Assignment #3 CSC376 Fall 2020

Robert Evans

November 27th, 2020

We declare that this assignment is solely our own work, and is in accordance with the University of Toronto Code of Behaviour on Academic Matters.

This submission has been prepared using LaTeX.

QUESTION 1.1:
a) $\dot{s}_{max} = \frac{3}{2T}$ and $\ddot{s}_{max} = \frac{6}{T^2}$ for third-order polynomials (from figure 9.2)

$$\theta_1(s) = s2\pi, \dot{\theta}_1(s) = \dot{s}2\pi, \ddot{\theta}_1(s) = \ddot{s}2\pi$$

$$max|\dot{\theta}_1(s)| = max|\dot{s}|2\pi = \frac{3\pi}{T_{min,v}}$$

$$max|\ddot{\theta}_1(s)| = max|\ddot{s}|2\pi = \frac{12\pi}{T^2_{min,a}}$$

Using the values for max velocity ($\frac{\pi}{4} rad/s$) and max acceleration ($\frac{\pi}{2} rad/s^2$), we solve for $T_{1min,v}$ and $T_{1min,a}$. Below are the final answers after solving for the respective T values.

$$T_{1min,v} = 12s, T_{1min,a} = 4.90s$$

Now for the θ_2 values.

$$\theta_2(s) = s\frac{5\pi}{4}, \dot{\theta}_2(s) = \dot{s}\frac{5\pi}{4}, \ddot{\theta}_2(s) = \ddot{s}\frac{5\pi}{4}$$

$$max|\dot{\theta}_2(s)| = max|\dot{s}|\frac{5\pi}{4} = \frac{15\pi}{8T_{min,v}}$$

$$max|\ddot{\theta}_2(s)| = max|\ddot{s}|\frac{5\pi}{4} = \frac{15\pi}{2T^2_{min,a}}$$

Using the values for max velocity ($\frac{\pi}{4} rad/s$) and max acceleration ($\frac{\pi}{2} rad/s^2$), we solve for $T_{2min,v}$ and $T_{2min,a}$. Below are the final answers after solving for the respective T values.

$$T_{2min,v} = 7.5s, T_{2min,a} = 3.87s$$

The lowest value is $7.5s$ thus the fastest motion time for the third-order polynomial time-scaling is $7.5s$.

b) $\dot{s}_{max} = \frac{15}{8T}$ and $\ddot{s}_{max} = \frac{10}{T^2\sqrt{3}}$ for fifth-order polynomials (from figure 9.4)

$$\theta_1(s) = s2\pi, \dot{\theta}_1(s) = \dot{s}2\pi, \ddot{\theta}_1(s) = \ddot{s}2\pi$$

$$max|\dot{\theta}_1(s)| = max|\dot{s}|2\pi = \frac{15\pi}{4T_{min,v}}$$

$$max|\ddot{\theta}_1(s)| = max|\ddot{s}|2\pi = \frac{20\pi}{T^2_{min,a}\sqrt{3}}$$

Using the values for max velocity ($\frac{\pi}{4} rad/s$) and max acceleration ($\frac{\pi}{2} rad/s^2$), we solve for $T_{1min,v}$ and $T_{1min,a}$. Below are the final answers after solving for the respective T values.

$$T_{1min,v} = 15s, T_{1min,a} = 4.80s$$

Now for the θ_2 values.

$$\theta_2(s) = s\frac{5\pi}{4}, \dot{\theta}_2(s) = \dot{s}\frac{5\pi}{4}, \ddot{\theta}_2(s) = \ddot{s}\frac{5\pi}{4}$$

$$max|\dot{\theta}_2(s)| = max|\dot{s}|\frac{5\pi}{4} = \frac{75\pi}{32T_{min,v}}$$

$$max|\ddot{\theta}_2(s)| = max|\ddot{s}|\frac{5\pi}{4} = \frac{25\pi}{2T^2_{min,a}\sqrt{3}}$$

Using the values for max velocity ($\frac{\pi}{4} rad/s$) and max acceleration ($\frac{\pi}{2} rad/s^2$), we solve for $T_{2min,v}$ and $T_{2min,a}$. Below are the final answers after solving for the respective T values.

$$T_{2min,v} = 9.375s, T_{2min,a} = 3.80s$$

The lowest value is $9.375s$ thus the fastest motion time for the fifth-order polynomial time-scaling is $9.375s$.

c) The trapezoidal time scaling is quite different from the other two time-scalings as the fastest motion time T is achieved using a bang-bang motion as there is no coast time, which would increase the motion time. However, I could not find the proper $\dot{s}_{max}$ and $\ddot{s}_{max}$ values in the textbook or in the lectures and used a physics equation to determine the time overall.

$$\frac{T}{2} = \sqrt{\frac{d}{a}}$$

In this case, d is the distance traveled (over a πrad distance for θ_1), and a is the acceleration at the peak, which is $\frac{\pi}{2}\frac{rad}{s^2}$ for θ_1 and is given in the question. After isolating for the variable T, and inserting the variables, the final equation is the following for θ_1:

$$T_1 = \sqrt{\frac{2\pi}{\pi}} \times 2 = 2.84s$$

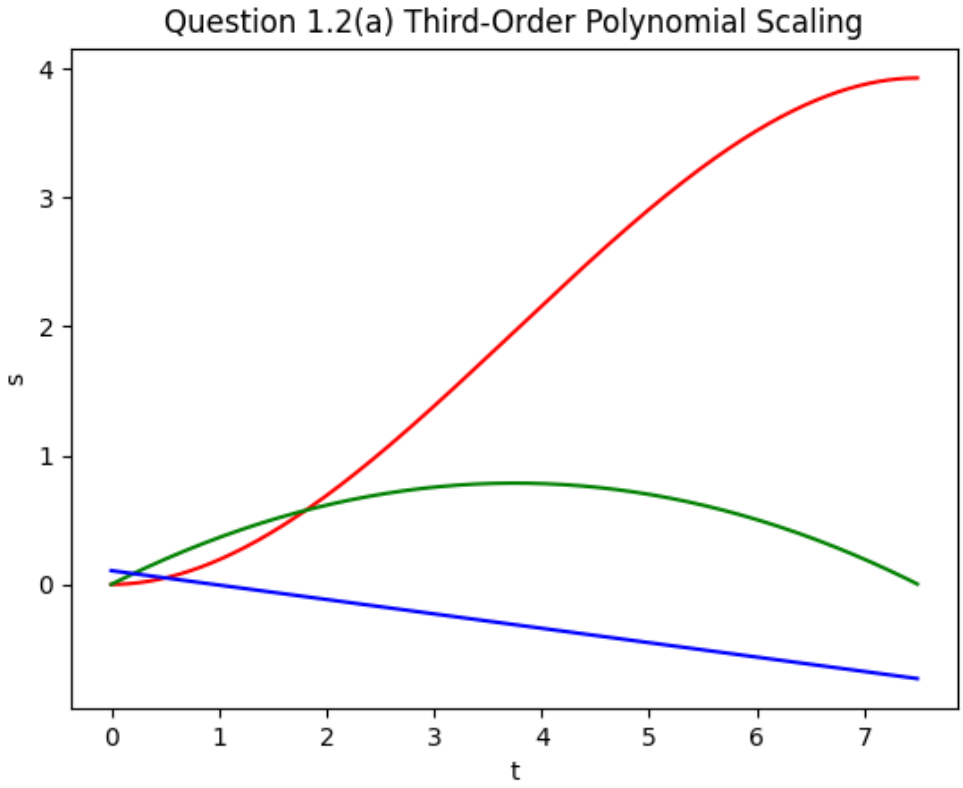
Since the acceleration restriction for both joins is the same, it is reused for θ_2. However, the distance value is $\frac{5\pi}{4} rad$ for θ_2. The following is the result for T_2:

$$T_2 = \sqrt{\frac{5\pi}{2\pi}} \times 2 = 3.16s$$

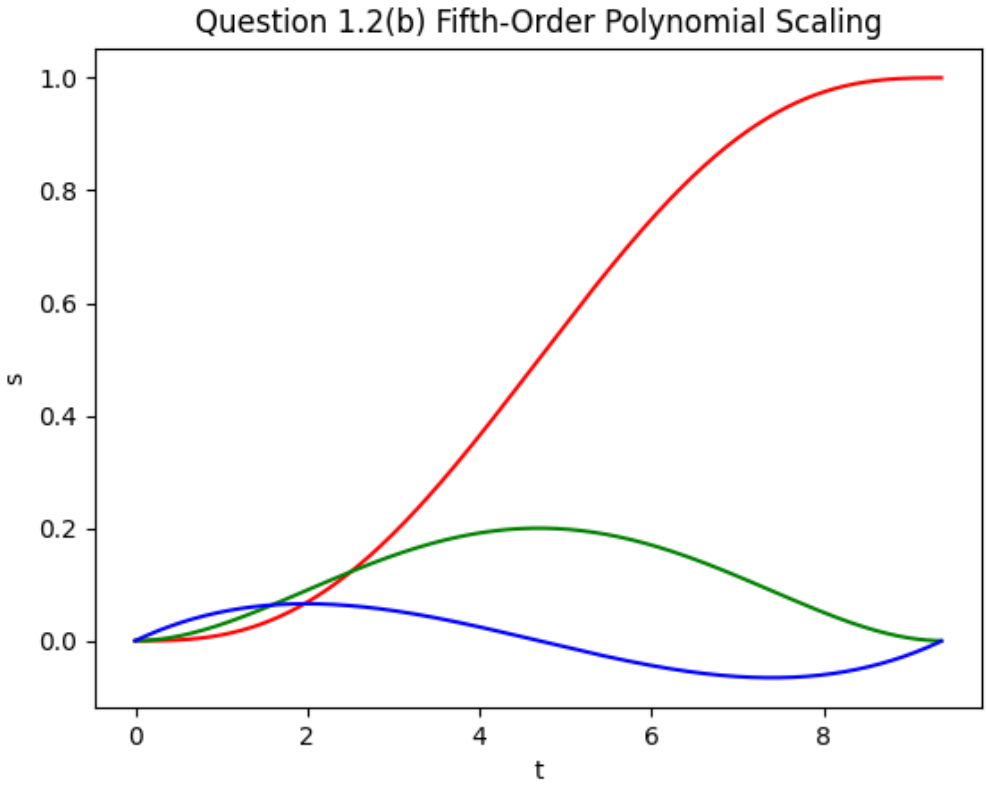
Since T_2 is the max between the 2 values, the fastest motion time for the trapezoidal time-scaling is $3.16s$.

QUESTION 1.2:

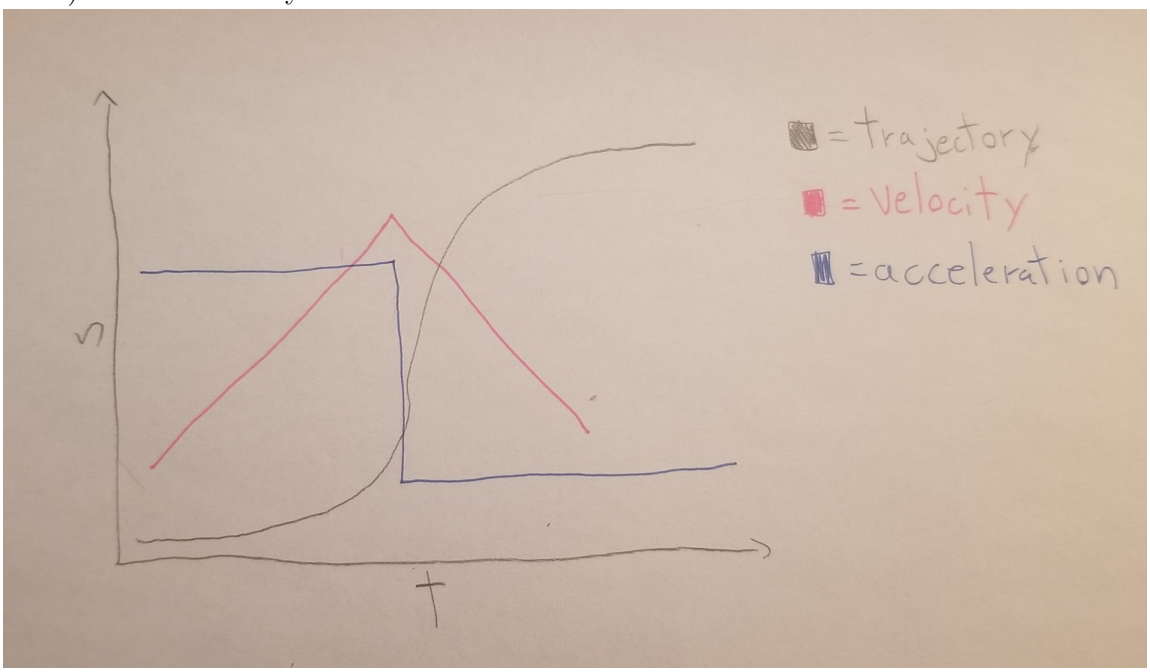
a) Below is the figure representing the third-order polynomial scaling plots. The red line is the trajectory, the green line is the velocity and the blue line is the acceleration.



b) Below is the figure representing the third-order polynomial scaling plots. The red line is the trajectory, the green line is the velocity and the blue line is the acceleration.



c) Below is a hand-drawn figure representing the trapezoidal (bang-bang) scaling plots. The black line is the trajectory, the pink line is the velocity and the blue line is the acceleration. The velocity should be longer as it shouldn't hit zero before the distance stops increasing. This was hand-drawn as the derivation used in Question 1.1 *c*) wasn't officially used from this course.



Question 1.3:

Third-Order Polynomial Time Scaling: The main advantage of third-order polynomial time scaling is that the maximum joint velocities are reached at $\frac{T}{2}$ and that there is only 4 constraints making it easy to calculate whether the motion is feasible as it uses the least amount of design freedoms. The main disadvantage is that the robot achieves a discontinuous jump in acceleration at both $t = 0$ and $t = T$ which can result in vibrations in the robot.

Fifth-Order Polynomial Time Scaling: The main advantage of fifth-order polynomial time scaling is that with the added constraints on the acceleration values allows for smoother motion than in third-order polynomial time scaling. The main disadvantage is that adding these constraints requires the addition of two more design freedoms in the polynomial.

Trapezoidal Time Scaling: The main advantage of trapezoidal motion is that it is the fastest straight-line motion possible. However, the main disadvantage is that the motion is not very smooth and the trajectory increased as a much sharper rate with more jerk.

Question 2.2:
The basic summary of my approach was very similar to the RRT algorithm shown in the textbook in Chapter 10.5.1, where I set several variable myself after testing the code. The *max_nodes*, *step_length*, *n_goal*, *move_rate* and *move_rate_epsilon* variables were pre-determined through testing. The general approach is as follows:

1. Sample a random position using the $MotionPlanner :: drawRandomConfig()$ function, unless the iteration number is at a multiple of *n_goal* (from the second point of Practical 8) where the *q_goal* is used as the sample instead.

2. The sample node is then put into the $getNearestNode()$ function where the Euclidean Distance from each Node in the tree T is calculated. The node with the smallest Euclidean Distance is then the closest node.

3. A new node is generated in the direction from the nearest node to the sample node limited by a rate variable called *move_rate* (determined beforehand).

4. A path is linearly interpolated between the sample and nearest nodes. This line is split up into a certain amount of partitions, determined by *move_rate_epsilon*. At each partition, the line is checked if it is colliding with any other objects until the end of the line is reached.

5. This edge is then added to T. The final check is if the new node is at the same location as the goal, and if it is, the list of parents (reversed path) is added to a vector then reversed and is the final return value.

The main issue encountered was that the motion planner stops after a couple iterations so there wasn't many optimizations or changes that were able to be made. The overall result was that the robot moved through a couple attempts and iterations and they eventually stopped. The steps followed the algorithm and should work logically however, there seems to be an error in the edges being added to T properly.

The main improvement for the RRT algorithm would be to add tweak the algorithm so that not every node of T would have to be visited individually. My main idea for that would be to have weighted trees that would have their weight updated over time with each sample to increase the efficiency to find the nearest node as not every node would need to be visited. Another improvement is to split the workspace into quadrants and ensure that the next sample would be in a certain quadrant, to reduce the randomness of each sampled node.