} h_{i-1}z_{i-1} + 2(h_{i-1} + h_i)z_i + h_i z_{i+1} = 6(e_i - e_{i-1}), \quad i = 1, 2, ..., n-1 \\ z_0 = z_n = 0 \end{cases} \tag{32}$$

- After setting them equal to each other,

$$\begin{cases} h_{i-1}z_{i-1} + 2(h_{i-1}+h_i)z_i + h_i z_{i+1} = 6(e_i - e_{i-1}), & i = 1,2,\dots,n-1 \\ z_0 = z_n = 0 \end{cases} \tag{32}$$

- However, $z_i = S^{(2)}(t_i) = 2c_i$

$$\begin{cases} h_{i-1}c_{i-1} + 2(h_{i-1}+h_i)c_i + h_i c_{i+1} = 3(e_i - e_{i-1}), & i = 1,2,\dots,n-1 \\ z_0 = z_n = 0 \end{cases} \tag{33}$$

- Here both h_i both e_i are known, only the $\{c_i\}_{i=0}^n$ are unknown which can be solved by solving the following system of equations, where $\mathbf{A}$ is a $(n+1)\times(n+1)$ matrix and $\mathbf{Az}=\mathbf{b}$, in which $\mathbf{A}$

$$\mathbf{A}=\begin{pmatrix} 2(h_0+h_1) & h_1 & & & & \\ h_1 & 2(h_1+h_2) & h_2 & & & \\ & h_2 & 2(h_2+h_3) & h_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} \\ & & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{pmatrix}$$

- Here both h_i both e_i are known, only the $\{c_i\}_{i=0}^{n}$ are unknown which can be solved by solving the following system of equations, where $\mathbf{A}$ is a $(n+1)\times(n+1)$ matrix and $\mathbf{Az}=\mathbf{b}$, in which $\mathbf{A}$

$$\mathbf{A}=\begin{pmatrix} 2(h_0+h_1) & h_1 & & & & \\ h_1 & 2(h_1+h_2) & h_2 & & & \\ & h_2 & 2(h_2+h_3) & h_3 & & \\ & & \ddots & \ddots & \ddots & \\ & & & h_{n-3} & 2(h_{n-3}+h_{n-2}) & h_{n-2} \\ & & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) \end{pmatrix}$$

- However, after incorporating the boundary condition, i.e., $z_0 = S^{(2)}(t_i) = 2c_0 + 6d_0(t_0 - t_0) = 0 \Rightarrow c_0 = 0,\ c_n = 0.$

$$\mathbf{A}=\begin{pmatrix} 1 & 0 & & & & \\ h_0 & 2(h_0+h_1) & h_1 & & & \\ & h_1 & 2(h_1+h_2) & h_2 & & \\ & & \ddots & \ddots & \ddots & \\ & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ & & & 0 & 0 & 1 \end{pmatrix}$$

- In general, $\mathbf{A}$ is tri-diagonal, symmetric, and diagonal dominant. i.e., $2|h_{i-1}+h_i| > |h_i| + |h_{i-1}|$, which implies unique solution.

$$\mathbf{b} = \begin{pmatrix} 0 \\ 3(e_1 - e_0) \\ 3(e_2 - e_1) \\ \vdots \\ 3(e_{n-2} - e_{n-3}) \\ 3(e_{n-1} - e_{n-2}) \\ 0 \end{pmatrix}, \quad \mathbf{z} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-2} \\ c_{n-1} \\ c_n \end{pmatrix}$$

where $e_{i+1} - e_i = \frac{1}{h_{i+1}}(a_{i+2} - a_{i+1}) - \frac{1}{h_i}(a_{i+1} - a_i), \forall\, i = 0, \ldots, n.$

- In general, $\mathbf{A}$ is tri-diagonal, symmetric, and diagonal dominant. i.e., $2|h_{i-1}+h_i| > |h_i|+|h_{i-1}|$, which implies unique solution.

$$\mathbf{b} = \begin{pmatrix} 0 \\ 3(e_1-e_0) \\ 3(e_2-e_1) \\ \vdots \\ 3(e_{n-2}-e_{n-3}) \\ 3(e_{n-1}-e_{n-2}) \\ 0 \end{pmatrix}, \quad \mathbf{z} = \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{n-2} \\ c_{n-1} \\ c_n \end{pmatrix}$$

where $e_{i+1}-e_i = \frac{1}{h_{i+1}}(a_{i+2}-a_{i+1}) - \frac{1}{h_i}(a_{i+1}-a_i), \forall\ i = 0,\ldots,n.$

- Solving for d_i in eq.(31),

$$y_{i+1} = a_{i+1} = a_i + b_i h_i + \frac{h_i^2}{3}(2c_i + c_{i+1})$$

$$\Rightarrow b_i = \frac{1}{h_i}(a_{i+1}-a_i) - \frac{h_i}{3}(c_{i+1}+2c_i), \tag{34}$$

Example 02

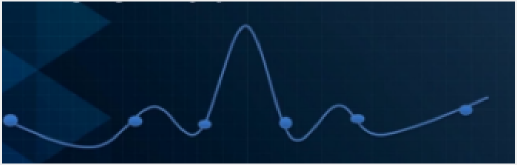
A fit cubic spline that passes these points: $(0,1),(1,e),(2,e^2),(3,e^3)$

Example 02

A fit cubic spline that passes these points: $(0,1),(1,e),(2,e^2),(3,e^3)$

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 \\ 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 0 \\ 3(e^2 - 2e + 1) \\ 3(e^3 - 2e^2 + e) \\ 0 \end{bmatrix}, \quad \mathbf{z} = \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} \tag{35}$$

Cubic splines are continuous and smooth at the **connecting points**.



- B-Spline: can generate control commands without smoothing
- Bezier
- Minimum-span
- Dubins curve: can not generate control commands without smoothing