



Universidade Federal da Bahia
Instituto de Computação / Escola Politécnica

Programa de Pós-Graduação em Mecatrônica

**DESENVOLVIMENTO DE NOVO MODELO
VEICULAR QUASE-ESTÁTICO E NOVO
ALGORITMO DE OTIMIZAÇÃO PARA A
PREDIÇÃO DO TEMPO MÍNIMO DE
VOLTA.**

ALAM ROSATO MACÊDO

DISSERTAÇÃO DE MESTRADO

Salvador
22 de dezembro de 2023

ALAM ROSATO MACÊDO

**DESENVOLVIMENTO DE NOVO MODELO VEICULAR
QUASE-ESTÁTICO E NOVO ALGORITMO DE OTIMIZAÇÃO
PARA A PREDIÇÃO DO TEMPO MÍNIMO DE VOLTA.**

Esta Dissertação de Mestrado foi apresentada ao Programa de Pós-Graduação em Mecatrônica da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre em Mecatrônica.

Orientador: MARCUS VINÍCIUS AMERICANO DA COSTA FILHO

Salvador
22 de dezembro de 2023

Ficha catalográfica elaborada pela Biblioteca Bernadete
Sinay Neves, Escola Politécnica - UFBA.

M141 Macêdo, Alam Rosato.

Desenvolvimento de novo modelo veicular quase-estático e novo algoritmo de otimização para a predição do tempo mínimo de volta/ Alam Rosato Macêdo. – Salvador, 2023.

137 f.: il. color.

Orientador: Prof. Dr. Marcus Vinícius Americano da Costa Filho.

Dissertação (mestrado) – Programa de Pós-Graduação em Mecatrônica, Escola Politécnica, Universidade Federal da Bahia, 2023.

1. Veículo – modelagem. 2. Veículo - competição. 3. Otimização. 4. Simulação. I. Costa Filho, Marcus Vinícius Americano da. II. Universidade Federal da Bahia. III. Título.

CDD: 629.282

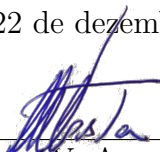
TERMO DE APROVAÇÃO

ALAM ROSATO MACÊDO

DESENVOLVIMENTO DE NOVO MODELO VEICULAR QUASE-ESTÁTICO E NOVO ALGORITMO DE OTIMIZAÇÃO PARA A PREDIÇÃO DO TEMPO MÍNIMO DE VOLTA.

Esta Dissertação de Mestrado foi julgada adequada à obtenção do título de Mestre em Mecatrônica e aprovada em sua forma final pelo Programa de Pós-Graduação em Mecatrônica da Universidade Federal da Bahia.

Salvador, 22 de dezembro de 2023



Prof. Dr. Marcus V. Americano da Costa
Universidade Federal da Bahia

Prof. Dr. Daniel Diniz Santana
Universidade Federal da Bahia

Prof. Dr. Gustavo Artur de Andrade
Universidade Federal de Santa Catarina

✓

Dedico este trabalho ao projeto estudantil KRT de Fórmula SAE. Que os conhecimentos acumulados aqui possam contribuir de alguma forma com desenvolvimento desta equipe, que foi parte fundamental do meu desenvolvimento profissional durante a graduação.

AGRADECIMENTOS

- Primeiramente aos meus pais, pelo suporte e incentivo nesta fase do meu desenvolvimento profissional/acadêmico;
- Ao meu orientador, Marcus Americano, por receber minha proposta de trabalho e me orientar por todo o curso do mestrado;
- à Fundação de Amparo à Pesquisa do Estado da Bahia - FAPESB, por apoiar via bolsa o desenvolvimento deste trabalho;
- à Universidade Federal da Bahia - UFBA, por receber minha proposta de trabalho e aprovar meu ingresso no Programa de Pós-Graduação em Mecatrônica - PPGM;
- Aos professores Daniel Santana - UFBA e Gustavo Artur - UFSC, por aceitarem participar da banca de defesa;
- Aos professores do Programa de Pós-Graduação em Mecatrônica - PPGM, pelos conhecimentos transmitidos durante o curso do mestrado;
- E acima de tudo, a Deus, a quem devo minha vida e às oportunidades que pavimentaram meu caminho até este momento.

“Driving a car as fast as possible is all about maintaining the highest possible acceleration level in appropriate direction.”

—PETER G. WRIGHT (Diretor técnico da Lotus Team)

RESUMO

Neste trabalho, um novo método para a modelagem quase-estática de veículos de competição foi desenvolvido. O método utiliza apenas informações facilmente disponíveis para categorias amadoras de competição automotiva. Um problema de otimização foi formulado para determinar o traçado ótimo e um novo método de otimização foi desenvolvido para realizar a otimização do traçado. Este novo método (Método do Gradiente Duplo) permite prever linhas de corrida descritas por splines cúbicas (problemas resolvidos na maioria dos casos por métodos estocásticos) em tempo semelhante aos métodos determinísticos. Solucionar o problema de minimização do tempo de volta utilizando um método gráfico de otimização mostrou ser viável e aplicável. O método do Gradiente Duplo foi avaliado em comparação a quatro outros métodos de otimização capazes de otimizar o traçado graficamente e foi o único a executar a simulação em tempo hábil para que a análise possa ser executada durante eventos esportivos.

Palavras-chave: Simulação, Tempo de Volta, Otimização, Tempo Mínimo de Manobra

ABSTRACT

In this work, a new method for quasi-static modeling of competition vehicles was developed. The method uses only information that is readily available for amateur automotive competition categories. An optimization problem was formulated to determine the optimal trajectory and a new optimization method was developed to perform the trajectory optimization. This new method (Double Gradient Method) makes it possible to predict racing lines described by cubic splines (problems solved in most cases by stochastic methods) in a similar time as deterministic methods. Solving the minimum lap time problem using a graphical optimization method proved to be feasible and applicable. The double gradient method was evaluated against four other optimization methods capable of graphically optimizing the trajectory and was the only one that could run the simulation in a timely manner, allowing the analysis to be performed during sporting events.

Keywords: Lap Time, Simulation, Optimization, Minimum Maneuvering Time

SUMÁRIO

Capítulo 1—Introdução	1
1.1 Objetivo do trabalho	2
1.2 Objetivos específicos	2
1.3 Justificativa	3
1.4 Estrutura do trabalho	4
Capítulo 2—Problemas fundamentais em uma competição veicular	5
2.1 Aceleração máxima na direção correta	6
2.2 Traçando a Melhor Rota	7
2.3 Trabalhos anteriores	9
2.4 Conclusão do capítulo	12
Capítulo 3—Virtualização da pista e interpolação do traçado	15
3.1 Circuitos virtualizados e dados disponíveis	16
3.2 Curva de Bézier e splines cúbicas	17
3.3 Seleção do método de interpolação	21
3.3.1 Spline para virtualização do circuito	22
3.3.2 Spline para a construção do traçado	25
3.4 Conclusão do Capítulo	29
Capítulo 4—Modelagem do veículo	31
4.1 Disponibilidade de dados de pneus	32
4.2 Disponibilidade de dados de projeto	33
4.3 Modelagem com um grau de liberdade	37
4.3.1 Modelo super-elíptico de um grau de liberdade	42
4.3.2 Identificação dos parâmetros do modelo super-elíptico	45
4.4 Conclusão do Capítulo	51
Capítulo 5—Formulação do problema de otimização	53
5.1 Métodos probabilísticos	55
5.1.1 Algoritmo genético	56
5.1.2 Evolução Diferencial	56
5.2 Métodos livres de derivadas	57
5.2.1 Nelder-Mead	57

5.2.2	Região de Confiança	59
5.3	Método do Gradiente Duplo	60
5.4	Conclusão do Capítulo	64
Capítulo 6—Simulações e Resultados		67
6.1	Simulações	67
6.2	Resultados	68
6.3	Conclusão do Capítulo	74
Capítulo 7—Conclusões, contribuições e perspectivas		77
7.1	Contribuições	77
7.2	Sumário de conclusões	78
7.3	Perspectivas para trabalhos futuros	79
Apêndice A—Interpolação por curvas de Bézier		85
Apêndice B—Controle LQR + FF aplicado a modelo não linear de dinâmica lateral veicular.		89
B.1	Modelo linear	89
B.2	Modelo não-linear	91
B.3	Regulador Quadrático Ótimo	94
B.4	Implementação do LQR no modelo linear	95
B.5	Controle Feedforward	99
B.6	Implementação do LQR no modelo não-linear	101
Apêndice C—Código Python do método do gradiente duplo		105

LISTA DE FIGURAS

2.1	Representação em ponto de massa de um veículo se deslocando em uma pista (limites da pista representados pelas linhas pretas). Em vermelho está representado o traçado descrito pelo veículo e em azul o vetor velocidade. Adaptado de (MILLIKEN et al., 2003)	6
2.2	Representação em ponto de massa de um veículo se deslocando em uma pista (limites da pista representados pelas linhas pretas). Em vermelho está representado o traçado descrito pelo veículo. Os vetores azul, verde e roxo representam a aceleração longitudinal, lateral e resultante respectivamente. Adaptado de (MILLIKEN et al., 2003)	7
2.3	Representação do conceito básico de círculo de tração. Em roxo está representado o vetor de aceleração resultante e em azul e verde os vetores de aceleração longitudinal e lateral respectivamente.	8
2.4	Comparação do estilo de pilotagem de dois pilotos. a) Tempo médio no segmento de pista e velocidade mínima. b) Velocidade (piloto um). c) Velocidade (piloto dois). Fonte: (KEGELMAN; HARBOTT; GERDES, 2017)	9
3.1	Dados aquisitados em veículo de competição na pista de Goiânia.	17
3.2	Circuitos de Goiânia (a) e Curitiba (b)	17
3.3	Pontos GPS coletados para virtualização do limite interno da pista de Goiânia	18
3.4	Curva de Bézier de grau 3 via algoritmo de Casteljau. O ponto $b_{0,3}$ é obtido via sucessivas interpolações lineares entre os pontos de controle (azuis) e entre os pontos intermediários.	19
3.5	a) Polinômios de Bernstein da curva de Bézier. b) Continuidade C^0 dos polinômios de Bernstein.	20
3.6	a) Funções base da spline de Catmull-Rom. b) Continuidade C^0 das funções base.	20
3.7	a) Funções base da B-spline. b) Continuidade C^0 das funções base.	22
3.8	Curvas de Bézier, spline de Catmull-Rom e B-spline interpoladas de P_0 a P_3 (linha cheia) e de P_1 a P_4 (linha tracejada).	22
3.9	Pares de coordenadas de controle para o circuito de Goiânia.	23
3.10	Curvas de Bézier, spline de Catmull-Rom e B-Spline interpoladas de P_0 a P_3 (linha cheia) e de P_1 a P_4 (linha tracejada). [Pontos P_1 e P_2 aproximados]	24
3.11	Vetores de construção para a spline de Catmull-Rom	24

3.12	Curvas de Bézier, spline de Catmull-Rom balanceada e B-Spline interpoladas de $P0$ a $P3$ (linha cheia) e de $P1$ a $P4$ (linha tracejada).	25
3.13	a) Primeira derivada das funções base da spline de Catmull-Rom. b) Segunda derivada das funções base da spline de Catmull-Rom.	26
3.14	a) Primeira derivada das funções base da B-Spline. b) Segunda derivada das funções base da B-Spline.	27
3.15	Spline formada por curvas de Bézier de grau 3 interpolando cinco pontos de controle.	28
3.16	Plotagem da curvatura do traçado para um trecho do circuito de Goiânia.	29
4.1	Modelo de "bicicleta" (dois graus de liberdade)	31
4.2	Comparação entre atrito de Coulomb e uma curva típica de atrito em pneus. Adaptado de SEWARD (2014)	32
4.3	"Flat-track" para teste de pneus. Fonte: https://calspan.com/automotive/tire-performance-testing/motorsports-tire-testing	33
4.4	Curvas de força lateral (Cornering Force) e momento auto-alinhante (Self Aligning Torque) disponibilizados pela Avon (AVON, 2023). "Slip Angle" corresponde ao ângulo de deriva do pneu (diferença entre a direção em que a roda aponta e a direção em que se desloca).	33
4.5	Determinação da posição do centro de massa do veículo. Fonte: (MILLIKEN et al., 2003)	34
4.6	Determinação da altura do centro de gravidade de um veículo via tilt table.	35
4.7	Comparação entre a velocidade angular de um veículo com momento principal de inércia I_z e o mesmo veículo com momento principal de inércia $0.7 \cdot I_z$	37
4.8	Diagrama g-g. Em azul os dados coletados de um carro de competição no circuito de Goiânia e em laranja o círculo de tração teórico.	38
4.9	Diagrama g-g. Em azul os dados coletados de um carro de competição no circuito de Goiânia, em laranja o círculo de tração teórico e em verde a elipse de tração saturada.	39
4.10	Diagrama g-g. Em azul, os dados coletados em um carro de competição no circuito de Goiânia. Em verde, a elipse saturada de tração. Em vermelho, os quatro pontos usados para a construção da elipse.	40
4.11	Diagrama de momento de yaw de um carro de competição genérico.	41
4.12	Diagrama de momento de yaw de um carro de competição genérico em aceleração.	41
4.13	Condição de equilíbrio estático para um veículo (princípio de d'Lembert).	42
4.14	Limites de aceleração longitudinal ($\ddot{x} _{acc}(\dot{x})$ e $\ddot{x} _{fre}(\dot{x})$) e parcelas das equações.	45
4.15	Dados para determinar a curva de tração limitada pela potência.	47
4.16	Dados filtrados para determinar a curva de tração limitada pela potência.	47
4.17	Torque do motor a 100% de abertura da borboleta de aceleração. Linha vermelha destaca o torque médio.	48
4.18	Curva de tração limitada pela potência.	48

4.19	Dados para a construção do modelo super-elíptico: velocidade, acelerações lateral e longitudinal nos trechos de máxima performance.	49
4.20	Dados calculados pelo modelo super-elíptico comparados aos dados adquiridos no veículo de competição nos trechos de máxima performance. . .	51
4.21	Dados calculados pelo modelo super-elíptico comparados aos dados adquiridos no veículo de competição para volta completa (aceleração lateral espelhada).	51
5.1	Coordenadas de controle para geração do traçado para o circuito de Goiânia. Os círculos destacam os pontos de solução trivial.	54
5.2	Ciclo de um algoritmo genético. Adaptado de: FILHO; TRELEAVEN; ALIPPI (1994)	56
5.3	Funcionamento do algoritmo de Evolução Diferencial. Adaptado de MELO (2009)	57
5.4	Construção de um novo simplex. Adaptado de RAVINDRAN; RAGSD-DELL; REKLAITIS (2006)	58
5.5	Contração e expansão de um simplex. Adaptado de RAVINDRAN; RAGSD-DELL; REKLAITIS (2006)	58
5.6	Otimização por região de confiança. Adaptado de CONN (2000)	59
5.7	Topologia de uma rede neural artificial com retropropagação do erro . . .	60
5.8	Topologia básica do método do gradiente duplo	61
5.9	Atualização da posição dos pontos de controle do traçado durante o processo de otimização.	63
5.10	Função sigmoide modificada.	64
6.1	Posição do veículo ao longo da pista em relação a margem interna da pista.	69
6.2	Velocidade do veículo ao longo da pista.	69
6.3	Curvatura do traçado executado ao longo da pista.	70
6.4	Diagrama g-g simulado em comparação aos obtidos com pilotos profissionais. O modelo veicular utilizado para o simulador foi calibrado para representar o comportamento médio do veículo.	70
6.5	Curvas do circuito misto de Goiânia. Fonte: https://wlps-ge-stuff.ucoz.com/photo/autodromo_internacional_ayrton_senna_goiania/3-0-249	71
6.6	Curva Um do circuito misto de Goiânia.	71
6.7	Curva do Mergulho do circuito misto de Goiânia.	72
6.8	Curva do Miolo do circuito misto de Goiânia.	72
6.9	Curva do Bico de Pato do circuito misto de Goiânia.	73
6.10	Curva do Esse do circuito misto de Goiânia.	73
6.11	Curva Zero do circuito misto de Goiânia.	74
B.1	Modelo de "bicicleta". Fonte: RAJAMANI (2006)	89
B.2	Sistemas de coordenadas do modelo não-linear. Fonte: RAJAMANI (2006)	92
B.3	Slip Angle. Fonte: RAJAMANI (2006)	93
B.4	Diagrama do sistema controlado pela matriz K. Fonte: OGATA (2009) .	95
B.5	Modelo linear implementado em Matlab®	96

B.6	Modelo linear em função dos erros e_1 e e_2 . Fonte: RAJAMANI (2006) . . .	97
B.7	Modelo linear em função dos erros e_1 e e_2 implementado em Matlab® . . .	98
B.8	Modelo linear com controle LQR + FF implementado em Matlab® . . .	100
B.9	Modelo linear com controle LQR + FF filtrado implementado em Matlab®	101
B.10	Modelo não linear implementado em Matlab®	103
B.11	Controle LQR + FF aplicado ao modelo não linear	104

LISTA DE TABELAS

4.1	Tabela para o cálculo do momento principal de inércia em torno do eixo vertical para um Fórmula 3. Adaptado de: (NOWLAN, 2020)	36
4.2	Informações básicas do veículo	46
6.1	Tempos de volta comparados.	68

Capítulo

1

Neste capítulo serão apresentados o objetivo, justificativa e a estrutura do trabalho.

INTRODUÇÃO

Problemas de tempo mínimo de volta ou tempo mínimo de manobra são problemas de otimização em que o objetivo é determinar o traçado ótimo para que uma rota seja percorrida no menor tempo possível. Em se tratando de competições veiculares, isso implica em dizer que o objetivo de um problema de otimização como esse é determinar o traçado de corrida que retorna o menor tempo de volta numa determinada pista. Importante ressaltar que traçado diz respeito ao caminho pelo qual o veículo realmente passou, enquanto a pista ou circuito é o espaço físico no qual uma série de possíveis traçados podem ser percorridos.

O traçado do veículo pode ser mensurado de três formas. A maneira mais precisa de se mensurar o traçado é através de dados coletados via GPS, assim é possível avaliar com precisão a latitude, longitude e altitude e geolocalizar esses pontos em relação aos pontos GPS que definem o circuito onde o traçado está sendo realizado. A segunda forma de se mensurar o traçado é através de sensores inerciais (acelerômetros, por exemplo), sendo necessário tratar esses dados para aproximar o traçado realizado. Entretanto, não há como relacioná-los com as coordenadas GPS do circuito. A terceira forma é o uso de câmeras, utilizadas em eventos esportivos para avaliar o desempenho do piloto e orientá-lo de modo a melhorar seu desempenho ou em associação a visão computacional para orientação de veículos autônomos.

Retomando os problemas de tempo mínimo de volta, para solucioná-los é necessário modelar o veículo levando-se em consideração o máximo possível de dados estruturais, cinemáticos e dinâmicos. Em competições amadoras, semiprofissionais ou estudantis, nem sempre é possível ter acesso aos dados que descrevem a dinâmica dos pneus, por exemplo. Cada tipo de pneu possui uma curva de atrito específica que é avaliada em laboratório. Os dados coletados geralmente ficam restritos às empresas que desenvolvem os pneus, não sendo de acesso livre a equipes de competição, em especial equipes amadoras. Ainda que se opte pela aquisição de pneus de empresas que disponibilizam as curvas de atrito, geralmente os dados das curvas disponíveis já foram tratados e são disponibilizados com limitações. Para modelagem veicular detalhada, seria necessário a disponibilidade dos dados completos.

Com relação a dinâmica do veículo, dados de projeto também são difíceis de se obter em competições amadoras e semiprofissionais (estão disponíveis em competições estudantis em que o veículo é fabricado pelos próprios estudantes). Existem alguns procedimentos que serão melhor abordados no Capítulo 4 que permitem aproximar uma série de informações sobre o projeto do veículo que são essenciais para a modelagem computacional, mas ainda assim, essa aproximação não é totalmente precisa. É importante notar que a ferramenta desenvolvida neste trabalho visa a aplicação em eventos amadores, semiprofissionais ou estudantis, para os dois primeiros o mais usual é adaptar um veículo de rua para as pistas, logo, não há como ter acesso aos dados de projeto do veículo original.

1.1 OBJETIVO DO TRABALHO

Desenvolver um simulador capaz de solucionar o problema de tempo mínimo de volta sem recorrer às equações que descrevem a dinâmica dos pneus e às equações diferenciais que descrevem a dinâmica do veículo.

A simulação deve ser executada em tempo, dentro de uma janela de tempo que permita a aplicação dessa ferramenta em eventos esportivos. Durante uma seção de treinos, por exemplo, uma janela de tempo de poucos minutos (menos de 30 minutos preferencialmente) é o suficiente para simular e avaliar os dados para a preparação do veículo e do piloto.

A ferramenta desenvolvida neste trabalho visa a aplicação em eventos amadores, semiprofissionais ou estudantis.

1.2 OBJETIVOS ESPECÍFICOS

- Desenvolver um novo método de modelagem quase estática para representar as características dinâmicas do veículo incluindo o estilo de pilotagem do piloto.
- Realizar um estudo para verificar as características de splines cúbicas (Bézier, B-spline e Catmull-Rom) com o intuito de apontar as vantagens específicas de cada tipo de spline que justifiquem à sua adoção em pontos distintos do algoritmo a ser desenvolvido.
- Desenvolver uma metodologia para identificação de parâmetros para a modelagem quase-estática a partir dos dados de acelerômetro do veículo.
- Solucionar o problema de minimização do tempo de volta utilizando um método gráfico de otimização.
- Desenvolver um novo método de otimização gráfico específico para o problema de minimização de tempo de volta.
- Comparar a capacidade do simulador desenvolvido (modelo quase-estático associado ao otimizador) com dados reais coletados com pilotos profissionais para validar a ferramenta.

1.3 JUSTIFICATIVA

Para determinar o traçado que retorna o menor tempo de volta é necessário que o piloto execute várias voltas no circuito. Durante a execução dessas voltas o piloto vai se familiarizando com o veículo e com a pista e, em conjunto com uma equipe de engenheiros, vai melhorando as configurações do veículo e sua abordagem (a forma como conduz o veículo na pista) para atingir a melhor volta possível. Esse processo de melhoria do tempo de volta é iterativo e custoso.

Quando se trata de pilotos amadores ou semiprofissionais o problema se agrava. Pilotos profissionais são capazes de executar várias voltas consecutivas no circuito se adaptando as condições do veículo (que se deterioram naturalmente ao longo das voltas) e mantendo a consistência entre as voltas, ou seja, aproximadamente o mesmo traçado exigindo o máximo que o veículo pode oferecer. Pilotos semiprofissionais possuem alguma consistência em termos de uso da totalidade da capacidade do veículo, mas nem sempre realizam o melhor traçado, o que se reflete em tempos de volta inconsistentes. Pilotos amadores são inconsistentes, mas capazes de atingir o limite da capacidade do veículo em certos pontos do traçado.

A ferramenta a ser desenvolvida precisa ser capaz de solucionar o problema de tempo mínimo de volta requerendo o mínimo de informações sobre o veículo e o mínimo de dados sobre a condução do piloto. O primeiro é de difícil aquisição para competições amadoras, semiprofissionais e estudantis e o segundo não é útil em sua totalidade devido a inconsistência. Disso resulta que:

- O modelo computacional não pode depender das equações que representam a dinâmica do veículo, assim, um modelo quase-estático passa a ser a única opção viável;
- Para a modelagem quase-estática é necessário conhecer os limites de desempenho do veículo medindo-os em pista. Devido a inconsistência nos dados, uma metodologia precisa ser desenvolvida para separar apenas os dados úteis para a modelagem;
- Um modelo quase-estático não possui as equações necessárias para que métodos de otimização determinísticos solucionem o problema de tempo mínimo de volta, sendo necessário recorrer a métodos que não avaliem as equações do modelo.
- Sem as equações do veículo, calcular o traçado também se torna um problema, neste caso é necessário recorrer a um método gráfico para a construção de traçados: splines cúbicas.
- Uma vez que o traçado é construído graficamente, o método de otimização precisa ser capaz de manipulá-lo diretamente. Um novo método de otimização será desenvolvido especificamente para esse fim, embora outros possam realizar a tarefa. Desenvolver um método específico para essa tarefa visa reduzir o tempo necessário para realizar o processo de otimização.

1.4 ESTRUTURA DO TRABALHO

Este trabalho está organizado em capítulos. Cada capítulo visa abordar um problema específico para o desenvolvimento do simulador de tempo mínimo de volta, são eles:

- Calcular o traçado sem acessar as equações do veículo, através de representação gráfica. Será abordado no Capítulo 3.
- Desenvolver uma metodologia para tratar os dados necessários para construir um modelo quase-estático que represente o veículo. Será abordado no Capítulo 4.
- Desenvolver um método de otimização que otimize o traçado construído graficamente e que o faça mais rápido que otimizadores já disponíveis. Será abordado no Capítulo 5.

No Capítulo 2 serão apresentados alguns conceitos básicos e uma revisão bibliográfica para auxiliar no entendimento do trabalho. O Capítulo 6 apresenta os resultados das simulações realizadas pelo simulador desenvolvido enquanto o Capítulo 7 apresenta conclusões, contribuições e perspectivas para trabalhos futuros.

Neste capítulo serão apresentados alguns conceitos básicos e revisão bibliográfica

PROBLEMAS FUNDAMENTAIS EM UMA COMPETIÇÃO VEICULAR

Um carro de corrida existe unicamente para permitir que uma pessoa possa percorrer uma determinada distância em menos tempo que qualquer outra combinação de homem e máquina presente no circuito ou pista em um determinado evento. Desse modo, um carro de corrida é apenas uma ferramenta para o piloto (SMITH, 1978).

“Driving a car as fast as possible (in a race) is all about maintaining the highest possible acceleration level in appropriate direction.”

Peter G. Wright – Diretor técnico da Lotus Team

A declaração acima foi retirada do primeiro capítulo do livro Race Car Vehicle Dynamics (MILLIKEN et al., 2003). Neste capítulo os autores apresentam o problema fundamental imposto por uma corrida: acelerar o máximo possível um veículo na direção correta. Na seção 2.1 será discutido o que de fato significa “direção correta” em uma competição veicular.

Assim como uma chave de fenda é fabricada para apertar um tipo específico de parafuso, um carro de corrida, enquanto ferramenta, é preparado para obter o melhor desempenho em um determinado circuito (pista de corrida ou de teste). Para cada circuito, portanto, o veículo possui um conjunto de ajustes específico para garantir o desempenho máximo.

Uma vez que o veículo está devidamente configurado para um circuito, cabe ao piloto utilizá-lo da melhor forma, ou seja, conseguir percorrer a pista no menor tempo. Esse problema se divide em dois outros: manter o máximo de aceleração possível na direção correta, como apontou Peter G. Wright e traçar uma rota tal que minimize o tempo de volta.

Assim, ao todo, três são os problemas fundamentais em uma competição veicular: preparar o veículo para obter o máximo de desempenho, manter o veículo acelerando

ao máximo na direção correta e traçar a melhor rota para reduzir o tempo de volta. O primeiro diz respeito apenas ao conjunto mecânico do veículo, o segundo depende tanto do conjunto mecânico quanto da habilidade do piloto e o terceiro diz respeito apenas a habilidade do piloto.

2.1 ACELERAÇÃO MÁXIMA NA DIREÇÃO CORRETA

Se o objetivo é reduzir o tempo de volta de um carro em uma pista de corrida, é de se imaginar como solução trivial o aumento da velocidade tanto quanto o possível ao longo do traçado. Contudo, velocidade não é uma grandeza escalar, é um vetor, e como tal possui magnitude e direção.

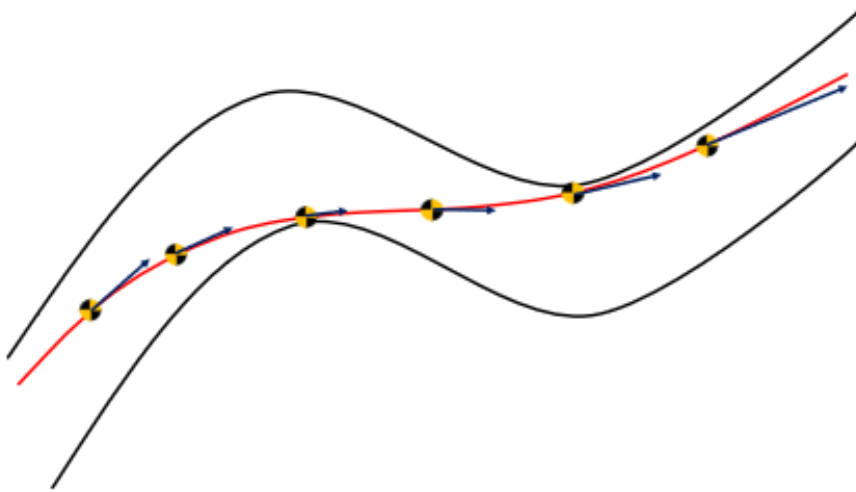


Figura 2.1: Representação em ponto de massa de um veículo se deslocando em uma pista (limites da pista representados pelas linhas pretas). Em vermelho está representado o traçado descrito pelo veículo e em azul o vetor velocidade. Adaptado de (MILLIKEN et al., 2003)

A Figura 2.1 representa um veículo se deslocando em uma pista como um ponto de massa. Em vermelho está representado o traçado, que é o caminho percorrido pelo veículo, o qual precisa ser otimizado pelo piloto para que a pista seja percorrida no menor tempo possível. Em azul está representado o vetor velocidade.

Observe que o vetor velocidade está sempre mudando conforme o veículo se desloca e sua direção é sempre tangente à trajetória percorrida. Esta mudança contínua não é acidental, a velocidade de um veículo de competição nunca deve ser constante, a menos que seja exigido por limite de potência do veículo ou outros fatores como tráfego na pista (MILLIKEN et al., 2003). Se a velocidade não deve ser constante, isso implica que o veículo estará sempre acelerando durante a volta e, portanto, sempre tentando atingir seu máximo potencial.

A Figura 2.2 representa o mesmo veículo executando a mesma manobra apresentada na Figura 2.1. Em azul está representado o vetor de aceleração longitudinal e em verde

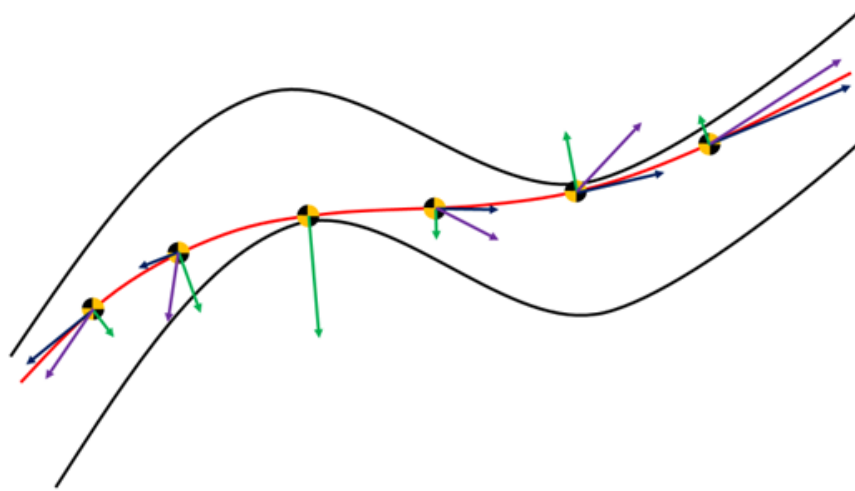


Figura 2.2: Representação em ponto de massa de um veículo se deslocando em uma pista (limites da pista representados pelas linhas pretas). Em vermelho está representado o traçado descrito pelo veículo. Os vetores azul, verde e roxo representam a aceleração longitudinal, lateral e resultante respectivamente. Adaptado de (MILLIKEN et al., 2003)

o vetor de aceleração lateral. O segundo problema fundamental de uma competição veicular, conforme estabelecido no início do capítulo, diz respeito a maximização do vetor de aceleração resultante (representado em roxo na Figura 2.2). Isso implica que o balanço entre a aceleração longitudinal e aceleração lateral precisa ser tal que ambos atinjam o máximo valor que a configuração do veículo permita para uma dada velocidade, o que leva ao conceito de círculo de tração.

A Figura 2.3 representa o conceito básico de círculo de tração. Um pneu de competição é capaz de gerar aproximadamente a mesma quantidade de atrito nos sentidos longitudinal e lateral, o que é representado pela figura de um círculo, uma vez que $a_r = \sqrt{a_x^2 + a_y^2}$, onde a_r é a capacidade de atrito total do pneu e a_x e a_y são as capacidades longitudinal e lateral respectivamente.

Observando a Figura 2.3 duas coisas ficam evidentes: primeiro, o pneu pode gerar um valor "x" de atrito longitudinal ou esse mesmo valor para atrito lateral, mas não pode gerar este valor para os dois sentidos ao mesmo tempo e segundo, se o objetivo é atingir o máximo de aceleração, então é necessário usar toda a capacidade do pneu, ou seja, manter o vetor resultante sempre no limite do diagrama (SMITH, 1978). Assim, usar totalmente a capacidade dos pneus é a premissa básica dos dois primeiros problemas fundamentais: preparar o veículo para obter o máximo de desempenho e manter o veículo acelerando ao máximo na direção correta.

2.2 TRAÇANDO A MELHOR ROTA

Na seção 2.1 foi apresentado o conceito básico de círculo de tração. Nela foi visto que os pneus possuem capacidade limitada de geração de atrito e que este atrito precisa ser

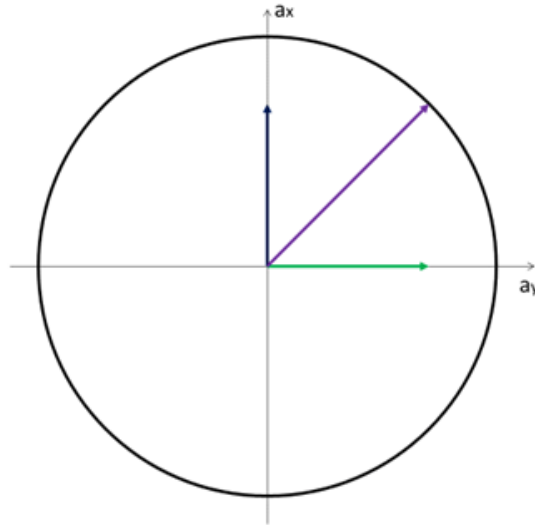


Figura 2.3: Representação do conceito básico de círculo de tração. Em roxo está representado o vetor de aceleração resultante e em azul e verde os vetores de aceleração longitudinal e lateral respectivamente.

distribuído nos sentidos longitudinal e lateral.

O terceiro problema fundamental apresentado na seção 2.1 sugere que existe um traçado no qual o tempo total de volta é mínimo. De fato, uma vez que a capacidade de geração de atrito dos pneus é limitada e precisa ser dosada entre acelerar o veículo no sentido longitudinal ou resistir a aceleração lateral, é necessário escolher um traçado adequado para otimizar essa relação de compromisso (tradeoff).

Um estudo apresentado por KEGELMAN; HARBOTT; GERDES (2017) a respeito da técnica de pilotagem de pilotos profissionais, comparou o traçado realizado por dois pilotos distintos num mesmo circuito por repetidas voltas (ver Figura 2.4). Eles observaram que os pilotos profissionais são bastante consistentes quanto ao traçado que realizam, contudo, o traçado realizado difere de um piloto para outro. A conclusão do estudo mostra que, embora as técnicas individuais dos pilotos sejam diferentes, os resultados em tempo de volta são semelhantes, o que aponta para a existência de uma família de soluções (traçados possíveis para um mesmo veículo) para o problema de tempo mínimo de volta.

São necessários anos de treino para que um piloto adquira a consistência necessária para reproduzir o mesmo traçado repetidas vezes durante uma prova. Ainda que tenha alcançado o nível profissional, um piloto investe horas de treino, seja em simuladores, seja em pista, para apreender o traçado mais adequado para um circuito. As simulações de tempo de volta são usadas no esporte a motor para reduzir o tempo necessário (e o custo, uma vez que parte do treino é feito em pista) para que o piloto aprenda o melhor traçado, também são usadas para que o processo de preparação do veículo seja mais assertivo, não necessitando de ajustes por tentativa e erro entre as sessões de treino.

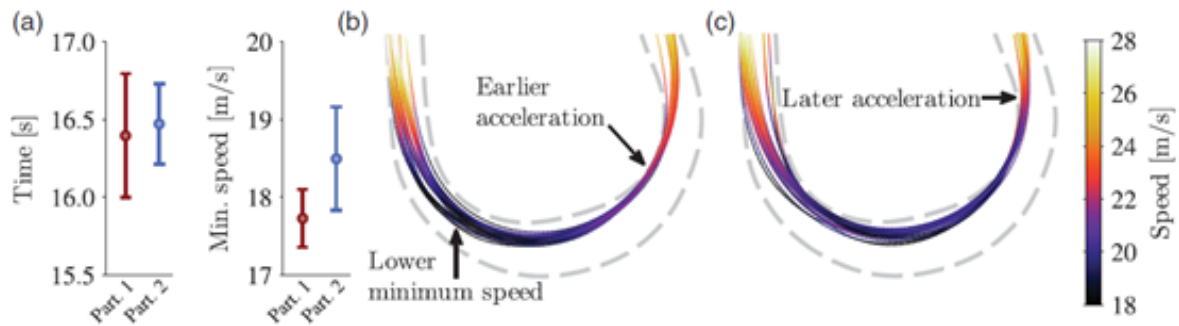


Figura 2.4: Comparação do estilo de pilotagem de dois pilotos. a) Tempo médio no segmento de pista e velocidade mínima. b) Velocidade (piloto um). c) Velocidade (piloto dois). Fonte: (KEGELMAN; HARBOTT; GERDES, 2017)

2.3 TRABALHOS ANTERIORES

Ao longo dos anos diversos autores se debruçaram sobre os problemas abordados nas seções anteriores, resultando numa série de contribuições no campo da simulação veicular.

Em 1996, GADOLA et al. utilizaram algoritmos genéticos para a predição de traçados ótimos. Para a construção do modelo de veículo usado no algoritmo, eles optaram por um modelo não linear de pneus para computar as forças laterais, um coeficiente constante para as forças longitudinais e, para computar as forças resultantes nos pneus, utilizaram a teoria da elipse de fricção. O modelo não linear de pneus requer como dado de entrada a carga vertical sobre o pneu. Para computar as cargas sobre os pneus o algoritmo utiliza um modelo linear de chassi com molas, barras anti-rolagem, rigidez vertical de pneus (representado como uma mola no modelo) e rigidez a torção do chassi. Com base nas rigidezes é possível computar a transferência de carga e, portanto, as cargas verticais nos pneus. Para a construção do traçado, a pista foi seccionada de modo a separar os trechos de curva dos trechos de reta. O processo de otimização é realizado nos trechos de curva, que são construídos através de uma spline cúbica que interpola quatro pontos de controle. O modelo do veículo é quase-estático e, portanto, não computa os efeitos transientes do sistema de amortecimento na transferência lateral de carga. Os resultados obtidos não foram melhores que os alcançados por um piloto profissional (o algoritmo foi 0,02 segundos mais lento) e a estratégia de seccionamento do traçado é um entrave para a reprodução de trechos de curvas seguidos (curvas em “s”).

Em 2000, CASANOVA desenvolve um método para usar programação não linear na predição de traçados de competição para um carro de fórmula 1, com capacidade de representar detalhadamente tanto o traçado, quanto a dinâmica do veículo. Dado que muitas informações sobre a dinâmica do veículo são tabeladas e alguns sistemas são naturalmente descontínuos (sistema de transmissão, por exemplo) o autor optou por solucionar o problema de controle ótimo (OCP na sigla em inglês) por método direto. Para converter o OCP em um problema de programação não-linear o traçado é discretizado de modo que as ações de controle aconteçam em pontos específicos do circuito. O algoritmo atua diretamente sobre o acelerador, freio e direção do veículo. O modelo do veículo não

é explicitamente vinculado ao modelo da pista, uma vez que esta foi representada em coordenadas cartesianas, assim, qualquer modelo complexo de veículo pode ser utilizado neste algoritmo. O autor optou por um modelo de sete graus de liberdade (três para a carroceria e mais quatro para os pneus). A aerodinâmica é representada simplesmente via coeficientes (arrasto e sustentação), uma vez que o modelo de sete graus de liberdade não leva em consideração os movimentos angulares da carroceria. Os pneus são completamente modelados via fórmula mágica (PACEJKA, 2006), computando o regime combinado de forças. O motor é representado via mapas, viabilizando o controle diretamente sobre o “pedal de aceleração”. Para iniciar o processo de otimização, é necessária uma referência inicial (um traçado conhecido próximo ao ideal) que será aplicado a um segundo algoritmo responsável por seguir este traçado e gerar os valores iniciais para os parâmetros de controle. O algoritmo de predição foi usado para analisar o efeito do deslocamento do centro de massa do veículo e da inércia de guinada no tempo de volta.

Em 2008, KELLY apresenta um método numérico para o controle ótimo de um carro de corrida que, simultaneamente, encontra o traçado ótimo e os sinais de controle do acelerador, direção e freio para percorrer o traçado. Assim como em CASANOVA (2000), o método consiste na solução de um problema de programação não linear. O algoritmo é capaz de simular a resposta transiente do veículo, o que permite analisar o efeito do amortecimento no sistema. Ele também desenvolve um modelo termodinâmico para os pneus, permitindo ao algoritmo calcular o efeito do aumento de temperatura dos pneus no desempenho do veículo ao longo das voltas no circuito. O método transiente desenvolvido apresentou resultados consistentes na presença de comportamentos altamente não lineares (próximos ao limite de controle e tração). O tempo total para o processo de otimização foi de aproximadamente 8 horas.

Em 2014, LOT; BIRAL desenvolvem uma metodologia para predição de traçado ótimo que utiliza as coordenadas tridimensionais da pista como referência implícita nas equações que modelam o veículo. Utilizando o triedro de Darboux, eles modelaram o circuito em coordenadas curvilíneas, que por sua vez são utilizadas para a definição das equações de movimento do veículo. Essa abordagem tem por intuito substituir os referenciais cartesianos que demandam estratégias como: tabelamento de características do circuito ponto-a-ponto (características como curvatura, rampa e inclinação) e avaliação da localização do veículo para checagem de violação de restrições locais. Uma vez que as equações de movimento do veículo são construídas a partir das coordenadas curvilíneas do traçado, as restrições locais (limites laterais da pista) estão implicitamente definidas e são localmente convexas. O autor considera que uma pista pode ser entendida como uma fita, uma vez que sua extensão é muito maior que sua largura. A linha central da pista é determinada por: $C(s) = [x(s) \ y(s) \ z(s)]^T$, onde s é a abscissa curvilínea. As equações de movimento são derivadas a partir da definição de uma tríade de coordenadas $Tc = [x_c \ y_c \ z_c]$ que se deslocam ao longo de $C(s)$. O processo de otimização do traçado é realizado via solução de um problema de controle ótimo que atua em duas variáveis de controle: impulso longitudinal e velocidade da direção. O modelo de pneus escolhido foi um modelo linear saturado e os efeitos transientes são computados via funções de transferência calibradas para emular as dinâmicas de transferência de carga com custo computacional reduzido. O método de solução utilizado foi o método indireto de Pontryagin. O método proposto

é capaz de otimizar traçados em menos de um minuto, contudo, necessita da formulação das equações a nível simbólico para derivar o modelo que será inserido no solver.

Em 2015, KOUTRIK, assim como CASANOVA (2000) e KELLY (2008), desenvolve um método para predição de traçado ótimo onde um problema de controle ótimo é convertido em um problema de programação não linear. A conversão foi realizada via método de colocação e o problema solucionado por programação quadrática sequencial. O objetivo é obter os sinais de controle ótimo do acelerador, freio e direção e utilizá-los no estudo de controle de tração para veículos de competição. Diferente de CASANOVA (2000) e KELLY (2008), o autor optou por negligenciar a dinâmica das rodas, isso reduziu os graus de liberdade do modelo do veículo, mas exigiu um conjunto de restrições para o problema que excluiu a dinâmica instável dos pneus do espaço de soluções. O circuito é modelado em coordenadas curvilíneas e as equações que modelam o veículo são derivadas em função destas coordenadas (ao invés de coordenadas cartesianas e ângulos de Euler).

Em 2018, BIANCO; LOT; GADOLA utilizaram um problema de controle ótimo para solução de tempo mínimo de volta com o objetivo de representar com mais qualidade a dinâmica de veículos altamente dependentes de dispositivos aerodinâmicos, neste caso, um carro de GP2. Eles observaram que a abordagem usual de construir modelos dinâmicos de 7 graus de liberdade (4 graus para as inércias rotativas das rodas e três para as inércias da carroceria) associado a um modelo quase-estático para computar os efeitos de transferência de carga na suspensão, não é suficiente para representar veículos com muitos dispositivos aerodinâmicos, como um fórmula. As cargas verticais aplicadas por asas são dependentes do ângulo de incidência do ar, que por sua vez é dependente das alturas de roll center do veículo. Para representar melhor veículos assim o sistema de suspensão requer uma modelagem completa resultando em 14 graus de liberdade.

Em 2019, KAPANIA; SUBOSITS; GERDES propuseram um algoritmo de dois estágios para geração de traçados com baixa demanda computacional e de rápido processamento. Para reduzir o custo computacional eles substituíram um problema de controle ótimo por dois subproblemas sequenciais: computar a velocidade em função do traçado fixo e otimizar o traçado em função do perfil de velocidade. O modelo do veículo foi derivado considerando as coordenadas curvilíneas do traçado, dessa forma o perfil de velocidade é integrado em função da curvatura do traçado. Para otimização do traçado, um problema de otimização foi proposto considerando a curvatura do traçado como uma variável explícita no modelo, garantido assim a convexidade do problema. Os resultados obtidos confirmaram a validade do algoritmo e o tempo de processamento (apenas 27 segundos) abre a possibilidade de implementação como preditor em tempo real para veículos autônomos.

Em 2020, HERRMANN et al. formularam um problema não linear com restrições de igualdade e desigualdade aplicado a um problema de minimização do tempo de volta para um veículo elétrico autônomo. Diferente de veículos a combustão, o sistema de força de um veículo elétrico é sensivelmente afetado pela temperatura de seus componentes, dessa forma, os autores se preocuparam em prever estratégias distintas para a predição do traçado, levando em consideração na construção do modelo do veículo a termodinâmica dos componentes do sistema de força.

Em 2020, JAIN; MORARI propuseram o uso de otimização Bayesiana para solucionar

problemas de minimização do tempo de volta. O algoritmo proposto utiliza uma série de pontos de controle distribuídos ao longo de um circuito e interpolados por uma spline cúbica como vetor de controle. O modelo do veículo possui apenas um grau de liberdade e é baseado no conceito de círculo de tração. O algoritmo foi usado para prever traçados de veículos elétricos autônomos em escalas 1/43 e 1/10.

Em 2020, LENZO; ROSSI propuseram um modelo veicular quase-estático de um grau de liberdade com base no conceito de círculo de tração, que, no caso do trabalho dos autores, foi estendido para uma elipse de tração. Os autores adicionaram ao conceito tradicional de círculo de tração os limites em função das capacidades aerodinâmicas do veículo bem como limites distintos para as capacidades de aceleração, lateral e frenagem. O modelo proposto é unidimensional e foi aplicado para computar a velocidade do veículo em uma trajetória conhecida. A trajetória foi representada em coordenadas curvilíneas o que permitiu a vinculação das características do traçado (curvatura e comprimento) às equações do veículo, desta forma o algoritmo proposto integra a velocidade do veículo diretamente a partir das informações do traçado, simplificando o processo de cálculo do perfil de velocidades.

Em 2021, LOVATO; MASSARO propuseram um modelo quase-estático para predição de tempo mínimo de volta em um circuito tridimensional. O traçado sendo representado em coordenadas curvilíneas permite o uso do triedro de Darboux como base para derivar as equações de movimento do veículo. O tradicional círculo de tração (ou curva g-g) foi estendido (curva g-g-g) para computar os efeitos de rampa e descida e suas variações, uma vez que o traçado é tridimensional. Uma vez calculada a curva g-g-g, esta é interpolada utilizando o algoritmo modificado de Akima, para garantir uma superfície continuamente diferenciável. Para predição do traçado, um problema de controle ótimo é solucionado utilizando como variáveis de controle as acelerações longitudinal, lateral e vertical. As restrições do problema são fornecidas pelo diagrama g-g-g e pelos limites laterais da pista.

Em 2022, GARLICK; BRADLEY propuseram uma rede neural artificial para predição de traçado em tempo real para aplicação em competições de veículos autônomos. Eles observaram que métodos de predição de traçado como os apresentados por JAIN; MORARI (2020) e KAPANIA; SUBOSITS; GERDES (2019) careciam de velocidade de processamento e capacidade de representação acurada da trajetória do veículo respectivamente. Para treinar a rede neural eles construíram um dataset com mais de 6000 circuitos distintos solucionados via problema de controle ótimo. A predição de um novo traçado para um circuito não visto pela rede neural durante o processo de treino levou menos de 33 milissegundos e não apresentou grandes desvios em relação ao traçado predito via problema de controle ótimo.

2.4 CONCLUSÃO DO CAPÍTULO

Neste capítulo foram apresentados alguns conceitos necessários para o entendimento do processo de construção do traçado de corrida. Ao compreender as limitações do veículo quanto sua capacidade de lidar com acelerações laterais e longitudinais, pilotos são capazes de determinar a trajetória que extraia em pista o melhor desempenho possível e retorne o menor tempo de volta.

Neste trabalho, o objetivo é desenvolver um simulador capaz de solucionar o problema de tempo mínimo de volta, ou seja, fazer com que um computador realize em menos tempo o que pilotos profissionais necessitam de horas de treinamento para realizar em pista. Sendo assim, foram pesquisados na literatura trabalhos com objetivo semelhante. Ao analisar os trabalhos apresentados neste capítulo, nota-se que os simuladores receberam muita atenção em aplicações profissionais de alto nível, onde dados e sensores estão amplamente disponíveis. Também é possível notar uma mudança de foco nos últimos anos, onde simuladores de tempo mínimo de volta altamente detalhados deram lugar a soluções mais simplificadas com objetivo de serem implementadas em dispositivos embarcados em veículos elétricos e/ou autônomos.

Neste trabalho, o objetivo básico - desenvolver um simulador capaz de solucionar o problema de tempo mínimo de volta - será expandido de modo a contemplar aplicações semi-profissionais ou amadoras. Para este tipo de aplicação, dados e sensores estão pouco disponíveis, o que limita a complexidade do modelo a ser construído para representar a dinâmica do veículo em pista. A simplificação compulsória do modelo para atender aplicações semi-profissionais ou amadoras, aproxima, conseqüentemente, a modelagem desenvolvida neste trabalho das modelagens que veem sendo desenvolvidas nos últimos anos para aplicação em veículos autônomos, embora não seja o foco deste trabalho.

O problema da disponibilidade de dados será melhor abordado no Capítulo 4, que tem por foco a modelagem do veículo e, por tanto, contextualiza melhor a relação entre dados disponíveis e modelagem veicular. Entretanto, antes de se pensar em modelagem veicular é necessário virtualizar o circuito onde o veículo será simulado. Como foi visto neste capítulo, a forma como o circuito é representado pode ter influência na modelagem do veículo (KOUTRIK, 2015) (LOVATO; MASSARO, 2021), sendo assim, o capítulo 3 se dedica à virtualização do circuito e desenvolvimento das equações que definem o traçado percorrido.

Neste capítulo serão apresentados os métodos de interpolação usados para virtualizar a pista e descrever o traçado do veículo

VIRTUALIZAÇÃO DA PISTA E INTERPOLAÇÃO DO TRAÇADO

Simulações de tempo de volta – Lap Time Simulation (LTS) – como referido por KAPANIA; SUBOSITS; GERDES (2019), GADOLA et al. (1996) e KELLY (2008), ou mínimo tempo de volta – Minimum Lap Time (MLT) – como referido por BIANCO et al. (2019) e LOVATO; MASSARO (2021) ou ainda mínimo tempo de manobra – Minimum Time Maneuvering (MTM) – como referido por CASANOVA (2000) e KOUTRIK (2015), têm por objetivo principal determinar o melhor traçado para um veículo percorrer um circuito de prova (pista de corrida ou de teste) no menor tempo possível.

Para que o simulador possa determinar o melhor traçado, é preciso que o circuito a ser percorrido esteja definido e, para isso, existem algumas abordagens possíveis:

- Construir o circuito em coordenadas cartesianas, separá-lo em seções e utilizar a largura da pista como restrições para o problema de otimização de trajetória livre tomando a aceleração, frenagem e direção como variáveis de controle (CASANOVA, 2000)
- Construir o circuito em coordenadas cartesianas e utilizar a largura da pista como restrições para o problema de otimização interpolando pontos de controle dentro dos limites da pista via spline cúbica (GADOLA et al., 1996) (JAIN; MORARI, 2020) (GARLICK; BRADLEY, 2022).
- Construir o circuito em coordenadas curvilíneas tomando a linha central da pista como referência e utilizando uma variável de erro em relação a linha central como variável de controle, além da aceleração, frenagem e direção (LOT; BIRAL, 2014) (KOUTRIK, 2015) (KAPANIA; SUBOSITS; GERDES, 2019).

Descrever o traçado em coordenadas curvilíneas é conveniente quando se tem acesso aos dados do pneu e aos dados de projeto do veículo. Com estes dados é possível modelar completamente sua dinâmica e o traçado curvilíneo pode ser inserido nas equações derivando-as a partir de uma tríade de coordenadas (triedro de Darboux) que se desloca ao longo do traçado junto com o veículo (LOT; BIRAL, 2014).

A descrição cartesiana do circuito para otimização de livre trajetória – onde a aceleração, frenagem e ângulo de direção são usadas como variável de controle – não é conveniente e, dentre as referências pesquisadas, não é usado desde 2000 com Casanova. Contudo utilizá-la em conjunto com interpolação por splines cúbicas pode ser vantajoso para métodos de otimização heurísticos (GADOLA et al., 1996) (JAIN; MORARI, 2020) ou para aplicação de inteligência artificial, como redes neurais (GARLICK; BRADLEY, 2022).

Um dos intuitos deste trabalho é desenvolver uma metodologia para simulação de veículos de competição que permita representar um veículo com o mínimo de dados possível, uma vez que categorias amadoras nacionais têm dificuldade de acesso a esses dados. A estratégia de virtualização da pista para este trabalho foi a descrição em coordenadas cartesianas com a construção do traçado por splines cúbicas. Descrever o traçado via splines cúbicas permite substituir o problema de controle ótimo – dependente das variáveis aceleração, frenagem e ângulo de direção e, portanto, da modelagem do veículo com no mínimo dois graus de liberdade – por um problema de otimização em que as coordenadas de controle são manipuladas diretamente e o perfil de velocidade é integrado com base na curvatura da spline, requerindo o uso de um modelo de veículo de apenas um grau de liberdade.

3.1 CIRCUITOS VIRTUALIZADOS E DADOS DISPONÍVEIS

Neste trabalho foram utilizados dados coletados via datalogger em veículos de competição. Ao todo estão disponíveis para análise os dados de pista de três pilotos profissionais obtidos em dois circuitos distintos, totalizando cinco conjuntos de dados disponíveis. Os veículos e a competição não podem ser divulgados por questão de confidencialidade.

Os dados utilizados para este trabalho foram: coordenadas GPS (longitude e latitude), aceleração longitudinal, aceleração lateral, posição da borboleta de aceleração, rotação do motor, marcha engatada e velocidade. Estes dados estão disponíveis para todas as categorias profissionais e amadoras e sua aquisição é permitida em todos os regulamentos. A Figura 3.1 apresenta como exemplo os dados de um dos pilotos em sua melhor volta no circuito de Goiânia.

Foram coletados dados em dois circuitos: Autódromo Internacional Ayrton Senna (Goiânia) e Autódromo Internacional de Curitiba (ver Figura 3.2). Estes circuitos não possuem disponíveis para o público os dados de GPS da pista. Para conseguir esses dados foi necessário converter as coordenadas GPS obtidas através do *Google Earth*[®] em coordenadas UTM, estas, por sua vez, podem ser convertidas em uma escala métrica conveniente ou usadas diretamente, uma vez que são naturalmente planejadas.

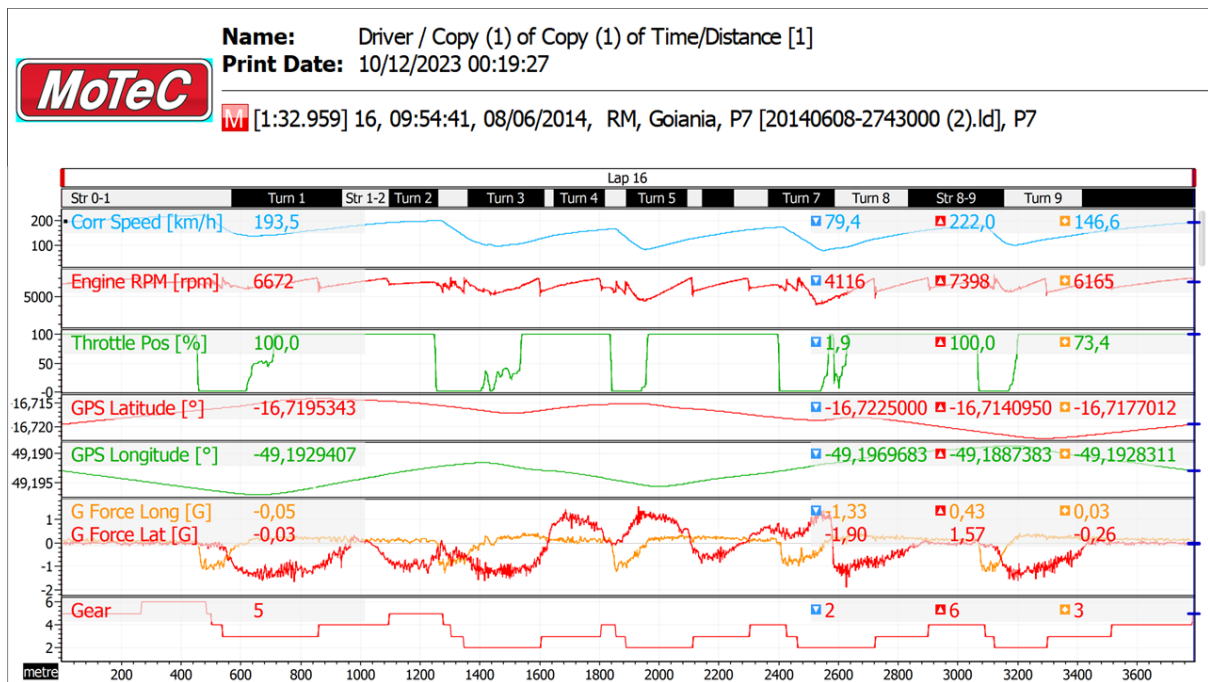


Figura 3.1: Dados aquistados em veículo de competição na pista de Goiânia.

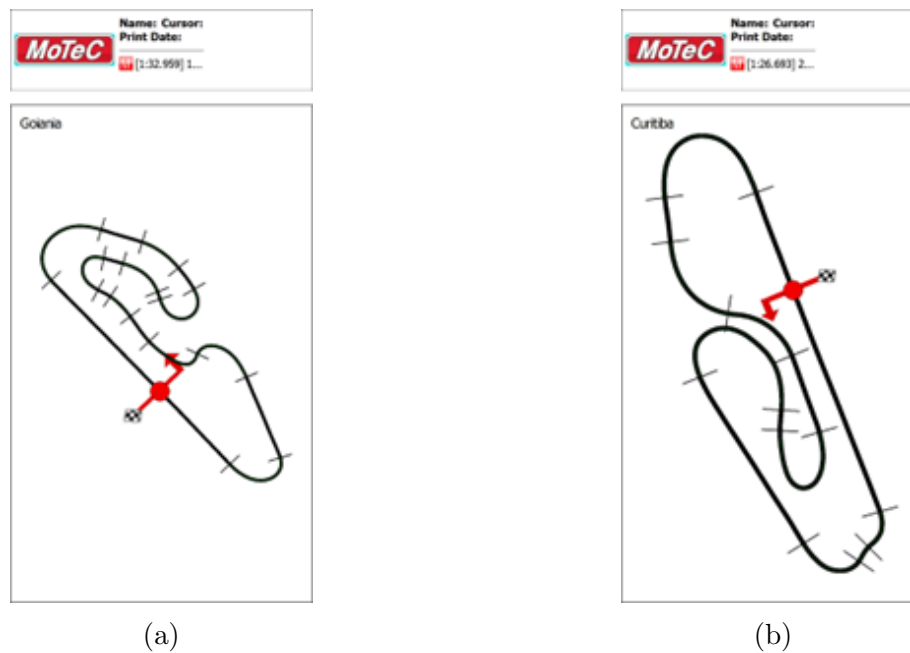


Figura 3.2: Circuitos de Goiânia (a) e Curitiba (b)

3.2 CURVA DE BÉZIER E SPLINES CÚBICAS

A Figura 3.3 apresenta o recorte de um trecho do circuito de Goiânia onde as coordenadas GPS do limite interno da pista foram coletadas via *Google Earth*[®]. É possível

observar que a distribuição de pontos coletados não é homogênea ao longo do circuito – o que seria inviável de realizar manualmente – assim, além da representação do traçado, a virtualização do circuito depende da interpolação dos pontos coletados para que as coordenadas sejam distribuídas homogeneamente.



Figura 3.3: Pontos GPS coletados para virtualização do limite interno da pista de Goiânia

O conceito mais fundamental de interpolação se refere a interpolação polinomial, sendo os primeiros métodos de interpolação deste tipo atribuídos a Newton. Embora conceitualmente simples, este tipo de interpolação apresenta um comportamento oscilatório que inviabiliza sua aplicação prática na engenharia, seja na construção de formas em CAD, seja na descrição de trajetórias via coordenadas de controle (FARIN, 2014). Curvas de Bézier e splines cúbicas são mais flexíveis e não apresentam este tipo de comportamento, sendo, neste caso, preferíveis (GADOLA et al., 1996) (ALMEIDA, 2015).

As curvas de Bézier podem ser definidas através de algoritmos recursivos via algoritmo de Casteljau (como foram originalmente desenvolvidas, ver Figura 3.4) ou de forma explícita via polinômios de Bernstein (FARIN, 2014). Utilizando os polinômios de Bernstein, temos:

$$Bz^n(t) = \sum_{j=0}^n b_j \cdot B_j^n(t) \quad (3.1)$$

Onde $Bz^n(t)$ corresponde a uma curva de Bézier de grau n , b_j corresponde aos pontos de Bézier no algoritmo de Casteljau (os pontos de controle) e $B_j^n(t)$ corresponde aos polinômios de Bernstein de grau n em função do parâmetro de interpolação t .

Reorganizando a Equação 3.1, obtém-se a forma matricial da curva de Bézier:

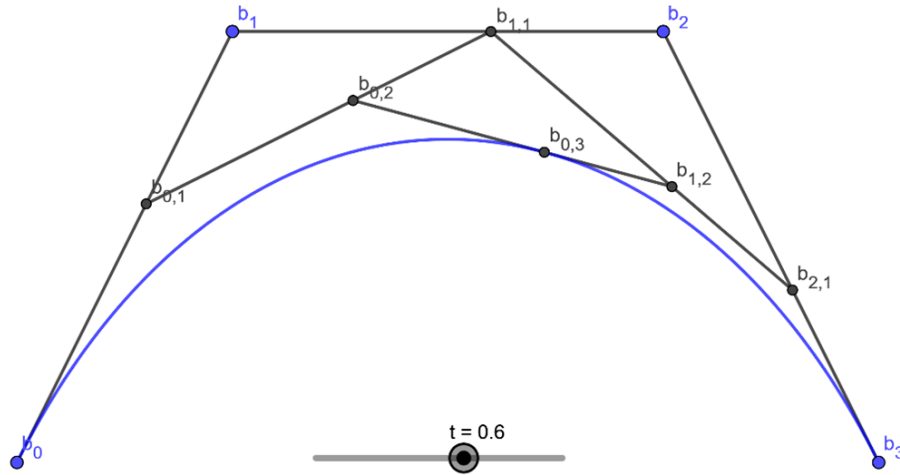


Figura 3.4: Curva de Bézier de grau 3 via algoritmo de Casteljau. O ponto $b_{0,3}$ é obtido via sucessivas interpolações lineares entre os pontos de controle (azuis) e entre os pontos intermediários.

$$Bz^3(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \quad (3.2)$$

Observe que para a construção de uma curva de grau 3 são necessários quatro pontos de controle, assim curvas de Bézier são formadas por $n + 1$ pontos. Para interpolar as coordenadas GPS do circuito ou o traçado do veículo será necessário interpolar dezenas ou centenas de pontos. Plotando-se os polinômios de Bernstein para a curva de Bézier é fácil observar que ela não possui continuidade C^0 (Figura 3.5), dessa forma, não é possível interpolar uma sequência de j pontos P na forma P_{n+m} , $n = 0, \dots, j - 4$, $m = 0, \dots, 3$, com continuidade entre as curvas, como fica evidente na Figura 3.8.

Para interpolar uma sequência de j pontos P na forma P_{n+m} , $n = 0, \dots, j - 4$, $m = 0, \dots, 3$ podemos recorrer ao uso de splines cúbicas. Dentre as splines desta família, as de Catmull-Rom (CATMULL; ROM, 1974) e B-spline interpolam diretamente os pontos de dados. Outras splines cúbicas como as Hermite por exemplo, interpolam em função de pontos de controle e suas derivadas (FARIN, 2014), não sendo convenientes para aplicação neste trabalho.

As splines de Catmull-Rom são construídas de modo que uma direção tangente L_i no ponto P_i seja paralela ao segmento $\overline{P_{i-1}P_{i+1}}$. Uma vez que esta direção é determinada, os pontos de Bézier são alocados em L_i (WATT; MARK, 1992). Este tipo de spline interpola apenas os pontos de controle intermediários, sendo contínuas em C^0 (ver Figura 3.6) e permitindo uma sequência de interpolação de j pontos P na forma P_{n+m} , $n =$

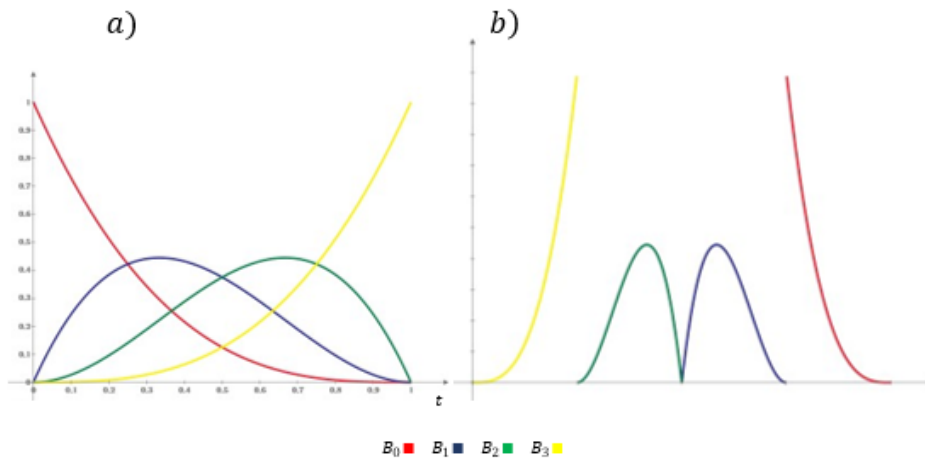


Figura 3.5: a) Polinômios de Bernstein da curva de Bézier. b) Continuidade C^0 dos polinômios de Bernstein.

$0, \dots, j-4, \quad m = 0, \dots, 3$ (ver Figura 3.8). Sua representação matricial é dada pela Equação 3.3.

$$CR_j(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \cdot \frac{1}{2} \cdot \begin{bmatrix} 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & -5 & 4 & -1 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} P_j \\ P_{j+1} \\ P_{j+2} \\ P_{j+3} \end{bmatrix} \quad (3.3)$$

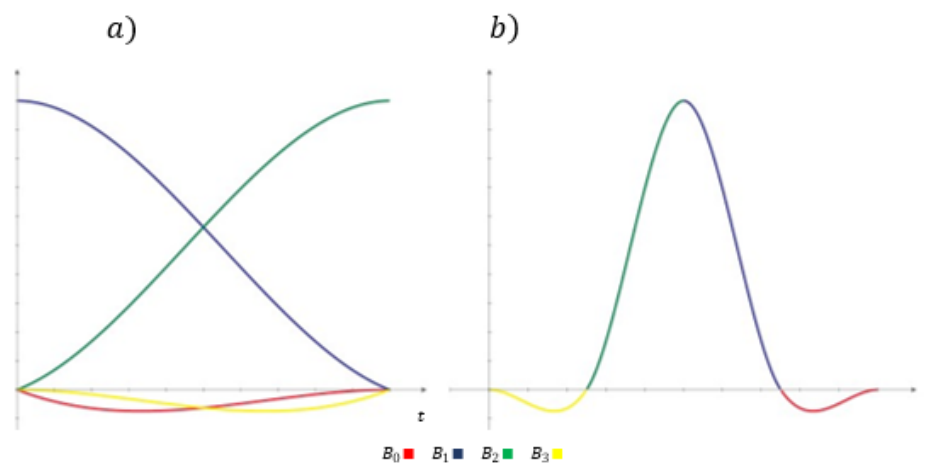


Figura 3.6: a) Funções base da spline de Catmull-Rom. b) Continuidade C^0 das funções base.

Seguintos B-spline são derivados aplicando-se uma série de restrições às funções base de modo a garantir continuidade em C^0 (ver Figura 3.7), C^1 e C^2 (WATT; MARK, 1992). Defina-se uma B-spline conforme a equação:

$$BS_j(t) = \sum_{m=0}^3 P_{n+m} \cdot B_m(t) \quad (3.4)$$

Onde $B_m(t)$ são as funções base da B-Spline. Sabendo que os pontos de controle de uma spline podem assumir valores arbitrários (WATT; MARK, 1992), as condições de continuidade da B-Spline são atingidas tomando-se as seguintes restrições:

$$\begin{array}{lll} B_0(1) = 0 & \dot{B}_0(1) = 0 & \ddot{B}_0(1) = 0 \\ B_1(1) = B_0(0) & \dot{B}_1(1) = \dot{B}_0(0) & \ddot{B}_1(1) = \ddot{B}_0(0) \\ B_2(1) = B_1(0) & \dot{B}_2(1) = \dot{B}_1(0) & \ddot{B}_2(1) = \ddot{B}_1(0) \\ B_3(1) = B_2(0) & \dot{B}_3(1) = \dot{B}_2(0) & \ddot{B}_3(1) = \ddot{B}_2(0) \\ 0 = B_3(0) & 0 = \dot{B}_3(0) & 0 = \ddot{B}_3(0) \end{array}$$

Foram assumidas 15 restrições, contudo $B_m(t)$ são funções cúbicas e, portanto, gera um sistema de equações com 16 incógnitas. A última equação para solucionar o sistema pode ser determinada assumindo a propriedade de envoltória convexa. Considerando a somatória das funções base igual a um em um ponto arbitrário:

$$B_0(0) + B_1(0) + B_2(0) + B_3(0) = 1$$

Podemos solucionar o sistema de equações e obter a função da B-spline em sua forma matricial:

$$BS_j(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \cdot \frac{1}{6} \cdot \begin{bmatrix} 1 & 4 & 1 & 0 \\ -3 & 0 & 3 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} P_j \\ P_{j+1} \\ P_{j+2} \\ P_{j+3} \end{bmatrix} \quad (3.5)$$

A Figura 3.8 traz a comparação entre as Curvas de Bézier, spline de Catmul-Rom e B-Spline interpolando cinco pontos na forma P_{n+m} , $n = 0, \dots, 1$, $m = 0, \dots, 3$. Observe que apenas as splines possuem continuidade em C_0 . Para interpolar com Curvas de Bézier seria necessário fazê-lo na forma P_{n+m} , $n = 0, 4, j$, $m = 0, \dots, 3$.

3.3 SELEÇÃO DO MÉTODO DE INTERPOLAÇÃO

A seleção do método de interpolação se baseou em três critérios: praticidade, carga computacional e curvatura. O processo de interpolação acontece em duas etapas distintas neste trabalho: na virtualização do circuito e na descrição do traçado. Os critérios de seleção do método de interpolação apresentam pesos distintos para cada uma das etapas.

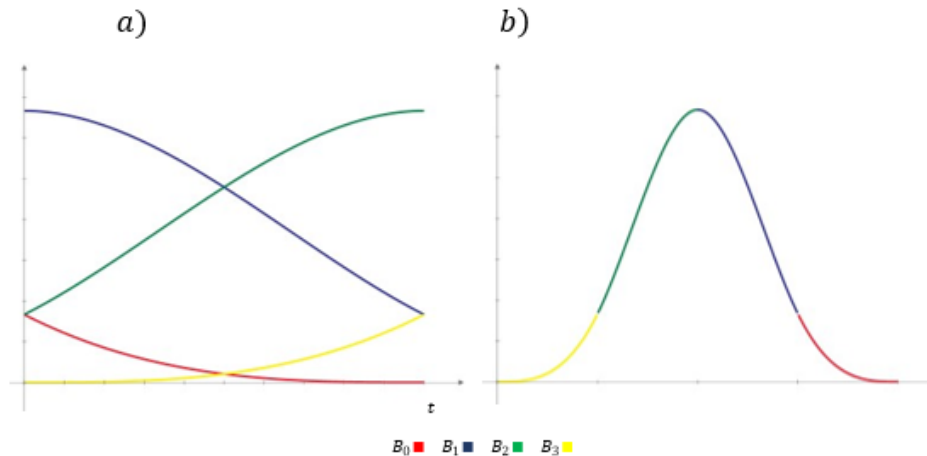


Figura 3.7: a) Funções base da B-spline. b) Continuidade C^0 das funções base.

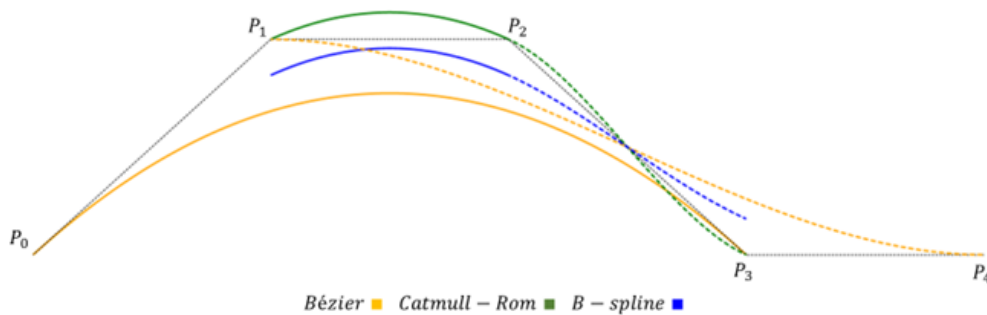


Figura 3.8: Curvas de Bézier, spline de Catmull-Rom e B-spline interpoladas de P_0 a P_3 (linha cheia) e de P_1 a P_4 (linha tracejada).

3.3.1 Spline para virtualização do circuito

O processo de interpolação para virtualização do circuito tem por objetivo fornecer uma malha de pontos adequada para a determinação das coordenadas de controle (ver Figura 3.9) que serão usadas no problema de otimização do traçado. Como foi apresentado na introdução deste capítulo, o processo de coleta dos pontos de dados GPS do circuito é realizado manualmente, sendo impraticável construir uma malha adequada para a determinação das coordenadas de controle, daí a importância do processo de interpolação.

A curva de Bézier, como apresentada na Seção 3.2, não possui continuidade em C^0 e foi, a princípio, descartada. Serão analisadas as duas splines apresentadas com base nos critérios estabelecidos.

Observando a Figura 3.8 uma característica importante nas splines fica evidente: a capacidade de interpolação. Somente a spline de Catmull-Rom tem capacidade de interpolar os pontos de controle com exceção do primeiro e do último, enquanto a B-Spline apenas aproxima estes pontos. Do ponto de vista de praticidade, a spline de Catmull-Rom se destaca, uma vez que, para a virtualização do circuito, a curvatura da spline não oferece nenhum tipo de empecilho, o único objetivo desta interpolação é a construção de

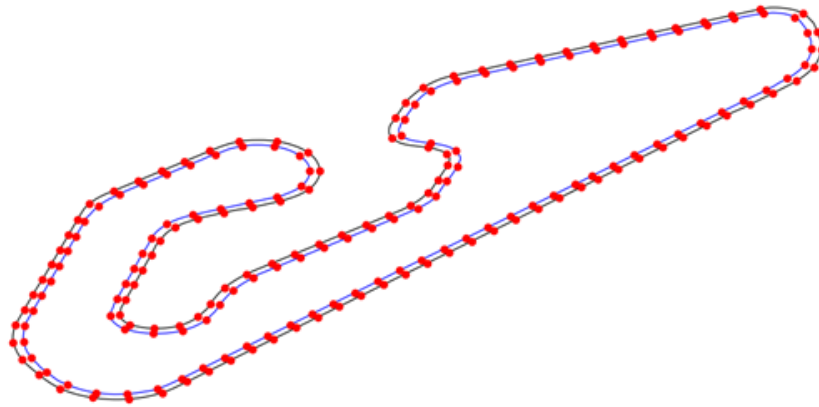


Figura 3.9: Pares de coordenadas de controle para o circuito de Goiânia.

uma malha apropriada e, sendo assim, as características de curvatura da B-Spline não compensam sua incapacidade de interpolar os pontos de controle.

As coordenadas de controle para o problema de otimização precisam ser distribuídas de modo que a spline de interpolação do traçado tenha flexibilidade o bastante para moldar o traçado, mas não necessite de um número elevado de pontos de interpolação, uma vez que o tamanho do vetor das variáveis de decisão afeta o desempenho do problema de otimização. Com base nisso, uma malha de interpolação apropriada possui pontos distribuídos em intervalos de centímetros (cerca de 50cm no máximo, como foi verificado empiricamente) o que confere flexibilidade para alocação das coordenadas de controle em pontos específicos do circuito.

Aqui vale uma breve distinção de termos para o melhor entendimento:

- Pontos de controle são os pontos usados para a construção da spline, eles podem ou não coincidir com as coordenadas a serem interpoladas.
- Coordenadas de controle são pares de coordenadas localizadas nos limites interno e externo do circuito (ver 3.9) e são usadas para gerar os pontos de interpolação do traçado no problema de otimização. Estes, por sua vez, são gerados via interpolação linear das coordenadas de controle.
- O vetor de variáveis de decisão para o problema de otimização é composto pelos parâmetros de interpolação linear que vão originar os pontos de interpolação do traçado.

Retomando a análise das splines, a de Catmull-Rom se destacou como a mais prática para a virtualização do traçado, contudo sua aplicação direta não é viável. O processo manual de determinação das coordenadas GPS que serão usadas como pontos de interpolação para a virtualização do circuito, gera uma distribuição irregular de pontos (ver Figura 3.3). Disso resulta que: é possível estabelecer uma distribuição de pontos tal que a spline de Catmull-Rom intercepte a si mesma (ver Figura 3.10).

Esta característica da spline de Catmull-Rom, a princípio, parece oferecer à B-Spline uma nova chance, bastaria solucionar o problema de interpolação dos pontos. Contudo,

uma solução mais simples torna a spline de Catmull-Rom ideal para a virtualização do traçado.

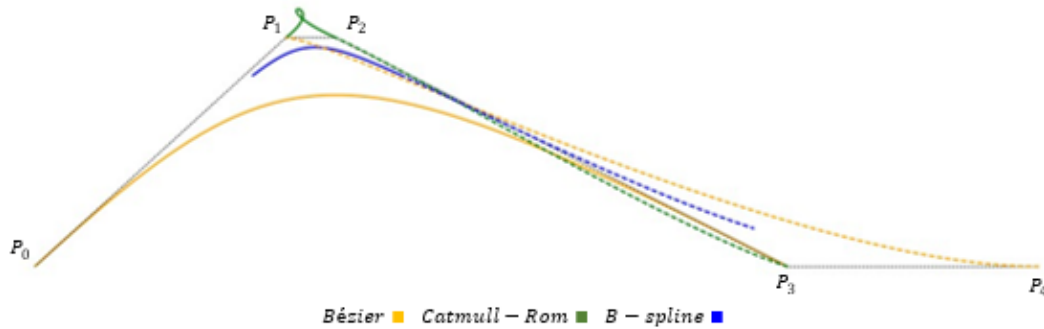


Figura 3.10: Curvas de Bézier, spline de Catmull-Rom e B-Spline interpoladas de P_0 a P_3 (linha cheia) e de P_1 a P_4 (linha tracejada). [Pontos P_1 e P_2 aproximados]

Retomando o processo de derivação da spline de Catmull-Rom, ela é construída de modo que uma direção tangente L_i no ponto P_i seja paralela ao segmento $\overline{P_{i-1}P_{i+1}}$. A Figura 3.11 apresenta os vetores v_1 e v_2 alocados nas direções L_1 e L_2 . O comprimento desses vetores afeta o comportamento da spline. Sabendo que estes vetores são função de $\overline{P_{i-1}P_{i+1}}$ será proposto um ajuste na Equação 3.3 para solucionar o problema de auto interceptação da spline.

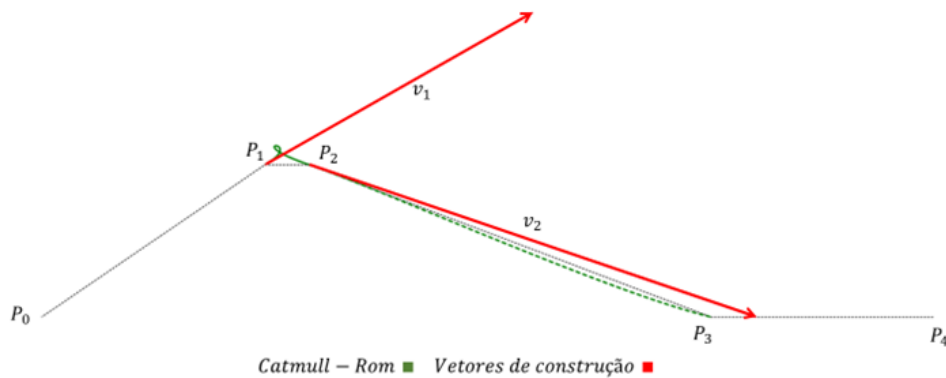


Figura 3.11: Vetores de construção para a spline de Catmull-Rom

O ajuste proposto tem por objetivo balancear a spline de Catmull-Rom para que a influência dos vetores v_1 e v_2 não seja superior a influência do segmento $\overline{P_1P_2}$, sendo assim serão considerados para a construção da spline pontos virtuais nos segmentos $\overline{P_jP_{j+1}}$ e $\overline{P_{j+2}P_{j+3}}$ que os torne equivalentes a $\overline{P_{j+1}P_{j+2}}$:

$$CR_j(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \cdot \frac{1}{2} \cdot \begin{bmatrix} 0 & 2 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 2 & -5 & 4 & -1 \\ -1 & 3 & -3 & 1 \end{bmatrix} \cdot \begin{bmatrix} \left(1 - \frac{|P_{j+2}-P_{j+1}|}{|P_{j+1}-P_j|}\right) (P_{j+1} - P_j) + P_j \\ P_{j+1} \\ P_{j+2} \\ \frac{|P_{j+2}-P_{j+1}|}{|P_{j+3}-P_{j+2}|} (P_{j+3} - P_{j+2}) + P_{j+2} \end{bmatrix} \quad (3.6)$$

A Figura 3.12 apresenta o resultado do balanceamento proposto na Equação 3.6.

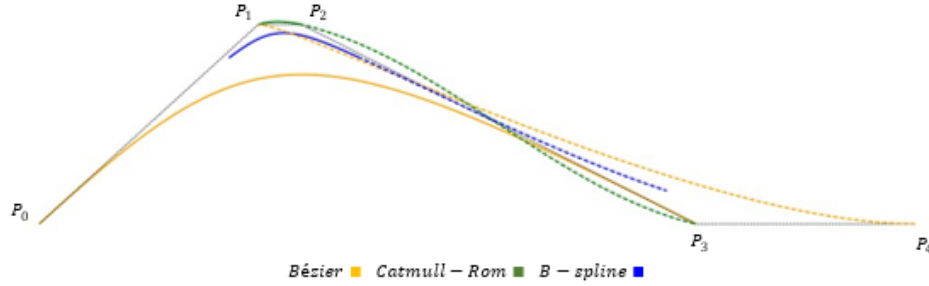


Figura 3.12: Curvas de Bézier, spline de Catmull-Rom balanceada e B-Spline interpoladas de P_0 a P_3 (linha cheia) e de P_1 a P_4 (linha tracejada).

3.3.2 Spline para a construção do traçado

O modelo de dinâmica veicular utilizado neste trabalho será derivado em função da curvatura do traçado, isso impõe à spline responsável pela interpolação dos pontos de controle uma característica imprescindível: curvatura contínua. A continuidade da curvatura é essencial para evitar mudanças de comportamento instantâneas na dinâmica veicular, se impondo como fator de decisão mais importante frente à praticidade e carga computacional.

A curvatura de uma curva paramétrica é dada pela seguinte equação:

$$\kappa(t) = \frac{\|\dot{x} \times \ddot{x}\|}{\|\dot{x}\|^3} \quad (3.7)$$

Onde x é uma curva paramétrica qualquer.

Observe que a curvatura é função tanto da primeira derivada da curva quanto da segunda. Sendo assim, a curvatura contínua da spline de interpolação do traçado depende tanto da continuidade em C^1 quanto em C^2 .

A curva de Bézier, como apresentada na Seção 3.2, não possui continuidade em C^0 e foi descartada a princípio, tal qual na Seção 3.3.1.

Derivando-se a Equação 3.6 duas vezes podemos analisar a continuidade C^1 e C^2 da spline de Catmull-Rom. A Figura 3.13 apresenta as derivadas das funções base e mostra que apenas a continuidade C^1 é atingida. Não havendo continuidade em C^2 não é possível haver continuidade de curvatura – como fica evidente na Figura 3.16 – e, portanto, interpolar o traçado via Catmull-Rom é inviável.

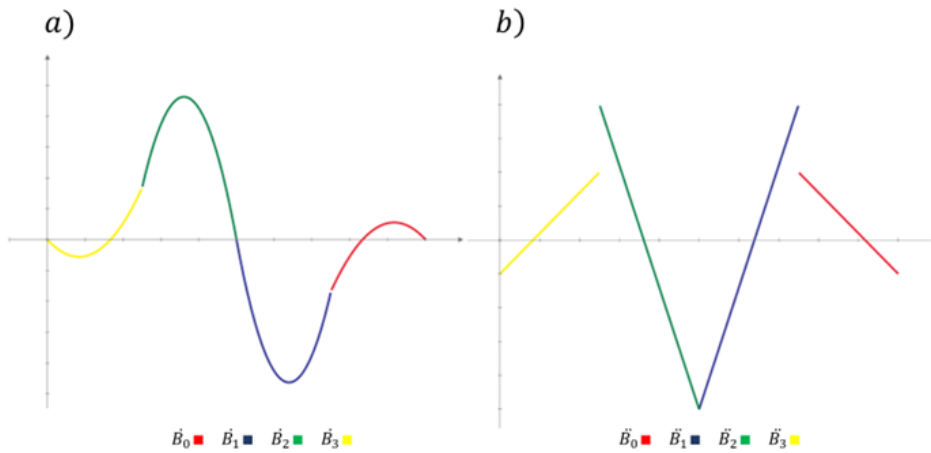


Figura 3.13: a) Primeira derivada das funções base da spline de Catmull-Rom. b) Segunda derivada das funções base da spline de Catmull-Rom.

Como foi demonstrado na Seção 3.3.1, a Equação 3.5 foi derivada de modo a garantir a continuidade C^1 e C^2 da B-Spline (ver Figura 3.14), cumprindo o requisito fundamental para interpolação do traçado. Em termos de carga computacional esta spline não difere da Catmull-Rom, uma vez que também é expressa matricialmente de forma direta. O problema principal das B-Splines foi demonstrado na Figura 3.8; a B-spline é uma spline de aproximação e, portanto, não interpola os pontos de controle, o que do ponto de vista da praticidade de implementação não é interessante, uma vez que os pontos de interpolação do traçado não coincidiriam com os pontos de controle da B-Spline.

Do ponto de vista da praticidade de implementação, uma spline de interpolação é ideal, uma vez que os pontos de controle poderiam ser definidos via interpolação linear das coordenadas de controle (ver Figura 3.9) e coincidiriam com os pontos de controle da spline.

Com os pontos de interpolação do traçado definidos por:

$$P_j(t) = t \cdot (H_j^e - H_j^i) + H_j^i \quad (3.8)$$

E coincidindo com os pontos de controle da spline, o problema de otimização do traçado é convexo. Na equação 3.8, H corresponde as coordenadas de controle e os sobrescritos e e i correspondem a externo e interno respectivamente, referindo-se aos limites da pista. Para o caso de coincidência entre os pontos de controle e os pontos de interpolação o parâmetro de interpolação t varia de zero a um.

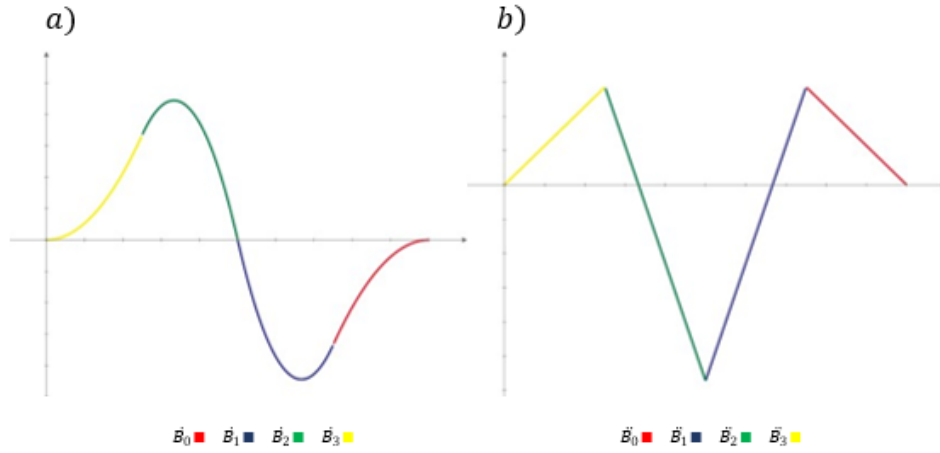


Figura 3.14: a) Primeira derivada das funções base da B-Spline. b) Segunda derivada das funções base da B-Spline.

A B-spline, embora contínua em C^1 e C^2 , não atinge o requisito de praticidade. A curva de Bézier como demonstrada na Seção 3.2 e a spline de Catmull-Rom foram descartadas. Existiria alguma forma de interpolar o traçado de maneira mais prática que por B-Spline?

Sabe-se, até aqui, que a curva de Bézier não possui continuidade C^0 para uma sequência de interpolação de j pontos P na forma P_{n+m} , $n = 0, \dots, j-4$, $m = 0, \dots, 3$. Contudo, estas curvas interpolam os pontos P_n e P_{n+3} (ver Figura 3.4). Partindo deste princípio poderíamos considerar os pontos P_{n+1} e P_{n+2} como pontos auxiliares a_j e b_j respectivamente e enxertá-los na sequência de j pontos P para construir uma spline via curvas de Bézier interpolando j pontos P na forma P_{n+m} , $n = 0, \dots, j-2$, $m = 0, \dots, 1$ (AFLAK, 2020). Reescrevendo a Equação 3.2 para acomodar os pontos auxiliares, temos:

$$Bz^3(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} P_n \\ a_n \\ b_n \\ P_{n+1} \end{bmatrix} \quad (3.9)$$

O objetivo é criar uma transição suave entre duas curvas de Bézier consecutivas, ou seja, garantir continuidade em C^1 e C^2 , isso implica em:

$$\dot{Bz}_{n-1}^3(t=1) = \dot{Bz}_n^3(t=0), \quad n = 1, \dots, j-2 \quad (3.10)$$

$$\ddot{Bz}_{n-1}^3(t=1) = \ddot{Bz}_n^3(t=0), \quad n = 1, \dots, j-2 \quad (3.11)$$

Cada curva Bz_n^3 possui dois pontos auxiliares e, por tanto, é necessário determinar

$2n$ pontos auxiliares numa interpolação de traçado, dos quais $2n - 1$ foram determinados pelas Equações 3.10 e 3.11. Restam dois pontos que são determinados impondo-se as seguintes condições arbitrárias:

$$\ddot{B}z_0^3(t=0) = 0 \quad (3.12)$$

$$\ddot{B}z_{j-2}^3(t=1) = 0 \quad (3.13)$$

Montando-se o sistema a partir das Equações 3.10, 3.11, 3.12 e 3.13 obtém-se as Equações 3.14, 3.15 e 3.16. O procedimento para construção deste sistema está descrito no Apêndice A.

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 4 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 4 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 4 & 1 & 0 \\ 0 & \cdots & 0 & 0 & 1 & 4 & 1 \\ 0 & \cdots & 0 & 0 & 0 & 2 & 7 \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{j-4} \\ a_{j-3} \\ a_{j-2} \end{bmatrix} = \begin{bmatrix} P_0 + 2P_1 \\ 2 \cdot (2P_1 + P_2) \\ 2 \cdot (2P_2 + P_3) \\ \vdots \\ 2 \cdot (2P_{j-4} + P_{j-3}) \\ 2 \cdot (2P_{j-3} + P_{j-2}) \\ 8P_{j-2} + P_{j-1} \end{bmatrix} \quad (3.14)$$

$$b_n = 2P_{n+1} - a_{n+1}, \quad n = 0, \dots, j-3 \quad (3.15)$$

$$b_{j-2} = \frac{a_{j-2} + P_{j-1}}{2}, \quad n = j-2 \quad (3.16)$$

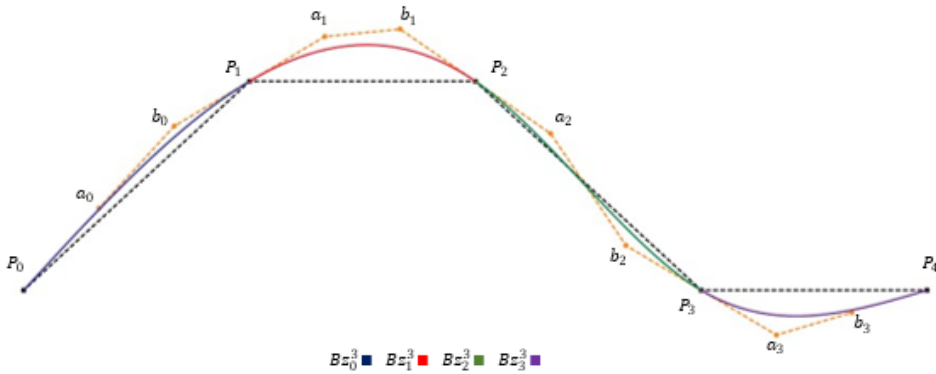


Figura 3.15: Spline formada por curvas de Bézier de grau 3 interpolando cinco pontos de controle.

As Equações 3.14, 3.15 e 3.16 mostram que os pontos auxiliares enxertados na sequência de pontos de interpolação do traçado (ver Figura 3.15) precisam ser determinados sempre que uma alteração no conjunto de j pontos P for realizada, neste caso, para cada iteração no processo de otimização do traçado, um sistema de equações deverá ser solucionado,

umentando a carga computacional. Entretanto essa estratégia de interpolação cumpre tanto com o requisito de praticidade quanto com o de curvatura.

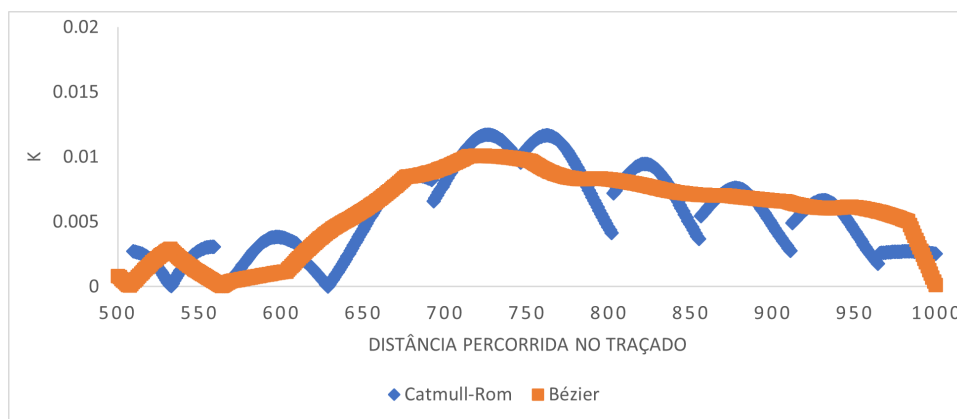


Figura 3.16: Plotagem da curvatura do traçado para um trecho do circuito de Goiânia.

A Figura 3.16 apresenta a curvatura do traçado para um trecho do circuito de Goiânia. Observe como a spline de Catmull-Rom dá “saltos” de curvatura ao longo do traçado, cada “salto” ocorre em um ponto de interpolação. Em contrapartida, a curvatura via spline formada por curvas de Bézier é totalmente contínua, ideal para computar a dinâmica do veículo.

3.4 CONCLUSÃO DO CAPÍTULO

Neste capítulo foi apresentado um estudo sobre interpolação por splines cúbicas visando determinar o melhor método de interpolação para virtualização do circuito e para a otimização do traçado. Foram analisadas as splines de Catmull-Rom e B-Spline, além das curvas de Bézier. O processo de seleção do método de interpolação se baseou em três critérios: praticidade, carga computacional e curvatura.

A virtualização do circuito é o passo inicial do processo de otimização de traçado de competição, uma vez que os limites da pista são restrições para o problema de otimização. Entretanto, o objetivo deste passo é apenas fornecer as coordenadas de controle que serão utilizadas para determinar os pontos de interpolação do traçado, tornando o critério de curvatura dispensável. Sendo assim, a spline de Catmull-Rom foi escolhida atendendo os critérios de praticidade e carga computacional.

A spline de Catmull-Rom, em certas condições, pode gerar uma curva que intercepta a si mesma, interpolando os pontos de forma indesejada. Como solução para este problema, foi proposta uma modificação no processo de derivação da spline, visando balancear as tangentes usadas na determinação dos pontos de Bézier. Como resultado, a spline de Catmull-Rom balanceada é capaz de interpolar pontos com espaçamento irregular sem interceptar a si mesma.

O processo de otimização do traçado envolve a simulação do veículo de competição percorrendo o traçado que se pretende otimizar. As equações que descrevem a dinâmica do veículo são derivadas em função da curvatura do traçado, tornando esta o critério

imprescindível para a seleção da spline que o interpolará. Com base nisso, apenas a B-Spline cumpre com o critério de curvatura contínua, embora seja problemática do ponto de vista de praticidade de implementação. Neste caso, construir uma spline através de segmentos de curva Bézier arranjados de modo a garantir continuidade C^1 e C^2 foi a opção escolhida, visto que cumpre com os requisitos de praticidade e curvatura contínua, embora aumente a carga computacional no processo de otimização.

Neste capítulo será apresentado o processo de modelagem do veículo para as simulações.

MODELAGEM DO VEÍCULO

Em simulação veicular, um modelo de “bicicleta” (MILLIKEN et al., 2003) (GILLESPIE, 2021) (PACEJKA, 2006) como o apresentado na Figura 4.1 é considerado o modelo mais simples com alguma capacidade de representar a dinâmica de um veículo. Este modelo pode ser aplicado para diversos fins, desde aplicações em controle de tração (RAJAMANI, 2006), até predição de traçado ótimo (KAPANIA; SUBOSITS; GERDES, 2019) (LOT; BIRAL, 2014).

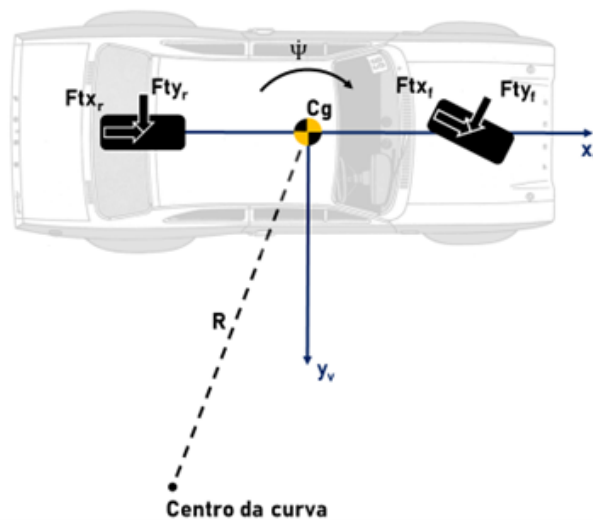


Figura 4.1: Modelo de “bicicleta” (dois graus de liberdade)

A Figura 4.1 destaca algumas variáveis importantes para modelagem com dois ou mais graus de liberdade: as forças laterais e longitudinais atuantes nos pneus ($F_{t(x \text{ ou } y)}$), a velocidade de guinada do veículo ($\dot{\psi}$) e a posição do centro de gravidade (CG). Nas seções 4.1 e 4.2 serão analisados os problemas de disponibilidade de dados para determinação dessas variáveis e o impacto sobre a modelagem veicular.

4.1 DISPONIBILIDADE DE DADOS DE PNEUS

Os pneus, como única interface de contato com o solo, são responsáveis por todos os esforços de controle do veículo. Existem dois mecanismos pelos quais os esforços de controle lateral e longitudinal de um veículo são transmitidos para o solo: histerese e adesão (GILLESPIE, 2021). O termo atrito é utilizado na engenharia automotiva como conveniência para referir-se ao efeito combinado destes dois mecanismos.

Segundo a formulação de Coulomb para o atrito, a força de atrito é linearmente proporcional a carga vertical aplicada: $F_{at} = F_z \cdot \mu$, onde μ representa o coeficiente de atrito. Graças aos fenômenos de histerese e adesão, a força de atrito gerada na interface pneu/solo não obedece à formulação de Coulomb. Em pneus, o comportamento da força de atrito é não linear (ver Figura 4.2).

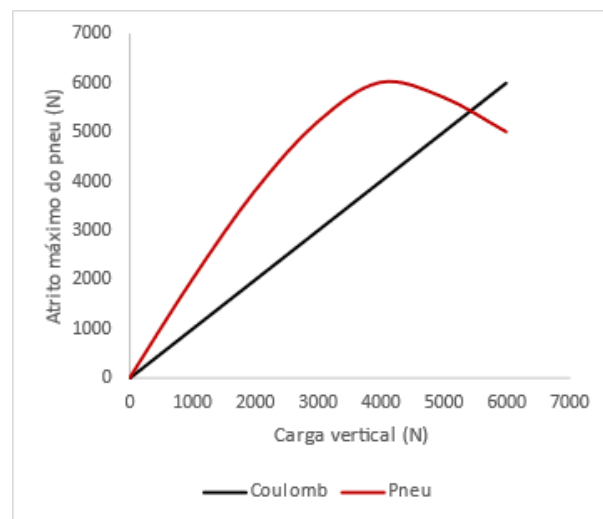


Figura 4.2: Comparação entre atrito de Coulomb e uma curva típica de atrito em pneus. Adaptado de SEWARD (2014)

Cada tipo de pneu possui uma curva de atrito específica que é avaliada em laboratório em um equipamento geralmente chamado de “flat-track” (ver Figura 4.3). Os dados coletados em laboratório geralmente ficam restritos às empresas que desenvolvem os pneus não sendo de acesso livre a equipes de competição, em especial equipes amadoras. Estes testes de laboratório podem ser contratados por iniciativa das equipes em empresas especializadas, embora o custo não seja convidativo.

Ainda que se opte pela aquisição de pneus de empresas como a Avon, que disponibilizam as curvas de atrito de seus pneus, as curvas disponíveis já foram tratadas via Pacejka '96 Model (PACEJKA, 2006) e só estão disponíveis para forças laterais e momento auto-alinhante puros (ver Figura 4.4). Para modelagem veicular apropriada, seria necessário a disponibilidade dos dados de forças longitudinais e forças combinadas.

Alguns autores contornam a indisponibilidade de dados utilizando modelos lineares saturados de pneus (LOT; BIRAL, 2014), isso é possível extraíndo-se dos dados das curvas de força lateral os valores de rigidez à curva do pneu (derivada das curvas para ângulo de deriva nulo) e de coeficiente de atrito máximo (Força lateral/Carga vertical aplicada).



Figura 4.3: "Flat-track" para teste de pneus. Fonte: <https://calspan.com/automotive/tire-performance-testing/motorsports-tire-testing>

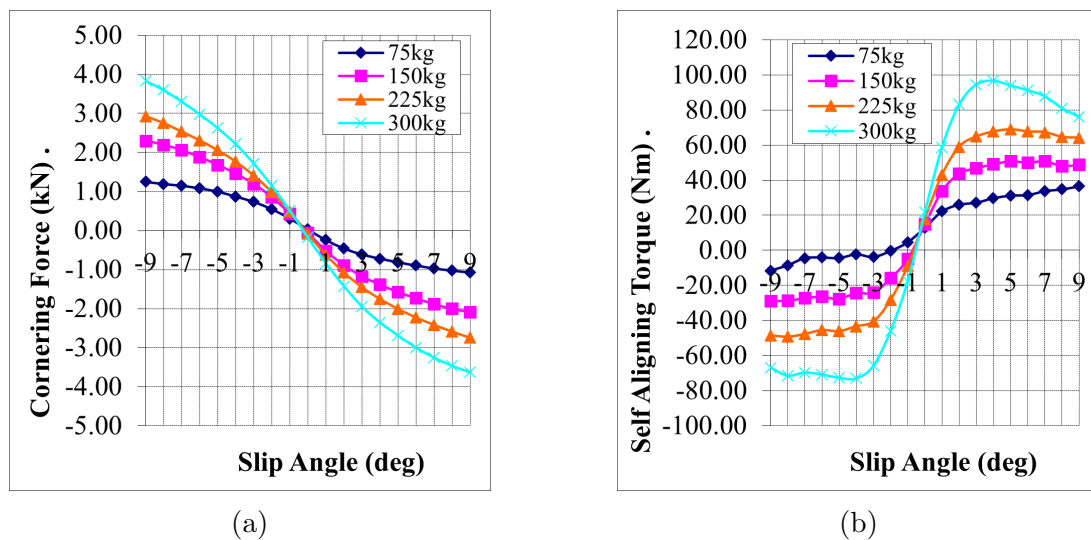


Figura 4.4: Curvas de força lateral (Cornering Force) e momento auto-alinhante (Self Aligning Torque) disponibilizados pela Avon (AVON, 2023). "Slip Angle" corresponde ao ângulo de deriva do pneu (diferença entre a direção em que a roda aponta e a direção em que se desloca).

Outros autores optam por extrapolar os dados via Brush Model (KAPANIA; SUBOSITS; GERDES, 2019).

O problema de modelagem veicular se agrava quando não há sequer os dados básicos como os apresentados na Figura 4.4, neste caso não é possível modelar os pneus e, portanto, não é possível modelar o veículo com dois ou mais graus de liberdade.

4.2 DISPONIBILIDADE DE DADOS DE PROJETO

Para analisar o problema da disponibilidade de dados de projeto do veículo vamos avaliar as equações que modelam um carro com dois graus de liberdade (modelo de "bicicleta").

Considerando o veículo da Figura 4.1 se deslocando em uma superfície plana em trajetória circular e um sistema de coordenadas cartesiano, onde x e y correspondem ao sistema de coordenadas móvel que se desloca com o veículo em relação a um sistema de coordenadas global (sendo x a direção longitudinal e y a direção lateral), as equações diferenciais para aceleração lateral e velocidade de guinada são (RAJAMANI, 2006):

$$m \left(\ddot{y} + V_x \cdot \dot{\psi} \right) = Ft_{yf} + Ft_{yr} \quad (4.1)$$

$$I_{zz} \cdot \dot{\psi} = a \cdot Ft_{yf} - b \cdot Ft_{yr} \quad (4.2)$$

Em que m corresponde a massa do veículo, a e b correspondem as distâncias dos eixos dianteiro e traseiro do veículo em relação ao centro de gravidade respectivamente e I_{zz} corresponde ao momento principal inércia do veículo em relação ao eixo vertical.

A posição do centro de massa do veículo é um parâmetro verificável. Dispondo de quatro balanças sob as rodas coleta-se a carga vertical nas quatro rodas do veículo. Com os valores coletados executa-se o seguinte procedimento (MILLIKEN et al., 2003):

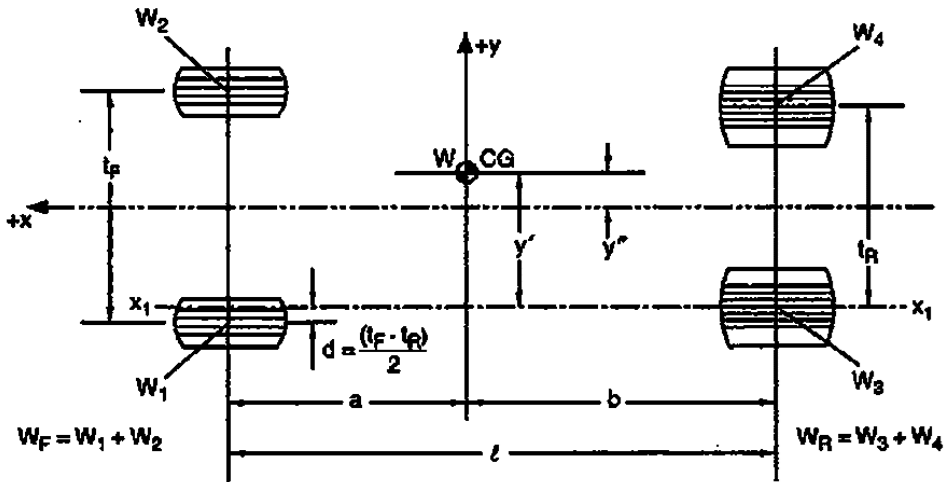


Figura 4.5: Determinação da posição do centro de massa do veículo. Fonte: (MILLIKEN et al., 2003)

Toma-se um eixo de referência – eixo $x_1 - x_1$ na Figura 4.5 – e calcula-se o somatório de momentos em relação a este eixo. Seguindo o exemplo da Figura 4.5:

$$y' = \frac{W_2}{W} (t_f - d) - \frac{W_1}{W} \cdot d + \frac{W_4 \cdot t_r}{W} \quad (4.3)$$

$$y'' = y' - \frac{t_r}{2} \quad (4.4)$$

Onde W_1 a 4 correspondem as cargas coletadas e t_f ou t_r correspondem as bitolas dianteira e traseira respectivamente. As Equações 4.3 e 4.4 determinam a posição do centro

de gravidade em relação ao eixo longitudinal do veículo, resta determinar a posição em relação ao eixo transversal:

$$a = \frac{W_3 + W_4}{W} \cdot l \quad (4.5)$$

$$b = \frac{W_1 + W_2}{W} \cdot l \quad (4.6)$$

Uma vez determinadas as posições do centro de gravidade em relação aos eixos transversal e longitudinal do veículo, resta determinar sua altura em relação ao solo através do seguinte procedimento prático:

- Aumentar a pressão dos pneus;
- Substituir os amortecedores por gabaritos;
- Travar o sistema de direção do veículo;
- Com o auxílio de uma tilt table inclinar o veículo até alcançar o ponto de equilíbrio (ver Figura 4.6);
- Calcular a altura do centro de gravidade via Equação 4.7, onde h_{cg} corresponde a altura do centro de gravidade e τ corresponde a largura do pneu.

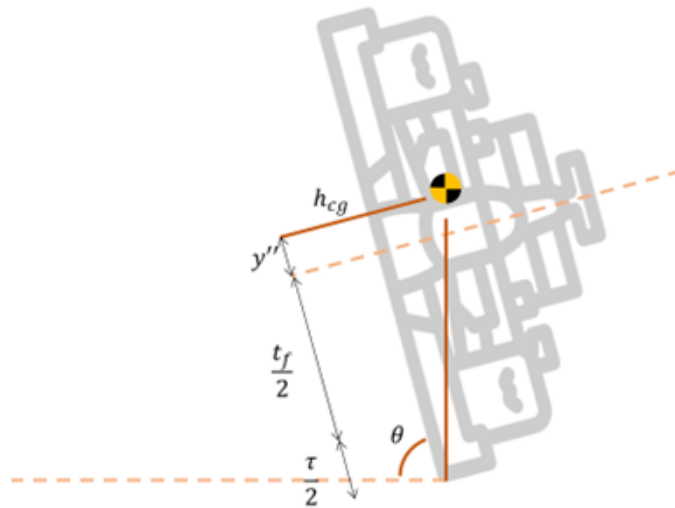


Figura 4.6: Determinação da altura do centro de gravidade de um veículo via tilt table.

$$h_{cg} = \tan(90^\circ - \theta) \cdot \left(\frac{t_f}{2} + y'' + \frac{\tau}{2} \right) \quad (4.7)$$

No caso de a equipe possuir acesso ao projeto do veículo, o valor do momento principal de inércia I_{zz} está disponível, porém, grande parte dos veículos de competição amadora ou

semiprofissional são carros de rua adaptados para as pistas e seus projetos de adaptação não fornecem esse dado. Sendo assim, os valores dos momentos principais de inércia do veículo devem ser aproximados via Equação 4.8 (BLUNDELL; HARTY, 2014):

$$I = m \cdot D^2 \quad (4.8)$$

Em que D é a distância entre o centro de gravidade (CG) de um componente do veículo e o eixo principal de inércia de interesse. Para ilustrar o procedimento tomemos o exemplo apresentado por Danny Nowlan para um carro de fórmula 3 (ver Tabela 4.1) (NOWLAN, 2020).

Tabela 4.1: Tabela para o cálculo do momento principal de inércia em torno do eixo vertical para um Fórmula 3. Adaptado de: (NOWLAN, 2020)

Componente	Massa	Distância do CG do componente para o CG do carro	$m \cdot D^2$
Massa não suspensa dianteira	20kg	1.6m	51.2kgm ²
Massa não suspensa traseira	30kg	1.1m	36.3kgm ²
Kit aerodinâmico frontal	100kg	1.2m	144kgm ²
Motor	100kg	0.4m	16kgm ²
Caixa de marchas/asa traseira	100kg	1.2m	144kgm ²
Piloto	70kg	0.4m	11.2kgm ²
Total			402.7kgm ²

Comparando o valor de I_{zz} encontrado na Tabela 4.1 com o valor aproximado de projeto de um Fórmula 3, 600kgm² (NOWLAN, 2020), o erro é de aproximadamente 33%. Isso mostra que na falta de valores de projeto, aproximações, mesmo que necessárias, são bastante imprecisas.

A Figura 4.7 apresenta uma comparação de resultados simulados de velocidade angular para um veículo de competição genérico com momento principal de inércia I_z e este mesmo veículo com momento principal de inércia $0.7 \cdot I_z$. Nota-se que a diferença entre I_z e $0.7 \cdot I_z$ tem grande influência sobre a representação do regime transiente do veículo em simulações dinâmicas e influência desprezível em regime permanente. É, portanto, desprezível em simulações quase-estáticas.

Os resultados apresentados na Figura 4.7 foram obtidos via simulação de um veículo de competição genérico utilizando um modelo não linear de dois graus de liberdade (modelo de “bicicleta”). Este tipo de modelagem não faz parte do escopo deste trabalho, porém está completamente descrita no Apêndice B.

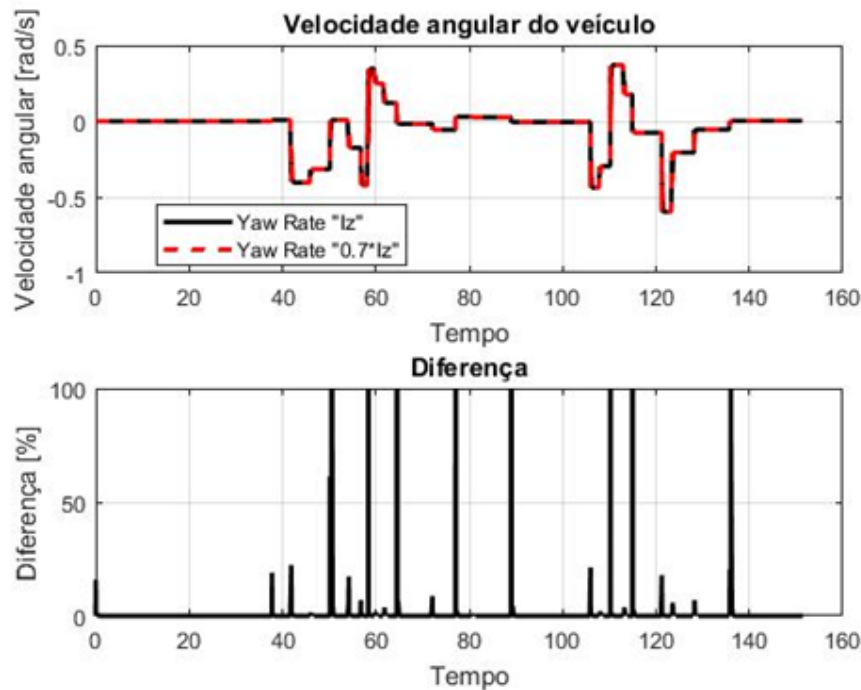


Figura 4.7: Comparação entre a velocidade angular de um veículo com momento principal de inércia I_z e o mesmo veículo com momento principal de inércia $0.7 \cdot I_z$.

4.3 MODELAGEM COM UM GRAU DE LIBERDADE

Como foi demonstrado nas Seções 4.1 e 4.2, os momentos principais de inércia são dados que, quando não disponíveis, podem ser estimados sem grandes prejuízos para simulações quase-estáticas, contudo, a falta de dados sobre a dinâmica dos pneus é proibitiva para qualquer simulação com dois ou mais graus de liberdade. Nos resta, então, as simulações com apenas um grau de liberdade.

O conceito de círculo de tração apresentado na Seção 2.1 diz respeito a capacidade de geração de atrito de apenas um dos pneus, entretanto este conceito pode ser expandido para representar o veículo como um todo, levando-se em consideração o efeito combinado dos quatro pneus.

Quando utilizado para representar o veículo, o círculo de tração não mais utiliza a capacidade de atrito do pneu como limite do diagrama, mas a capacidade combinada dos quatro pneus expressa em termos de capacidade de aceleração lateral e longitudinal do veículo. Dessa forma, os limites do veículo são aquisitáveis via acelerômetro (ver Figura 4.8).

A Figura 4.8 apresenta os dados coletados de um carro de competição no circuito de Goiânia (em azul). Observe que o limite teórico (círculo em laranja) não é atingido em todos os momentos, isso acontece devido às seguintes razões (MILLIKEN et al., 2003):

- Limite de tração – a maioria dos veículos não possui potência o bastante para requisitar todo o potencial dos pneus;

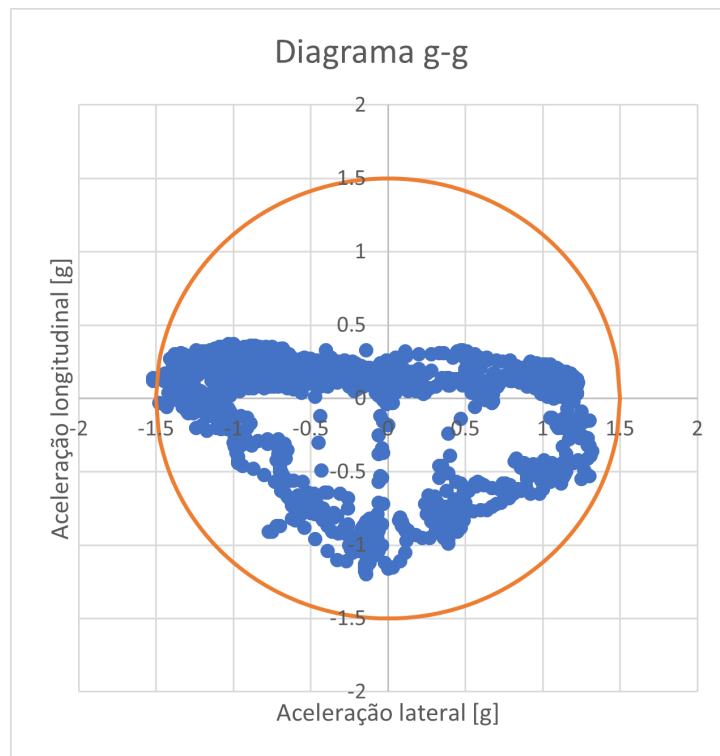


Figura 4.8: Diagrama g-g. Em azul os dados coletados de um carro de competição no circuito de Goiânia e em laranja o círculo de tração teórico.

- Efeito da transferência de carga – durante as manobras acontecem transferências de carga nos sentidos longitudinal e transversal (lateral) do carro. Os pneus respondem de forma não linear as cargas aplicadas, alterando os limites de fricção;
- Efeitos da suspensão – Camber (ângulo de rebatimento dos pneus) e deformações elásticas na geometria da suspensão, bem como nas buchas, afetam a orientação das rodas e, por tanto, as forças resultantes nos pneus;
- Balanço ou equilíbrio do carro – as condições de operação das quatro rodas de um veículo raramente permitem que elas atinjam seus limites de fricção ao mesmo tempo. As rodas que atingem seus limites mais cedo tornam-se fator limitante para a capacidade total do veículo. Em carros de competição este efeito é minimizado, uma vez que os veículos são projetados para atingir os limites de fricção dos pneus aproximadamente ao mesmo tempo nos ápices de curva.
- Balanço de freios – a distribuição de carga de frenagem entre os eixos dianteiro e traseiro pode inibir o uso da capacidade total de um dos eixos devido a saturação do outro.

Para melhor representar as capacidades de um veículo real, o conceito de círculo de tração é expandido para uma elipse saturada na aceleração (parte positiva do eixo y na Figura 4.8). A equação para a elipse de tração saturada é:

$$\begin{cases} \frac{g_x^2}{g_{x_{max}}^2} + \frac{g_y^2}{g_{y_{max}}^2} = 1, & \text{se } g_x \leq T_{limite} \\ T_{limite}, & \text{se } g_x \geq T_{limite} \end{cases} \quad (4.9)$$

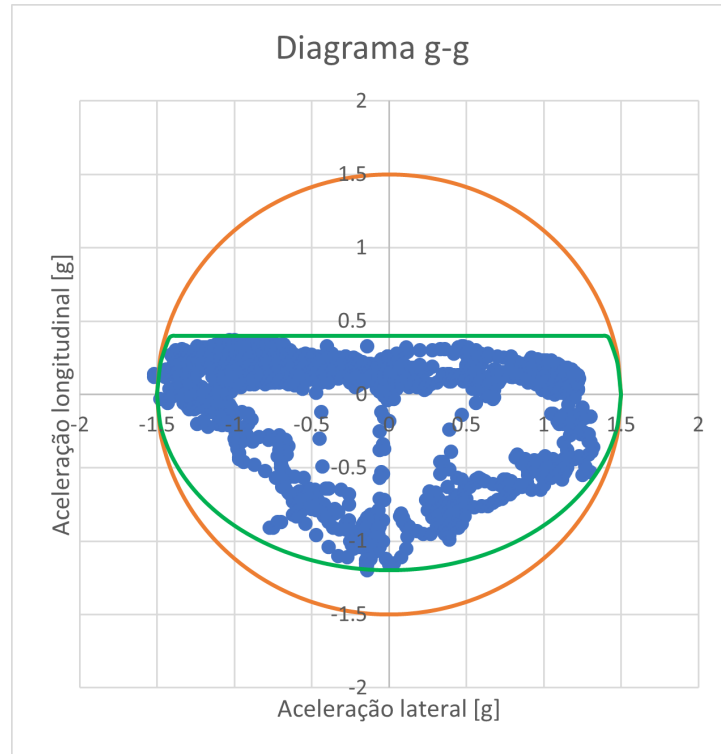


Figura 4.9: Diagrama g-g. Em azul os dados coletados de um carro de competição no circuito de Goiânia, em laranja o círculo de tração teórico e em verde a elipse de tração saturada.

A Figura 4.9 apresenta os dados coletados de um carro de competição no circuito de Goiânia (em azul), comparados com o círculo de tração (em laranja) e a elipse saturada de tração (em verde). Note que, apesar de representar melhor os limites do veículo, a elipse saturada ainda sobre-estima as capacidades do veículo e isso acontece devido aos parâmetros usados para construí-la. Analisando a Equação 4.9, apenas três parâmetros são usados para representar o veículo:

- A capacidade de aceleração longitudinal máxima $g_{x_{max}}$;
- A capacidade de aceleração lateral máxima $g_{y_{max}}$;
- O limite de tração do veículo T_{limite} .

Estes parâmetros definem os quatro pontos vermelhos mostrados na Figura 4.10, fica por conta da elipse a extrapolação dos dados e representação das capacidades do veículo nos demais pontos. Os quatro pontos destacados representam os limites do veículo em

condições de “força pura”, ou seja, a demanda de atrito nos pneus ou é puramente longitudinal ou puramente lateral. Para estas condições a elipse saturada é suficiente para representar o veículo, contudo, para as condições de “força combinada” (demais pontos) ela falha em representá-lo.

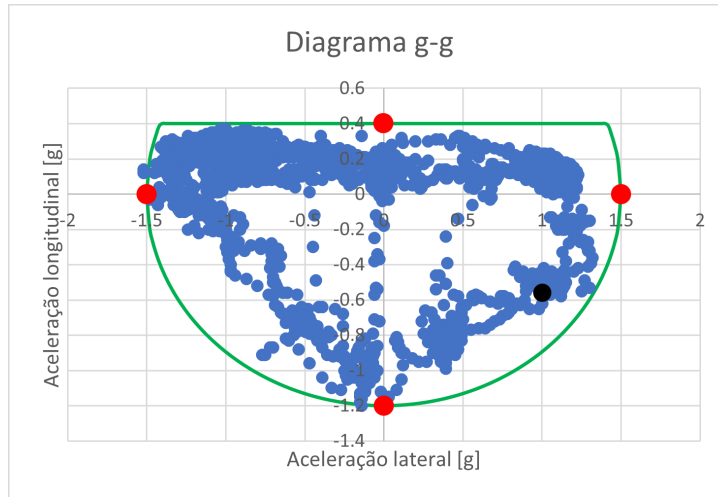


Figura 4.10: Diagrama g-g. Em azul, os dados coletados em um carro de competição no circuito de Goiânia. Em verde, a elipse saturada de tração. Em vermelho, os quatro pontos usados para a construção da elipse.

Para entender como as condições de força combinada afetam o balanço do veículo é necessário recorrer a uma ferramenta de análise conhecida como diagrama de momento de yaw (DMY). O processo de construção de um DMY está descrito em ROUELLE (2017), aqui iremos nos ater à interpretação do DMY em relação aos pontos de dados do diagrama g-g.

A Figura 4.11 apresenta um DMY para um carro de competição genérico configurado para atingir a mesma capacidade de aceleração lateral do veículo no qual os dados apresentados na Figura 4.10 foram coletados (o DMY do próprio veículo não pôde ser construído devido à falta de dados dos pneus). Os pontos definidos pela elipse de tração são os pontos de equilíbrio dinâmico do veículo, mais especificamente os pontos de capacidade máxima de aceleração lateral/longitudinal em equilíbrio dinâmico.

Analisando a Figura 4.11, todos os pontos sobre o eixo das abscissas são pontos de equilíbrio dinâmico do veículo, sendo os pontos azuis em destaque os pontos de capacidade de aceleração lateral máximos em equilíbrio dinâmico. Todos os outros pontos do diagrama, onde o momento de yaw (M_{yaw}) $\neq 0$, são pontos atingidos apenas em regime transiente durante a execução de alguma manobra e sua análise está fora do escopo deste trabalho.

Na Figura 4.11, os pontos em destaque correspondem aos pontos de aceleração lateral máximos de $\pm 1.5g$ em destaque na Figura 4.10. Para esta condição apenas aceleração lateral está sendo requisitada dos pneus. A Figura 4.12 apresenta o ponto de aceleração lateral máxima numa curva para esquerda em frenagem, requisitando dos pneus aceleração longitudinal e lateral simultaneamente. Observe que a capacidade lateral máxima do

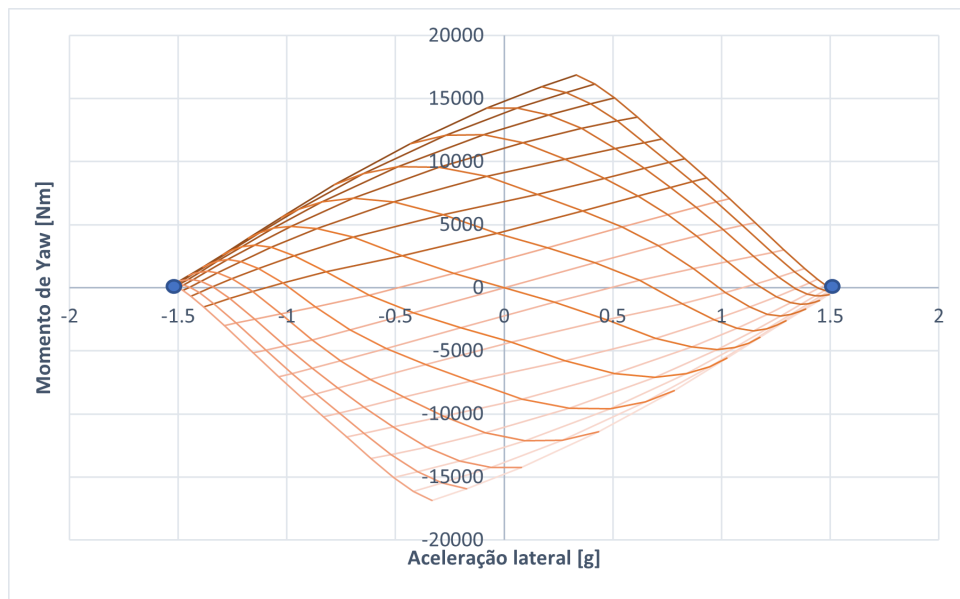


Figura 4.11: Diagrama de momento de yaw de um carro de competição genérico.

veículo reduz para 1.0g aproximadamente (ponto preto na Figura 4.10) e não pertence mais ao conjunto de pontos dados pela elipse de fricção. Este exemplo ilustra o maior problema da modelagem com um grau de liberdade: a impossibilidade de prever os efeitos da transferência de carga (dentre outros fenômenos) na capacidade trativa do veículo.

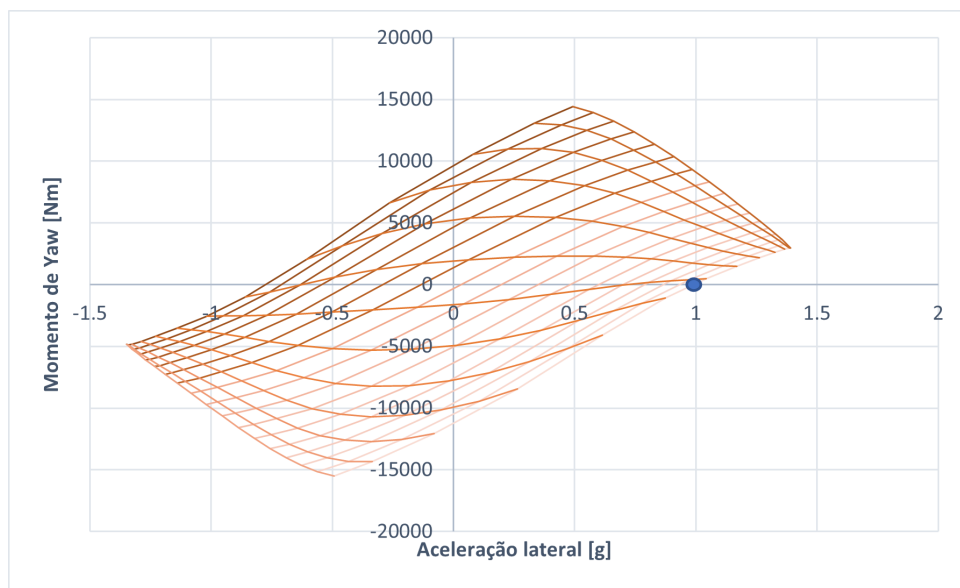


Figura 4.12: Diagrama de momento de yaw de um carro de competição genérico em aceleração.

Para compensar parte da limitação da modelagem com um grau de liberdade será desenvolvido um modelo super-elíptico capaz de regredir o comportamento dinâmico de

um veículo com base em dados de acelerômetro. A predição do comportamento, por sua vez, permanece como um problema a ser solucionado.

4.3.1 Modelo super-elíptico de um grau de liberdade

Na Figura 4.10, os pontos vermelhos no eixo das ordenadas representam as capacidades de aceleração (positiva) e desaceleração (negativa) máximas do veículo. Para um modelo quase estático recorre-se ao princípio de d'Alembert para determiná-los (SEWARD, 2014).

Segundo o Princípio de d'Alembert (MEIROVITCH, 2012): O trabalho virtual total realizado pelas forças impressas para deslocamentos reversíveis, compatíveis com as forças de vínculo, é zero:

$$\sum_{i=1}^N (F_i - \dot{p}_i) \cdot \delta r_i = 0 \quad (4.10)$$

A Equação 4.10 encapsula o princípio de d'Alembert e o princípio dos trabalhos virtuais, note que o termo referente as forças de vínculo (F'_i) não consta na equação, apenas o termo referente as forças aplicadas (F_i), o termo referente as forças inerciais (\dot{p}_i) e o termo referente aos deslocamentos virtuais (δr_i). N corresponde às n partículas que compõem um sistema.

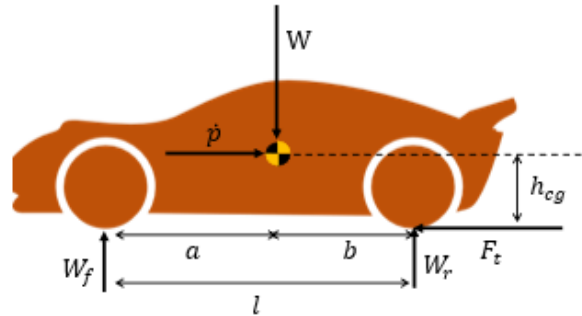


Figura 4.13: Condição de equilíbrio estático para um veículo (princípio de d'Alembert).

Partindo da condição de equilíbrio estático de um veículo se deslocando sobre uma superfície plana (ver Figura 4.13) obtém-se as seguintes equações para o caso de tração traseira:

$$F_t = \mu_{lgt} \cdot W_r = \dot{p} = \frac{W}{g} \cdot \ddot{x} \quad (4.11)$$

$$W = W_f + W_r \quad (4.12)$$

$$b \cdot W_r = a \cdot W_f + F_t \cdot h_{cg} \quad (4.13)$$

Em que F_t corresponde à força de tração, g corresponde à aceleração da gravidade e μ_{lgt} corresponde ao coeficiente de atrito longitudinal dos pneus. Solucionando o sistema de Equações 4.11 a 4.13 obtém-se a equação para o limite de tração do veículo:

$$\ddot{x} = T_{pneu}|_{limite}^t = \frac{a \cdot g \cdot \mu_{lgt}}{l - h_{cg} \cdot \mu_{lgt}} \quad (4.14)$$

Para o caso de tração dianteira o procedimento é semelhante e resulta em:

$$\ddot{x} = T_{pneu}|_{limite}^d = \frac{b \cdot g \cdot \mu_{lgt}}{l + h_{cg} \cdot \mu_{lgt}} \quad (4.15)$$

As Equações 4.14 e 4.15 definem a capacidade de aceleração máxima do veículo limitada pela capacidade dos pneus, contudo, também é preciso considerar as limitações de potência do motor.

O objetivo da construção deste modelo é representar a dinâmica do veículo utilizando o mínimo possível de dados aquisitados. Assim, o torque do motor também será determinado com base nos dados de acelerômetro do veículo via Equação 4.16 (SEGERS, 2014):

$$Tq_{motor} = \frac{m \cdot (1.04 + 0.0025 \cdot i_{total}^2) \cdot \ddot{x} \cdot r_e + (f_r \cdot W + F_d) \cdot r_e}{i_{total}} \quad (4.16)$$

Em que i_{total} representa a redução total da transmissão (marcha + diferencial) e f_r corresponde ao coeficiente de resistência à rolagem dos pneus. O coeficiente de resistência a rolagem varia entre 0.01 e 0.04, sendo 0.015 o valor típico para pneus de carro de passeio (RAJAMANI, 2006).

O termo F_d na Equação 4.16 corresponde à força de arrasto aerodinâmico do veículo calculada pela Equação 4.17 (MCBEATH, 2015), com ρ correspondendo à densidade do ar, A_f correspondendo à área frontal do veículo e C_d correspondendo ao coeficiente de arrasto aerodinâmico do veículo.

O termo r_e na Equação 4.16 corresponde ao raio efetivo do pneu calculado pela Equação 4.18 (RAJAMANI, 2006), em que r corresponde ao raio do pneu sem carga e r_s corresponde ao raio do pneu carregado.

$$F_d(\dot{x}) = 0.5 \cdot \rho \cdot A_f \cdot C_d \cdot \dot{x}^2 \quad (4.17)$$

$$r_e = \frac{\sin(\cos^{-1}(\frac{r_s}{r}))}{\cos^{-1}(\frac{r_s}{r})} \cdot r \quad (4.18)$$

Uma vez determinado o torque do motor é possível aproximar a curva de tração limitada pela potência do veículo via Equações 4.19 e 4.20:

$$T_{i_{total}}|_i = \frac{Tq_{motor} \cdot i_{total}|_i}{m \cdot r_e}, \quad i = 1, \dots, n^o \text{ de marchas} \quad (4.19)$$

$$T_{pot}|_{limite}(\dot{x}) = T_a \cdot \dot{x}^2 - T_b \cdot \dot{x} + T_c \quad (4.20)$$

A Equação 4.19 fornece n pontos de limite de tração correspondentes ao número de marchas do veículo, enquanto a Equação 4.20 é a função que interpola os pontos da Equação 4.19, sendo $T_{a,b \ e \ c}$ os parâmetros de regressão da função.

Com as Equações 4.16 e 4.20 é possível construir a curva total de limite de tração do veículo (capacidade de aceleração):

$$\ddot{x}|_{acc}(\dot{x}) = \begin{cases} T_{pot}|_{limite} - \frac{F_d}{m}, & T_{pot}|_{limite} \leq T_{pneu}|_{limite}^{t ou d} \\ T_{pneu}|_{limite}^{t ou d} - \frac{F_d}{m}, & T_{pot}|_{limite} > T_{pneu}|_{limite}^{t ou d} \end{cases} \quad (4.21)$$

O modelo que está sendo desenvolvido tem por objetivo representar veículos de competição, sendo assim, a Equação 4.20 só é válida para o regime de operação do veículo em pista, ou seja, com a borboleta de aceleração totalmente aberta ou totalmente fechada na maior parte do tempo. Esta equação foi proposta como uma simplificação dos sistemas de força e transmissão do veículo que permite executar a simulação sem recorrer aos mapas desses sistemas.

Para determinar a curva de desaceleração máxima, será considerado que o sistema de freio é capaz de utilizar toda a capacidade dos pneus (o que é verdadeiro para veículos de competição), neste caso o limite de desaceleração corresponde ao próprio μ_{lgt} . Contudo as forças aerodinâmicas também contribuem para a frenagem, assim, a capacidade de desaceleração (frenagem) máxima do veículo é calculada via:

$$\ddot{x}|_{fre}(\dot{x}) = - \left(\mu_{lgt} \cdot \tanh(\dot{x}) + \mu_{aero} \cdot \mu_{lgt} + \frac{F_d}{m} \right) \quad (4.22)$$

Em que o termo $\tanh(\dot{x})$ foi adicionado como recurso de simulação para evitar que o veículo entre em aceleração negativa após atingir velocidade nula (ver Figura 4.14) e μ_{aero} contabiliza o ganho em capacidade de frenagem devido a força de sustentação aerodinâmica, conforme:

$$\mu_{aero}(\dot{x}) = \frac{-F_l}{W} \quad (4.23)$$

Na Equação 4.23, F_l representa a força de sustentação aerodinâmica conforme Equação 4.24 (MCBEATH, 2015), onde C_l corresponde ao coeficiente de sustentação aerodinâmica do veículo:

$$F_l(\dot{x}) = 0.5 \cdot \rho \cdot A_f \cdot C_l \cdot \dot{x}^2 \quad (4.24)$$

Até aqui foram determinados os limites de aceleração longitudinais do veículo, sumarizados na Figura 4.14, resta o limite de aceleração lateral, que é composto pelo coeficiente de fricção lateral do veículo μ_{lat} acrescido da contribuição dos efeitos aerodinâmicos, conforme:

$$\ddot{y}|_{limite}(\dot{x}) = \mu_{lat} + \mu_{aero} \cdot \mu_{lat} \quad (4.25)$$

Com todos os limites de aceleração definidos, o modelo super-elíptico proposto para a modelagem de veículos com um grau de liberdade é dado por:

$$\ddot{x}(\dot{x}, \dot{y}) = \begin{cases} \left(|\ddot{x}|_{acc}(\dot{x})|^{m_a} \left(1 - \left| \frac{\ddot{y}}{\ddot{y}|_{limite}} \right|^{n_a} \right) \right)^{\frac{1}{m_a}}, & \text{aceleração} \\ \left(|\ddot{x}|_{fre}(\dot{x})|^{m_b} \left(1 - \left| \frac{\ddot{y}}{\ddot{y}|_{limite}} \right|^{n_b} \right) \right)^{\frac{1}{m_b}}, & \text{frenagem} \end{cases} \quad (4.26)$$

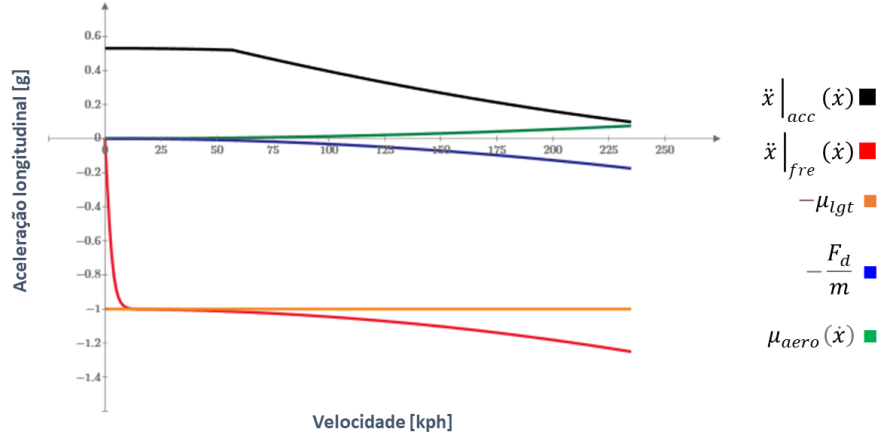


Figura 4.14: Limites de aceleração longitudinal ($\ddot{x}|_{acc}(\dot{x})$ e $\ddot{x}|_{fre}(\dot{x})$) e parcelas das equações.

Analisando a Equação 4.26, nota-se que o modelo está definido em função da velocidade do veículo \dot{x} e da aceleração lateral \ddot{y} . Definir as equações do veículo em função da velocidade é conveniente para uma simulação quase-estática, uma vez que podemos integrar numericamente a velocidade ao longo do traçado via Equação 4.27. Da mesma forma, vincular o modelo do veículo a aceleração lateral \ddot{y} também é conveniente, quando se representa o traçado por spline de interpolação, neste caso \ddot{y} é calculada diretamente em função da curvatura do traçado via Equação 4.28.

$$\dot{x}_i = \begin{cases} \sqrt{\dot{x}_{i-1}^2 + (2 \cdot \ddot{x}_{i-1} \cdot s_i)}, & \text{aceleração} \\ \sqrt{\dot{x}_{i+1}^2 + (2 \cdot \ddot{x}_{i+1} \cdot s_{i+1})}, & \text{frenagem} \end{cases} \quad (4.27)$$

$$\ddot{y}_i = \dot{x}_i^2 \cdot \kappa_i \quad (4.28)$$

Nas Equações 4.27 e 4.28, i corresponde a um ponto no traçado discretizado em n pontos e s_i corresponde à distância percorrida entre dois pontos consecutivos $\overline{p_{i-1}p_i}$. Note que o processo de integração acontece em dois sentidos: progressivamente (aceleração), enquanto $\ddot{y}_i \leq \ddot{y}_{limite}$ e regressivamente (frenagem), até que $\dot{x}_i \geq \dot{x}_{i-1}$.

O modelo super-elíptico de um grau de liberdade está completo. Os parâmetros m_a , m_b , n_a e n_b são determinados em função dos dados de acelerômetro do veículo de competição a ser modelado. Uma metodologia para identificação para esses parâmetros foi desenvolvida e será apresentada na Seção 4.3.2.

4.3.2 Identificação dos parâmetros do modelo super-elíptico

O modelo super-elíptico apresentado na Seção 4.3.1 não é preditivo, ele apenas regride os dados do sensoriamento do veículo para modelar suas características dinâmicas. O intuito deste modelo é ser aplicado como ferramenta de auxílio no aprimoramento de

pilotos amadores. A principal diferença entre um piloto amador e um piloto profissional é a consistência durante a pilotagem. Ambos os pilotos são capazes de atingir o limite do veículo, mas só o piloto profissional atinge e sustenta o veículo no limite de sua capacidade durante várias voltas consecutivas.

A metodologia para identificação dos parâmetros será apresentada em uma série de passos. Serão usados dados coletados em um veículo de competição no circuito de Goiânia pilotado por um profissional. Embora os dados tenham sido obtidos com piloto profissional – o que permite a validação da metodologia –, a metodologia será apresentada para aplicação com pilotos amadores.

- Passo 1 – Identificação dos dados básicos.

A Tabela 4.2 apresenta os dados necessários para a construção do modelo super-elíptico:

Tabela 4.2: Informações básicas do veículo

Informações do veículo	
Distância entre-eixos	2.6m
Distribuição de massa	62% dianteira
Altura do CG	0.42m
Massa	1090kg
Densidade do ar	1.162kg/m ³
Área frontal	2.16m ²
Raio do pneu	0.32m
Redução total 1 ^a marcha	9.857
Redução total 2 ^a marcha	6.571
Redução total 3 ^a marcha	5.163
Redução total 4 ^a marcha	4.381
Redução total 5 ^a marcha	3.977
Redução total 6 ^a marcha	3.651
Tração	Dianteira

- Passo 2 – Determinar a curva de tração limitada pela potência.

Os dados apresentados na Figura 4.15 se referem a melhor volta do piloto na pista. Para determinar a curva de tração limitada pela potência serão necessários os dados de velocidade (azul), abertura da borboleta de aceleração (verde), aceleração longitudinal (laranja) e marcha engatada (vermelho).

A curva de abertura da borboleta de aceleração será usada para filtrar os demais dados, serão usados apenas os pontos em que a abertura da borboleta é 100%. A curva de marcha

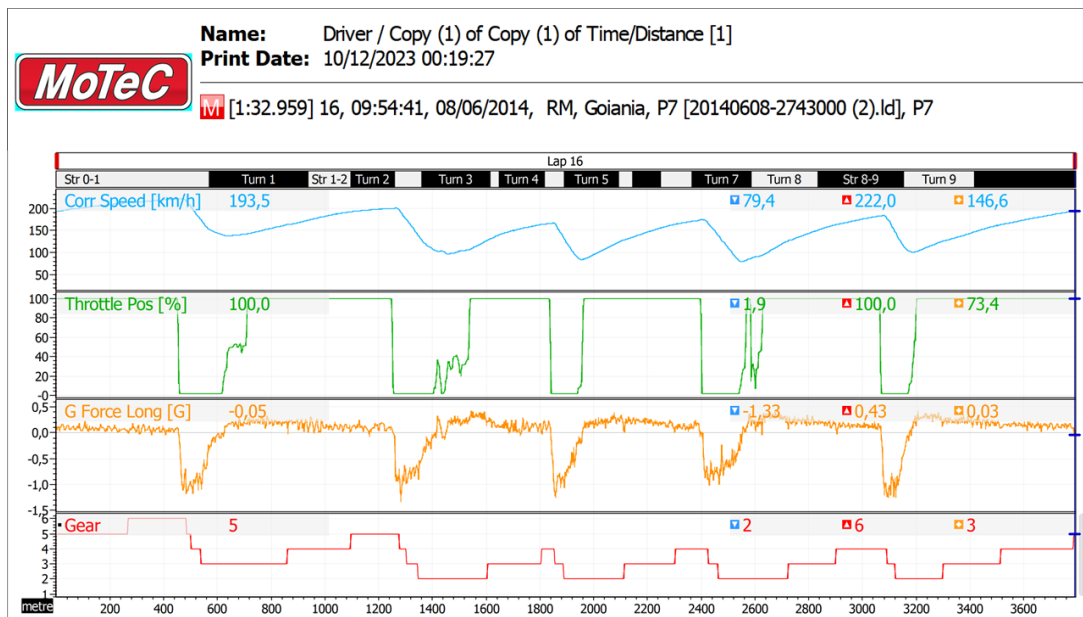


Figura 4.15: Dados para determinar a curva de tração limitada pela potência.

engatada deve ser convertida em uma curva de redução da transmissão, substituindo os índices de marcha engatada (de 1 a 6) pelas suas respectivas reduções (ver Figura 4.16).

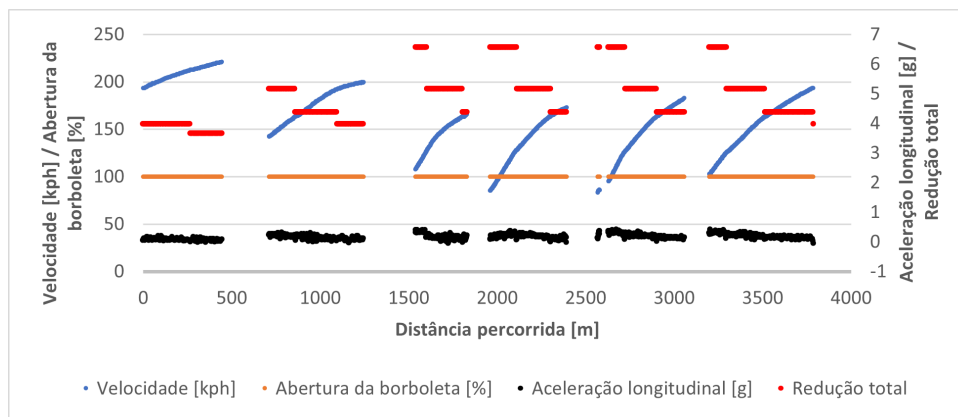


Figura 4.16: Dados filtrados para determinar a curva de tração limitada pela potência.

Aplicando a Equação 4.16 nos dados filtrados da Figura 4.16, calcula-se o torque do motor. Observe que para abertura de 100% da borboleta de aceleração, o torque apresenta uma tendência linear constante (ver Figura 4.17). Será utilizado para a construção do modelo super-elíptico o torque médio, que para o veículo em análise é de aproximadamente 220Nm.

Aplicando o valor de torque médio na Equação 4.19 calculam-se os limites de tração por marcha (ver Figura 4.18). A Equação 4.20 interpola os pontos da Equação 4.19, com os parâmetros $T_{a,b}$ e c iguais a $0.0008 \frac{1}{m}$, $0.1112 \frac{1}{s}$ e $6.2189 \frac{m}{s^2}$, respectivamente.

- Passo 3 – Determinar coeficientes e parâmetros

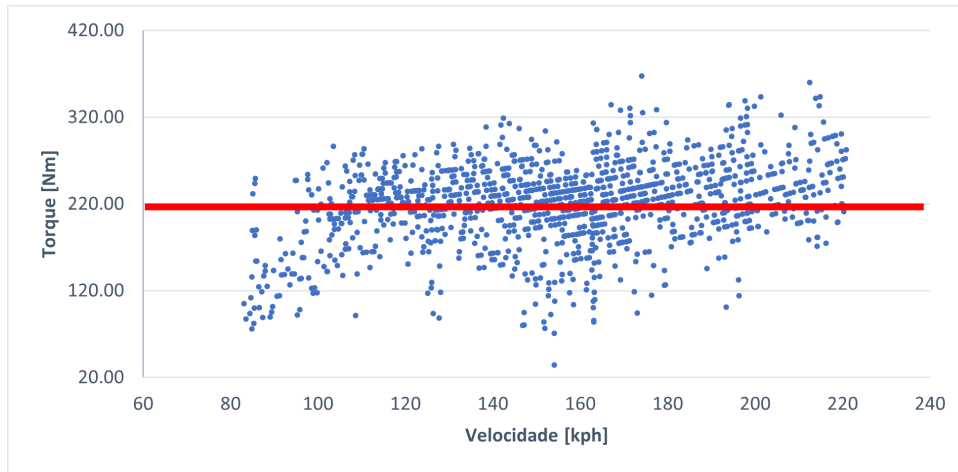


Figura 4.17: Torque do motor a 100% de abertura da borboleta de aceleração. Linha vermelha destaca o torque médio.

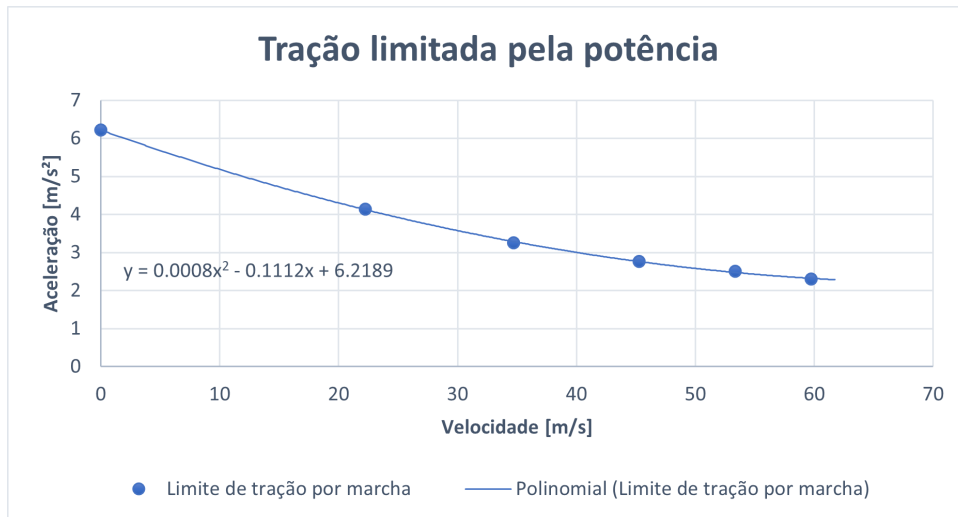


Figura 4.18: Curva de tração limitada pela potência.

Para determinar os parâmetros de construção da super-elipse é preciso solucionar um problema de otimização. O objetivo é determinar o conjunto de parâmetros para que a super-elipse melhor represente as características do veículo.

Um problema de otimização genérico é dado por:

$$\min_u F_{obj}(u) \quad (4.29)$$

Sujeito a:

$$g(u) \leq 0 \quad (4.30)$$

$$q(u) = 0 \quad (4.31)$$

$$u_i \leq u \leq u_s \quad (4.32)$$

Para o problema em questão, a super-elipse precisa aproximar um determinado conjunto de dados aqusitados no veículo. Sabendo que um piloto amador é capaz de atingir o limite de desempenho dinâmico do veículo, embora de modo não consistente durante as voltas no circuito, o conjunto de dados de referência para o problema de otimização será coletado nos trechos em que o piloto conseguiu alcançar o desempenho máximo do veículo, conforme a Figura 4.19:



Figura 4.19: Dados para a construção do modelo super-elíptico: velocidade, acelerações lateral e longitudinal nos trechos de máxima performance.

Na Figura 4.19 os dados de aceleração lateral foram espelhados e isso é feito para facilitar o processo de otimização. Sabendo que o veículo possui configuração simétrica e que a Equação 4.26 é simétrica em relação ao eixo das ordenadas, espelhar os dados aqusitados melhora o resultado obtido no problema de otimização.

Com os dados da Figura 4.19, a função objetivo $Fobj$ é definida como:

$$Fobj(u, \dot{x}_{aq}, \ddot{y}_{aq}) = |\ddot{x}_{aq}(\dot{x}_{aq}) - \ddot{x}(u, \dot{x}_{aq}, \ddot{y}_{aq})| \quad (4.33)$$

Onde \dot{x}_{aq} corresponde à velocidade aqusitada, \ddot{x}_{aq} corresponde à aceleração longitudinal aqusitada e \ddot{y}_{aq} corresponde à aceleração lateral aqusitada.

Os parâmetros a serem determinados para a construção da super-elipse são: C_l , μ_{lat} , C_d , μ_{lgt} , n_a , n_b , m_a e m_b .

Os coeficientes C_l e C_d entram como variáveis de decisão para o problema de otimização quando não se conhece os dados de projeto do veículo, neste caso, estes coeficientes devem estar dentro da faixa típica para um carro de competição de turismo: entre -0.23 e -0.15 para o C_l e entre 0.35 e 0.43 para o C_d (MCBEATH, 2015).

Os parâmetros m_a , m_b , n_a e n_b determinam a forma da super-elipse, valores menores que 1 geram quadrantes côncavos que não correspondem ao comportamento real do veículo.

Os limites para os parâmetros μ_{lat} e μ_{lgt} não possuem valores específicos, devem ser estabelecidos com base nos dados aqusitados, de forma coerente.

Assim, o vetor de variáveis de decisão e os limites são dados por:

$$u = (C_l \ \mu_{lat} \ C_d \ \mu_{lgt} \ n_a \ n_b \ m_a \ m_b)^T \quad (4.34)$$

$$u_i = (-0.23 \ 1.0 \ 0.35 \ 1.0 \ 1.0 \ 1.0 \ 1.0 \ 1.0)^T \quad (4.35)$$

$$u_s = (-0.15 \ 1.5 \ 0.43 \ 1.5 \ 20 \ 20 \ 20 \ 20)^T \quad (4.36)$$

Para garantir que o resultado da otimização esteja coerente com os dados aquisitados, o limite de aceleração lateral calculado deve ser, no mínimo, igual ao limite aquisitado:

$$g(u, \dot{x}_{aq}, \ddot{y}_{aq}) = |\ddot{y}_{aq}(\dot{x}_{aq})| - |\ddot{y}_{limite}(\dot{x}_{aq})| \leq 0 \quad (4.37)$$

O problema de otimização para determinar os parâmetros de construção da super-elipse é dado por:

$$\min_u Fobj(u, \dot{x}_{aq}, \ddot{y}_{aq}) = |\ddot{x}_{aq}(\dot{x}_{aq}) - \ddot{x}(u, \dot{x}_{aq}, \ddot{y}_{aq})| \quad (4.38)$$

Sujeito a:

$$g(u, \dot{x}_{aq}, \ddot{y}_{aq}) = |\ddot{y}_{aq}(\dot{x}_{aq})| - |\ddot{y}_{limite}(\dot{x}_{aq})| \leq 0 \quad (4.39)$$

$$u_i \leq u \leq u_s \quad (4.40)$$

- Passo 4 – Verificar o resultado.

Solucionando problema 4.38 via Região de Confiança, os parâmetros da super-elipse são:

$$u = (-0.15 \ 1.40 \ 0.43 \ 1.00 \ 11.93 \ 1.00 \ 2.31 \ 1.00)^T \quad (4.41)$$

A Figura 4.20 apresenta os dados calculados pelo modelo super-elíptico em função dos dados aquisitados no veículo de competição ($\ddot{x}(u, \dot{x}_{aq}, \ddot{y}_{aq})$) em comparação aos dados aquisitados nos trechos em que o piloto conseguiu alcançar o desempenho máximo do veículo.

A Figura 4.21 apresenta os dados calculados pelo modelo super-elíptico em função dos dados aquisitados no veículo de competição ($\ddot{x}(u, \dot{x}_{aq}, \ddot{y}_{aq})$) em comparação aos dados aquisitados para a melhor volta completa no circuito.

Os parâmetros de construção da super-elipse (Equação 4.41) modelam o comportamento médio do veículo ao longo do circuito (ver áreas destacadas na Figura 4.21), enquanto o modelo elíptico superestima a dinâmica do veículo (ver Figura 4.9).

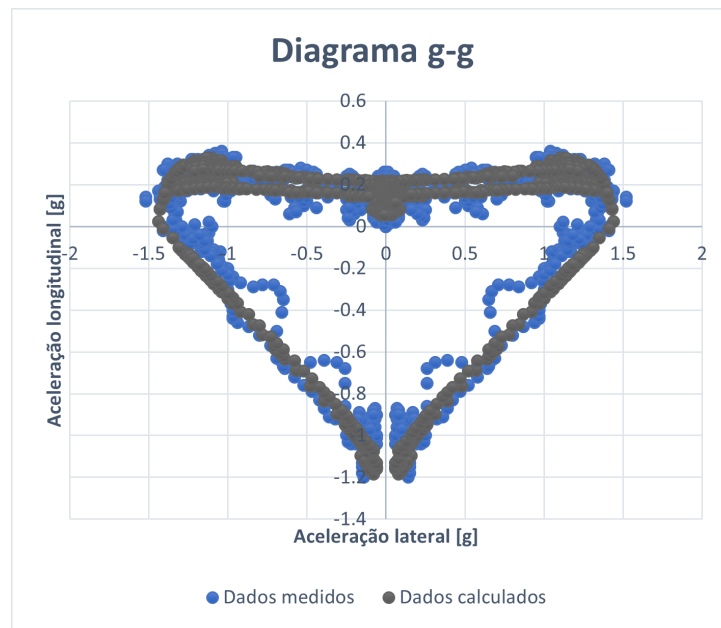


Figura 4.20: Dados calculados pelo modelo super-elíptico comparados aos dados adquiridos no veículo de competição nos trechos de máxima performance.

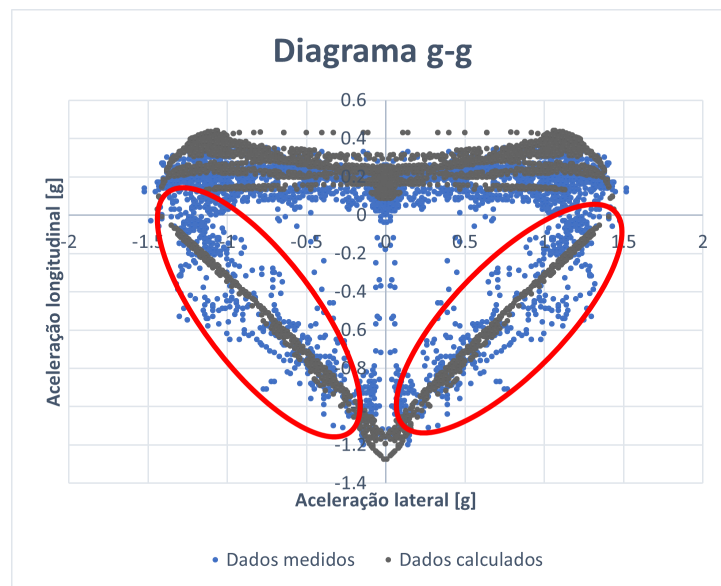


Figura 4.21: Dados calculados pelo modelo super-elíptico comparados aos dados adquiridos no veículo de competição para volta completa (aceleração lateral espelhada).

4.4 CONCLUSÃO DO CAPÍTULO

Neste capítulo, os principais problemas relativos à aquisição de dados para a modelagem veicular foram apresentados e, como alternativa para situações em que a modelagem com dois ou mais graus de liberdade é proibitiva, um novo método de modelagem com um

grau de liberdade foi proposto.

A modelagem super-elíptica de um grau de liberdade é incapaz de prever o comportamento do veículo com base em dados de projeto, entretanto, a metodologia proposta utiliza o mínimo possível de dados coletados de um veículo de competição e tem por objetivo ser aplicada no auxílio ao desenvolvimento de pilotos amadores, aplicação na qual a disponibilidade de dados é insuficiente para a modelagem do veículo com dois ou mais graus de liberdade e o método apresentado apresenta benefício objetivo.

O benefício principal da ferramenta associada a metodologia apresentada é modelar, com base em pequena quantidade de dados, o comportamento geral do par piloto/veículo em competições amadoras. Pilotos amadores, embora capazes de atingir a capacidade máxima de operação do veículo, não o fazem de forma consistente ao longo das voltas em um circuito, o modelo desenvolvido utiliza os dados adquiridos do veículo para gerar uma “volta virtual” consistente, para servir de base de referência para o aprimoramento do par piloto/veículo.

Neste capítulo será apresentado o novo método para otimização de traçados.

FORMULAÇÃO DO PROBLEMA DE OTIMIZAÇÃO

A predição do traçado ótimo envolve a solução de um problema de otimização no qual os pontos de controle usados para a interpolação do traçado via spline cúbica são manipulados pelo otimizador para determinar o traçado no qual o tempo de volta é mínimo.

O problema de otimização na formulação padrão é dado por:

$$\min_u Fob(u) = \sum \frac{2 \cdot s_i(u)}{\dot{x}_i(u) + \dot{x}_{i-1}(u)} \quad (5.1)$$

Sujeito a:

$$u_i \leq u \leq u_s \quad (5.2)$$

A função objetivo $Fob(u)$ é a integração numérica da distância s_i em função da velocidade (tempo de volta), sendo s_i a distância linear entre dois pontos discretos consecutivos $\overline{p_{i-1} p_i}$. A distância s_i não pode ser integrada em função da spline por curvas de Bézier por se tratar de uma integral elíptica, daí o uso de aproximação linear entre os pontos interpolados que exige uma malha de interpolação adequada (menos de 1m entre dois pontos consecutivos, para erros na terceira casa decimal). \dot{x}_i é determinada via Equação 4.27.

Como foi apresentado no Capítulo 3, a interpolação do traçado é realizada via pontos de controle distribuídos de forma homogênea ao longo da pista (ver Figura 3.9). A homogeneidade na distribuição de pontos é uma necessidade em função das características de interpolação da spline por curvas de Bézier. Os pontos interpolados são distribuídos em função da interpolação linear dos pontos de Bézier (ver Figura 3.4), de modo que uma distribuição irregular dos pontos de controle leva a uma distribuição irregular dos pontos interpolados. Dado que os pontos interpolados correspondem a discretização do problema no domínio do espaço, é necessário que estes estejam distribuídos o mais homoganeamente possível para garantir uma malha adequada.

O vetor u de variáveis de decisão contém os parâmetros de interpolação linear dos pares de coordenadas de controle mostrados na Figura 3.9 que dão origem aos pontos de

controle da spline de interpolação, assim o problema 5.1 possui $j - 1$ graus de liberdade para um vetor u de j parâmetros t (parâmetro de interpolação linear entre as coordenadas de controle do traçado). Como a redução dos graus de liberdade corresponde a redução do tempo de solução do problema (EDGAR; HIMMELBLAU, 2001) e a distribuição homogênea de coordenadas de controle é mandatória para uma malha discreta adequada, os pontos de solução trivial serão eliminados do problema bloqueando os limites do vetor u .

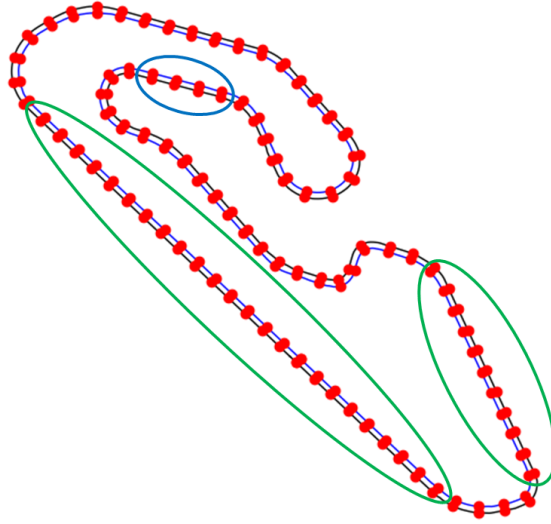


Figura 5.1: Coordenadas de controle para geração do traçado para o circuito de Goiânia. Os círculos destacam os pontos de solução trivial.

A Figura 5.1 apresenta os pontos de solução trivial, note que trechos de reta entre duas curvas consecutivas para a mesma direção são trechos em que o piloto mantém uma trajetória mais “aberta”. Para os pontos destacados em verde a solução trivial é $t_n = 1$ e, para os pontos destacados em azul, a solução trivial é $t_n = 0$. Assim o problema 5.1 está sujeito a:

$$u = \langle t_n \rangle_j^T, \quad t = u_i, \dots, u_s \quad n = 0, \dots, j - 1 \quad (5.3)$$

$$u_i = \begin{cases} 0, & \text{se } PO \\ 0, & \text{se } PA \\ 1, & \text{se } PV \end{cases} \quad (5.4)$$

$$u_s = \begin{cases} 1, & \text{se } PO \\ 0, & \text{se } PA \\ 1, & \text{se } PV \end{cases} \quad (5.5)$$

Onde PA corresponde aos pontos triviais de controle do tipo destacado pelo círculo azul na Figura 5.1, PV corresponde aos pontos triviais de controle do tipo destacado

pelos círculos verdes e PO corresponde aos pontos ordinários (todos os outros pontos de controle). t corresponde ao parâmetro de interpolação linear entre as coordenadas de controle internas (linha azul na Figura 5.1) e externas (linha preta na Figura 5.1) que formam os j pares de coordenadas de controle que dão origem aos j pontos P de controle da spline de interpolação.

A solução de $Fob(u)$ envolve a simulação do veículo em pista via modelo, neste caso o modelo super-elíptico desenvolvido no Capítulo 4, de modo que o otimizador não pode usar a própria função objetivo para a solução do problema via métodos determinísticos. Somente após a simulação a solução de $Fob(u)$ estará disponível para avaliação, sendo assim, métodos probabilísticos e métodos livre de derivadas são mais adequados para solucionar este tipo de problema.

Dois métodos probabilísticos serão analisados: algoritmos genéticos e evolução diferencial. Assim como dois métodos livres de derivadas: Nelder-Mead e Região de Confiança. Um quinto método de solução foi proposto (Gradiente Duplo) que também será avaliado. Este novo método foi inspirado na estrutura de redes-neurais e visa extrair o benefício do uso de gradientes no processo de solução de problemas de otimização: a velocidade. O tempo de solução é fator determinante na aplicabilidade do método, visto que sessões de treino em eventos automobilísticos possuem janela de tempo limitada.

5.1 MÉTODOS PROBABILÍSTICOS

Os métodos de otimização baseados nos algoritmos probabilísticos usam somente a avaliação da função objetivo e introduzem no processo de otimização parâmetros estocásticos, não exigindo que a função objetivo seja contínua e diferenciável. Estes algoritmos conseguem operar adequadamente, tanto com parâmetros contínuos quanto com parâmetros discretos, ou mesmo com a combinação deles. Diferente dos métodos determinísticos, os métodos probabilísticos não têm restrição quanto ao ponto de partida no espaço de busca da solução – nos métodos determinísticos o ponto de partida pode conduzir a um mínimo local –, entretanto, o tempo de processamento ainda é uma desvantagem para estes métodos (BASTOS, 2004).

Alguns métodos utilizados para otimização via algoritmos probabilísticos são conhecidos como heurísticas. Define-se heurística como sendo uma técnica que procura soluções próximas da otimalidade sem, no entanto, estar capacitada a garantir a otimalidade. A principal característica dos métodos heurísticos é a flexibilidade, contudo, possuem dificuldade de escapar de mínimos locais. Para melhorar estes métodos muitas pesquisas foram realizadas nas últimas décadas para solucionar a dificuldade de escape dos mínimos locais dando origem às meta-heurísticas (CHAVES, 2005).

As meta-heurísticas são estruturas algorítmicas gerais que podem ser aplicadas a diferentes problemas de otimização, sem que seja necessário grandes modificações para adaptá-la a um problema específico (BRAGA, 2007). Algoritmos Genéticos e algoritmos de Evolução Diferencial são exemplos de técnicas meta-heurísticas de otimização.

5.1.1 Algoritmo genético

O professor John Holland propôs em 1975 uma classe atraente de modelos computacionais, chamados Algoritmos Genéticos (Genetic Algorithms - GA), que imitam o processo de evolução biológica para resolver problemas em um amplo domínio. O GA possui três principais aplicações: pesquisa inteligente, otimização e aprendizado de máquina. Atualmente, o GA é usado juntamente com redes neurais e lógica difusa para resolver problemas mais complexos. Devido ao uso conjunto em muitos problemas, estas associações são frequentemente referidas por um nome genérico: "soft-computing" (KONAR, 1999).

Um algoritmo genético opera seguindo um simples ciclo de estágios (ver Figura 5.2) (FILHO; TRELEAVEN; ALIPPI, 1994):

- Criação de uma população de indivíduos
- Avaliação de cada indivíduo
- Seleção do melhor indivíduo
- Manipulação genética para a criação de uma nova população de indivíduos.

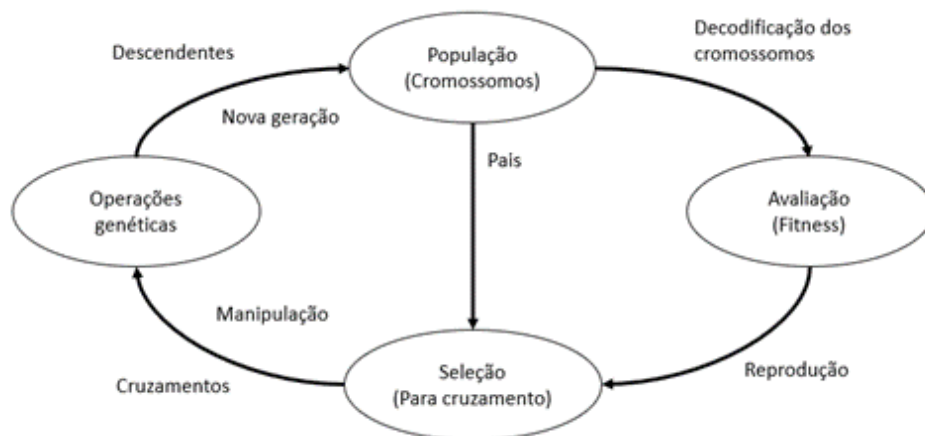


Figura 5.2: Ciclo de um algoritmo genético. Adaptado de: FILHO; TRELEAVEN; ALIPPI (1994)

5.1.2 Evolução Diferencial

Evolução Diferencial é um método de busca direta paralela que utiliza conjuntos de vetores de n parâmetros para formar a população de uma geração. A população inicial é construída de forma aleatória, cobrindo todo espaço de busca. O algoritmo gera novos vetores de parâmetros a partir de três vetores iniciais, sendo o resultado da diferença ponderada de dois dos vetores escolhidos adicionada ao terceiro. Esse procedimento é denominado de mutação. Após a mutação os vetores passam por um procedimento de

cruzamento para aumentar a diversidade de vetores e, em seguida, por um processo de seleção para manter os melhores resultados (ver Figura 5.3) (STORN; PRICE, 1997).

A depender do problema a ser otimizado, a função de avaliação pode exigir de minutos a horas para ser analisada. Uma vez que os algoritmos de Evolução Diferencial utilizam uma população de vetores, é possível avaliar estes vetores separadamente em processos paralelos. A capacidade de paralelizar a função de avaliação em diferentes núcleos de processamento ou threads ajuda a reduzir o tempo de execução da otimização (MÓR, 2016).

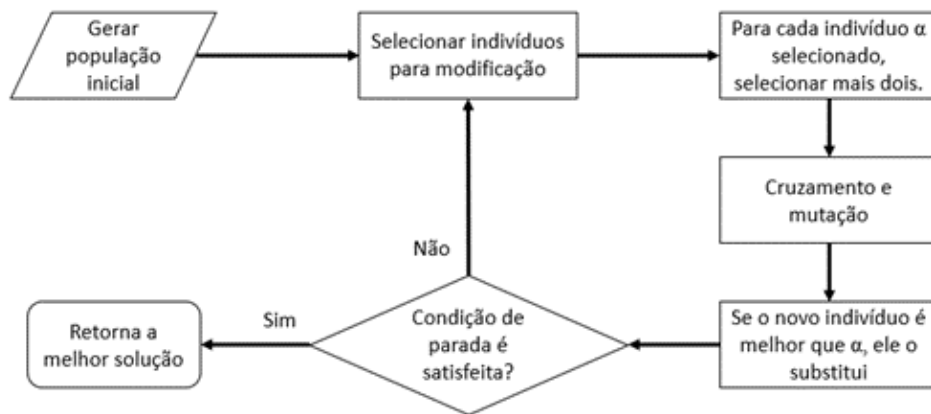


Figura 5.3: Funcionamento do algoritmo de Evolução Diferencial. Adaptado de MELO (2009)

5.2 MÉTODOS LIVRES DE DERIVADAS

Métodos de otimização livres de derivadas podem ser classificados como métodos de busca direta ou métodos baseados em modelo. De forma geral, os métodos de busca direta – Nelder-Mead, por exemplo – utilizam a comparação direta de valores da função objetivo para determinar uma nova iteração. Já os métodos baseados em modelo – Região de Confiança, por exemplo – utilizam os valores da função objetivo para construir modelos para substituir a própria função e, se necessário, as restrições, que são otimizados para obter novas iterações (GROSS, 2022).

5.2.1 Nelder-Mead

Em 1962 SPENDLEY; HEXT; HIMSWORTH (1962) propõem uma técnica de otimização empírica, na qual uma sequência de objetos simplex irregulares é usada para otimização de uma função sem a necessidade de avaliar a primeira derivada. Um simplex é uma figura geométrica de n dimensões que forma uma região convexa com $n + 1$ vértices (GAO; HAN, 2012). Para um problema bidimensional o simplex é formado por três vértices e, por tanto, tem a forma de um triângulo.

O método simplex é aplicável na otimização de problemas de múltiplas variáveis. Contudo, o método proposto por SPENDLEY; HEXT; HIMSWORTH (1962) é relativamente

“rígido”, ou seja, os passos usados para variar os fatores são fixos (NELDER; MEAD, 1965). Por exemplo, para um problema bidimensional o simplex é um triângulo cuja distância entre os vértices é determinada por um fator fixo predeterminado, o algoritmo conseguirá apenas rebater o simplex na direção de melhor resultado, de modo que o novo simplex será uma versão espelhada do anterior, conforme pode ser observado na Figura 5.4.

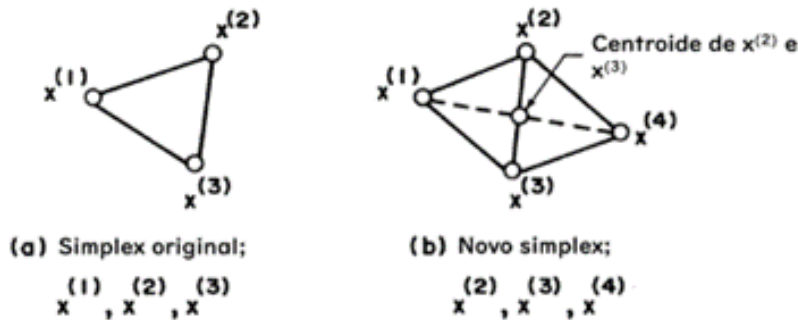


Figura 5.4: Construção de um novo simplex. Adaptado de RAVINDRAN; RAGSDELL; REKLAITIS (2006)

NELDER; MEAD (1965) propõem, então, um algoritmo simplex capaz de contrair ou expandir durante o processo de rebatimento (ver Figura 5.5). Eles observaram que, uma vez determinado o simplex inicial, não há incentivo para a permanência deste como uma figura regular ao longo do processo de otimização, sendo assim eles adicionaram mais dois fatores ao algoritmo, totalizando três: reflexão (α), contração (β) e expansão (γ) (RAVINDRAN; RAGSDELL; REKLAITIS, 2006).

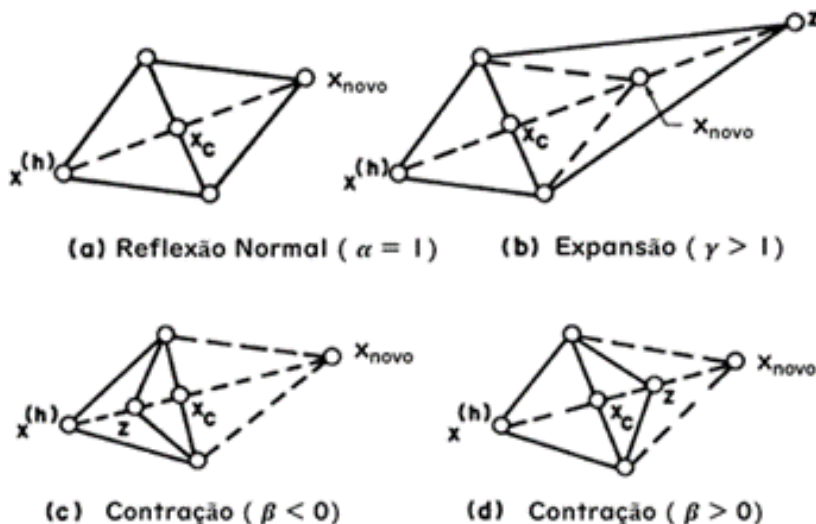


Figura 5.5: Contração e expansão de um simplex. Adaptado de RAVINDRAN; RAGSDELL; REKLAITIS (2006)

Segundo GAO; HAN (2012), o método Nelder-Mead padrão é eficiente para problemas de poucas dimensões, entretanto, para problemas de grandes dimensões ele perde eficiência e pode falhar em convergir para o ponto crítico da função objetivo. Para o Nelder-Mead padrão é recomendado que os fatores α, β e γ sejam: 1, 0.5 e 2, respectivamente (RAVINDRAN; RAGSDELL; REKLAITIS, 2006). Visando melhorar o desempenho do algoritmo para problemas de grandes dimensões, GAO; HAN (2012) propuseram que estes fatores sejam vinculados à dimensão do problema, sendo por tanto, adaptativos.

5.2.2 Região de Confiança

O método de otimização de Nelder-Mead é um tipo de método de busca direta que implementa uma estratégia de exploração em linha, ou seja, o algoritmo busca pela direção na qual a função tende ao ótimo e determina um novo ponto de exploração a partir desta direção, seguindo uma linha de busca. Para os algoritmos de região de confiança a abordagem difere um pouco, o algoritmo constrói uma aproximação “confiável” da função objetivo para uma região específica da função e, para esta pequena região, determina o ponto ótimo. O ponto ótimo de uma iteração do algoritmo de região de confiança determina o centro no qual será construída a nova região de confiança e o processo se repete (ver Figura 5.6) (RAVINDRAN; RAGSDELL; REKLAITIS, 2006).

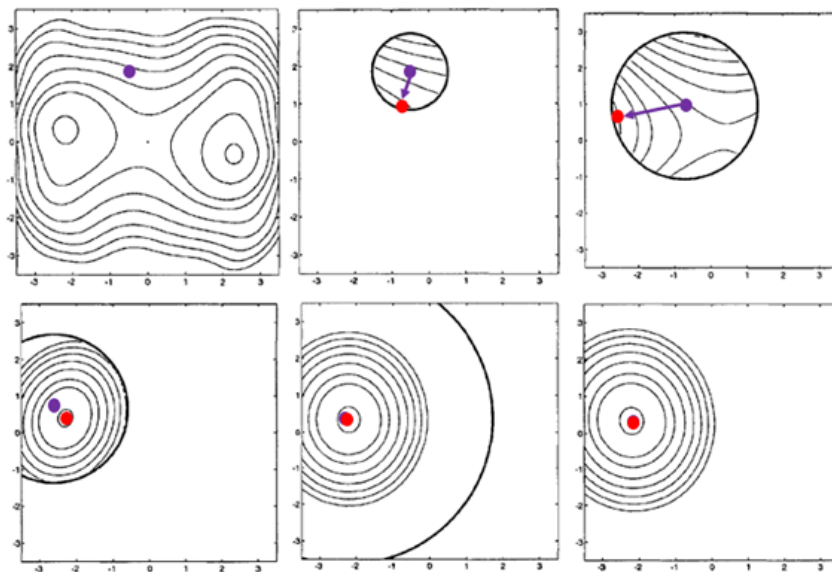


Figura 5.6: Otimização por região de confiança. Adaptado de CONN (2000)

O que se define como região de confiança é um modelo baseado na função objetivo do problema. Este modelo é construído para simplificar o processo de otimização da função original, uma vez que, por representar apenas uma região específica da função objetivo, é mais adequado à aplicação de técnicas clássicas de otimização com derivadas (GIULIANI, 2013).

Os métodos de otimização por região de confiança foram originalmente elaborados para solução de problemas não lineares minimizando um modelo quadrático para os

mínimos quadrados da função objetivo (CONN, 2000). O modelo quadrático é o modelo não-linear mais simples para modelagem de funções, apesar disso, outros modelos podem ser usados para a construção dos subproblemas de otimização das regiões de confiança, como o modelo cônico (YUAN, 2015).

Para a construção dos subproblemas e aplicação dos métodos clássicos de otimização com derivadas é necessário que a função objetivo seja uma função contínua avaliável, contudo, os algoritmos de região de confiança podem ser adaptados para a solução de problemas sem o uso de derivadas, usando interpolação ao invés de expansões de Taylor (YUAN, 2015). Também é possível substituir a função objetivo do problema de otimização por um simulador, neste caso, sempre que a função objetivo precisa ser avaliada, um experimento é executado no simulador e o resultado é usado para construção do subproblema via interpolação (GIULIANI, 2013).

5.3 MÉTODO DO GRADIENTE DUPLO

Redes neurais artificiais são equivalentes elétricos das redes neurais biológicas nas quais foram inspiradas. Os neurônios são representados por funções lineares enquanto as sinapses são representadas por funções de ativação. As funções de ativação funcionam como inibidoras, elas limitam a amplitude do sinal que será processado em uma célula (KONAR, 1999).

O método que será desenvolvido nesta seção foi inspirado em uma topologia particular de redes neurais artificiais: a rede de camada única com retropropagação do erro, conforme Figura 5.7.

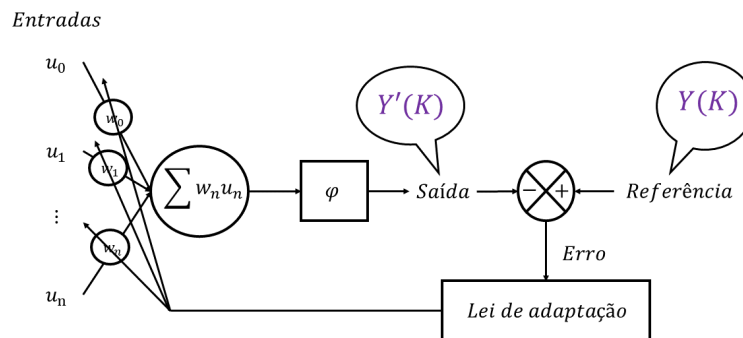


Figura 5.7: Topologia de uma rede neural artificial com retropropagação do erro

Na forma matricial, a topologia da Figura 5.7 é representado da seguinte forma:

$$Y'(k) = \varphi(W(k) \cdot U(k)) \quad (5.6)$$

Onde $Y'(k)$ é a saída da rede na iteração k . W e U são as matrizes de pesos e entradas respectivamente e φ é a função de ativação. A lei de adaptação é dada pela

regra do gradiente:

$$e = Y - Y' \quad (5.7)$$

$$\frac{\partial e}{\partial W} = -\frac{\partial Y'}{\partial W} = -\frac{\partial \varphi(W \cdot U)}{\partial W} \quad (5.8)$$

$$W(k+1) = W(k) - \alpha \cdot \frac{\partial e}{\partial W} \quad (5.9)$$

Para que os pesos W da rede neural se adaptem, uma série de dados de entrada U e suas respectivas saídas Y de referência são apresentados à rede durante o processo de aprendizado. Para a solução do problema de otimização, apenas uma entrada está disponível e não existe referência para o erro, ainda assim, o método do gradiente duplo utiliza da topologia básica apresentada para construção do algoritmo.

Para aplicação na solução do problema de otimização específico deste trabalho (otimização de traçado via pontos de controle para interpolação por spline) a topologia da Figura 5.7 foi modificada conforme a Figura 5.8:

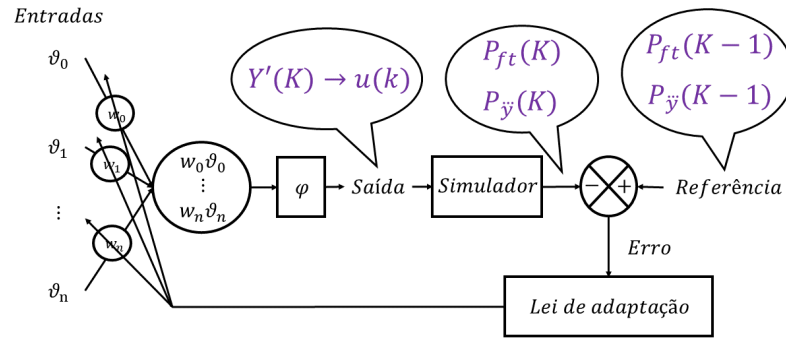


Figura 5.8: Topologia básica do método do gradiente duplo

Na Figura 5.8, o vetor de entradas ϑ (u na rede neural) passa a ser uma referência fixa imutável durante o processo de otimização e o vetor u passa a ser a saída da “rede” e entrada para o simulador executando o modelo super-elíptico do Capítulo 4, ou seja, passa a ser o vetor de variáveis de decisão do problema de otimização.

Na forma matricial:

$$u(k) = \varphi(W(k) * \vartheta(k)) \quad (5.10)$$

Em que $*$ indica o produto linha por linha das matrizes W e ϑ , logo u é uma matriz $n \times 1$. Para a construção da lei de adaptação três gradientes serão considerados:

$$\partial P_{ft}(k)_{[n \times 1]} = P_{ft}(k)_{[n \times 1]} - P_{ft}(k-1)_{[n \times 1]} \quad (5.11)$$

$$\partial u(k)_{[n \times 1]} = u(k)_{[n \times 1]} - u(k-1)_{[n \times 1]} \quad (5.12)$$

$$\partial P_{\ddot{y}}(k)_{[n \times 1]} = P_{\ddot{y}}(k)_{[n \times 1]} - P_{\ddot{y}}(k-1)_{[n \times 1]} \quad (5.13)$$

Onde P_{ft} , definida pela Equação 5.14, é uma função que contabiliza o ganho/perda de tempo entre os pontos de controle da spline que forma o traçado e $P_{\ddot{y}}$, definida pela Equação 5.15, é uma função que contabiliza a variação de aceleração lateral do veículo entre os pontos de controle da spline que forma o traçado.

$$P_{ft} = \begin{cases} \text{Fob}(u)^2 \cdot \sum_{i_{P(n-1)}}^{i_{P(n)}} \frac{2 \cdot s_i}{\dot{x}_i + \dot{x}_{i-1}} \cdot \sum_{i_{P(n)}}^{i_{P(n+1)}} \frac{2 \cdot s_i}{\dot{x}_i + \dot{x}_{i-1}}, & \text{se } n = 1, \dots, j-2 \\ \text{Fob}(u)^2 \cdot \sum_{i_{P(n-1)}}^{i_{P(n)}} \frac{2 \cdot s_i}{\dot{x}_i + \dot{x}_{i-1}} \cdot \sum_{i_{P(0)}}^{i_{P(1)}} \frac{2 \cdot s_i}{\dot{x}_i + \dot{x}_{i-1}}, & \text{se } n = j-1 \end{cases} \quad (5.14)$$

$$P_{\ddot{y}} = \begin{cases} \sum_{i_{P(n-1)}}^{i_{P(n)}} |\ddot{y}_i - \ddot{y}_{i-1}| + \sum_{i_{P(n)}}^{i_{P(n+1)}} |\ddot{y}_i - \ddot{y}_{i-1}|, & \text{se } n = 1, \dots, j-2 \\ \sum_{i_{P(n-1)}}^{i_{P(n)}} |\ddot{y}_i - \ddot{y}_{i-1}| + \sum_{i_{P(0)}}^{i_{P(1)}} |\ddot{y}_i - \ddot{y}_{i-1}|, & \text{se } n = j-1 \end{cases} \quad (5.15)$$

Assim, a lei de adaptação para o algoritmo é dada por:

$$W(k+1)_{[n \times 1]} = \begin{cases} W(k) - \alpha(k) * \text{sgn}(\partial P_{ft}(k)) * \text{sgn}(\partial u(k)), & \text{se } \kappa|_n \geq 0.002 \\ W(k) - \alpha(k) * \text{sgn}(\partial P_{\ddot{y}}(k)) * \text{sgn}(\partial u(k)), & \text{se } \kappa|_n < 0.002 \end{cases} \quad (5.16)$$

A atualização dos pesos do algoritmo, como mostra a Equação 5.16, depende apenas do sinal dos gradientes. O processo de otimização do traçado é realizado via modificação da localização dos pontos de controle da spline que forma o traçado (ver Figura 5.9), para isso, o valor dos gradientes não é importante e dificulta o controle sobre o grau de alteração do traçado entre as iterações no processo de otimização, de modo que apenas o sinal dos gradientes é necessário para a lei de adaptação.

Comparando-se as Equações 5.16 e 5.9, observa-se que o passo de atualização dos pesos na Equação 5.9 vai reduzindo conforme o erro em relação a referência diminui, já para Equação 5.16 esse passo é constante, devido à ausência de um parâmetro de erro. Para emular o processo de redução do passo de atualização dos pesos, algumas características da Equação 5.16 precisam ser analisadas:

- Os pesos são atualizados com base em $\partial P_{ft}|_n$ caso a curvatura do traçado no ponto P_n seja tal que ele possa ser considerado uma curva ($\kappa|_n \geq 0.002$). Para essa condição, pequenas alterações na posição dos pontos de controle podem gerar ganhos/perdas razoáveis no tempo de volta.

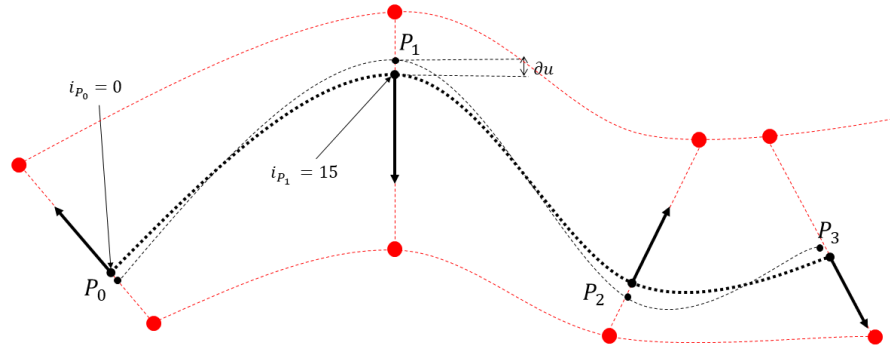


Figura 5.9: Atualização da posição dos pontos de controle do traçado durante o processo de otimização.

- Os pesos são atualizados com base em $\partial P \cdot \dot{y} \cdot |_n$ caso a curvatura do traçado no ponto P_n seja tal que ele possa ser considerado uma reta ($\kappa|_n < 0.002$). Para essa condição, pequenas alterações na posição dos pontos de controle não afetam efetivamente o tempo de volta, neste caso obtém-se melhores resultados priorizando a redução da variação de aceleração lateral \dot{y} .
- Os gradientes atuam apenas como “bússolas” para indicar a direção de atualização dos pesos. O passo de atualização é definido pela taxa de atualização α .
- O segundo termo da Equação 5.16 é sempre diferente de zero em todos os pontos exceto nos pontos de solução trivial. Quando a função se aproxima do ponto sub-ótimo ela estabiliza e oscila em torno do ponto indefinidamente.

Sabendo que α é o único parâmetro que altera o passo de atualização dos pesos e que ao aproximar de um sub-ótimo a Equação 5.1 deixa de progredir, a taxa de atualização α receberá o seguinte tratamento:

$$\alpha(k+1) = \begin{cases} \frac{\alpha(k)}{2}, & \text{se } \varsigma > 0 \text{ e } \varsigma \% nk = 0 \text{ e } \alpha > 0.0001 \\ \alpha(k), & \text{caso contrário} \end{cases} \quad (5.17)$$

Onde o parâmetro ς contabiliza quantas iterações k consecutivas a $Fob(u)$ não progride e $\%$ representa o resto da divisão ς/nk . O parâmetro nk indica quantas iterações sem progressão de resultado são permitidas. O limite de 0.0001 é estabelecido pois alterações no traçado a nível de décimos de milímetro são inexpressivas em relação ao tempo de volta.

Por fim, a função φ garante que $0 \leq u < 1$ via sigmoide modificada conforme Equação 5.18 (ver Figura 5.10):

$$\varphi(f) = \frac{1}{1 + e^{-15.637 \cdot f + 7.819}} \quad (5.18)$$

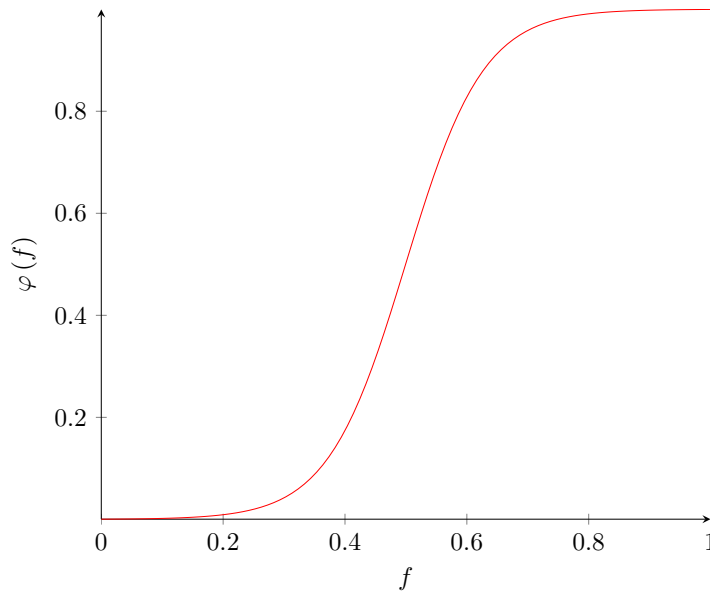


Figura 5.10: Função sigmoide modificada.

5.4 CONCLUSÃO DO CAPÍTULO

Neste capítulo foi apresentado o problema de otimização a ser solucionado para determinar o traçado ótimo de um veículo de competição, utilizando pontos de controle para manipular uma spline responsável por representar o traçado do veículo. Uma vez que a função objetivo não pode ser usada para solução do problema por métodos determinísticos, dois métodos probabilísticos e dois métodos livres de derivadas foram selecionados. Também foi feita uma apresentação dos conceitos básicos a respeito dos quatro métodos de solução de problemas de otimização selecionados.

Inspirado na topologia de redes neurais, um novo método de solução para o problema de otimização apresentado neste capítulo foi desenvolvido. Uma vez que a solução do problema envolve a manipulação direta dos pontos de controle de uma spline para a construção do traçado (ver Figura 5.9) e que estes pontos de controle são manipulados via vetor de parâmetros de interpolação linear que variam entre zero e um, a estrutura de produto matricial modulada por uma função de ativação das redes neurais seria conveniente para esta aplicação. De fato, a modificação da topologia da rede neural para deixar de atuar como um aproximador e passar a atuar como um otimizador, funcionou para o problema apresentado no início do capítulo.

A alteração na topologia modificou o papel das variáveis da rede neural da seguinte forma:

- As variáveis de entrada passaram a atuar como um vetor de referência. Para o

problema de otimização apresentado neste capítulo, o vetor de referência mais conveniente é a linha central do traçado.

- Em redes neurais, os pesos são adaptados para serem capazes de aproximar o resultado de uma série de problemas distintos com base em uma determinada rotina de aprendizado. É possível que o processo de treinamento de uma rede modifique os pesos de modo que a rede se especialize num problema específico, perdendo a sua função principal de aproximar resultados para problemas não apresentados durante o treinamento. As modificações aplicadas na topologia da rede foram realizadas no intuito de emular o fenômeno de especialização de uma rede neural, adaptando os pesos para que eles determinem o melhor traçado.
- O vetor de saída de uma rede neural é resultado do problema apresentado a ela. Na rede modificada, a saída da rede é apresentada ao simulador responsável por calcular o tempo de volta do veículo. O papel da rede passa a ser o de adaptar o vetor de variáveis de decisão do problema de otimização, ou seja, modificar o traçado da linha central da pista até que se alcance o menor tempo de volta possível.
- As funções de ativação das redes neurais limitam a amplitude do sinal a ser processado pelos neurônios. Para a rede modificada, a função de ativação cumpre o mesmo papel, mantendo o sinal de saída entre zero e um.
- A lei de adaptação da rede neural é construída em função do gradiente do erro, na rede modificada a lei de adaptação é construída com base nos gradientes das saídas do simulador (tempo de volta e aceleração lateral) e do gradiente do vetor de variáveis de decisão. A lei de adaptação utiliza duas funções, nas quais dois gradientes são aplicados, daí o nome do método proposto.

Os resultados das simulações realizadas para otimização do traçado utilizando os quatro métodos de otimização escolhidos e o método desenvolvido neste capítulo serão analisados entre si e comparados aos dados obtidos com piloto profissional no próximo capítulo.

Neste capítulo serão apresentados os resultados das simulações.

SIMULAÇÕES E RESULTADOS

No Capítulo 4, foi apresentada a modelagem super-elíptica para veículos de competição. Neste mesmo capítulo foram apresentadas a capacidade do modelo em representar o comportamento do veículo e suas limitações.

Para a otimização do traçado o modelo do veículo precisa ser avaliado em uma ferramenta de otimização. No Capítulo 5 foram apresentadas algumas ferramentas já disponíveis e uma nova ferramenta foi proposta. Os resultados das simulações realizadas serão analisados a seguir.

6.1 SIMULAÇÕES

Todas as simulações foram realizadas em Python 3.9. Os algoritmos para os métodos de otimização Nelder-Mead, Região de Confiança e Evolução Diferencial estão disponíveis na biblioteca Scipy. O Algoritmo Genético utiliza uma biblioteca específica, a `Geneticalgorithm` 1.0.2. O código Python para o método de Gradiente Duplo apresentado na Seção 5.3 está disponível no Apêndice C.

O equipamento utilizado para rodar as simulações possui as seguintes características: CPU 11th Gen Intel® Core™ i5-11600K @ 3.90GHz e 16GB de memória 3200MHz DDR5.

Os algoritmos de Região de Confiança, Nelder-Mead permitem a inserção de um vetor de “chute” inicial enquanto o método do Gradiente Duplo exige um vetor de referência para executar as simulações. Para estes três métodos o vetor de “chute” inicial/referência foi:

$$\vartheta = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.5, \\ 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 1.0, 1.0, 1.0, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 1.0, \\ 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.0, 0.0, 0.5, 0.5, 0.5, 0.5, 0.5, \\ 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, \\ 0.5, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 1.0, 1.0, \\ 1.0, 1.0, 1.0, 1.0, 1.0]$$

Tabela 6.1: Tempos de volta comparados.

Método	Tempo de simulação	Tempo de volta [s]	Diferença [s]
Evolução Diferencial	06:37:50	93.29	+0.37
Algoritmo Genético	19:55:22	96.72	+3.80
Gradiente Duplo	00:07:29	93.30	+0.38
Nelder-Mead	01:15:29	94.23	+1.31
Thrust Region	01:59:45	94.30	+1.38
Piloto 1	NA	92.92	0.00
Piloto 2	NA	93.20	+0.28

No vetor acima os pontos diferentes de 0.5 são os pontos de solução trivial bloqueados conforme proposto na Figura 5.1. O circuito virtualizado para as simulações foi o circuito de Goiânia, com 93 pares de coordenadas de controle conforme Figura 3.9.

O algoritmo de Evolução Diferencial usa como “chute” inicial um vetor aleatório, contudo, permite a adoção de um seed que foi fixado como 1. Já o Algoritmo Genético inicia o vetor aleatoriamente sem adoção de seed ou vetor de “chute” inicial.

Para reduzir a carga computacional as simulações foram realizadas com uma malha de um metro entre cada i ponto do traçado. Uma vez finalizado o processo de otimização, uma última iteração foi realizada para o refino do resultado, desta vez utilizando uma malha de aproximadamente 20cm entre os i pontos do traçado. O aumento de resolução de malha, embora não impacte no perfil do traçado, altera a precisão do cálculo de velocidade do veículo. O ganho em tempo de volta com o aumento de resolução é da ordem de 3 centésimos de segundo.

6.2 RESULTADOS

A Tabela 6.1 apresenta os resultados obtidos após as simulações utilizando os quatro métodos de otimização selecionados e o método do Gradiente-Duplo, desenvolvido e apresentado neste trabalho. O método proposto foi o único a solucionar o problema dentro de uma janela de minutos viável para aplicação sessões de treino de eventos competitivos. A diferença de tempo de volta entre simulador (que engloba o modelo super-elíptico e o MGD) e pilotos profissionais se deve ao modelo quase-estático utilizado nas análises calibrado para representar apenas o comportamento médio do veículo (ver Figura 4.21).

As figuras 6.1 a 6.4 apresentam as comparações de métricas entre o método proposto e os pilotos profissionais. Para uma comparação válida entre os dados apresentados nas figuras a seguir - com exceção da Figura 6.4, que apresenta a comparação entre diagramas g-g -, os dados foram pareados tomando-se uma referência fixa comum entre

eles, a margem interna da pista.

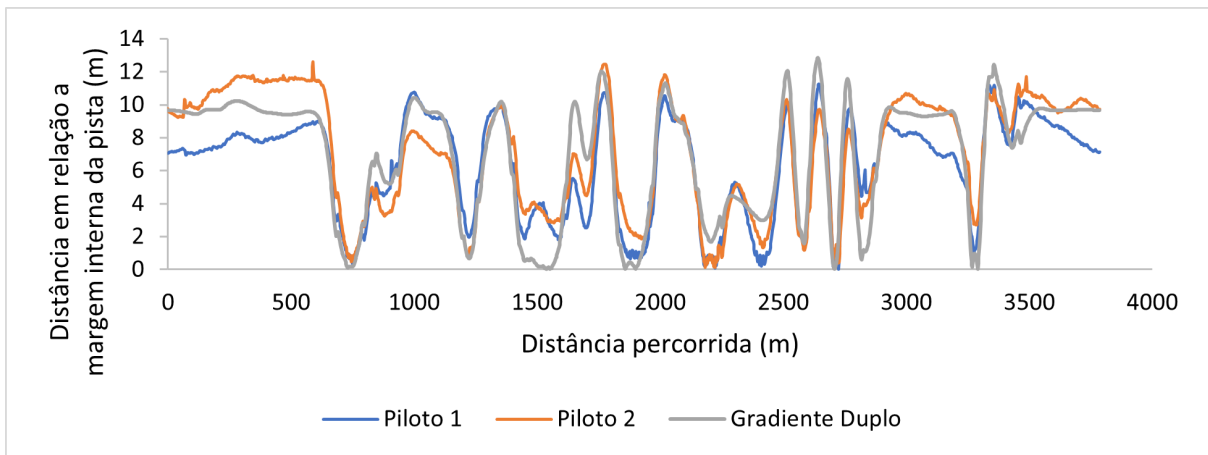


Figura 6.1: Posição do veículo ao longo da pista em relação a margem interna da pista.

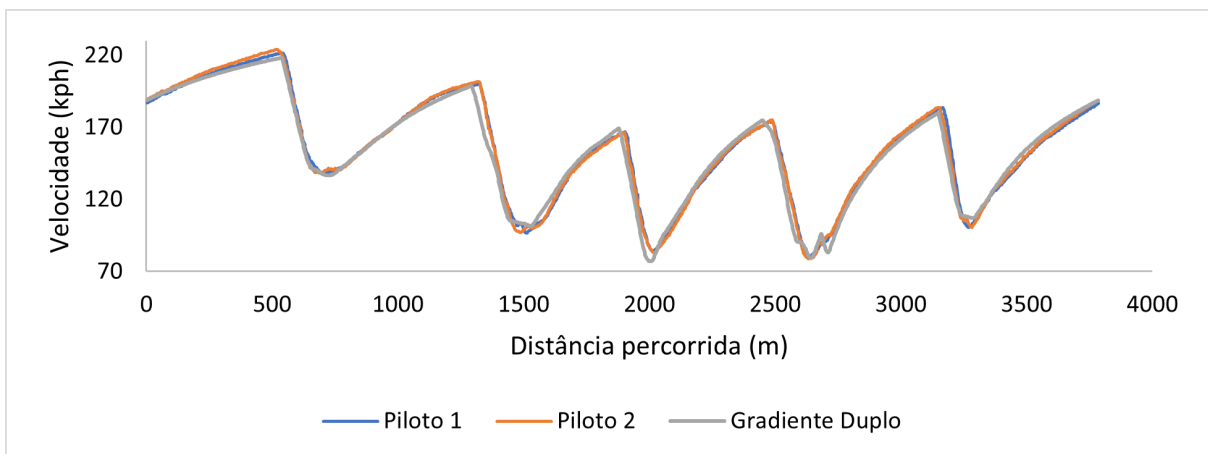


Figura 6.2: Velocidade do veículo ao longo da pista.

De modo geral, o simulador desenvolvido (modelo super-elíptico associado ao MGD para solução do problema de tempo mínimo de volta) foi capaz de prever com boa aproximação tanto a velocidade do veículo quanto a curvatura do traçado ao longo da pista (ver Figuras 6.2 e 6.3). O diagrama g-g da Figura 6.4 confirma, desta vez em uma simulação completa, o comportamento calibrado via metodologia apresentada na Seção 4.3.2.

A figura 6.5 apresenta as curvas do circuito misto de Goiânia. Essas curvas serão usadas para analisar o traçado previsto pelo simulador em comparação ao traçado realizado pelos dois pilotos profissionais de referência.

A Figura 6.1 mostra que o simulador tem a tendência de aproveitar ao máximo o espaço disponível da pista para construir o traçado. Esse comportamento é mais facilmente identificado nas Figuras 6.6 a 6.11. As curvas Um, Mergulho, Miolo, Bico de

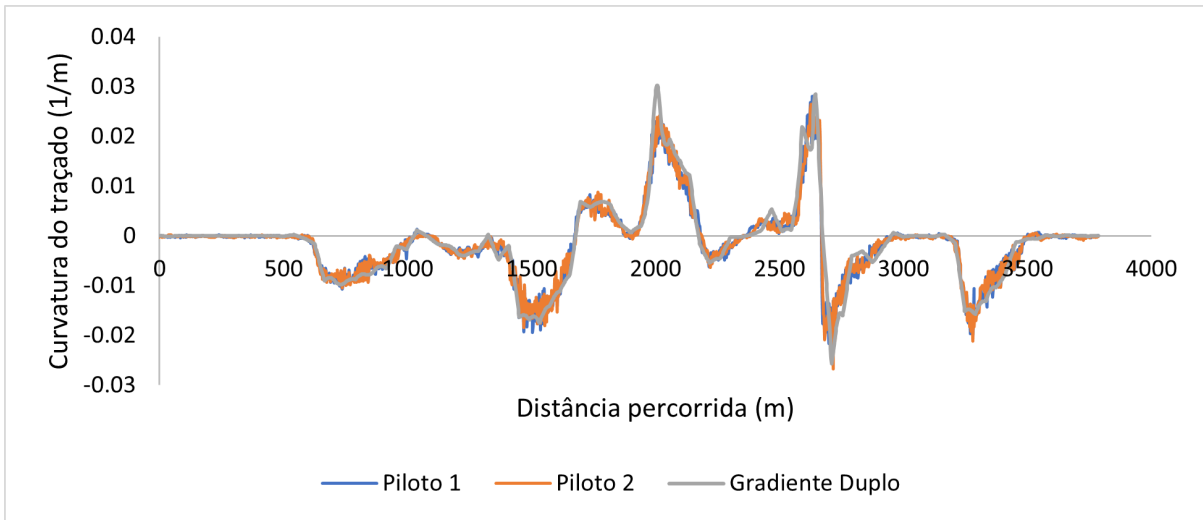


Figura 6.3: Curvatura do traçado executado ao longo da pista.

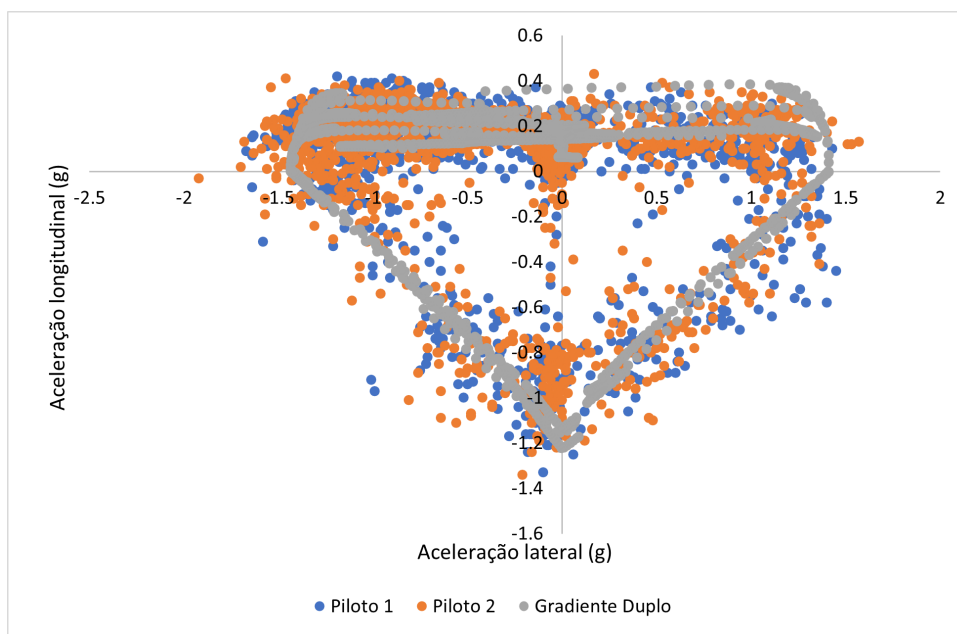


Figura 6.4: Diagrama g-g simulado em comparação aos obtidos com pilotos profissionais. O modelo veicular utilizado para o simulador foi calibrado para representar o comportamento médio do veículo.

Pato e Zero foram contornadas seguindo um mesmo padrão de comportamento, qual seja: aproximar à curva o mais aberto possível contornar o ápice o mais internamente possível e sair da curva reduzindo gradualmente a curvatura. Este padrão de abordagem também pode ser observado na Figura 6.3, a curvatura varia mais bruscamente da entrada da curva ao ápice do que do ápice à saída da curva. De modo geral, isso corresponde ao que os pilotos optaram por fazer e está relacionado às características dinâmicas do veículo.

Observando a Figura 6.4, nota-se que o veículo lida melhor com a aceleração e curva combinados (primeiro e segundo quadrante do diagrama g-g) do que com desaceleração e curva combinados.

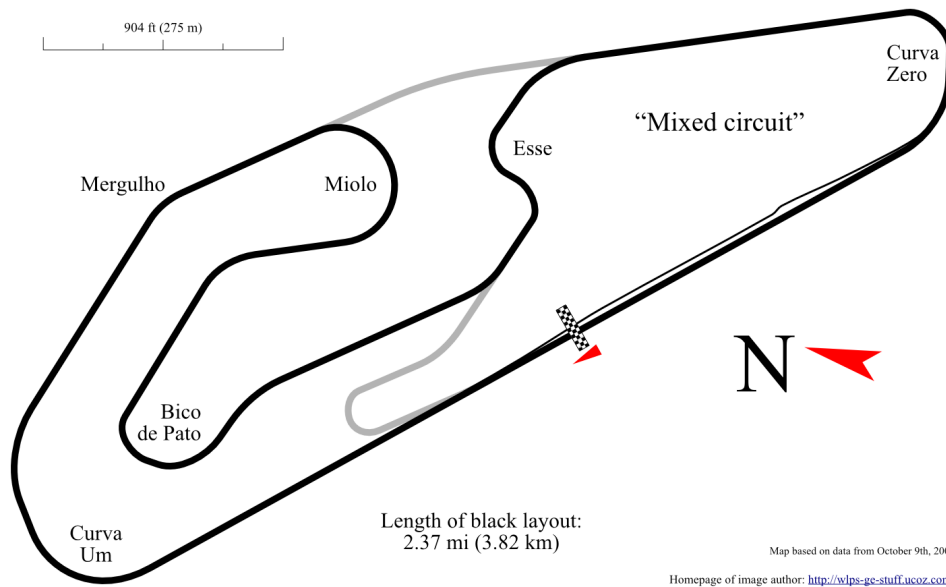


Figura 6.5: Curvas do circuito misto de Goiânia. Fonte: https://wlps-ge-stuff.ucoz.com/photo/autodromo_internacional_ayrton_senna_goiania/3-0-249

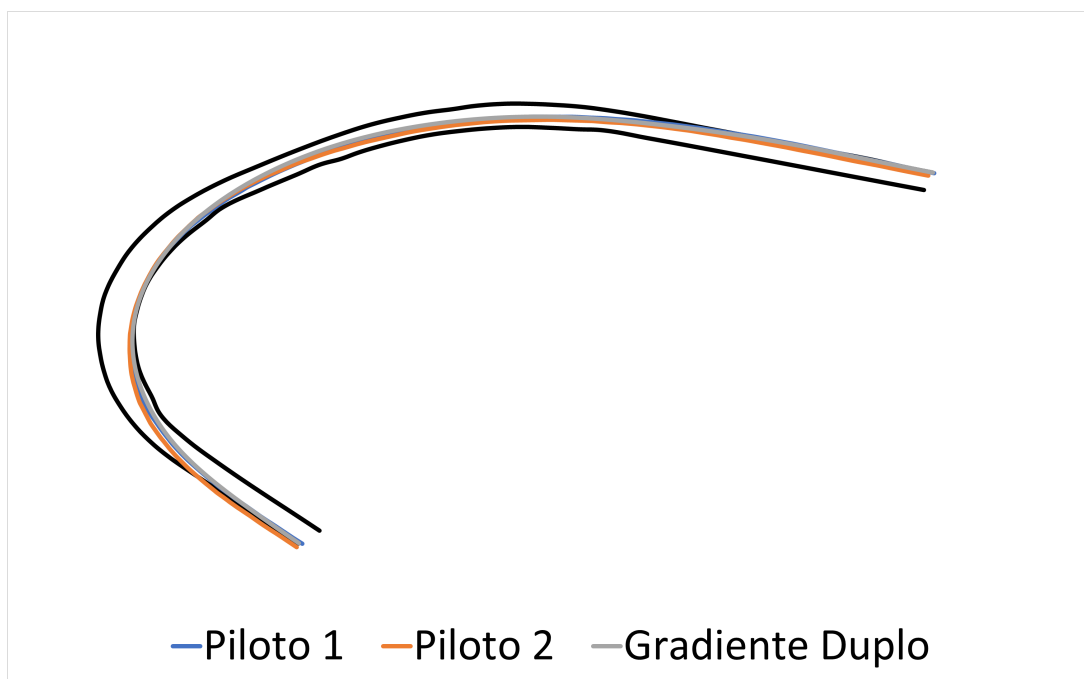


Figura 6.6: Curva Um do circuito misto de Goiânia.

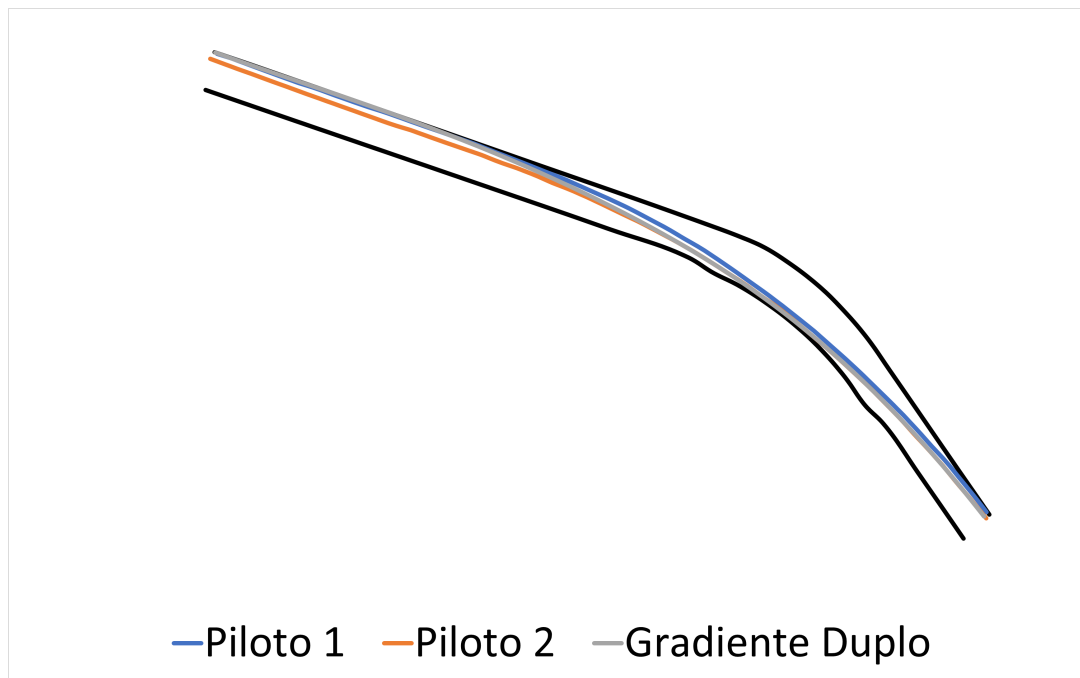


Figura 6.7: Curva do Mergulho do circuito misto de Goiânia.

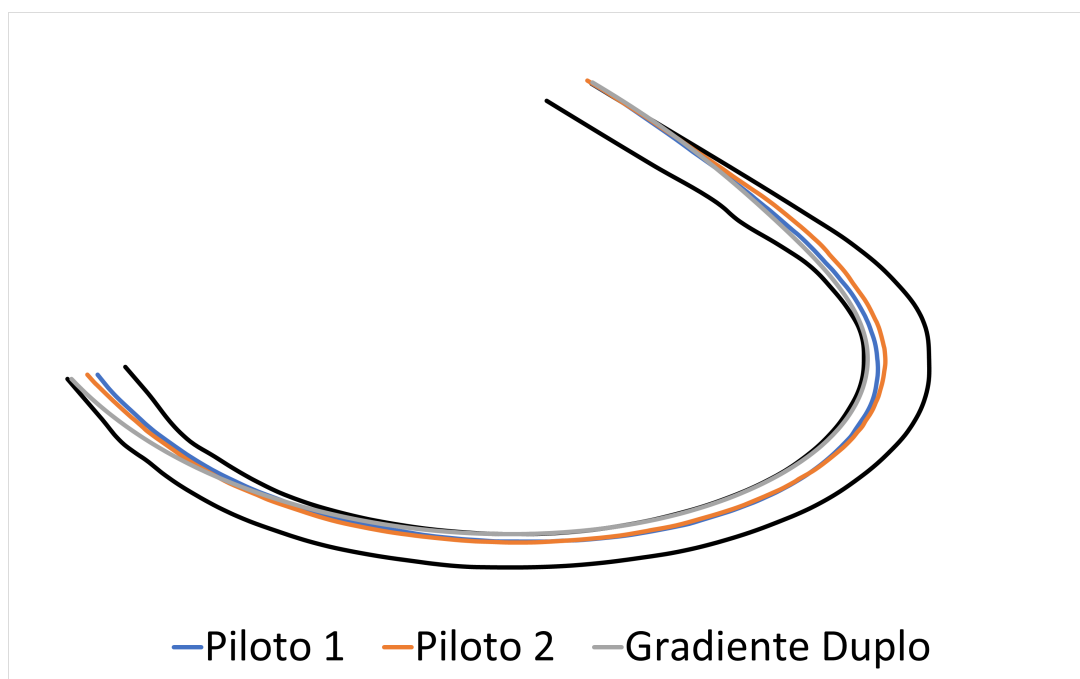


Figura 6.8: Curva do Miolo do circuito misto de Goiânia.

Embora o comportamento geral tenha sido similar entre o simulador e os pilotos, o posicionamento no circuito foi distinto. A diferença entre o posicionamento do ápice nas curvas Um, Mergulho, Bico de Pato e Zero corresponde ao esperado ao analisar o

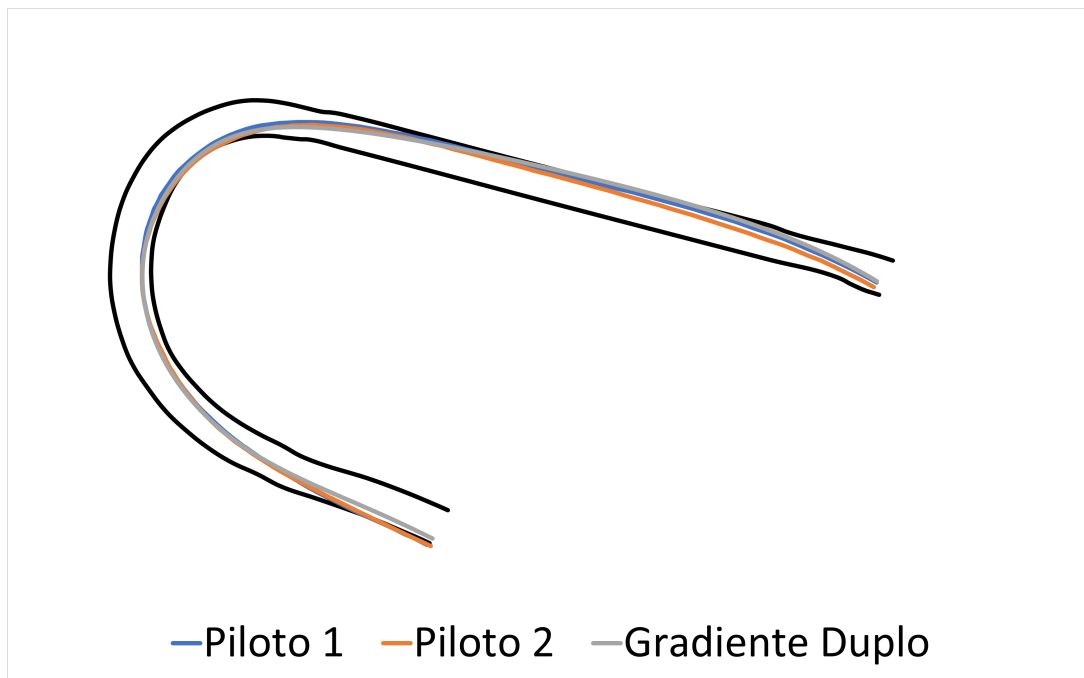


Figura 6.9: Curva do Bico de Pato do circuito misto de Goiânia.

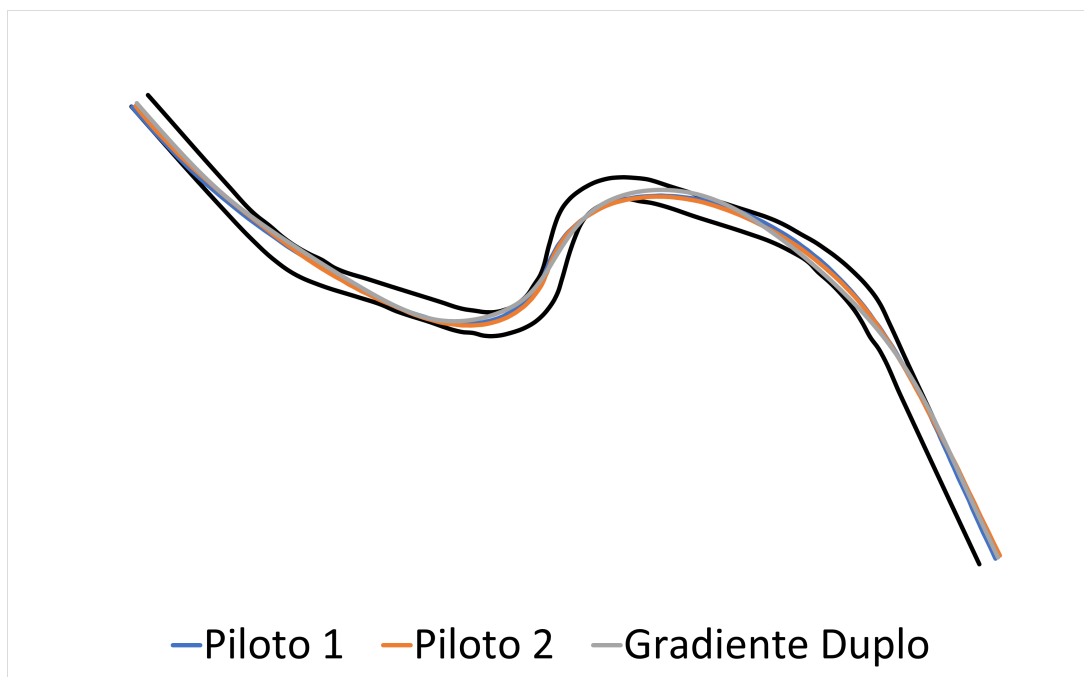


Figura 6.10: Curva do Esse do circuito misto de Goiânia.

trabalho de (KEGELMAN; HARBOTT; GERDES, 2017) apresentado na Seção 2.2, ou seja, o algoritmo encontrou uma técnica para abordar as curvas do circuito que minimiza o tempo de volta, mas que não necessariamente é idêntica as técnicas dos pilotos de

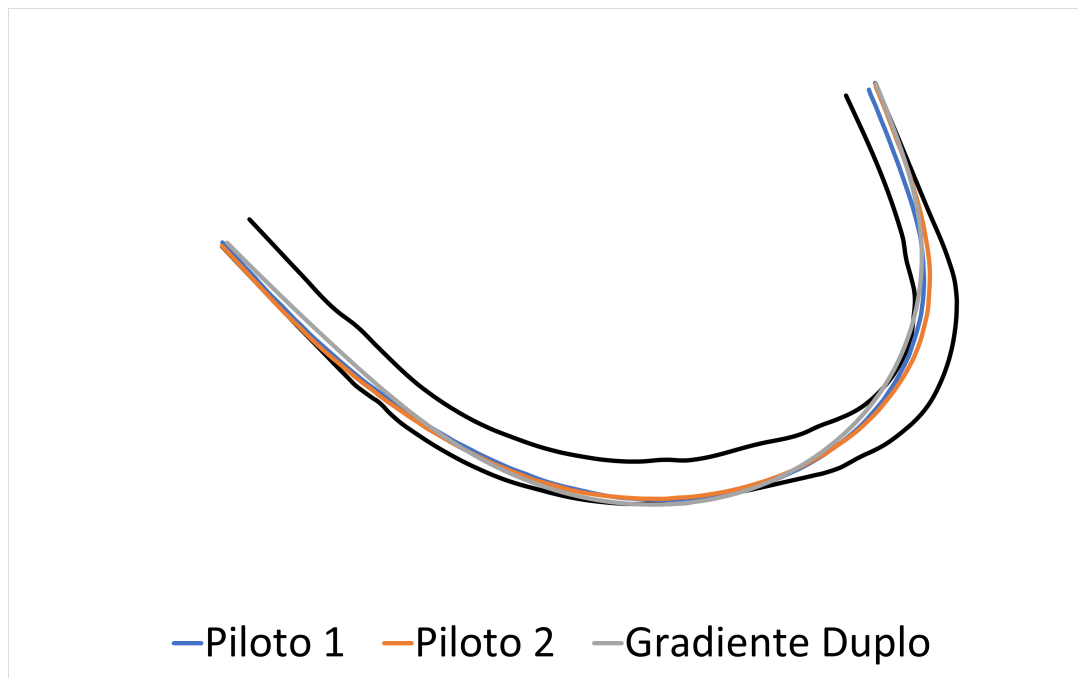


Figura 6.11: Curva Zero do circuito misto de Goiânia.

referência (que também não são idênticas entre si). Contudo, a Curva do Miolo apresenta uma abordagem excessivamente distinta. Possivelmente, a falta de informação sobre a qualidade da aderência da pista neste trecho tenha levado o algoritmo a entender que contornar a curva "atacando" a zebra interna e por tanto, reduzindo a distância percorrida, é mais benéfico que abrir um pouco mais a curva e contorná-la com mais velocidade (vide o perfil de velocidade próximo ao metro 1500 do circuito na Figura 6.2).

A Curva do Esse é na verdade uma sequência de quatro curvas consecutivas e de execução bastante técnica, uma vez a abordagem de uma curva interfere diretamente na curva seguinte. Ambos os pilotos optam pela redução da curvatura do traçado, permitindo carregar mais velocidade nas curvas, enquanto o simulador prevê uma abordagem um pouco mais agressiva visando a redução da distância percorrida.

6.3 CONCLUSÃO DO CAPÍTULO

O método de otimização proposto na Seção 5.3 soluciona o problema de otimização de traçado graficamente, atuando apenas sobre as coordenadas do traçado que o veículo deve percorrer. Apesar de utilizar apenas um modelo quase-estático para representar o veículo nas simulações, o otimizador foi capaz de aproximar bem as curvas de velocidade (Figura 6.2) e de curvatura (Figura 6.3) realizada por pilotos profissionais.

Solucionar o problema de minimização do tempo de volta utilizando um método gráfico de otimização mostrou ser viável e aplicável, considerando as dificuldades apresentadas no Capítulo 4 para obtenção de dados necessários para adoção de outros métodos de otimização.

O simulador desenvolvido (modelo super-elíptico associado ao MGD) possui algumas

limitações, como, por exemplo: longos trechos de retas podem produzir um mínimo local. Esta condição tem um pequeno efeito no tempo total da volta, mas pode produzir um percurso incomum para pilotos profissionais. Esse inconveniente foi minimizado adotando-se a fixação de pontos triviais conforme demonstrado no Capítulo 5.

Outra limitação é que o MGD assume que o modelo de simulação do veículo é capaz de seguir o caminho definido pela spline cúbica. Isso é feito vinculando os dados de curvatura e comprimento da spline às equações do modelo (como foi feito durante a construção do modelo super-elíptico) ou executando um algoritmo de controle que mantém um modelo de veículo percorrendo a trajetória definida pela spline cúbica.

Em relação ao tempo de execução do algoritmo, o método do Gradiente Duplo foi o único a executar a simulação em tempo hábil para que a análise possa ser executada durante eventos esportivos, seja nas sessões de treino, seja na prova em si. Contudo, se o objetivo é apenas a otimização do traçado sem qualquer tipo de compromisso com o tempo de execução da simulação, o método de Evolução Diferencial apresentou um tempo de volta menor que o método do Gradiente Duplo, além disso, por ser estocástico, permite uma busca mais ampla no espaço de soluções que o MGD é incapaz de realizar por ser baseado em gradiente descendente.

Neste capítulo serão apresentadas as conclusões do trabalho, contribuições e as perspectivas para trabalhos futuros.

CONCLUSÕES, CONTRIBUIÇÕES E PERSPECTIVAS

Neste trabalho, um novo método para a modelagem quase-estática de veículos de competição foi desenvolvido. O método utiliza apenas informações facilmente disponíveis para categorias amadoras de competição automotiva, a saber: dados de acelerômetro, massa do veículo, relação de troca de marchas, posição do centro de massa, entre-eixos, densidade do ar e área frontal do veículo. Um problema de otimização foi formulado para determinar o traçado ótimo e um novo método de otimização foi desenvolvido para realizar a otimização do traçado. Este novo método (Método do Gradiente Duplo) permite prever linhas de corrida descritas por splines cúbicas (problemas resolvidos na maioria dos casos por métodos estocásticos) em tempo semelhante aos métodos determinísticos.

7.1 CONTRIBUIÇÕES

- Um estudo foi realizado para verificar as características de splines cúbicas (Bézier, B-spline e Catmull-Rom) com o intuito de apontar as vantagens específicas de cada tipo de spline que levaram à sua adoção em pontos distintos do algoritmo desenvolvido: Catmull-Rom para a virtualização da pista e Bézier para interpolação do traçado. No processo de estudo, um ajuste com o objetivo de balancear a spline de Catmull-Rom foi proposto e o resultado apresentado.
- Os simuladores de tempo de volta tradicionais necessitam de informações nem sempre disponíveis para os engenheiros de categorias automotivas nacionais, especialmente as amadoras. Dados para a modelagem dos pneus são especialmente difíceis de conseguir, mesmo em categorias profissionais. Para contornar esse problema um novo método de modelagem quase estática foi desenvolvido. O novo método amplia o conceito de círculo de tração utilizando super-elipse para representar as características dinâmicas do veículo incluindo o estilo de pilotagem do piloto.

- A super-elipse utilizada no método de modelagem desenvolvido neste trabalho demanda a identificação de quatro parâmetros. Uma metodologia para identificação foi desenvolvida para determiná-los a partir dos dados de acelerômetro do veículo.
- A determinação do traçado ótimo demanda a solução de um problema de otimização. Um método baseado no conceito de gradiente descendente e inspirado em redes neurais foi desenvolvido especificamente para solucionar o problema formulado neste trabalho. Um estudo foi realizado para determinar qual o melhor método de otimização para o problema, comparando o método desenvolvido e mais quatro métodos já disponíveis. Este estudo foi apresentado na *26th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines* e o artigo foi aceito para publicação no CLAWAR 2023 Proceedings sob o título: "Double Gradient Method: A new optimization method for the trajectory optimization problem." (MACÊDO; YOUSSEF; COSTA, EasyChair, 2023). O preprint deste artigo pode ser acessado em <https://easychair.org/publications/preprint/Kxxj> embora não seja a versão final, que ainda está em processo de publicação.

7.2 SUMÁRIO DE CONCLUSÕES

Este trabalho chegou as seguintes conclusões:

- A spline de Catmull-Rom balanceada é capaz de interpolar pontos com espaçamento irregular sem interceptar a si mesma e por isso foi escolhida para realizar a virtualização do circuito.
- Construir uma spline através de seguimentos de curva Bézier arranjados de modo a garantir continuidade C^1 e C^2 garante ao traçado a continuidade de curvatura necessária para o processo de otimização, uma vez que as equações que descrevem a dinâmica do veículo são derivadas em função da curvatura do traçado.
- A modelagem super-elíptica de um grau de liberdade é incapaz de prever o comportamento do veículo com base em dados de projeto.
- A modelagem super-elíptica de um grau de liberdade utiliza o mínimo possível de dados coletados de um veículo de competição, não é preditiva, mas aplicável mesmo em competições amadoras ou semiprofissionais.
- A estrutura de produto matricial modulada por uma função de ativação das redes neurais é conveniente para problemas envolvem a manipulação direta dos pontos de controle de uma spline para a construção de traçados.
- A modificação da topologia da rede neural para deixar de atuar como um aproximador e passar a atuar como um otimizador, funcionou para o problema de otimização de traçado.

- Solucionar o problema de minimização do tempo de volta utilizando um método gráfico de otimização mostrou ser viável e aplicável.
- Longos trechos de retas podem produzir um mínimo local e isso é uma limitação do simulador desenvolvido (modelo super-elíptico associado ao MGD).
- O método gráfico assume que o modelo de simulação do veículo é capaz de seguir o caminho definido pela spline cúbica. Isso é feito vinculando os dados de curvatura e comprimento da spline às equações do modelo (como foi feito durante a construção do modelo super-elíptico) ou executando um algoritmo de controle que mantém um modelo de veículo percorrendo a trajetória definida pela spline cúbica.
- O método do Gradiente Duplo foi o único a executar a simulação em tempo hábil para que a análise possa ser executada durante eventos esportivos.
- O método de Evolução Diferencial, por ser estocástico, permite uma busca mais ampla no espaço de soluções que o MGD é incapaz de realizar por ser baseado em gradiente descendente.

7.3 PERSPECTIVAS PARA TRABALHOS FUTUROS

Com base no que foi desenvolvido, recomenda-se para trabalhos futuros:

- Generalizar o método para resolver outros problemas graficamente por meio de splines cúbicas ou manipulação de superfícies.
- Aplicar o MGD em otimização de trajetória em espaço 3D restrito, como pistas de corrida ou pistas de teste modeladas em três dimensões.
- Aplicar o MGD em otimização de trajetória em espaço 3D irrestrito para aplicações com drones, veículos aéreos não tripulados (UAV) ou veículos subaquáticos autônomos (AUV).
- Substituir o modelo super-elíptico por um Modelo de Controle Preditivo (MPC) para realizar simulações com modelos veiculares mais complexos e testar os limites do MGD.

REFERÊNCIAS BIBLIOGRÁFICAS

- AFLAK, O. *Bézier Interpolation - Create smooth shapes using Bézier curves*. [s.n.], 2020. Disponível em: (<https://towardsdatascience.com/bzier-interpolation-8033e9a262c2>).
- ALMEIDA, E. E. B. D. *Curvas de Bézier*. João Pessoa: Universidade Federal da Paraíba, 2015.
- AVON. *Formula continental*. 2023. (<https://www.avontyres.com/en-gb/tyre-care/motorsport-technical-data/technical-data-resources/formula-continental/>.) Accessed: 2023-11-30.
- BASTOS, E. A. *Otimização de Seções Retangulares de Concreto Armado Submetidas à Flexo-Compressão Oblíqua Utilizando Algoritmos Genéticos*. [S.l.]: Universidade Federal do Rio de Janeiro, 2004.
- BIANCO, N. D. et al. Comparison of direct and indirect methods for minimum lap time optimal control problems. *Veh. Syst. Dyn.*, Informa UK Limited, v. 57, n. 5, p. 665–696, maio 2019.
- BIANCO, N. D.; LOT, R.; GADOLA, M. Minimum time optimal control simulation of a GP2 race car. *Journal of Automobile Engineering*, v. 232, p. 1180–1195, 2018.
- BLUNDELL, M.; HARTY, D. *The multibody systems approach to vehicle dynamics*. 2. ed. Oxford, England: Butterworth-Heinemann, 2014.
- BRAGA, E. A. S. *Modelagem e Otimização do Problema do Caixeiro Viajante com Restrições de Tempo, Distância e Confiabilidade via Algoritmos Genéticos*. Recife, PE: Universidade Federal de Pernambuco, 2007.
- CASANOVA, D. *On Minimum Time Vehicle Manoeuvring: The Theoretical Optimal Lap*. [S.l.]: Cranfield University, 2000.
- CATMULL, E.; ROM, R. A class of local interpolating splines. In: *Computer Aided Geometric Design*. [S.l.]: Elsevier, 1974. p. 317–326.
- CHAVES, A. A. *Heurísticas Híbridas com Busca Através de Agrupamentos para o Problema do Caixeiro Viajante com Coleta de Prêmios*, Instituto Nacional de Pesquisas Espaciais. São José dos Campos: [s.n.], 2005.
- CONN, A. R. a. R. . *Trust-region methods*. Philadelphia, Pa.: SIAM, 2000. (MPS-SIAM series on optimization).

EDGAR, T. F.; HIMMELBLAU, D. M. *Optimization of chemical processes*. 2. ed. London, England: McGraw-Hill Publishing, 2001.

FARIN, G. E. *Curves and Surfaces for Computer-Aided Geometric Design - A Practical Guide*. Arizona: Academic Press Inc, 2014.

FILHO, J. L. R.; TRELEAVEN, P. C.; ALIPPI, C. *Genetic Algorithm Programming Environments*. [S.l.]: IEEE Computer Society Press, 1994. 28–43 p.

GADOLA, M. et al. A tool for lap time simulation. In: *SAE Technical Paper Series*. 400 Commonwealth Drive, Warrendale, PA, United States: SAE International, 1996.

GAO, F.; HAN, L. Implementing the Nelder-Mead simplex algorithm with adaptive parameters. *Comput. Optim. Appl.*, Springer Science and Business Media LLC, v. 51, n. 1, p. 259–277, jan. 2012.

GARLICK, S.; BRADLEY, A. Real-time optimal trajectory planning for autonomous vehicles and lap time simulation using machine learning. *Veh. Syst. Dyn.*, Informa UK Limited, v. 60, n. 12, p. 4269–4289, dez. 2022.

GILLESPIE, T. D. *Fundamentals of vehicle dynamics*. Warrendale: SAE International, 2021.

GIULIANI, C. M. *Estratégias de Otimização não Diferenciável Aplicadas à Maximização da Produção de Campos de Petróleo*. Florianópolis: [s.n.], 2013.

GROSS, J. *Derivative-free methods for high-dimensional optimization with application to centrifugal pump design*. [S.l.]: Apollo - University of Cambridge Repository, 2022.

HERRMANN, T. et al. Minimum race-time planning-strategy for an autonomous electric racecar. In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. [S.l.]: IEEE, 2020.

JAIN, A.; MORARI, M. Computing the racing line using bayesian optimization. fev. 2020.

KAPANIA, N. R.; SUBOSITS, J.; GERDES, J. C. A sequential two-step algorithm for fast generation of vehicle racing trajectories. fev. 2019.

KEGELMAN, J. C.; HARBOTT, L. K.; GERDES, J. C. Insights into vehicle trajectories at the handling limits: analysing open data from race car drivers. *Veh. Syst. Dyn.*, Informa UK Limited, v. 55, n. 2, p. 191–207, fev. 2017.

KELLY, D. P. *Lap Time Simulation with Transient Vehicle and Tyre Dynamics*. [S.l.]: Cranfield University, 2008.

KONAR, A. *Artificial Intelligence and Soft Computing: Behavioral and Cognitive Modeling of the Human Brain*. Boca Raton: CRC Press, 1999.

- KOUTRIK, S. V. *Optimal Control for Race Car Minimum Time Maneuvering*. Delft: Delft University of Technology, 2015.
- LENZO, B.; ROSSI, V. A simple mono-dimensional approach for lap time optimisation. *Appl. Sci. (Basel)*, MDPI AG, v. 10, n. 4, p. 1498, fev. 2020.
- LOT, R.; BIRAL, F. A curvilinear abscissa approach for the lap time optimization of racing vehicles. *IFAC Proc. Vol.*, Elsevier BV, v. 47, n. 3, p. 7559–7565, 2014.
- LOVATO, S.; MASSARO, M. A three-dimensional free-trajectory quasi-steady-state optimal-control method for minimum-lap-time of race vehicles. *Veh. Syst. Dyn.*, Informa UK Limited, p. 1–19, jan. 2021.
- MACÊDO, A. R.; YOUSSEF, E. S. E.; COSTA, M. V. A. da. *Double Gradient Method: a New Optimization Method for the Trajectory Optimization Problem*. EasyChair, 2023. Disponível em: (<https://easychair.org/publications/preprint/Kxxj>).
- MCBEATH, S. *Competition Car Aerodynamics - A Practical Handbook*. [S.l.]: Veloce Publishing, 2015.
- MEIROVITCH, L. *Methods of analytical dynamics*. Mineola, NY: Dover Publications, 2012. (Dover Civil and Mechanical Engineering).
- MELO, V. V. *Técnicas de Aumento de Eficiência para Metaheurísticas Aplicadas a Otimização Global Contínua e Discreta*. USP, São Carlos: [s.n.], 2009.
- MILLIKEN, D. L. et al. *Race car vehicle dynamics*. Warrendale: SAE International, 2003. (Premiere Series Books).
- MÓR, F. N. *An Evolutionary Approach for the Task Mapping Problem*. Porto Alegre: Pontifícia Universidade Católica do Rio Grande do Sul, 2016.
- NELDER, J. A.; MEAD, R. A simplex method for function minimization. *Comput. J.*, Oxford University Press (OUP), v. 7, n. 4, p. 308–313, jan. 1965.
- NOWLAN, D. *The dynamics of the racecar*. Australia: ChassisSim, 2020.
- OGATA, K. *Modern Control Engineering*. 5. ed. Upper Saddle River, NJ: Pearson, 2009.
- PACEJKA, H. B. *Tyre and vehicle dynamics*. [S.l.]: Elsevier Science & Technology, 2006.
- RAJAMANI, R. *Vehicle dynamics and control*. New York, NY: Springer, 2006. (Mechanical Engineering Series).
- RAVINDRAN, A.; RAGSDELL, K. M.; REKLAITIS, G. V. *Engineering Optimization: Methods and Applications*. Hoboken, New Jersey: John Wiley and Sons, Inc, 2006.
- ROUELLE, C. Getting more from your yaw diagrams. *Racecar-Engineering*, p. 44–46, 2017.

SEGERS, J. *Analysis Techniques for Race Car Data Acquisition*. Warrendale, PA: SAE international, 2014.

SEWARD, D. *Race Car Design*. London, England: Red Globe Press, 2014.

SMITH, C. *Tune to win*. [S.l.]: Carroll Smith Consulting, 1978.

SPENDLEY, W.; HEXT, G. R.; HIMSWORTH, F. R. Sequential application of simplex designs in optimisation and evolutionary operation. *Technometrics*, JSTOR, v. 4, n. 4, p. 441, nov. 1962.

STORN, R.; PRICE, K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, n. 11, p. 341–359, 1997.

WATT, A.; MARK, W. *Advanced Animation and Rendering Techniques - Theory and Practice*. New York: Addison-Wesley Professional, 1992.

YUAN, Y.-X. Recent advances in trust region algorithms. *Math. Program.*, Springer Science and Business Media LLC, v. 151, n. 1, p. 249–281, jun. 2015.

APÊNDICE A

INTERPOLAÇÃO POR CURVAS DE BÉZIER

A curva de Bézier não possui continuidade C^0 para uma sequência de interpolação de j pontos P na forma P_{n+m} , $n = 0, \dots, j-4$, $m = 0, \dots, 3$. Contudo, estas curvas interpolam os pontos P_n e P_{n+3} (ver Figura 3.4). Partindo deste princípio poderíamos considerar os pontos P_{n+1} e P_{n+2} como pontos auxiliares a_j e b_j respectivamente e enxertá-los na sequência de j pontos P para construir uma spline via curvas de Bézier interpolando j pontos P na forma P_{n+m} , $n = 0, \dots, j-2$, $m = 0, \dots, 1$ (AFLAK, 2020). Partindo da equação para a curva de Bézier, temos:

$$Bz^3(t) = \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} P_n \\ a_n \\ b_n \\ P_{n+1} \end{bmatrix} \quad (\text{A.1})$$

O objetivo é criar uma transição suave entre duas curvas de Bézier consecutivas, ou seja, garantir continuidade em C^1 e C^2 , isso implica em:

$$\dot{B}z_{n-1}^3(t=1) = \dot{B}z_n^3(t=0), \quad n = 1, \dots, j-2 \quad (\text{A.2})$$

$$\ddot{B}z_{n-1}^3(t=1) = \ddot{B}z_n^3(t=0), \quad n = 1, \dots, j-2 \quad (\text{A.3})$$

Cada curva Bz_n^3 possui dois pontos auxiliares e, por tanto, é necessário determinar $2n$ pontos auxiliares numa interpolação de traçado, dos quais $2n-1$ foram determinados pelas Equações A.2 e A.3. Restam dois pontos que são determinados impondo-se as seguintes condições arbitrárias:

$$\ddot{B}z_0^3(t=0) = 0 \quad (\text{A.4})$$

$$\ddot{B}z_{j-2}^3(t=1) = 0 \quad (\text{A.5})$$

Para montar o sistema de equações para a interpolação por curvas de Bézier, é necessário determinar a primeira e segunda derivadas de $Bz_n^3(t)$:

$$\dot{B}z_n^3(t) = 3 \cdot [-(1-t)^2 \cdot P_n + (1-3 \cdot t) \cdot (1-t) \cdot a_n + t \cdot (2-3 \cdot t) \cdot b_n + t^2 \cdot P_{n+1}] \quad (\text{A.6})$$

$$\ddot{B}z_n^3(t) = 6 \cdot [(1-t) \cdot P_n + (3t-2) \cdot a_n + (1-3 \cdot t) \cdot b_n + t \cdot P_{n+1}] \quad (\text{A.7})$$

Substituindo a Equação A.6 na Equação A.2:

$$3 \cdot [-b_{n-1} + P_n] = 3 \cdot [-P_n + a_n] \leftrightarrow 2 \cdot P_n = a_n + b_{n-1} \quad (\text{A.8})$$

Substituindo a Equação A.7 na Equação A.3:

$$6 \cdot [a_{n-1} - 2 \cdot b_{n-1} + P_n] = 6 \cdot [P_n - 2 \cdot a_n + b_n] \leftrightarrow a_{n-1} + 2 \cdot a_n = 2 \cdot b_{n-1} + b_n \quad (\text{A.9})$$

Substituindo a Equação A.7 na Equação A.4:

$$P_0 - 2 \cdot a_0 + b_0 = 0 \quad (\text{A.10})$$

Substituindo a Equação A.7 na Equação A.5:

$$a_{j-2} - 2 \cdot b_{j-2} + P_{j-1} = 0 \quad (\text{A.11})$$

Montando o sistema, temos:

$$\begin{cases} 2 \cdot P_n = a_n + b_{n-1}, & n = 1, \dots, j-2 \\ a_{n-1} + 2 \cdot a_n = 2 \cdot b_{n-1} + b_n, & n = 1, \dots, j-2 \\ P_0 - 2 \cdot a_0 + b_0 = 0 \\ a_{j-2} - 2 \cdot b_{j-2} + P_{j-1} = 0 \end{cases} \quad (\text{A.12})$$

Isolando b_n na Equação A.8:

$$2 \cdot P_n = a_n + b_{n-1}, \quad n = 1, \dots, j-2 \leftrightarrow b_n = 2 \cdot P_{n+1} - a_{n+1}, \quad n = 0, \dots, j-3 \quad (\text{A.13})$$

Substituindo a Equação A.13 na Equação A.9:

$$\begin{aligned} a_{n-1} + 2 \cdot a_n &= 2 \cdot b_{n-1} + b_n, \quad n = 1, \dots, j-2 \leftrightarrow \\ \begin{cases} a_{n-1} + 4 \cdot a_n + a_{n+1} = 2 \cdot (2 \cdot P_n + P_{n+1}), & n = 1, \dots, j-3 \\ a_{j-3} + 2 \cdot a_{j-2} = 2 \cdot b_{j-3} + b_{j-2}, & n = j-2 \end{cases} \end{aligned} \quad (\text{A.14})$$

Substituindo a Equação A.13 na Equação A.10:

$$P_0 + 2 \cdot P_1 = a_1 + 2 \cdot a_0 \quad (\text{A.15})$$

Substituindo as Equações A.13 e A.14 na Equação A.11:

$$8 \cdot P_{j-2} + P_{j-1} = 7 \cdot a_{j-2} + 2 \cdot a_{j-3} \quad (\text{A.16})$$

Com isso obtém-se o seguinte sistema:

$$\begin{cases} P_0 + 2 \cdot P_1 = a_1 + 2 \cdot a_0 \\ a_{n-1} + 4 \cdot a_n + a_{n+1} = 2 \cdot (2 \cdot P_n + P_{n+1}), \quad n = 1, \dots, j-3 \\ 8 \cdot P_{j-2} + P_{j-1} = 7 \cdot a_{j-2} + 2 \cdot a_{j-3} \end{cases} \quad (\text{A.17})$$

Na forma matricial:

$$\begin{bmatrix} 2 & 1 & 0 & 0 & 0 & \cdots & 0 \\ 1 & 4 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 4 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 4 & 1 & 0 \\ 0 & \cdots & 0 & 0 & 1 & 4 & 1 \\ 0 & \cdots & 0 & 0 & 0 & 2 & 7 \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_{j-4} \\ a_{j-3} \\ a_{j-2} \end{bmatrix} = \begin{bmatrix} P_0 + 2P_1 \\ 2 \cdot (2P_1 + P_2) \\ 2 \cdot (2P_2 + P_3) \\ \vdots \\ 2 \cdot (2P_{j-4} + P_{j-3}) \\ 2 \cdot (2P_{j-3} + P_{j-2}) \\ 8P_{j-2} + P_{j-1} \end{bmatrix} \quad (\text{A.18})$$

Para determinar os pontos b_n temos as equações

$$b_n = 2P_{n+1} - a_{n+1}, \quad n = 0, \dots, j-3 \quad (\text{A.19})$$

$$b_{j-2} = \frac{a_{j-2} + P_{j-1}}{2}, \quad n = j-2 \quad (\text{A.20})$$

Com as Equações A.18, A.18 e A.18 é possível interpolar uma série de pontos de controle como mostrado na Figura 3.15.

APÊNDICE B

CONTROLE LQR + FF APLICADO A MODELO NÃO LINEAR DE DINÂMICA LATERAL VEICULAR.

B.1 MODELO LINEAR

Um modelo linear só é válido sob certas restrições que serão abordadas adiante e descreve o movimento do veículo com base em parâmetros geométricos, sem considerar as forças que afetam o movimento.

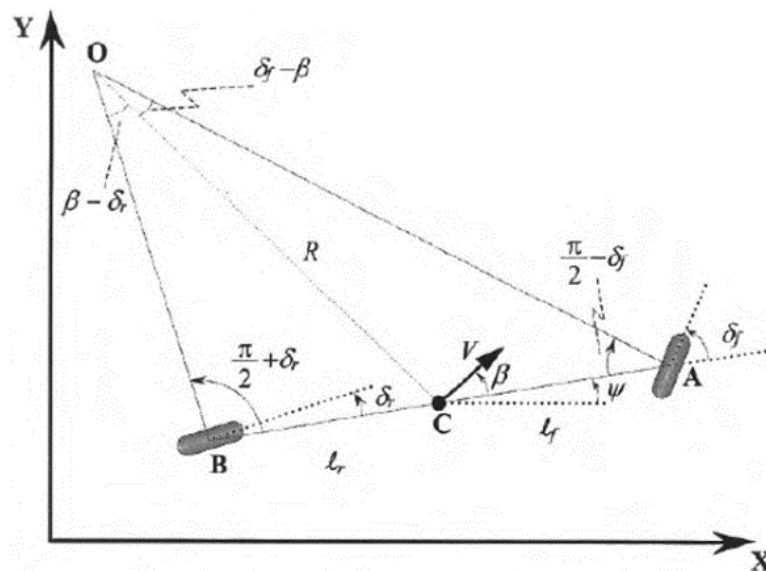


Figura B.1: Modelo de "bicicleta". Fonte: RAJAMANI (2006)

A Figura B.1 representa o "modelo de bicicleta", esse nome se deve ao fato de as rodas dianteiras serem representadas por uma única roda no centro do eixo dianteiro e da mesma forma para as rodas traseiras. Na Figura B.1, os símbolos representam os seguintes parâmetros:

- δ_f = ângulo de esterço dianteiro;
- δ_r = ângulo de esterço traseiro (o modelo é derivado considerando esterço na roda traseira);
- A = roda dianteira;

- B = roda traseira;
- C = localização do centro de gravidade do veículo (CG);
- l_f = distância entre A e C ;
- l_r = distância entre B e C ;
- $L = l_f + l_r$ = distância entre eixos;
- ψ = posição angular (yaw - define a orientação do veículo);
- V = velocidade;
- β = body slip angle (ou ângulo de deriva da carroceria).

Para construção do modelo linear serão tomadas as seguintes considerações:

- Os vetores velocidade nas rodas apontam na direção de orientação das rodas (válido para velocidades inferiores a 5 m/s);
- O ponto O é o centro instantâneo de curva e é definido pela interseção de AO com BO , que são perpendiculares às direções das rodas;
- O raio de curva R é definido por OC e o vetor velocidade no CG é perpendicular a OC , formando o ângulo β com a longitudinal do veículo;
- $\gamma = \psi + \beta$ é chamado de ângulo de curso.

Aplicando a regra dos senos ao triângulo OCA :

$$\frac{\sin(\delta_f - \beta)}{l_f} = \frac{\sin\left(\frac{\pi}{2} - \delta_f\right)}{R} \quad (\text{B.1})$$

$$\frac{\cos(\beta) \sin(\delta_f) - \sin(\beta) \cos(\delta_f)}{l_f} = \frac{\cos(\delta_f)}{R} \quad (\text{B.2})$$

$$\cos(\beta) \tan(\delta_f) - \sin(\beta) = \frac{l_f}{R} \quad (\text{B.3})$$

Aplicando a regra dos senos ao triângulo OCB :

$$\frac{\sin(\beta - \delta_r)}{l_r} = \frac{\sin\left(\frac{\pi}{2} - \delta_r\right)}{R} \quad (\text{B.4})$$

$$\frac{\sin(\beta) \cos(\delta_r) - \cos(\beta) \sin(\delta_r)}{l_r} = \frac{\cos(\delta_r)}{R} \quad (\text{B.5})$$

$$\sin(\beta) - \cos(\beta) \tan(\delta_r) = \frac{l_r}{R} \quad (\text{B.6})$$

Substituindo a Equação B.6 na Equação B.3 e manipulando as equações, temos:

$$(\tan(\delta_f) - \tan(\delta_r)) \cos(\beta) = \frac{l_f + l_r}{R} \quad (\text{B.7})$$

Manipulando as Equações B.3 e B.6, temos:

$$(\cos(\beta) \tan(\delta_f) - \sin(\beta)) l_r = \frac{l_f l_r}{R} \quad (\text{B.8})$$

$$(\sin(\beta) - \cos(\beta) \tan(\delta_r)) l_f = \frac{l_f l_r}{R} \quad (\text{B.9})$$

$$(\sin(\beta) - \cos(\beta) \tan(\delta_r)) l_f = (\cos(\beta) \tan(\delta_f) - \sin(\beta)) l_r \quad (\text{B.10})$$

$$\tan(\delta_f) l_r + \tan(\delta_r) l_f = \tan(\beta) (l_r + l_f) \quad (\text{B.11})$$

$$\beta = \tan^{-1} \left(\frac{\tan(\delta_f) l_r + \tan(\delta_r) l_f}{l_r + l_f} \right) \quad (\text{B.12})$$

Para baixas velocidades podemos assumir que a taxa de variação da orientação do veículo é igual a velocidade angular do veículo. Uma vez que a velocidade angular é V/R , então:

$$\dot{\psi} = \frac{V}{R} \quad (\text{B.13})$$

$$\dot{\psi} = \frac{V \cos(\beta)}{l_f + l_r} \cdot (\tan(\delta_f) - \tan(\delta_r)) \quad (\text{B.14})$$

As equações cinemáticas do movimento são, portanto:

$$\dot{X} = V \cos(\psi + \beta) \quad (\text{B.15})$$

$$\dot{Y} = V \sin(\psi + \beta) \quad (\text{B.16})$$

$$\dot{\psi} = \frac{V \cos(\beta)}{l_f + l_r} \cdot (\tan(\delta_f) - \tan(\delta_r)) \quad (\text{B.17})$$

B.2 MODELO NÃO-LINEAR

Para velocidades acima de 5 m/s, não é mais possível considerar que os vetores velocidade apontam na direção das rodas, o modelo linear deixa de ser válido e é necessário desenvolver um modelo não-linear. As Equações B.15 e B.16 continuarão a ser válidas para o modelo não-linear, contudo, a formulação usada para determinar os ângulos β e ψ precisarão ser atualizadas uma vez que os vetores velocidade não apontam na direção das rodas.

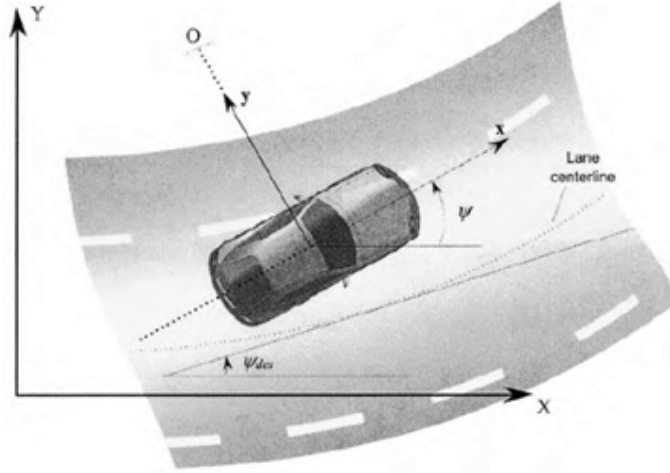


Figura B.2: Sistemas de coordenadas do modelo não-linear. Fonte: RAJAMANI (2006)

Considerando um veículo se deslocando sob uma superfície plana, o modelo de bicicleta da Figura B.2 possui apenas dois graus de liberdade, são eles: a posição lateral y e o ângulo de yaw ψ .

Existem dois sistemas de coordenadas neste modelo, um fixado no CG do veículo - que define a posição lateral y no eixo perpendicular ao eixo longitudinal do veículo e a velocidade longitudinal V_x no eixo longitudinal x do veículo - e um sistema global - que define o ângulo de yaw ψ com referência ao eixo X .

Ignorando a influência do ângulo de inclinação da pista (bank angle):

$$m \cdot a_y = F_{yf} + F_{yr}, \text{ onde } a_y = \ddot{y} \quad (\text{B.18})$$

Dois termos influenciam na aceleração lateral: a aceleração lateral \ddot{y} que é devido ao movimento ao longo do eixo y e a aceleração centrípeta $V_x \dot{\psi}$. Assim:

$$a_y = \ddot{y} + V_x \dot{\psi} \quad (\text{B.19})$$

$$m \cdot (\ddot{y} + V_x \dot{\psi}) = F_{yf} + F_{yr} \quad (\text{B.20})$$

A aceleração lateral é definida em dois termos devido à utilização de dois sistemas de coordenadas para definição do problema, um sistema de coordenadas móvel (que acompanha o deslocamento do veículo) e um sistema de coordenadas inerciais. A descrição completa do movimento de uma partícula em relação a um sistema de coordenadas móvel pode ser encontrada em MEIROVITCH (2012).

Tomando os momentos sobre o eixo z , definimos a equação para a dinâmica de yaw como:

$$I_z \dot{\psi} = l_f F_{yf} - l_r F_{yr} \quad (\text{B.21})$$

O próximo passo é definir os termos F_{yf} e F_{yr} . Para pequenos slip angles a força lateral dos pneus cresce proporcionalmente ao slip angle. O slip angle define a direção do vetor velocidade em relação a direção da roda (não confundir com o body slip angle, que define a direção do vetor velocidade em relação ao eixo x).

Slip angle na dianteira: $\alpha_f = \delta - \theta_{Vf}$, onde θ_{Vf} é o ângulo entre o vetor velocidade e o eixo longitudinal do veículo (eixo x) e δ é o ângulo de esterço.

Slip angle na traseira: $\alpha_r = -\theta_{Vr}$

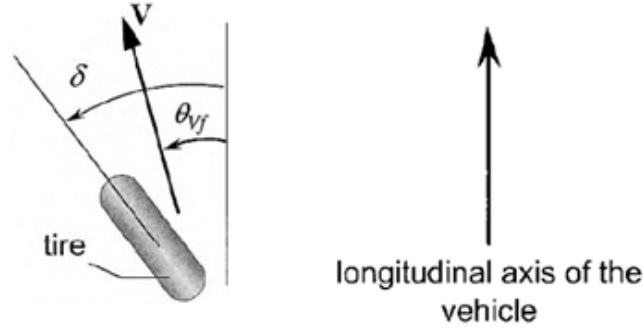


Figura B.3: Slip Angle. Fonte: RAJAMANI (2006)

A força lateral na roda dianteira para pequenos slip angles é definida por: $F_{yf} = 2C_{\alpha f}(\delta - \theta_{Vf})$, onde $C_{\alpha f}$ é chamado de cornering stiffness do pneu (o modelo de bicicleta considera os dois pneus de um eixo como um único pneu no centro, por isso o fator "2" na equação).

Da mesma forma, define-se a força lateral na roda traseira por: $F_{yr} = 2C_{\alpha r}(-\theta_{Vr})$.

Sabendo que $\tan(\theta_{Vf}) = \frac{V_y + l_f \dot{\psi}}{V_x}$ e que $\tan(\theta_{Vr}) = \frac{V_y - l_r \dot{\psi}}{V_x}$, aproximando por pequenos ângulos e tomando $V_y = \dot{y}$, temos:

$$\theta_{Vf} = \frac{\dot{y} + l_f \dot{\psi}}{V_x} \quad (\text{B.22})$$

$$\theta_{Vr} = \frac{\dot{y} - l_r \dot{\psi}}{V_x} \quad (\text{B.23})$$

Substituindo as Equações B.22 e B.23 nas equações B.20 e B.21, temos a definição do sistema não-linear da seguinte forma:

$$\ddot{y} = - \left(\frac{2C_{\alpha f} + 2C_{\alpha r}}{m \cdot V_x} \right) \dot{y} - \left(\frac{2C_{\alpha f} l_f - 2C_{\alpha r} l_r}{m \cdot V_x} + V_x \right) \dot{\psi} + \left(\frac{2C_{\alpha f}}{m} \right) \delta \quad (\text{B.24})$$

$$\ddot{\psi} = - \left(\frac{2C_{\alpha f} l_f - 2C_{\alpha r} l_r}{V_x I_z} \right) \dot{y} - \left(\frac{2C_{\alpha f} l_f^2 + 2C_{\alpha r} l_r^2}{V_x I_z} \right) \dot{\psi} + \left(\frac{2C_{\alpha f} l_f}{I_z} \right) \delta \quad (\text{B.25})$$

Ou na forma matricial $\dot{\chi} = A\chi + B\delta$, onde:

$$\chi = \begin{bmatrix} y \\ \dot{y} \\ \psi \\ \dot{\psi} \end{bmatrix} \quad (\text{B.26})$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\left(\frac{2C_{\alpha f} + 2C_{\alpha r}}{m \cdot V_x}\right) & 0 & -\left(\frac{2C_{\alpha f} l_f - 2C_{\alpha r} l_r}{m \cdot V_x} + V_x\right) \\ 0 & 0 & 0 & 1 \\ 0 & -\left(\frac{2C_{\alpha f} l_f - 2C_{\alpha r} l_r}{V_x I_z}\right) & 0 & -\left(\frac{2C_{\alpha f} l_f^2 + 2C_{\alpha r} l_r^2}{V_x I_z}\right) \end{bmatrix} \quad (\text{B.27})$$

$$B = \begin{bmatrix} 0 \\ \left(\frac{2C_{\alpha f}}{m}\right) \\ 0 \\ \left(\frac{2C_{\alpha f} l_f}{I_z}\right) \end{bmatrix} \quad (\text{B.28})$$

Considerando o efeito da inclinação da pista (bank angle):

$$m \cdot (\ddot{y} + V_x \dot{\psi}) = F_{yf} + F_{yr} + F_{bank} \quad (\text{B.29})$$

$$F_{bank} = mg \sin(\phi) \quad (\text{B.30})$$

Considerando pequenos ângulos, $F_{bank} = mg\phi$. Logo:

$$\dot{\chi} = A\chi + B\delta + B_2\phi \quad (\text{B.31})$$

$$B_2 = \begin{bmatrix} 0 \\ g \\ 0 \\ 0 \end{bmatrix} \quad (\text{B.32})$$

B.3 REGULADOR QUADRÁTICO ÓTIMO

O problema de controle ótimo que, dado um sistema de equações do tipo $\dot{\chi} = A\chi + Bu$, determina uma matriz K para o vetor de controle ótimo $u(t) = -K\chi(t)$ que minimize a função de custo $J = \int_0^\infty (\chi^t Q \chi + u^t R u) dt$, onde Q e R são matrizes Hermitianas definidas positivas ou matrizes simétricas reais, está representado em diagrama de blocos na Figura B.4.

Note que o segundo termo do lado direito da equação contabiliza a quantidade de energia despendida no sinal de controle. As matrizes Q e R determinam a importância relativa dos erros e deste gasto de energia.

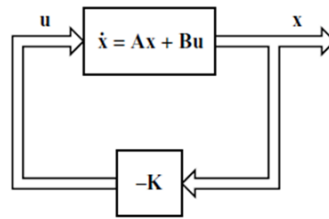


Figura B.4: Diagrama do sistema controlado pela matriz K . Fonte: OGATA (2009)

Substituindo o vetor de controle ótimo $u(t) = -K\chi(t)$ na equação do sistema, temos:

$$\dot{\chi} = A\chi - BK\chi = (A - BK)\chi \quad (\text{B.33})$$

Se $(A - BK)$ for estável, haverá uma matriz P definida positiva que satisfaz a equação reduzida de Riccati:

$$A^t P + PA - PBR^{-1}B^t P + Q = 0 \quad (\text{B.34})$$

Sabendo que a lei de controle ótimo para o problema de controle ótimo quadrático quando o índice de performance é dado por $J = \int_0^\infty (\chi^t Q \chi + u^t R u) dt$ é igual a $u(t) = -K\chi(t) = -R^{-1}B^t P\chi(t)$, determina-se a matriz K substituindo P na equação de controle ótimo.

Uma outra maneira de determinar a matriz K é utilizando o comando do Matlab® “[K,P,E] = lqr(A,B,Q,R)”. Este comando retorna as matrizes K e P e o vetor de autovalores E que representam os polos do sistema.

Uma maneira de saber se o sistema pode ser controlado, é determinar a controlabilidade do sistema através do seguinte comando em Matlab®: $\text{rank}(\text{ctrb}(A,B))$. O resultado deste comando é o posto da matriz de controlabilidade que, se for igual ao número de estados no sistema, indica que o sistema pode ser controlado.

B.4 IMPLEMENTAÇÃO DO LQR NO MODELO LINEAR

Uma vez implementado em Matlab o modelo linear desenvolvido na Seção B.2 e verificando se o modelo desenvolvido pode ser controlado a partir de um controlador quadrático (aplicando o comando “ $\text{rank}(\text{ctrb}(A,B))$ ”), temos que o posto da matriz de controlabilidade do sistema é igual a 3 enquanto o número de estados no sistema é 4, logo, para o modelo desenvolvido não há uma matriz K que forneça o vetor de controle ótimo $u(t)$.

Sabendo que o modelo desenvolvido não é útil para aplicação de LQR, é preciso desenvolver um modelo que represente a dinâmica lateral do veículo, mas que seja controlável.

Para adequar o sistema à aplicação de um controle LQR, o vetor de estados χ será redefinido da seguinte forma:

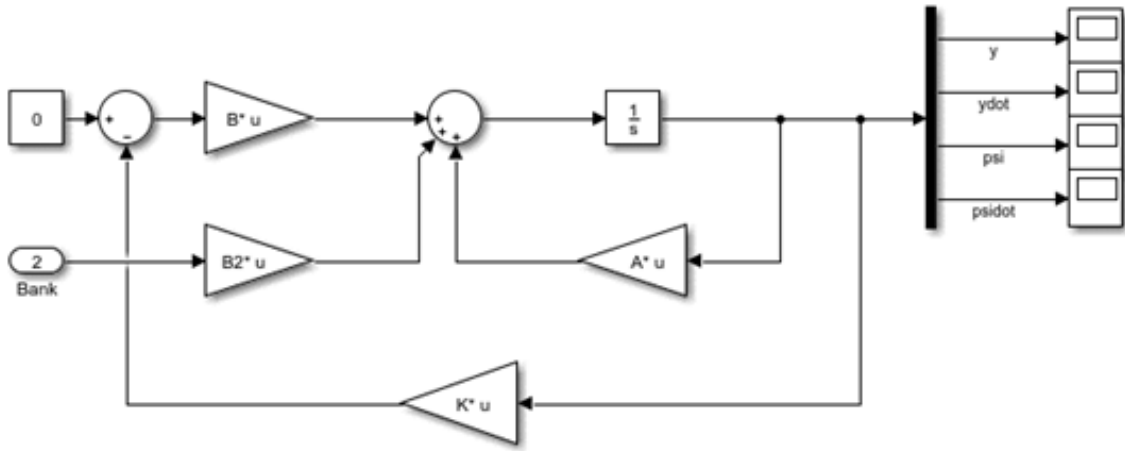


Figura B.5: Modelo linear implementado em Matlab®

$$\chi = \begin{bmatrix} e_1 \\ \dot{e}_1 \\ e_2 \\ \dot{e}_2 \end{bmatrix} \quad (\text{B.35})$$

Onde:

- e_1 = distância entre o CG do veículo e a linha que descreve o traçado
- e_2 = erro de orientação do veículo em relação à tangente do traçado

Considerando um veículo trafegando em velocidade longitudinal V_x constante em um traçado de raio R constante, em que o raio seja grande o suficiente para que as considerações de pequenos ângulos permaneçam válidas, a taxa de mudança de orientação desejada é dada por:

$$\dot{\psi}_{des} = \frac{V_x}{R} \quad (\text{B.36})$$

A aceleração lateral desejada é então definida por:

$$\frac{V_x^2}{R} = V_x \dot{\psi}_{des} \quad (\text{B.37})$$

O erro \ddot{e}_1 será definido da seguinte forma:

$$\ddot{e}_1 = \ddot{y} + V_x \dot{\psi} - V_x \dot{\psi}_{des} = \ddot{y} + V_x (\dot{\psi} - \dot{\psi}_{des}) \quad (\text{B.38})$$

Logo:

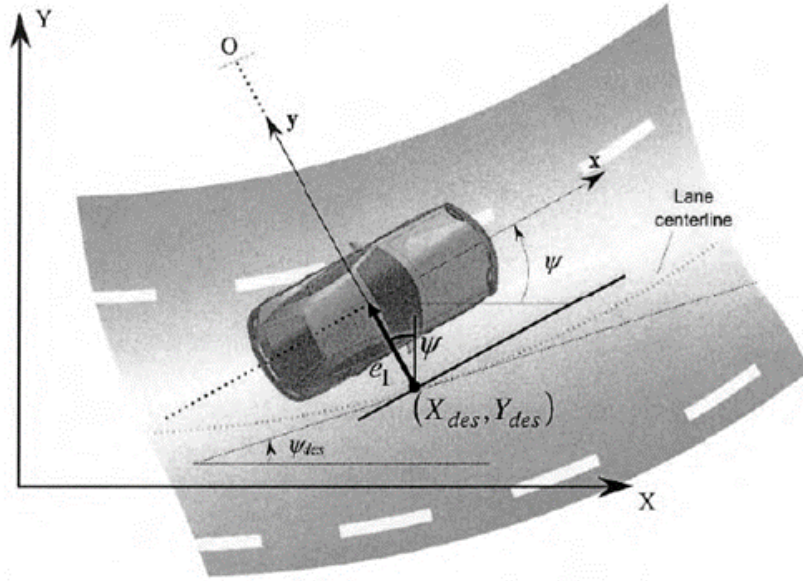


Figura B.6: Modelo linear em função dos erros e_1 e e_2 . Fonte: RAJAMANI (2006)

$$\dot{e}_1 = \dot{y} + V_x (\psi - \psi_{des}) \quad (\text{B.39})$$

O erro e_2 será definido da seguinte forma:

$$e_2 = \psi - \psi_{des} \quad (\text{B.40})$$

Substituindo as Equações B.38 e B.39 nas Equações B.24 e B.25, o modelo linear passa a ser descrito da seguinte forma:

$$\dot{\chi} = A\chi + B\delta + B_2\phi + B_3\dot{\psi}_{des} \quad (\text{B.41})$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -\left(\frac{2C_{\alpha f} + 2C_{\alpha r}}{m \cdot V_x}\right) & \left(\frac{2C_{\alpha f} + 2C_{\alpha r}}{m}\right) & -\left(\frac{2C_{\alpha f} l_f - 2C_{\alpha r} l_r}{m \cdot V_x}\right) \\ 0 & 0 & 0 & 1 \\ 0 & -\left(\frac{2C_{\alpha f} l_f - 2C_{\alpha r} l_r}{V_x I_z}\right) & \left(\frac{2C_{\alpha f} l_f - 2C_{\alpha r} l_r}{I_z}\right) & -\left(\frac{2C_{\alpha f} l_f^2 + 2C_{\alpha r} l_r^2}{V_x I_z}\right) \end{bmatrix} \quad (\text{B.42})$$

$$B = \begin{bmatrix} 0 \\ \left(\frac{2C_{\alpha f}}{m}\right) \\ 0 \\ \left(\frac{2C_{\alpha f} l_f}{I_z}\right) \end{bmatrix} \quad (\text{B.43})$$

$$B_2 = \begin{bmatrix} 0 \\ g \\ 0 \\ 0 \end{bmatrix} \quad (\text{B.44})$$

$$B_3 = \begin{bmatrix} 0 \\ -\left(\frac{2C_{\alpha f}l_f - 2C_{\alpha r}l_r}{m \cdot V_x} + V_x\right) \\ 0 \\ -\left(\frac{2C_{\alpha f}l_f^2 + 2C_{\alpha r}l_r^2}{V_x I_z}\right) \end{bmatrix} \quad (\text{B.45})$$

Verificando se o modelo linear em função dos erros e_1 e e_2 pode ser controlado a partir de um controlador quadrático, temos que o posto da matriz de controlabilidade do sistema representado pela Equação B.41 é igual a 4, enquanto o número de estados no sistema também é 4, logo, existe uma matriz K que fornece o vetor de controle ótimo $u(t)$.

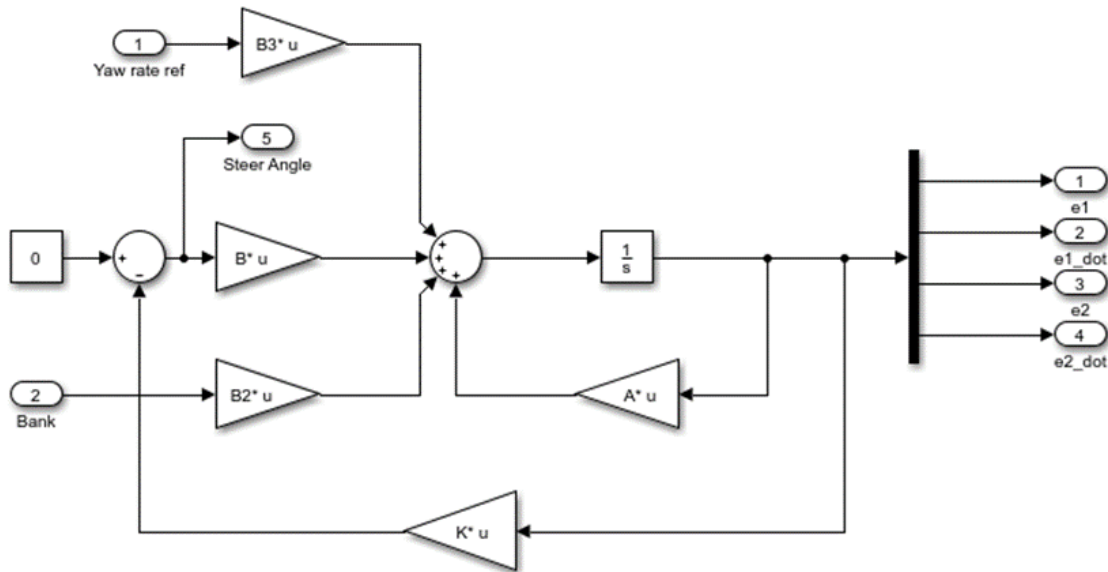


Figura B.7: Modelo linear em função dos erros e_1 e e_2 implementado em Matlab®

Por fim, o ângulo β é definido como o ângulo entre a direção em que aponta o vetor velocidade V do veículo e o eixo longitudinal do veículo (eixo x), logo:

$$\beta = \frac{\dot{y}}{V_x} = \frac{1}{V_x} (\dot{e}_1 - V_x e_2) = \frac{1}{V_x} \dot{e}_1 - e_2 \quad (\text{B.46})$$

B.5 CONTROLE FEEDFORWARD

Para forçar a convergência do estado e_1 para zero em regime permanente é necessário adicionar ao sistema um mecanismo que permita rejeitar o erro em regime permanente. Supondo que o sinal de controle (ângulo de esterço) seja composto por dois componentes, o feedback dos estados e um sinal que prevê o erro em regime permanente (feedforward), sua equação seria:

$$\delta = -K\chi + \delta_{ff} \quad (\text{B.47})$$

Substituindo a Equação B.47 na Equação B.41, temos:

$$\dot{\chi} = (A - BK)\chi + B\delta_{ff} + B_2\phi + B_3\dot{\psi}_{des} \quad (\text{B.48})$$

Tomando a transformada de Laplace:

$$X(s) = (sI - A + BK)^{-1} \left(B\mathcal{L}(\delta_{ff}) + B_2\mathcal{L}(\phi) + B_3\mathcal{L}(\dot{\psi}_{des}) \right) \quad (\text{B.49})$$

Sabendo que $L(\delta_{ff}) = \frac{\delta_{ff}}{s}$, $L(\phi) = \frac{\phi}{s}$ e $L(\dot{\psi}_{des}) = \frac{\dot{\psi}_{des}}{s} = \frac{V_x}{R \cdot s}$ e aplicando o teorema do valor final:

$$\chi_{ss} = \lim_{t \rightarrow \infty} \chi(t) = \lim_{s \rightarrow 0} sX(s) = (-A + BK)^{-1} \left(B\delta_{ff} + B_2\phi + B_3\dot{\psi}_{des} \right) \quad (\text{B.50})$$

$$\chi_{ss} = \begin{bmatrix} \frac{\delta_{ff}}{k_1} \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -\frac{mV_x^2(l_r C_{\alpha r} - l_f C_{\alpha f} + k_3 l_f C_{\alpha f})}{2Rk_1 C_{\alpha r} C_{\alpha f} (l_r + l_f)} - \frac{l_r + l_f - k_3 l_r}{Rk_1} \\ 0 \\ -\frac{2C_{\alpha r} l_r^2 - m l_f V_x^2 + 2l_f C_{\alpha r} l_r}{2RC_{\alpha r} (l_r + l_f)} \\ 0 \end{bmatrix} + \begin{bmatrix} \frac{g\phi m}{2k_1 (l_r + l_f)} \left(\frac{l_r}{C_{\alpha f}} + \frac{k_3 l_f - l_f}{C_{\alpha r}} \right) \\ 0 \\ -\frac{gm\phi l_f}{2C_{\alpha r} (l_r + l_f)} \\ 0 \end{bmatrix} \quad (\text{B.51})$$

A Equação B.51 demonstra que não existe ação de controle que possa ser tomada para rejeitar o erro do estado e_2 em regime permanente. Para rejeição do erro do estado e_1 , o sinal de controle feedforward é dado por:

$$\delta_{ff} = \frac{mV_x^2 (l_r C_{\alpha r} - l_f C_{\alpha f} + k_3 l_f C_{\alpha f})}{2RC_{\alpha r} C_{\alpha f} (l_r + l_f)} + \frac{l_r + l_f - k_3 l_r}{R} - \frac{g\phi m}{2(l_r + l_f)} \left(\frac{l_r}{C_{\alpha f}} + \frac{k_3 l_f - l_f}{C_{\alpha r}} \right) \quad (\text{B.52})$$

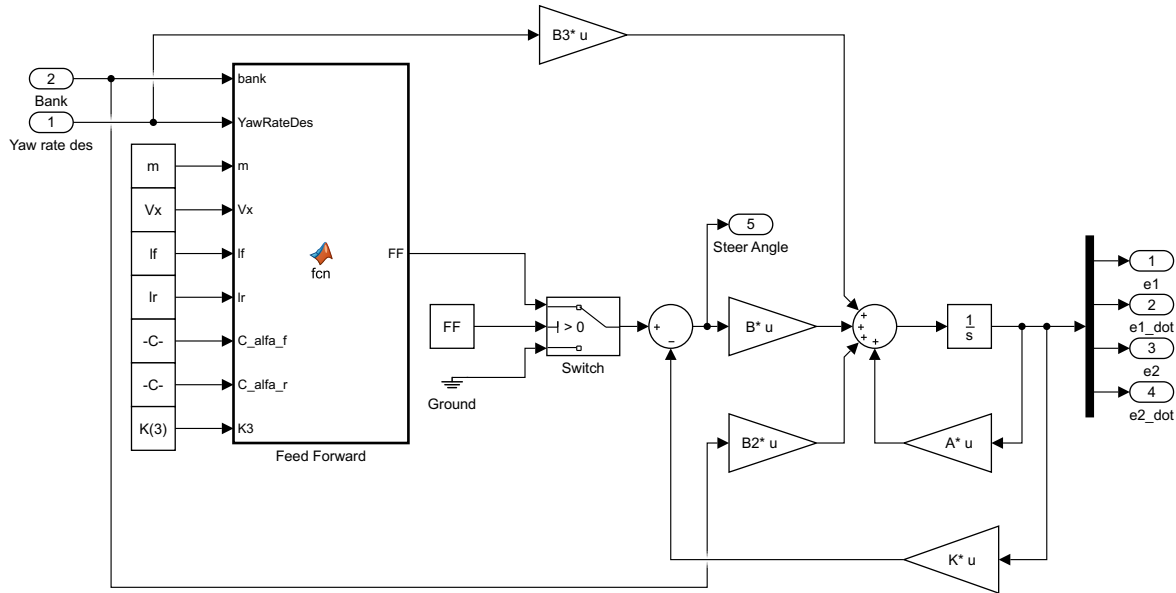


Figura B.8: Modelo linear com controle LQR + FF implementado em Matlab®

A Equação B.52 produz sinais de controle instantâneos, incompatíveis com o sistema físico real. Para contornar a instantaneidade do sinal de controle feedforward, a Equação B.52 receberá um termo de atraso:

$$\delta_{ff} + \dot{\delta}_{ff} = \frac{mV_x^2 (l_r C_{\alpha r} - l_f C_{\alpha f} + k_3 l_f C_{\alpha f})}{2RC_{\alpha r} C_{\alpha f} (l_r + l_f)} + \frac{l_r + l_f - k_3 l_r}{R} - \frac{g\phi m}{2(l_r + l_f)} \left(\frac{l_r}{C_{\alpha f}} + \frac{k_3 l_f - l_f}{C_{\alpha r}} \right) \quad (B.53)$$

No domínio de Laplace isso equivale a:

$$\delta_{ff} = \frac{1}{s+1} \left(\frac{mV_x^2 (l_r C_{\alpha r} - l_f C_{\alpha f} + k_3 l_f C_{\alpha f})}{2RC_{\alpha r} C_{\alpha f} (l_r + l_f)} + \frac{l_r + l_f - k_3 l_r}{R} - \frac{g\phi m}{2(l_r + l_f)} \left(\frac{l_r}{C_{\alpha f}} + \frac{k_3 l_f - l_f}{C_{\alpha r}} \right) \right) \quad (B.54)$$

$$C_{\alpha i} \left(F_{zi}, V_x, \dot{\psi}_{des}, \delta, \dot{e}_1, \dot{e}_2, e_2 \right) = \frac{F_{yi} \left(F_{zi}, V_x, \dot{\psi}_{des}, \delta, \dot{e}_1, \dot{e}_2, e_2 \right)}{\alpha_i \left(V_x, \dot{\psi}_{des}, \delta, \dot{e}_1, \dot{e}_2, e_2 \right)} \quad (\text{B.57})$$

Onde o subscrito “i” pode ser substituído por “f” ou “r” para indicar a qual eixo do veículo a equação se aplica.

Existe uma faixa de operação em que o comportamento das forças nos pneus é linear (as restrições já foram indicadas na construção do modelo linear), para essa faixa é possível assumir $C_{\alpha i} = \frac{F_{yi}}{\alpha_i}$, o que leva às equações utilizadas na Seção B.1:

$$F_{yf} = 2C_{\alpha f} (\delta - \theta_{Vf}) \quad (\text{B.58})$$

$$F_{yr} = 2C_{\alpha r} (-\theta_{Vr}) \quad (\text{B.59})$$

Onde o termo “2” é introduzido para contabilizar o efeito dos dois pneus pertencentes ao eixo do veículo no modelo de bicicleta.

O mecanismo de geração de forças nos pneus, além de depender de uma série de parâmetros possui uma dinâmica característica em função de dois fenômenos: histerese e adesão. Dessa forma, a força de atrito em um pneu não cresce linearmente em função da carga sobre ele como seria de esperar pela formulação de Coulomb para o atrito (ver Figura 4.2).

Existem diversos modelos utilizados para representar o comportamento dos pneus, desde os modelos empíricos (utilizados para regressão de dados de testes) até os modelos físicos (PACEJKA, 2006). O modelo utilizado nesta seção é um modelo semi-empírico simplificado desenvolvido especificamente para regredir dados pré-tratados de testes realizados em pneus reais. Um modelo semi-empírico é um modelo cujos parâmetros de regressão possuem algum significado em relação a fenômenos físicos dos pneus.

As equações do modelo dos pneus são:

$$f_y(\alpha, F_z) = (\mu_y F_z - (FC \cdot F_z - FC \cdot F_{z0})) \frac{\tan^{-1} \left(\frac{2\pi\alpha}{C_{\alpha}\alpha_{max}} \right)}{\sqrt{2} - 0.135C_{\alpha} + 0.135} \quad (\text{B.60})$$

$$f_x(\alpha, F_z) = (\mu_x F_z - (FC \cdot F_z - FC \cdot F_{z0})) \left(\frac{S_x}{2} \cos \left(\frac{\alpha\pi}{\alpha_{max}} \right) + \left(1 - \frac{S_x}{2} \right) \right) \quad (\text{B.61})$$

$$F_y(F_x, \alpha, F_z) = \text{sgn}(\alpha) \sqrt{f_y(\alpha, F_z)^2 \left(1 - \frac{F_x^2}{f_x(\alpha, F_z)^2 + 0.0001} \right)} \quad (\text{B.62})$$

- f_y = força lateral pura
- f_x = força longitudinal pura
- F_y = força lateral combinada

- F_x = força longitudinal combinada
- μ_i = coeficiente de atrito
- F_z = carga vertical
- F_{z0} = carga de referência
- FC = fator de carga
- α = slip angle
- α_{max} = slip angle máximo
- S_x = fator de escorregamento
- C_α = fator de rigidez

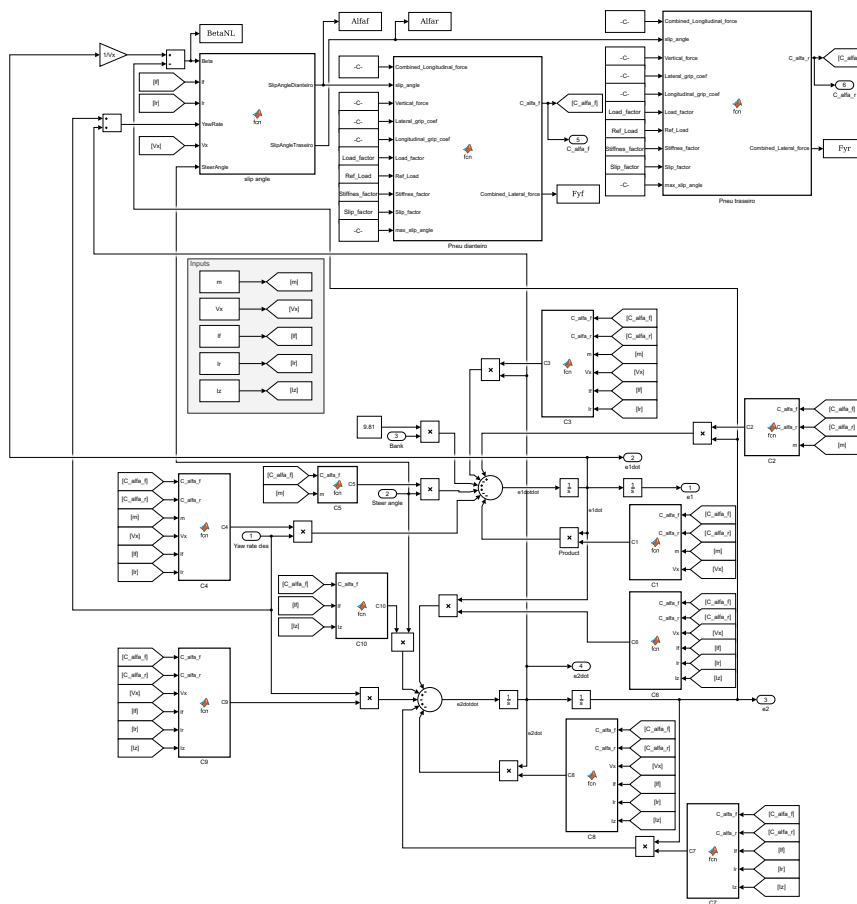


Figura B.10: Modelo não linear implementado em Matlab®

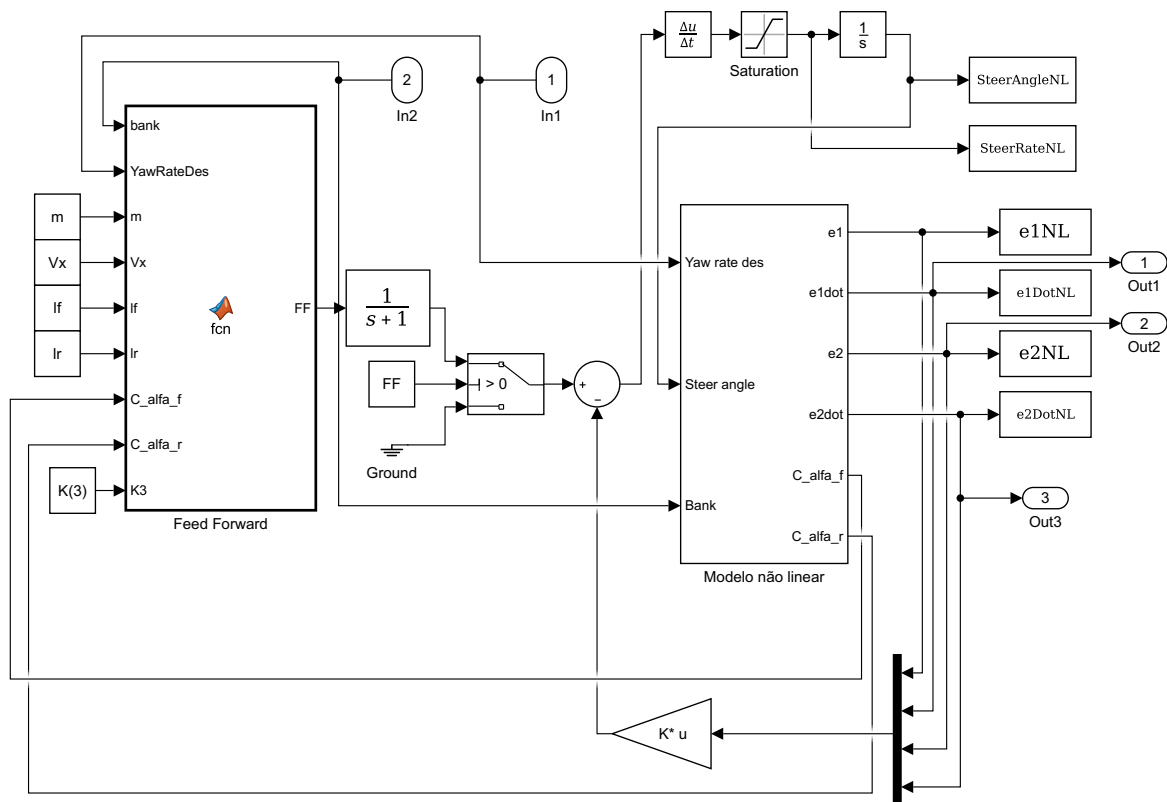


Figura B.11: Controle LQR + FF aplicado ao modelo não linear

APÊNDICE C

CÓDIGO PYTHON DO MÉTODO DO GRADIENTE DUPLO

```
1
2 import numpy as np
3 import numpy.linalg as alg
4 import csv
5 import Interpolador_Bezier as Bezier
6 import datetime
7 from numba import jit
8 import matplotlib.pyplot as plt
9
10
11 def sigmoide(x):
12     sig = 1/(1+np.exp(-15.637*x+7.819))
13     return sig
14
15
16 def load_setup(localfilename):
17     with open(localfilename, mode='r') as infile:
18         setup = [linha.split(',') for linha in infile]
19         dict_setup = {'pista': str(setup[0][1]),
20                      'lim_trac_g': float(setup[1][1]),
21                      'cap_lgt_g': float(setup[2][1]),
22                      'cap_lat_g': float(setup[3][1]),
23                      'largura_do_carro': float(setup[4][1]),
24                      'peso_do_carro': float(setup[5][1]),
25                      'ro': float(setup[6][1]),
26                      'area_frontal_do_carro': float(setup[7][1]),
27                      'cd': float(setup[8][1]),
28                      'cl': float(setup[9][1]),
29                      'na': float(setup[10][1]),
30                      'nb': float(setup[11][1]),
31                      'Ta': float(setup[12][1]),
32                      'Tb': float(setup[13][1]),
33                      'Tc': float(setup[14][1]),
34                      'ma': float(setup[15][1]),
35                      'mb': float(setup[16][1])}
36
37     return dict_setup
38
39
40 def load_data(localfilename):
41     with open(localfilename, mode='r') as infile:
42         reader = csv.reader(infile)
43         # pontos = [matriz[x, y]]
```

```

44     pontos = np.array([np.array([float(long), float(lat)]) for long
, lat in reader])
45
46     return pontos
47
48
49 def tracado_carro(genes, pts_ints, pts_exts, beta=1.0):
50     pts_n_interpolados = np.array([pts_ints[i] + (pts_exts[i] -
pts_ints[i]) * (genes[i]) for i in range(len(genes))])
51     pts_interpolados, prim_derivada, seg_derivada, controles = Bezier.
evaluate_bezier(pts_n_interpolados, beta)
52
53     return pts_interpolados, prim_derivada, seg_derivada, controles
54
55
56 def parametros_de_curva(pontos, d_pontos, dd_pontos):
57     # determinar o arco entre pontos
58     dist = ([alg.norm(pontos[i] - pontos[i - 1]) for i in range(1, len(
pontos))])
59     dist.insert(0, dist[-1])
60     dist = np.asarray(dist)
61
62     # determinar curvatura
63     sgn_kapa = np.array([np.sign(np.cross((d_pontos[i] / alg.norm(
d_pontos[i])),
64                                     (dd_pontos[i] / (alg.norm(
dd_pontos[i]) + 0.0000001))))
65                             for i in range(len(d_pontos))])
66
67     kapa = np.array([(alg.norm(np.cross(d_pontos[i], dd_pontos[i])) /
68                             np.power(alg.norm(d_pontos[i]), 3)) * sgn_kapa[i]
69                             for i in range(len(d_pontos))])
70
71     # determinar raio
72     raio = np.array([abs(1 / (kapa[i] + 0.0000001)) for i in range(len(
kapa))])
73
74     direcao = [np.cross(d_pontos[i - 1], d_pontos[i]) for i in range(1,
len(d_pontos))]
75     direcao.insert(0, direcao[-1])
76     direcao = np.asarray(direcao)
77
78     return raio, dist, direcao
79
80
81 @jit(nopython=True)
82 def velocidades(raio, dist, lim_trac, cap_lgt, cap_lat, weight, ro, af,
cl, cd, na, nb, ta, tb, tc, ma, mb):
83     vel = np.array([float(0) for _ in range(len(raio))])
84     acel = np.array([lim_trac for _ in range(len(raio))])
85     freio = np.array([float(0) for _ in range(len(raio))])
86     acel_lat = np.array([float(0) for _ in range(len(raio))])
87     plot_acel = np.array([float(0) for _ in range(len(raio))])

```

```

88
89     k = 2
90     while k > 0:
91         k -= 1
92         for i in range(1, len(raio)):
93             if k == 0:
94                 vel[0] = vel[len(raio) - 1]
95                 acel[0] = acel[len(raio) - 1]
96                 acel_lat[0] = acel_lat[len(raio) - 1]
97
98                 vel_i = np.sqrt(vel[i - 1] ** 2 + (2 * acel[i - 1] * dist[i
99 ]))
100                 plot_acel[i - 1] = acel[i - 1]
101                 if k == 0 and vel_i == vel[i]:
102                     break
103                 vel[i] = vel_i
104                 mi_aero = (- 0.5 * ro * af * cl * vel[i]**2) / weight
105                 mi_y = mi_aero*cap_lat + cap_lat
106                 gb_aero = (0.5 * ro * af * cd * vel[i] ** 2) / (weight
107 /9.81)
108                 acel_lat[i] = vel[i] ** 2 / raio[i]
109                 if acel_lat[i] > mi_y:
110                     vel[i] = np.sqrt(mi_y * raio[i])
111                     acel_lat[i] = mi_y
112                     freio[i] = 0
113                     j = i
114                     while True:
115                         j -= 1
116                         if j == -1:
117                             j = len(raio) - 2
118                             vel[len(raio) - 1] = vel[0]
119                             freio[len(raio) - 1] = freio[0]
120                             acel_lat[len(raio) - 1] = acel_lat[0]
121
122                 vel_j = np.sqrt(vel[j + 1] ** 2 + (2 * freio[j + 1]
123 * dist[j + 1]))
124                 plot_acel[j + 1] = -freio[j + 1]
125                 if vel_j >= vel[j]:
126                     break
127                 else:
128                     vel[j] = vel_j
129                     acel_lat[j] = vel[j] ** 2 / raio[j]
130                     mi_aero_j = (- 0.5 * ro * af * cl * vel[j] **
131 2) / weight
132                     mi_y_j = mi_aero_j*cap_lat + cap_lat
133                     gb_aero_j = (0.5 * ro * af * cd * vel[j] ** 2)
134 / (weight/9.81)
135                     if acel_lat[j] > mi_y_j:
136                         vel[j] = np.sqrt(mi_y_j * raio[j])
137                         acel_lat[j] = mi_y_j
138                         gxb = cap_lgt * np.tanh(vel[j]) + mi_aero_j*
139 cap_lgt + gb_aero_j
140                         freio[j] = (abs(gxb) ** mb * (1 - abs(acel_lat[

```

```

135         j] / mi_y_j) ** nb))**(1/mb)
136                 continue
137         trac = ta * vel[i]**2 - tb * vel[i] + tc
138         gxa = trac - gb_aero if trac <= lim_trac else lim_trac -
gb_aero
139         acel[i] = (abs(gxa) ** ma * (1 - abs(acel_lat[i] / mi_y)**
na))**(1/ma)
140
141     return vel, plot_acel, acel_lat
142
143
144 def exportar(x):
145
146     pontos_ints = load_data('controles_internos.csv')
147     pontos_exts = load_data('controles_externos.csv')
148     pontos, d_pontos, dd_pontos, _ = tracado_carro(x, pontos_ints,
pontos_exts, beta=5)
149
150     # Dados do carro #####
151     setup = load_setup('setup.csv')
152     lim_trac = setup.get('lim_trac_g') * 9.81
153     cap_lgt = setup.get('cap_lgt_g') * 9.81
154     cap_lat = setup.get('cap_lat_g') * 9.81
155     weight = setup.get('peso_do_carro')
156     ro = setup.get('ro')
157     af = setup.get('area_frontal_do_carro')
158     cd = setup.get('cd')
159     cl = setup.get('cl')
160     na = setup.get('na')
161     nb = setup.get('nb')
162     ta = setup.get('Ta')
163     tb = setup.get('Tb')
164     tc = setup.get('Tc')
165     ma = setup.get('ma')
166     mb = setup.get('mb')
167     # #####
168
169     raio, dist, direcao = parametros_de_curva(pontos, d_pontos,
dd_pontos)
170     vels, ax, ay = velocidades(raio, dist, lim_trac, cap_lgt, cap_lat,
weight,
171                               ro, af, cl, cd, na, nb, ta, tb, tc, ma,
mb)
172
173     tempo = [0]
174     for i in range(1, len(vels)):
175         delta_v = (vels[i] + vels[i - 1]) / 2
176         tempo.append(dist[i] / delta_v)
177
178     print('Tempo de volta = ', sum(tempo))
179
180     return pontos, raio, dist, direcao, vels, ax, ay, tempo

```

```

181
182
183 def fobjetivo(x):
184
185     pontos_ints = load_data('controles_internos.csv')
186     pontos_exts = load_data('controles_externos.csv')
187     pontos, d_pontos, dd_pontos, controles = tracado_carro(x,
188     pontos_ints, pontos_exts, beta=1)
189
190     # Dados do carro #####
191     setup = load_setup('setup.csv')
192     lim_trac = setup.get('lim_trac_g') * 9.81
193     cap_lgt = setup.get('cap_lgt_g') * 9.81
194     cap_lat = setup.get('cap_lat_g') * 9.81
195     weight = setup.get('peso_do_carro')
196     ro = setup.get('ro')
197     af = setup.get('area_frontal_do_carro')
198     cd = setup.get('cd')
199     cl = setup.get('cl')
200     na = setup.get('na')
201     nb = setup.get('nb')
202     ta = setup.get('Ta')
203     tb = setup.get('Tb')
204     tc = setup.get('Tc')
205     ma = setup.get('ma')
206     mb = setup.get('mb')
207     # #####
208
209     raio, dist, direcao = parametros_de_curva(pontos, d_pontos,
210     dd_pontos)
211     vels, ax, ay = velocidades(raio, dist, lim_trac, cap_lgt, cap_lat,
212     weight,
213     ro, af, cl, cd, na, nb, ta, tb, tc, ma,
214     mb)
215
216     tempo = [0]
217     for i in range(1, len(vels)):
218         delta_v = (vels[i] + vels[i - 1]) / 2
219         tempo.append(dist[i] / delta_v)
220
221     tempo_total = sum(tempo)
222
223     deriv_ay = [0]
224     for i in range(1, len(ay)):
225         deriv = abs(ay[i] - ay[i - 1])
226         deriv_ay.append(deriv)
227
228     deriv_ay[0] = deriv_ay[-1]
229
230     deriv_ay_controles = [0]
231     tempo_controles = [0]
232     curvatura_controles = [0]
233     for i in range(1, len(controles)+1):

```

```

230     ji = sum(controles[0:i-1]) # controle anterior
231     ja = sum(controles[0:i]) # controle atual
232     jf = sum(controles[0:i+1]) if i != len(controles) else sum(
controles[0:1]) # controle futuro
233     if i != len(controles):
234         tempo_controles.append((tempo_total**2)*sum(tempo[ji:ja-1])
*sum(tempo[ja:jf-1]))
235         deriv_ay_controles.append(sum(deriv_ay[ji:ja-1])+sum(
deriv_ay[ja:jf-1]))
236         curvatura_controles.append(1/raio[ja - 1])
237     else:
238         tempo_controles.append((tempo_total**2)*sum(tempo[ji:ja -
1])*sum(tempo[0:jf - 1]))
239         deriv_ay_controles.append(sum(deriv_ay[ji:ja - 1])+sum(
deriv_ay[0:jf - 1]))
240         curvatura_controles.append(1/raio[ja - 1])
241
242     tempo_controles[0] = tempo_controles[-1]
243     tempo_controles = np.asarray(tempo_controles)
244     tempo_controles = np.reshape(tempo_controles, (len(tempo_controles)
, 1))
245
246     deriv_ay_controles[0] = deriv_ay_controles[-1]
247     deriv_ay_controles = np.asarray(deriv_ay_controles)
248     deriv_ay_controles = np.reshape(deriv_ay_controles, (len(
deriv_ay_controles), 1))
249
250     acy = ay*np.sign(direcao)
251
252     return tempo_total, tempo_controles, pontos, pontos_ints,
pontos_exts, acy, deriv_ay_controles, curvatura_controles
253
254
255 if __name__ == '__main__':
256
257     x0 = np.ones((len(load_data('controles_internos.csv')), 1))/2
258
259     starttime = datetime.datetime.now()
260     print('Simulacao iniciada - {}'.format(str(starttime)))
261
262     entra = len(x0) # tamanho da camada de entrada
263     alfa = 0.01 # taxa de aprendizagem
264     stuck = 0
265
266     w = np.ones((entra, 1)) # inicializacao dos pesos
267     w[0:15], w[24:27], w[33:34], w[45:47], w[71:78], w[86:93] = 2, 2,
2, 0, 2, 2
268     memt = np.zeros((entra, 1)) # memoria de tempo
269     memp = x0 # memoria de posicao
270     memday = np.zeros((entra, 1)) # memoria da derivada da aceleracao
lateral
271
272     besty = x0

```

```

273     bestt = 9999
274     bestpts = []
275
276     plt.figure(1, figsize=(6, 2))
277     thismanager = plt.get_current_fig_manager()
278     thismanager.window.wm_geometry("+0+0")
279
280     plt.figure(2, figsize=(8, 8))
281     thismanager = plt.get_current_fig_manager()
282     thismanager.window.wm_geometry("+600+0")
283
284     plt.figure(3, figsize=(6, 2))
285     thismanager = plt.get_current_fig_manager()
286     thismanager.window.wm_geometry("+0+250")
287
288     plotx = []
289     ploty1 = []
290     ploty2 = []
291
292     for it in range(1, 40000 + 1):
293
294         y = sigmoide(w*x0) # saida da rede
295
296         t, tcon, pts, pin, pout, oay, daycon, curvatura = fobjetivo(y)
297         stuck = stuck + 1 if t >= bestt else 0
298         bestt = t if t < bestt else bestt
299         besty = y if t == bestt else besty
300         bestpts = pts if t == bestt else bestpts
301         print('t=', datetime.datetime.now() - starttime, 'i=', it, 'obj
= ', t, 'best=', bestt, 'stuck=', stuck)
302
303         plotx.append(it)
304         ploty1.append(bestt)
305         ploty2.append(t)
306
307         gradt = np.sign(tcon - memt)
308         gradp = np.sign(y - memp)
309         graday = np.sign(daycon - memday)
310
311         alfa = alfa/2 if stuck > 0 and stuck % 100 == 0 and alfa >
0.0001 else alfa
312         print(alfa)
313
314         for jt in range(len(w)):
315             if curvatura[jt] >= 0.002:
316                 w[jt] = w[jt] - alfa * gradt[jt] * gradp[jt]
317             else:
318                 w[jt] = w[jt] - alfa * graday[jt] * gradp[jt]
319                 # w[jt] = w[jt] - alfa * gradt[jt] * gradp[jt]
320
321         w[0] = w[-1]
322         w[0:15], w[24:27], w[33:34], w[45:47], w[71:78], w[86:93] = 2,
2, 2, 0, 2, 2

```

```

323     memt = tcon
324     memp = y
325     memday = daycon
326
327
328     if stuck == 400:
329         break
330
331     if it % 10 == 0:
332         plt.figure(1)
333         plt.clf()
334         plt.plot(plotx, ploty1, color='b')
335         plt.plot(plotx, ploty2, color='r')
336
337         plt.figure(2)
338         plt.clf()
339         plt.plot(bestpts[:, 0], bestpts[:, 1], 'b')
340         plt.plot(pts[:, 0], pts[:, 1], 'r')
341         plt.plot(pin[:, 0], pin[:, 1], 'black')
342         plt.plot(pout[:, 0], pout[:, 1], 'black')
343
344         plt.figure(3)
345         plt.clf()
346         plt.plot(oay, color='b')
347
348         plt.pause(0.0001)
349
350     sol = besty.flatten()
351
352     timediff = datetime.datetime.now() - starttime
353     print('Simulacao concluida. Tempo de simulacao = {}'.format(str(
354     timediff)))
355     print(sol)
356
357     e_pontos, e_raio, e_dist, e_direcao, e_vels, e_ax, e_ay, e_tempo =
358     exportar(sol)
359
360     with open('Dados_exportados.csv', mode='w+') as outfile:
361         print('Simulacao iniciada - {}'.format(str(starttime)), file=
362         outfile)
363         print('Simulacao concluida. Tempo de simulacao = {}'.format(str
364         (timediff)), file=outfile)
365         print('...', file=outfile)
366         print('Vetor x0:', file=outfile)
367         print(x0.flatten(), file=outfile)
368         print('...', file=outfile)
369         print('Pontos de controle = {}'.format(len(x0)), file=outfile)
370         print('...', file=outfile)
371         print('Vetor x resultante:', file=outfile)
372         print(sol, file=outfile)
373         print('...', file=outfile)
374         print('Setup usado:', file=outfile)
375         print(load_setup('setup.csv'), file=outfile)

```



```
372     print('...', file=outfile)
373     print('Longitude,Latitude,Raio,Distancia,Direcao,Velocidade,ax,
ay,Tempo', file=outfile)
374     for it in range(len(e_pontos)):
375         print('{},{},{},{},{},{},{},{},{}'.format(e_pontos[it][0],
e_pontos[it][1], e_raio[it], e_dist[it],
376                                                     e_direcao[it],
e_vels[it], e_ax[it], e_ay[it], e_tempo[it]),
377             file=outfile)
378
379 plt.show()
```

Listing C.1: Código Python do método do gradiente duplo