Motion Profiling for FRC Robots

Finn O'Toole Boire
Programmer
FRC 1678, Davis, CA

June 4, 2017

1 Model

For this paper, when we are using example and showing models of a system, it's going to be that of a drivetrain. This is an overview of the tank drive drivetrain model.

Consider both sides of the drivetrain as single wheels, which both have their own velocities. Then, the forwards velocity of the drivetrain is equal to the average of the two velocities.

$$v_{forwards} = \frac{v_{left} + v_{right}}{2}$$

However, the drivetrain also has the potential to rotate, by driving the two sides at different velocities. Since when you drive one side backwards and the other side forwards, it goes at its maximum angular velocity, the velocities must be subtracted, not added. Finally, since this calculated velocity is still linear velocity, it can be divided by the robot radius to acquire the robot's angular velocity.

$$\omega = \frac{v_{left} - v_{right}}{r_{robot}}$$

2 Motion Profiling

Motion profiling is the practice of setting non-static goals for a system to reduce sudden accelerations and impossible movements. It's important to reduce the wear and tear on mechanisms on robots and controlled systems, and allows more effective use of certain aspects of controllers, like integral terms.

2.1 Systems without Motion Profiling

When you have a system which doesn't profile its goals, it ends up getting given unrealistic goals and causes motors and inputs to suffer. One example, from Citrus Circuits' 2016 robot Adrian, was the large pivot arm. With so much

mass, the arm put serious strain on the physical components of the robot, like the central shaft on the pivot, and on the motors, which contributed to them breaking multiple times. You can see this in Fig. 1 an image of the time the central shaft of the pivot sheared completely through, due to slight imperfections in manufacturing along with repeated heavy stress on the part.

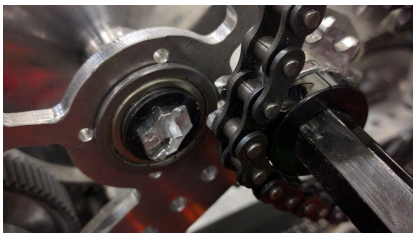


Figure 1: A sheared hex shaft from Citrus Circuit's 2016 pivot

Programmatically, systems without motion profiling are simple. You set a goal, the controller responds by trying to make the system converge towards the goal, giving it a voltage or input which was tuned by a person to make it converge most effectively. Below, in Figure 2, is a graph of modeled system responding to an unprofiled system:

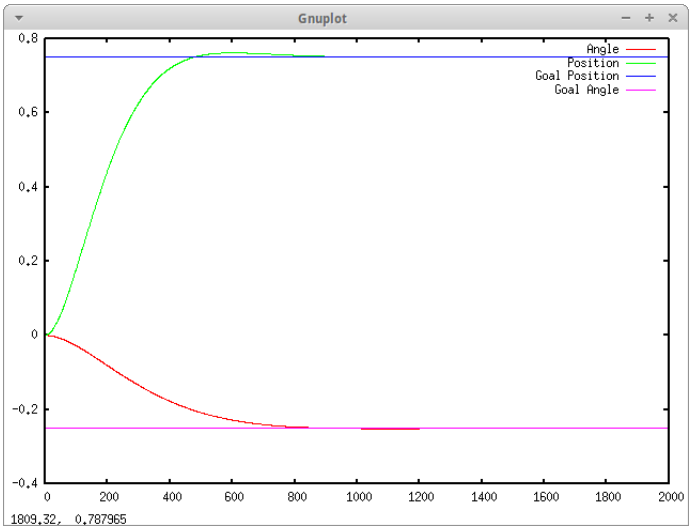
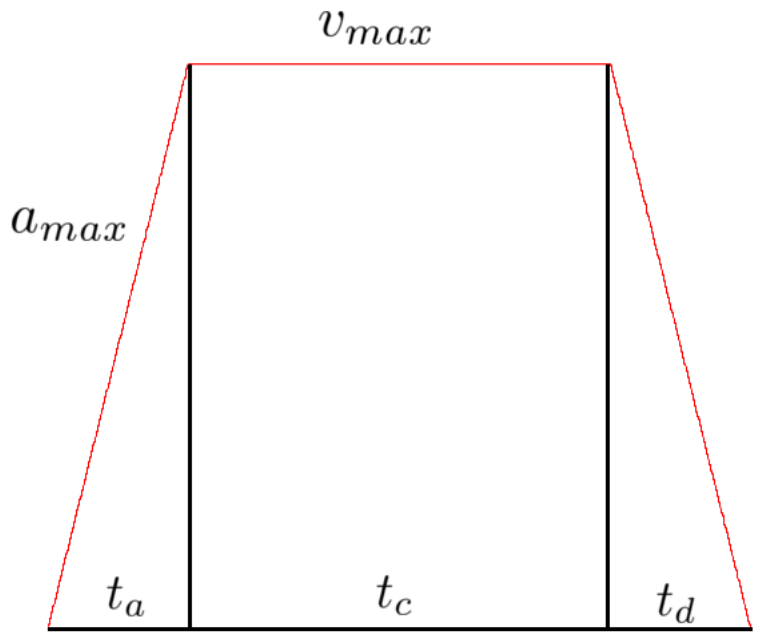


Figure 2: The drivetrain responding to an unprofiled goal

It responds well, but not ideally. In addition, instead of ramping up at the maximum acceleration of the drivetrain, it gives full voltage to the motors, which draws a lot of current and creates brownouts on an actual system.

2.2 Trapezoidal Motion Profiling

To solve these problems, you can profile the goal, accounting for the maximum accelerations and velocities of the drivetrain or other system you are modelling. Profiling the goal also allows you to have a lot of feed-forwards control, not relying as heavily on sensors and other inputs to the system. Deriving the trapezoidal motion profile goes like this. You have two constraints: a_{max} and v_{max}. When you are given a goal distance d_{goal}, the distance under the velocity profile must equal this distance, or $\int_0^{t_{total}} (v)dt = d_{goal}$.[1] Instead of contemplating this integral however, we can break down the trapezoid into multiple parts; the acceleration, cruising, and deceleration.



We need to find first the times it takes to accelerate and decelerate, and identify the distance that we travel in that time. Since you know the slope of the line a_{max}, and you know the starting point, some v_s, and the maximum velocity v_{max}, you can figure out the time travelled by considering the time $t_a * a_{max}$ to be equal to the difference between v_s and v_{max}. You can do the same for the deceleration time, but with an ending velocity v_e instead.

[1] Don't get worried, because although calculus can indeed be used as a way to determine the constraints of the profile more intuitively and easily, you can also use simple geometry and kinematic equations.

$$t_a = \frac{v_{max} - v_s}{a_{max}}$$

$$t_d = \frac{v_{max} - v_e}{a_{max}}$$

Notice we are not assuming v_s or v_e to be zero - this gives us greater flexibility in generating profiles. For example, if we were already moving forwards while going into the profile, you don't want to miscalculate the time to accelerate because you are already moving at a velocity. Similarly, if you have a target ending velocity, subtracting v_e accommodates for this. With t_a and t_d calculated, you can now calculate the distance travelled while accelerating and decelerating, which only leaves you with t_c to calculate to reach the desired distance.

The distance travelled during acceleration is equal to the average velocity of that section times the time that it travels.

$$d_a = t_a \frac{v_s + v_{max}}{2}$$

This can be simplified by substituting the times for their respective equations.

$$d_a = (\frac{v_{max} - v_s}{a_{max}})(\frac{v_s + v_{max}}{2})$$

$$d_a = \frac{v_{max}^2 - v_s^2}{2a_{max}}$$

The sum of each part's distance should be equal to the total distance.

$$d_{goal} = d_a + d_c + d_d$$

Having calculated d_a and d_d, you can substitute this into the equation. Using basic kinematic equations you can also determine $d_c = t_c v_{max}$, and substitute that as well.

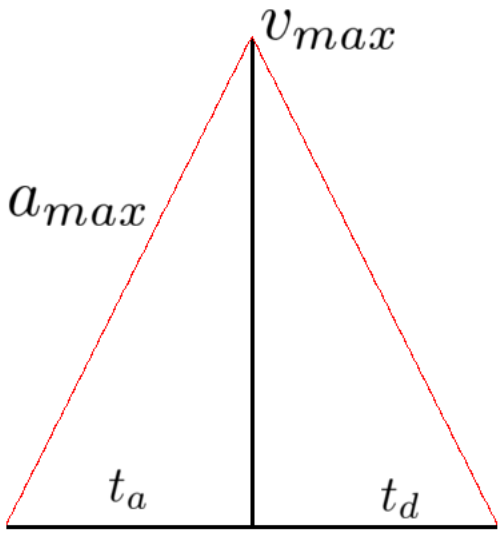
$$d_{goal} = \frac{v_{max}^2 - v_s^2}{2a_{max}} + \frac{v_{max}^2 - v_e^2}{2a_{max}} + t_c v_{max}$$

Solving for t_c, our unknown value which is the only thing preventing us from knowing the trapezoidal profile's constraints.

$$t_c = \frac{d_{goal} - \frac{v_{max}^2 - v_s^2}{2a_{max}} + \frac{v_{max}^2 - v_e^2}{2a_{max}}}{v_{max}}$$

This equation is what finally allows us to determine the profile's length and shape. You can also see that the only three values needed to calculate this cruising time is the maximum velocity, maximum acceleration, and the desired distance. With this calculation, you can then write code which calculates the time for each portion of the profile, then sets a goal based on where it is in the profile and how much time has elapsed.

However, there is one problem with this. If the target distance d_{goal} is less than the distance it takes to accelerate and decelerate from your starting velocity v_s and ending velocity v_e, then instead of making a trapezoidal motion profile, you will have to make a triangular motion profile. This math is slightly different, because you now have a lower maximum velocity than the physical v_{max} constraint on the drivetrain or system.



Without a cruising section, you must now calculate the new maximum velocity to be able to calculate the time for acceleration and deceleration. Assuming again to have non-zero starting and ending velocities, the acceleration times are $t_a = \frac{v_{max}-v_s}{a_{max}}$ and $t_d = \frac{v_{max}-v_e}{a_{max}}$. In addition, the equation resulting in d_{goal} still holds.

$$d_{goal} = \frac{v_{max}+v_s}{2}t_a + \frac{v_{max}+v_e}{2}t_d$$

However, since we no longer know v_{max}, we'll need to substitute the times and solve for it.

$$d_{goal} = (\frac{v_{max}+v_s}{2})(\frac{v_{max}-v_s}{a_{max}}) + (\frac{v_{max}+v_e}{2})(\frac{v_{max}-v_e}{a_{max}})$$

$$d_{goal} = \frac{(v_{max}+v_s)(v_{max}-v_s)}{2a_{max}} + \frac{(v_{max}+v_e)(v_{max}-v_e)}{2a_{max}}$$

$$d_{goal} = \frac{v_{max}^2-v_s^2}{2a_{max}} + \frac{v_{max}^2+v_e^2}{2a_{max}}$$

$$2a_{max}d_{goal} = 2v_{max}^2 - v_s^2 - v_e^2$$
$$\frac{2a_{max}d_{goal} + v_s^2 + v_e^2}{2} = v_{max}^2$$
$$v_{max} = \sqrt{\frac{2a_{max}d_{goal} + v_s^2 + v_e^2}{2}}$$

This new maximum velocity can then be plugged into the two equations for time to acquire the constraints on time for the profile, and again you are able to calculate the velocity at any given point by following these constraints.