Math 151A Project 2 Report

Jerry Liu, 404474229

May 27, 2016

1 User Guide

1.1 Function `cspline`

1.1.1 Purpose

The purpose of `cspline` function is to calculate the cubic spline for some given data points. This function uses natural boundary condition.

1.1.2 Input

The function `cspline` takes 2 arguments `X, F`. The vector `X` contains interpolating points $\{x_0, x_1, \ldots, x_n\}$; `F` is the corresponding function values at the interpolating points $\{f(x_0), f(x_1), \ldots, f(x_n)\}$.

1.1.3 Output

The function `cspline` returns a single variable `S`, a matrix of coefficients of cubic spline interpolation.

$$S = \begin{bmatrix} f_0 & b_0 & c_0 & d_0 \\ f_1 & b_1 & c_1 & d_1 \\ \vdots & \vdots & \vdots & \vdots \\ f_{n-1} & b_{n-1} & c_{n-1} & d_{n-1} \end{bmatrix}$$

We can easily get the corresponding cubic spline interpolation:

$$S(t) = \begin{cases} s_0(t) = f_0 + b_0(t-t_0) + c_0(t-t_0)^2 + d_0(t-t_0)^3 & t \in [t_0, t_1] \\ s_1(t) = f_1 + b_1(t-t_1) + c_1(t-t_1)^2 + d_1(t-t_1)^3 & t \in [t_1, t_2] \\ \vdots & \vdots \\ s_{n-1}(t) = f_{n-1} + b_{n-1}(t-t_{n-1}) + c_{n-1}(t-t_{n-1})^2 + d_{n-1}(t-t_{n-1})^3 & t \in [t_{n-1}, t_n] \end{cases}$$

1.2 Function `track`

1.2.1 Purpose

This function utilizes the `track` function to interpolate a cubic spline and approximate the path of the vehicle given a specific time.

1.2.2 Input

The function `track` takes 4 arguments `T, X, Y, t`.
The vector `T` contains interpolating time points $\{t_0, t_1, \ldots, t_n\}$;
`X` is the corresponding x values at corresponding time $\{x_0, x_1, \ldots, x_n\}$.
`Y` is the corresponding y values at corresponding time $\{y_0, y_1, \ldots, y_n\}$.
`t` is the time we want to approximate.

1.2.3 Output

The function `track` returns 2 variables, `fx` and `fy`.
`fx` gives the approximate x value at time t and `fy` gives the approximate y value at time t.

1.3 Preprocess

In order to load variables efficiently, we need to do some preprocessing.

```
load('data.mat');

T = ip(:, 1);
X = ip(:, 2);
Y = ip(:, 3);
```

So now we have three vectors

$$\begin{aligned} T &= \{t_0, t_1, \ldots, t_n\} \\ X &= \{x_0, x_1, \ldots, x_n\} \\ Y &= \{y_0, y_1, \ldots, y_n\} \end{aligned}$$

2 Solutions

We can use the following code to test our correctness of cubic spline function implementation.

```
tt = (linspace(0, 6.2, 10000))';
len = size(tt, 1);
dx = zeros(len, 1); dy = zeros(len, 1);

for i = 1 : len
    [dx(i) dy(i)] = track(T, X, Y, tt(i));
end

plot(dx, dy);
legend('Path of Vehicle');
```

From Figure 1, we can see that this matches with the plot from `plot(X, Y)`.

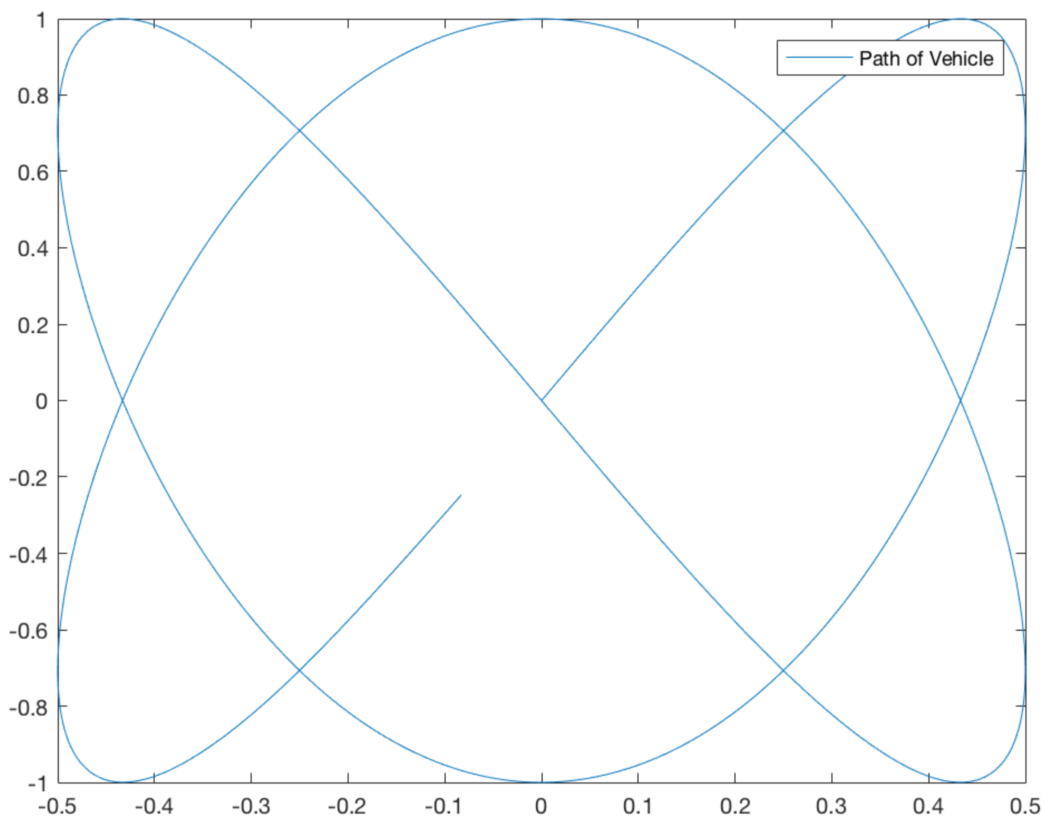

Figure 1: Plot of x path and y path of the tracked vehicle at time $t_0, t_1, \ldots, t_n$

3 Experiments

3.1 Velocity Calculation

We calculate the approximated speed at $t_0, t_1, \ldots, t_n$ with both cubic spline method and three-point methods with the code below.

```
len = size(T, 1); h = 0.2;
speed = @(S) S(:, 2) + 2 * S(:, 3) * h + 3 * S(:, 4) * h^2;
cx = cspline(T, X); cy = cspline(T, Y);
u = [cx(:, 2) cy(:, 2); speed(cx(len-1, :)) speed(cy(len-1, :))];

% Three point end point
v0 = [tpe([X(1); X(2); X(3)], h) ...
      tpe([Y(1); Y(2); Y(3)], h)];
vn = [tpe([X(len); X(len-1); X(len-2)], -h) ...
      tpe([Y(len); Y(len-1); Y(len-2)], -h)];

% Three point mid point

v = [v0; tpm(X, h) tpm(Y, h); vn];
```

Note: u, v are 2D vectors where

$$u = \begin{bmatrix} u_{x_0} & u_{y_0} \\ u_{x_1} & u_{y_1} \\ \vdots & \vdots \\ u_{x_n} & u_{y_n} \end{bmatrix} \text{ and } v = \begin{bmatrix} v_{x_0} & v_{y_0} \\ v_{x_1} & v_{y_1} \\ \vdots & \vdots \\ v_{x_n} & v_{y_n} \end{bmatrix}$$

For the vector u, we calculate the speed $u_i = \begin{bmatrix} u_{xi} & u_{y_i} \end{bmatrix}$ by taking the derivative of the cubic spline.

$$u_{xi} = \begin{cases} b_{x0} & \text{for } i = 0 \\ b_{xi} + 2c_{xi}(t_i - t_{i-1}) + 3d_{xi}(t_i - t_{i-1})^2 & \text{for } i = 1, 2, \ldots, n \end{cases}$$

$$u_{y_i} = \begin{cases} b_{y0} & \text{for } i = 0 \\ b_{y_i} + 2c_{y_i}(t_i - t_{i-1}) + 3d_{y_i}(t_i - t_{i-1})^2 & \text{for } i = 1, 2, \ldots, n \end{cases}$$

For the vector v, we calculate the speed $v_i = \begin{bmatrix} v_{xi} & v_{y_i} \end{bmatrix}$ with three-point endpoint or three-point midpoint method.

$$v_{xi} = \begin{cases} \frac{1}{2h}(-3x_0 + 4x_1 - x_2) & \text{for } i = 0 \\ \frac{1}{2h}(x_{i+1} - x_{i-1}) & \text{for } i = 1, 2, \ldots, n-1 \\ -\frac{1}{2h}(-3x_n + 4x_{n-1} - x_{n-2}) & \text{for } i = n \end{cases}$$

$$v_{y_i} = \begin{cases} \frac{1}{2h}(-3y_0 + 4y_1 - y_2) & \text{for } i = 0 \\ \frac{1}{2h}(y_{i+1} - y_{i-1}) & \text{for } i = 1, 2, \ldots, n-1 \\ -\frac{1}{2h}(-3y_n + 4y_{n-1} - y_{n-2}) & \text{for } i = n \end{cases}$$

3.2 Plot

Then we plotted the velocity fields by:

```
quiver(X, Y, u(:, 1), u(:, 2));
hold on;
quiver(X, Y, v(:, 1), v(:, 2), 'r');
legend('Cubic Spline Method', 'Three-point Method');
```

The resulting graph is shown in Figure 2. As we can see, the difference is not significant.

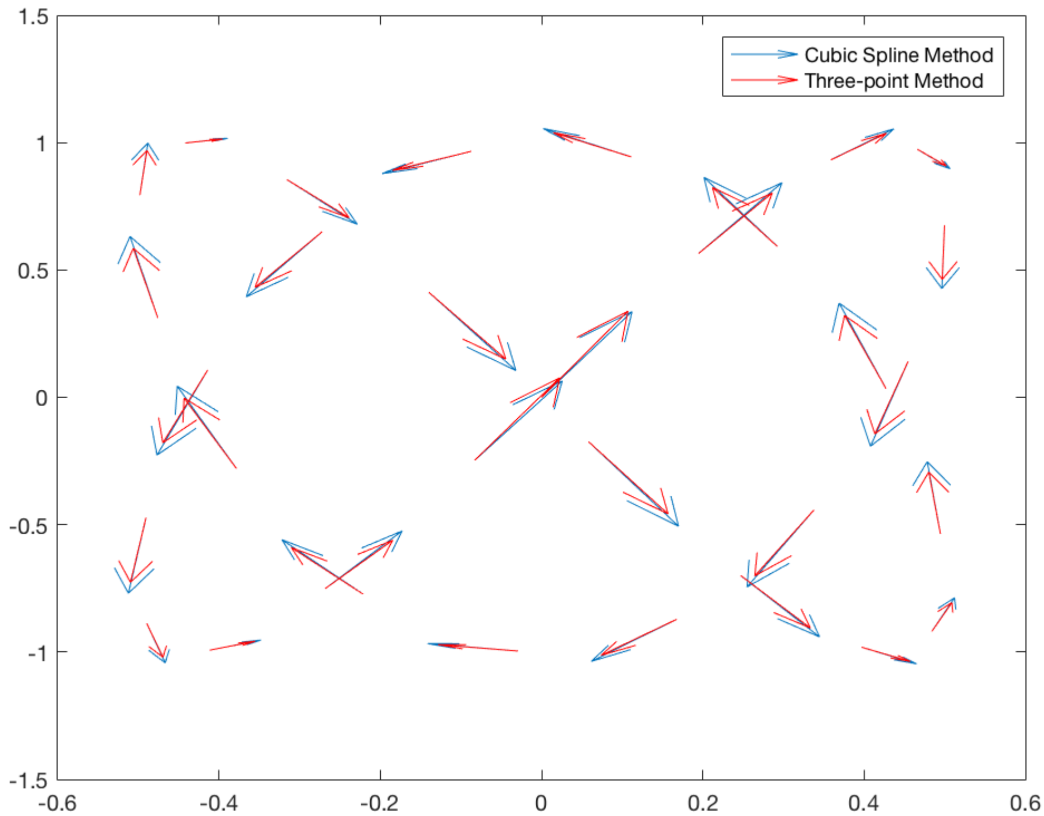

Figure 2: Plot of speed vector u (blue) with cubic spline method and v (red) with three-point method of the tracked vehicle at time $t_0, t_1, \ldots, t_n$