Optimal Racing Line Control

Shayegan Omidshafiei

May 9, 2014

Abstract

Analysis of race car dynamics and control is conducted. A kinematic race car system is constructed for preliminary construction of optimal racing lines. An extension to a 6-state dynamical system is then made using a half-car model. Performance is analyzed on a variety of race track types, using the MATLAB GPOPS library. A brief analysis of code optimization techniques is presented, motivated by the computational complexity of calculating the optimal trajectory for the dynamical model. Finally, performance on a complex track comprised of 18 phases is analyzed.

1 Introduction

Optimal control of car racing lines has been of interest for researchers in the last several years. The problem of designing optimal trajectories for race cars and high performance vehicles has been analyzed using a wide range of kinematic and dynamical models [5][2][7]. The optimal racing line problem can be considered a dual problem of the optimal vehicle control problem, and solving it in an efficient manner can be extremely useful in future autonomous vehicle systems.

This paper aims to investigate optimal control of vehicles on a race track, using the GPOPS MATLAB toolbox, starting with a relatively simple kinematic system model. Additional extensions are made to a dynamical model including tire friction forces, allowing more accurate trajectories to be obtained. Following this, an improved method of initial guessing is implemented to allow for problems up to 18 phases to be solved in GPOPS. The paper concludes with a brief discussion and analysis of improving performance in GPOPS, as the complexity of the problem requires significant effort to be spent on code optimization to allow solutions to be obtained in reasonable time.

2 Kinematic Model

Initial analysis is performed using a kinematic single-track model for the vehicle, following the derivation outlined in [5]. This formulation is similar to the well-known Dubins car model. This is used as a means of validating track design and initial optimal control results without considering the complications of a dynamical model.

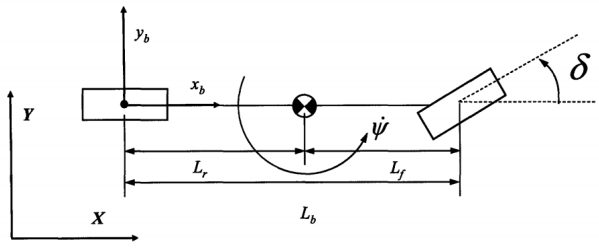


Figure 1: Single-track kinematic model.

In this model, the front and rear wheels of the car are paired together, essentially treating the car as a bicycle (Figure 1). Define the inertial frame of reference as x, y. Let V be the constant speed of the vehicle, ψ the vehicle yaw, and δ the steering angle. Given a vehicle body length of L_b, the kinematic equations are summarized as,

$$\dot{x} = Vcos(\psi) \tag{1}$$

$$\dot{y} = Vsin(\psi) \tag{2}$$

$$\dot{\psi} = \frac{Vtan(\delta)}{L_b} \tag{3}$$

3 Dynamical Model

A variety of dynamical models are considered for the vehicle. A 10-state full car model is presented in [10], including an engine torque map as well as

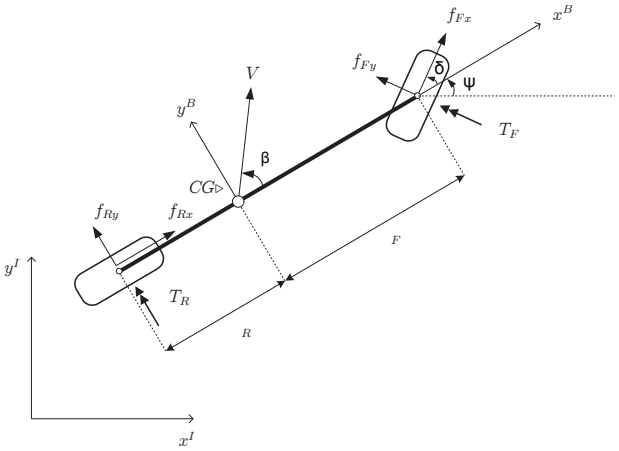


Figure 2: Single-track, half-car dynamical model [1].

aerodynamic drag and downforce effects. Given computational limitations, however, comprehensive models including downforce and air drag effects are not considered. A three-dimensional vehicle model is used in [8] for analyzing performance on banked turns. Single-track, constant speed models are considered in [5], but are deemed inappropriate for this application due to the severe assumptions made.

The dynamical model used in this paper is a single-track, half-car model, with tire friction forces taken into consideration [1]. Figure 2 illustrates the body diagram of the system. Suspension dynamics are not modeled, due to an addition of vertical vehicle dynamics which increase the state-space size, making the problem infeasible to solve in GPOPS. Aerodynamic forces as well as engine dynamics are neglected for the same reason.

Additional assumptions include lack of rear wheel steering and a center of gravity (CG) centered laterally on the vehicle (with varying longitudinal position). It is assumed that both the front and rear wheels can apply acceleration and deceleration independently, making this system a four-wheel drive car.

To derive the vehicle dynamical equations, let x, y represent the coordinates of the vehicle CG in an inertial frame of reference, m the mass of the vehicle, ψ the yaw angle, δ the steering angle, l_F and l_R the distance of the CG from the front and rear of the vehicle, respectively. Then, the dynamics are summarized,

$$m\ddot{x} = f_{F_x}cos(\psi+\delta) - f_{F_y}sin(\psi+\delta) + f_{R_x}cos(\psi) - f_{R_y}sin(\psi) \quad (4)$$

$$m\ddot{y} = f_{F_x}sin(\psi+\delta) + f_{F_y}cos(\psi+\delta) + f_{R_x}sin(\psi) + f_{R_y}cos(\psi) \quad (5)$$

$$I_z\ddot{\psi} = (f_{F_y}cos(\delta) + f_{F_x}sin(\delta))l_F - f_{R_y}l_R \quad (6)$$

Note that the problem, as posed, has a 6-dimensional state space (the first order differential equations for x, y, ψ can be trivially computed). The control inputs can be considered the friction forces in addition to the vehicle steering angle δ.

The longitudinal and lateral friction forces of the vehicle can be calculated using the so-called "Pacejka's Magic Formula" [11], which offers accurate modeling of tire dynamics with low computational overhead,

$$f_{ij} = -\frac{s_{ij}}{s_i}F_i(s),\ i = F, R,\ j = x, y \quad (7)$$

$$F_i(s_i) = f_{iz}Dsin(Catan(Bs_i)),\ i = F, R \quad (8)$$

The constants B, C, D appearing in the above formula are found empirically and can be used to model friction forces for various tires and road conditions. Pacejka's Magic Formula links longitudinal and lateral slip, $s_{ij}, i = F, R,\ j = x, y$ to the longitudinal and lateral friction forces. The normal force on the front and rear axles can be calculated using,

$$f_{R_z} = mg\frac{l_F}{l_F + l_R} \quad (9)$$

$$f_{F_z} = mg\frac{l_R}{l_F + l_R} \quad (10)$$

The longitudinal and lateral slip quantities can be individually calculated as follows,

$$s_{R_y} = \frac{vsin(\beta) - \dot{\psi}l_R}{vcos\beta} \quad (11)$$

$$s_{F_y} = \frac{vsin(\beta-\delta) + \dot{\psi}l_Fcos(\delta)}{vcos(\beta-\delta) + \dot{\psi}l_Fsin(\delta)} \quad (12)$$

$$s_{R_x} = \begin{cases} +1 - \frac{vcos(\beta) - \dot{\psi}l_R}{\omega_F r} \\ -1 + \frac{\omega_F r}{vcos(\beta) - \dot{\psi}l_R} \end{cases} \quad (13)$$

$$s_{F_x} = \begin{cases} +1 - \frac{v cos(\beta - \delta) + \dot{\psi} l_F cos(\delta)}{\omega_F r} \\ -1 + \frac{\omega_F r}{v cos(\beta - \delta) + \dot{\psi} l_F cos(\delta)} \end{cases} \quad (14)$$

where the total slip s_i in a given tire can be calculated as,

$$s_i = \sqrt{s_{ix}^2 + s_{iy}^2}, \; i = F, R \quad (15)$$

Finally, the vehicle speed and slip angle are calculated as,

$$v = \sqrt{\dot{x}^2 + \dot{y}^2} \quad (16)$$

$$\beta = tan^{-1}(\dot{y}/\dot{x}) - \psi \quad (17)$$

The above collection of equations can be used to reasonably model vehicle dynamics, with suspension, aerodynamic forces, and engine dynamics neglected. The control inputs are considered as the slip quantities s_{ij} and the steering angle δ.

3.1 Problem Initialization and Bounds

The cost function for the optimal control problem is simply the total travel time for the vehicle, with a free final time,

$$J = \int_0^{t_f} dt \quad (18)$$

The racetrack is constructed in piecewise fashion, using phases. Let $(xCoords_i, yCoords_i)$ define the center-line of the track, with $i \in [1, finalPhase{+}1]$. Let $arclengths_i$ define the arclength of the track up to phase i. In a given phase, the initial time is fixed at 0, with the maximum allowable final time defined heuristically as follows,

$$t_{f_{max},i} = \frac{arclengths_{i+1} - arclengths_i}{topSpeed/3} + t_{f_{max},const} \quad (19)$$

where topSpeed is the predefined top speed of the car, and $t_{f_{max},const}$ is a buffer time allotted to ensure completion of the phase is possible.

Vehicle parameters are defined in Table 1.

Note that road condition specifications (captured in Pacejka parameters B, C, D) are obtained from [1], and approximate a dry tarmac surface.

Bounds are placed on the car's x, y coordinates in a given phase i using the minimum and maximum values of the track's coordinates that phase,

$$x_{min,i} = min([xEdge_{1,i}, xEdge_{1,i+1}, xEdge_{2,i}, xEdge_{2,i+1}]); \quad (20)$$

Parameter	Value
v_{max}	varies per track, average 15 ms^{-1}
g	9.81 ms^{-2}
m	600 kg
I_{zz}	1000 kgm^2
l_R	0.8 m
l_F	0.7 m
B	7
C	1.6
D	0.7

Table 1: Vehicle parameters

$$x_{max,i} = max([xEdge_{1,i}, xEdge_{1,i+1}, xEdge_{2,i}, xEdge_{2,i+1}]); \quad (21)$$

$$y_{min,i} = min([yEdge_{1,i}, yEdge_{1,i+1}, yEdge_{2,i}, yEdge_{2,i+1}]); \quad (22)$$

$$y_{max,i} = max([yEdge_{1,i}, yEdge_{1,i+1}, yEdge_{2,i}, yEdge_{2,i+1}]); \quad (23)$$

Initial conditions for the car (in phase 1) are summarized as follows,

$$x_0 = xCoords_1 \quad (24)$$

$$y_0 = yCoords_1 \quad (25)$$

$$\dot{x}_0 = 5\frac{dx}{dL} \quad (26)$$

$$\dot{y}_0 = 5\frac{dy}{dL} \quad (27)$$

$$\psi_0 = atan2(dy, dx) \quad (28)$$

$$\dot{\psi}_0 = 0 \quad (29)$$

where $\frac{dy}{dx}$ is the slope of the track in the given phase, and dL is the arclength. The above equations indicate that the vehicle is initialized moving down the centerline of the track in the first phase, with a speed of $5ms^{-1}$.

Initial conditions for subsequent phases are bounded as follows,

$$x_0 \in [x_{min,i}, x_{max,i}] \quad (30)$$

$$y_0 \in [y_{min,i}, y_{max,i}] \quad (31)$$

$$\dot{x}_0 \in [-v_{max}, v_{max}] \quad (32)$$

$$\dot{y}_0 \in [-v_{max}, v_{max}] \quad (33)$$

$$\psi_0 \in [0, 2\pi] \quad (34)$$

$$\dot{\psi}_0 \in [-\pi, \pi] \quad (35)$$

3.2 Constraints

The above initial condition constraints allow the vehicle to start off in any position and orientation in a given phase. Thus, state constraints are equivalent to the above equations as well.

Control input bounds are based on several assumptions. A primary assumption is that both the front and rear wheels can apply throttle and braking. Hence, the control input bounds on slip quantities are,

$$s_i \in [-1, 1] \tag{36}$$

The car wheels are assumed to be bounded as follows, slightly less constrained than a typical racing vehicle steering angle of $20 - 23^\circ$,

$$\delta \in [-30^\circ, 30^\circ] \tag{37}$$

To ensure the vehicle stays within the track throughput all phases, two varieties of constraints were considered.

The first was a path constraint, derived by considering the problem of identifying whether a point (vehicle) lies within a convex quadrilateral. Consider the cross product of a vector $((x_{i+1}, y_{i+1}) - (x_i, y_i))$ connecting two adjacent vertices of the quadrilateral, and vector $(p - (x_i, y_i))$ connecting the point (vehicle) to vertex i (see Figure 3). Given vehicle position $p = (x, y)$, the process is repeated for all four combinations of adjacent vertices. If the Z-component of the resulting cross products all have the same sign, then the point (vehicle) resides in the quadrilateral. However, such a path constraint can be costly to evaluate. To save computational effort, it can be noted from Figure 3 that the cross product needs only be checked for edges $((x_2, y_2) - (x_1, y_1))$ and $((x_4, y_4) - (x_3, y_3))$, since the same path constraints from the next phase of the track will ensure that the car never leaves the track, making it redundant to perform the cross product check on the two additional edges $((x_3, y_3) - (x_2, y_2))$ and $((x_1, y_1) - (x_4, y_4))$.

Using the above information, it can be shown that the region of transition for a given vehicle between two phases is a quadrilateral Q (indicated in Figure 3). This makes the use of manifold constraints redundant as well, as the vehicle will naturally tend towards the end of the current phase.

Despite the novel nature of the track constraint posed as above, initial experiments indicated that the use of such a path constraint was still highly expensive computationally, making it extremely difficult for GPOPS to solve the problem on interesting tracks.

Therefore, a simpler, computationally inexpensive phase manifold constraint was utilized in the final implementation. Let $(x_{i,tf}, y_{i,tf})$ be the final position of the vehicle in a given track phase i. Then three distance metrics can be defined using this position as well as the coordinates of the left and right side of the track at the end of the phase (see Figure 4),

$$d_1 = \left[(y_{tf} - yEdge_{1,i+1})^2 + (x_{tf} - xEdge_{1,i+1})^2\right]^{1/2}$$

$$d_2 = \left[(y_{tf} - yEdge_{2,i+1})^2 + (x_{tf} - xEdge_{2,i+1})^2\right]^{1/2}$$

$$d_3 = \left[(yEdge_{2,i+1} - yEdge_{1,i+1})^2 + (xEdge_{2,i+1} - xEdge_{1,i+1})^2\right]^{1/2}$$

Then a constraint ensuring that the vehicle finishes on the terminal manifold of the track phase is,

$$d_1 + d_2 \leq d_3 + \epsilon \tag{38}$$

Upon closer inspection, this is the equation of a region bounded by an ellipse, with its foci on the left side and right side of the track's terminal manifold. Initial experiments reveal that the square root function involved in all the distance metrics imposes heavy computational burden, due to the constraint being checked several thousand times in each run of GPOPS. To reduce this burden, it is noted that the distance metrics are all positive. Therefore, squaring both sides of the equation,

$$(d_1 + d_2)^2 \leq (d_3 + \epsilon)^2 \tag{39}$$

and expanding,

$$d_1^2 + 2d_1d_2 + d_2^2 \leq d_3^2 + 2d_3\epsilon + \epsilon^2 \tag{40}$$

Reorganzing the equation,

$$d_1^2 + d_2^2 \leq d_3^2 + 2(d_3\epsilon - d_1d_2) + \epsilon^2 \tag{41}$$

it is clear that ϵ can be chosen to be arbitrarily large enough that $2(d_3\epsilon - d_1d_2)$ is a positive quantity. Hence, defining $\hat{\epsilon}^2 = 2(d_3\epsilon - d_1d_2) + \epsilon^2$, the equation can be re-written as,

$$d_1^2 + d_2^2 \leq d_3^2 + \hat{\epsilon}^2 \tag{42}$$

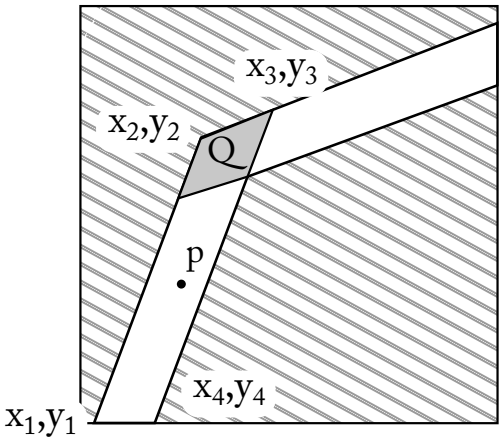


Figure 3: Track constraints can be described using a cross product rule, which allows manifold constraints to be avoided.

This new track constraint utilizes only the squares of the distance metrics, avoiding the square roots and lessening the computational complexity of the constraint checks. The value of $\hat{\epsilon}^2$ was determined empirically. For the given track sizes, a value of 0.025 was found to be suitable in ensuring the car finishes near the manifold of each track phase.

State constraints across phases were also imposed using eventgroups. For most of the tracks considered, each element of the 6-D state vector was allowed to vary across a phase within a tolerance of $1e-3$.

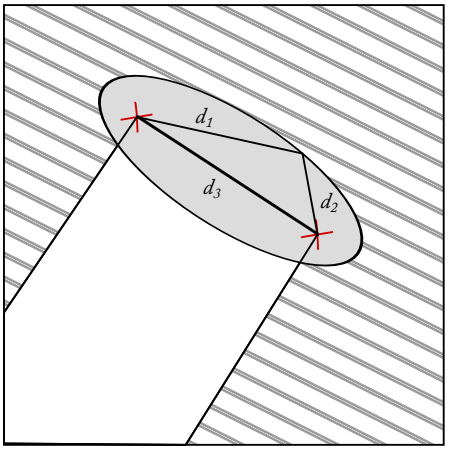


Figure 4: Manifold constraints can be imposed using an ellipse with its foci centered on the endpoints of each track phase.

Additionally, the vehicle top speed is imposed as a path constraint, as it cannot be exceeded throughout the entire track. The speed is obtained from the vehicle as,

$$v = \sqrt{\dot{x}^2 + \dot{y}^2} \tag{43}$$

with the lower and upper bounds being 0 and v_{max}, respectively.

3.3 Initial Guesses

The initial guess in a given phase i is constructed by aligning the vehicle with the track at the beginning and end of the phase, with $v = v_{max}/2$. The following equations summarize the initial guess,

$$x_{guess,i,start} = xCoords_i \tag{44}$$

$$y_{guess,i,start} = yCoords_i \tag{45}$$

$$\dot{x}_{guess,i,start} = \frac{v_{max}}{2}\frac{dx_i}{dL_i} \tag{46}$$

$$\dot{y}_{guess,i,start} = \frac{v_{max}}{2}\frac{dy_i}{dL_i} \tag{47}$$

$$\psi_{guess,i,start} = atan2(dy_i, dx_i) \tag{48}$$

$$\dot{\psi}_{guess,i,start} = 0 \tag{49}$$

$$x_{guess,i,end} = xCoords_{i+1} \tag{50}$$

$$y_{guess,i,end} = yCoords_{i+1} \tag{51}$$

$$\dot{x}_{guess,i,end} = \frac{v_{max}}{2}\frac{dx_{i+1}}{dL_{i+1}} \tag{52}$$

$$\dot{y}_{guess,i,end} = \frac{v_{max}}{2}\frac{dy_{i+1}}{dL_{i+1}} \tag{53}$$

$$\psi_{guess,i,end} = atan2(dy_{i+1}, dx_{i+1}) \tag{54}$$

$$\dot{\psi}_{guess,i,end} = 0 \tag{55}$$

The control guess is constructed in a simpler manner, by assuming that the car is slightly accelerating in all phases. The steering angle is assumed to be 0, since the curvature of the track is not explicitly calculated. The following equations summarize the initial guess for control,

$$s_{ix} = [0.1; 0.1],\ i = F, R \tag{56}$$

$$\delta = [0; 0] \tag{57}$$

3.4 Refined Initial Guessing

Using the above initial guessing scheme, GPOPS is able to compute the optimal control for a track consisting of 5-6 phases. With more phases, GPOPS typically reports that the problem is "infeasible", or simply runs out of iterations before mesh tolerances are met.

A refined guessing scheme is constructed by incrementally building up the total number of phases, using past solutions as initial guesses for the earlier phases. For the latest phase, however, the guess is constructed as in the previous guessing scheme. The pseudo-code is presented in Algorithm 1.

Data: $track, numPhasesTotal$
Result: u_{opt}
$numPhasess \leftarrow 1$;
$solution \leftarrow$ racinglineMainDyn(numPhases);
for $numPhases = 1$ *to* $numPhasesTotal$ **do**
  for $iphase = 1$ *to* $numPhases\text{-}1$ **do**
    $guess(iphase) \leftarrow solution(iphase)$;
  end
  $solution \leftarrow$ racinglineMainDyn(numPhases, guess);
end
$u_{opt} \leftarrow solution(:).control(:,:)$;

Algorithm 1: Refined guessing algorithm

4 Kinematic Model Results

Initial trials with GPOPS were conducted using the kinematic model in order to ascertain computational performance and correct problem setup in MATLAB, prior to converting to a full dynamical model. The kinematic, constant-speed model was tested on a single-phase straight track, along with a multi-phase U-turn track. The resulting trajectory for the U-turn track is visualized in Figure 5, with the car's initial position indicated as a red circle. Note that the car behaves as expected, steering left in order to hit the turn's apex.

Additionally, the car hits an early apex, and leaves the final phase going straight. The early apex behavior is as expected, as it allows the car to take the shortest straight-line distance to the next phase. The late apex behavior is, at first, counter-intuitive as it appears that a shorter total time might have been possible if the vehicle continued to hug the turn. However, once consideration for the ellipse manifold terminal constraint (Figure 4) is made, the trajectory makes sense. The ellipse manifold is notably larger in the middle of the track rather than the sides. Therefore, at the final phase, the car will prefer to finish in the middle of the track rather than on the sides. Note that this is also a consequence of the constant-speed kinematic model.

In the dynamical model, more interesting trajectories are obtained as the vehicle has the opportunity to accelerate out of the turn. This concludes analysis of the kinematic model, as it was simply a means of evaluating initial GPOPS performance. More detailed control analysis is performed for the dynamical model.

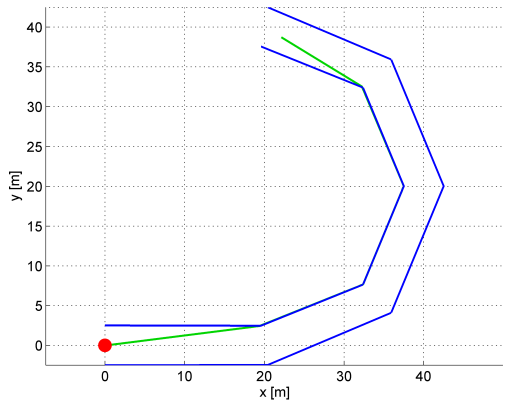


Figure 5: Kinematic model optimal trajectory for a U-turn. The car's initial position is indicated in red.

5 Dynamical Model Results

Full dynamical model trials were conducted on a straight track, a U-turn track, an S-curve track, as well as a track based on the Formula 1 Australian GP circuit.

5.1 Friction Ellipse

Slip quantities s_{ij}, $i = F, R$ $j = x, y$ correspond directly to longitudinal and lateral friction forces, as shown in Equation 3. Thus, for a range of slip quantities, the corresponding friction force can be calculated directly. This result is plotted in Figure 6 for the longitudinal direction. As shown on the plot,

a maximum friction force can be calculated, which corresponds to the point of maximum acceleration for the car. If the slip quantity increases above this point, the car loses traction and acceleration reduces. If the slip quantity decreases, then the driver is not exerting maximum control input and the overall track time will be sup-optimal.

For a given set of vehicle dynamical parameters and road/tire friction characteristics (captured through Pacejka's Magic Formula), the maximum possible friction force leads to the concept of the friction ellipse. Using Pontryagin's Minimum Principle, it can be concluded that for a car to be optimally on a track, the total friction force must lie on the friction ellipse. More comments on this behavior will be made for the specific tracks considered below.

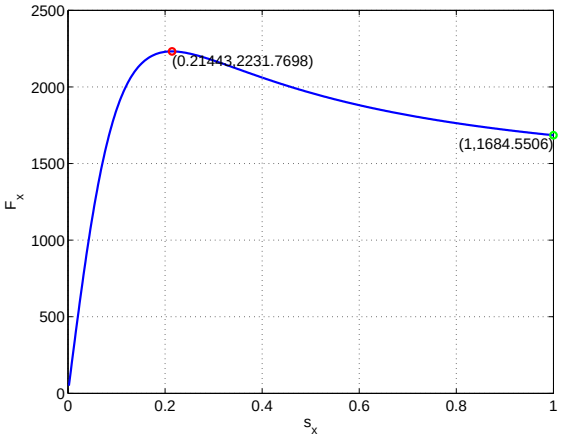


Figure 6: Longitudinal friction force vs. longitudinal slip quantity. The maximum friction value is also indicated, corresponding to the radius of the friction circle.

5.2 Straight Track Results

The optimal racing line for the straight track is shown in Figure 7. More detailed results, including vehicle velocity, control inputs, and friction ellipse plots are shown in Figure 8.

Note that the vehicle speed is indicated by the color of the racing line (with red hues indicating a slower speed, and green hues indicating speeds near v_{max}). Intuitively, the vehicle accelerates on a straight line along the track. As Figure 8(b) reveals, $\dot{y} = 0$ for the entire duration of the track. On the other hand, $\dot{x}$ increases until top speed is hit.

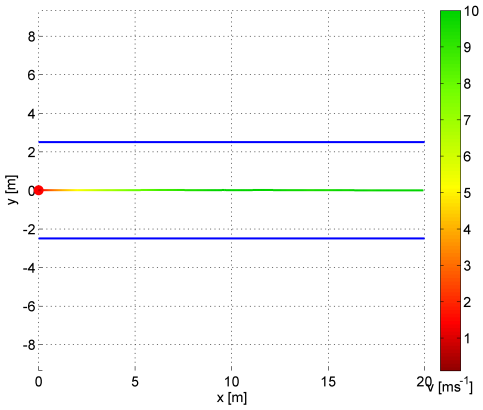
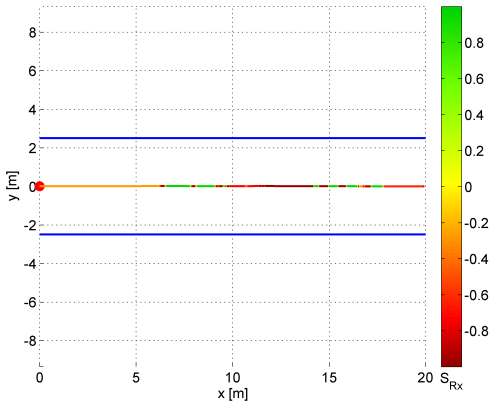


Figure 7: Straight track optimal trajectory and speed. Red marker indicates starting position.

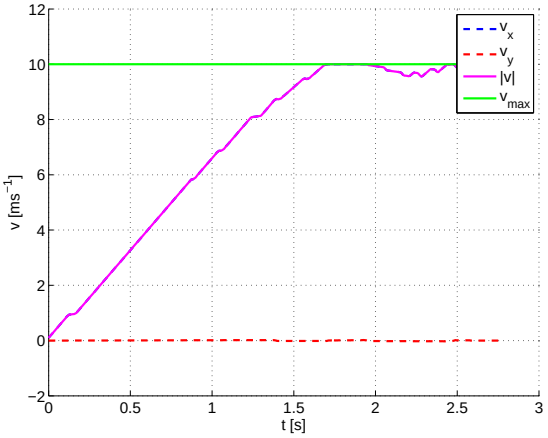
The control input plots reveal that while the car is accelerating, both the front and rear wheels have a slip quantity which lead the vehicle to have maximal friction force (the same maximum force indicated in Figure 6). This control input continues until the vehicle accelerates to v_{max}, at which point oscillations between maximum braking and maximum acceleration occur in order to maintain speed at v_{max}.

Additionally, at a large number of instances in Figure 8(d), the slip quantities reach their maximum and minimum allowable values, +1 and -1. It is hypothesized that this issue is caused by the numerical methods used by GPOPS to solve the problem. It is likely that reducing the convergence tolerances in GPOPS will help remove these discontinuities. However, at the same time, these have minimal impact on the car trajectory as slip quantities of +1 or -1 correspond to zero traction, meaning that the vehicle does not accelerate or decelerate.

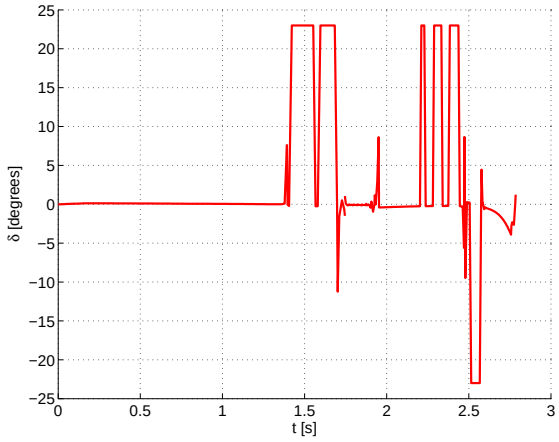
Figure 8(e) indicates the longitudinal and lateral friction forces of the front and rear tires, respectively. Note that for the majority of the track, the car is applying maximum longitudinal friction (top of the friction ellipse). Once top speed is reached, the input oscillates between the top and bottom of the friction ellipse, in order to satisfy the velocity constraint. The friction data appearing inside the ellipse is due to the discontinuous control inputs mentioned earlier, which are a result of the numerical approximations used by GPOPS.



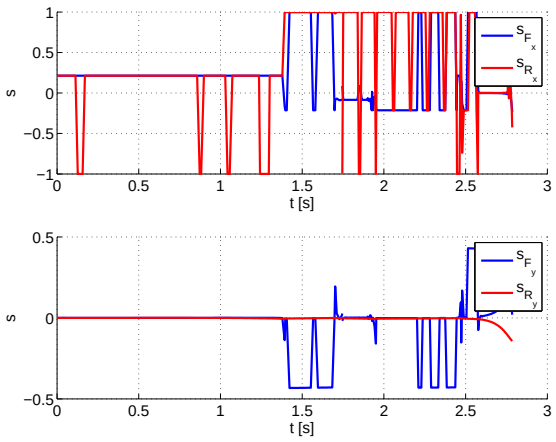
(a) Optimal trajectory and input S_{Rx}.



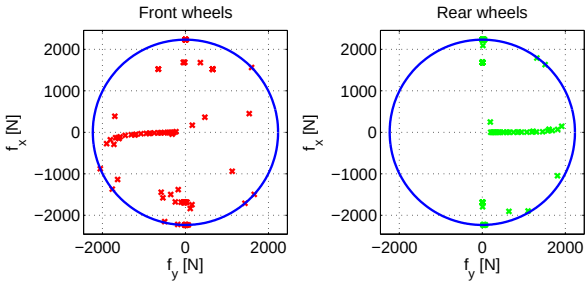
(b) Speed profile.



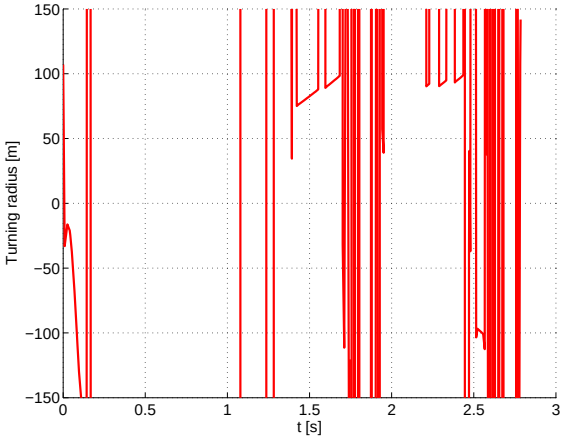
(c) Steering control input history.



(d) Throttle control input history.



(e) Friction ellipse and slip quantities.



(f) Turning radius.

Figure 8: Straight track results

5.3 U-Turn Track Results

Figure 9 shows the optimal trajectory for the U-Turn track. Figure 10 shows detailed optimal solution plots for the track. The behavior of the vehicle on this track is much more interesting. For instance, note in Figure 9 that the vehicle smoothly hits the apex of the turn upon entry, and leaves with a near straight line motion in order to maximize acceleration. This behavior is also seen in the speed plot (Figure 10(b)). At around $t = 5s$, the vehicle clears the turn and starts accelerating towards the finish line.

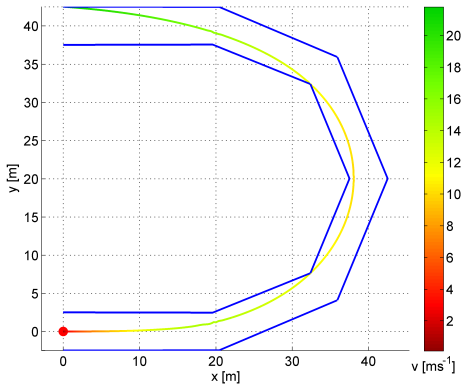


Figure 9: U-Turn track optimal trajectory and speed. Red marker indicates starting position.

Additionally, the speed profile in Figure 10 indicates that the vehicle slows into the turn early, at around $t = 2.1s$, in order to decrease cornering radius and hug the inside of the curve tightly. Due to the discretized nature of the track, as well as the nature of the terminal constraint manifolds for each phase, the vehicle trajectory hits the jagged edges of the track. This phenomenon is explained easily when consideration is made for the dynamical nature of the vehicle model.

For the vehicle to perfectly hug the inside of the track, including at the corner points, the velocity must be drastically reduced in order to allow for a nearly infinite trajectory curvature. However, this is infeasible as it tremendously increases the overall time objective. Hence, unlike the kinematic model (which does indeed hug the track perfectly), the dynamical model instead takes the approach of balancing vehicle speed with distance from the inner edge of the track.

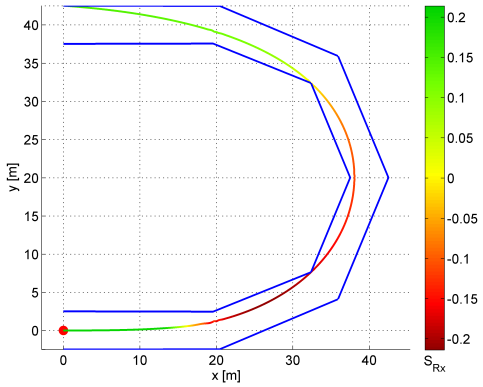
As Figure 10(d) indicates, the U-Turn track does not exhibit the same oscillatory slip quantity input between +1 and -1 which the straight track did. Looking at the velocity plot 10(b) in conjunction, it is concluded that since the vehicle does not reach top speed, the same discontinuous behaviors exhibited in the straight track are disallowed. The control effort is fully dedicated to balancing longitudinal acceleration or decreasing cornering (lateral) radius on this track. This also explains why the control effort is fully located on the perimeter of the friction ellipse plots in Figure 10(e).

Figure 10(f) reveals that the vehicle has infinite turning radius until approximately $t = 10s$, at which point the steering input (Figure 10(c)) turns away from the curve and then back towards the curve in order to lead the car into the U-Turn. At this point, the lateral slip quantities s_{iy} $(i = F, R)$ increase nearly to their maximal limits, in order to perform a hard turn to the left. The turning radius then approache $20m$, the radius of the U-Turn, after which it drops off to infinity again due to the track straightening out. The lateral slip quantities also reveal that early into the turn, the front wheel has a slightly larger slip than the rear wheel. On the other hand, when the turn concludes, the rear wheel has a larger slip quantity. These results correspond to the dynamical model used, since the vehicle has non-negligible length and the behavior of the front wheels tends to lead the behavior of the rear wheels.

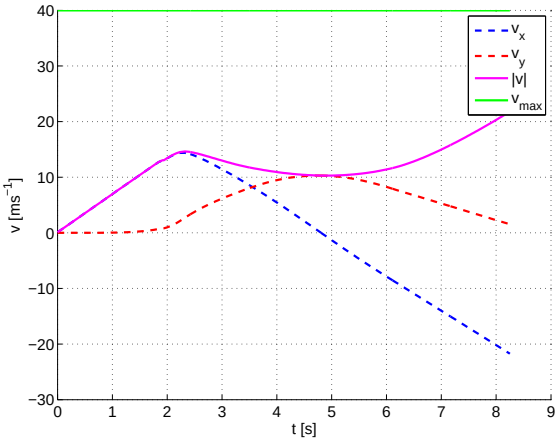
5.4 S-Curve Track Results

Figure 11 shows the optimal trajectory for the S-Curve track. Figure 12 shows detailed optimal solution plots for the track. This trajectory is fairly similar to the U-Turn trajectory, with the exception that the vehicle is able to retain its longitudinal velocity much more efficiently as it does not have to turn excessively. In fact, as Figure 12(b) shows, the vehicle is able to hit top speed at the end of the track.

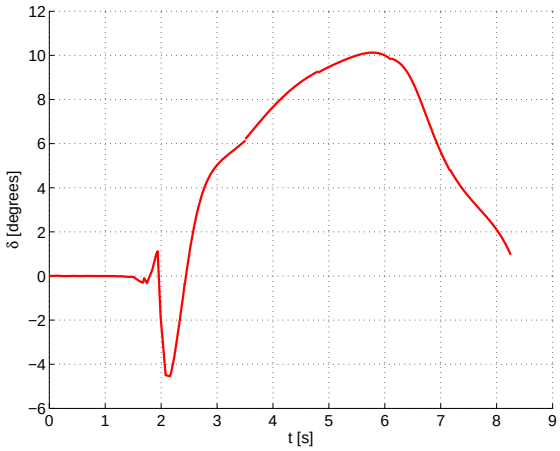
In addition, the friction ellipse plots in Figure 12(e) are more symmetrical than those for the U-Turn track, due to the presence of two opposite turns in the S-Curve. Another interesting behavior is the "leading in" that the steering angle performs whenever the car approaches the turn. For instance, at $t = 10s$ in Figure 12(c), the car steers right, before taking a hard left into the first turn. A conjugate behavior is repeated when coming out of the S-Curve, at $t = 12s$.



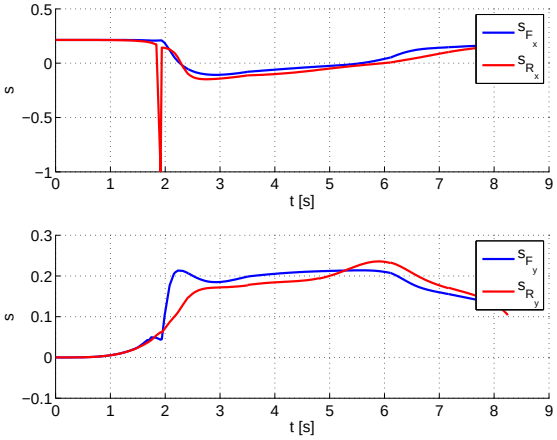
(a) Optimal trajectory and input S_{Rx}.



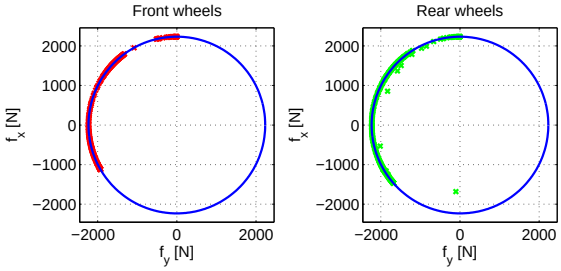
(b) Speed profile.



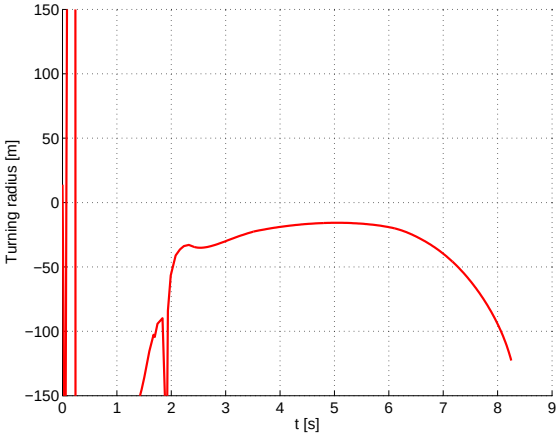
(c) Steering control input history.



(d) Throttle control input history.



(e) Friction ellipse and slip quantities.



(f) Turning radius

Figure 10: U-Turn track results

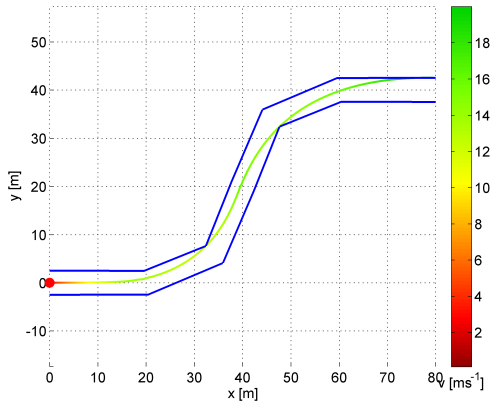


Figure 11: S-Curve track optimal trajectory and speed. Red marker indicates starting position.

Another interesting note is that the friction plots (Figure 12(e)) reveal the rear wheels to be doing more braking than the front wheels. This behavior is also observed for the U-Turn track, and resembles the behavior of race car driving using the front throttle and rear brake simultaneously in order to drift the car around a turn.

5.5 Formula 1 Australian GP Track Results - Dry Track

To ascertain vehicle performance in the real world, the Formula 1 Melbourne Grand Prix Circuit was modeled, with the vehicle placed at the usual starting line of the track for F1 vehicles. The F1 track was constructed by utilizing longitude and latitude coordinates from Google Maps, along with MATLAB's mapping toolbox for changing from a mercator to a cartesian coordinate system.

However, vehicle dynamics were not tuned to match F1 vehicles, as aerodynamic forces (downforce and drag) have considerable impact on F1 performance, but were not modeled in this report.

The F1 track model consisted of 80 phases in total. However, solving for optimal trajectories on such a large number of phases was determined to be infeasible in MATLAB, even after redesigning the initial guessing scheme in order to use previous phases' results. Instead, the code was redesigned to allow cutoff at any given phase, meaning that the program could be run for several hours and simply terminated, with the results for the most up-to-date number of phases saved.

Figure 17 shows the optimal trajectory of the vehicle on the F1 track. This behavior is consistent with those observed in the U-Turn and S-Curve tracks, with the car veering to the opposite direction of a corner before turning back in order to hit the apex.

Additionally, the slip control inputs in Figure 18(c) appear much noisier than on the other tracks. This is due to the significant length of straightaways on this track, allowing the vehicle to hit top speed, at which point the control oscillates between maximum and minimum bounds (as well as maximum and minimum traction limits) to hold the vehicle speed at v_{max}.

A final note is the presence of an apparent "inner" friction ellipse in Figure 18(d). These lower friction forces correspond to instances where the optimal slip control is +1 or -1, as mentioned earlier, and are indicated in Figure 18(e).

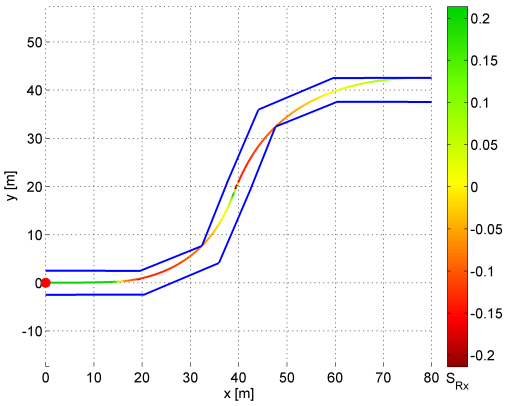
5.6 Formula 1 Australian GP Track Results - Wet Track

The effects of changing tire/road interaction parameters (or traction limits) were also considered. Instead of using the Pacejka tire parameters shown in Table 1, the parameters in Table 2 were utilized, which correspond to a wet road. Note that the top speed was also increased to $29ms^{-1}$. The corresponding friction limit graph is shown in Figure 20(e), with a maximum friction force value of 2040N, compared to the dry road's maximum friction force of 2231N.

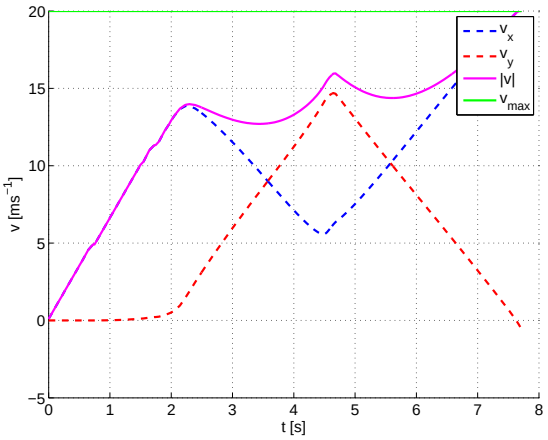
Figure 21 shows a comparison of racing lines for dry and wet conditions. Note that the turning radius is, on average, notably larger on the wet track than the dry track. This makes intuitive sense, as a wet tarmac makes for a more difficult car to control, with the lack of friction reducing the total amount of lateral acceleration the car can obtain during a turn.

Parameter	Value
B	7
C	1.6
D	0.60

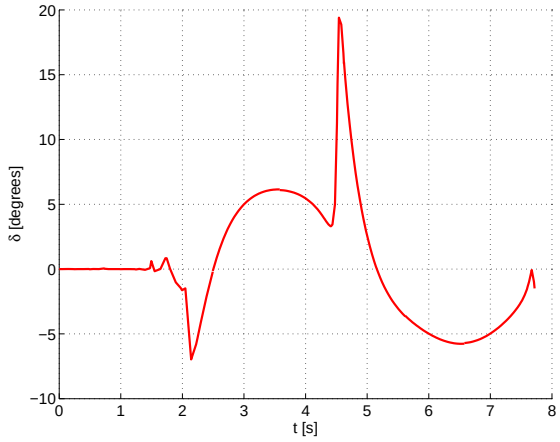
Table 2: Pacejka parameters for a wet road



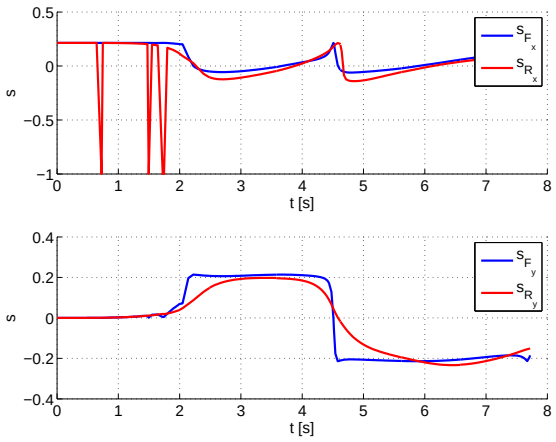
(a) Optimal trajectory and input S_{Rx}.



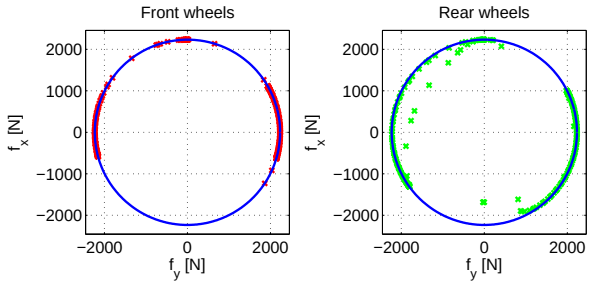
(b) Speed profile.



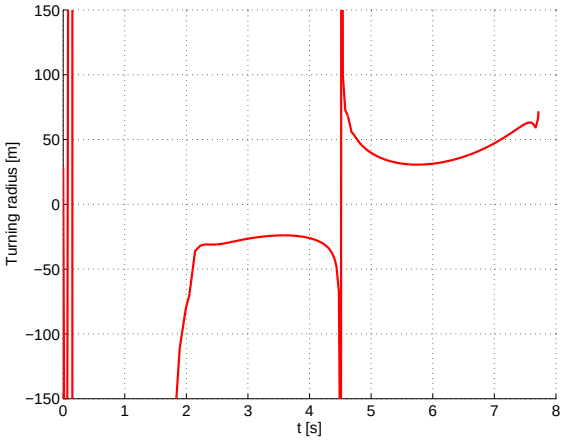
(c) Steering control input history.



(d) Throttle control input history.



(e) Friction ellipse and slip quantities.



(f) Turning radius.

Figure 12: S-Curve track results

6 Constraint Satisfaction

The minimum time racing line problem, as posed, imposes several constraints. Dynamical and state constraints are verified to be met through GPOPS output log for all tracks, which indicates satisfaction of these constraints. A more interesting constraint is that on the top speed, v_{max}, which is verified to be met in the speed plots (see Figures 8(b), 10(b), 12(b), 18(b)).

Control input constraints are also verified to be met, although in some cases (especially in longer tracks) GPOPS sets control inputs to maximum or minimum limits. It is conjectured that these discontinuities are caused by slightly loose convergence tolerances set in the problem configuration. The track constraint was posed in the form of a terminal elliptical manifold for each phase, as shown in Figure 4. As trajectory Figures 8(a), 10(a), 12(a), and 18(a) indicate, the vehicle always passes inside the terminal manifold of each phase. Note that the phases can be visually seen as discontinuities in the track, as the track was discretized.

7 Computational Optimization

Due to the large state space considered in the problem, as well as the large number of phases in the tracks, a variety of methods were used to allow the code to converge faster. These computational optimization techniques were deemed noteworthy and of possible use to future students.

The MATLAB profiler (Figure 13) was extremely useful in determining code bottlenecks and avoiding micro-optimization. Initial trials revealed that significant time was spent in both the `continuous.m` and `endpoint.m` functions in GPOPS, despite the functions' relative simplicity. This was determined to be due to GPOPS calling each function up to several hundred thousand times, as seen in Figure 13. Efforts were placed in optimizing the code in several ways, outlined in Table 3.

Optimization Level 1 involved using the "@double class" MATLAB toolbox, obtained from [12]. This toolbox offers improvements in runtime for elementary functions in MATLAB, such as `sqrt()` and trig functions, which appear to be a major bottleneck in the code based on the profiler. A full plot of various elementary function runtime improvements is presented in Figure 14. On average, runtimes decrease by factors of 2 to 7. For a 4 phase track, this

racinglineOptLevel4	1	61.271 s	0.184 s	
gpops2	1	60.839 s	0.013 s	
gpopsMeshShell	1	60.617 s	0.009 s	
gpopsMeshHP1	1	60.564 s	0.022 s	
gpopsSolveRPMI	1	60.366 s	0.021 s	
gpopsIpoptHandlerRPMI	1	60.232 s	0.010 s	
ipopt (MEX-file)	1	60.158 s	19.134 s	
gpopsIpoptHesRPMI	537	31.107 s	0.318 s	
gpopsHesnzRPMI	537	30.789 s	2.606 s	
gpopsContSecondCD	537	13.881 s	5.051 s	
gpopsEndpSecondCD	537	12.051 s	3.358 s	
racinglineEndpointDyn	198999	11.663 s	11.663 s	
racinglineContinuousDyn	76873	11.626 s	11.626 s	
gpopsIpoptJacRPMI	561	8.341 s	0.944 s	
gpopsJacnzRPMI	561	7.397 s	1.208 s	
gpopsContFirstCD	1098	4.231 s	1.731 s	
gpopsEventFirstCD	561	3.681 s	1.162 s	

Figure 13: MATLAB code profiler

Opt. Level	Description	Runtime [s]
0	No optimization	60.01
1	MATLAB @double toolbox	55.98
2	Struct initialization	50.91
3	Improved initial guesses	**28.64**

Table 3: Optimization methods, details, and runtimes for 4 phase track. Note that each level builds on top of previous levels.

results in a runtime improvement of 5 seconds, or approximately 9%.

After implementation of the @double class, it was noted that a significant amount of time was spent initializing the `phaseout` struct in GPOPS' `continuous.m` function. This can be seen on line 63 in Figure 15. By instead pre-allocating the struct (Figure 16), an additional 5 second improvement was obtained in Optimization Level 2.

Optimization Level 3 involved implementing the improved initial guessing method discussed earlier, by using previous results as guesses for longer phase tracks. This resulted in a significant improvement, seen in Table 3.

Overall, computational optimization of the code led to runtime improvements of more than 50% for a 4 phase track. For longer tracks, this improvement was even more pronounced.

It is concluded that simple optimization techniques for a dynamically-typed language such as MATLAB can still yield significant runtime improvements. For future work, an investigation of porting the `continuous.m` and `endpoint.m` functions to a compiled language (or MEX files) can be

conducted.

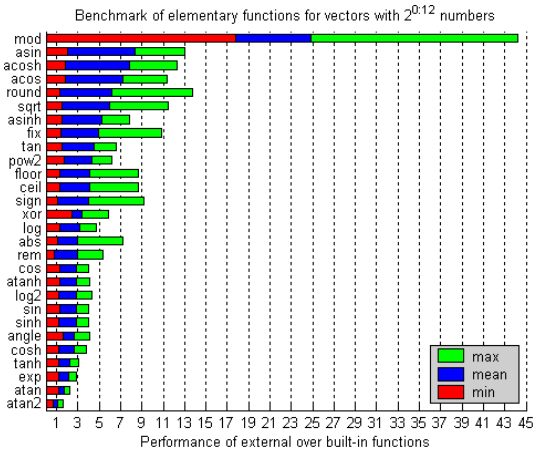


Figure 14: MATLAB @double class elementary function improvements [12].

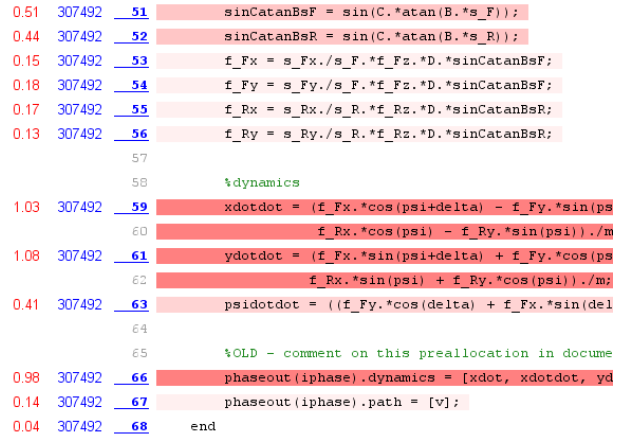


Figure 16: MATLAB code profiler results after Optimization Level 2. Line 66 (the same struct set in Line 63 of Figure 15) has its runtime significantly reduced due to struct pre-allocation.

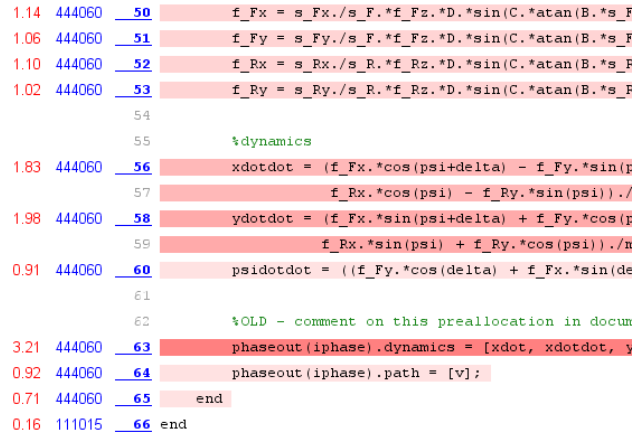


Figure 15: MATLAB code profiler results before Optimization Level 2. Note that th struct on line 63 has not been initialized, and is more red (indicating longer relative time spent on it)

8 Conclusion

Optimal control of a vehicle dynamical model was performed in order to minimize total race time. The chosen vehicle model was a single-track dynamical model, including tire friction forces as governed by Pacejka's Magic Formula. Two forms of the track constraint were considered, one being a path constraint ensuring that the vehicle never leaves the track boundaries in a given phase. The second was a less constraining manifold constriant (for each phase), treating the track similar to a series of slalom gates. The latter constraint was chosen in the final implementation in order to allow for extremely long tracks, up to 18 phases, to be solved.

It was noted that the control inputs of the vehicle typically stay on the perimeter of the friction ellipse. However, due to top speed constraints, inputs within the friction ellipse were occasionally occurred as well. These were determined to be due to loose convergence tolerances set in GPOPS, due to the level of complexity of the problem.

Future work includes an extension of the dynamical model to include simple notions of aerodynamic forces, such as downforce and drag, in addition to investigation of further computational performance improvements by implementing the code in a compiled language such as C or C++.

References

[1] Velenis, E., "Analysis and Control of High-Speed Wheeled Vehicles". *PhD Thesis*, 2006.

[2] Jeon, J., "Optimal Motion Planning with the Half-Car Dynamical Model for Autonomous High-Speed Driving". *ACC*, 2013.

[3] Pepy, R., "Path Planning using a Dynamic Vehicle Model". *ACC*, 2013.

[4] Velenis, E., "Optimality Properties and Driver Input Parameterization for Trail-braking Cornering". *EUCA*, 2008.

[5] Campbell, S., "Steering Control of an Autonomous Ground Vehicle with Application to the DARPA Urban Challenge". *Master of Science Thesis*, 2007.

[6] Rucco, A., "Computing Minimum Lap-time Trajectories for a Single-Track Car with Load Transfer". *51st IEEE Conference on Decision and Control*, 2012.

[7] Perantoni, G., "Optimal Control for a Formula One Car With Variable Parameters". *Vehicle System Dynamics 2014*, 2014.

[8] Xiong, Y., "Racing Line Optimization". *Master of Science Thesis*, 2010.

[9] Beltman, F., "Optimization of Ideal Racing Line". *BMI Paper*, 2008.

[10] Gustafsson, T., "Computing the Ideal Racing Line Using Optimal Control". *Thesis*, 2008.

[11] Pacejka, H., "Tyre modelling for use in vehicle dynamics studies". *SAE Paper No. 870421*, 1987.

[12] Leutenegger, M., "MATLAB Toolbox: double class". Available at `http://documents.epfl.ch/users/l/le/leuteneg/www/MATLABToolbox/DoubleClass.html`, 2014.

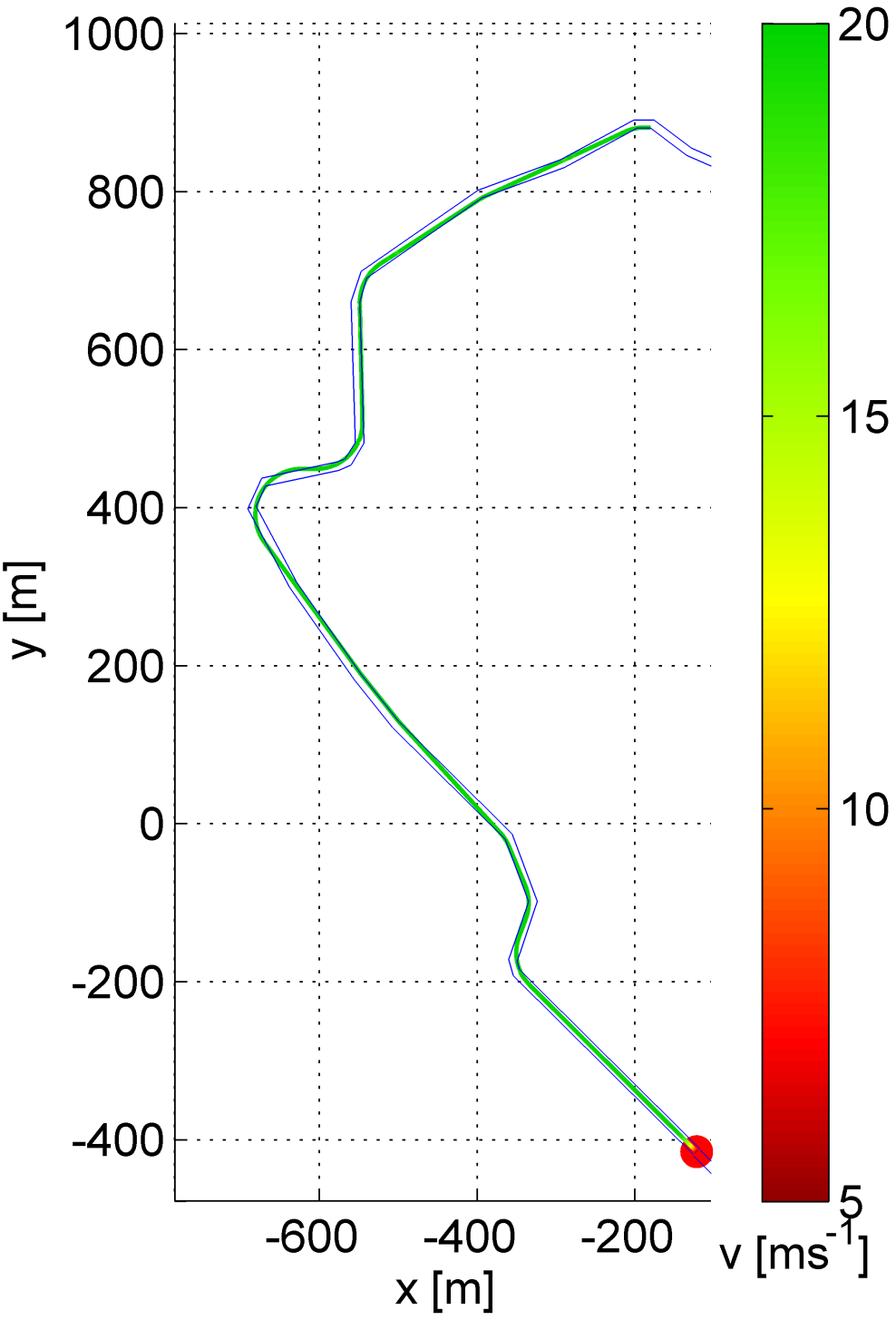
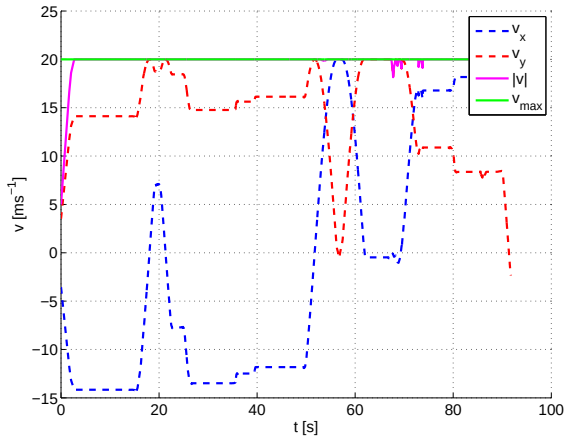
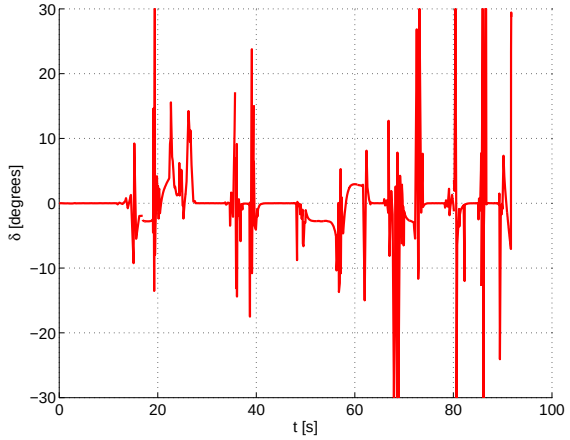


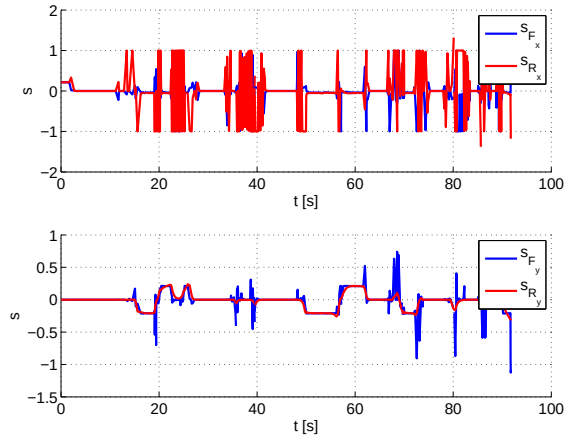
Figure 17: Formula 1 Australian GP track sector, calculated optimal trajectory and speed. Dry conditions.



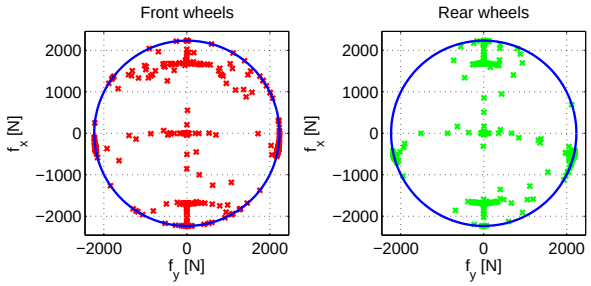
(a) Speed profile.



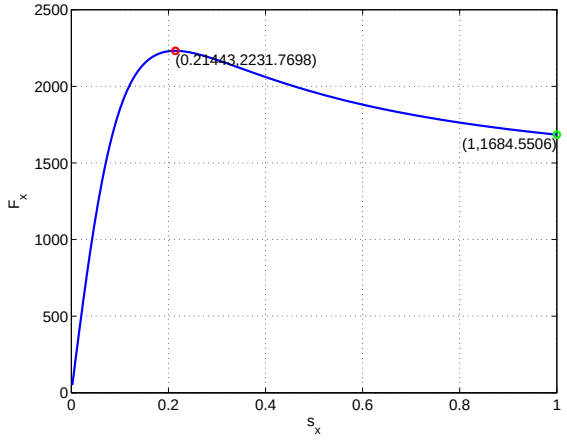
(b) Steering control input history.



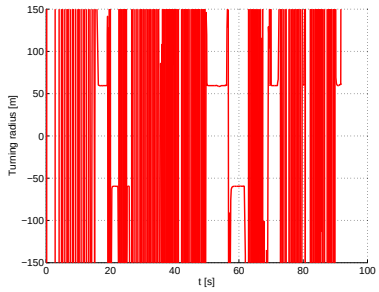
(c) Throttle control input history.



(d) Friction ellipse and slip quantities.



(e) Longitudinal friction force vs. longitudinal slip quantity.



(f) Turning radius.

Figure 18: Formula 1 Australian GP track results. Dry conditions.

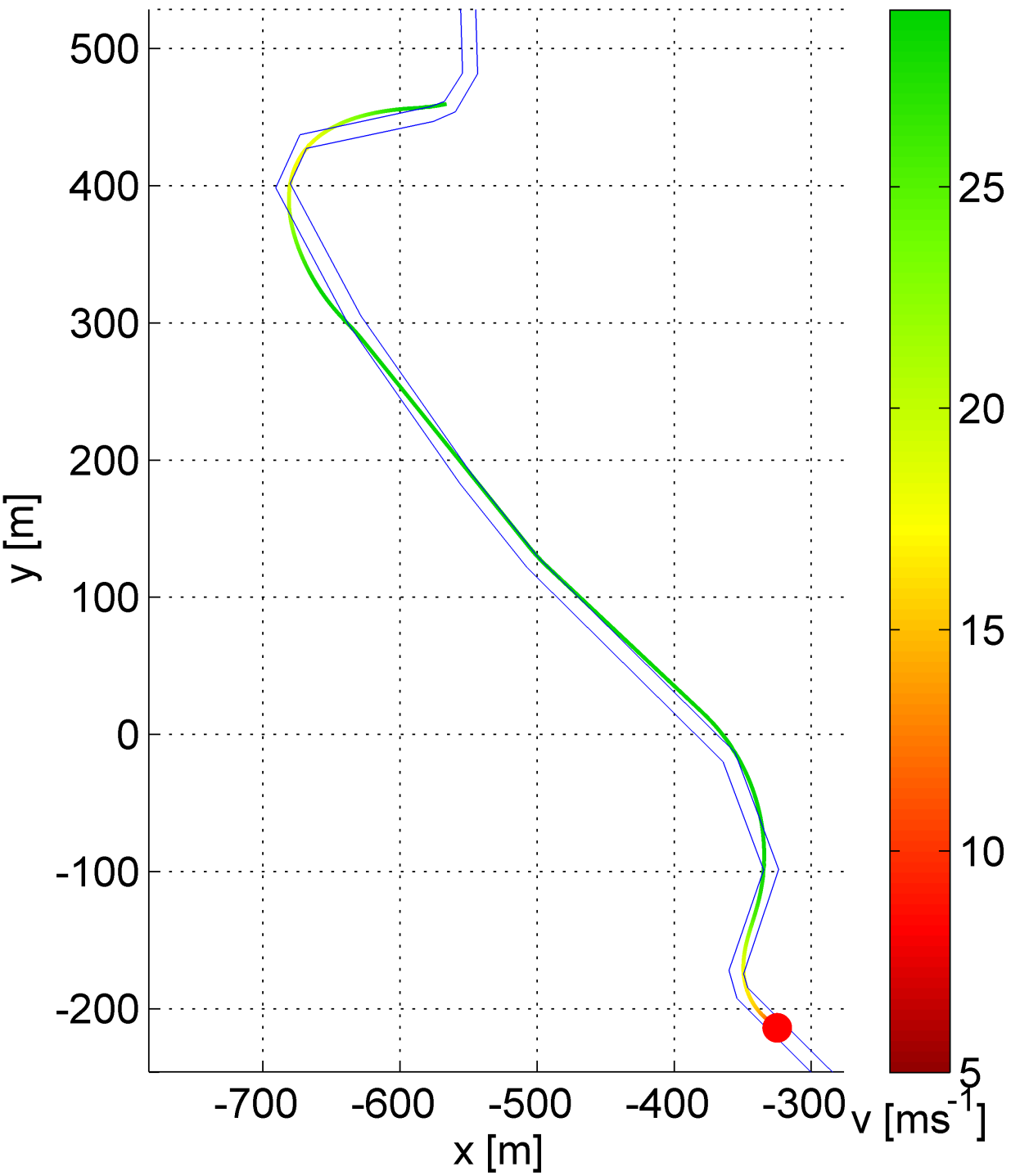
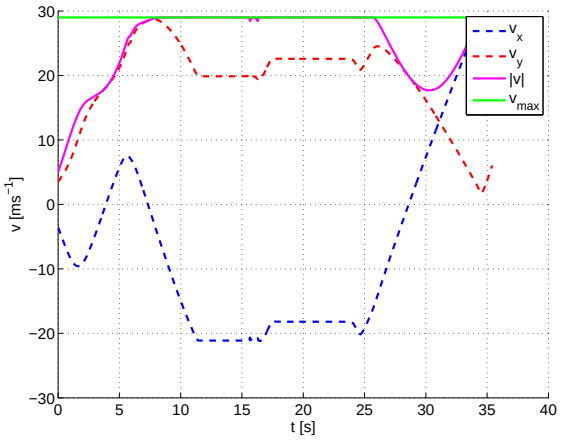
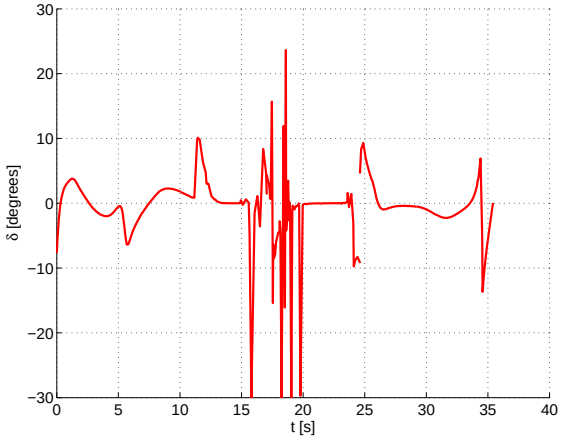


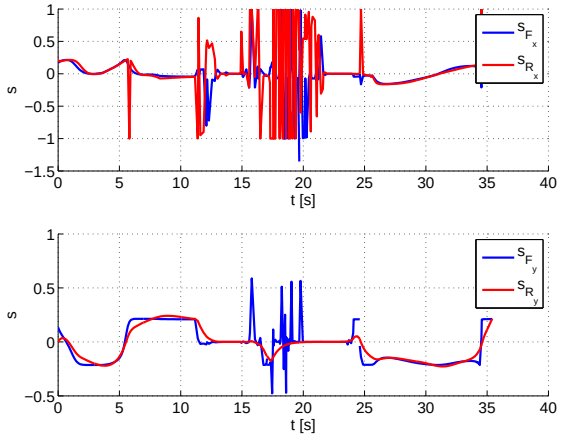
Figure 19: Formula 1 Australian GP track sector, calculated optimal trajectory and speed. Wet conditions.



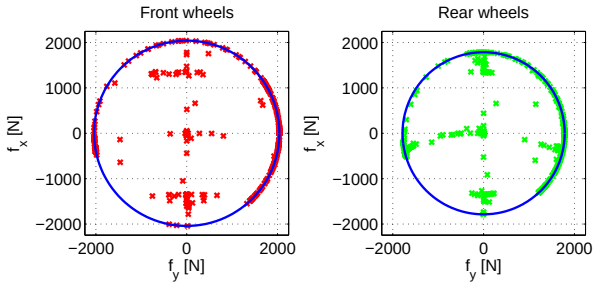
(a) Speed profile.



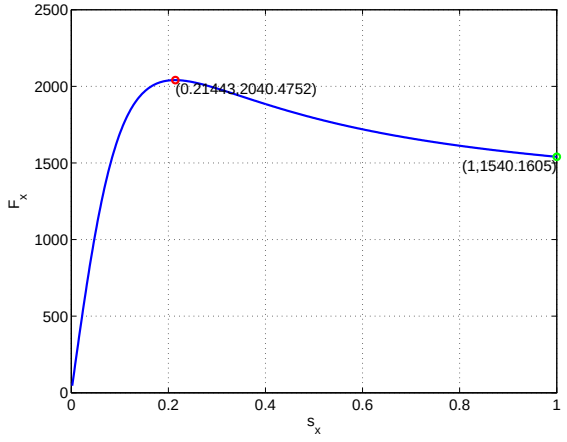
(b) Steering control input history.



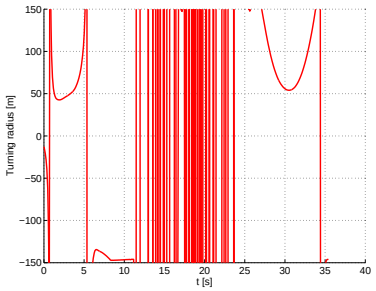
(c) Throttle control input history.



(d) Friction ellipse and slip quantities.



(e) Longitudinal friction force vs. longitudinal slip quantity.



(f) Turning radius.

Figure 20: Formula 1 Australian GP track results. Wet conditions.

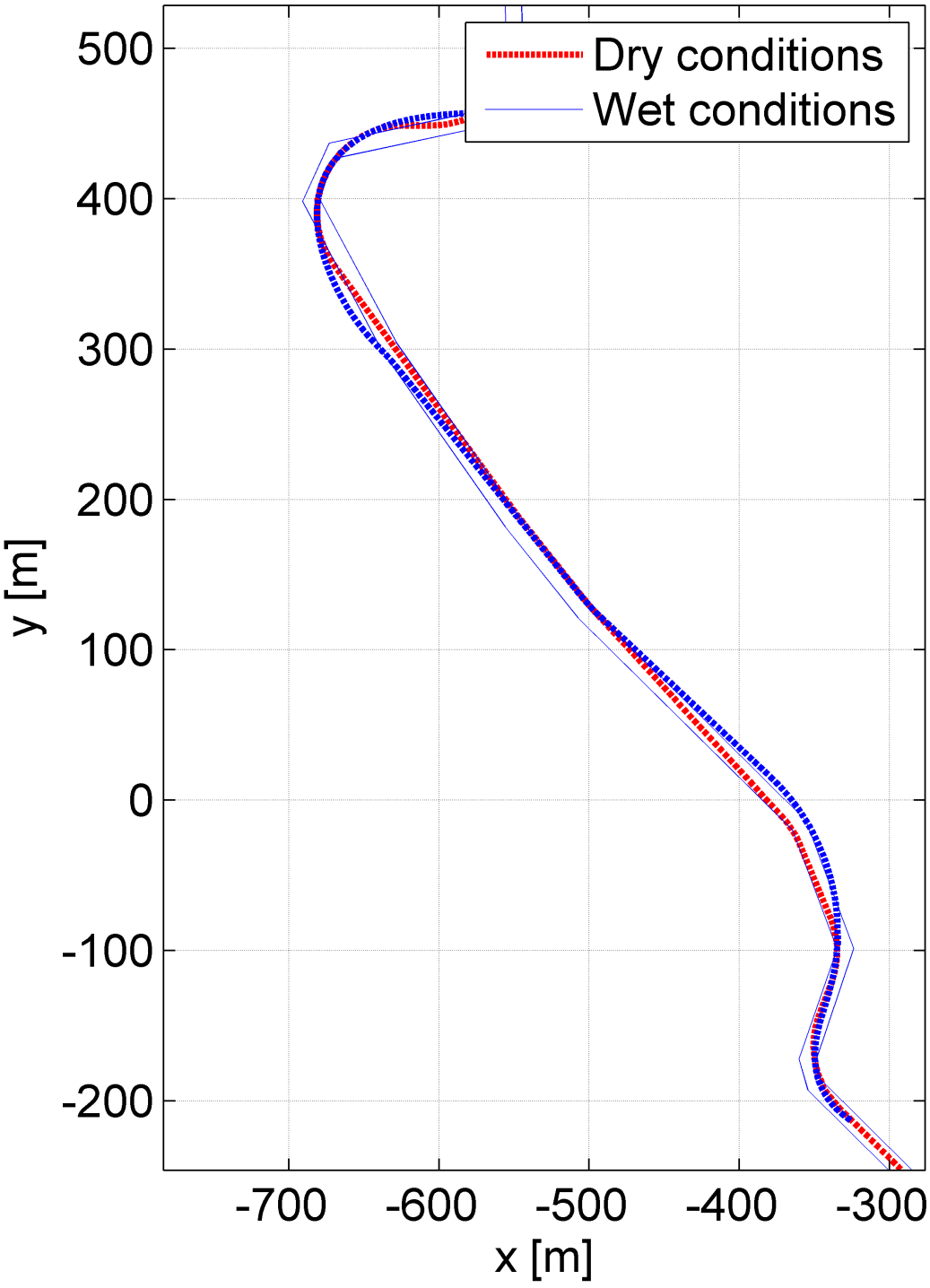


Figure 21: Formula 1 Australian GP track sector, calculated optimal trajectory. Comparison of dry and wet conditions.