FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Planeamento Otimizado de Movimento de Robô Submarino

Pedro Diogo Pinto da Silva

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientador: Aníbal Castilho Coimbra de Matos

Co-orientador: Maria do Rosário Marques Fernandes Teixeira de Pinho

27 de Outubro de 2014

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MIEEC - MESTRADO INTEGRADO EM ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES 2013/2014

A Dissertação intitulada

"Planeamento Otimizado de Movimento de Robô Submarino"

foi aprovada em provas realizadas em 08-10-2014

o júri

Presidente Professor Doutor Fernando Manuel Ferreira Lobo Pereira
Professor Catedrático do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Professor Doutor José Manuel do Valle Cardoso Igreja
Professor Coordenador do Departamento de Engenharia Eletrotécnica - Energia e Automação do Instituto Superior de Engenharia de Lisboa

Professora Doutora Maria do Rosário Marques Fernandes Teixeira de Pinho
Professora Associada do Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

O autor declara que a presente dissertação (ou relatório de projeto) é da sua exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros extratos tomados de ou inspirados em trabalhos de outros autores, e demais referências bibliográficas usadas, são corretamente citados.

Autor - Pedro Diogo Pinto da Silva

Faculdade de Engenharia da Universidade do Porto

Resumo

Nos últimos anos tem havido uma crescente pesquisa e uma necessidade enorme, nas sociedades atuais, do uso de sistemas robóticos autónomos, que possibilitem a utilização de robots para atividades praticadas pelo homem; poupando assim recursos e eliminando o nível de perigo para a componente humana outrora usada.

A autonomia nos robots, permite poupar tempo e dinheiro nas suas várias utilizações; desde o mais simples robot, que substitui aquele empregado, encarregue de ordenar e coordenar um armazém, até aos sistemas mais complexos como é o caso, dos submarinos autónomos, que vão até aos locais mais profundos possíveis, em rios e oceanos, fazer missões de reconhecimento.

Ora, com isso é conveniente manter-se um nível de desenvolvimento constante, que permite fazer mais e melhor, inserindo-se nesta parte o estudo do controlo ótimo, onde se pretende melhorar, ou seja, otimizar o complexo sistema de um veículo submarino autônomo.

Os submarinos autónomos, daqui em diante, *AUV* (*Autonomous Underwater Vehicles*), não foram referidos por acaso; eles englobam-se no estudo deste documento, e assim, imagine-se que um determinado *AUV*, estará destinado a fazer uma missão de reconhecimento num rio profundo e com um terreno acidentado; quer-se então fazer com que o robot se desloque de um ponto **A** a um ponto **B**, pré-determinados.

Será que se pode fazer que, com o controlo atual do mesmo ele se desloque no mínimo tempo possível, evitando obstáculos e contrariando correntes aquáticas, ou seja, minimizar o tempo que ele demora a percorrer o proposto? E será que se pode melhorar o seu consumo de energia, minimizando-o?

É aqui que se inserem as técnicas de controlo ótimo, modelizando o problema de forma matemática, com o estado e os controlos do sistema pretendido, pode-se responder às questões atrás colocadas, efetuando uma análise analítica do problema e validando resultados.

Como tal, neste projeto, começa-se por fundamentar um modelo que descreva o problema, com o seu estado e os seus controlos, fazer uma análise analítica do mesmo e validar os resultados obtidos, podendo mais tarde serem empregues em ambiente real; respondendo assim às questões levantadas.

Abstract

In the last years, there has been a growing research and a huge necessity, in today's society, on the field of autonomous robotics systems, that enable the use of robots to activities otherwise practiced by man, saving resources and eliminating the danger to the human component.

The robots autonomy, allows save time and money in its various uses, since from the simplest robot that replaces a employee in charge of organizing and coordinate a repository, to more complex systems as is the case of autonomous submarines, ranging up to the deepest sites in rivers and oceans, to do recognition missions.

Now, it is convenient to keep a constant level of development that lets you do more and better, and here enters the study of optimal control, which aims to improve, or to optimize, the complex system such as an autonomous underwater vehicle.

The autonomous submarine, from now on *AUV* (*Autonomous Underwater Vehicles*), were not referred by chance, they include themselves in the study of this document, and thus, imagine that a given *UUV*, is destined to do a reconnaissance mission deep in a river and whit a rugged terrain; the objective then, its to move the robot from a point A to a point B, previously established.

Is it possible to do that, whit the current control of the robots, it makes the path mentioned before, in the minimum time possible, avoiding obstacles and opposing the river currents, or in other words, to minimize the time to run the proposed? And can we improve the energy consumption, minimizing it?

And its here that the optimal control techniques appears by modeling the mathematical problem, whit the state and the controls of the desired system, we can answer those questions, performing an analytical analysis of the problem and validating results.

As such, this project begins by support a model that describes the problem with their states and controls, make an analytical analysis of it and validate the results obtained, and later those results can be used in a real environment, thus responding to the raised issues.

Agradecimentos

Antes de mais agradeço a Deus por me ter posto neste mundo, apesar da situação atual que se vive.

Agradeço também aos meus pais e irmã, que sempre me ajudaram neste percurso atribulado desde o momento de nascimento - sendo que a minha irmã nada poderia fazer durante dois anos, pois é mais nova - até ao culminar de um ciclo que foi cheio de percalços e peripécias.

Agradeço a todas as pessoas que entraram na minha vida e me aturaram, sei que não foi fácil, mas lembrem-se que por causa disso, um lugarzinho no céu estará porventura garantido.

Especial agradecimento aos meus orientadores da Dissertação (Professor Aníbal Matos e a Professora Maria do Rosário Pinho), que foram sempre pessoas impecáveis, ajudando em todos os obstáculos que apareciam no projeto , nunca desistindo de mim, o que me faz pensar que deveria ter sido um melhor orientando.

Agradeço de uma forma especial também à Professora Margarida, que apesar de não me dever nada - pois nada a haver tinha com a Dissertação - não deixou de ajudar, sempre que o precisei.

Agradeço imenso também ao pessoal do laboratório que me foi ofertado - (Laboratório I202) especialmente à Ana, ao Tiago, à Filipa, à Sofia, à Juliana, à Amélia, ao Luís e ao Igor - para concluir este projeto, que me ajudaram e me aturaram nas minhas imensas dúvidas e questões, nem sempre relacionadas com trabalho, mas na sua maioria sim.

Mais uma vez agradeço, a todo o pessoal com quem convivi nestes tempos de faculdade, na pala do DEEC (Departamento de Engenharia Electrotécnica e de Computadores), no próprio DEEC, pessoal de energia, automação e telecomunicações, e em especial quem me aturou mais e ajudou apesar da minha preguicite aguda - Diana Soares e Fernando Cunha. E agradecimento especial, às cozinheiras que mantiveram os meus valores "nutricionais"nos eixos, não revelo nomes para não haver represálias, deixando apenas as iniciais N.(D.A.),C.S,J.S referidas, são umas "Migas"muito "fixes".

E agora uma palavra de apreço especial, naquela que foi porventura um pilar imenso na minha vida acadêmica (mas não só), e sim, só poderia estar a falar da Praxe. Passo-vos então a fazer de pessoa *Narradora* nesta história bela mas atribulada, tal como uma Rosa pequena (*Little Rose - Rosinha*) com *Pikos* (espinhos), que não começou há *19* anos, mas bem que poderia ter começado, com as suas regras e os seus *Jingle*(s) sempre presentes, desde pessoas com nomes mais comuns como *Inácio*, *Cardoso* e *Maia*, até nomes fictícios como o do *Chuck*, *MPT*, *Martelo* ou o do *Timoneiro* que nos alumia o caminho, como uma *Tocha*, até chegarmos a um local *Vazio* para fazermos "refil"ao nosso stock de genialidade com o *Dred* ali da zona, que na compra de dois quilogramas da mesma, nos oferecia um *Cacete*. Vivemos sempre momentos especiais, ou nem sempre, pois a vida oferece sempre *Pro*'s e *Contra*'s mas poderemos sempre retirar lições de cada um deles. Ouvir discursos fantásticos com mais ou menos *Volume*, no restaurante a comer *Leitão* da Bairrada, ao som do *Quim* (Barreiros ou não) por vezes a cantar em *Alemão*. Por ironia nesse dia comi *Alheira* ("salvo-seja- que ganda *Lolj* (lê-se "LOL")), até que à saída do restaurante "*Ocorreu*"uma situação bastante peculiar, uns "*Jonas*", ou seja, os maninhos ali da

zona, projetavam um efeito "cheerleader"(tipo mais que uma pessoa *Torcedora*, lá na terra dos americanos) muito confuso e enigmático; um tinha um *Barreto(C3)* com as inicias estampadas dentro de uma *Seta* no mesmo, reparei que não tinha *Etiqueta*, tinha um ar de *Tarado*, suposição essa satisfeita, com o seu diálogo engraçado a tentar imitar o *Marco Horácio* a dizer piadas menos próprias; outro dava saltos como um *Kanguru*, mas mesmo esquisito, de um modo *Autómato* mesmo aluado como alguém que não é desta nossa *Terra*; havia uma moçoila, que apresentava-se bem vestida, parecia uma *Aeromoça*, mas apresentava uma cor de quem viajava muito acima do limite da lei, quase a chamar o *Greg*, se havia ali um *Radar*, acusava excesso de velocidade; outra personagem tinha uma cana muito pequena (*Caninha*), mas em vez de estar a pescar, como um *Pescador*, parecia usá-la como uma enxada, devia porventura ser filho de um *Lavrador*; havia ainda um outro que era um bocado jabardo (ouvi chamar *Jarbas*, senti logo que fosse para ele, sou mesmo perspicaz), esse mesmo encontrava-se a fazer um desenho na parede - mas que *Pintor*-, era azul, a parede, e desenhava com *Corrector*, pareceu-me desenhar um jogador de futebol pela relva toda, devia ser o *Rodriguez* que jogou no porto, que corria muito - se calhar foi de encontro com a outra rapariga que estava mal disposta, mas estou a divagar -, olhando melhor para o desenho apercebemo-nos que também podia ser o *Cantona* mas faltava uma *Meia* ao "player", não deve ter acabado o desenho ainda (só pensava, quando viria a polícia checkar a *ID* do people); havia uma outra figura, com uma camisola às listas - fez-me tanto lembrar uma *Zebra* -, mas esquisito, tinha dois *Raminhos* numa caixa, que parecia uma *Estante* pequena, e andava com eles à roda como aquela senhora que *Torce* a roupa num tanque, parecia querer sintonizar uma televisão - talvez para ver um jogo de futebol, daí o player de à bocado - e os pequenos paus serviam de *Antenas*, este rapaz tinha uma figura que se assemelhava ao *Mr. Bogus*, personagem realizado pelo Gerard Depardieu num filme, há muitos anos atrás, entrava também a Whoopi Goldberg que fez também "Do Cabaré para o Convento", lindo filme, tipo menina fora da lei à noite e irmã *Lúcia* a rezar a *Nossa Senhora Que* fica sempre bem durante o dia; Havia ainda outro jovem que devia ser um bocado autista, encontrava-se a brincar com uma maquete - deve ser *Maquetista* o rapaz - de uma estação de comboios, e ia mesmo a passar uma *Locomotiva* só e abandonada pela estação, e eis que ele lhe deixa cair em cima uma *Bigorna* que a arrebenta toda, fiquei confuso; Finalmente no fim daquele espetáculo encontrava-se a maior personagem de todas, um rapaz com uma *Jukebox* com a forma do *Bender* (desenho animado), que servia para dar o "beat"para o seu solo de *Speaker*/rapper, o rapaz até tinha vontade, as rimas eram bem feitas cheias de *Métrica*, mas com o seu sotaque de *Viseu*, falhava nas entradas, era *Precoce* em relação à música apesar do seu "swagg", mas que grande *Ameba*.

Há muitas mais histórias destas, com montes de personagens, uma sempre mais *Banana* que outra, de todos os anos, palavra de apreço a todos também, e vou parar senão "ides"começar a pensar "*Taz* a ficar muito sentimental meu".

Espero não me ter esquecido de ninguém, mas já agora fica aqui um grande *Espaço* no fim, para os que aí virão, a porta fica aberta.

Pedro Diogo Pinto da Silva

"It has been said, 'time heals all wounds'. I do not agree. The wounds remain. In time, the mind, protecting its sanity, covers them with scar tissue and the pain lessens. But it is never gone."

Rose Kennedy

Conteúdo

1 Introdução **1**
- 1.1 Motivação . . . 1
- 1.2 Objetivo . . . 2
- 1.3 Estrutura do Documento . . . 2

2 Constituição e Movimentação de Robots Submarinos **3**
- 2.1 Robots Submarinos . . . 3
  - 2.1.1 *Slocum Glider* . . . 3
  - 2.1.2 *Mares* . . . 6
- 2.2 *Path Planning* . . . 7
- 2.3 Modelo Geral de um Robot Submarino . . . 8
  - 2.3.1 Equações de Cinemática . . . 9
  - 2.3.2 Equações da Dinâmica . . . 10
  - 2.3.3 Equações Simplificadas de Movimento . . . 10

3 Fundamentos Teóricos **13**
- 3.1 Teoria do Controlo . . . 13
  - 3.1.1 Exemplificação . . . 13
  - 3.1.2 Problemática . . . 14
- 3.2 Otimização . . . 16
- 3.3 Controlo Ótimo Determinístico . . . 17
  - 3.3.1 Introdução . . . 17
  - 3.3.2 Formulação . . . 18
  - 3.3.3 Existência de Solução . . . 20
  - 3.3.4 Condições Necessárias de Otimalidade . . . 21

4 Software Utilizado **23**
- 4.1 AMPL . . . 23
  - 4.1.1 *IPOPTS* . . . 24
- 4.2 Exemplificação do Ficheiro em *AMPL* . . . 24
  - 4.2.1 Método de *Euler* . . . 25
  - 4.2.2 Apresentação do Ficheiro a Usar no *AMPL* . . . 25
- 4.3 *Matlab* . . . 27

5 Planeamento das Trajetórias **29**
- 5.1 Introdução aos Casos 1 e 2 . . . 29
  - 5.1.1 Caso 1 - Tempo Mínimo . . . 30
  - 5.1.2 Outros Exemplos idênticos ao Caso 1 . . . 36

5.1.3 Caso 2 - Consumo de Energia Mínimo 37
5.1.4 Outros Exemplos idênticos ao Caso 2 42
5.2 Introdução aos Casos 3 e 4 43
5.2.1 Caso 3 - Tempo Mínimo com Correntes 43
5.2.2 Outros Exemplos idênticos ao Caso 3 49
5.2.3 Caso 4 - Minimização de Energia com Correntes 50
5.2.4 Outros Exemplos idênticos ao Caso 4 55
5.3 Introdução ao Caso 5 56
5.3.1 Caso 5 - Tempo Mínimo com Dinâmica e Correntes 56
5.3.2 Outros Exemplos idênticos ao Caso 5 62

6 Conclusões e Trabalhos Futuros **65**
6.1 Conclusões 65
6.2 Trabalho Futuro 65

A Caso 1 **67**
A.1 Ficheiro *AMPL* 67
A.2 Ficheiro *Matlab* 69

B Caso 2 **73**
B.1 Ficheiro *AMPL* 73
B.2 Ficheiro *Matlab* 75

C Caso 3 **79**
C.1 Ficheiro *AMPL* 79
C.2 Ficheiro *Matlab* 81

D Caso 4 **83**
D.1 Ficheiro *AMPL* 83
D.2 Ficheiro *Matlab* 85

E Caso 5 **89**
E.1 Ficheiro *AMPL* 89
E.2 Ficheiro *Matlab* 91

Lista de Figuras

2.1 *Slocum Glider* em alto-mar 4
2.2 Estrutura constituinte do *Slocum Glider* 4
2.3 Ilustração da movimentação do *Slocum Glider* 5
2.4 Mares em terra 6
2.5 Estrutura do *Mares* 6
2.6 Representação do veículo e referencial 8

3.1 Reservatório de Água 14
3.2 Configuração de Controlo 14
3.3 Configuração de Controlo Moderno 15
3.4 Componentes de um Sistema de Controlo Moderno 16
3.5 Processo de Otimização 17

5.1 *AMPL* a correr na linha de comandos ambiente *Linux* 31
5.2 Estados e Controlos Ótimos para o Caso 1 31
5.3 Percurso X vs Y 32
5.4 Multiplicadores das equações de estado 33
5.5 Função $func$ 35
5.6 Todos os percursos do Caso 1 (Plano Y vs X) 36
5.7 *AMPL* a correr na linha de comandos ambiente *Linux* 38
5.8 Estados e controlos ótimos para o caso 2 38
5.9 Percurso X vs Y 39
5.10 Multiplicadores das equações de estado 40
5.11 $u_{\text{analítico}}$ vs u numérico 41
5.12 Todos os percursos do Caso 2 (Plano Y vs X) 42
5.13 *AMPL* a correr na linha de comandos ambiente *Linux* 44
5.14 Estados e controlos ótimos para o caso 3 44
5.15 Percurso 2D Y vs X 45
5.16 Multiplicadores das equações de estado 46
5.17 Função $func$ 48
5.18 Todos os percursos do Caso 3 (Plano Y vs X) 49
5.19 *AMPL* a correr na linha de comandos ambiente *Linux* 51
5.20 Estados e controlos ótimos do caso 4 51
5.21 Gráfico y (m) *Vs* x(m) 52
5.22 Multiplicadores das equações de estado 53
5.23 $u_{analtico}$ vs $u$numérico 54
5.24 Todos os percursos do Caso 4 (Plano Y vs X) 55
5.25 *AMPL* a correr na linha de comandos ambiente *Linux* 57

5.26 Estados e controlos ótimos para o caso 3 57
5.27 Percurso 2D X vs Y 58
5.28 Multiplicadores das equações de estado 59
5.29 Hamiltoniano 60
5.30 Todos os percursos do Caso 5 (Plano X vs Y) 62

Lista de Tabelas

2.1 Especificações do *Slocum Glider* 5
2.2 Especificações do *Mares* 7

5.1 Resultados dos Casos de Estudo de Tempo Mínimo Simples 36
5.2 Resultados dos Casos de Estudo de Tempo Mínimo Simples 42
5.3 Resultados dos Casos de Estudo de Tempo Mínimo com Correntes 50
5.4 Resultados dos Casos de Estudo de Consumo de Energia Mínima com Correntes 56
5.5 Resultados dos Casos de Estudo de Consumo de Energia Mínima com Correntes 63

Abreviaturas

AMPL	A Mathematical Programming Language
AUV	Autonomous Underwater Vehicle
AUVG	Autonomous Underwater Vehicle Glider
GPS	Global Positioning System
GUI	Graphical User Interface
IPOPT	Library for large scale nonlinear optimization of continuous systems (Interior Point OPTimizer)
Mares	Modular Autonomous Robot for Environment Sampling
Matlab	Software for Numerical computation, visualization and programming
prcol	primeira coluna
seccol	segunda coluna
thircol	terceira coluna

Capítulo 1

Introdução

1.1 Motivação

Nos últimos anos as missões com recurso a robots submarinos autónomos (AUV's), têm vindo a aumentar, na medida em que são fundamentais na monitorização de zonas de difícil acesso ao homem não correndo o risco de haver perdas de vida humana, já que estes podem ser tripulados por controlo remoto, sem necessidade de pessoas a bordo. Exemplos como o *Slocum Glider* e o Mares do grupo OceanSYS - submarinos presentes na Faculdade de Engenharia da Universidade do Porto, local de realização desta dissertação -, são bem justificativos do uso feito por estes robots, em que a principal característica é o seu sistema de navegação com boa poupança de energia, o que deixa estes aparelhos com boa autonomia às custas da sua velocidade relativamente lenta. Daí os mesmos poderem ficar longos períodos de tempo submersos em água, embora sejam fortemente afetados pelas correntes marítimas segundo [4].

Surge neste percurso um verdadeiro problema que é o de traçar rotas de movimentação dos submarinos. Ora, como fazê-los percorrer um trajeto predefinido evitando obstáculos, e como o fazer no menor trajeto possível poupando assim energia? A primeira pergunta pode ser tratada com os estudos já feitos na parte de *path planning* da movimentação de robots submarinos, estudada na área de robótica, tema esse que será tratado mais abaixo no capítulo 2.

Quanto à segunda questão, abordam-se aqui possíveis métodos de otimizar este tema, recorrendo a técnicas de controlo ótimo – capítulo 3.

O planeamento da movimentação de um robot predefinindo um trajeto a percorrer faz-se, com a intenção do mesmo ir de um dado ponto (a - inicial) até um ponto (b - final) ultrapassando obstáculos. Este problema poderá porventura ser de difícil resolução devido a enumeras variáveis, tais como: tamanho, forma e graus de liberdade do robot.

Quer-se assim delinear o movimento do robot com recurso à modelação do estado do sistema do mesmo, aplicando controlos e otimizando esse sistema. Assim, passa-se para a fase de compreensão analítica do sistema com os resultados obtidos, fazendo-se de seguida uma validação dos mesmos que será parte fundamental desta dissertação.

1.2 Objetivo

O objetivo fulcral do projeto é desenvolver e implementar um mecanismo de planeamento de movimentos de um robot submarino recorrendo a técnicas de controlo ótimo. Pretende-se assim ter em conta fatores importantes como o consumo energético, o tempo de execução, entre outros, e aplicar controlo ótimo para resolver problemas, otimizando o percurso referente aos robots acima enunciados. Exemplo disso serão porventura as variações de correntes marítimas, possibilitando um dado trajeto ser feito com o menor consumo de energia possível, aumentando assim o tempo de operação do robot.

1.3 Estrutura do Documento

Aqui apresenta-se a ordem lógica com que se estruturou este documento.

Assim, no capítulo 2 faz-se um estudo referente aos robots submarinos presentes no local de realização desta dissertação (Faculdade de Engenharia da Universidade do Porto), ao nível da sua constituição e da sua movimentação.

No capítulo 3 faz-se uma introdução ao estudo da área do controlo ótimo. Inicia-se este capítulo, com uma explicação do controlo moderno, passa-se pela necessidade de otimização do mesmo, e termina-se o capítulo, abordando técnicas relativas a controlo ótimo para modelar, analisar e validar resultados.

No capítulo 4 faz-se uma apresentação do software a utilizar, para trabalhar com problemas de controlo ótimo.

No capítulo 5 apresentam-se os resultados simulados, com os modelos específicos do nosso problema aplicando, no software atrás mencionado, técnicas de controlo ótimo, e assim obtiveram-se resultados que prontamente foram validados. Ainda no capítulo 5 além dos casos abordados ao pormenor, referem-se dentro de cada um, alguns exemplos com definições inicias e finais diferentes do caso geral em si.

Finalmente no capítulo 6 discutem-se os resultados obtidos, e apresentam-se algumas propostas de trabalho futuro.

Capítulo 2

Constituição e Movimentação de Robots Submarinos

Neste capítulo faz-se, não só, uma revisão do estudo referente aos robots submarinos que se pretendem considerar para este projeto, bem como, às possíveis metodologias de planeamento da movimentação de um robot submarino.

2.1 Robots Submarinos

Aqui faz-se uma breve comparação entre os dois tipos de *AUV's* (*Autonumous Underwater Vehicles*), possíveis de serem usados, devido à sua grande acessibilidade, pois encontram-se nas instalações da Faculdade de Engenharia da Universidade do Porto, local da realização desta Dissertação. E então são eles:

- ***Slocum Glider***
- ***Mares***

2.1.1 *Slocum Glider*

O *slocum Glider* é um *AUV* para pesquisa oceânica muito versátil e manobrável. As suas capacidades de flutuabilidade impulsionada e o seu longo alcance fazem dele uma boa ferramenta para missões de observação e reconhecimento de regiões. Devido às características da sua parte frontal "*nose bump*" e aos sensores incorporados; dá ao utilizador uma grande flexibilidade de otimizar a sua rota de movimentação conforme seja requerido nas missões a efetuar.

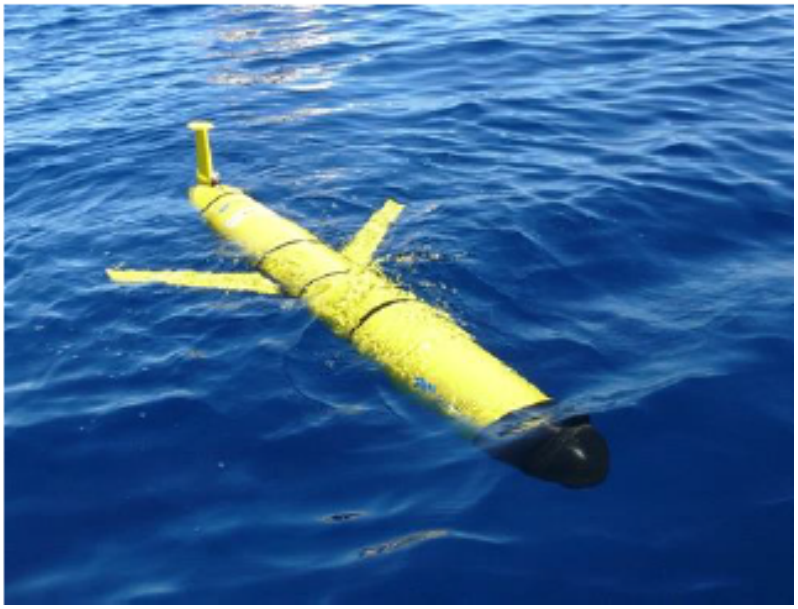

Figura 2.1: *Slocum Glider* em alto-mar [2]

O seu fornecimento energético provém, de baterias alcalinas, que possibilitam a sua manutenção em alto-mar durante períodos de 15 a 30 dias, podendo percorrer distâncias desde 600 km a 1500 km. Permite a instalação de sensores personalizados, havendo dois modelos dependendo da profundidade desejada: modelo de costa (profundidades entre 4-200 metros) e o modelo 1-km (para profundidades de operação de 1 km).

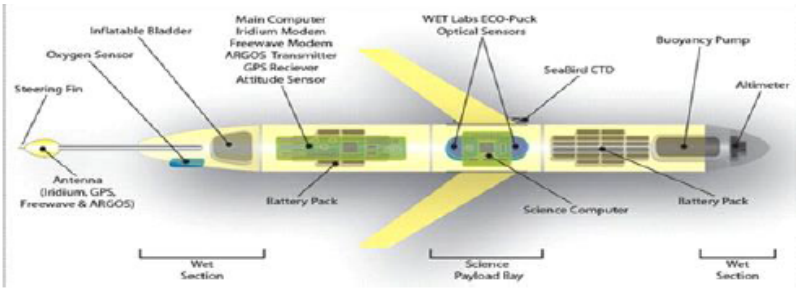

Figura 2.2: Estrutura constituinte do *Slocum Glider* [11]

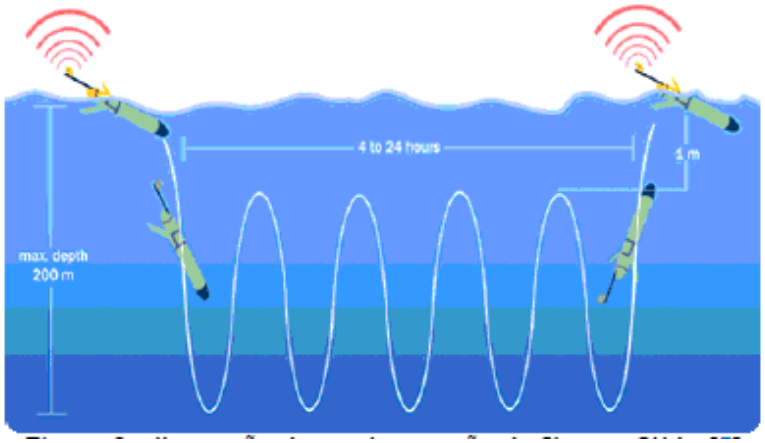

Figura 2.3: Ilustração da movimentação do *Slocum Glider* [2]

- **Características do *Slocum Glider***

Tabela 2.1: Especificações do *Slocum Glider*

Peso	52 Kg
Circunferência	21.3 cm
Comprimento	1.5 m
Alcance de Profundidade	4–200 m (modelo de costa) ou 1000 m (modelo 1-km)
Velocidade	0.4 m/s
Resistência	Média de 30 dias (depende de medições e comunicações)
Alcance	1500 km
Navegação	GPS, compasso magnético, altímetro, *subsurface dead reckoning*
Sensores	Condutividade, temperatura, profundidade
Comunicações	*RF modem, Iridium satellite, ARGOS, Telesonar modem*

- **Teoria operacional do veículo:**

Os *AUVG's* são bastante bons para percorrer longas distâncias e têm boa resistência, se operados a baixas-médias velocidades. O seu padrão de movimentação periódico é ótimo para observações nas colunas de água, quer para a vertical, quer para a horizontal. A sua superfície regular é excelente para a captura de *GPS* (*Global Positioning System* - Sistema de Posicionamento Global) e comunicações de duas vias, não sendo requisitadas quaisquer ajudas extras de navegação, sendo também o sistema bastante portátil. Os *Gliders* são únicos no mundo dos *AUV's*, na medida em que o seu deslocamento é feito por variações no balanceamento do veículo, com ajuda das asas e das superfícies de controlo que ajudam na velocidade de deslocamento do mesmo.

2.1.2 *Mares*

O *AUV Mares*, desenvolvido pelo grupo *The Ocean Systems Group (OceanSYS)* da Faculdade de Engenharia da Universidade do Porto juntamente com o IRS-Porto (Instituto de Sistemas Robóticos - Porto), é um robot submarino com 1,5 metros de comprimento e 20 centímetros de diâmetro, podendo atingir os 2 metros por segundo de velocidade horizontal máxima.

Segundo [1] este robot submarino, pode ser modificado, para comportar sensores capazes de obter informações acerca de um percurso conhecido, predefinido anteriormente. A novidade neste veículo é o seu propulsor, que foi configurado sem necessidade de superfícies de controlo, permitindo-lhe assim, percorrer trajetos que rasem a parte mais funda de um curso de água, ou como no caso da faculdade referida acima, num tanque pouco profundo, executando assim visualizações do que se queira observar, tem assim um movimento totalmente desacoplado nas direções vertical e horizontal, o que permite ao robot ficar totalmente parado submerso na água.

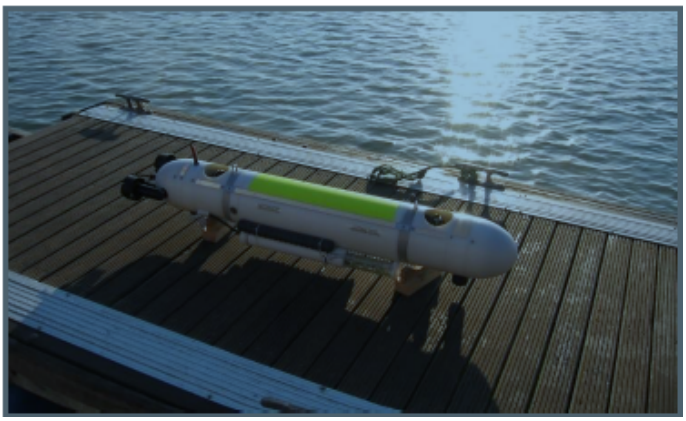

Figura 2.4: Mares em terra [11]

Figura 2.5: Estrutura do *Mares* [11]

- **Características do *Mares***

Tabela 2.2: Especificações do *Mares*

Peso	32 kg
Diâmetro	20 cm
Comprimento	1.5 m
Velocidade na horizontal	0 - 2 m/s
Autonomia	10 h (até 40 km)
Alcance de Profundidade	100 m

Apresenta grande flexibilidade devido à sua estrutura modular, podendo assim adaptá-lo a diferentes missões, adicionando/removendo módulos/extensões ao casco central.

Estas missões são programadas na sua *GUI* (*Graphical User Interface* - Interface Gráfica com Utilizador), onde se definem os objetivos/caraterísticas das mesmas, tais como velocidade, profundidade, pontos chaves do percurso, etc.. A sua localização é feita com recurso a balizas acústicas, fazendo-se o seu seguimento em tempo real.

É alimentado a baterias recarregáveis de ião-lítio que podem durar até 10 horas, relacionadas com a sua velocidade, podendo atingir 40 *Km*.

2.2 *Path Planning*

No planeamento de movimento ("*path planning*") de um robot, define-se um caminho/trajetória, entre dois pontos que se pretendam seguir, evitando colisões com obstáculos que surjam. Ora, este problema, pode ser de difícil resolução, devido a vários fatores, como sejam, o tamanho do robot, a sua forma e graus de liberdade. Para prevenir estas situações, segundo [14, 22, 21], pode-se usar *C-space* (*Configuration Space* – Espaço de Configuração), onde o robot é definido como apenas um ponto no espaço, denominando-se configuração ou estado. A dimensão do *C-space* é o número de parâmetros do estado, apresentando-se cada configuração como um vetor que carateriza o robot no ambiente envolvente, como a sua posição, a sua orientação, a sua energia, entre outros. Seguidamente, faz-se uma discretização do mapa do meio envolvente ao robot, para ser incorporado no algoritmo de planeamento de movimentação, havendo três meios de o fazer segundo [14]:

- ***Road Map:*** Sabendo a posição dos obstáculos por onde será efetuado o percurso do robot, definir caminhos no espaço envolvente que possibilitem contorná-los. O espaço disponível para o robot se movimentar, é o *C-space* do robot removendo os obstáculos.

- **Decomposição em células:** Faz-se a discretização do espaço disponível em células, verificando se cada célula se encontra livre ou cheia, permitindo passagem ou não do robot. De seguida ligam-se as células livres, definindo-se um grafo resultante com o percurso a fazer entre o ponto inicial e o ponto final.

- **Campo de Potencial:** Através de funções matemáticas aplicadas sobre o espaço, atrai-se o robot da posição final, ou repele-se o mesmo dos obstáculos, definindo-se assim o movimento do robot, e a trajetória a percorrer.

Findada esta etapa, o seguinte passo, segundo [22], consiste em planear o caminho do mapa obtido, e realizar os controlos requeridos para a movimentação. Não é obrigatoriamente uma solução para o problema; quando um algoritmo reporta inexistência de solução, ou cria um percurso sem haver colisão com obstáculos, diz-se que o mesmo é completo. Se se conseguir minimizar a energia gasta, ou realizar o percurso no menor tempo possível, dizemos estar na presença de um algoritmo ótimo (objetivo desta dissertação).

2.3 Modelo Geral de um Robot Submarino

Segundo [5] e [26], um robot submarino pode ser representado como se observa na figura seguinte:

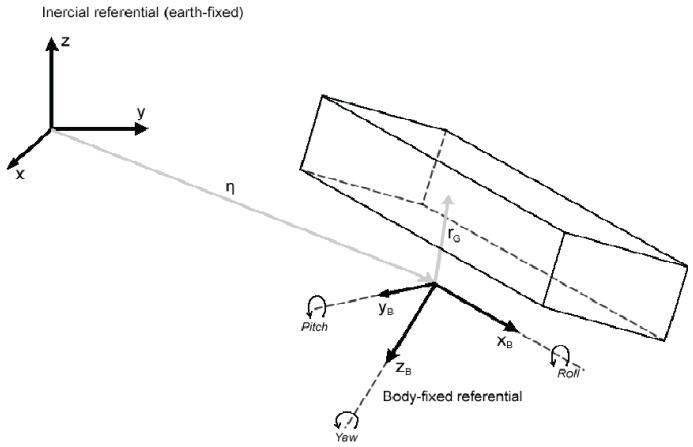

Figura 2.6: Representação do veículo e referencial [5]

Donde se pode definir:

$\eta_1 = [x\ y\ z]^T \longrightarrow$ é a posição da origem do submarino expressa no referencial inercial.

$\eta_2 = [\phi\ \theta\ \psi]^T \longrightarrow$ é a orientação do submarino em relação ao referencial inercial.

$\nu_1 = [u\ v\ w]^T \longrightarrow$ são as velocidades lineares expressas no submarino com respeito ao referencial inercial.

$\nu_2 = [p\ q\ r]^T \longrightarrow$ são as velocidades angulares expressas no submarino com respeito ao referencial inercial.

$\tau_1 = [X\ Y\ Z]^T \longrightarrow$ são as forças expressas no submarino.

$\tau_2 = [K\ M\ N]^T \longrightarrow$ são os momentos expressos no submarino.

$Roll\ (\dot{\phi}, pitch\ (\dot{\theta}), yaw\ ((\dot{\psi}))$ são as rotações que o submarino executa em torno do seu referencial, no eixo dos $xx's$, dos $yy's$ e dos $zz's$ respetivamente.

2.3.1 Equações de Cinemática

As equações de cinemática da movimentação de um AUV, tendo em conta aspetos geométricos podem-se definir como:

$$\begin{bmatrix} \dot{\eta}_1 \\ \dot{\eta}_2 \end{bmatrix} = \begin{bmatrix} R(\eta_1) & 0_{3\times3} \\ 0_{3\times3} & J(\eta_2) \end{bmatrix} \cdot \begin{bmatrix} \nu_1 \\ \nu_2 \end{bmatrix} \tag{2.1}$$

onde:

$$prcol = \begin{bmatrix} cos(\psi) \cdot cos(\theta) \\ sin(\psi) \cdot cos(\theta) \\ -sin(\theta) \end{bmatrix} \tag{2.2}$$

$$seccol = \begin{bmatrix} -sin(\psi) \cdot cos(\theta) + cos(\psi) \cdot sin(\theta) \cdot sen(\phi) \\ cos(\psi) \cdot cos(\theta) + sin(\psi) \cdot sin(\theta) \cdot sen(\phi) \\ cos(\theta) \cdot sin(\phi) \end{bmatrix} \tag{2.3}$$

$$thircol = \begin{bmatrix} sin(\psi) \cdot sin(\theta) + cos(\psi) \cdot sin(\theta) \cdot sin(\phi) \\ -cos(\psi) \cdot sin(\theta) + sin(\psi) \cdot sin(\theta) \cdot cos(\phi) \\ cos(\theta) \cdot cos(\phi) \end{bmatrix} \tag{2.4}$$

Assim:

$$R(\eta_1) = \begin{bmatrix} prcol & seccol & thircol \end{bmatrix} \tag{2.5}$$

Onde $R(\eta_1)$ é a matriz de rotação de $\{B\}$ referente a $\{I\}$, rodando em (x,y,z) com os ângulos *roll* (ϕ), *pitch* (θ) e *yaw* (ψ) e:

$$J(\eta_2) = \begin{bmatrix} 1 & sin(\phi) \cdot tan(\theta) & cos(\phi) \cdot tan(\theta) \\ 0 & cos(\phi) & -sin(\phi) \\ 0 & \dfrac{sin(\phi)}{cos(\theta)} & \dfrac{cos(\phi)}{cos(\theta)} \end{bmatrix} \tag{2.6}$$

Em que $J(\eta_2)$ relaciona as componentes da velocidade angular (expressas no referencial móvel, e que são p,q,r) com as derivadas temporais dos ângulos de *Euler*, ou seja $\dot{\phi}$, $\dot{\theta}$ e $\dot{\psi}$.

2.3.2 Equações da Dinâmica

Uma vez que as forças de propulsão e também as interações hidrodinâmicas com o meio ambiente são mais facilmente descritas no referencial móvel $\{B\}$, segundo [7] é habitual representar a dinâmica neste referencial:

$$M_{RB} \cdot \dot{\nu} + C_{RB}(\nu) \cdot \nu = \tau_{RB} \tag{2.7}$$

Onde M_{RB} é a matriz inercial de corpo rígido e $C_{RB}(\nu)$ é a matriz que inclui termos de força centrípeta e força de *Coriolis* (que é uma força inercial). O vetor τ_{RB} é um vetor geral de forças externas e momentos composta por:

$$\tau_{RB} = \tau + \tau_A + \tau_D \tag{2.8}$$

Onde τ é o vetor de forças e momentos gerados pelos propulsores do *AUV* e τ_A o vetor força e momento devidos à hidrodinâmica adicionada à massa do corpo:

$$\tau_A = -M_A \cdot \dot{\nu} - C_A(\nu) \cdot \nu \tag{2.9}$$

com $M_A = M_A^T > 0$ e $C_A(\nu) = -C_A^T(\nu)$.

Ter-se-á também de ter em conta as forças hidrodinâmicas causadas por arrasto (*drag*) e sustentação (*lift*).

$$\tau_D = -D(\nu) \cdot \nu \tag{2.10}$$

Onde $D(\nu) = D^T(\nu) > 0$ denota a matriz de amortecimento hidrodinâmico.

Das equações (2.7), (2.8) , (2.9) e (2.10) resulta:

$$M \cdot \dot{\nu} + C(\nu) \cdot \nu + D(\nu) \cdot \nu = \tau \tag{2.11}$$

Com $M = M_{RB} + M_A$ e $C(\nu) = C_{RB}(\nu) + C_A(\nu)$.

2.3.3 Equações Simplificadas de Movimento

Assumindo que o veículo se movimenta num plano horizontal apresenta-se de seguida o estado do sistema para a cinemática do submarino:

$$\begin{aligned} \dot{x} &= u \cdot cos(\psi) \; - \; v \cdot sin(\psi) \\ \dot{y} &= u \cdot sin(\psi) \; - \; v \cdot cos(\psi) \\ \dot{\psi} &= r \end{aligned}$$

Pode-se simplificar ainda mais as equações acima descritas pois, a forma usual dos corpos de *AUV's* (e outros veículos) faz com que "v"(velocidade lateral) seja muito inferior a "u"; há mesmo

situações, por exemplo movimento em linha reta em regime permanente, em que v é zero, e assim obtém-se:

$$\begin{aligned}\dot{x} &= u \cdot cos(\psi)\\ \dot{y} &= u \cdot sin(\psi)\\ \dot{\psi} &= r\end{aligned}$$

Estas equações simplificadas da cinemática, são as equações base para o desenvolvimento desta dissertação, que serão usadas a partir de agora no resto do documento.

Para o caso da dinâmica toma-se em conta o facto de existir uma força a atuar no controlador, a principal diferença é que no modelo cinemático, a velocidade longitudinal u é tratada como um controlo, enquanto no modelo dinâmico, esta velocidade passa a ser uma variável de estado e o controlo será uma força de propulsão; em ambos os casos a velocidade de rotação é sempre tomada como um controlo. Denomine-se f a força que afeta o controlo u exercida pelo propulsor, assim:

$$\begin{aligned}\dot{x} &= u \cdot cos(\psi)\\ \dot{y} &= u \cdot sin(\psi)\\ \dot{\psi} &= r\\ \dot{u} &= f - u \cdot |u|\end{aligned}$$

Quanto às unidades de medida deste sistema, tem-se u em m/s, r em rad/s, e a força f em N. A equação acima de $\dot{u} = f - u \cdot |u|$ obteve-se de uma equação mais complexa que é:

$$\dot{u} = \frac{1}{m} \cdot f - c \cdot u \cdot |u|$$

Em que c dado em kg/m é um coeficiente de $drag$ (arrasto) devido às correntes. Porém definiu-se que a massa do veículo tendo valor 1 e que o coeficiente também seria 1 pois não interessa para o problema em questão abordá-lo aprofundadamente.

Capítulo 3

Fundamentos Teóricos

Neste capítulo faz-se uma abordagem sucinta, à Teoria do Controlo Ótimo. Apresentam-se os modelos do sistema matemático; abordam-se os métodos de verificação de existência de solução, para um dado problema. Referem-se as regras para decidir as Condições Necessárias de Otimalidade. Apresenta-se a teoria de validação de resultados baseada no Principio do Máximo. Por fim, faz-se uma abordagem ao tipo de problemas utilizados neste projeto.

3.1 Teoria do Controlo

A Teoria do Controlo analisa o comportamento de um sistema dinâmico, e avalia a relação entre as entradas(*inputs*) e as saídas(*outputs*) do mesmo. O objetivo principal deste ramo é controlar (manipular) um sistema, para que, dada uma entrada de referência, se efetue uma variação controlada da mesma, com o intuito de se obterem as saídas pretendidas como se observa nas figuras 3.2 e 3.3.

3.1.1 Exemplificação

Apresenta-se agora um exemplo de [15], para uma melhor percepção do problema e de como abordá-lo.

- **Armazenamento e Fornecimento de Água:** Desde há muito tempo, que sistemas idênticos ao da figura 3.1 são usados em reservatórios de água. Há medida que o nível de líquido sobe, a bóia (float) vai regular a quantidade de água que entra no tanque (inlet flow); a mesma cessa se se ultrapassar um determinada altura h. Pode-se libertar água pela válvula de saída (outlet) a uma certa taxa, havendo um ajuste pela bóia para regular a quantidade de água a entrar no tanque, mantendo-se um nível desejado da mesma. Podemos aqui considerar um **sistema** formado por: bóia, água no reservatório, quantidade que entra e saí do mesmo, sendo o **controlo**, a posição da bóia. O **estado** a qualquer instante é um vetor, contendo a altura h de água no tanque, a taxa da corrente de água que entra e sai do sistema. Neste caso, temos um **estado do sistema** que ajusta automaticamente o **controlo** (posição da bóia),

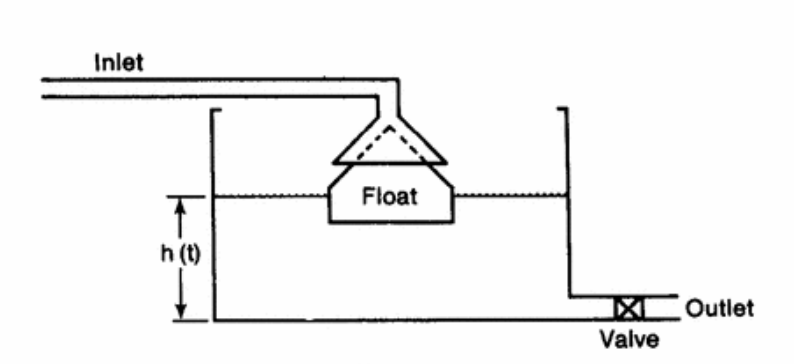

Figura 3.1: Reservatório de Água [16]

sendo um exemplo de um **sistema de controlo** com *feedback*; em que o **estado** dá o seu parecer ao **controlador** que faz os ajustes necessários ao **sistema** sem auxílio de terceiros.

3.1.2 Problemática

Consultando [16, 19, 18] , podemos fazer uma abordagem geral à modelação deste tipo de problemas. Faça-se a distinção entre:

- **Teoria do Controlo Clássico (Convencional):** baseia-se principalmente na Transformada de *Laplace*, para resolver sistemas do tipo SISO (*Single Input Single Output*), ou seja, sistemas com uma entrada e a respetiva saída; tal como se apresenta na figura 3.2.

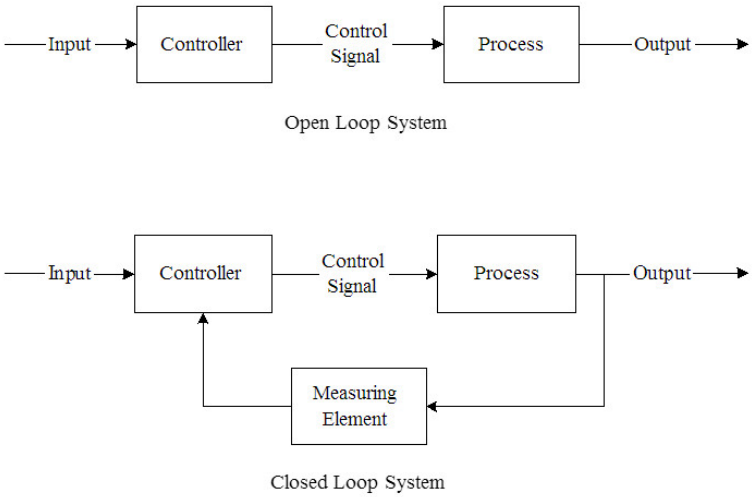

Figura 3.2: Controlo Clássico: malha aberta (em cima) e malha fechada (em baixo) [18]

Inicialmente nesta teoria, aplicavam-se controlos em sistemas de malha aberta (*open-loop system*), para resolver os problemas, nestes sistemas a entrada não depende da saída, referenciado em 3.2 no sistema superior (*open loop system*); porém a certa altura, era necessário

corrigir perturbações no sistema, e obter saídas controladas. Surgiu então o conceito de controlo de sistemas em malha fechada (*closed loop control*), onde o sistema é realimentado (efeito *feedback*) como se representa na figura 3.2 no sistema de baixo (*closed loop system*). Note-se que nesta teoria convencional, a entrada de um sistema é determinada com um erro e um compensador, porém nem todas as variáveis estarão disponíveis para estarão disponíveis para realimentar (*feedback*) o sistema, assim surge a Teoria de Controlo Moderno.

- **Teoria do Controlo Moderno:** contempla sistemas com múltiplas entradas e múltiplas saídas; é um sistema do tipo MIMO (*Multiple Inputs and Multiple Outputs*). Resolve-se este tipo de sistemas, recorrendo a equações diferenciais de primeira ordem, sendo caracterizado por variáveis de estado, de forma linear e invariante no tempo. Pode-se afirmar esta Teoria do Controlo Moderno dita que, todas as variáveis de estado podem ser realimentadas, depois de uma boa ponderação, observe-se a figura 3.3 .

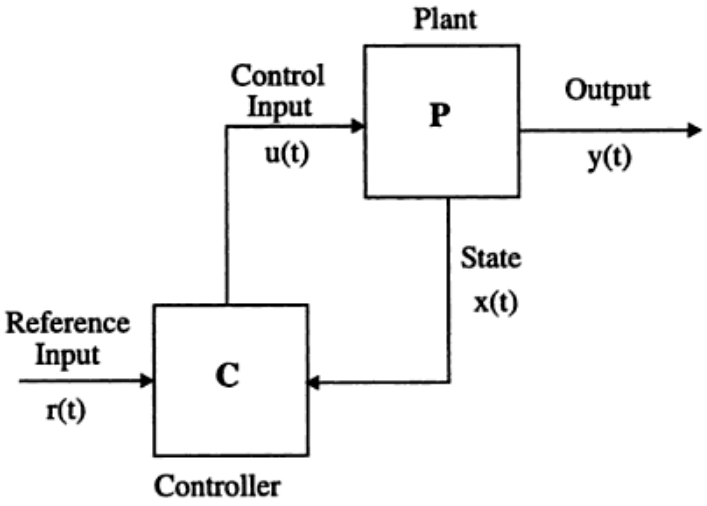

Figura 3.3: Configuração de Controlo Moderno [16]

A entrada **u(t)** é determinada pelo controlador, impulsionado pelo estado do sistema **x(t)** e pelo sinal de referência **r(t)**; a maior parte das variáveis de estado estão disponíveis para serem controladas. Se num universo de variáveis de estado de uma dada função de transferência, existe uma que especifica o nosso sistema de forma singular, obtém-se assim uma descrição mais completa desse mesmo sistema, com essa referida variável de estado. Na figura 3.4, mostram-se os componentes de um sistema de controlo moderno.

A primeira etapa de qualquer teoria de um sistema de controlo moderno é formular a dinâmica do nosso sistema, através das equações diferenciais dessa mesma dinâmica. A dinâmica do sistema é baseada na função *Lagrangiana*. Seguidamente o sistema é analisado ao nível da sua atuação, para se obter a estabilidade do sistema; estabilidade essa que pode ser confirmada com as contribuições de *Lyapunov*. Finalmente, se a atuação do sistema não

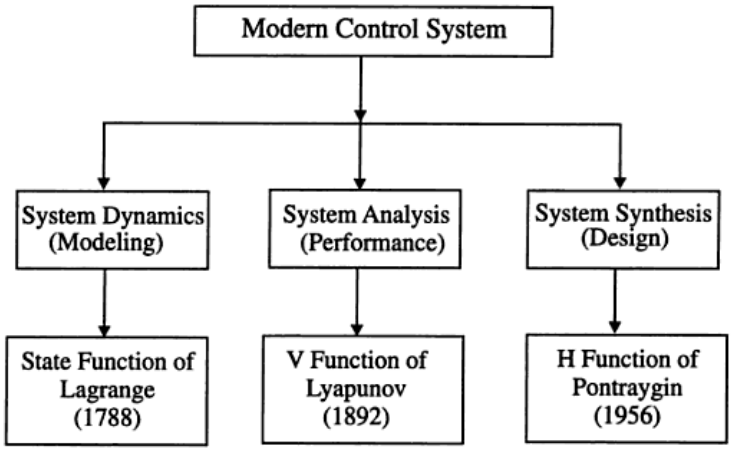

Figura 3.4: Componentes de um Sistema de Controlo Moderno Moderno [16]

se encontra dentro do especificado, pode-se recorrer às contribuições de *Pontyagrin* para validar os mesmo resultados.

3.2 Otimização

Hoje em dia há uma grande necessidade de otimização nos aspetos diários da vida atual. Por exemplo, há uma necessidade cada vez maior, de haver uma gestão otimizada do tempo disponível, gerindo-o da melhor maneira; ou então usar os recursos disponíveis em qualquer área da melhor maneira possível, e assim sucessivamente. Segundo [16, 20, 15, 6] o processo de otimização pode ser visto de várias maneiras, dependendo: **da abordagem ao problema** (algébrica ou geométrica); **do interesse** (único ou múltiplos interesses); **da natureza do sinal** (determinístico ou estocástico); **do estado do sistema** (único ou múltiplos estados), usados neste processo, tal como é representado na figura 3.5:

Pode-se observar que o "cálculo da variações", é somente um pequeno grão de areia que é esta praia do campo da otimização; todavia, é o que dá a forma ao nosso estudo do controlo ótimo. A otimização pode ser classificada como:

- **Otimização Estática:** controlar um sistema sobre condições de estado fixas, ou seja, as variáveis do sistema não se alteram ao longo do tempo; sendo o sistema descrito por equações algébricas, que podem ser resolvidas, recorrendo-se ao cálculo, multiplicadores *lagrangianos*, ou então a programação linear e não-linear.

- **Otimização Dinâmica:** designa problemas envolvendo sistemas dinâmicos, que é como dizer, sistemas com variáveis que se alteram ao longo do tempo, podendo o tempo estar envolvido na descrição do próprio sistema. Podem-se descrever esses sistemas, por equações diferenciais. O problema de otimização obtido poderá ser um problema de cálculo de variações ou de controlo ótimo.

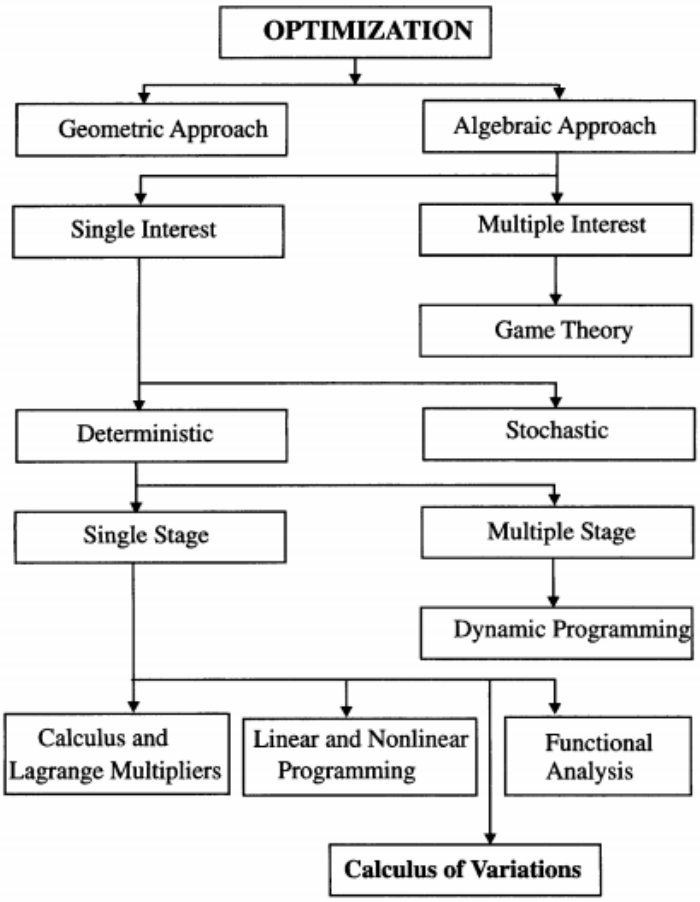

Figura 3.5: Processo de Otimização [16]

3.3 Controlo Ótimo Determinístico

3.3.1 Introdução

O objetivo da Teoria do Controlo Ótimo segundo [16, 20, 15, 6, 15] , é o de calcular uma função, do tempo, ou do tempo e do estado de um sistema dinâmico, que minimiza um funcional de desempenho restringido por vários fatores, englobam equações diferenciais ordinárias variável de estado e possivelmente, restrições estáticas no controlo e/ou estado. O controlo ótimo engloba:

- **Funcional de Desempenho:** Função objetivo que se quer minimizar (função custo) ou maximizar (função lucro), donde se retiram elações do funcionamento do sistema que, aliada à dinâmica do mesmo, especifica a forma de o controlar;

- **Equação Dinâmica do Sistema:** Um sistema dinâmico varia ao longo do tempo. A resposta do sistema em cada instante, depende da ação de controlo e do estado no instante inicial.

Esta equação, normalmente, diferencial ordinária, possibilita a determinação completa da evolução temporal da variável de estado;

- **Restrições no Controlo:** Define o leque de valores que o controlo pode tomar;
- **Restrições de Estado:** Condicionam o sistema, e deverão ser satisfeitas para a que a estratégia do controlo seja admissível.

O estado final e os instantes inicial e final do intervalo de tempo a considerar para cálculo do controlo ótimo podem ser ou não definidos à partida. No caso particular em que se pretende minimizar o tempo mínimo para a realização de uma tarefa (caso abordado nesta dissertação), o tempo final não se encontra definido, é uma variável. Dependendo da formulação do problema, estes fatores podem ser convertidos uns nos outros, aplicando mudanças de variável adequadas e/ou extensões de vetor do estado.

3.3.2 Formulação

Em [16, 20, 6, 15], no controlo ótimo quer-se determinar um controlo *u*(t)* ("*"indica condição ótima) que conduza um determinado sistema ("P") de um estado inicial até um estado final, tomando em consideração as restrições que se apliquem ao controlo e/ou ao estado, ao mesmo tempo que se extremiza uma função objetivo (maximizar - função lucro ou minimizar - função custo).

A formulação de um problema de controlo ótimo, pode-se dividir em:

- **I -** Modelo matemático do sistema a ser controlado;
- **II -** A função objetivo (F.O.);
- **III -** Restrições de controlo e/ou do estado do sistema;

Assim, e segundo [20], o problema do controlo ótimo, pode ser formulado como:

$$
\begin{aligned}
\textbf{(P) Minimizar} \quad & g(t_f, x(t_f)) \\
\textbf{através da escolha de um controlo} \quad & u : [0,T] \longrightarrow \mathbb{R}^m \\
\textbf{à qual corresponde uma trajetória} \quad & x : [0,T] \longrightarrow \mathbb{R}^n \\
\textbf{ambos satisfazendo} \quad & \dot{x}(t) = f(t, x(t), u(t)), \quad q.t. \quad t \in [0,T], && (3.1) \\
& x(0) = x_0, && (3.2) \\
& u(t) \in \Omega \qquad q.t. \quad t \in [0,T] && (3.3) \\
& x(t_f) \in C && (3.4)
\end{aligned}
$$

Defina-se:
q.t. como "para quase todo",
t_f é o tempo final da função custo, sendo $t_f < T$ (aqui T é um limite máximo),
$[0,T]$ é o intervalo definido para o problema,
$u : [0,T] \longrightarrow \mathbb{R}^m$ é a função controlo,
$x : [0,T] \longrightarrow \mathbb{R}^n$ é a trajetória,
$g : [0,T] \times \mathbb{R}^n \longrightarrow \mathbb{R}$,
$f : [0,T] \times \mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}^n$, descreve a dinâmica do sistema (3.1).

Noções Básicas:

- **Sistema Dinâmico:** Sistema que depende dos estímulos externos (*inputs*) e do valor do estado do sistema.
- **Trajetória:** $x : [0,T] \longrightarrow \mathbb{R}^n$, tal que x é absolutamente contínua e $\dot{x}(t) = f(t,x(t),u(t))$ pata todo o domínio $[0,T]$, ou seja, solução da equação diferencial (3.1) com a condição fronteira (3.2) para uma dada função de controlo que satisfaz (3.3).
- **Função de Controlo:** $u \in U$ tal que:

$$U = \{u : [0,T] \longrightarrow \mathbb{R}^m : u \textbf{ é mensurável, e } u(t) \in \Omega \textbf{ para todo o domínio } t \in [0,T]\}$$

3.3.2.1 Função Objetivo

Pode-se notar que num problema geral de Controlo Ótimo quer-se minimizar/maximizar uma Função Objetivo, função essa, que pode ser estruturados de três tipos diferentes:

- ***Bolza*:**

$$g(t_f,x(t_f)) + \int_0^{t_f} L(s,x(s),u(s))ds \tag{3.5}$$

- ***Lagrange*:**

$$\int_0^{t_f} L(s,x(s),u(s))ds \tag{3.6}$$

- ***Mayer*:**

$$g(t_f,x(t_f)) \tag{3.7}$$

3.3.2.2 Restrições

A nível das restrições tem-se:

- ***Estado Final*:**

$$x(t_f) \in C \tag{3.8}$$

- ***Estado inicial ou final*:**

$$(x(0), x(t_f)) \in C_0 \times C_1 \tag{3.9}$$

$$\tag{3.10}$$

ou

$$(x(0), x(t_f)) \in C \subset \mathbb{R}^n \times \mathbb{R}^n \tag{3.11}$$

- ***Estado*:**

$$h(t, x(t)) \leq 0 \ , \forall\, t \in [0,1] \tag{3.12}$$

- ***Estado e Controlo*:**

$$h(t, x(t), u(t)) \leq 0 \tag{3.13}$$

- ***Dinâmica*:**

$$\dot{x}(t) = f(t, x(t), u(t)) \tag{3.14}$$

- ***Mínimos globais e locais*:** O problema (P) pode ter soluções globais ou soluções locais. Diz-se que um processo admissível (t_f^*, x^*, u^*) é um ótimo global se $g(t_f^*, x^*(t_f^*)) \leq g(t_f, x(t_f))$ qualquer que seja o processo admissível (t_f, x, u) do nosso sistema.

  Muitas vezes, interessa determinar apenas ótimos "locais". Há muitas definições de ótimos locais. A mais usada é a seguinte: (t_f^*, x^*, u^*) é um ótimo local se minimizar o custo entre todos os processos (t_f, x, u) tais que $|t_f - t_f^*| < \varepsilon$ e $max|x(t) - x^*(t)| < \varepsilon, \quad \forall\, t \in [0, max\{t_f^*, t_f\}]$, e para algum $\varepsilon > 0$.

  Aqui supõe-se que $(x, x^*) : [0, max\{t_f^*, t_f\}] \longrightarrow \mathbb{R}$, com $x(t) = X(t_f)$ para $t \in [t_f, T]$ e $x^*(t) = x^*(t_f^*), \quad \forall\, t \in [0, max\{t_f^*, t_f\}]$.

3.3.3 Existência de Solução

Antes de resolver qualquer problema de controlo ótimo, é preciso saber se este tem solução. Consideram-se as seguintes hipóteses:

H1: $(t,x,u) \longrightarrow f(t,x,u)$ contínua e existe $M > 0 : |f(t,x,u)| \leq M \quad \forall (t,x,u) \in [0,T] \times \mathbb{R}^n \times \Omega$

H2: existe um K_f *constante*, tal que, para todo o $t \in [0,T]$:
$|f(t,x,u) - f(t,y,u)| \leq K_f \cdot |x-y|, \ \forall x,y \in \mathbb{R}^n, \ \forall u \in \Omega$

H3: C_0 é compacto

H4: Ω é compacto

H5: g é contínua

H6: existem processos admissíveis

H7: $f(t,x,\Omega) = \{f(t,x,u) : u \in \Omega\}$ é convexo para todo $(t,x) \in [0,T] \times \mathbb{R}^n$

Conclusão: Se **H1**-**H7** forem satisfeitas, então existe uma solução para um controlo ótimo do problema **(P)**.

3.3.4 Condições Necessárias de Otimalidade

Neste tópico apresentam-se as condições necessárias de otimalidade para problemas de controlo ótimo, na forma de um "Princípio do Máximo de *Pontyagrin*", retirado de [20, 16, 15, 6, 24].

$$\textbf{(P)} = \begin{cases} \text{minimizar} & g(t_f, x(t_f)) & \\ \text{sujeito a} & & \\ & \dot{x}(t) = f(t,x(t),u(t)) & \text{para todo } t \in [0,t_f] \\ & u(t) \in \Omega & \text{para todo } t \in [0,t_f] \\ & (x(0),x(t_f)) \in C_0 \times C_1 & \end{cases}$$

Seja (t_f^*, x^*, u^*) uma solução ótima local de (P) e suponha-se que as seguintes condições são satisfeitas:

H1': A função f é ontínua e $(t,x) \longrightarrow f(t,x,u)$ é continuamente diferenciável para todo o $u \in \Omega$

H2': g é continuamente diferenciável numa dada vizinhança de $(t_f^*, x^*(t_f))$

H3': Ω é compacto e C_0, C_1 são fechados e convexos

Então, segundo [24] existe um $\lambda \in [0,1]$ e funções absolutamente contínuas $p : [0,t_f^*] \longrightarrow \mathbb{R}^n$ e $q : [0,t_f^*] \longrightarrow \mathbb{R}$ tais que, para $H(t,x,u,p) = p(t) \cdot f(t,x,u)$:

[NT]: $$\lambda + |p(t_f)| \neq 0$$

[EA]: (1) $$\dot{q}(t) = \nabla_t H(t, x^*(t), p(t), u^*(t)) \quad \text{q.s } t \in [0, t_f^*]$$

(2) $$-\dot{p}(t) = \nabla_x H(t, x^*(t), p(t), u^*(t)) \quad \text{q.s } t \in [0, t_f^*]$$

[MH]: $$H(t, x^*(t), p(t), u^*(t)) = max_{u \in \Omega} H(t, x^*(t), p(t), u(t)) \quad \text{q.s } t \in [0, t_f^*]$$

[CT]: (1) $$q(t_f^*) = \lambda \cdot g_t(t_f^*, x^*(t_f^*))$$

(2) $$-p(t_f^*) \in \lambda \cdot g_x(t_f^*, x^*(t_f^*)) + N_{c_1}(x^*(t_f^*)$$

[CH]: (1) $$q(t) = p(t) \cdot f(t, x^*(t), u^*(t)) \quad \text{q.s } t \in [0, t_f^*]$$

(2) $$p(0) \in N_{C_0}(x^*(0))$$

Onde:

$p \longrightarrow$ multiplicador associado ao estado do sistema,
$\lambda \longrightarrow$ multiplicador do custo,
NT $\longrightarrow$ não trivialidade,
EA $\longrightarrow$ equação adjunta,
MH $\longrightarrow$ maximização do hamiltoniano,
CT $\longrightarrow$ condições de transversalidade,
CH $\longrightarrow$ constância do hamiltoniano.

Suponha-se agora que t_f é fixo. Neste caso $g(t_f, x(t_f)) = \tilde{g}(x(t_f))$
Então, as condições necessárias de otimalidade reduzem-se a:

[NT']: $$\lambda + |p(t_f)| \neq 0$$

[EA']: $$-\dot{p}(t) = \nabla_x H(t, x^*(t), p(t), u^*(t)) \quad \text{q.s } t \in [0, t_f^*]$$

[MH']: $$H(t, x^*(t), p(t), u^*(t)) = max_{u \in \Omega} H(t, x^*(t), p(t), u(t)) \quad \text{q.s } t \in [0, t_f^*]$$

[CT']: $$(p(0), -p(t_f)) \in \{0\} \times \lambda \nabla g(x^*(t_f)) + N_{C_0}(x^*(0)) \times N_{C_1}(x^*(1))$$

Ou seja, q não está presente nas Condições Necessárias de Otimalidade.

Capítulo 4

Software Utilizado

4.1 AMPL

AMPL (*A Mathematical Programming Languge* - Programação de Linguagem Matemática) segundo [8, 9], é uma linguagem de modelação algébrica para descrever e resolver problemas de computação matemática com altos níveis de complexidade, neste caso, problemas de otimização. Pode conter diversos *solvers* (termo genérico para indicar uma parte de software matemático que resolve ("solves") um problema da mesma natureza), entre os quais se encontra o *IPOPT*, que é utilizado nesta dissertação devido a ser o mais aconselhado para resolver problemas de programação não linear, e que é descrito mais à frente em 4.1.1.

Uma das grandes vantagens deste software é a semelhança ao nível da sintaxe da notação matemática para problemas de otimização; o que permite uma definição dos mesmos de forma concisa e de fácil compreensão, sendo por isso, um programa bastante usado nos casos de programação matemática. O *AMPL* representa uma mistura de estilos de programação declarativa e imperativa, formulando-se os problemas a otimizar através de conjuntos de elementos, escalares, vetores multidimensionais, variáveis de decisão, objetivos e restrições, que permitem uma descrição sucinta da maioria dos problemas de otimização matemática.

Este software está disponível para funcionar em diversos sistemas operativos, incluindo o ambiente Linux, que foi o escolhido para trabalhar neste projeto.

Existe na internet, uma versão da ferramenta *AMPL* (*AMPL Student edition*), que possibilita a qualquer pessoa usufruir deste software gratuitamente; no entanto esta mesma versão, terá algumas restrições, ao nível de variáveis e iterações que o programa poderá fazer, o que implica que os resultados obtidos possam nem sempre ser os melhores possíveis.

No local de realização do projeto (Faculdade de Engenharia da Universidade do Porto), há uma versão completa do *AMPL*, que em nada restringe a ferramenta, podendo-se usar quantos parâmetros, variáveis e outros componentes, necessários à otimização matemática nos modelos definidos.

4.1.1 *IPOPTS*

Não é do interesse desta dissertação um estudo aprofundado do funcionamento desta ferramenta, aqui refere-se só uma conceptualização muito simplificado do mesmo, para tal indica-se a bibliografia [12], para um estudo mais profundo da mesma.

IPOPT (*Interior Point OPTimizer*), é uma libraria de software "opensource"(*Open Source Software - OSS* - é *software* computacional, que tem o seu código fonte com a licença disponível, onde a entidade que detêm os direitos de autor, fornece esses mesmos direitos, para casos de estudo, onde qualquer pessoa pode aplicar e editar o software, para qualquer situação desejável), para otimização linear de sistemas contínuos. Pode ser usado para resolver problemas gerais de programação não-linear da forma:

$$\begin{aligned} min_{x\in\mathbb{R}^n} \quad & f(x) \\ sujeito\ a \quad & g^L \leq g(x) \leq g^U \\ & x^L \leq x \leq x^U \end{aligned}$$

Onde $x \in \mathbb{R}^n$ são as variáveis de otimização (possivelmente com limites máximo x^U e mínimo x^L: $x^L \in (\mathbb{R} \cup \{-\infty\})^n$ e $x^U \in (\mathbb{R} \cup \{+\infty\})^n$), $f : \mathbb{R}^n \longrightarrow R$ é a função objetivo e $g : \mathbb{R}^n \longrightarrow \mathbb{R}^n$ são as restrições do problema.

As funções $f(x)$ e $g(x)$ podem ser lineares ou não-lineares e convexas ou não (mas devem ser duas vezes continuamente diferenciáveis). As restrições $g(x)$ têm limites máximo e mínimos ($g^L \in (\mathbb{R} \cup \{-\infty\})^m$ e $g^U \in (\mathbb{R} \cup \{+\infty\})^m$).

O *IPOPT* implementa um método de procura por "interior point-line", para encontrar a solução local. Este tópico não é de grande interesse aprofundar muito o seu funcionamento, sendo que se mais informação o leitor desejar acerca do tema, poderá consultar [17, 25].

4.2 Exemplificação do Ficheiro em *AMPL*

Antes de abordar o conteúdo de um ficheiro tipo para resolver em *AMPL* comece-se por fazer uma pequena apresentação de um método iterativo que se visa usar nos casos apresentados mais à frente nesta dissertação. Não se pretende aprofundar este tipo de conhecimentos nesta dissertação, como tal se o leitor desejar conhecer melhor, este e outros tipos de métodos para resolver este tipo de problemas, refere-se aqui [13].

4.2.1 Método de *Euler*

Quer-se aproximar a solução de um problema de valor final:

$$\dot{x}(t) = f(t, x(t)), \quad x(t_0) = x_0 \tag{4.1}$$

Denote-se:

$h \longrightarrow$ tamanho de um passo na divisão temporal t;

$t_n = t_0 + n \cdot h \longrightarrow$ atribuição de um ponto do intervalo a cada passo.

O próximo passo t_{n+1} a partir do passo anterior t_n fica: $t_{n+1} = t_n + h$, logo:

$$x_{n+1} \; = \; x_n + h \cdot f(t_n, x_n).$$

Assim, para um h menor, existirão mais passos no intervalo; porém terá melhor aproximação do valor analítico.
x_n é uma aproximação da solução da Equação Diferencial Ordinária no ponto $t_n : x_n \approx x(t_n)$.
Este método é explícito, implicando que x_{n+1} é função explícita de x_i para $i \leq n$.

4.2.2 Apresentação do Ficheiro a Usar no *AMPL*

Aqui estrutura-se o modelo do ficheiro que o *AMPL* vai percorrer para resolver um determinado problema. Refira-se que no *AMPL*, é necessário que o utilizador faça uma discretização de linha temporal requerida. Como tal divide-se um determinado período de tempo, em intervalos variados, conforme o requerido, e vai-se percorrer cada intervalo, fazendo iterações para se obterem resultados e os erros associados, em cada ponto da barra temporal, que se dividiu previamente. Denote-se como $h = T/n$, em que h é o valor referente ao tempo total T da nossa linha temporal, a dividir pelo número de divisórias, n, desejadas para a mesma. Existem vários métodos iterativos, para resolver problemas do tipo pretendido nesta dissertação. Devido à sua simplicidade e elevado grau de eficácia, usa-se, neste projeto, o "Método de *Euler*", sendo bastante usado e referenciado, para resolver interativamente equações matemáticas, apresentado na secção 4.2.1.

Assim apresenta-se a constituição do ficheiro, não sendo necessário perceber o contexto do problema nem o tipo de problema que é, apenas se quer dar ao leitor uma contextualização e uma ideia do tipo de linguagem usada na implementação deste ficheiro, que corre com o *AMPL*; constituição essa, que será um guia dos casos abordados no capítulo 5 ao nível de estrutura e linguagem:

- Primeiro definem-se os **parâmetros** do problema, que são os valores fixos do problema, não se alteram esses valores. Um exemplo de definição de parâmetro:

$$param\ X = 0;$$

- **Variáveis de Estado:** Neste ponto definem-se os valores que vão variar ao longo do problema, implicadas no estado so sistema. E a linguagem aplicada é:

$$\begin{aligned} var\ Y >= 0; \\ var\ X\ \{i\ in\ 0..n\}; \\ s.t\ IX1\ :\ X[0] = 40; \end{aligned}$$

Nota: *s.t.* (*"subject to"*) significa "sujeito a".

- **Variáveis de Controlo :** Aqui definem-se os valores que controlam o estado do sistema:

$$var\ U\{in\ 0..n\};$$

- **Função Objetivo:** Nesta parte defini-se a função custo ou função objetivo do problema, o que se pretende minimizar ou maximizar. E escreve-se da seguinte maneira:

$$minimize/maximize \qquad OBJ;$$

Em que "*OBJ*", será a função objetivo pretendida.

- **Equações Diferenciais Ordinárias (Estado do Sistema):** Aqui definem-se as equações diferenciais do estado do sistema; e definem-se como:

$$var\ func\_X\ \{i\ in\ 0..n\}\ =\ U[i]\ \times cos(psi[i]);$$

- **Método de Euler:** Neste tópico aplicamos o método para fazer as iterações nos pontos pretendidos, e saber o valor das equações de estado nesse ponto. Usa-se o Método de Euler indicado em 4.2.1.

$$s.t.\ lX\ \{i\ in\ 0..n\}\ =\ X[i+1] = X[i] + h \times func\_X[i];$$

- **Restrições de Controlo:** São os limites das variáveis de controlo do sistema, ou restrições que se deseje adicionar ao sistema, para cumprir certos requisitos ao nível dos controladores.

$$s.t.\ mu\_U\ \{i\ in\ 0..n-1\}\ :\ 0\ <=\ U[i]\ <=\ 2;$$

- **Estimativa Inicial:** Valores que se supõe/estimam, aplicáveis ao sistema no ponto inicial.

$$let\ \{i\ in\ 0..n\}\ \ X[i]\ :=\ 0;$$

- ***Solver*:** Como referido em 4.1.1, neste projeto o *solver* aplicado é o *IPOPT*, como tal no ficheiro define-se:

$$option\ solver\ ipopt;\ solve;$$

  Nesta parte tem-se ainda a opção de definir o número máximo de iterações que o *AMPL* deve efetuar e um erro aceitável; sendo que estes valores diferenciam-se para cada problema que se pretenda estudar. E assim acrescenta-se ao código acima, antes de "*solve*;":

$$option\ ipopt\_options\ "max\_iter = 9999\ \ acceptable\_tot\ = 1e-8";$$

- Finalmente obtêm-se os valores que se pretendem imprimir, guardando-se os mesmos num ficheiro "*.dat*";

$$printf\ "\%18.10f",\ OBJ\ >\ 'obj.DAT';$$

  e mostra-se o valor da função objetivo:

$$display\ \ OBJ; \tag{4.2}$$

4.3 *Matlab*

Segundo [10], o *Matlab* é um programa de cálculo numérico que se pode usar interativamente. Tem como estrutura de dados fundamental a aplicação matricial, ou seja, utiliza matrizes por base. Contém diversas *toolboxes* (ferramentas de trabalho) que possibilitam a sua aplicação em diversos domínios.

Neste projeto, utiliza-se o *Matlab*, para desenhar os gráficos com os resultados obtidos do ficheiro a correr no *AMPL*, referenciado na secção 4.1.

Capítulo 5

Planeamento das Trajetórias

Neste capítulo apresentam-se dois problemas de planeamento de trajetórias recorrendo a controlo ótimo: tempo mínimo e energia mínima. Estes problemas são resolvidos numericamente e os resultados numéricos são validados. A validação dos resultados consiste em mostrar que a solução numérica satisfaz o Princípio do Máximo.

Observe-se que os valores numéricos dos multiplicadores são dados pelo *software* usado, que se abordou no capítulo 4.

5.1 Introdução aos Casos 1 e 2

Comece-se por referenciar como **Caso 1** e **Caso 2** a situação mais simplificada da movimentação de um *AUV*. No primeiro caso, pretende-se que o submarino percorra um determinado percurso no mínimo tempo possível. Já no segundo caso, quer-se que o mesmo cumpra a mesma tarefa num tempo fixo mas com o menor consumo de energia possível.

Supõe-se aqui, que o submarino é apenas um ponto no mapa, ou seja, uma partícula que percorrerá um determinado percurso desejado. As equações referentes ao movimento simplificado de um *AUV*, já abordadas no capítulo 2, podem ser simplificadas da forma:

$$\begin{aligned} \dot{x} &= u \cdot \cos(\psi); \\ \dot{y} &= u \cdot \sin(\psi); \\ \dot{\psi} &= r; \end{aligned}$$

Onde:

$\dot{x}, \dot{y}, \dot{\psi} \longrightarrow$ correspondem ao estado do sistema;

$u, r \longrightarrow$ correspondem ao controlo do sistema;

$u \longrightarrow$ é a velocidade linear do submarino em relação ao eixo das ordenadas ($xx's$), dada em metros por segundo ($m \cdot s^{-1}$);

$r \longrightarrow$ é a velocidade angular do submarino em torno do eixo dos $zz's$, dada em radianos por segundo ($rad \cdot s^{-1}$);

$\psi \longrightarrow$ é o ângulo que o submarino descreve ao rodar em torno do seu eixo dos $zz's$, dada em radianos (rad).

x e $y \longrightarrow$ descrevem a distância percorrida pelos submarino, nos eixos $xx's$ e $yy's$, respetivamente, dada em metros (m).

O percurso do *AUV* em todos os casos aqui apresentados, inicia-se no ponto (40,-2), e termina no ponto (0,0). Foram efetuados mais exercícios com diferentes posições de partida e chegada, apresentados de forma mais simplificada, sem apresentar todas as justificações, pois aplicam-se sempre os mesmos métodos que se aplicaram no estudo dos casos mais aprofundados.

Faz-se notar também que a velocidade máxima a considerar para um *AUV*, neste trabalho será de 2 $m \cdot s^{-1}$, e o ângulo de rotação em torno do seu eixo dos $zz's$ varia de com uma velocidade $-\pi$ *a* π dada em *radianos por segundo* ($rad \cdot s^{-1}$).

5.1.1 Caso 1 - Tempo Mínimo

Os ficheiros para este caso; quer o ficheiro que corre no *AMPL*, quer o ficheiro que serve para desenhar os gráficos em *Matlab*, podem ser consultados no anexo A.

5.1.1.1 Formulação do Problema

Quer-se então:

$$
\begin{aligned}
&\textbf{\textit{Minimizar:}} && t_f \\
&\textbf{\textit{sujeito a:}} && \dot{x}(t) = u(t)\cdot\cos(\psi(t)); \\
& && \dot{y}(t) = u(t)\cdot\sin(\psi(t)); \\
& && \dot{\psi}(t) = r(t); \\
& && (x(0),y(0),\psi(0)) = (40,-2,\pi) \\
& && (u(t),r(t)) \in [0,2]\times[-\pi,\pi]; \qquad\qquad q.t.\ t \in [0,t_f] \\
& && (x(t_f),y(t_f),\psi(t_f)) \in B_r \times \mathbb{R} \qquad onde\ B_r = \{(x,y) : x^2+y^2 \leq r\}
\end{aligned}
$$

Note-se que não se deseja chegar ao ponto $(0,0)$ mas sim a uma dada vizinhança do mesmo definida pela bola B_r

5.1.1.2 Resultados Numéricos

Correndo o *AMPL*, com as condições apresentadas em 5.1, para este caso de minimização de tempo, obtiveram-se os resultados analíticos apresentados a seguir.

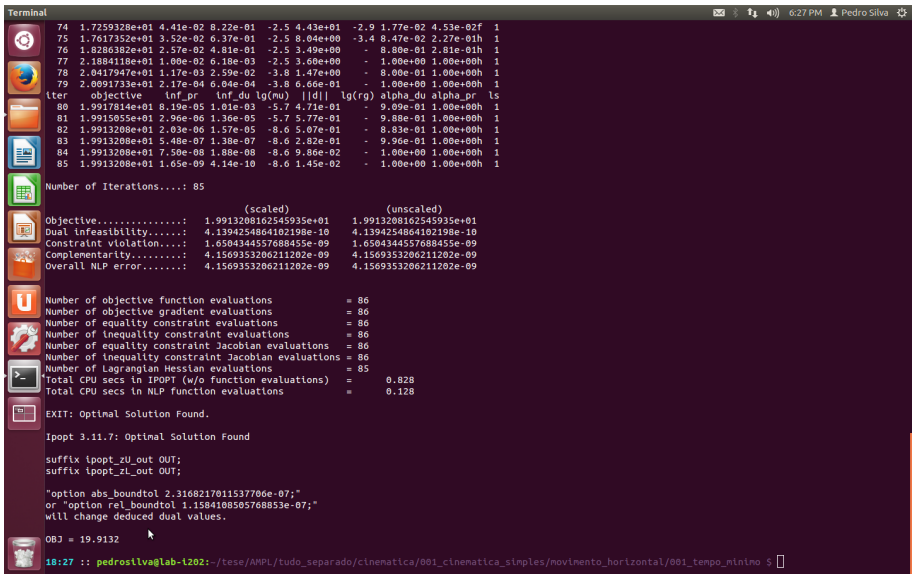

Figura 5.1: *AMPL* a correr na linha de comandos ambiente *Linux*

Da figura (5.1), retira-se o valor do tempo mínimo ótimo para o percurso do submarino, neste caso, sendo ele de 19.9132 *segundos*.

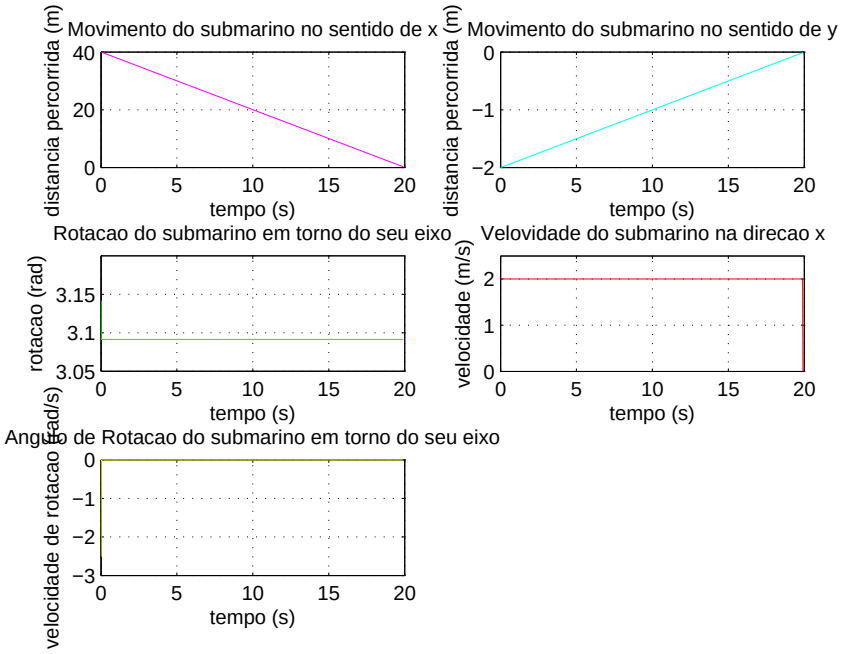

Figura 5.2: Estados e Controlos Ótimos para o Caso 1

Observar-se, na figura 5.2 a distância percorrida e a forma como ela se altera ao longo do percurso nos gráficos do "Movimento do submarino no sentido de x e y"(figuras ilustradas a magenta

e azul ciano, respetivamente). Pode-se ainda verificar a rotação que tem em torno do seu eixo, para fazer um direcionamento para o ponto objetivo na figura ilustrada a verde.

Observam-se também na figura, os gráficos das velocidades linear (gráfico a vermelho) e angular (gráfico a amarelo) do robot.

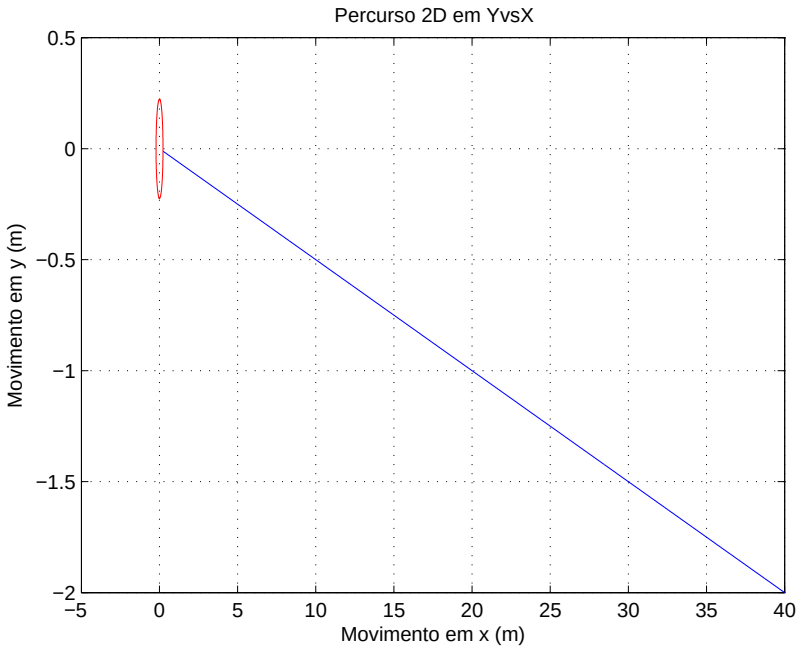

Figura 5.3: Percurso X vs Y

Note-se que as escalas de x e y são diferentes.

Finalmente, na figura 5.3, apresenta-se a movimentação do submarino a duas dimensões ($XvsY$). Pode-se observar a distância percorrida e a correção da sua rotação em relação ao eixo dos $zz's$.

Definiu-se uma circunferência como local de chegada do *AUV*, pois, na análise de controlo ótimo, chegar ao ponto $(0,0)$ (ponto pretendido) é quase impossível, havendo sempre um erro, por mais pequeno que seja, associado. O submarino atingiu o alvo, tocando na periferia do círculo. Sabendo o raio da circunferência, é fácil calcular a distância ao centro, ou seja, ao ponto pretendido.

Validação dos Resultados Para validar os resultados, usam-se as condições necessárias de otimalidade enunciadas em 3.3.4. Para o fazer é preciso extrair os valores numéricos dos multiplicadores p_x, p_y, p_ψ *e* q.

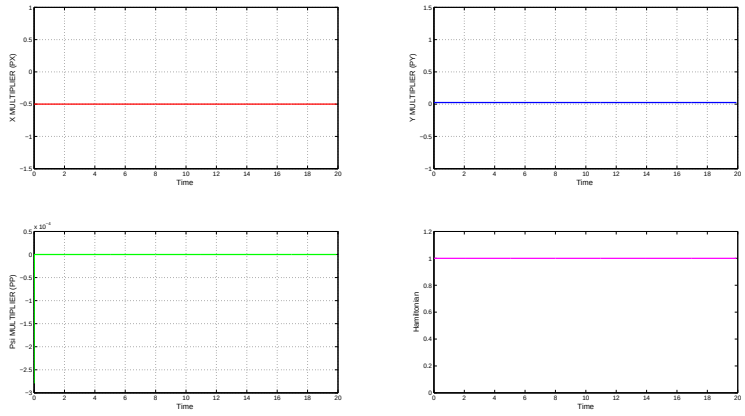

Figura 5.4: Multiplicadores das equações de estado

No caso 1, tem-se:

$$H = p \cdot f(t,x,u) =< (p_x, p_y, p_\psi) \cdot (u \cdot cos(\psi), u \cdot sen(\psi), r) >= \\ = p_x \cdot u \cdot cos(\psi) + p_y \cdot u \cdot sen(\psi) + p_\psi \cdot r$$

Verifica-se pelos multiplicadores apresentados em cima, que a primeira condição [**NT**] é válida.

De [**EA**] obtém-se:

$$-\dot{p}_x = \nabla_x H = 0 \quad \longrightarrow \quad p_x = constante$$
$$-\dot{p}_y = \nabla_y H = 0 \quad \longrightarrow \quad p_y = constante$$
$$-\dot{p}_\psi = \nabla_{\psi^*} H = -p_x \cdot u^* \cdot sen(\psi^*) + p_y \cdot u^* \cdot cos(\psi^*)$$
$$\dot{q} = \nabla_t H = 0$$

Note-se que $C_0 = \{(40, -2, \pi)\}$, $C_1 = \{(x,y) : x^2 + y^2 \leq 0.05\} \times \mathbb{R}$, $g(t_f, x(t_f)) = t_f$.

Assim, de [**CT**], obtém-se $(-q(0), p_x(0), p_y(0), p_\psi(0)) = (\xi_0, \xi_1, \xi_2, \xi_3)$ onde $\xi_i \in \mathbb{R}$, $i = 0,1,2,3$ e $(q(t_f), -p_x(t_f), -p_y(t_f), -p_\psi(t_f)) \in \lambda(1,0,0,0) + \{0\} \times N_{C_1}(x^*(t_f^*), y^*(t_f^*)) \times \{0\}$. Assim $q(t_f) = \lambda$.

Seja $(\eta_1, \eta_2) \in N_C(x^*(t_f^*), y^*(t_f^*)$ tais que $p_x(t_f) = -\eta_1$ e $p_y(t_f) = -\eta_2$.

Assim, tem-se $p_x(t_f) = -\eta_1 = \xi_1$ e $p_y(t_f) = -\eta_2 = \xi_2$

Observe-se na figura 5.4 que os multiplicadores p_x e p_y são realmente constantes e aproximadamente iguais a -0.5 e 0.

Já para p_ψ não se verifica o mesmo. Obtém-se $p_\psi(0) = \xi_3 \in \mathbb{R}$ e

$$\dot{p}_\psi(t) = -p_x(t) \cdot u^*(t) \cdot sen(\psi^*(t)) + p_y(t) \cdot u^*(t) \cdot cos(\psi^*(t)) = \\ = -\xi_1 \cdot u^*(t) \cdot sen(\psi^*(t)) + \xi_2 \cdot u^*(t) \cdot cos(\psi^*(t))$$

Por [**CH**] sabe-se que

$$H(t,p(t),x^*(t),u^*(t)) \;=\; q(t) \quad \forall\, t \;\in\; [0,t_f]$$

e assim tem-se que $q(t)$ é constante e igual a λ, o valor de λ e q fica determinado calculando $H(t,p(t),x^*(t),u^*(t))$. A função H é traçada na figura 5.4. Como se pode ver é constante e igual a 1. Verifica-se pois que $q(t) = 1 = \lambda$.

De [**MH**] Maximiza-se o Hamiltoniano:

$$H(t,x^*,p,u^*,r^*) \geq \{H(t,x^*,p,u,r)\}, \quad u \;\in\; [0,2],\; r \;\in\; [-\pi,\pi];$$

Assim (ocultando a dependência de t):

$$\begin{aligned} p_x \cdot (u^* \cdot cos(\psi^*)) + p_y \cdot (u^* \cdot sen(\psi^*)) + p_\psi \cdot r^* \geq \\ \geq p_x \cdot (u \cdot cos(\psi^*)) + p_y \cdot (\cdot sen(\psi^*)) + p_\psi \cdot r \quad u \;\in\; [0,2],\; r \;\in\; [-\pi,\pi], \end{aligned}$$

O que equivale a

$$p_x \cdot cos(\psi^*) \cdot (u^* - u) + p_y \cdot sen(\psi^*) \cdot (u^* - u) + p_\psi \cdot (r^* - r) \geq 0 \quad u \;\in\; [0,2],\; r \;\in\; [-\pi,\pi] \tag{5.1}$$

Se se fixar $u \;=\; u^*$ em cima, tem-se:

$$p_\psi \cdot (r^* - r) \geq 0, \quad \forall\, r \;\in\; [-\pi,\pi] \tag{5.2}$$

Ou seja

$$r^*(t) = \begin{cases} singular & se \quad p_\psi = 0 \\ \pi & se \quad p_\psi > 0 \\ -\pi & se \quad p_\psi < 0 \end{cases}$$

Nota: Singular significa que o valor da função nesse ponto, fica entre os extremos.

Observe-se o gráfico de r na figura 5.2, e o gráfico de p_ψ da figura 5.4, e conclui-se que este passo é corroborado pelos mesmos.

Fixando $r \;=\; r^*$ em (5.1) tem-se

$$(p_x \cdot cos(\psi^*) + p_y \cdot sen(\psi^*)) \cdot (u^* - u) \geq 0$$

Seja $func = (p_x \cdot cos(\psi^*) + p_y \cdot sen(\psi^*))$. Tem-se assim

$$u^*(t) = \begin{cases} singular & se \quad Func = 0 \\ \text{máximo} & se \quad Func > 0 \\ \text{mínimo} & se \quad Func < 0 \end{cases}$$

O gráfico seguinte da figura 5.5, apresenta a função $func$:

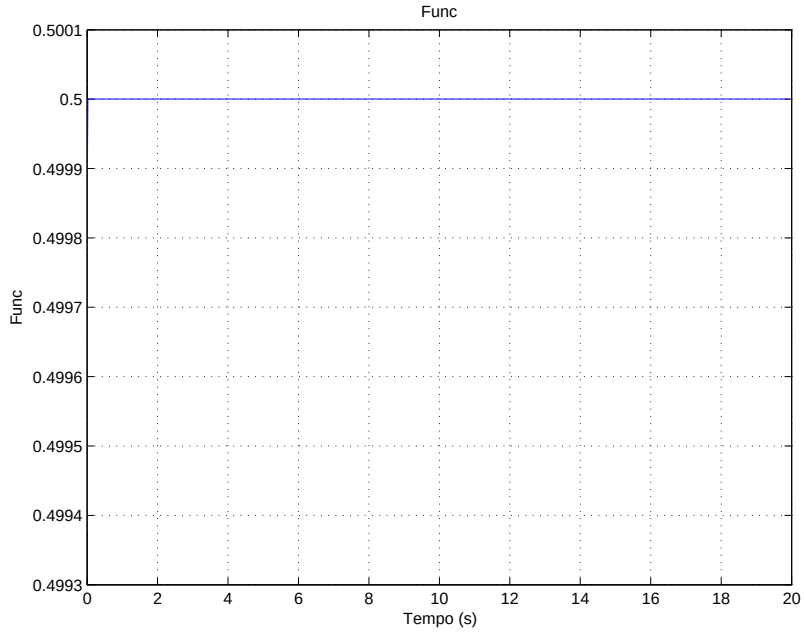

Figura 5.5: Função $func$

Como se visualiza na figura 5.5, $func > 0$. Então o controlo u deverá estar sempre no valor máximo; e é isso que se verifica na figura 5.2.

Mais informação:

$$\begin{aligned} t_f &= 19.9132 \\ x(t_f) &= 0.223327 \\ y(t_f) &= -0.011174 \\ \psi(t_f) &= 3.091600 \\ \eta_1 &= 50.155238 \\ \eta_2 &= -2.509480 \end{aligned}$$

Verifica-se ainda que

$$x(t_f)^2 + y(t_f)^2 \leq 0.05$$

Valida-se este caso, pois tem-se $x(t_f)^2 + y(t_f)^2 = 0.05$. Logo o ponto atinge a fronteira da circunferência que se determinou.

No ponto $(x(t_f), y(t_f))$ a normal da circunferência $x^2 + y^2 \leq raio^2$, (η_1, η_2) tem primeira componente grande e a segunda componente mais pequena. Daí se poder observar que, essa mesma normal será um vetor a apontar para fora da circunferência, com ângulo $tan^{-1}(\frac{y(t_f)}{x(t_f)}) = -0.0500$. Conclui-se que o vetor normal à tangente se situa no quarto quadrante.

5.1.2 Outros Exemplos idênticos ao Caso 1

Nesta secção apresentam-se outros casos estudados de Tempo Mínimo, sendo que se mudaram os valores iniciais e finais do problema mostrado em 5.1.1.

Nesta parte não se justificará validar os resultados por extenso, da explicação das equações do Principio do Máximo, pois são em tudo iguais ás apresentadas no 5.1.1, daí não ser necessário haver uma repetição do que já foi explicado.

Assim, obteve-se a seguinte figura com todos os percursos abordados:

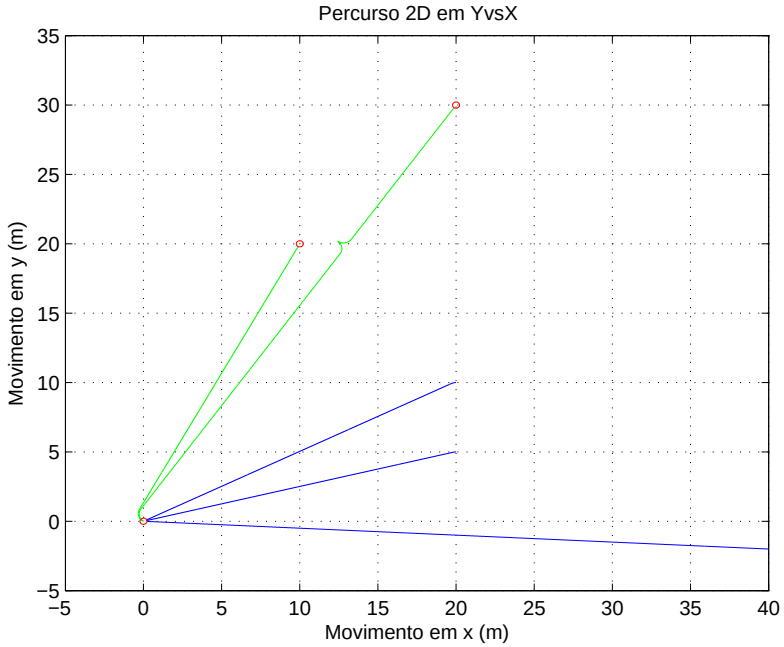

Figura 5.6: Todos os percursos do Caso 1 (Plano Y vs X)

Na figura 5.6, apresentam-se: a azul, todos os casos com ponto de chegada à vizinhança de $(0,0)$, e a verde os casos com ponto de partida de $(0,0)$.

O seguinte quadro (5.1) refere os valores específicos de cada situação:

Tabela 5.1: Resultados dos Casos de Estudo de Tempo Mínimo Simples

t_f	**Ponto Partida**	**Ponto Chegada**	$x(t_f)$	$y(t_f)$	**Objetivo:** $x_f^2+y_f^2 \leq raio^2$
19.9132	(40,-2)	(0,0)	0.223327	-0.011174	0.0500
11.0744	(20,10)	(0,0)	0.199672	0.100654	0.0500
10.2221	(20,5)	(0,0)	0.167991	0.042178	0.0300
18.3025	(0,0)	(20,30)	19.886133	29.835579	0.0400
11.408	(0,0)	(10,20)	9.894298	19.802954	0.0500

5.1.3 Caso 2 - Consumo de Energia Mínimo

Os ficheiros para este caso; quer o ficheiro que corre no *AMPL*, quer o ficheiro que serve para desenhar os gráficos em *Matlab*, podem ser consultados no anexo B.

5.1.3.1 Formulação do Problema

A grande quantidade de energia dispendida pelo *AUV* deve-se, quase na sua totalidade ao controlo que afeta a velocidade do mesmo na direção *xx*. Quanto menor é u, menor é a energia. Tendo isto em mente, considera-se o problema:

$$
\begin{aligned}
&\textbf{\textit{Minimizar:}} && \textstyle\int_0^{t_f} \frac{u(t)^2}{2} dt \\
&\textbf{\textit{sujeito a:}} && \dot{x}(t) = u(t)\cdot\cos(\psi(t)); \\
&&& \dot{y}(t) = u(t)\cdot\sin(\psi(t)); \\
&&& \dot{\psi}(t) = r(t); \\
&&& (x(0), y(0), \psi(0)) = (40, -2, \pi) \\
&&& (u(t), r(t)) \in [0,2]\times[-\pi,\pi]; \qquad q.t.\ t \in [0, t_f] \\
&&& (x(t_f), y(t_f), \psi(t_f)) \in B_r \times \mathbb{R} \qquad com\ B_r = \{(x,y) : x^2 + y^2 \leq r\}
\end{aligned}
$$

Transforma-se este problema, num problema do tipo de *Mayer*, referido no capítulo 3, e obtém-se:

$$
\begin{aligned}
&\textbf{\textit{Minimizar:}} && z(t_f) \\
&\textbf{\textit{sujeito a:}} && \dot{x}(t) = u(t)\cdot\cos(\psi(t)); \\
&&& \dot{y}(t) = u(t)\cdot\sin(\psi(t)); \\
&&& \dot{\psi}(t) = r(t); \\
&&& \dot{z}(t) = \frac{u(t)^2}{2}; \\
&&& (x(0), y(0), \psi(0), z(0)) = (40, -2, \pi, 0) \\
&&& (u(t), r(t)) \in [0,2]\times[-\pi,\pi]; \qquad q.t.\ t \in [0, t_f] \\
&&& (x(t_f), y(t_f), \psi(t_f)) \in B_r \times \mathbb{R} \qquad com\ B_r = \{(x,y) : x^2 + y^2 \leq r\}
\end{aligned}
$$

Aqui t_f , o tempo final, é definido à partida, comparando o caso com os problemas do tipo de Tempo Mínimo. Para este **caso 2** usou-se um tempo final de 20 segundos; como se pode verificar no anexo B.

5.1.3.2 Resultados Numéricos

Correndo o *AMPL*, com as condições apresentadas em 5.1, para este caso de minimização de custo de energia, obtiveram-se resultados analíticos apresentados a seguir.

Correndo o programa na linha de comandos em ambiente linux:

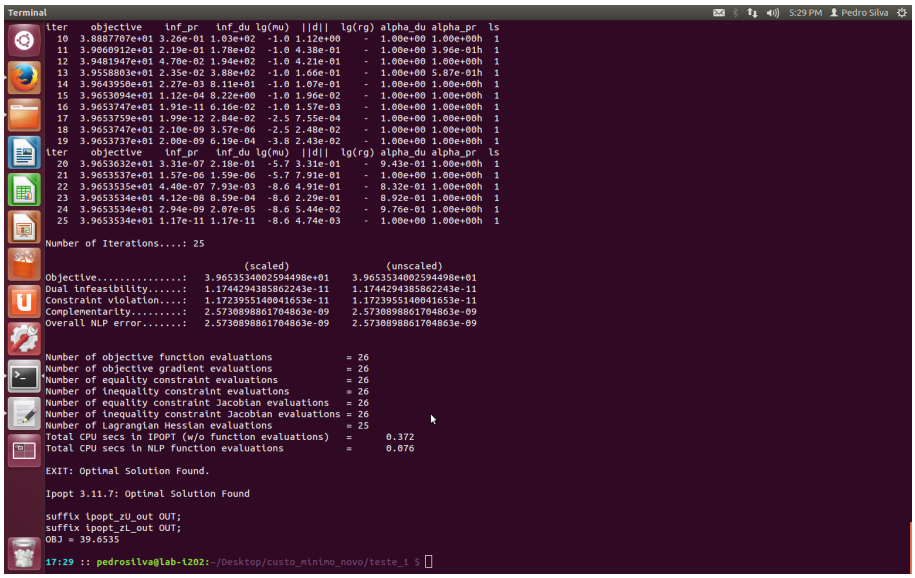

Figura 5.7: *AMPL* a correr na linha de comandos ambiente *Linux*

Pode-se observar na figura 5.7 que, além de se ter encontrado a solução ótima, também se apresenta o valor da função objetivo.

Então, os estados e controlos ótimos obtidos, apresentam-se de seguida:

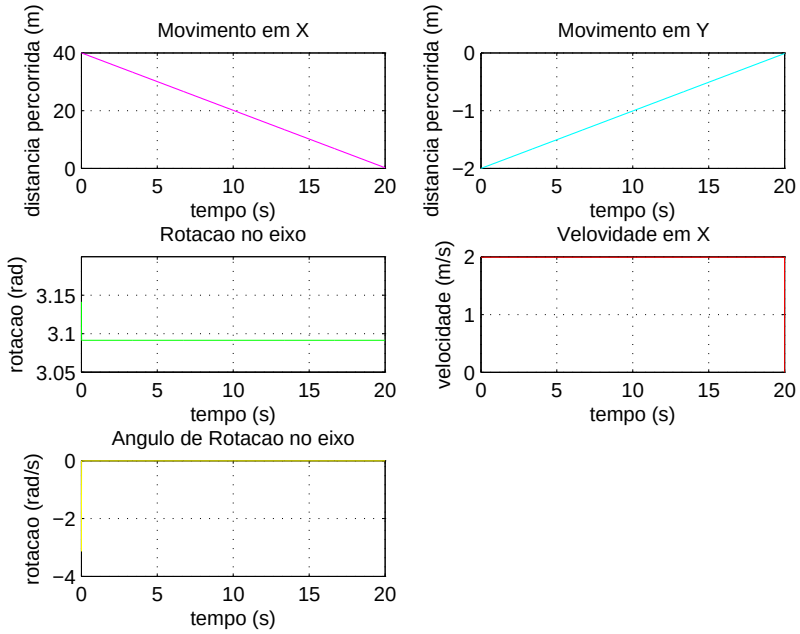

Figura 5.8: Estados e controlos ótimos para o caso 2

Observar-se, na figura 5.8 a distância percorrida e a forma como ela se altera ao longo do percurso nos gráficos do "Movimento do submarino no sentido de x e y"(figuras ilustradas a magenta

e azul ciano, respetivamente). Pode-se ainda verificar a rotação que tem em torno do seu eixo, para fazer um direcionamento para o ponto objetivo na figura ilustrada a verde.

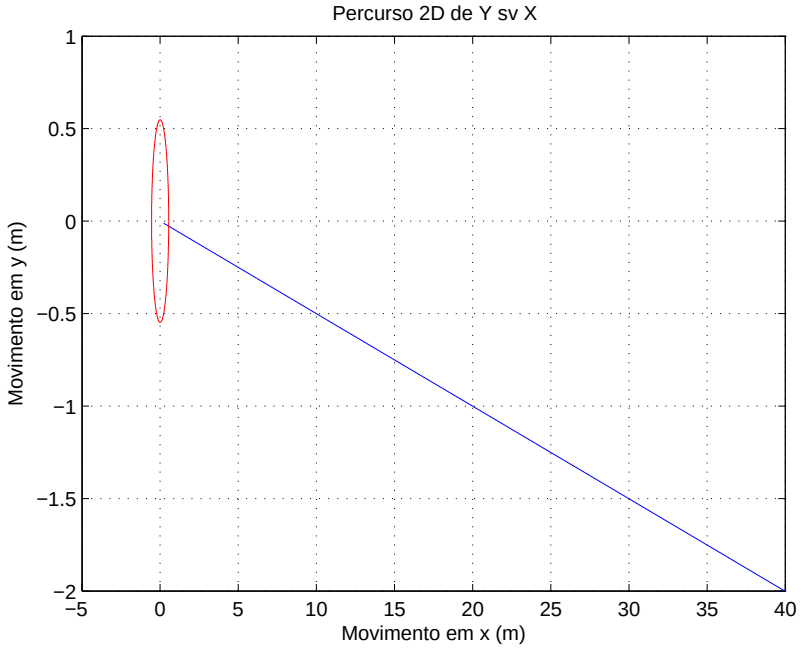

Figura 5.9: Percurso X vs Y

Note-se que as escalas de x e y são diferentes.

Finalmente, na figura 5.9, apresenta-se a movimentação do submarino a duas dimensões ($XvsY$). Pode-se observar a distância percorrida e a correção da sua rotação em relação ao eixo dos $zz's$. Tal como já referido no **Caso 1**, definiu-se uma circunferência como local de chegada do *AUV*.

Validação de Resultados Mais uma vez, usam-se as condições necessárias de otimalidade descritas em 3.3.4 para validar resultados. E assim extraem-se os valores numéricos dos multiplicadores p_x, p_y, p_ψ.

Os gráficos dos multiplicadores obtidos para este caso são apresentados na figura 5.10.

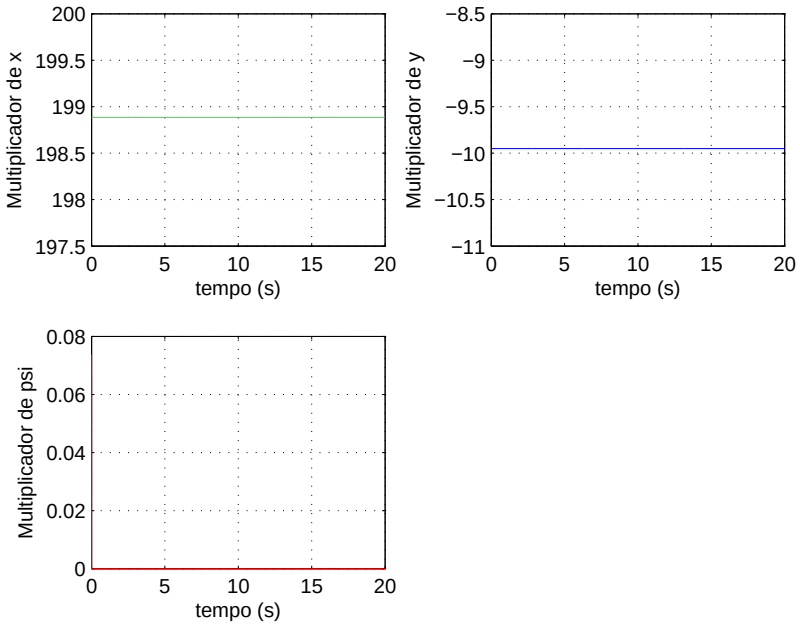

Figura 5.10: Multiplicadores das equações de estado

Agora o estado é $X = (x, y, \psi, z)$ e o controlo é (u, r).

O Hamiltoniano é

$$H =< (p_x, p_y, p_\psi, p_z) \cdot (u \cdot cos(\psi), u \cdot sin(\psi), r, \frac{u^2}{2}) >= \\ = p_x \cdot u \cdot cos(\psi) + p_y \cdot u \cdot sin(\psi) + p_\psi \cdot r + p_z * \frac{u^2}{2}$$

Por [**EA'**] tem-se $-\dot{p}_x = 0,\ -\dot{p}_y = 0,\ -\dot{p}_z = 0$ e $-\dot{p}_\psi = -p_x \cdot u^* \cdot sin(\psi^*) + p_y \cdot u^* \cdot cos(\psi^*)$

Logo $p_x(t), p_y(t)$ e $p_z(t)$ são constantes. Recorrendo à figura 5.10 constata-se que p_x e p_y são realmente constantes.

Considere-se agora [**CT'**]. Então, com $C_0 = \{(40, -2, \pi, 0)\}$ e $C_1 = \{(x,y) : x^2 + y^2 \leq 0.05\} = C \times \{(0,0)\}$, tem-se

$$(-p_x(t_f), -p_y(t_f)) = (\eta_1, \eta_2) \in N_C(x^*(t_f), y^*(t_f)) \\ p_\psi(t_f) \in \mathbb{R} \\ p_z(t_f) = -\lambda$$

Como $\dot{p}_z(t) = 0$, obtém-se $p_z(t) = -\lambda$.

Doravante ignora-se p_z e refere-se apenas a λ.

Além disso, $p_\psi(t_f) = 0$ o que realmente acontece numericamente (ver gráfico em baixo à esquerda na figura 5.10).

De [**MH'**] tem-se

$$p_x \cdot u^* \cdot cos(\psi^*) + p_y \cdot u^* \cdot sin(\psi^*) + p_\psi \cdot r^* - \lambda \cdot \frac{(u^*)^2}{2}$$
$$\geq p_x \cdot u \cdot cos(\psi^*) + p_y \cdot u \cdot sin(\psi^*) + p_\psi \cdot r - \lambda \cdot \frac{u^2}{2}$$

para todo o $u \in [0,2]$ e $r \in [-\pi, \pi]$.

Se $u^* \in]0,2[$, então u^* é tal que $\nabla_u \cdot H = 0$, ora

$$\nabla_u = p_x \cdot cos(\psi^*) + p_y \cdot sin(\psi^*) - \lambda \cdot u^* = 0$$

Supondo $\lambda = 1$, tem-se

$$u_{\text{analítico}} = p_x \cdot cos(\psi) + p_y \cdot sin(\psi)$$

Então, o u^* é dado por

$$u^*(t) = max\{0, min\{2, p_x \cdot cos(\psi) + p_y \cdot sin(\psi)\}\}$$

Na figura 5.11 comprova-se a validade dos resultados numéricos: o controlo ótimo numérico é interior ao intervalo $[0,2]$ e realmente o u^* coincide com $u_{\text{analítico}}$.

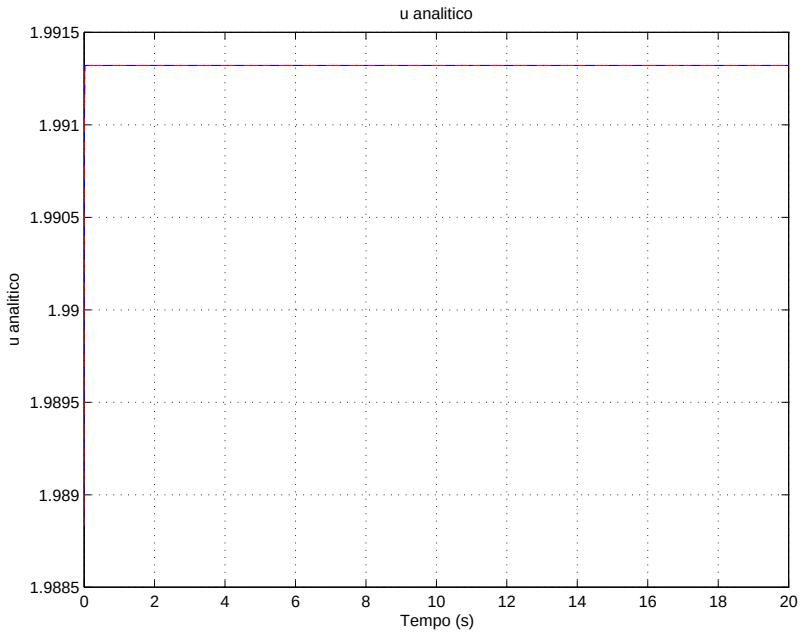

Figura 5.11: $u_{\text{analítico}}$ vs u numérico

Esta coincidência aponta para que λ seja realmente 1.

5.1.4 Outros Exemplos idênticos ao Caso 2

Nesta secção apresentam-se outros casos estudados de minimização do custo de energia, sendo que se mudaram os valores iniciais e finais do problema mostrado em 5.1.3.

Nesta parte não se justificará validar os resultados por extenso, da explicação das equações do Principio do Máximo, pois são em tudo iguais ás apresentadas no 5.1.3, daí não ser necessário haver uma repetição do que já foi explicado.

Obteve-se a seguinte figura com todos os percursos abordados:

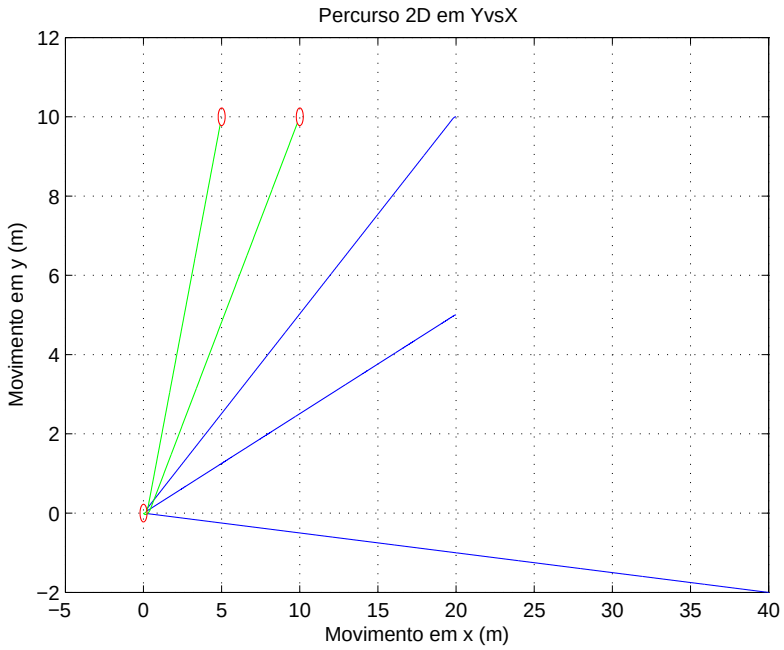

Figura 5.12: Todos os percursos do Caso 2 (Plano Y vs X)

Na figura 5.12, apresentam-se: a azul, todos os casos com ponto de chegada à vizinhança de $(0,0)$, e a verde os casos com ponto de partida de $(0,0)$. A azul ciano apresentam-se as movimentações das correntes marítimas, para o problema específico.

O seguinte quadro (5.2) refere os valores específicos de cada situação:

Tabela 5.2: Resultados dos Casos de Estudo de Tempo Mínimo Simples

Energia Mínima	**Ponto Partida**	**Ponto Chegada**	$x(t_f)$	$y(t_f)$	**Objetivo:** $x_f^2 + y_f^2 \leq raio^2$
39.6535	(40,-2)	(0,0)	0.223328	-0.011173	0.0500
16.347	(20,10)	(0,0)	0.199777	0.100444	0.0500
17.3288	(20,5)	(0,0)	0.216885	0.054414	0.500
10.7688	(0,0)	(10,10)	9.844664	9.839156	0.0500
6.75023	(0,0)	(5,10)	4.903524	9.798276	0.0500

5.2 Introdução aos Casos 3 e 4

Nesta secção apresentam-se novamente dois casos (**Caso 3** e **Caso 4**), nos quais se tratam duas novas situações de movimentação de um robot submarino, com aplicação de conceitos de controlo ótimo. Dificulta-se o problema acrescentando as correntes marítimas às equações de estado atrás descritas. Tornou-se assim objetivo encontrar soluções para estas novas equações ao nível de Tempo Mínimo (Caso 3) e ao nível do Custo Mínimo de Energia (Caso 4).

Decidiu-se assim, afetar o eixo referencial dos $xx's$, com a seguinte função:

$$correntes \;=\; 0.8 \cdot tanh(y); \tag{5.3}$$

Obtendo-se assim, as equações do estado do sistema:

$$\begin{aligned}
\dot{x}(t) &= u(t)\cdot\cos(\psi(t)) + 0.8\cdot tanh(y(t));\\
\dot{y}(t) &= u(t)\cdot\sin(\psi(t));\\
\dot{\psi}(t) &= r(t);
\end{aligned}$$

5.2.1 Caso 3 - Tempo Mínimo com Correntes

Os ficheiros para este caso; quer o ficheiro que corre no *AMPL*, quer o ficheiro que serve para desenhar os gráficos em *Matlab*, podem ser consultados no anexo C.

5.2.1.1 Formulação do Problema

$$\begin{aligned}
&\textbf{\textit{Minimizar:}} && t_f\\
&\textbf{\textit{sujeito a:}} && \dot{x}(t) = u(t)\cdot\cos(\psi(t)) + 0.8\cdot tanh(y(t));\\
&&& \dot{y}(t) = u(t)\cdot\sin(\psi(t));\\
&&& \dot{\psi}(t) = r(t);\\
&&& (x(0),y(0),\psi(0)) = (40,-2,\pi)\\
&&& (u(t),r(t)) \in [0,2]\times[-\pi,\pi]; \qquad q.t.\ t \in [0,t_f]\\
&&& (x(t_f),y(t_f),\psi(t_f)) \in B_r\times\mathbb{R} \qquad com\ B_r = \{(x,y) : x^2+y^2 \le r\}
\end{aligned}$$

5.2.1.2 Resultados Numéricos

Correndo o *AMPL*, com as condições apresentadas em 5.2, para este caso de minimização de tempo final com correntes, obtiveram-se os seguintes resultados analíticos:

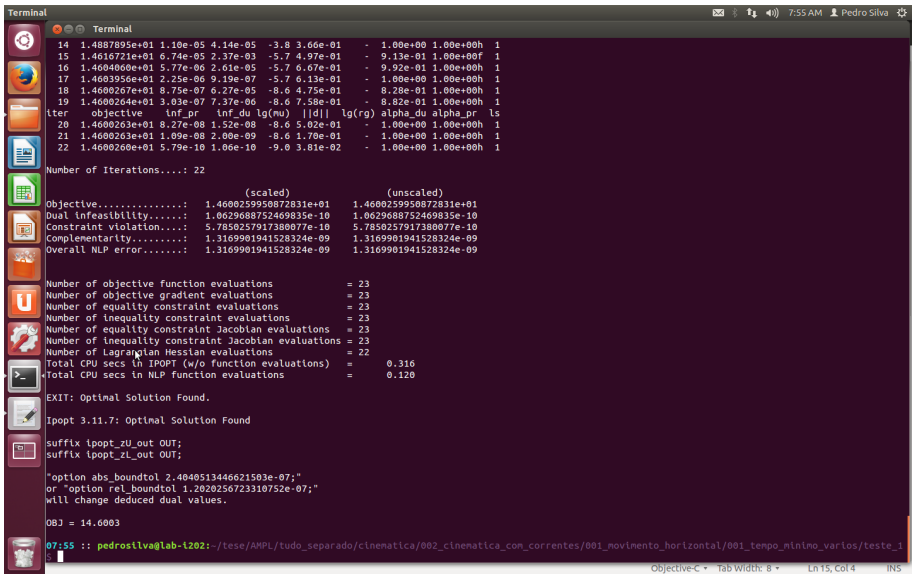

Figura 5.13: *AMPL* a correr na linha de comandos ambiente *Linux*

Pode-se observar na figura 5.13 que, além de se ter encontrado a solução ótima, também se apresenta o valor da função objetivo.

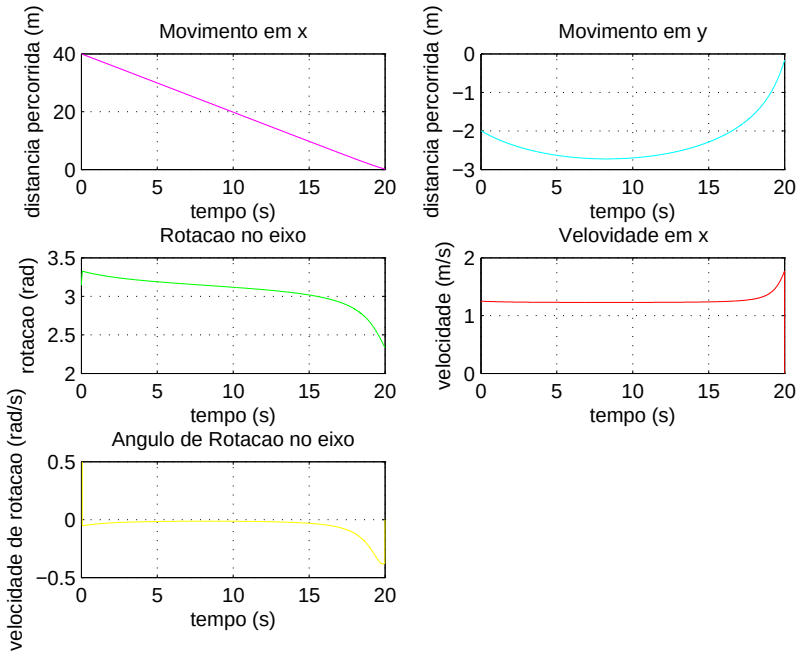

Figura 5.14: Estados e controlos ótimos para o caso 3

Observar-se, na figura 5.14 a distância percorrida e a forma como ela se altera ao longo do percurso nos gráficos do "Movimento do submarino no sentido de x e y"(figuras ilustradas a magenta e azul ciano, respetivamente). Pode-se ainda verificar a rotação que tem em torno do seu

eixo, para fazer um direcionamento para o ponto objetivo na figura ilustrada a verde.

Observam-se também na figura, os gráficos das velocidades linear (gráfico a vermelho) e angular (gráfico a amarelo) do robot.

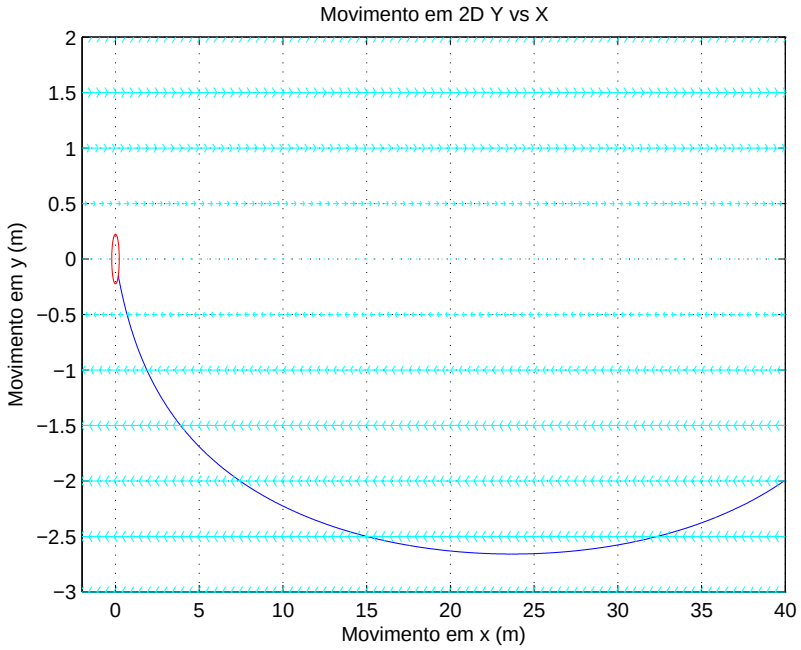

Figura 5.15: Percurso 2D Y vs X

Note-se que as escalas de x e y são diferentes.

Finalmente, na figura 5.15, apresenta-se a movimentação do submarino a duas dimensões ($XvsY$). Pode-se observar a distância percorrida e a correção da sua rotação em relação ao eixo dos $zz's$.

Definiu-se novamente uma circunferência como local de chegada do *AUV*.

Validação de Resultados Para validar os resultados, mais uma vez, usam-se as condições necessárias de otimalidade enunciadas em 3.3.4. Para o fazer é preciso extrair os valores numéricos dos multiplicadores p_x, p_y, p_ψ *e* q.

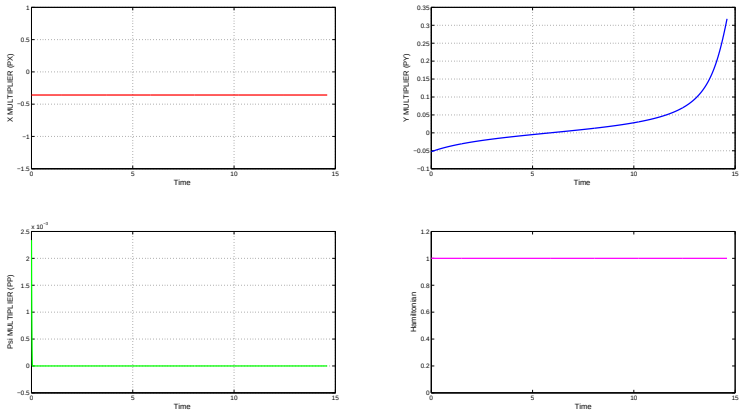

Figura 5.16: Multiplicadores das equações de estado

No caso 2, tem-se:

$$H = p \cdot f(t,x,u) =< (p_x, p_y, p_\psi) \cdot (u \cdot cos(\psi) + 0.8 \cdot tanh(y), u \cdot sen(\psi), r) >= \\ = p_x \cdot (u \cdot cos(\psi) + 0.8 \cdot tanh(y)) + p_y \cdot u \cdot sen(\psi) + p_\psi \cdot r$$

Verifica-se pelos multiplicadores apresentados em cima, que a primeira condição [**NT**] é válida.

De [**EA**] obtém-se:

$$-\dot{p}_x = \nabla_x H = 0 \quad \longrightarrow \quad p_x = constante$$
$$-\dot{p}_y = \nabla_y H = 0.8 \cdot \frac{\partial(tanh(y^*))}{\partial y}$$
$$-\dot{p}_\psi = \nabla_\psi H = -p_x \cdot u^* \cdot sen(\psi^*) + p_y \cdot u^* \cdot cos(\psi^*)$$
$$\dot{q} = \nabla_t H = 0$$

Note-se que $C_0 = \{(40,-2,\pi)\}$, $C_1 = \{(x,y) : x^2 + y^2 \leq 0.05\} \times \mathbb{R}$, $g(t_f, x(t_f)) = t_f$.

Assim, de [**CT**], obtém-se $(-q(0), p_x(0), p_y(0), p_\psi(0)) = (\xi_0, \xi_1, \xi_2, \xi_3)$ onde $\xi_i \in \mathbb{R}$, $i = 0,1,2,3$ e $(q(t_f), -p_x(t_f), -p_y(t_f), -p_\psi(t_f)) \in \lambda(1,0,0,0) + \{0\} \times N_{C_1}(x^*(t_f^*), y^*(t_f^*)) \times \{0\}$. Assim $q(t_f) = \lambda$.

Seja $\eta_1 \in N_C(x^*(t_f^*))$ tal que $p_x(t_f) = -\eta_1$.

Assim, tem-se $p_x(t_f) = -\eta_1 = \xi_1$.

Observe-se na figura 5.16 que o multiplicador p_x é realmente constante e aproximadamente igual a -0.4.

Já para p_ψ não se verifica o mesmo. Obtém-se $p_\psi(0) = \xi_3 \in \mathbb{R}$ e

$$\dot{p}_\psi(t) = -p_x(t) \cdot u^*(t) \cdot sen(\psi^*(t)) + p_y(t) \cdot u^*(t) \cdot cos(\psi^*(t)) = \\ = -\xi_1 \cdot u^*(t) \cdot sen(\psi^*(t)) + \xi_2 \cdot u^*(t) \cdot cos(\psi^*(t))$$

Por [**CH**] sabe-se que

$$H(t,p(t),x^*(t),u^*(t)) \;=\; q(t) \quad \forall\, t \;\in\; [0,t_f]$$

e assim tem-se que $q(t)$ é constante e igual a λ, o valor de λ e q fica determinado calculando $H(t,p(t),x^*(t),u^*(t))$. A função H é traçada na figura 5.16. Como se pode ver é constante e igual a 1. Verifica-se pois que $q(t) = 1 = \lambda$.

De [**MH**] Maximiza-se o Hamiltoniano:

$$H(t,x^*,p,u^*,r^*) \geq \{H(t,x^*,p,u,r)\}, \quad u \;\in\; [0,2],\; r \;\in\; [-\pi,\pi];$$

Assim (ocultando a dependência de t):

$$\begin{aligned} p_x \cdot (u^* \cdot cos(\psi^*)) + p_y \cdot (u^* \cdot sen(\psi^*)) + p_\psi \cdot r^* \geq \\ \geq p_x \cdot (u \cdot cos(\psi^*)) + p_y \cdot (\cdot sen(\psi^*)) + p_\psi \cdot r \quad u \;\in\; [0,2],\; r \;\in\; [-\pi,\pi], \end{aligned}$$

O que equivale a

$$p_x \cdot cos(\psi^*) \cdot (u^* - u) + p_y \cdot sen(\psi^*) \cdot (u^* - u) + p_\psi \cdot (r^* - r) \geq 0 \quad u \;\in\; [0,2],\; r \;\in\; [-\pi,\pi] \tag{5.4}$$

Se se fixar $u \;=\; u^*$ em cima, tem-se:

$$p_\psi \cdot (r^* - r) \geq 0, \quad \forall\, r \;\in\; [-\pi,\pi] \tag{5.5}$$

Ou seja

$$r^*(t) = \begin{cases} singular & se \quad p_\psi = 0 \\ \pi & se \quad p_\psi > 0 \\ -\pi & se \quad p_\psi < 0 \end{cases}$$

Nota: Singular significa que o valor da função nesse ponto, fica entre os extremos.

Observe-se o gráfico de r na figura 5.14, e o gráfico de p_ψ da figura 5.4, e conclui-se que este passo é corroborado pelos mesmos.

Fixando $r \;=\; r^*$ em (5.4) tem-se

$$(p_x \cdot cos(\psi^*) + p_y \cdot sen(\psi^*)) \cdot (u^* - u) \geq 0$$

Seja $func = (p_x \cdot cos(\psi^*) + p_y \cdot sen(\psi^*))$. Tem-se assim

$$u^*(t) = \begin{cases} singular & se \quad Func = 0 \\ \text{máximo} & se \quad Func > 0 \\ \text{mínimo} & se \quad Func < 0 \end{cases}$$

O gráfico seguinte da figura 5.17, apresenta a função $func$:

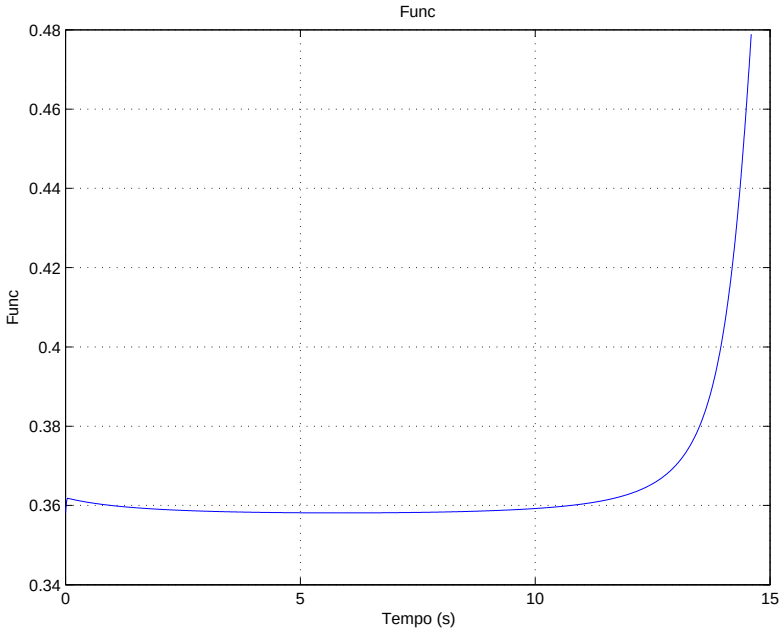

Figura 5.17: Função $func$

Como se visualiza na figura 5.17, $func > 0$. Então o controlo u deverá estar sempre no valor máximo; e é isso que se verifica na figura 5.14.

Mais informação:

$$t_f = 14.600259$$
$$x(t_f) = 0.167452$$
$$y(t_f) = -0.148189$$
$$\psi(t_f) = 2.417189$$
$$\eta_1 = 49.057088$$
$$\eta_2 = -43.413757$$

Verifica-se ainda que

$$x(t_f)^2 + y(t_f)^2 \leq 0.05$$

Valida-se este caso, pois tem-se $x(t_f)^2 + y(t_f)^2 = 0.05$. Logo o ponto atinge a fronteira da circunferência que se determinou.

No ponto $(x(t_f), y(t_f))$ a normal da circunferência $x^2 + y^2 \leq raio^2$, (η_1, η_2) tem primeira componente grande e a segunda componente mais pequena. Daí se poder observar que, essa mesma normal será um vetor a apontar para fora da circunferência, com ângulo $tan^{-1}(\frac{y(t_f)}{x(t_f)}) = -0.7144$. Conclui-se que o vetor normal à tangente se situa no quarto quadrante.

5.2.2 Outros Exemplos idênticos ao Caso 3

Nesta secção apresentam-se outros casos estudados de minimização do tempo, agora com a movimentação do veículo influenciado por correntes marinhas, sendo que se mudaram os valores iniciais e finais do problema mostrado em 5.2.1.

Nesta parte não se justificará validar os resultados por extenso, da explicação das equações do Principio do Máximo, pois são em tudo iguais ás apresentadas no 5.2.1, daí não ser necessário haver uma repetição do que já foi explicado.

Obteve-se a seguinte figura com todos os percursos abordados:

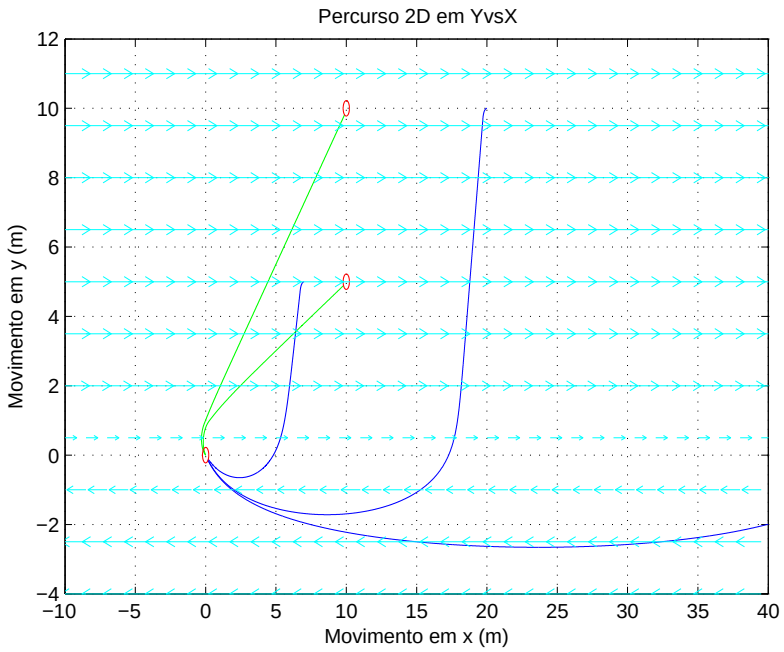

Figura 5.18: Todos os percursos do Caso 3 (Plano Y vs X)

Na figura 5.18, apresentam-se: a azul, todos os casos com ponto de chegada à vizinhança de $(0,0)$, e a verde os casos com ponto de partida $(0,0)$. A azul ciano apresentam-se as movimentações das correntes marítimas, para o problema específico.

O seguinte quadro (5.3) refere os valores específicos de cada situação:

Tabela 5.3: Resultados dos Casos de Estudo de Tempo Mínimo com Correntes

t_f	**Ponto Partida**	**Ponto Chegada**	$x(t_f)$	$y(t_f)$	**Objetivo:** $x_f^2+y_f^2 \leq raio^2$
14.6003	(40,-2)	(0,0)	0.167452	-0.148189	0.05
13.0207	(20,10)	(0,0)	0.169957	-0.145308	0.0452
5.49164	(7,5)	(0,0)	0.189364	-0.118917	0.05
4.63126	(0,0)	(10,5)	9.805737	4.889266	0.0272
5.99283	(0,0)	(10,10)	9.877584	9.812872	0.05

5.2.3 Caso 4 - Minimização de Energia com Correntes

Os ficheiros para este caso; quer o ficheiro que corre no *AMPL*, quer o ficheiro que serve para desenhar os gráficos em *Matlab*, podem ser consultados no anexo D.

5.2.3.1 Formulação do Problema

Idêntico ao **Caso 2** estudado em 5.1.3, converteu-se o problema original num do tipo de *Mayer*:

$$
\begin{aligned}
&\textbf{\textit{Minimizar:}} && z(t_f)\\
&\textbf{\textit{sujeito a:}} && \dot{x}(t) = u(t)\cdot\cos(\psi(t)) + 0.8\cdot tanh(y(t));\\
&&& \dot{y}(t) = u(t)\cdot\sin(\psi(t));\\
&&& \dot{\psi}(t) = r(t);\\
&&& \dot{z}(t) = \frac{u(t)^2}{2};\\
&&& (x(0),y(0),\psi(0),z(0)) = (40,-2,\pi,0)\\
&&& (u(t),r(t)) \in [0,2]\times[-\pi,\pi]; && q.t.\ t \in [0,t_f]\\
&&& (x(t_f),y(t_f),\psi(t_f)) \in B_r\times\mathbb{R} && com\ B_r = \{(x,y) : x^2+y^2 \leq r\}
\end{aligned}
$$

Refere-se como no **Caso 2**, que t_f , o tempo final, é definido à partida, comparando o caso com os problemas do tipo de Tempo Mínimo. Para este **caso 4** usou-se um tempo final de 20 segundos; como se pode verificar no anexo D.

5.2.3.2 Resultados Numéricos

Correndo o *AMPL*, com as condições apresentadas em 5.2, para este caso de minimização de custo de energia com correntes, obtiveram-se os seguintes resultados analíticos:

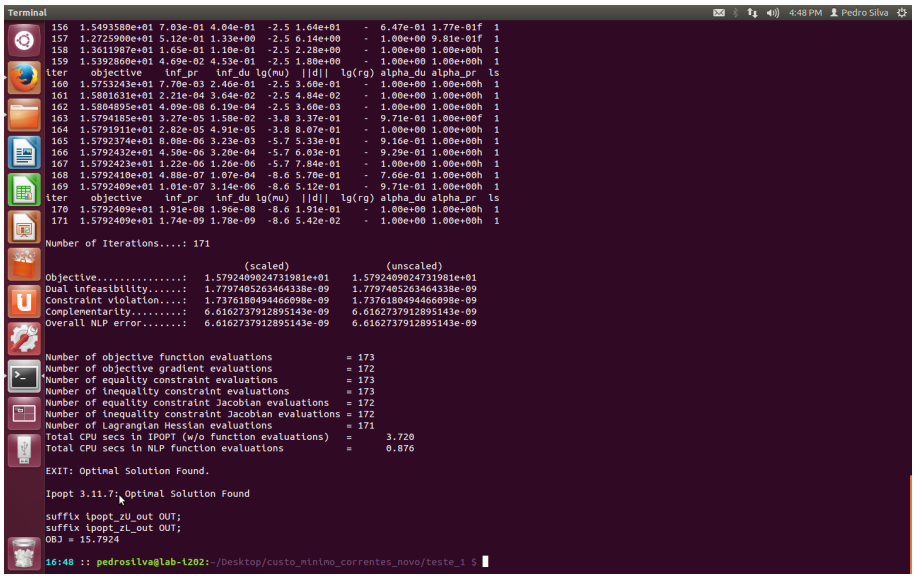

Figura 5.19: *AMPL* a correr na linha de comandos ambiente *Linux*

Pode-se observar na figura 5.19 que, além de se ter encontrado a solução ótima, também se apresenta o valor da função objetivo.

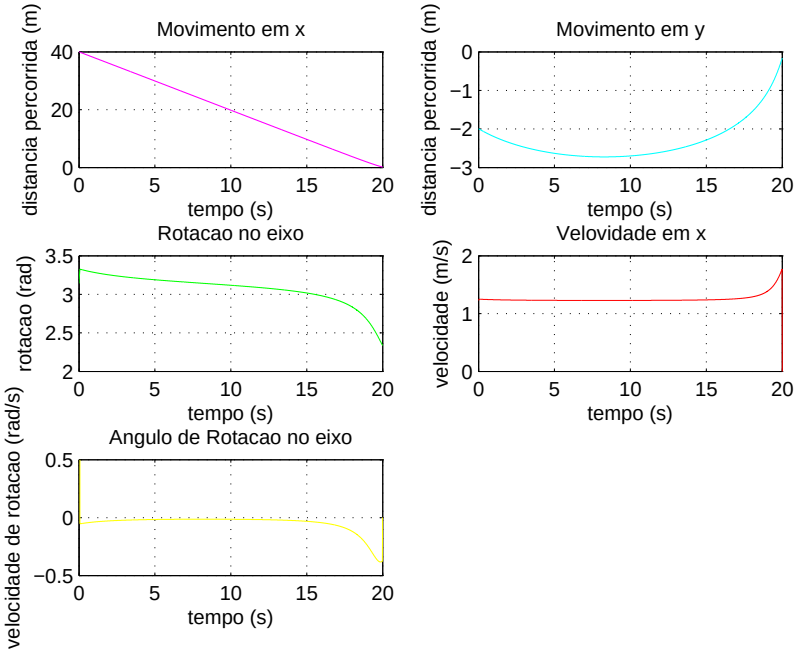

Figura 5.20: Estados e controlos ótimos do caso 4

Observar-se, na figura 5.20 a distância percorrida e a forma como ela se altera ao longo do percurso nos gráficos do "Movimento do submarino no sentido de x e y"(figuras ilustradas a magenta e azul ciano, respetivamente). Pode-se ainda verificar a rotação que tem em torno do seu

eixo, para fazer um direcionamento para o ponto objetivo na figura ilustrada a verde.

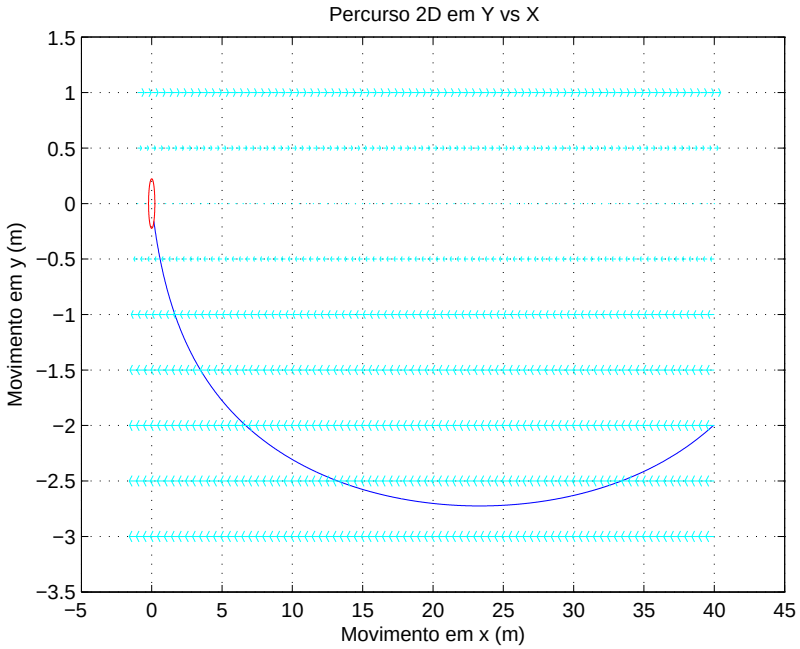

Figura 5.21: Gráfico y (m) *Vs* x(m)

Note-se que as escalas de x e y são diferentes.

Finalmente, na figura 5.21, apresenta-se a movimentação do submarino a duas dimensões ($XvsY$). Pode-se observar a distância percorrida e a correção da sua rotação em relação ao eixo dos $zz's$. Tal como já referido no **Caso 1**, definiu-se uma circunferência como local de chegada do *AUV*.

Validação de Resultados Mais uma vez, usam-se as condições necessárias de otimalidade descritas em 3.3.4 para validar resultados. E assim extraem-se os valores numéricos dos multiplicadores p_x, p_y, p_ψ.

Os gráficos dos multiplicadores obtidos para este caso são apresentados na figura 5.22.

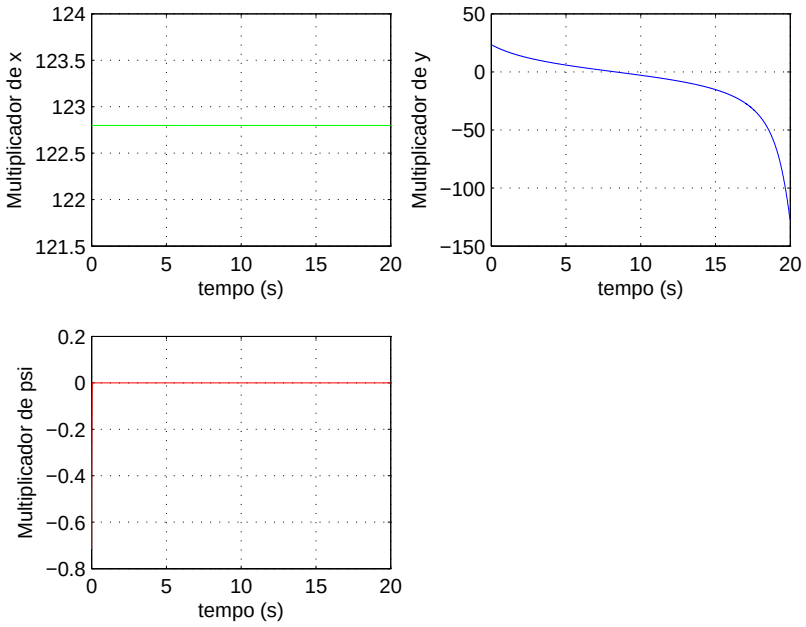

Figura 5.22: Multiplicadores das equações de estado

Agora o estado é $X = (x, y, \psi, z)$ e o controlo é (u, r).

O Hamiltoniano é

$$H =< (p_x, p_y, p_\psi, p_z) \cdot (u \cdot cos(\psi) + 0.8 \cdot tanh(y), u \cdot sin(\psi), r, \tfrac{u^2}{2}) >= \\ = p_x \cdot (u \cdot cos(\psi) + 0.8 \cdot tanh(y)) + p_y \cdot u \cdot sin(\psi) + p_\psi \cdot r + p_z \cdot \tfrac{u^2}{2}$$

Por [**EA'**] tem-se $-\dot{p}_x = 0$, $-\dot{p}_y = p_x \cdot 0.8 \cdot tanh(y)$, $-\dot{p}_z = 0$ e $-\dot{p}_\psi = -p_x \cdot u^* \cdot sin(\psi^*) + p_y \cdot u^* \cdot cos(\psi^*)$

Logo $p_x(t)$ e $p_z(t)$ são constantes. Recorrendo à figura 5.22 constata-se que p_x é realmente constante.

Considere-se agora [**CT'**]. Então, com $C_0 = \{(40, -2, \pi, 0)\}$ e $C_1 = \{(x, y) : x^2 + y^2 \leq 0.05\} = C \times \{(0,0)\}$, tem-se

$$(-p_x(t_f), -p_y(t_f)) = (\eta_1, \eta_2) \in N_C(x^*(t_f), y^*(t_f)) \\ p_\psi(t_f) \in \mathbb{R} \\ p_z(t_f) = -\lambda$$

Como $\dot{p}_z(t) = 0$, obtém-se $p_z(t) = -\lambda$.

Doravante ignora-se p_z e refere-se apenas a λ.

Além disso, $p_\psi(t_f) = 0$ o que realmente acontece numericamente (ver gráfico em baixo à esquerda na figura 5.22).

De [**MH'**] tem-se

$$p_x \cdot (u^* \cdot cos(\psi^*) + 0.8 \cdot tanh(y^*)) + p_y \cdot u^* \cdot sin(\psi^*) + p_\psi \cdot r^* - \lambda \cdot \frac{(u^*)^2}{2}$$
$$\geq p_x \cdot (u \cdot cos(\psi^*) + 0.8 \cdot tanh(y^*)) + p_y \cdot u \cdot sin(\psi^*) + p_\psi \cdot r - \lambda \cdot \frac{u^2}{2}$$

para todo o $u \in [0,2]$ e $r \in [-\pi, \pi]$.

Se $u^* \in]0,2[$, então u^* é tal que $\nabla_u \cdot H = 0$, ora

$$\nabla_u = p_x \cdot cos(\psi^*) + p_y \cdot sin(\psi^*) - \lambda \cdot u^* = 0$$

Supondo $\lambda = 1$, tem-se

$$u_{\text{analítico}} = p_x \cdot cos(\psi) + p_y \cdot sin(\psi)$$

Então, o u^* é dado por

$$u^*(t) = max\{0, min\{2, p_x \cdot cos(\psi) + p_y \cdot sin(\psi)\}\}$$

Na figura 5.23 comprova-se a validade dos resultados numéricos: o controlo ótimo numérico é interior ao intervalo $[0,2]$ e realmente o u^* coincide com $u_{\text{analítico}}$.

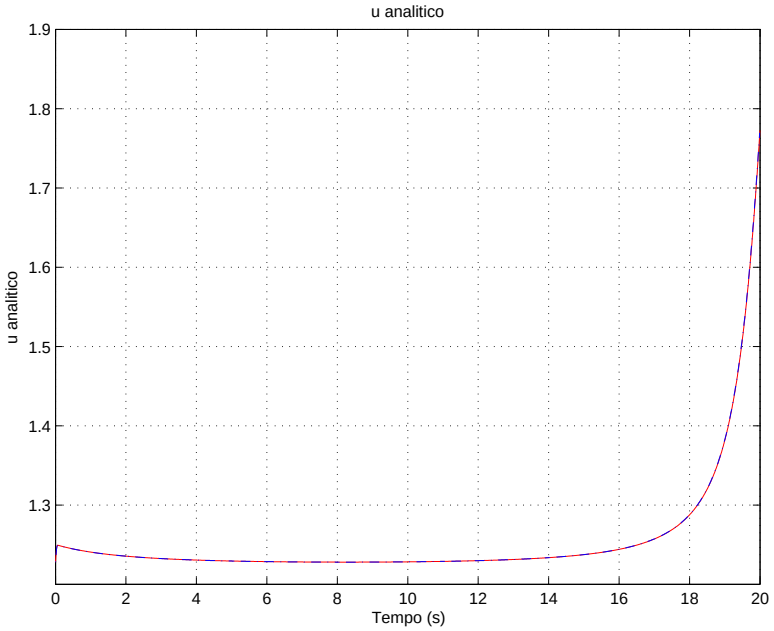

Figura 5.23: $u_{analtico}$ vs $u$numérico

Esta coincidência aponta para que λ seja realmente 1.

5.2.4 Outros Exemplos idênticos ao Caso 4

Nesta secção apresentam-se outros casos estudados de minimização do tempo, agora com a movimentação do veículo influenciado por correntes marinhas, sendo que se mudaram os valores iniciais e finais do problema mostrado em 5.2.3.

Nesta parte não se justificará validar os resultados por extenso, da explicação das equações do Principio do Máximo, pois são em tudo iguais ás apresentadas no 5.2.3, daí não ser necessário haver uma repetição do que já foi explicado.

Obteve-se a seguinte figura com todos os percursos abordados:

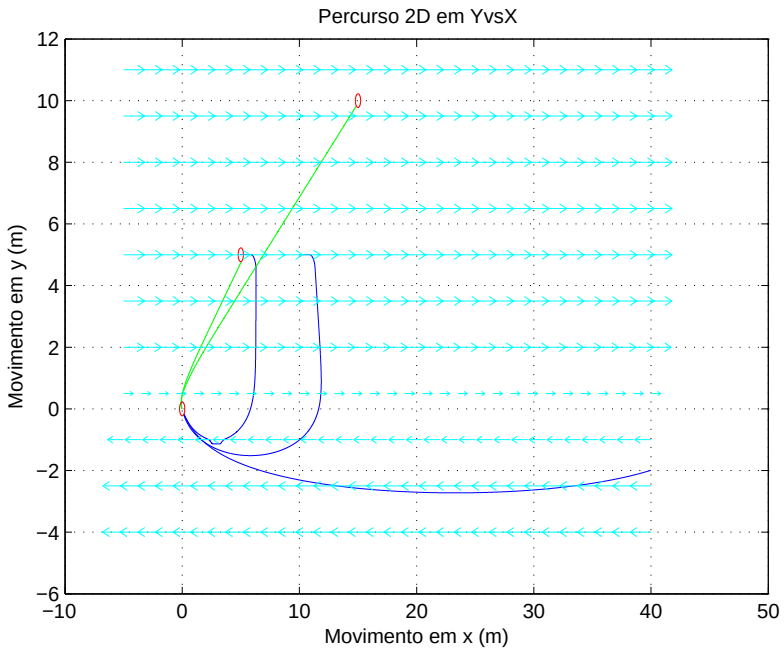

Figura 5.24: Todos os percursos do Caso 4 (Plano Y vs X)

Na figura 5.24, apresentam-se: a azul, todos os casos com ponto de chegada à vizinhança de $(0,0)$, e a verde os casos com ponto de partida de $(0,0)$. A azul ciano apresentam-se as movimentações das correntes marítimas, para o problema específico.

O seguinte quadro (5.4) refere os valores específicos de cada situação:

Tabela 5.4: Resultados dos Casos de Estudo de Consumo de Energia Mínima com Correntes

Energia Mínima	**Ponto Partida**	**Ponto Chegada**	$x(t_f)$	$y(t_f)$	**Objetivo:** $x_f^2+y_f^2 \leq raio^2$
15.7924	(40,-2)	(0,0)	0.155201	-0.160974	0.0500
6.49338	(10,5)	(0,0)	0.128860	-0.182743	0.0500
3.7603	(5,5)	(0,0)	0.149868	-0.165951	0.0300
1.27806	(0,0)	(5,5)	5.054256	4.783075	0.0500
8.09743	(0,0)	(15,10)	14.857454	9.827719	0.0500

5.3 Introdução ao Caso 5

Nesta secção torna-se o problema ainda mais complexo, entra-se aqui com a dinâica do *AUV*. Resolve-se o problema com a situação de controlo ótimo aplicado para obtenção do Tempo Mínimo. Assim acresce-se o que anteriormente foi feito relativamente ao **Caso 4**, e aplica-se uma força f ao propulsor afetado pelo controlo u, passando este a ser um estado e ficando a força referido como um novo controlo. Variou-se o f de $[0,4]$.

5.3.1 Caso 5 - Tempo Mínimo com Dinâmica e Correntes

Os ficheiros para este caso; quer o ficheiro que corre no *AMPL*, quer o ficheiro que serve para desenhar os gráficos em *Matlab*, podem ser consultados no anexo E.

5.3.1.1 Formulação do Problema

$$
\begin{aligned}
&\textbf{\textit{Minimizar:}} && t_f \\
&\textbf{\textit{sujeito a:}} && \dot{x}(t) = u(t)\cdot\cos(\psi(t)) + 0.8\cdot tanh(y(t)) \\
&&& \dot{y}(t) = u(t)\cdot\sin(\psi(t)) \\
&&& \dot{\psi}(t) = r(t) \\
&&& \dot{u}(t) = f(t) - u(t)\cdot|u(t)| \\
&&& (x(0),y(0),\psi(0),u(0)) = (40,-20,\pi,0) \\
&&& (f(t),r(t)) \in [0,4]\times[-\pi,\pi] && q.t.\ \ t \in [0,t_f] \\
&&& (x(t_f),y(t_f)) \in B_r, \psi(t_f) = \pi, u(t_f) = 0 && com\ B_r = \{(x,y) : x^2+y^2 \leq 0.05\}
\end{aligned}
$$

Pode-se observar pelo modelo, que ainda se dificultou mais o exercício, obrigando o *AUV* a chegar ao ponto final rodado π sobre o seu eixo, e com uma velocidade nula.

5.3.1.2 Resultados Numéricos

Correndo o *AMPL*, com as condições apresentadas em 5.3, para este caso de minimização de tempo final com correntes, obtiveram-se os seguintes resultados analíticos:

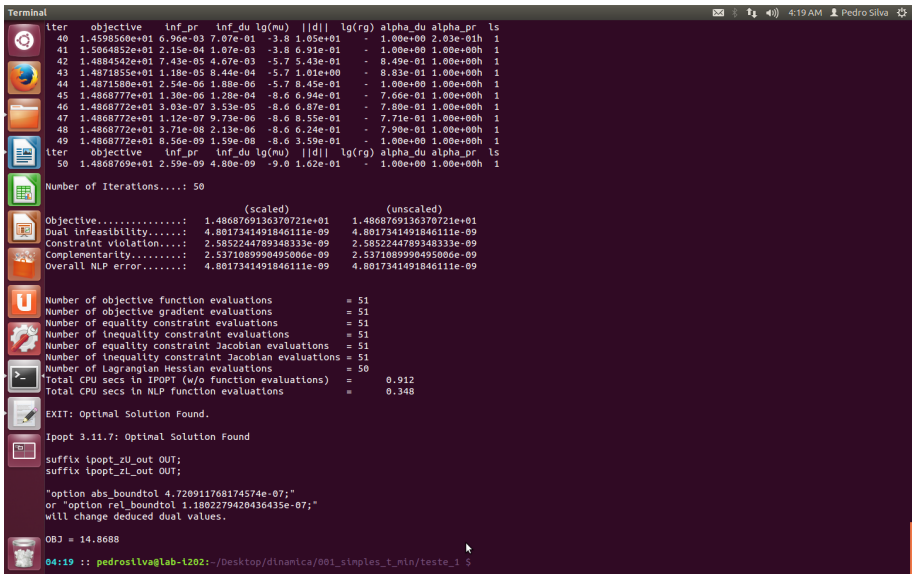

Figura 5.25: *AMPL* a correr na linha de comandos ambiente *Linux*

Pode-se observar na figura 5.25 que, além de se ter encontrado a solução ótima, também se apresenta o valor da função objetivo que é de 14.8688 *segundos*.

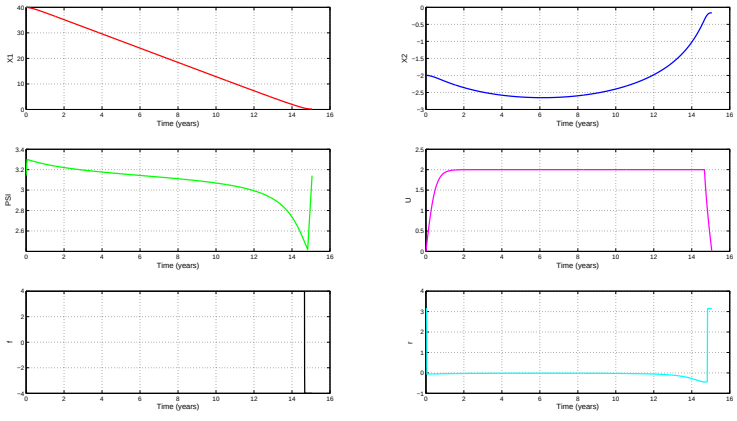

Figura 5.26: Estados e controlos ótimos para o caso 3

Observar-se, na figura 5.26 a distância percorrida e a forma como ela se altera ao longo do percurso nos gráficos do "Movimento do submarino no sentido de x e y"(figuras ilustradas a magenta e azul ciano, respetivamente). Pode-se ainda verificar a rotação que tem em torno do seu eixo, para fazer um direcionamento para o ponto objetivo na figura ilustrada a verde.

Observam-se também na figura, os gráficos das velocidades linear (gráfico a vermelho) e angular (gráfico a amarelo) do robot.

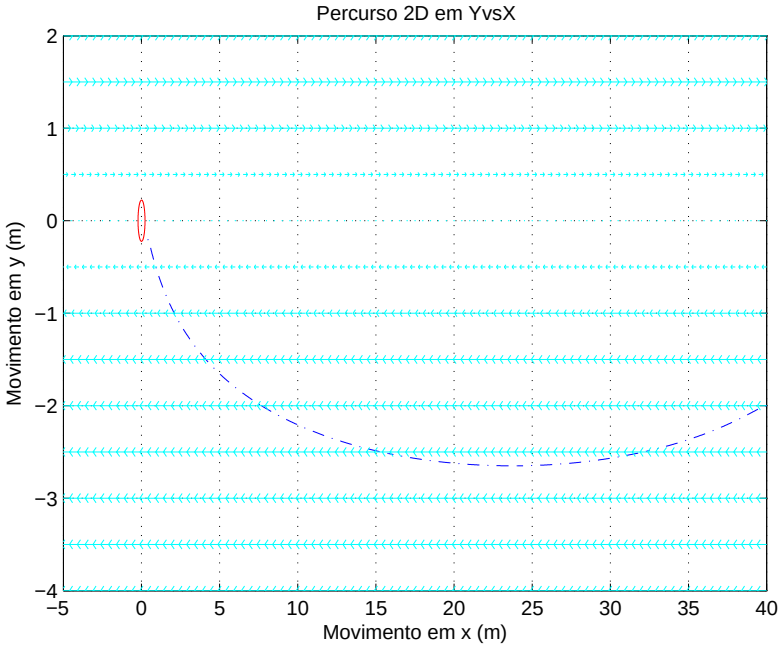

Figura 5.27: Percurso 2D X vs Y

Note-se que as escalas de x e y são diferentes.

Finalmente, na figura 5.27, apresenta-se a movimentação do submarino a duas dimensões ($XvsY$). Pode-se observar a distância percorrida e a correção da sua rotação em relação ao eixo dos $zz's$. Observa-se também o comportamento do novo controlo f.

Definiu-se novamente uma circunferência como local de chegada do *AUV*.

Validação de Resultados Para validar os resultados, mais uma vez, usam-se as condições necessárias de otimalidade enunciadas em 3.3.4. Para o fazer é preciso extrair os valores numéricos dos multiplicadores p_x, p_y, p_ψ *e* q.

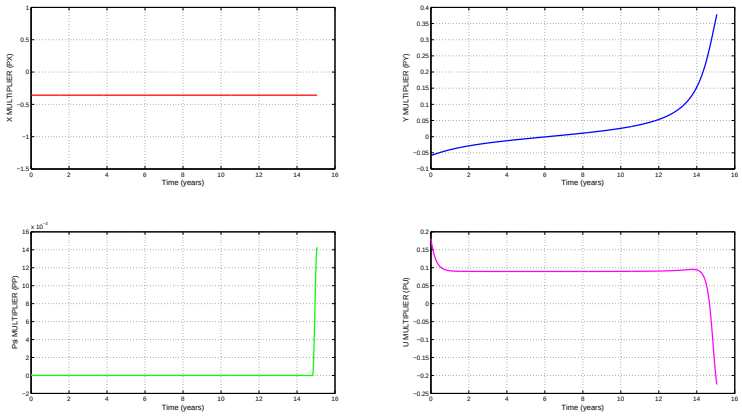

Figura 5.28: Multiplicadores das equações de estado

No caso 2, tem-se:

$$H = p \cdot f(t,x,u) =< (p_x, p_y, p_\psi, p_u) \cdot (u \cdot cos(\psi) + 0.8 \cdot tanh(y), u \cdot sen(\psi), r, f - u \cdot |u|) >= \\ = p_x \cdot (u \cdot cos(\psi) + 0.8 \cdot tanh(y)) + p_y \cdot u \cdot sen(\psi) + p_\psi \cdot r + p_u \cdot (f - u \cdot |u|)$$

Verifica-se pelos multiplicadores apresentados em cima, que a primeira condição [**NT**] é válida.

De [**EA**] obtém-se:

$$\begin{aligned} -\dot{p}_x = \nabla_x H = 0 \quad &\longrightarrow \quad p_x = constante \\ -\dot{p}_y = \nabla_y H = 0.8 \cdot \frac{\partial (tanh(y^*))}{\partial y} \\ -\dot{p}_\psi = \nabla_\psi H = -p_x \cdot u^* \cdot sen(\psi^*) + p_y \cdot u^* \cdot cos(\psi^*) \\ -\dot{u} = \nabla_u H = p_x \cdot cos(\psi) + p_y \cdot sin(\psi) - 2 \cdot u\dot{q} = \nabla_t H = 0 \end{aligned}$$

Note-se que $C_0 = \{(40,-2,\pi),0\}$, $C_1 = \{(x,y) : x^2 + y^2 \leq 0.05\} \times \mathbb{R} \times \mathbb{R}$, $g(t_f, x(t_f)) = t_f$.

Assim, de [**CT**], obtém-se $(-q(0), p_x(0), p_y(0), p_\psi(0), p_u(o) = (\xi_0, \xi_1, \xi_2, \xi_3, \xi_4)$ onde $\xi_i \in \mathbb{R}$, $i = 0,1,2,3,4$ e $(q(t_f), -p_x(t_f), -p_y(t_f), -p_\psi(t_f), -p_u(t_f) \in \lambda(1,0,0,0,0) + \{0\} \times N_{C_1}(x^*(t_f^*), y^*(t_f^*)) \times \{0\} \times \{0\}$. Assim $q(t_f) = \lambda$.

Seja $\eta_1 \in N_C(x^*(t_f^*))$ tal que $p_x(t_f) = -\eta_1$.

Assim, tem-se $p_x(t_f) = -\eta_1 = \xi_1$.

Observe-se na figura 5.16 que o multiplicador p_x é realmente constante e aproximadamente igual a -0.4.

Já para p_ψ não se verifica o mesmo. Obtém-se $p_\psi(0) = \xi_3 \in \mathbb{R}$ e

$$\dot{p}_\psi(t) = -p_x(t) \cdot u^*(t) \cdot sen(\psi^*(t)) + p_y(t) \cdot u^*(t) \cdot cos(\psi^*(t)) = \\ = -\xi_1 \cdot u^*(t) \cdot sen(\psi^*(t)) + p_y(t) \cdot u^*(t) \cdot cos(\psi^*(t))$$

Por [**CH**] sabe-se que

$$H(t,p(t),x^*(t),u^*(t)) \;=\; q(t) \quad \forall\, t \;\in\; [0,t_f]$$

e assim tem-se que $q(t)$ é constante e igual a λ, o valor de λ e q fica determinado calculando $H(t,p(t),x^*(t),u^*(t))$. A função H é traçada na figura 5.29. Como se pode ver é constante e igual a 1. Verifica-se pois que $q(t) = 1 = \lambda$.

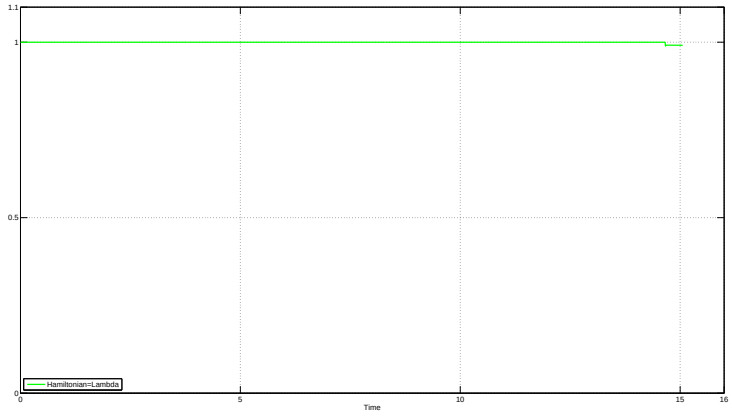

Figura 5.29: Hamiltoniano

De [**MH**] Maximiza-se o Hamiltoniano:

$$H(t,x^*,p,u^*,r^*) \geq \{H(t,x^*,p,f,r)\}, \quad u \;\in\; [0,4],\; r \;\in\; [-\pi,\pi];$$

Assim (ocultando a dependência de t):

$$p_x \cdot (u^* \cdot cos(\psi^*)) + p_y \cdot (u^* \cdot sen(\psi^*)) + p_\psi \cdot r^* + p_u(f^* - u \cdot |u|) \geq$$
$$\geq p_x \cdot (u \cdot cos(\psi^*)) + p_y \cdot (\cdot sen(\psi^*)) + p_\psi \cdot r + p_u(f - u \cdot |u|) \quad f \;\in\; [0,4],\; r \;\in\; [-\pi,\pi],$$

O que equivale a

$$p_\psi \cdot (r^* - r) + p_u(f^* - f) \geq 0 \quad f \;\in\; [0,4],\; r \;\in\; [-\pi,\pi] \tag{5.6}$$

Se se fixar $f \;=\; f^*$ em cima, tem-se:

$$p_\psi \cdot (r^* - r) \geq 0, \quad \forall\, r \;\in\; [-\pi,\pi] \tag{5.7}$$

Ou seja

$$r^*(t) = \begin{cases} singular & se \quad p_\psi = 0 \\ \pi & se \quad p_\psi > 0 \\ -\pi & se \quad p_\psi < 0 \end{cases}$$

Nota: Singular significa que o valor da função nesse ponto, fica entre os extremos.

Observe-se o gráfico de r na figura 5.26, e o gráfico de p_ψ da figura 5.28, e conclui-se que este passo é corroborado pelos mesmos.

Fixando $r = r^*$ em (5.6) tem-se

$$p_u(f^* - f) \geq 0$$

Ou seja

$$f^*(t) = \begin{cases} singular & se \quad p_u = 0 \\ 4 & se \quad p_u > 0 \\ 0 & se \quad p_u < 0 \end{cases}$$

Observe-se o gráfico de f na figura 5.26, e o gráfico de p_u da figura 5.28, e conclui-se que este passo é corroborado pelos mesmos.

Mais informação:

$$\begin{aligned} t_f &= 14.8688 \\ x(t_f) &= 0.161417 \\ y(t_f) &= -0.154740 \\ \psi(t_f) &= 3.141593 \\ u(t_f) &= 0 \\ \eta_1 &= 36.129816 \\ \eta_2 &= -34.635266 \end{aligned}$$

Verifica-se ainda que

$$x(t_f)^2 + y(t_f)^2 \leq 0.05$$

Valida-se este caso, pois tem-se $x(t_f)^2 + y(t_f)^2 = 0.05$. Logo o ponto atinge a fronteira da circunferência que se determinou.

No ponto $(x(t_f), y(t_f))$ a normal da circunferência $x^2 + y^2 \leq raio^2$, (η_1, η_2) tem primeira componente grande e a segunda componente mais pequena. Daí se poder observar que, essa mesma normal será um vetor a apontar para fora da circunferência, com ângulo $tan^{-1}(\frac{y(t_f)}{x(t_f)}) = -0.7643$. Conclui-se que o vetor normal à tangente se situa no quarto quadrante.

5.3.2 Outros Exemplos idênticos ao Caso 5

Aqui apresenta-se o caso de movimento com a dinâmica do *AUV*, no qual se aplica controlo ótimo de forma a se obter o tempo mínimo do percurso. Halteraram-se os valores inicias e finais do apresentado em 5.3.

Nesta parte não se justificará validar os resultados por extenso, da explicação das equações do Principio do Máximo, pois são em tudo iguais ás apresentadas no 5.3.1, daí não ser necessário haver uma repetição do que já foi explicado.

Obteve-se a seguinte figura com todos os percursos abordados:

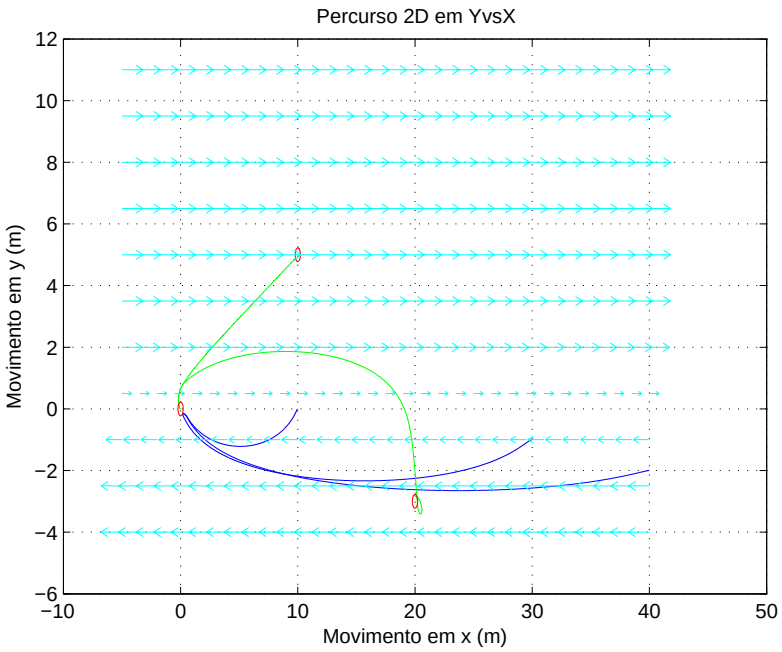

Figura 5.30: Todos os percursos do Caso 5 (Plano X vs Y)

Na figura 5.30, apresentam-se: a azul, todos os casos com ponto de chegada à vizinhança de $(0,0)$, e a verde os casos com ponto de partida de $(0,0)$. A azul ciano apresentam-se as movimentações das correntes marítimas, para o problema específico.

O seguinte quadro (5.5) refere os valores específicos de cada situação:

Tabela 5.5: Resultados dos Casos de Estudo de Consumo de Energia Mínima com Correntes

Energia Mínima	**Ponto Partida**	**Ponto Chegada**	$x(t_f)$	$y(t_f)$	**Objetivo:** $x_f^2 + y_f^2 \leq raio^2$
14.8688	(40,-2)	(0,0)	0.161417	-0.154740	0.0500
14.5202	(10,5)	(0,0)	0.152982	-0.163084	0.0500
4.4382	(5,5)	(0,0)	0.168760	-0.1466967	0.0300
5.18254	(0,0)	(5,5)	9.697510	4.907309	0.0500
11.3382	(0,0)	(15,10)	19.874846	-2.814695	0.0500

Capítulo 6

Conclusões e Trabalhos Futuros

6.1 Conclusões

Conclui-se que o principal objetivo desta dissertação foi alcançado, que seria o de validar através do principio do máximo os resultados obtidos. Teoricamente, com este passo dado, os controladores que se obtiveram, nos casos simulados, funcionariam no mundo real.

Concluiu-se também que a ferramenta usada (*AMPL*), consegue obter bons resultados, na modelação matemática de problemas e simular o seu comportamento.

As técnicas de controlo ótimo, como se observou, podem resolver imensos problemas em diversas áreas. Em engenharia então, tem grande relevância para se aprimorar os diversos tipos de controlo presentes nos mais diversos processos e sistemas.

Este projeto foi possivelmente a ponta do "*iceberg*"para estudar o planeamento otimizado do movimento de um robot submarino, simplificaram-se bastante os sistemas envolvidos, dando um ponto de partida para trabalhos cada vez mais próximos do mundo real. Deixam-se sugestões acerca deste ponto em 6.2.

Podemos observar pelos resultados, que o tempo mínimo nas situações com correntes é menor que nas situações sem correntes, isso deve-se ao facto do *AUV* aproveitar as correntes para poupar tempo, aumentando a sua velocidade. Daí nesses casos haver um menor consumo de energia, pois poderá gerir melhor as suas movimentações com a ajuda dos fluxos de água. De salientar também que o AUV é um sistema holinômico, ou seja, consegue mudar a sua direção sem se mover do ponto onde se encontra. Ao obrigar o mesmo a terminar rodado para uma posição específica, mesmo que essa em nada tenha a ver com a sua direção ao ponto pretendido, poupa-se tempo e consumo de energia, pois o AUV não terá que se movimentar para o conseguir, bastando apenas fazer uma rotação no seu próprio eixo.

6.2 Trabalho Futuro

Deixam-se aqui sugestões, como possíveis de serem estudadas em trabalho futuro.

Pode-se começar por falar do software aplicado (*AMPL*), abordado no capítulo 4; apesar de ser um bom software na resolução de problemas de otimização, obriga a que seja declarada uma discretização manual bem detalhada no ficheiro a correr no mesmo; deixando-se aqui, uma sugestão de outras ferramentas, possíveis de serem usadas, em que essa discretização é feita já pelo próprio programa tal como se verifica na ferramenta *ICLOCS*, que para uma compreensão mais detalhada do mesmo, o leitor poderá consultar [3].

Nesta dissertação foram apresentados casos de estudo para *AUV's*, com equações simplificadas, com correntes e já numa fase mais complexa com a dinâmica dos veículos. Deixa-se aqui como ideia, a realização e validação de resultados, com percursos que contenham obstáculos; poder-se-á também dimensionar o circuito, nos moldes de uma área para testes reais, a estudar com os submarinos; e também se poderá dimensionar o veículo, passando-se a tratar de um objeto com dimensões próximas, ou quem sabe, reais, e deixando-se de lado a ideia de uma partícula.

Refere-se a possibilidade de procura de métodos iterativos, mais complexos, que obtenham melhores resultados para o desempenho dos robots; e pode-se também procurar outros *solvers*, possíveis de melhorar os resultados obtidos futuramente.

Notou-se ao longo das simulações no tempo mínimo, que o programa era bastante sensível quanto à definição de parâmetros iniciais. Ora, desconfiou-se que ao discretizar o problema e fazendo uma avaliação profunda, que se pode pôr em hipótese a existência de mínimos locais próximos da solução ótima dos problemas pretendidos. Aliás, em conversa particular com um especialista na área, o mesmo garantiu ser esse o caso em problemas deste gênero, os denominados "Problemas de *Zermelo*"(para mais informação acerca do assunto aqui referenciado, o leitor pode consultar [23]). Assim, como trabalho futuro, propõe-se que, seja feito um estudo acerca deste fenômeno.

Anexo A

Caso 1

A.1 Ficheiro *AMPL*

#!/opt/ampl/ampl
shell "clear";

param n = 2000 ;
param pi = 4*atan(1);
param xf = 0;
param yf = 0;

State variables var tf >=0;
var x {i in 0..n} ;
s.t. Ix1 : x[0] = 40 ;
var y {i in 0..n} ;
s.t. Iy1 : y[0] = -2;
var psi {i in 0..n} ;
s.t. Ipsi1 : psi[0] = pi ;

Control variables
var u {i in 0..n};
var r {i in 0..n};

OBJECTIVE :
minimize OBJ: tf ;

Righ hand sides of ODEs
var fx {i in 0..n} = u[i]*cos(psi[i]);
var fy {i in 0..n} = u[i]*sin(psi[i]);

```
var fpsi {i in 0..n} = r[i];
```

EULER method

```
s.t. lx {i in 0..n-1} : x[i+1] = x[i] + tf/n*fx[i] ;
s.t. ly {i in 0..n-1} : y[i+1] = y[i] + tf/n*fy[i] ;
s.t. lpsi{i in 0..n-1} :psi[i+1]=psi[i] +tf/n*fpsi[i] ;
```

Control constaints

```
s.t. mu_u {i in 0..n-1} : 0 <= u[i] <= 2 ;
s.t. mu_r {i in 0..n-1} : -pi <= r[i] <= pi ;
s.t. bc : (xf- x[n])^ 2+(yf-y[n])^ 2 <= 0.05;
```

Initial estimates

```
let {i in 0..n} x[i] := 0;
let {i in 0..n} y[i] := 0;
let {i in 0..n} psi[i] := 0;
let {i in 0..n} u[i] := 0;
let {i in 0..n} r[i] := 0;
```

SOLVER

```
option solver ipopt;
option ipopt_options "max_iter=9999 acceptable_tol=1e-8 ";
solve;
```

PRINT SOLUTION in DATA FILES

```
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, x[i] > 'x.dat';
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, y[i] > 'y.dat';
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, psi[i] > 'psi.dat';
printf {i in 0..n} "%12.6f %12.6f \ n", i*tf/n, u[i] > 'u.dat';
printf {i in 0..n} "%12.6f %12.6f \ n", i*tf/n, r[i] > 'r.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, lx[i]/(tf/n) > 'mu_lx.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, ly[i]/(tf/n) > 'mu_ly.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, lpsi[i]/(tf/n) > 'mu_lpsi.dat';
printf "%18.10f ", OBJ > 'OBJ.dat';
display OBJ ;
```

A.2 Ficheiro *Matlab*

```
close all


load psi.dat
load x.dat
load y.dat
load u.dat
load r.dat
load OBJ.dat
load mu_lx.dat
load mu_ly.dat
load mu_lpsi.dat


n=2000;
tf=OBJ;
h=tf/n;
xf=0;
yf=0;
t=0:h:tf;


X=x(:,2);
Y=y(:,2);
Psi=psi(:,2);
U=u(:,2);
R=r(:,2);
MU_LX=mu_lx(:,2);
MU_LY=mu_ly(:,2);
MU_LPSI=mu_lpsi(:,2);


epsilon=sqrt(0.05);
g=0:h:tf-h;


figure();
subplot(3,2,1);
```

```
plot(t,X,'m');
title('Movimento em x')
axis([0,20,0,40]);
xlabel('tempo (s)');
ylabel('distancia percorrida (m)');
grid on;
subplot(3,2,2),plot(t,Y,'c');
title('Movimento em y')
axis([0,20,-2,0]);
xlabel('tempo (s)');
ylabel('distancia percorrida (m)');
grid on;
subplot(3,2,3),plot(t,Psi,'g');
title('Rotacao no eixo')
xlabel('tempo (s)');
ylabel('rotacao (rad)');
grid on;
subplot(3,2,4),plot(t,U,'r');
title('Velovidade em x')
axis([0,20,0,2.5])
xlabel('tempo (s)');
ylabel('velocidade (m/s)');
grid on;
subplot(3,2,5),plot(t,R,'y');
title('Angulo de Rotacao no eixo')
xlabel('tempo (s)');
ylabel('velocidade de rotacao (rad/s)');
grid on;


figure();
title('Multiplicadores')
subplot(2,2,1), plot(g,MU_LX,'g');
xlabel('tempo (s)');
ylabel('Multiplicador de x');
grid on;
subplot(2,2,2), plot(g,MU_LY,'b');
xlabel('tempo (s)');
ylabel('Multiplicador de y');
grid on;
```

```
subplot(2,2,3), plot(g,MU_LPSI,'r');
xlabel('tempo (s)');
ylabel('Multiplicador de psi');
grid on;


Fig_X_Y=figure();
plot(X,Y)
title('Percurso 2D em YvsX')
xlabel('Movimento em x (m)');
ylabel('Movimento em y (m)');
grid on
hold on
%bola=(xf- X(n+1))^2+(yf-Y(n+1))^2 <= 0.05;
t=0:0.01:2*pi;
x=epsilon*cos(t);
y=epsilon*sin(t);
plot(xf+x,yf+y,'r')
```

Anexo B

Caso 2

B.1 Ficheiro *AMPL*

```
#!/opt/ampl/ampl

shell "clear";


param n = 2000 ;
param pi = 4*atan(1);
param xf = 0;
param yf = 0;


State variables var tf = 20;
var x {i in 0..n} ;
s.t. Ix1 : x[0] = 40 ;
var y {i in 0..n} ;
s.t. Iy1 : y[0] = -2;
var psi {i in 0..n} ;
s.t. Ipsi1 : psi[0] = pi ;
var z {i in 0..n} ;
s.t. Iz1 : z[0] = 0;


Control variables
var u {i in 0..n};
var r {i in 0..n};
```

OBJECTIVE :

```
minimize OBJ: z[n] ;
```

Righ hand sides of ODEs

```
var fx {i in 0..n} = u[i]*cos(psi[i]);
var fy {i in 0..n} = u[i]*sin(psi[i]);
var fpsi {i in 0..n} = r[i];
var fz {i in 0..n} = 0.5*u[i]*u[i];
```

EULER method

```
s.t. lx {i in 0..n-1} : x[i+1] = x[i] + tf/n*fx[i] ;
s.t. ly {i in 0..n-1} : y[i+1] = y[i] + tf/n*fy[i] ;
s.t. lpsi{i in 0..n-1} :psi[i+1]=psi[i] +tf/n*fpsi[i] ;
s.t. lz {i in 0..n-1} : z[i+1] = z[i] + tf/n*fz[i];
```

Control constaints

```
s.t. mu_u {i in 0..n-1} : 0 <= u[i] <= 2 ;
s.t. mu_r {i in 0..n-1} : -pi <= r[i] <= pi ;
s.t. bc : (xf- x[n])^ 2+(yf-y[n])^ 2 <= 0.05;
```

Initial estimates

```
let {i in 0..n} x[i] := 0;
let {i in 0..n} y[i] := 0;
let {i in 0..n} psi[i] := 0;
let {i in 0..n} u[i] := 0;
let {i in 0..n} r[i] := 0;
let {i in 0..n} z[i] := 0;
```

SOLVER

```
option solver ipopt;
option ipopt_options "max_iter=9999 acceptable_tol=1e-8 ";
solve;
```

PRINT SOLUTION in DATA FILES

```
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, x[i] > 'x.dat';
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, y[i] > 'y.dat';
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, psi[i] > 'psi.dat';
```

```
printf {i in 0..n} "%12.6f %12.6f \ n", i*tf/n, u[i] > 'u.dat';
printf {i in 0..n} "%12.6f %12.6f \ n", i*tf/n, r[i] > 'r.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, lx[i]/(tf/n) > 'mu_lx.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, ly[i]/(tf/n) > 'mu_ly.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, lpsi[i]/(tf/n) > 'mu_lpsi.dat';
printf {i in 0..n-1} "%12.6f %12.6f\n", i*(tf/n), lz[i]/(tf/n) > 'mu_lz.dat';
printf "%18.10f ", OBJ > 'OBJ.dat';
display OBJ ;
```

B.2 Ficheiro *Matlab*

```
close all


load psi.dat
load x.dat
load y.dat
load u.dat
load r.dat
load mu_lx.dat
load mu_ly.dat
load mu_lpsi.dat
load mu_lz.dat


n=2000;
tf=20;
h=tf/n;
xf=0;
yf=0;
t=0:h:tf;
X=x(:,2);
Y=y(:,2);
Psi=psi(:,2);
U=u(:,2);
R=r(:,2);
MU_LX=mu_lx(:,2);
MU_LY=mu_ly(:,2);
MU_LPSI=mu_lpsi(:,2);
MU_LZ=mu_lz(:,2);
```

```
epsilon=sqrt(0.3);
g=0:h:tf-h;


figure();
subplot(3,2,1);
plot(t,X,'m');
title('Movimento em X')
xlabel('tempo (s)');
ylabel('distancia percorrida (m)');
grid on;
subplot(3,2,2),plot(t,Y,'c');
title('Movimento em Y')
xlabel('tempo (s)');
ylabel('distancia percorrida (m)');
grid on;
subplot(3,2,3),plot(t,Psi,'g');
title('Rotacao no eixo')
xlabel('tempo (s)');
ylabel('rotacao (rad)');
grid on;
subplot(3,2,4),plot(t,U,'r');
title('Velovidade em X')
xlabel('tempo (s)');
ylabel('velocidade (m/s)');
grid on;
subplot(3,2,5),plot(t,R,'y');
title('Angulo de Rotacao no eixo')
xlabel('tempo (s)');
ylabel('rotacao (rad/s)');
grid on;


figure();
title('Multiplicadores')
subplot(2,2,1), plot(g,MU_LX,'g');
xlabel('tempo (s)');
ylabel('Multiplicador de x');
grid on;
subplot(2,2,2), plot(g,MU_LY,'b');
```

```
xlabel('tempo (s)');
ylabel('Multiplicador de y');
grid on;
subplot(2,2,3), plot(g,MU_LPSI,'r');
xlabel('tempo (s)');
ylabel('Multiplicador de psi');
grid on;


Fig_X_Y=figure();
plot(X,Y)
title('Percurso 2D de Y sv X')
xlabel('Movimento em x (m)');
ylabel('Movimento em y (m)');
grid on
hold on
%bola=(xf- X(n+1))^2+(yf-Y(n+1))^2 <= 0.05;
t=0:0.01:2*pi;
x=epsilon*cos(t);
y=epsilon*sin(t);
plot(xf+x,yf+y,'r')
```

Anexo C

Caso 3

C.1 Ficheiro *AMPL*

```
#!/opt/ampl/ampl
shell "clear";

param n = 2000 ;
param pi = 4*atan(1);
param xf = 0;
param yf = 0;

State variables var tf >=0;
var x {i in 0..n} ;
s.t. Ix1 : x[0] = 40 ;
var y {i in 0..n} ;
s.t. Iy1 : y[0] = -2;
var psi {i in 0..n} ;
s.t. Ipsi1 : psi[0] = pi ;

Control variables
var u {i in 0..n};
var r {i in 0..n};

OBJECTIVE :
minimize OBJ: tf ;

Righ hand sides of ODEs
var fx {i in 0..n} = u[i]*cos(psi[i])+0.8*tanh(y);
var fy {i in 0..n} = u[i]*sin(psi[i]);
```

```
var fpsi {i in 0..n} = r[i];
```

EULER method

```
s.t. lx {i in 0..n-1} : x[i+1] = x[i] + tf/n*fx[i] ;
s.t. ly {i in 0..n-1} : y[i+1] = y[i] + tf/n*fy[i] ;
s.t. lpsi{i in 0..n-1} :psi[i+1]=psi[i] +tf/n*fpsi[i] ;
```

Control constaints

```
s.t. mu_u {i in 0..n-1} : 0 <= u[i] <= 2 ;
s.t. mu_r {i in 0..n-1} : -pi <= r[i] <= pi ;
s.t. bc : (xf- x[n])^ 2+(yf-y[n])^ 2 <= 0.05;
```

Initial estimates

```
let {i in 0..n} x[i] := 0;
let {i in 0..n} y[i] := 0;
let {i in 0..n} psi[i] := 0;
let {i in 0..n} u[i] := 0;
let {i in 0..n} r[i] := 0;
```

SOLVER

```
option solver ipopt;
option ipopt_options "max_iter=9999 acceptable_tol=1e-8 ";
solve;
```

PRINT SOLUTION in DATA FILES

```
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, x[i] > 'x.dat';
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, y[i] > 'y.dat';
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, psi[i] > 'psi.dat';
printf {i in 0..n} "%12.6f %12.6f \ n", i*tf/n, u[i] > 'u.dat';
printf {i in 0..n} "%12.6f %12.6f \ n", i*tf/n, r[i] > 'r.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, lx[i]/(tf/n) > 'mu_lx.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, ly[i]/(tf/n) > 'mu_ly.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, lpsi[i]/(tf/n) > 'mu_lpsi.dat';
printf "%18.10f ", OBJ > 'OBJ.dat';
display OBJ ;
```

C.2 Ficheiro *Matlab*

```
close all


load psi.dat
load x.dat
load y.dat
load u.dat
load r.dat
load OBJ.dat
load mu_lx.dat
load mu_ly.dat
load mu_lpsi.dat


n=2000;
tf=OBJ;
h=tf/n;
xf=0;
yf=0;
t=0:h:tf;


X=x(:,2);
Y=y(:,2);
Psi=psi(:,2);
U=u(:,2);
R=r(:,2);
MU_LX=mu_lx(:,2);
MU_LY=mu_ly(:,2);
MU_LPSI=mu_lpsi(:,2);


epsilon=sqrt(0.05);
g=0:h:tf-h;


Fig_X=figure();
plot(t,X)
```

```
title('plot X')
Fig_Y=figure();
plot(t,Y)
title('plot Y')
Fig_Psi=figure();
plot(t,Psi)
title('plot Psi')
Fig_U=figure();
plot(t,U)
title('plot u')


Fig_R=figure();
plot(t,R)
title('plot r')
Fig_MU_LX=figure();
plot(g,MU_LX)
title('plot MU_LX')
Fig_MU_LY=figure();
plot(g,MU_LY)
title('plot MU_LY')
Fig_MU_LPSI=figure();
plot(g,MU_LPSI)
title('plot MU_LPSI')


Fig_X_Y=figure();
plot(X,Y)
title('plot Y vs X')
grid on
hold on
%bola=(xf- X(n+1))^2+(yf-Y(n+1))^2 <= 0.05;
t=0:0.01:2*pi;
x=epsilon*cos(t);
y=epsilon*sin(t);
plot(xf+x,yf+y,'r')
```

Anexo D

Caso 4

D.1 Ficheiro *AMPL*

```
#!/opt/ampl/ampl

shell "clear";


param n = 2000 ;
param pi = 4*atan(1);
param xf = 0;
param yf = 0;


State variables var tf = 20;
var x {i in 0..n} ;
s.t. Ix1 : x[0] = 40 ;
var y {i in 0..n} ;
s.t. Iy1 : y[0] = -2;
var psi {i in 0..n} ;
s.t. Ipsi1 : psi[0] = pi ;
var z {i in 0..n} ;
s.t. Iz1 : z[0] = 0;


Control variables
var u {i in 0..n};
var r {i in 0..n};
```

OBJECTIVE :

```
minimize OBJ: z[n] ;
```

Righ hand sides of ODEs

```
var fx {i in 0..n} = u[i]*cos(psi[i])+0.8*tanh(y);
var fy {i in 0..n} = u[i]*sin(psi[i]);
var fpsi {i in 0..n} = r[i];
var fz {i in 0..n} = 0.5*u[i]*u[i];
```

EULER method

```
s.t. lx {i in 0..n-1} : x[i+1] = x[i] + tf/n*fx[i] ;
s.t. ly {i in 0..n-1} : y[i+1] = y[i] + tf/n*fy[i] ;
s.t. lpsi{i in 0..n-1} :psi[i+1]=psi[i] +tf/n*fpsi[i] ;
s.t. lz {i in 0..n-1} : z[i+1] = z[i] + tf/n*fz[i];
```

Control constaints

```
s.t. mu_u {i in 0..n-1} : 0 <= u[i] <= 2 ;
s.t. mu_r {i in 0..n-1} : -pi <= r[i] <= pi ;
s.t. bc : (xf- x[n])^ 2+(yf-y[n])^ 2 <= 0.05;
```

Initial estimates

```
let {i in 0..n} x[i] := 0;
let {i in 0..n} y[i] := 0;
let {i in 0..n} psi[i] := 0;
let {i in 0..n} u[i] := 0;
let {i in 0..n} r[i] := 0;
let {i in 0..n} z[i] := 0;
```

SOLVER

```
option solver ipopt;
option ipopt_options "max_iter=9999 acceptable_tol=1e-8 ";
solve;
```

PRINT SOLUTION in DATA FILES

```
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, x[i] > 'x.dat';
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, y[i] > 'y.dat';
printf {i in 0..n} "%12.6f %12.6f\n", i*tf/n, psi[i] > 'psi.dat';
```

```
printf {i in 0..n} "%12.6f %12.6f \ n", i*tf/n, u[i] > 'u.dat';
printf {i in 0..n} "%12.6f %12.6f \ n", i*tf/n, r[i] > 'r.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, lx[i]/(tf/n) > 'mu_lx.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, ly[i]/(tf/n) > 'mu_ly.dat';
printf {i in 0..n-1} "%12.6f %12.6f \ n", i*tf/n, lpsi[i]/(tf/n) > 'mu_lpsi.dat';
printf {i in 0..n-1} "%12.6f %12.6f\n", i*(tf/n), lz[i]/(tf/n) > 'mu_lz.dat';
printf "%18.10f ", OBJ > 'OBJ.dat';
display OBJ ;
```

D.2 Ficheiro *Matlab*

```
close all


load psi.dat
load x.dat
load y.dat
load u.dat
load r.dat
load OBJ.dat
load mu_lx.dat
load mu_ly.dat
load mu_lpsi.dat


n=2000;
tf=20;
h=tf/n;
xf=0;
yf=0;
t=0:h:tf;


X=x(:,2);
Y=y(:,2);
Psi=psi(:,2);
U=u(:,2);
R=r(:,2);
MU_LX=mu_lx(:,2);
MU_LY=mu_ly(:,2);
```

```
MU_LPSI=mu_lpsi(:,2);


epsilon=sqrt(0.05);
g=0:h:tf-h;


figure();
subplot(3,2,1);
plot(t,X,'m');
title('Movimento em x')
xlabel('tempo (s)');
ylabel('distancia percorrida (m)');
grid on;
subplot(3,2,2),plot(t,Y,'c');
title('Movimento em y')
xlabel('tempo (s)');
ylabel('distancia percorrida (m)');
grid on;
subplot(3,2,3),plot(t,Psi,'g');
title('Rotacao no eixo')
xlabel('tempo (s)');
ylabel('rotacao (rad)');
grid on;
subplot(3,2,4),plot(t,U,'r');
title('Velovidade em x')
xlabel('tempo (s)');
ylabel('velocidade (m/s)');
grid on;
subplot(3,2,5),plot(t,R,'y');
title('Angulo de Rotacao no eixo')
xlabel('tempo (s)');
ylabel('velocidade de rotacao (rad/s)');
grid on;


figure();
title('Multiplicadores')
subplot(2,2,1), plot(g,MU_LX,'g');
xlabel('tempo (s)');
```

```
ylabel('Multiplicador de x');
grid on;
subplot(2,2,2), plot(g,MU_LY,'b');
xlabel('tempo (s)');
ylabel('Multiplicador de y');
grid on;
subplot(2,2,3), plot(g,MU_LPSI,'r');
xlabel('tempo (s)');
ylabel('Multiplicador de psi');
grid on;


[xx1,yy1]=meshgrid(-1:0.5:40,-3:0.5:1);
Fig_X_Y=figure();
plot(X,Y)
title('Percurso 2D em Y vs X')
xlabel('Movimento em x (m)');
ylabel('Movimento em y (m)');
grid on
hold on
%bola=(xf- X(n+1))^2+(yf-Y(n+1))^2 <= 0.05;
t=0:0.01:2*pi;
x=epsilon*cos(t);
y=epsilon*sin(t);
plot(xf+x,yf+y,'r')
hold on
quiver(xx1,yy1,0.8*tanh(yy1),0*yy1,'c-')
```

Anexo E

Caso 5

E.1 Ficheiro *AMPL*

#!/opt/ampl/ampl
shell "clear";

param n = 2000 ;
param pi = 4*atan(1);
param xf = 0;
param yf = 0;

State variables var tf >=0;
var x {i in 0..n} ;
s.t. Ix1 : x[0] = 40 ;
var y {i in 0..n} ;
s.t. Iy1 : y[0] = -2;
var psi {i in 0..n} ;
s.t. Ipsi1 : psi[0] = pi ;
var u {i in 0..n} ;
s.t. Iu1 : u[0] = 0 ;

Control variables
var f {i in 0..n};
var r {i in 0..n};

OBJECTIVE :
minimize OBJ: tf ;

Righ hand sides of ODEs

```
var fx {i in 0..n} = u[i]*cos(psi[i])+0.8*tanh(y);
var fy {i in 0..n} = u[i]*sin(psi[i]);
var fpsi {i in 0..n} = r[i];
var fu {i in 0..n} = f[i] - u[i]*abs(u[i]);
```

EULER method

```
s.t. lx {i in 0..n-1} : x[i+1] = x[i] + tf/n*fx[i] ;
s.t. ly {i in 0..n-1} : y[i+1] = y[i] + tf/n*fy[i] ;
s.t. lpsi{i in 0..n-1} :psi[i+1]=psi[i] +tf/n*fpsi[i] ;
s.t. lu{i in 0..n-1} :u[i+1]=u[i] +tf/n*fu[i] ;
```

Control constaints

```
s.t. mu_u {i in 0..n-1} : 0 <= u[i] <= 2 ;
s.t. mu_r {i in 0..n-1} : -pi <= r[i] <= pi ;
s.t. bc : (xf- x[n])^ 2+(yf-y[n])^ 2 <= 0.05;
s.t. psifinal {i in 0..n-1} : psi[n]=pi;
s.t. ufinal {i in 0..n-1} : u[n]=0;
```

Initial estimates

```
let {i in 0..n} x[i] := 0;
let {i in 0..n} y[i] := 0;
let {i in 0..n} psi[i] := 0;
let {i in 0..n} u[i] := 0;
let {i in 0..n} r[i] := 0;
let {i in 0..n} f[i] := 0;
```

SOLVER

```
option solver ipopt;
option ipopt_options "max_iter=9999 acceptable_tol=1e-8 ";
solve;
```

PRINT SOLUTION in DATA FILES

```
printf i in 0..n "%12.6f %12.6f\n", i*tf/n, x[i] > 'x.dat';
printf i in 0..n "%12.6f %12.6f\n", i*tf/n, y[i] > 'y.dat';
printf i in 0..n "%12.6f %12.6f\n", i*tf/n, psi[i] > 'psi.dat';
printf i in 0..n "%12.6f %12.6f\n", i*tf/n, u[i] > 'u.dat';
```

```
printf i in 0..n "%12.6f %12.6f \ n", i*tf/n, f[i] > 'u.dat';
printf i in 0..n "%12.6f %12.6f \ n", i*tf/n, r[i] > 'r.dat';
printf i in 0..n-1 "%12.6f %12.6f \ n", i*tf/n, lx[i]/(tf/n) > 'mu_lx.dat';
printf i in 0..n-1 "%12.6f %12.6f \ n", i*tf/n, ly[i]/(tf/n) > 'mu_ly.dat';
printf i in 0..n-1 "%12.6f %12.6f \ n", i*tf/n, lpsi[i]/(tf/n) > 'mu_lpsi.dat';
printf i in 0..n-1 "%12.6f %12.6f \ n", i*tf/n, lpsi[i]/(tf/n) > 'mu_lu.dat';
printf "%18.10f ", OBJ > 'OBJ.dat';
display OBJ ;
```

E.2 Ficheiro *Matlab*

```
close all


load psi.dat
load x.dat
load y.dat
load u.dat
load r.dat
load f.dat
load OBJ.dat
load mu_lx.dat
load mu_ly.dat
load mu_lpsi.dat
load mu_lu.dat


n=2000;
tf=20;
h=tf/n;
xf=0;
yf=0;
t=0:h:tf;


X=x(:,2);
Y=y(:,2);
Psi=psi(:,2);
U=u(:,2);
R=r(:,2);
```

```
F=f(:,2);
MU_LX=mu_lx(:,2);
MU_LY=mu_ly(:,2);
MU_LPSI=mu_lpsi(:,2);
MU_LU=mu_lu(:,2);


epsilon=sqrt(0.05);
g=0:h:tf-h;


figure();
subplot(3,2,1);
plot(t,X,'m');
title('Movimento em x')
xlabel('tempo (s)');
ylabel('distancia percorrida (m)');
grid on;
subplot(3,2,2),plot(t,Y,'c');
title('Movimento em y')
xlabel('tempo (s)');
ylabel('distancia percorrida (m)');
grid on;
subplot(3,2,3),plot(t,Psi,'g');
title('Rotacao no eixo')
xlabel('tempo (s)');
ylabel('rotacao (rad)');
grid on;
subplot(3,2,4),plot(t,U,'r');
title('Velovidade em x')
xlabel('tempo (s)');
ylabel('velocidade (m/s)');
grid on;
subplot(3,2,5),plot(t,R,'y');
title('Angulo de Rotacao no eixo')
xlabel('tempo (s)');
ylabel('velocidade de rotacao (rad/s)');
grid on;
subplot(3,2,6),plot(t,F,'y');
title('Forca Aplicada ao Propulsor')
```

```
xlabel('tempo (s)');
ylabel('Forca (N)');
grid on;


figure();
title('Multiplicadores')
subplot(2,2,1), plot(g,MU_LX,'g');
xlabel('tempo (s)');
ylabel('Multiplicador de x');
grid on;
subplot(2,2,2), plot(g,MU_LY,'b');
xlabel('tempo (s)');
ylabel('Multiplicador de y');
grid on;
subplot(2,2,3), plot(g,MU_LPSI,'r');
xlabel('tempo (s)');
ylabel('Multiplicador de psi');
grid on;
subplot(2,2,3), plot(g,MU_LU,'r');
xlabel('tempo (s)');
ylabel('Multiplicador de u');
grid on;


[xx1,yy1]=meshgrid(-1:0.5:40,-3:0.5:1);
Fig_X_Y=figure();
plot(X,Y)
title('Percurso 2D em Y vs X')
xlabel('Movimento em x (m)');
ylabel('Movimento em y (m)');
grid on
hold on
%bola=(xf- X(n+1))^2+(yf-Y(n+1))^2 <= 0.05;
t=0:0.01:2*pi;
x=epsilon*cos(t);
y=epsilon*sin(t);
plot(xf+x,yf+y,'r')
hold on
quiver(xx1,yy1,0.8*tanh(yy1),0*yy1,'c-')
```

Bibliografia

[1] Nuno Abreu, Anıbal Matos, Patrıcia Ramos, and Nuno Cruz. Automatic interface for auv mission planning and supervision. In *OCEANS 2010*, pages 1–5. IEEE, 2010.

[2] Jaime Deetz. Navy's ocean-powered drones to wage underwater war, December 2013.

[3] Paola Falugi, Eric Kerrigan, and Eugene van Wyk. Imperial college london optimal control software user guide (iclocs). *Department of Electrical and Electronic Engineering, Imperial College London, London, England, UK*, 2010.

[4] Enrique Fernández-Perdomo, Daniel Hernández-Sosa, Josep Isern-González, Jorge Cabrera-Gámez, Antonio C Dominguez-Brito, and Vıctor Prieto-Maranón. Single and multiple glider path planning using an optimization-based approach. In *OCEANS, 2011 IEEE-Spain*, pages 1–10. IEEE, 2011.

[5] Bruno Ferreira, Miguel Pinto, Aníbal Matos, and Nuno Cruz. Control of the mares autonomous underwater vehicle. In *OCEANS 2009, MTS/IEEE Biloxi-Marine Technology for Our Future: Global and Local Challenges*, pages 1–10. IEEE, 2009.

[6] M. M. A. Ferreira. Optimal control problems, November 2010. Seminário FCUP- 26 de Novembro de 2010.

[7] Thor I Fossen. *Marine control systems: guidance, navigation and control of ships, rigs and underwater vehicles*. Marine Cybernetics, 2002.

[8] R. Fourer, D.M. Gay, and B.W. Kernighan. *AMPL: A Modeling Language for Mathematical Programming*. Scientific Press series. Thomson/Brooks/Cole, 2003.

[9] Robert Fourer, David M. Gay, and Brian W. Kernighan. A modeling language for mathematical programming. *Management Science*, 36(5):519–554, 1990.

[10] A. Gilat. *Matlab com Aplicações em Engenharia*. Bookman.

[11] OceanSYS Group. Technology - systems and prototypes.

[12] Lorenz T. Biegler Hans Pirnay, Rodrigo López-Negrete. Optimal sensitivity based on ipopt. 4:307–331, december.

[13] Pina Heitor. *Métodos Numéricos*. Lisboa, Portugal, McGraw-Hill, 1995.

[14] Emili Hernández, Marc Carreras, Pere Ridao, and Angelos Mallios. Homotopic path planning for an auv on maps improved with scan matching. In *Navigation, Guidance and Control of Underwater Vehicles*, volume 3, pages 204–209, 2012.

[15] Aaron Strauss Jack Macki. *Introduction to Optimal Control Theory*.

[16] D.S. Naidu. *Optimal Control Systems*. Electrical Engineering Series. Taylor & Francis, 2002.

[17] J. Nocedal, A. Wächter, and R. Waltz. Adaptive barrier update strategies for nonlinear interior methods. *SIAM Journal on Optimization*, 19(4):1674–1693, 2009.

[18] Katsuhiko Ogata. *Modern Control Engineering*. Prentice Hall, 1997.

[19] PN Paraskevopoulos. *Modern control engineering*, volume 10. CRC Press, 2001.

[20] Fernando Lobo Pereira. IntroduÇÃo ao controlo Óptimo, maio 2006. Mini-Curso do DINCON 2006 - V Congresso Temático de Dinâmica, Controle e Aplicações.

[21] Roland Siegwart and Illah R Nourbakhsh. Autonomous mobile robots. *Massachusetts Institute of Technology*, 2004.

[22] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[23] Laszlo Techy. Optimal navigation in planar time-varying flow: Zermelo's problem revisited. *Intelligent Service Robotics*, 4(4):271–283, 2011.

[24] R. Vinter. *Optimal Control*. Modern Birkhäuser Classics. Springer, 2010.

[25] Andreas Wächter. An interior point algorithm for large-scale nonlinear optimization with applications in process engineering. January 2002.

[26] Kiam Beng Yeo, Teck Ho Wong, and Cheah Meng Ong. Modelling and manoeuvrability design of autonomous underwater vehicle. *Journal of Applied Sciences*, 14(10), 2014.