OPTIMIZATION OF MARIO KART TRACKS

ZACHARY EVANS, TEAYOUN KIM, DANIEL TREUHAFT

Abstract. Our goal is to find the optimal path that minimizes the time on a Mario Kart track with speed and acceleration limits, which comes from the actual Mario Kart game for the Nintendo Wii. We mainly used Pontryagis Maximum Principle to find the optimal solution for two different evolution equations. The first ignores drag while the second includes it. We find our results using solve_bvp are often unstable, but we were able to find a few hopeful results.

1. Background

Growing up between the years of 2006-2013, one of the biggest consoles on the market was the Nintendo Wii, and when playing with a group of people one of the go to party games was Mario Kart Wii. As a kid trying to beat older siblings always seemed like a difficult task. Our group wants to accomplish what our childhood selves could not, and find out, using Pontryagis Maximum Principle (PMP), the optimal path around a Mario Kart Track.

2. First Approach

2.1. **Mathematical Representation.** In the actual Mario Kart game, there are many factors that make finding the optimal path very complicated, such as item boxes, bumping into other players, boost panels on the ground, drifting boosts, and varying character and vehicle stats. In this analysis, we will ignore all of these factors, and instead assume that our vehicle has a maximum velocity of 22 m/s, a maximum acceleration of 5 m/s^2. Additionally we assume that there are no items or objects that can speed us up, bump us, or slow us down [1,2]. We decided to start with a simple track defined by

$$1^4 \leq x^4 + y^4 \leq 2^4$$

for the inner and outer boundaries.

With this track 1, we decided that a reasonable starting position for our Mario Kart racer would be at $x = -1.5$, $y = 0$ (in kilometers), with no initial velocity.

Our goal is to find the fastest path and time to make one complete loop around the track. However, we had difficulties with this scenario because we start and finish on the same spot, so the lap would be considered completed

Date: January 13, 2026.

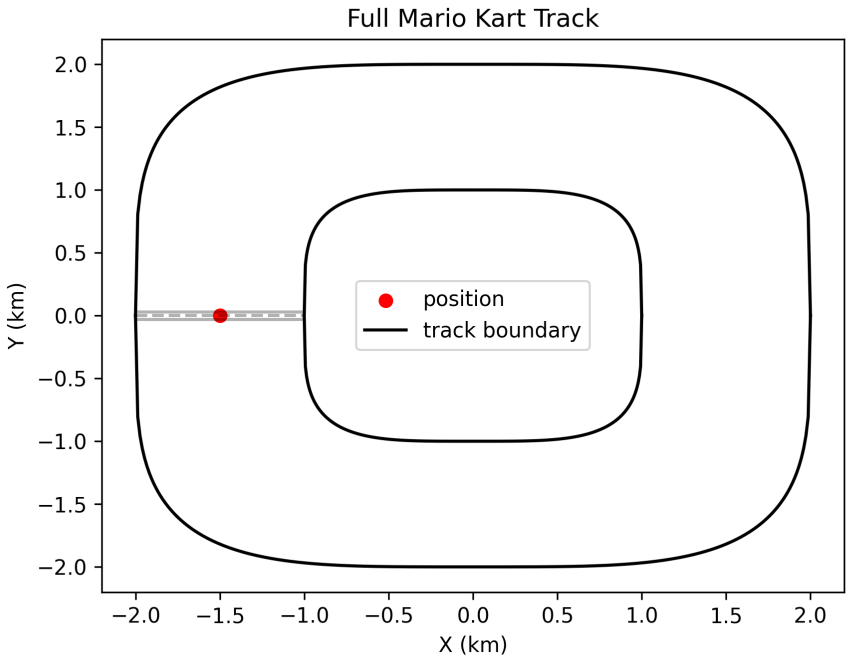

FIGURE 1. It describes the simplest full track.

as soon as it started. Thus, we instead decided to look at only the upper half of the track (see figure 2), given by:

$$\sqrt[4]{1^4 - x^4} \leq y \leq \sqrt[4]{2^4 - x^4}$$

With this new function, we also decided to start slightly above the line $y = 0$, so that we go up and around the track instead of down and around it backwards. Additionally, we changed our end goal to be $x = 1.5$.

Now that we have effectively set up our equation for our track we are going to be creating the following initial conditions.

Initial Conditions

Initial	Final
$x(0) = -1.5$	$x(t_f) = 1.5$
$y(0) = .01$	$y(t_f) = .01$
$x'(0) = 0$	$x'(t_f)$
$y'(0) = 0$	$y'(t_f)$

Note that both y' and x' start at zero to represent our initial velocity starting at 0. We also decided that our final velocities are free since we do not care how fast we are going at our final destination.

Note that we have our first initial conditions taken care of, we decided that rather than having hard boundaries and solving for those, we wanted to set

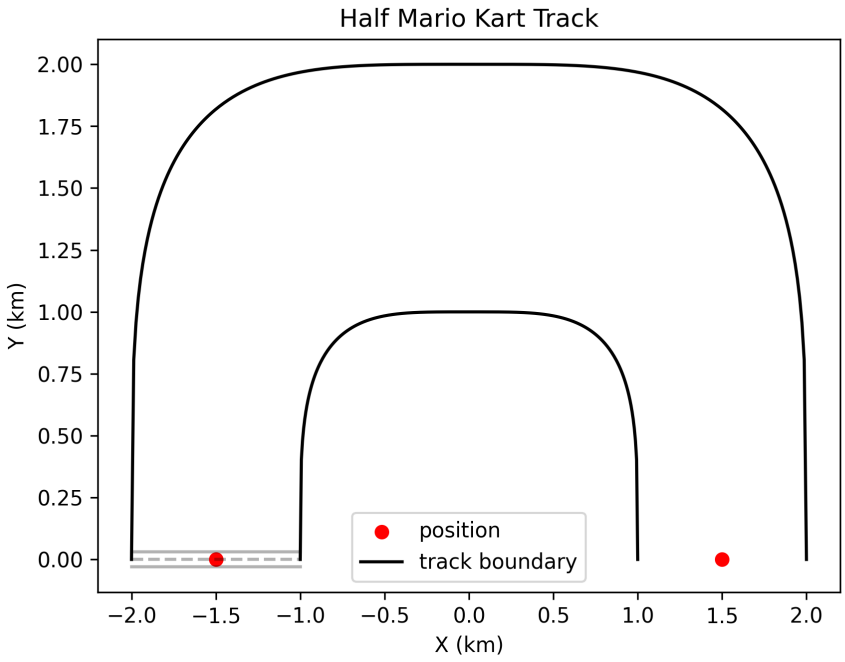

FIGURE 2. The simplest half track that we initially attempt to find optimal path.

up a cost functional with soft constraints. To set up a basic cost functional we knew we needed to minimize time. We also wanted to implement a cost for being on or outside the boundary as well as a cost for being above the maximum velocity. The last cost we wanted to implement was a cost on acceleration so that we could not turn to sharply.

To approach this we created the following cost functional:

$$J[u] = \int_0^{t_f} 1 + C_b(x, y) + C_v(x', y') + C_a(u_x, u_y)dx,$$

As you can see, we used soft constraints on our cost functional, not the hard inequality constraints. the This Jacobian would allow us to implement the cost we want. We then further define it by defining our C functions. They are defined as follows:

$$\begin{aligned} C_b(x, y) &= \alpha_1(x^4 + y^4 - 2^4) + \alpha_2(-x^4 - y^4 + 1^4) - \alpha_3 y \\ C_v(x', y') &= \beta_1(x' - 0.022) + \beta_2(y' - 0.022) - \beta_3 x' \\ C_a(u_x, u_y) &= w(u_x^2 + u_y^2), \end{aligned}$$

Note that α_1 increasing our cost if we cross the boundary on this inside while α_2 is increasing the cost if we cross the boundary on the outside and α_3 is increasing the cost for going underneath the track. Our β's similarly

prevent us from going to fast in the y direction or x direction. Additionally we realized our optimal path should always be moving forward along the x direction so we penalize when x' is negative. Finally our w is penalizing our acceleration as we desired.

With our function set up, we then decided to find a base time to try to beat. We approached with the initial guess for the path following a perfect semicircle with radius of 1.5km. We also knew our terminal velocity is $22m/s$ and the acceleration is $5m/s^2$ and used this to calculate our baseline time.

To calculate our baseline, we started by calculating the length of the course which by the definition of a semicircle is $\pi * r$ where our radius is 1500 meters (or 1.5 kilometers). This gives us our total distance or d_t of 1500π meters. Then since our acceleration is $5m/s^2$ and our max velocity is $22m/s$ we knew it would take $\frac{22}{5}$ seconds or 4.4 seconds to max our velocity which we will call t_1. We then used kinematic equations for distance to find how far we travel to go the max velocity which is defined by

$$d = v_0 t + \frac{1}{2}at^2$$

Note that our initial velocity is 0 and plugging in our acceleration and the time it takes to get to our terminal velocity we see that we travel 48.4 meters before getting to our max velocity. We will call this distance d_1

Now using the same kinematic equations we can now solve for our time to travel around the track in a perfect semicircle. We do this by setting up the following equation.

$$d_f - d_1 = v_0 * t + \frac{1}{2}at^2$$

Plugging in our variables we see that we have:

$$1500\pi m - 48.4m = 22\frac{m}{s}t_2$$

We then can simplify this to approximately

$$t_2 \approx 214.199$$

Thus adding $t_1 + t_2$ gives us our baseline time of 218.599 seconds which we will round down to 218 seconds since we are trying to get a faster time

2.2. **Solution.** We now will attempt to solve the first approach that we have made. We first set up our Hamiltonian as follows:

$$H = p_1 x' + p_2 y' + p_3 u_x + p_4 u_y - 1 - C_b(x, y) - C_v(x', y') - C_a(u_x, u_y)$$

By the Pontryagin's Maximum Principle, we calculate costate evolution equations,

$$p_1' = \frac{d}{dx}C_b(x, y),\ p_2' = \frac{d}{dy}C_b(x, y),\ p_3' = \frac{d}{dy}C_v(x, y) - p_1,\ p_4' = \frac{d}{dy}C_v(x, y) - p_2.$$

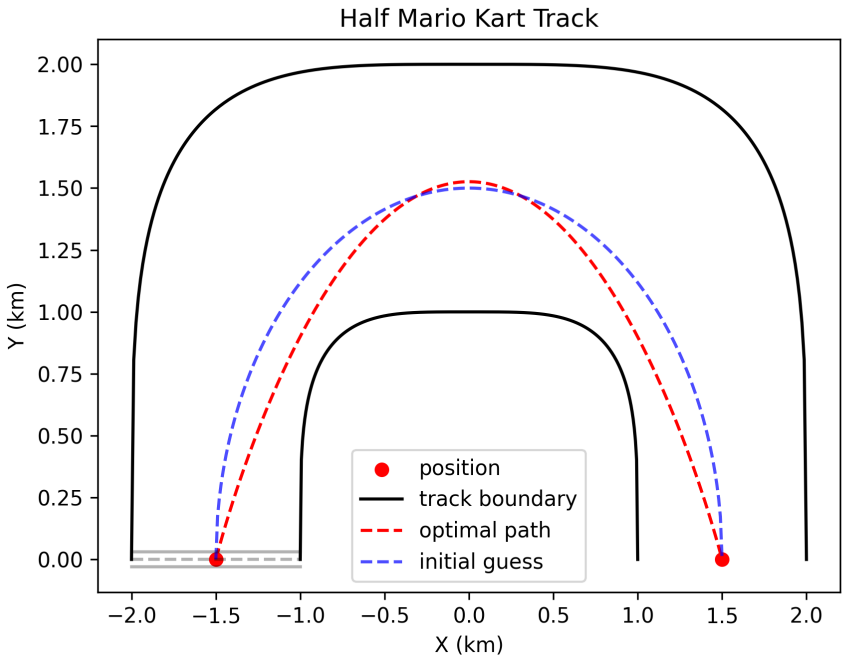

FIGURE 3. Optimal path that minimizes the time on the Mario Kart Track with limited speed and acceleration limits. (The weight constants are $\alpha_1 = 367, \alpha_2 = 348, \alpha_3 = 5,159,780,389, \beta_1 = 279,886, \beta_2 = 279,886, \beta_3 = 279,886$)

Also by the fact that $\frac{DH}{Du} = 0$, we see that

$$\frac{DH}{Du_x} = p_3 - 2wu_x = 0, \frac{DH}{Du_y} = p_4 - 2wu_y = 0 \ \rightarrow u_x = \frac{p_3}{2w}, u_y = \frac{p_4}{2w}$$

Now, we have state evolution equations and the costate evolution equations, which we can use a scipy method called, **solve_bvp()**, to solve the optimal path with a fixed end point.

$$s'(t) = \begin{bmatrix} x' \\ y' \\ u_x \\ u_y \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ \frac{p_3}{2w} \\ \frac{p_4}{2w} \end{bmatrix}, \ p'(t) = \begin{bmatrix} p_1(t)' \\ p_2(t)' \\ p_3(t)' \\ p_4(t)' \end{bmatrix} = \begin{bmatrix} 4(\alpha_1 - \alpha_2)x^3 \\ 4(\alpha_1 - \alpha_2)y^3 - \alpha_3 \\ \beta_1 - p_1 - \beta_3 \\ \beta_2 - p_2 \end{bmatrix}$$

,where $\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3$ are model choice constants. The Figure 3 describes the optimal solution path.

2.3. **Interpretation and analysis.** As a result of the first approach, we found a desirable optimal solution that seems reasonable. The following

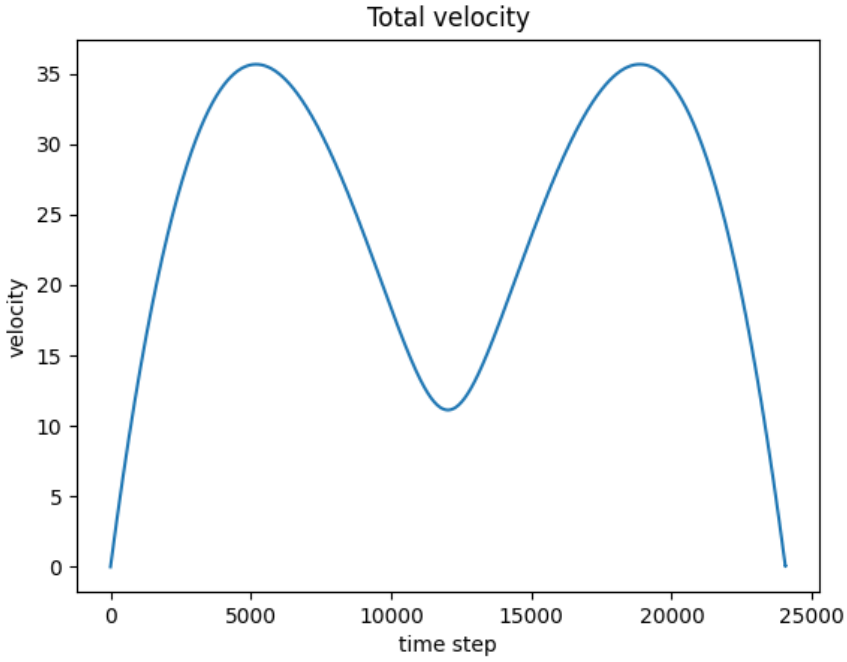

FIGURE 4. It describes the change of total velocity, that is, $v_{total} = \sqrt{v_x^2 + v_y^2}$ on the optimal path.

Figure 4 shows that the kart increases the total velocity, and it decreases the total velocity before it reaches to the middle turning point. After the turning point, it increases the total velocity again and decreases before it reaches to the end position. One thing to note is that the x-axis is time step not the exact time in the Figure 4.

As you noticed, the weight constants seem arbitrary and don't appear to be reasonable choices—almost like random values pulled out of nowhere. That is actually true. We noticed that PMP method or potentially *solve_bvp()* method is unstable in this problem, which gives us a totally different result with slightly different weight constants described in Figure 5. In fact when looking at our optimal path, we saw that the velocity went far above the max velocity and because of this we finished the track in .09 seconds, which although is much faster than our velocity, is unreasonable with the conditions we set. Because of the unstable result, we had a hard time trying to find the optimal solution for the simplest track.

3. Second Approach

In the previous setup, it was hard to get the right coefficients in our cost functional because higher velocities will naturally be incentivized in order to

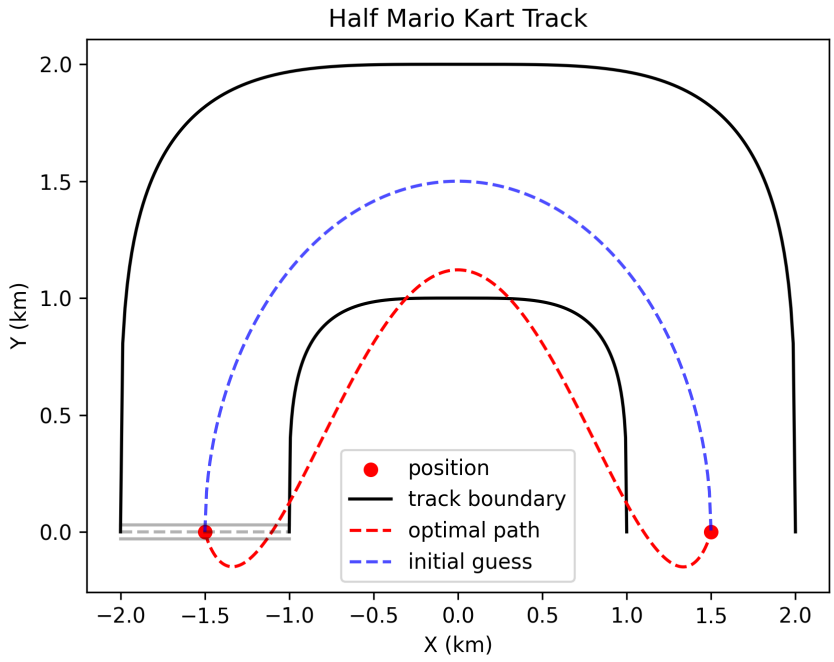

FIGURE 5. This figure has the same setting but different weight constant ($\beta_3 = 279,886 \rightarrow \beta_3 = 279,887$)

reduce time. Additionally, in tracks where the optimal path does cut across off-road sections, we want the final path to show the kart getting slowed down in these sections instead of just incurring a cost.

3.1. **Mathemaric Representation.** One of the main pieces we are missing to make the problem more true to the actual conditions in Mario Kart is drag. In the game, letting go of the acceleration button leads quickly to a noticeable decrease in speed, indicating that our analysis need to take this into account to get results that can be transferred to the game. Though resistance is usually taken out of problems to make them simpler, in our case there are a few simplifications that occur when we include drag. Specifically, we no longer need to think about a maximum velocity, because the maximum velocity will be enforced naturally by when the drag force is equal to our acceleration. Furthermore, if we have our drag coefficient be a function of the position, we can implement off-road sections without needing extra cost functions in our functional or hard constraints on our position.

This motivates the following equations of motion:

$$\begin{aligned} x' &= v_x \\ y' &= v_y \\ v_x' &= u\cos\theta - \gamma v_x\sqrt{v_x^2+v_y^2} \\ v_y' &= u\sin\theta - \gamma v_y\sqrt{v_x^2+v_y^2} \end{aligned} \tag{1}$$

These are the equations of motion for an object that experiences drag with a coefficient γ proportional to v^2, which is standard for fast-moving objects.

Assuming we want to enforce a v_{max}, we choose our γ such that if the kart is at v_{max} all in the x-direction, the acceleration v_x' is 0 when we accelerate in the x direction as much as possible. Thus,

$$\begin{aligned} v_y' = 0 &= u_{max} - \gamma v_{max}^2 \\ \gamma v_{max}^2 &= u_{max} \\ \gamma &= \frac{u_{max}}{v_{max}^2} \end{aligned}$$

We want to have different maximum velocities on and off of the track. On the track, we have the previously stated max velocity $v_max = 22\frac{m}{s}$ [2]. Off of the track, there is no standard maximum velocity, because there are many different off-road types with different amounts of penalty. This allowed us to keep the higher drag value flexible to see how it affected the optimal path. Redefining d to be the distance between the drag values and d_0 be the drag value for the track, we created the function

$$\gamma(x,y) = d(S(k(x^4+y^4-2^4) + S(-k(x^4+y^4-1^4)) + d_0 \tag{2}$$

where $S(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function and k is a constant we can to choose to determine how steeply it moves from the value on the track to the value off-road. This function is plotted in figure 6, where d_0 and d were chosen arbitrarily as 1 and k was chosen to be 3.

Finally, we had the pieces to put together for our optimization problem. Now, the only thing we are trying to minimize is time, so we want to minimize the functional

$$J[u] = \int_0^{t_f} 1dt$$

subject to the equations of motion in equation 1, with

$$\begin{aligned} u &\leq u_{max} \\ \theta &\in [-\pi, \pi] \end{aligned}$$

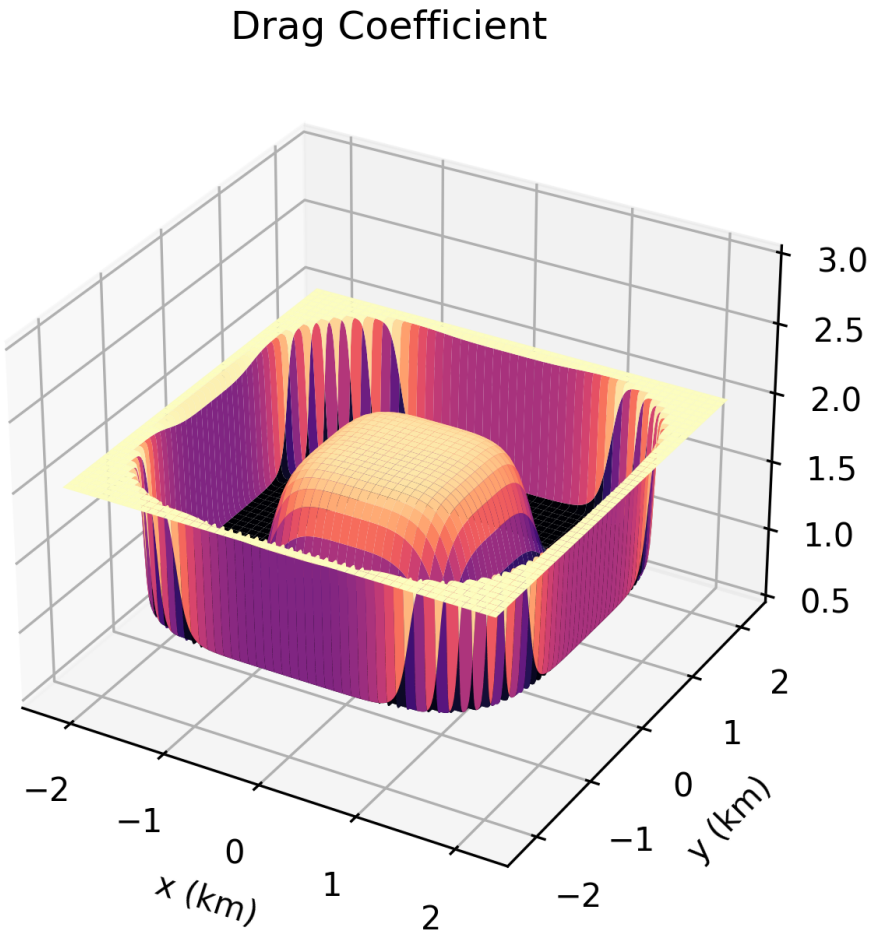

FIGURE 6. The drag coefficient around the track according to equation 2. On the track, we see the lower value of drag $d_0 = 1$, and off of the track, the drag is higher, at $d_0 + d = 2$. With the steepness constant $k = 3$, there is very little area on the track that is affected by the steep section between drag values.

3.2. **Solution.** To solve this problem, we utilize Pontryagin's Maximum Principle. First, we have state and costate vectors

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}, \mathbf{p} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \end{bmatrix}$$

The Hamiltonian for this problem is

$$\begin{aligned} H &= p \cdot f - L \\ &= p_1 v_x + p_2 v_y + p_3 \left(u \cos\theta - \gamma v_x \sqrt{v_x^2 + v_y^2} \right) + p_4 \left(u \sin\theta - \gamma v_y \sqrt{v_x^2 + v_y^2} \right). \end{aligned}$$

By Pontryagin's Maximum Principle, our control (u, θ) must maximize H. The only portions of H that depend on u or θ are the terms $p_3 u \cos\theta$ and $p_4 u \sin\theta$. Since θ is free, for any signs of p_3 and p_4, we can choose a value of θ such that both $p_3 \cos\theta$ and $p_4 \sin\theta$ are positive. Thus, the magnitude of our acceleration u should always be maximized, so $u = u_{max}$ everywhere.

Next, the value of θ should also maximize H, so $\frac{DH}{D\theta} = 0$. As long as p_3 and p_4 are nonzero, this gives us

$$\begin{aligned} 0 &= \frac{DH}{d\theta} \\ &= -p_3 u_{max} \sin\theta + p_4 u_{max} \cos\theta \\ \frac{p_4}{p_3} &= \tan\theta \end{aligned}$$

This motivates us consider a right triangle with the angle θ in one corner, with opposite edge having length p_4 and adjacent edge having length p_3. Then, the hypotenuse is $p = \sqrt{p_3^2 + p_4^2}$, which gives us $\cos\theta = \frac{p_3}{p}$ and $\sin\theta = \frac{p_4}{p}$. Note that these values make the terms $p_3 \cos\theta$ and $p_4 \sin\theta$ in H positive, so this extremizer is a maximum. Thus, the state evolution equation 1 can be written explicitly in terms of the state and costate.

To get the costate evolution equations, we used the other part of Pontryagin's Maximum Principle, specifically

$$\begin{aligned} \mathbf{p}' &= -\frac{DH}{D\mathbf{x}} \\ &= \begin{bmatrix} -\frac{DH}{Dx} \\ -\frac{DH}{Dy} \\ -\frac{DH}{Dv_x} \\ -\frac{DH}{Dv_y} \end{bmatrix} \\ &= \begin{bmatrix} p_3 \left(\frac{\partial}{\partial x} \gamma \right) v_x \sqrt{v_x^2 + v_y^2} + p_4 \left(\frac{\partial}{\partial x} \gamma \right) v_y \sqrt{v_x^2 + v_y^2} \\ p_3 \left(\frac{\partial}{\partial y} \gamma \right) v_x \sqrt{v_x^2 + v_y^2} + p_4 \left(\frac{\partial}{\partial y} \gamma \right) v_y \sqrt{v_x^2 + v_y^2} \\ -p_1 + p_3 \gamma \left(\sqrt{v_x^2 + v_y^2} + \frac{v_x^2}{\sqrt{v_x^2 + v_y^2}} \right) + p_4 \gamma \frac{v_x v_y}{\sqrt{v_x^2 + v_y^2}} \\ -p_2 + p_3 \gamma \frac{v_x v_y}{\sqrt{v_x^2 + v_y^2}} + p_4 \gamma \left(\sqrt{v_x^2 + v_y^2} + \frac{v_y^2}{\sqrt{v_x^2 + v_y^2}} \right) \end{bmatrix} \end{aligned}$$

Before being able to get analytical results, we notice that the evolution equations for our optimal velocity have a $p = \sqrt{p_3^2 + p_4^2}$ term in the denominator, and the evolution equations for our costate have a few $\sqrt{v_x^2 + v_y^2}$

terms in the denominator. With our current boundary conditions, we have $v_x = v_y = 0$ at the beginning of the track, and since the final v_x and v_y are free, both $p_3 = 0$ and $p_4 = 0$ at the end. To fix this, we slightly modified the starting state to have a small amount of velocity in the y-direction. Additionally, we chose to constrain the final velocities to allow the final p_3 and p_4 values to be free. For this constraint, we thought it would be reasonable to fix a x-velocity of 0, putting the kart in a good place for the next half of the lap. For the y-velocity, we thought it was most reasonable to be near the maximum velocity, so we constrained the final y-velocity to be $20\frac{m}{s}$. We chose this instead of the maximum velocity of $22\frac{m}{s}$ because the kart naturally gets slowed a little when it turns, to to achieve the maximum velocity it would have to go straight for a while before getting to the endpoint.

With these evolution equations and boundary conditions, we wrote a program to solve this optimization problem and find our optimal path.

3.3. **Interpretation.** The results of our solver on this problem are shown in figures 7 and 8. In this case, we used a value of γ on the off-road that halved our maximum velocity, forcing the best path to be along the track. This solution looks like a massive success, since the kart stayed within the boundaries and the total velocity remained bounded by v_{max} throughout the trip. The final time it found was $t_f = 202.06$ seconds, more than 15 seconds faster than our naive solution, or about 7%. This is an even better result when remembering that our baseline solution didn't account for air resistance, so it was able to accelerate to maximum velocity at u_{max} the entire time, whereas in this case as we go faster we accelerate less.

One thing that is interesting about this solution compared to our initial approach is that the path is much more rounded over the bend. This is likely a result of the acceleration being bounded, since only having a cost on acceleration allows brief periods of very high acceleration if they make the overall time much lower. We view this as a success for this model, since in the actual Mario Kart game, there is a strict upper bound on acceleration that can not be overcome, regardless of how much faster it would make our path.

We also tried reducing the drag coefficient to see if our equations could still find the optimal path if it becomes fastest to cut straight across the track. To do this, we let maximum velocity be $18\frac{m}{s}$ on the off-road, and used the same boundary conditions as before. The results of our model are summarized in figures 9 and 10. The final with this path and drag coefficient was 163.53 seconds, indicating that this path was more optimal than the one found with high drag.

4. Conclusion

Currently, we see the results we are getting as promising for future research, but the applicability of the paths we are getting is limited. This is because we still do not incorporate core game elements such as item boxes,

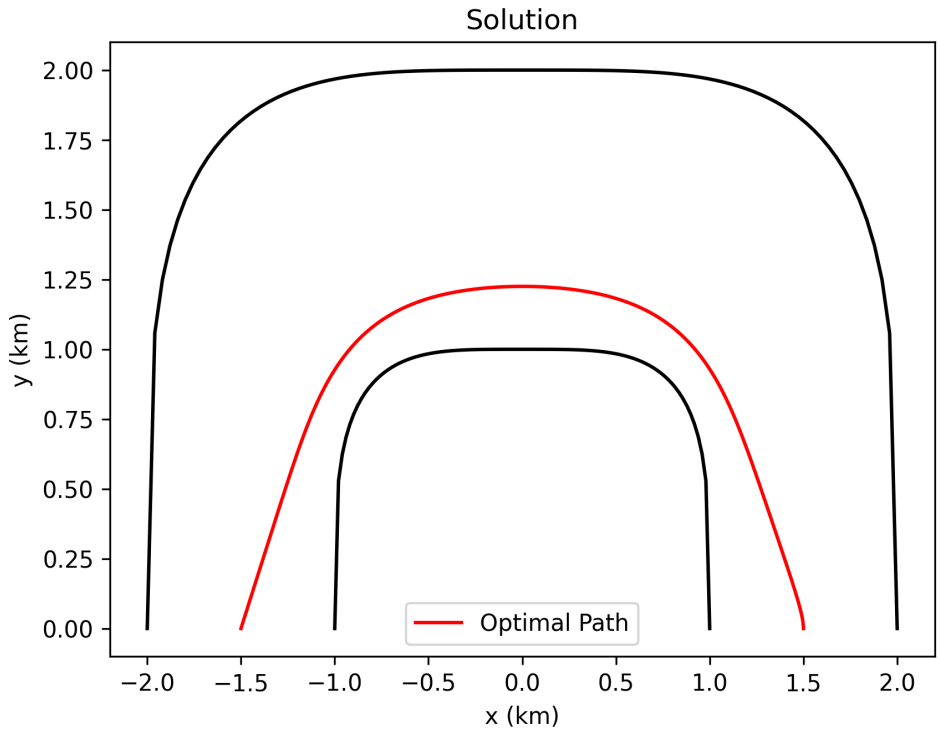

FIGURE 7. The optimal path using our second approach. This path takes 202.06 seconds, an improvement on our baseline of about 15 seconds.

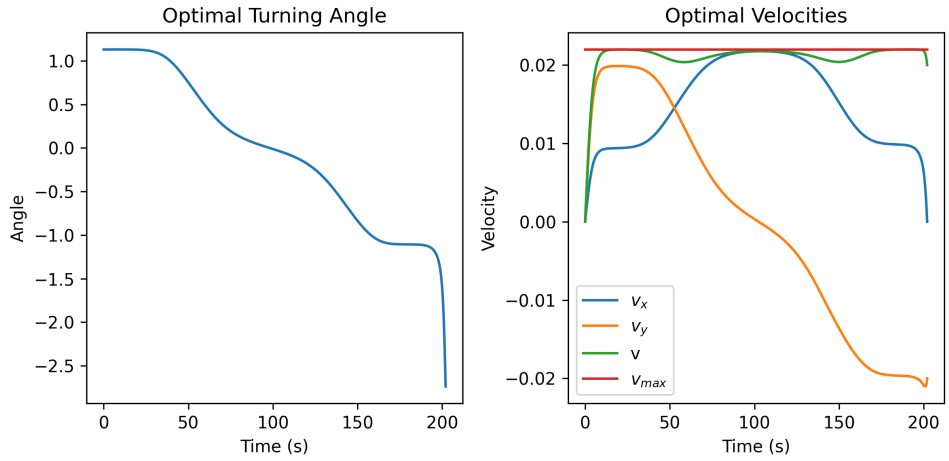

FIGURE 8. Plots our optimal turning angle and our optimal velocities over time assuming drag.

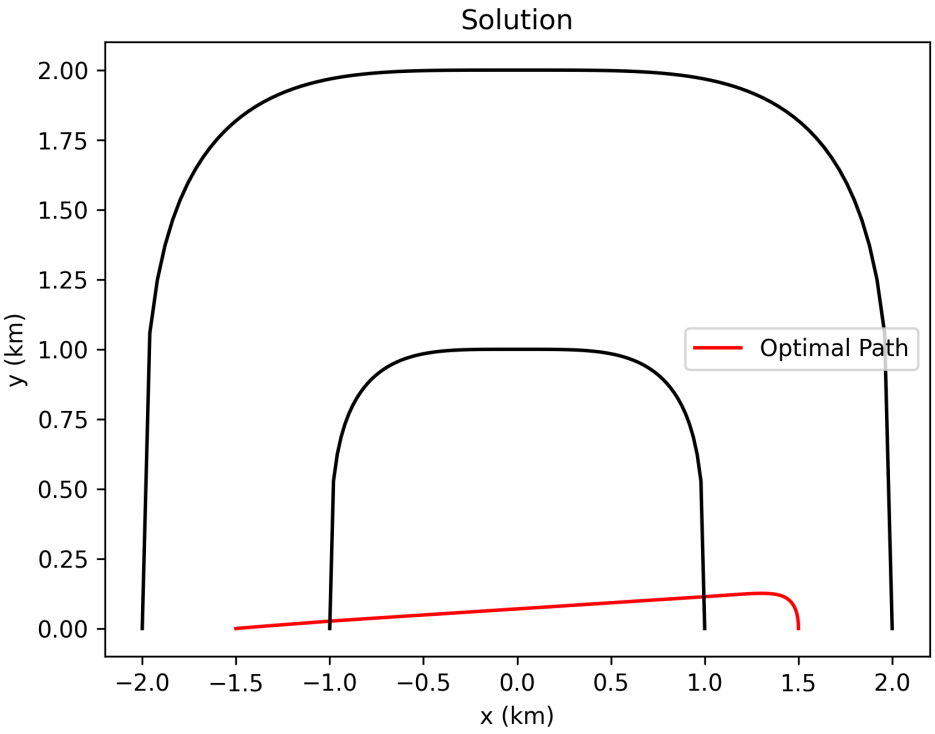

FIGURE 9. The optimal path when off-road drag is low. Because we didn't change the boundary conditions, it has some strange behavior at the end to force the final v_y to be $-22\frac{m}{s}$.

boost panels, and drifting mechanisms around turns. As a result, we would like to make our model better by adding more complexity to our track, such as a different course, areas that boost acceleration, and having both off-road and walled off areas that can't be accessed at all.

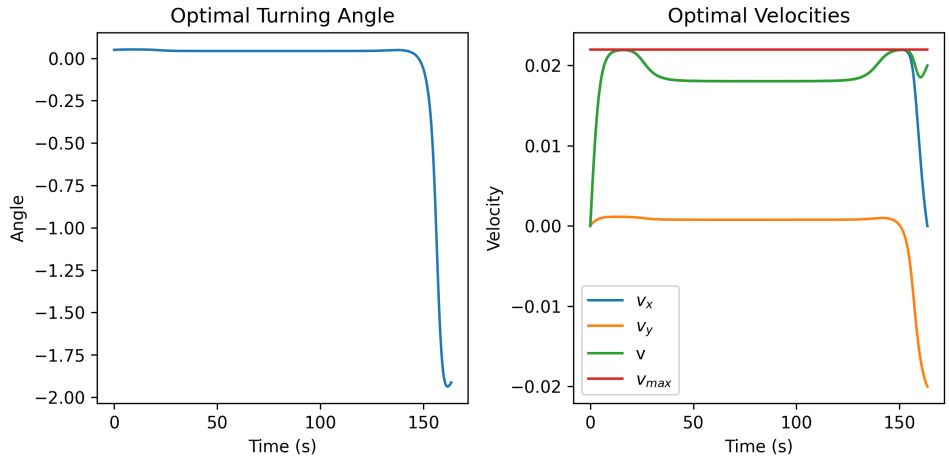

FIGURE 10. The optimal angle to steer at and velocities when off-road drag is low. Note that we can clearly see in the velocity graph where we are off-road, since our total velocity drops to about $18\frac{m}{s}$.

REFERENCES

(1) Mario Kart Wii Message Board for Wii - Gamefaqs, gamefaqs.gamespot.com/boards/942008-mario-kart-wii. Accessed 8 Apr. 2025.

(2) R/Theydidthemath on Reddit: [Request] How Fast Are the Karts in Mario Kart Actually Going?, www.reddit.com/r/theydidthemath/ comments/2meo0k/request_how_fast_are_the _karts_in_mario_kart/. Accessed 8 Apr. 2025.

We give Dr. Evans, Dr. Barker, and other instructors at BYU teaching ACME classes permission to share our project as an example of a good project in future classes they teach.