



CONTROLE DE ROBÔS MÓVEIS PARA FUTEBOL DE ROBÔS

Gabriel Antonio de Araujo Ribeiro
Lívia Chaves Paravidino

Projeto de Graduação apresentado ao Curso de Engenharia de Controle e Automação, da Escola Politécnica, Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Engenheiro de Controle e Automação.

Orientador: Alessandro Jacoud Peixoto D.Sc.

Rio de Janeiro
Fevereiro de 2017

CONTROLE DE ROBÔS MÓVEIS PARA FUTEBOL DE ROBÔS


Gabriel Antonio de Araujo Ribeiro


Lívia Chaves Paravidino

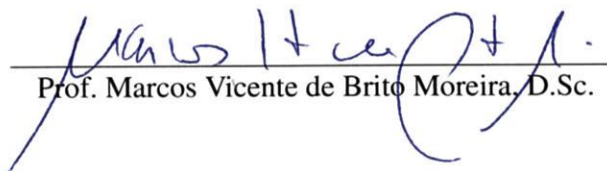
PROJETO DE GRADUAÇÃO SUBMETIDO AO CORPO DOCENTE DO CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO DA ESCOLA POLITÉCNICA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO DE CONTROLE E AUTOMAÇÃO.

Examinado por:


Prof. Alessandro Jacoud Peixoto, D.Sc.


Prof. Anna Carla Araújo, D.Sc.


Prof. Eduardo Nunes Vieira Leão, D.Sc.


Prof. Marcos Vicente de Brito Moreira, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
FEVEREIRO DE 2017

, Gabriel Antonio de Araujo Ribeiro

Lívia Chaves Paravidino

Controle de Robôs Móveis para Futebol de Robôs/Gabriel
Antonio de Araujo Ribeiro

Lívia Chaves Paravidino . – Rio de Janeiro: UFRJ/Escola
Politécnica, 2017.

XVII, 139 p.: il.; 29,7cm.

Orientador: Alessandro Jacoud Peixoto D.Sc.

Projeto de graduação – UFRJ/Escola Politécnica/Curso de
Engenharia de Controle e Automação, 2017.

Referências Bibliográficas: p. 114 – 116.

1. Futebol de Robôs. 2. Controle para Rastreamento.
3. Simulação. I. D.Sc., Alessandro Jacoud Peixoto. II. Universidade Federal do Rio de Janeiro, Escola Politécnica, Curso de Engenharia de Controle e Automação. III. Título.

*Dedicamos à nossa família e
amigos.*

Agradecimentos

Gostaríamos de agradecer à nossa família, por todo o incentivo durante a faculdade.

Ao nosso orientador Alessandro Jacoud, por todo o apoio e dedicação durante o desenvolvimento deste projeto.

Aos nossos amigos de Controle, Derek Chan, Ricardo Oliveira, Tiago Azevedo e Vinícius Plácido, que estiveram ao nosso lado durante o curso.

E principalmente, aos nossos amigos e companheiros da equipe Hefestos, Antonio Gaziera, Gabriel Lira, José Guilherme Monteiro, Pedro Gomes e Raphael Parreira que criaram esta equipe e muito se dedicaram à ela, sempre nos ajudando em tudo o que fosse necessário.

Resumo do Projeto de Graduação apresentado à POLI/UFRJ como parte dos requisitos necessários para a obtenção do grau de Engenheiro de Controle e Automação.

CONTROLE DE ROBÔS MÓVEIS PARA FUTEBOL DE ROBÔS

Gabriel Antonio de Araujo Ribeiro

Lívia Chaves Paravidino

Fevereiro/2017

Orientador: Alessandro Jacoud Peixoto D.Sc.

Curso: Engenharia de Controle e Automação

Este trabalho tem como objetivo descrever o desenvolvimento de uma bancada experimental e confecção de robôs. Detalhar a parte de programação que inclui a visão, comunicação e controle. Além de desenvolver um simulador e comparar os resultados obtidos para diferentes trajetórias e controles que possam ser aplicados ao futebol de robôs.

Palavras-chave: Futebol de Robôs, Controle para Rastreamento, Simulação.

Abstract of Undergraduate Project presented to POLI/UFRJ as a partial fulfillment of the requirements for the degree of Control and Automation Engineer.

MOBILE ROBOTS CONTROL FOR ROBOT SOCCER

Gabriel Antonio de Araujo Ribeiro

Lívia Chaves Paravidino

February/2017

Advisor: Alessandro Jacoud Peixoto D.Sc.

Course: Control and Automation Engineering

This work aims to describe the development of an experimental bench and robot making. Detail the part of programming that includes vision, communication and control. In addition to developing a simulator and comparing the results obtained for different trajectories and controls that can be applied to robot soccer.

Keywords: Robot Soccer, Tracking Control, Simulation.

Sumário

Lista de Figuras	x
1 Introdução	1
1.1 Revisão Bibliográfica	2
1.2 Objetivo	7
1.3 Organização do Trabalho	8
2 Hardware	9
2.1 Estrutura Mecânica	9
2.1.1 Estrutura Preliminar	9
2.1.2 Estrutura Desenvolvida	10
2.1.3 Robô Adquirido (Pololu 3PI Robot)	11
2.2 Eletrônica Embarcada	12
2.2.1 Eletrônica de Potência	12
2.2.2 Comunicação	13
2.2.3 Microcontrolador	14
3 Software	15
3.1 Interface Gráfica	16
3.2 Visão Computacional	19
3.3 Comunicação	22
3.3.1 Comunicação no Software	22
4 Modelagem	23
4.1 Descrição Expandida	26
5 Controle	30
5.1 Objetivo de Controle: Movimentos Usuais no Âmbito de Futebol de Robôs	30
5.2 Controle Desacoplado de Posição (Translação e Rotação)	31
5.2.1 Controle PD para Translação	32
5.2.2 Controle PD para Rotação	33
5.3 Controle de linearização por realimentação	35

5.3.1	Equações do controle de linearização por realimentação	35
6	Simulações	38
6.1	Simulador para o controle PD	38
6.2	Simulador para controle de linearização por realimentação	42
6.3	Movimentos com Bola Parada: Pênalti, Cobrança de Faltas e Início de Jogo	44
6.4	Movimentos de Defesa	50
6.5	Movimentos de Zagueiro e Atacante	56
7	Experimentos e Resultados	68
7.1	Estrutura da Bancada	68
7.2	Movimentos com Bola Parada: Pênalti, Cobrança de Faltas e Início de Jogo	69
7.3	Movimentos de Defesa	80
7.4	Movimentos de Zagueiro e Atacante	91
8	Conclusões, Contribuições e Trabalhos Futuros	112
	Referências Bibliográficas	114
A	Apêndice	117
A.1	Regras do Futebol de Robôs	117
A.2	Categorias das Competições	126
A.3	Simulador	130
A.4	Experimentos	138

Lista de Figuras

1.1	<i>Esquema do equipamento de comunicação via rádio utilizado neste projeto</i>	5
2.1	<i>Robô fabricado de acordo com o projeto anterior</i>	10
2.2	<i>Visão superior da estrutura mecânica com os eixos deslocados</i>	10
2.3	<i>Robô fabricado de acordo com o projeto atual</i>	11
2.4	<i>Robô adquirido</i>	11
2.5	<i>Esquemático Eletrônica</i>	12
2.6	<i>Esquemático de potência</i>	12
2.7	<i>Esquema do microcontrolador</i>	13
2.8	<i>Esquema do Driver do Motor</i>	13
2.9	<i>Microcontrolador</i>	14
3.1	<i>Interface: Tela principal</i>	16
3.2	<i>Interface: Botões de navegação</i>	16
3.3	<i>Interface: Tela de definição dos parâmetros do campo</i>	17
3.4	<i>Interface: Taxa de frames e robôs visíveis</i>	17
3.5	<i>Interface: Controles de interface</i>	18
3.6	<i>Interface: Níveis de bateria</i>	18
3.7	<i>Interface: Parâmetros do controle</i>	19
3.8	<i>Esquema de detecção da imagem</i>	19
3.9	<i>Padrão de cores no topo do robô</i>	20
3.10	<i>Blobs</i>	20
3.11	<i>Imagem da câmera</i>	21
3.12	<i>Detecção De Cores</i>	22
6.1	<i>Simulink simulador PD</i>	38
6.2	<i>Seletor de referência do simulador PD</i>	39
6.3	<i>Comparador de posição do simulador PD</i>	39
6.4	<i>Controlador PD do simulador PD</i>	39
6.5	<i>Conversor de sinal do simulador PD</i>	40
6.6	<i>Modelos cinemático e dinâmico do simulador PD</i>	40
6.7	<i>Função de exibição de dados do simulador PD</i>	41

6.8	<i>Simulink simulador para controle de linearização por realimentação . . .</i>	42
6.9	<i>Comparador de posição horizontal do simulador para controle de linearização por realimentação</i>	43
6.10	<i>Comparador de posição vertical do simulador para controle de linearização por realimentação</i>	43
6.11	<i>Comparador angular do simulador para controle de linearização por realimentação</i>	44
6.12	<i>Movimento Real da simulação de cobrança de pênalti utilizando o controle PD para o modelo cinemático.</i>	45
6.13	<i>Sinais de Controle da simulação de cobrança de pênalti utilizando o controle PD para o modelo cinemático.</i>	45
6.14	<i>Posições linear e angular da simulação de cobrança de pênalti utilizando o controle PD para o modelo cinemático.</i>	46
6.15	<i>Movimento Real da simulação de cobrança de pênalti utilizando o controle PD para o modelo dinâmico.</i>	46
6.16	<i>Sinais de Controle da simulação de cobrança de pênalti utilizando o controle PD para o modelo dinâmico.</i>	47
6.17	<i>Posições linear e angular da simulação de cobrança de pênalti utilizando o controle PD para o modelo dinâmico.</i>	47
6.18	<i>Comparação entre os modelos cinemático e dinâmico para a cobrança de pênalti utilizando o controle PD</i>	48
6.19	<i>Movimento Real da simulação de cobrança de pênalti utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	49
6.20	<i>Sinais de Controle da simulação de cobrança de pênalti utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	49
6.21	<i>Posições linear e angular da simulação de cobrança de pênalti utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	50
6.22	<i>Movimento Real da simulação de movimento de defesa utilizando o controle PD para o modelo cinemático.</i>	51
6.23	<i>Sinais de Controle da simulação de movimento de defesa utilizando o controle PD para o modelo cinemático.</i>	51
6.24	<i>Posições linear e angular da simulação de movimento de defesa utilizando o controle PD para o modelo cinemático.</i>	52
6.25	<i>Movimento Real da simulação de movimento de defesa utilizando o controle PD para o modelo dinâmico.</i>	52
6.26	<i>Sinais de Controle da simulação de movimento de defesa utilizando o controle PD para o modelo dinâmico.</i>	53
6.27	<i>Posições linear e angular da simulação de movimento de defesa utilizando o controle PD para o modelo dinâmico.</i>	53

6.28	<i>Comparação entre os modelos cinemático e dinâmico para o movimento de defesa utilizando o controle PD.</i>	54
6.29	<i>Movimento Real da simulação de movimento de defesa utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	55
6.30	<i>Sinais de Controle da simulação de movimento de defesa utilizando o controle de linearização por realimentação para o modelo cinemático.</i> . .	55
6.31	<i>Posições linear e angular da simulação de movimento de defesa utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	56
6.32	<i>Movimento Real da simulação de trajetória circular utilizando o controle PD para o modelo cinemático.</i>	57
6.33	<i>Sinais de Controle da simulação de trajetória circular utilizando o controle PD para o modelo cinemático.</i>	57
6.34	<i>Posições linear e angular da simulação de trajetória circular utilizando o controle PD para o modelo cinemático.</i>	58
6.35	<i>Movimento Real da simulação de trajetória circular utilizando o controle PD para o modelo dinâmico.</i>	58
6.36	<i>Sinais de Controle da simulação de trajetória circular utilizando o controle PD para o modelo dinâmico.</i>	59
6.37	<i>Posições linear e angular da simulação de trajetória circular utilizando o controle PD para o modelo dinâmico.</i>	59
6.38	<i>Comparação entre os modelos cinemático e dinâmico para trajetória circular utilizando o controle PD.</i>	60
6.39	<i>Movimento Real da simulação de trajetória circular utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	60
6.40	<i>Sinais de Controle da simulação de trajetória circular utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	61
6.41	<i>Posições linear e angular da simulação de trajetória circular utilizando o controle de linearização por realimentação para o modelo cinemático.</i> .	61
6.42	<i>Movimento Real da simulação de curva de Lissajous utilizando o controle PD para o modelo cinemático.</i>	62
6.43	<i>Sinais de Controle da simulação de curva de Lissajous utilizando o controle PD para o modelo cinemático.</i>	63
6.44	<i>Posições linear e angular da simulação de curva de Lissajous utilizando o controle PD para o modelo cinemático.</i>	63
6.45	<i>Movimento Real da simulação de curva de Lissajous utilizando o controle PD para o modelo dinâmico.</i>	64
6.46	<i>Sinais de Controle da simulação de curva de Lissajous utilizando o controle PD para o modelo dinâmico.</i>	64

6.47	<i>Posições linear e angular da simulação de curva de Lissajous utilizando o controle PD para o modelo dinâmico.</i>	65
6.48	<i>Comparação entre os modelos cinemático e dinâmico para curva de Lissajous utilizando o controle PD.</i>	65
6.49	<i>Movimento Real da simulação de curva de Lissajous utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	66
6.50	<i>Sinais de Controle da simulação de curva de Lissajous utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	66
6.51	<i>Posições linear e angular da simulação de curva de Lissajous utilizando o controle de linearização por realimentação para o modelo cinemático.</i>	67
7.1	<i>Campo Desmontável</i>	68
7.2	<i>Suporte para a câmera e luzes.</i>	69
7.3	<i>Movimento Real do robô desenvolvido na cobrança de pênalti utilizando o controle P.</i>	70
7.4	<i>Sinais de Controle do robô desenvolvido na cobrança de pênalti utilizando o controle P.</i>	70
7.5	<i>Posições linear e angular do robô desenvolvido na cobrança de pênalti utilizando o controle P.</i>	71
7.6	<i>Movimento Real do robô adquirido na cobrança de pênalti utilizando o controle P.</i>	71
7.7	<i>Sinais de Controle do robô adquirido na cobrança de pênalti utilizando o controle P.</i>	72
7.8	<i>Posições linear e angular do robô adquirido na cobrança de pênalti utilizando o controle P.</i>	72
7.9	<i>Movimento Real do robô desenvolvido na cobrança de pênalti utilizando o controle PD.</i>	73
7.10	<i>Sinais de Controle do robô desenvolvido na cobrança de pênalti utilizando o controle PD.</i>	74
7.11	<i>Posições linear e angular do robô desenvolvido na cobrança de pênalti utilizando o controle PD.</i>	74
7.12	<i>Movimento Real do robô adquirido na cobrança de pênalti utilizando o controle PD.</i>	75
7.13	<i>Sinais de Controle do robô adquirido na cobrança de pênalti utilizando o controle PD.</i>	75
7.14	<i>Posições linear e angular do robô adquirido na cobrança de pênalti utilizando o controle PD.</i>	76
7.15	<i>Movimento Real do robô desenvolvido na cobrança de pênalti utilizando o controle PID.</i>	77

7.16	<i>Sinais de Control do robô desenvolvido na cobrança de pênalti utilizando o controle PID.</i>	77
7.17	<i>Posições linear e angular do robô desenvolvido na cobrança de pênalti utilizando o controle PID.</i>	78
7.18	<i>Movimento Real do robô adquirido na cobrança de pênalti utilizando o controle PID.</i>	79
7.19	<i>Sinais de Controle do robô adquirido na cobrança de pênalti utilizando o controle PID.</i>	79
7.20	<i>Posições linear e angular do robô adquirido na cobrança de pênalti utilizando o controle PID.</i>	80
7.21	<i>Movimento Real do robô desenvolvido no movimento de defesa utilizando o controle P.</i>	81
7.22	<i>Sinais de Controle do robô desenvolvido no movimento de defesa utilizando o controle P.</i>	81
7.23	<i>Posições linear e angular do robô desenvolvido no movimento de defesa utilizando o controle P.</i>	82
7.24	<i>Movimento Real do robô adquirido no movimento de defesa utilizando o controle P.</i>	82
7.25	<i>Sinais de Controle do robô adquirido no movimento de defesa utilizando o controle P.</i>	83
7.26	<i>Posições linear e angular do robô adquirido no movimento de defesa utilizando o controle P.</i>	83
7.27	<i>Movimento Real do robô desenvolvido no movimento de defesa utilizando o controle PD.</i>	84
7.28	<i>Sinais de Controle do robô desenvolvido no movimento de defesa utilizando o controle PD.</i>	84
7.29	<i>Posições linear e angular do robô desenvolvido no movimento de defesa utilizando o controle PD.</i>	85
7.30	<i>Movimento Real do robô adquirido no movimento de defesa utilizando o controle PD.</i>	86
7.31	<i>Sinais de Controle do robô adquirido no movimento de defesa utilizando o controle PD.</i>	86
7.32	<i>Posições linear e angular do robô adquirido no movimento de defesa utilizando o controle PD.</i>	87
7.33	<i>Movimento Real do robô desenvolvido no movimento de defesa utilizando o controle PID.</i>	88
7.34	<i>Sinais de Controle do robô desenvolvido no movimento de defesa utilizando o controle PID.</i>	88

7.35	<i>Posições linear e angular do robô desenvolvido no movimento de defesa utilizando o controle PID.</i>	89
7.36	<i>Movimento Real do robô adquirido no movimento de defesa utilizando o controle PID.</i>	89
7.37	<i>Sinais de Controle do robô adquirido no movimento de defesa utilizando o controle PID.</i>	90
7.38	<i>Posições linear e angular do robô adquirido no movimento de defesa utilizando o controle PID.</i>	90
7.39	<i>Movimento Real do robô desenvolvido na trajetória circular utilizando o controle P.</i>	92
7.40	<i>Sinais de Controle do robô desenvolvido na trajetória circular utilizando o controle P.</i>	92
7.41	<i>Posições linear e angular do robô desenvolvido na trajetória circular utilizando o controle P.</i>	93
7.42	<i>Movimento Real do robô adquirido na trajetória circular utilizando o controle P.</i>	93
7.43	<i>Sinais de Controle do robô adquirido na trajetória circular utilizando o controle P.</i>	94
7.44	<i>Posições linear e angular do robô adquirido na trajetória circular utilizando o controle P.</i>	94
7.45	<i>Movimento Real do robô desenvolvido na trajetória circular utilizando o controle PD.</i>	95
7.46	<i>Sinais de Controle do robô desenvolvido na trajetória circular utilizando o controle PD.</i>	95
7.47	<i>Posições linear e angular do robô desenvolvido na trajetória circular utilizando o controle PD.</i>	96
7.48	<i>Movimento Real do robô adquirido na trajetória circular utilizando o controle PD.</i>	96
7.49	<i>Sinais de Controle do robô adquirido na trajetória circular utilizando o controle PD.</i>	97
7.50	<i>Posições linear e angular do robô adquirido na trajetória circular utilizando o controle PD.</i>	97
7.51	<i>Movimento Real do robô desenvolvido na trajetória circular utilizando o controle PID.</i>	98
7.52	<i>Sinais de Controle do robô desenvolvido na trajetória circular utilizando o controle PID.</i>	98
7.53	<i>Posições linear e angular do robô desenvolvido na trajetória circular utilizando o controle PID.</i>	99

7.54	<i>Movimento Real do robô adquirido na trajetória circular utilizando o controle PID.</i>	100
7.55	<i>Sinais de Controle do robô adquirido na trajetória circular utilizando o controle PID.</i>	100
7.56	<i>Posições linear e angular do robô adquirido na trajetória circular utilizando o controle PID.</i>	101
7.57	<i>Movimento Real do robô desenvolvido na curva de Lissajous utilizando o controle P.</i>	101
7.58	<i>Sinais de Controle do robô desenvolvido na curva de Lissajous utilizando o controle P.</i>	102
7.59	<i>Posições linear e angular do robô desenvolvido na curva de Lissajous utilizando o controle P.</i>	102
7.60	<i>Movimento Real do robô adquirido na curva de Lissajous utilizando o controle P.</i>	103
7.61	<i>Sinais de Controle do robô adquirido na curva de Lissajous utilizando o controle P.</i>	103
7.62	<i>Posições linear e angular do robô adquirido na curva de Lissajous utilizando o controle P.</i>	104
7.63	<i>Movimento Real do robô desenvolvido na curva de Lissajous utilizando o controle PD.</i>	105
7.64	<i>Sinais de Controle do robô desenvolvido na curva de Lissajous utilizando o controle PD.</i>	105
7.65	<i>Posições linear e angular do robô desenvolvido na curva de Lissajous utilizando o controle PD.</i>	106
7.66	<i>Movimento Real do robô adquirido na curva de Lissajous utilizando o controle PD.</i>	106
7.67	<i>Sinais de Controle do robô adquirido na curva de Lissajous utilizando o controle PD.</i>	107
7.68	<i>Posições linear e angular do robô adquirido na curva de Lissajous utilizando o controle PD.</i>	107
7.69	<i>Movimento Real do robô desenvolvido na curva de Lissajous utilizando o controle PID.</i>	108
7.70	<i>Sinais de Controle do robô desenvolvido na curva de Lissajous utilizando o controle PID.</i>	108
7.71	<i>Posições linear e angular do robô desenvolvido na curva de Lissajous utilizando o controle PID.</i>	109
7.72	<i>Movimento Real do robô adquirido na curva de Lissajous utilizando o controle PID.</i>	109

7.73	<i>Sinais de Controle do robô adquirido na curva de Lissajous utilizando o controle PID.</i>	110
7.74	<i>Posições linear e angular do robô adquirido na curva de Lissajous utilizando o controle PID.</i>	110
A.1	<i>Marcações do Campo</i>	117
A.2	<i>Bola Livre</i>	118
A.3	<i>Sistema Geral</i>	119
A.4	<i>Encobrimento da Bola</i>	120
A.5	<i>Tiro Livre</i>	124
A.6	<i>Tiro Livre</i>	125
A.7	<i>Tiro de Meta</i>	126
A.8	<i>Seletor de referência do simulador PD</i>	130
A.9	<i>Referência do pênalti para o simulador PD</i>	130
A.10	<i>Referência do movimento de defesa para o simulador PD</i>	131
A.11	<i>Referência circular para o simulador PD</i>	131
A.12	<i>Referência de uma curva de Lissajous para o simulador PD</i>	131
A.13	<i>Comparador de posição do simulador PD</i>	132
A.14	<i>Controlador PD do simulador PD</i>	133

Capítulo 1

Introdução

O intuito original na criação do futebol de robôs é colocar em campo uma equipe de robôs capazes de vencer uma partida de futebol contra uma equipe humana campeã da Copa do Mundo até 2050. A princípio pode parecer um objetivo bastante ousado de se alcançar nos próximos 34 anos, porém, observando-se o que eram os conceitos computacionais de 34 anos atrás em relação aos de agora, percebe-se que este objetivo pode ser alcançado. A primeira vez que vislumbrou-se esta possibilidade foi em 1997 quando a RoboCup, uma iniciativa científica internacional com o objetivo de avançar o estado da arte de robôs inteligentes, foi fundada. Desde então, a área expandiu-se para outros domínios de aplicação relevantes com base nas necessidades da sociedade moderna. Hoje, conceitos desenvolvidos através desta iniciativa de desenvolvimento no campo de futebol de robôs, podem ser aplicados em diversas áreas como robótica de serviços pessoais em espaços vivos, manipulação e fabricação no trabalho, e robótica de resgate.

No futebol de robôs dois times de 3 robôs com dimensões máximas de $75mm \times 75mm \times 75mm$ disputam uma partida de futebol segundo as regras da Confederação Brasileira de Robótica na categoria Very Small Size (VSS)[1]. Os robôs são controlados remotamente por um computador, onde as imagens obtidas por meio da câmera instalada sobre o campo são processadas e, com base nas informações extraídas destas imagens, algoritmos autônomos definem a estratégia a ser utilizada.

O jogo possui diversos desafios, desde a construção dos robôs que irão compor o time, passando pelo desenvolvimento dos algoritmos de visão computacional até o controle da posição dos robôs no campo de acordo com a estratégia elaborada.

Durante o jogo, as diversas situações colocam à prova o projeto. Uma simples disputa de bola engloba os mais diversos aspectos. Considerando essas situações por partes:

- A visão computacional deve ser capaz de observar e identificar tudo que está presente no jogo: jogadores, adversários e bola. Deve também ser adaptável à mudanças de iluminação durante o jogo, uma vez que durante o dia a iluminação do local

de disputa das partidas muda.

- Com as informações de posicionamento recebidas da visão processada, o programa define qual será o próximo ponto para o qual os robôs se movimentarão. Como são três robôs ao mesmo tempo controlados na partida, muitas vezes precisam ser definidas movimentações diferentes para cada robô. Nessa situação de disputa de bola tem-se um robô atacante se movendo em direção à bola, um robô zagueiro que deve se posicionar para roubar a bola caso o robô adversário ganhe a disputa e um robô goleiro cujo dever é proteger o gol caso o zagueiro seja driblado. Dessa maneira, o programa é responsável por elaborar três estratégias de movimentação diferentes e passar ao controle os pontos de referência de cada robô.
- O ponto de referência estabelecido pela estratégia é processado pelo controle que considera a distância de cada robô ao ponto desejado e se a referência do robô está direcionada para o ponto desejado. Vale ressaltar que é nesse ponto onde o controle decide se é melhor o robô movimentar-se para frente ou inverter a movimentação, uma vez que, precisando girar menos, o robô ganharia tempo e chegaria na frente na disputa de bola com o adversário. Com os devidos cálculos feitos, os parâmetros de controle são estabelecidos e os sinais de movimentação tanto de translação como de rotação do robô são passados ao algoritmo de comunicação.
- A comunicação recebe os sinais de controle e envia por meio de um comunicador na porta USB do computador. A comunicação é feita em um protocolo específico que evita a interferência de outros sinais que possam estar presentes no local da partida. O sinal então é recebido pelo microcontrolador presente na placa eletrônica de cada robô que processa a mensagem para pulsos de cada um dos dois motores do robô, finalmente executando a ação de movimentação.

Todas as situações anteriormente descritas compõem apenas um *loop* do processamento das informações do campo. A cada captura de imagem da câmera esse processo é feito, a estratégia analisa novamente o que deve fazer e os robôs são orientados a novos pontos.

Este trabalho aborda a avaliação experimental de métodos de controle convencional do robô móvel terrestre utilizado no futebol de robôs, sem abordar o problema de geração e planejamento de trajetória (estratégia do jogo).

1.1 Revisão Bibliográfica

Visando apresentar uma visão geral sobre os principais temas abordados neste projeto, nesta seção será apresentada uma revisão bibliográfica acerca dos seguintes temas: Futebol de Robôs, Visão Computacional, Comunicação e Controle.

Futebol de Robôs

No contexto latino americano, desde 2003, o Concurso Latino Americano de Robótica (LARC) e a Competição Brasileira de Robótica (CBR), que são promovidos pelo Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE) e pela RoboCup, englobam as competições de futebol de robôs, robôs de serviço, logística, de educação, de resgate, entre outros. As competições são abertas para estudantes e pesquisadores. [1]

No contexto global, a própria RoboCup desde 1997 é a organizadora das competições relacionadas ao futebol de robôs [2], assim como a *Federation of International Robot-soccer Association* (FIRA), que em 1995 criou o primeiro campeonato internacional na Coreia e tem realizado eventos anuais em diversos países, tendo a maior parte de competidores vindos de equipes asiáticas. [3]

As regras da categoria IEEE Very Small Size (VSS), que é a categoria abordada neste trabalho, assim como as demais categorias existentes estão disponíveis no apêndice.

Visão Computacional

A visão computacional engloba as mais diversas aplicações e está muito mais presente no cotidiano do que aparenta. As aplicações práticas ultrapassam as barreiras da robótica e inteligência artificial para alcançarem os campos mais relevantes da vida humana, como medicina, nanotecnologia, aplicações em exploração espacial entre outros.

Os temas mais comuns em visão computacional encontrados na literatura são sobre segmentação e identificação de objetos pela utilização de câmeras 2D. Os trabalhos elaborados nessas áreas podem ser divididos em dois grupamentos pela maneira como definem os algoritmos que realizam a identificação dos objetos detectados nas imagens.

O primeiro grupamento é caracterizado por algoritmos que agrupam *pixels* vizinhos de acordo com suas características em comum. Textura, cor e intensidade são algumas dessas características usadas para segmentação de imagens. O primeiro exemplo desses algoritmos são as técnicas por crescimento de regiões (*region growing*)[4, 5]. Nessa técnica são determinados pontos ou até regiões pequenas da imagem que vão se expandindo. A expansão é baseada em critérios de semelhança até que toda a imagem seja coberta por essas regiões.

Outra técnica é a de Modelos de contornos ativos (*Active Contours Model*) [6]. Onde os algoritmos possuem um contorno inicial que é deformado segundo uma função de energia global, para que o contorno inicial seja minimizado. A minimização ocorre pela interação entre a energia interna (que tende a manter o formato do contorno) e energia externa (que busca as zonas de grande gradiente para diminuir o contorno inicial proposto). Após a minimização, é alcançado o contorno real mais próximo dos objetos presentes na imagem.

Um terceiro exemplo é a divisão e fusão de regiões (*split and merge*)[7]. A imagem é definida inicialmente como apenas uma região que, de acordo com um critério de homo-

geneidade, é subdividida e reagrupada até estabelecer um contorno final.

O segundo grupamento não utiliza segmentações e busca identificar a imagem sem que sejam estabelecidos contornos ou fronteiras. Esses métodos tem um custo computacional maior, porém obtém resultados mais efetivos na identificação das imagens. O primeiro exemplo que podemos citar é o Detetor de Harris [8] que calcula diferenças de intensidade em diversas direções ao redor do ponto determinado para encontrar os limites do objeto. É interessante notar que este método se adapta à variações de escala e iluminação. Outro detetor, o Detetor de Kadir-Brady [9] busca as zonas mais raras na imagem, segundo critérios de entropia.

Podemos citar, também, os métodos SIFT(*Scale-Invariant Feature Transform*) [10] e SURF(*Speeded-Up Robust Feature*) [11]. O método SIFT primeiramente detecta os pontos de interesse da imagem segundo subtrações de uma pirâmide de imagens filtradas em variadas escalas. Esses pontos de interesse são caracterizados então em histogramas de direções de gradientes para a identificação da imagem. O método SURF foi, em parte, inspirado no método SIFT, que faz uma representação de distribuição de intensidade na vizinhança do ponto de interesse. O SURF é invariante à rotação da imagem pois utiliza o conceito de orientação dos *wavelets* de Harr. Além disso, esse método tem menor custo computacional se comparado ao SIFT, pois utiliza a técnica das imagens integrais na representação da imagem. [12, 13]

Existem ainda métodos de saliência visual [14–16] que são métodos inspirados no cérebro humano e na sua capacidade de detectar automaticamente as zonas de maior interesse de uma imagem. Um exemplo seria um método [17] que busca *pixels* nas áreas onde a probabilidade de encontrar um objetivo definido é a mais alta possível dado os descritores *features* do objeto procurado e a localização inicial do objeto na imagem.

Atualmente, o desenvolvimento de algoritmos para visão computacional ganhou o reforço das câmeras 3D. Existem câmeras 3D de baixo custo como *Kinect* e outras que possibilitam novas técnicas para segmentação e identificação de imagens como em [18, 19]. A idéia principal para a detecção de objetos é primeiramente detectar o chão e considerar todo o resto da imagem como objetos que são reagrupados conforme a distância definida. Dois pontos muito próximos são considerados como pertencentes ao mesmo objeto.

Nesse projeto, foi utilizada a técnica de reconhecimento de *blobs* [20] devido à sua facilidade de implementação no contexto da linguagem de programação C++ que foi implementada no software desenvolvido. Mais detalhes à respeito do que são *blobs* e como essa técnica foi empregada no projeto serão descritos no Capítulo 3.

Comunicação

Dentre todos os possíveis meios de comunicação sem fio, a mais comum em aplicações de robôs móveis é a comunicação via rádio.

A comunicação de rádio, independe se o transmissor e receptor estão voltados um

para o outro. Eles podem até mesmo operar através de paredes, em alguns casos. Sua transmissão é omnidirecional [21].



Figura 1.1: *Esquema do equipamento de comunicação via rádio utilizado neste projeto*

A comunicação por rádio conta com a propriedade elétrica chamada de indução. Quando a corrente elétrica é variada em um fio , gera uma onda eletromagnética correspondente que emana do fio e induz uma corrente elétrica em quaisquer outros fios do campo. A frequência do campo magnético é a mesma que a frequência da corrente no fio inicial. Isto significa que se quisermos enviar um sinal sem fio, podemos gerar uma corrente mudando em um fio em uma determinada frequência, e anexar um circuito para o segundo fio para detectar mudanças em curso a essa frequência. É dessa maneira que o rádio funciona [21].

A distância que podemos transmitir um sinal de rádio depende da intensidade do sinal, da sensibilidade do receptor, da natureza das antenas e dos obstáculos que bloqueiam o sinal. Quanto mais forte for a corrente original e mais sensíveis os receptores, o mais afastados o remetente e receptor podem estar. Os dois fios atuam como antenas. Qualquer condutor pode ser uma antena, mas alguns funcionam melhor do que outros. O comprimento e a forma da antena e a frequência do sinal afetam a transmissão [21].

Existem três tipos de dispositivos para sistemas de rádio : transmissores, que enviam um sinal , mas não conseguem receber um ; receptores , que recebem um sinal , mas não conseguem enviar um; e transceptores , que podem fazer as duas coisas. Pode-se questionar por que tudo não é um transceptor, visto que é o dispositivo mais flexível. Entretanto, é mais complexo fazer um transceptor do que fazer os outros dois. Em um transceptor , é necessário garantir que o receptor não está recebendo a transmissão do seu transmissor, ou eles vão interferir uns com os outros e não ouvir qualquer outro dispositivo. Para muitas aplicações , é mais barato usar um par transmissor - receptor e lidar com quaisquer erros por apenas transmitir a mensagem muitas vezes até o receptor recebê-lo. Assim funcionam controles remotos de TV, por exemplo. Isto os torna muito mais baratos.

Controle

Um controlador híbrido para a estabilização de um sistema não linear de n dimensões é abordado em [22, 23]. Este controlador consiste em uma parte de tempo discreto que

praticamente estabiliza um subconjunto de estados do sistema relacionado ao erro na direção frontal e em uma segunda parte de tempo contínuo que dirige os componentes de estado restantes arbitrariamente para pequena região ao redor de zero, estabilizando o erro lateral. Uma vantagem deste controle é que ele permite generalizações onde integradores podem ser colocados em cascata com as entradas do controle sem afetar a estabilidade da malha fechada, gerando entradas de controle mais suaves.

A junção de um controlador cinemático e um controlador de torque para o modelo dinâmico de um robô móvel não holonômico foi discutida em [24]. Este artigo fala que se existe um controlador adaptativo de rastreamento para o modelo cinemático com parâmetros desconhecidos, então esse controlador para o modelo dinâmico pode ser projetado através de uma abordagem adaptativa de *backstepping*.

Um método que utiliza um controlador de velocidade e um controlador de torque de forma independente é proposto em [25]. Nele, primeiro o controlador de velocidade é projetado para o sistema de direção cinemático com o objetivo de fazer com que o erro de rastreamento se aproxime de zero assintoticamente. Então, o controlador de torque calculado é projetado para que a verdadeira velocidade do robô convirja para o controlador de velocidade desejado. Dessa forma, o problema de regulação e rastreamento são tratados utilizando o controlador proposto, onde o robô pode seguir globalmente qualquer caminho, como uma reta, um círculo assim como para o caso de estacionamento em uma garagem.

O controle de rastreamento de posição/força de sistemas mecânicos lagrangianos com restrições não holonômicas clássicas é abordado em [26]. Nele, a estratégia de controle é desenvolvida no nível dinâmico para lidar com as incertezas do modelo mecânico do sistema, ela assegura o rastreamento da trajetória desejada do sistema em malha fechada, o erro de seguimento da força de restrição apresenta um limite controlável e é obtido um resultado de estabilidade assintótica global.

Um algoritmo com base na teoria da geometria diferencial que realiza a linearização exata da realimentação no modelo de erro cinemático de um robô móvel é discutido em [27]. Nele os controladores de rastreamento de trajetória são projetados através de uma atribuição de pólos. Quando a velocidade do ângulo do robô é permanentemente diferente de zero, um controlador assintoticamente estável é projetado. Já quando a essa velocidade é em algum momento igual a zero, a estratégia de controle de rastreamento de trajetória é dada com o rastreamento globalmente vinculado.

O artigo [28] propõe um controlador adaptativo para guiar um robô móvel tipo unicyclo em um rastreamento de trajetória. No início são gerados valores desejados para as velocidades linear e angular considerando apenas o modelo cinemático. Depois, esses valores são processados para compensar a dinâmica do robô, gerando os comandos de velocidades entregues aos atuadores do robô. A estabilidade do sistema é analisada usando a teoria de Lyapunov.

Um controlador dinâmico adaptativo de modo deslizante para um robô móvel é proposto em [29]. Primeiro um controlador cinemático é introduzido e depois um controlador dinâmico de modo deslizante adaptativo é proposto para fazer com que a velocidade real do robô alcance o comando de velocidade desejado, mesmo que o sistema apresente incertezas e perturbações. A convergência das equações é comprovada pela teoria de Lyapunov.

Um controle de modelo preditivo de rastreamento de trajetória é apresentado em [30]. Nele a dinâmica linearizada de rastreamento de erro é utilizada para prever o comportamento futuro do sistema e uma lei de controle é derivada de uma função de custo quadrática que penaliza o erro de rastreamento do sistema e o esforço do controle. O controlador proposto inclui restrições de velocidade e aceleração para evitar que o robô móvel escorregue e um preditor de Smith é usado para compensar o tempo morto do sistema de visão.

Um trabalho que utiliza a linearização por realimentação dinâmica como solução para o rastreamento de trajetória e regulação de setpoint para controlar um robô móvel com rodas em ambientes sem obstáculos é [31]. Este trabalho, pela proximidade do objetivo proposto, foi utilizado como base no desenvolvimento do controlador deste projeto.

A exigência de que o robô conheça seu ambiente de trabalho para adaptar seu funcionamento de acordo com o estado atual do mesmo implica em coleta e processamento de informação de diferentes tipos e sua adequada utilização no sistema de controle. Esta informação é gerada pelos sensores que, segundo a necessidade particular da tarefa a ser realizada, podem variar em número, em tipo e em complexidade [32]. A qualidade e quantidade de informação obtida permitirá controlar o veículo com trajetórias estáveis e sem oscilações, o que assegura que o robô móvel alcance o ponto de destino com mínimo erro e sem sofrer colisões ao longo do percurso.

As principais técnicas e sensores associados à estimação da posição de um robô móvel são resumidas em [33].

1.2 Objetivo

O objetivo deste trabalho é o desenvolvimento de uma bancada experimental para avaliar e desenvolver esquemas de controle de robôs móveis para serem utilizados em Futebol de Robôs.

O foco é desenvolver o veículo e avaliar esquemas de controle convencionais via simulações numéricas e experimentos, considerando movimentos pré-definidos.

1.3 Organização do Trabalho

Este trabalho será organizado da seguinte forma: o capítulo dois mostrará todas as informações relacionadas às estruturas mecânicas e à eletrônica embarcada consideradas neste projeto. Já a seção de software, que inclui a interface gráfica, visão computacional e comunicação será abordada no capítulo três.

O capítulo quatro explicará a modelagem de um robô móvel planar. As informações sobre os movimentos usuais no âmbito do futebol de robôs e o controle desenvolvido para este projeto serão descritos no capítulo cinco. No capítulo seis, serão abordadas as duas simulações desenvolvidas, assim como os resultados obtidos. O desenvolvimento da bancada, os experimentos e resultados verificados para diferentes táticas relacionadas ao futebol serão mostrados no capítulo sete.

Por fim, as conclusões obtidas, as contribuições e as metas para trabalhos futuros serão apresentadas no capítulo oito.

Capítulo 2

Hardware

Este capítulo apresenta a estrutura mecânica dos robôs considerados neste trabalho e o histórico da evolução do projeto. Serão abordadas as diferentes estruturas pensadas no desenvolvimento do robô, seus motivos de falha e seguintes evoluções.

2.1 Estrutura Mecânica

Nesta seção serão avaliadas três estruturas mecânicas abordadas no desenvolvimento deste projeto. Uma estrutura preliminar na qual se identificam algumas falhas, uma estrutura desenvolvida onde essas falhas são corrigidas, tornando-se então uma das estruturas foco do projeto e um robô adquirido para que se possa comparar a estrutura desenvolvida com uma encontrada no mercado especificamente criada para o mesmo fim do robô construído.

2.1.1 Estrutura Preliminar

Inicialmente, foi adotada uma estrutura que utilizava em sua base quatro peças de alumínio montadas em forma cúbica, onde ficavam os dois motores e duas omniballs para apoio. Neste bloco também estavam presas quatro hastes parafusadas que serviam de suporte para a placa que comportava os componentes eletrônicos com o circuito impresso, formando assim, a estrutura superior.

Eram utilizados motores de corrente contínua com caixa de redução, para aumentar o torque. O motor escolhido para esta tarefa foi o Faulhaber.

As rodas foram torneadas em resina e furadas para conexão com os motores. No torneamento foi feito um sulco para utilização de um O-Ring como "pneu" para aumentar a tração com o campo.

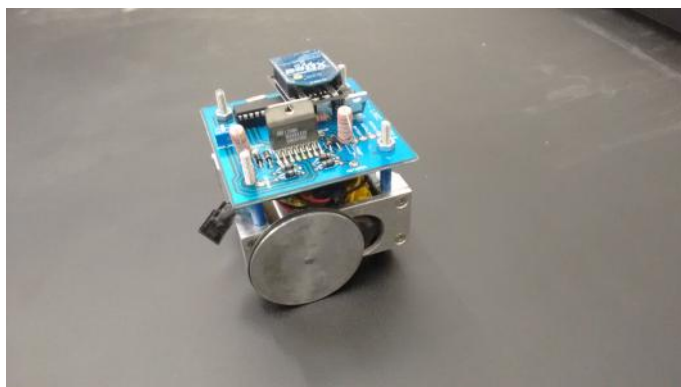


Figura 2.1: *Robô fabricado de acordo com o projeto anterior*

No entanto, essa estrutura aumentou consideravelmente o peso do robô, prejudicando o controle. A escolha dos motores também se mostrou equivocada, pois por serem superdimensionados e como o robô apresenta um limite de tamanho, foi necessário utilizá-los descentralizados na estrutura, como mostrado na figura 2.2, o que também prejudicou o controle.

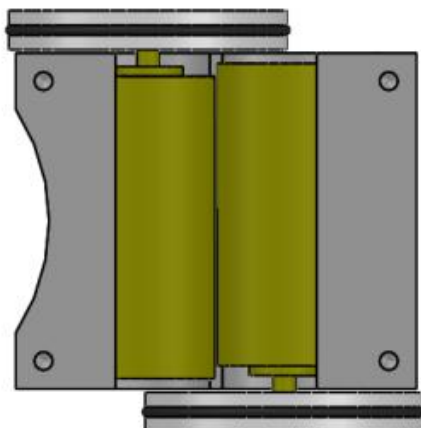


Figura 2.2: *Visão superior da estrutura mecânica com os eixos deslocados*

2.1.2 Estrutura Desenvolvida

Para contornar os problemas encontrados no projeto anterior descrito, foram propostas algumas mudanças no novo e atual projeto.

A base em quatro peças foi substituída por um novo modelo constituído em Lego com dois pontos de apoio e apenas duas astes para suporte da estrutura superior. As rodas agora utilizadas são pré fabricadas com pneus mais largos que proporcionam uma maior aderência.

Os motores atuais também foram redimensionados para um tamanho menor e com isso puderam ser colocados lado a lado na estrutura com seus eixos alinhados.

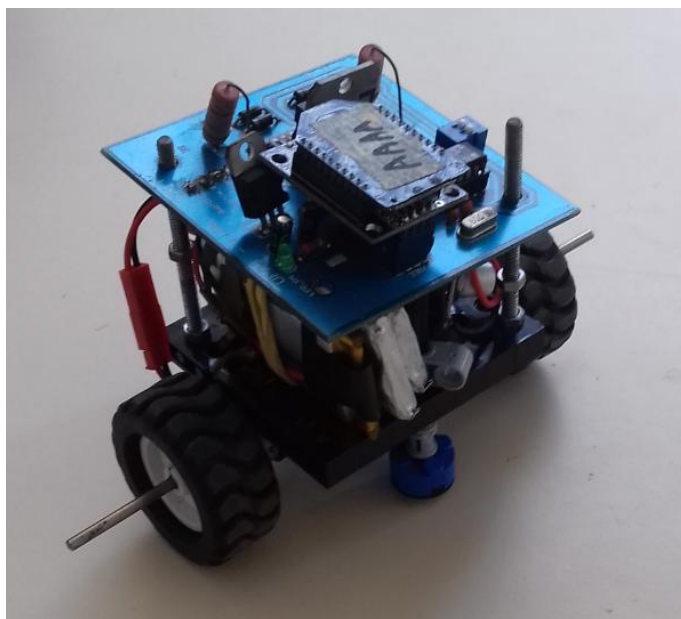


Figura 2.3: *Robô fabricado de acordo com o projeto atual*

2.1.3 Robô Adquirido (Pololu 3PI Robot)

Foi verificado entretanto, que apesar da melhora dos problemas do projeto inicial, algumas falhas de construção ainda poderiam atrapalhar a performance do robô a ser controlado. Por isso que, para efeito de teste, adquiriu-se o robô Pololu 3PI como mostra a figura 2.4.



Figura 2.4: *Robô adquirido*

Este robô foi projetado especificamente para competições em seguimento de linha e competições de resolução de labirinto. É um robô de pequeno tamanho (9,5 cm de diâmetro, 83 g sem baterias) quando comparado aos projetos anteriores. Seu regulador de tensão permite que atinja velocidades de até 1 m/s, enquanto faz curvas precisas e spins que não variam com a voltagem da bateria.[34]

O microcontrolador Atmel ATmega328P presente neste robô, fez que o projeto em si necessitasse de pouca adaptação em relação ao robô anterior que possui o mesmo microcontrolador. Pode-se inclusive chavear entre o uso deste robô e o robô do projeto anterior.

2.2 Eletrônica Embarcada

A eletrônica do robô é composta pelos quatro módulos principais expostos na Figura 2.5: power, Xbee, microcontrolador e driver do motor.

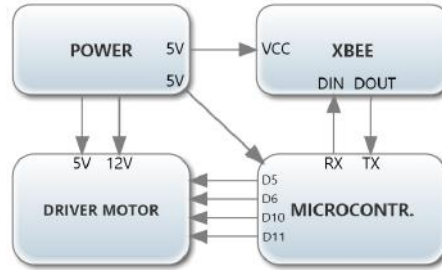


Figura 2.5: *Esquemático Eletrônica*

Escolheu-se o microcontrolador ATMEGA328 pela simplicidade e baixo custo. Para a transmissão, o módulo Xbee, que serve tanto como transmissor quanto receptor, proporcionando uma comunicação de duas vias entre o módulo central de controle e o robô. Este possui uma confiabilidade superior em relação à interferências se comparado a outros módulos, bem como um protocolo de comunicação mais robusto.

2.2.1 Eletrônica de Potência

Para o controle dos motores foi necessário utilizar um driver de motor (ponte H), sendo escolhido o circuito integrado DRV8835, que recebe do microcontrolador a direção e velocidade de rotação (PWM) e faz a conversão para a tensão de saída em cada motor.

Como o microcontrolador opera a 5V, os motores a 9V e o Xbee a 3.3V, foi utilizado um circuito externo para fornecer os 3.3V para o XBee bem como converter o nível lógico de 5V para 3.3V e vice-versa.

O esquemático do circuito eletrônico da parte de potência do jogador pode ser apreciado na Figura 2.6.

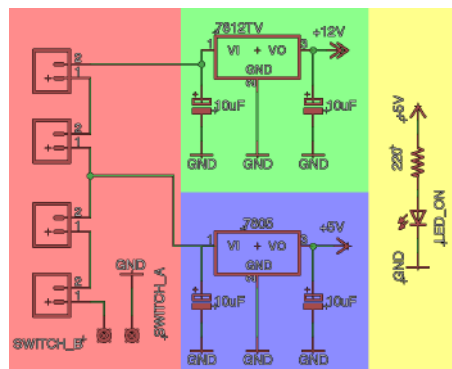


Figura 2.6: *Esquemático de potência*

Em vermelho estão os conectores padrão JST para as baterias. Como cada bateria tem 3.7V, a entrada na placa é de 11.1V. Em azul está destacado o regulador que alimenta o microcontrolador com 5V. O circuito que fornece 12V aos motores está marcado em verde. Nos dois casos os capacitores estão ligados em paralelo à entrada e à saída para filtrar variações nas mesmas. Em amarelo, o LED que indica que a placa está ligada é mostrado.

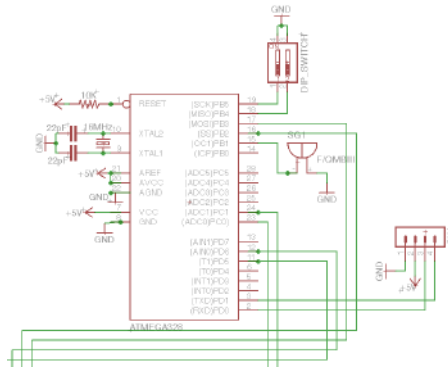


Figura 2.7: Esquema do microcontrolador

Na Figura 2.7 é mostrado o circuito do microcontrolador e as conexões que garantem seu funcionamento. Conectados a este está um terminal para conexão do circuito externo do XBee.

A Figura 2.8 mostra o driver do motor. Estão conectados ao L298, as entradas de 5V e 12V, os resistores para o sensor de corrente e os diodos responsáveis por evitar que a energia volte do motor para o circuito.

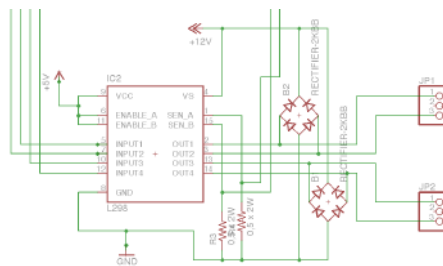


Figura 2.8: Esquema do Driver do Motor

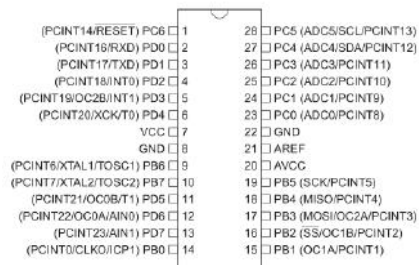
2.2.2 Comunicação

A escolha pelo comunicador XBee foi feita baseada na comparação com a comunicação via infra vermelho e sinal de rádio. Como em um contexto de futebol de robôs a linha de comunicação entre o componente de envio de dados e o robô pode estar obstruída ou sofrer interferências, optou-se pela comunicação via sinal de rádio. O comunicador XBee possibilita maior complexidade dos protocolos de comunicação, ou seja, é possível selecionar qual jogador em específico vai receber determinada mensagem.

2.2.3 Microcontrolador

O microcontrolador utilizado foi o ATmega328 que é um CMOS de baixa potência 8-bit baseado no AVR RISC. Ao executar instruções num único ciclo de relógio, o ATmega328 alcança taxa de transferência de aproximadamente 1 MIPS por MHz, o que permite um menor consumo de energia em função da velocidade de processamento.

Ele contém um processador, dois quilobytes de memória RAM para guardar dados, 32 quilobytes de memória EPROM (memória flash) para armazenar os programas, 1 quilobyte de memória EEPROM e ainda pinos de entrada e saída, que ligam o microcontrolador aos demais componentes eletrônicos. Essas entradas podem ler dados digitais (chave ligada/desligada) e analógicos (tensão em um pino). As saídas também podem ser analógicas ou digitais [35].



(a)



(b)

Figura 2.9: *Microcontrolador*

Capítulo 3

Software

O software desenvolvido é dividido em três áreas: visão, controle e comunicação.

A visão é responsável pela identificação e localização do robô no espaço. Estes dados de posicionamento são enviados então para o controle que calcula a velocidade que deve ser aplicada em cada motor afim de executar a ação desejada. Por fim, a comunicação tem como finalidade transmitir os comandos gerados via USB em um dispositivo de comunicação sem fio.

A visão captura e interpreta a imagem vinda da câmera. A captura é feita com uma biblioteca específica para isso, de modo a otimizar a taxa de aquisição. A interpretação da imagem é feita por padrões de imagem no topo do robô como mostrado na figura 3.9. Esses padrões indicam a direção do carrinho pelo modo como as duas circunferências estão dispostas. Um filtro é feito manualmente no espaço de cor HSV para organizar as cores de uma forma mais intuitiva, e portanto ser menos sensível à ruídos. Depois do filtro, os círculos são encontrados com um algoritmo de rastreamento, que encontra aglomerados de pontos de uma mesma cor. Com os centros de cada circunferência, é aplicada aritmética básica para encontrar o centro e a direção do robô.

Com os dados enviados pela visão, o controle mantém o carrinho seguindo o rumo que lhe foi determinado. No projeto do controle desenvolvido foram considerados a modelagem, a identificação de parâmetros da planta e o desenvolvimento do controlador. No controlador, as velocidades linear e angular do robô são desacopladas por meio de matrizes apropriadas, para que o controle de orientação seja projetado de forma independente do controle de posição. Tendo as velocidades calculadas, estas são passadas ao protocolo da comunicação.

A comunicação ocorre ao abrir uma porta de comunicação serial para que ocorra o envio de uma mensagem utilizando um protocolo que envia a velocidade calculada pelo controlador para o robô.

O robô possui um microcontrolador com um programa próprio para receber a mensagem e enviar a mesma para os motores.

Neste capítulo serão descritos a interface gráfica desenvolvida neste trabalho, as roti-

nas desenvolvidas para a comunicação do robô e os algoritmos associados à visão computacional necessários para obter a orientação e posição do veículo. Os algoritmos de controle serão descritos no Capítulo 5.

3.1 Interface Gráfica

Foi desenvolvida uma interface gráfica para possibilitar a interação do usuário com o software. A figura 3.1 mostra a interface base onde pode-se acessar a tela de controle, o controle de filtro das câmeras e observar quais robôs estão sendo capturados pela câmera.

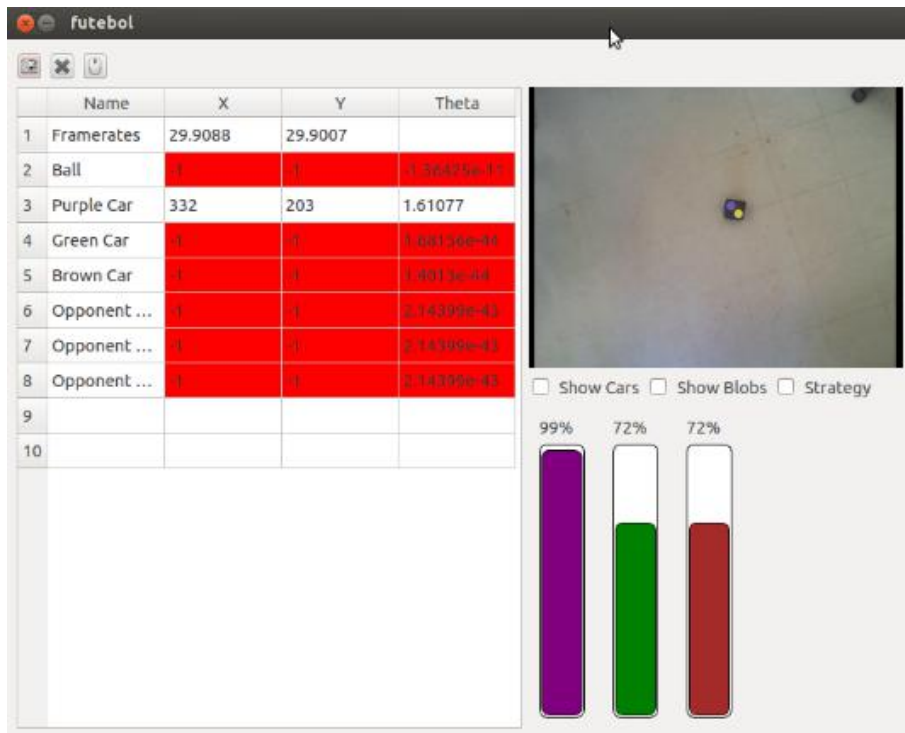


Figura 3.1: Interface: Tela principal

A figura 3.2 mostra os três botões para navegação através das demais telas do software. O primeiro deles (em formato de câmera) redireciona para a tela de controle dos filtros da câmera que será explicada na próxima seção. O segundo botão redireciona para a tela de definição dos parâmetros do campo de atuação do robô como mostrada na figura 3.3.



Figura 3.2: Interface: Botões de navegação

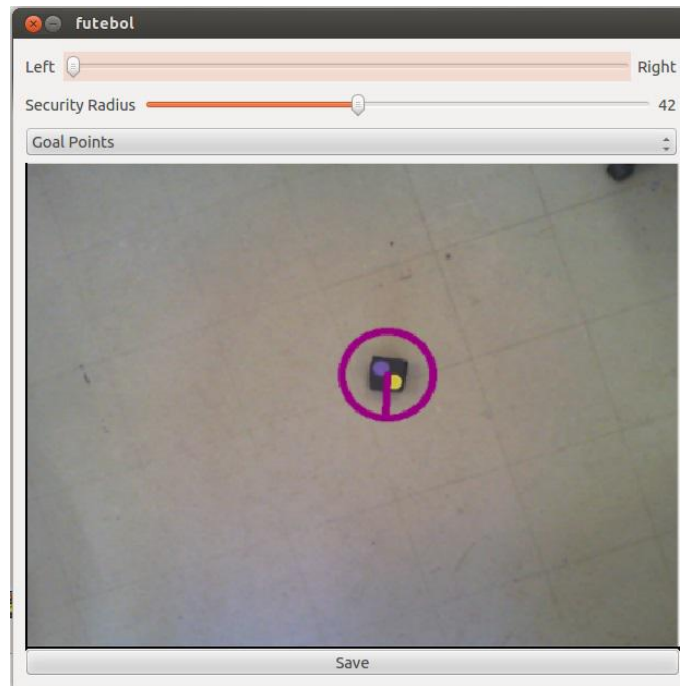


Figura 3.3: Interface: Tela de definição dos parâmetros do campo

E finalmente o terceiro botão da figura 3.2 exibe a tela do controle mostrado na figura 3.7.

A figura 3.4 mostra a taxa de frames de captura da câmera, a posição da bola, dos robôs dos dois times, bem como se estes estão visíveis ou não (cor vermelha).

	Name	X	Y	Theta
1	Framerates	29.9088	29.9007	
2	Ball	-1	-1	-1.36425e-11
3	Purple Car	332	203	1.61077
4	Green Car	-1	-1	1.00156e-44
5	Brown Car	-1	-1	1.4013e-13
6	Opponent ...	-1	-1	2.14399e-43
7	Opponent ...	-1	-1	2.14399e-43
8	Opponent ...	-1	-1	2.14399e-43

Figura 3.4: Interface: Taxa de frames e robôs visíveis

Na figura 3.5 são mostrados 3 *check boxes*, o primeiro permite mostrar os carros que estão sendo enxergados pela câmera. No segundo, observam-se os blobs captados pela câmera (o conceito de blob e sua função serão explicitados mais à frente) e o terceiro habilita a estratégia automática do software (trabalho futuro).



Figura 3.5: *Interface: Controles de interface*

A figura 3.6 mostra as barras que representam a carga das baterias em tempo real, ainda não implementado(trabalho futuro).

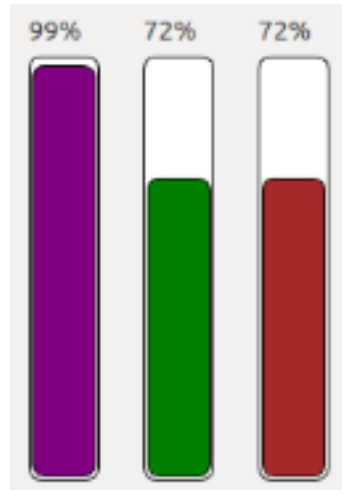


Figura 3.6: *Interface: Níveis de bateria*

A figura 3.7 mostra a tela de configuração dos parâmetros de controle. Na parte superior da tela é possível realizar a carga de um "perfil" de configuração de parâmetros. Nas caixas logo abaixo, pode-se selecionar o robô que deseja-se controlar bem como as trajetórias de referência utilizadas na coleta dos dados mostrados na parte de testes experimentais. Os dois *sliders* servem para definir a velocidade e a amplitude de variação da referência a ser seguida pelo robô. O *checkbox* "clockwise" inverte a direção da variação da referência na trajetória escolhida. As caixas de texto são as definições dos parâmetros do controle juntamente com os possíveis saltos em tempo real que podem ser executados nos mesmos através das teclas "para cima" e "para baixo". O botão "save" salva os parâmetros definidos no perfil escolhido.

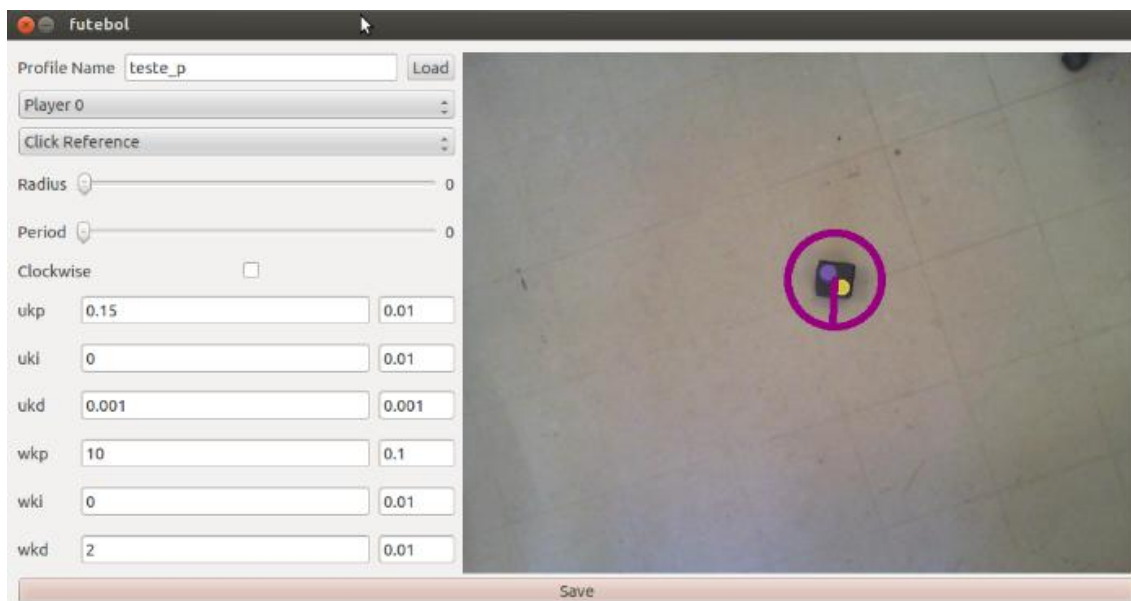


Figura 3.7: Interface: Parâmetros do controle

3.2 Visão Computacional

O desejado é que o programa atinja uma taxa de amostragem tão rápida quanto possível, para que a posição e orientação do robô sejam detectadas.



Figura 3.8: Esquema de detecção da imagem

Serão descritos nessa seção os algoritmos desenvolvidos para a detecção de imagens.

Detecção de Imagens

O pacote OpenCV fornece utilitários para a leitura a partir de uma grande variedade de tipos de arquivo de imagem, bem como de vídeo e câmeras, que fazem parte de um conjunto de ferramentas chamado HighGUI.

A biblioteca HighGUI pode ser dividida em três partes: a parte de hardware, a parte do sistema de arquivos e a parte GUI.

A parte de hardware está principalmente preocupada com a operação de câmeras. HighGUI permite uma maneira fácil de consultar uma câmera e recuperar sua última imagem.

A parte do sistema de arquivos está preocupada principalmente com carga e armazenamento das imagens. Um recurso dessa biblioteca é que ele nos permite ler vídeos usando os mesmos métodos usados para ler uma câmera. Ela também nos fornece um par universal de funções para carregar e salvar imagens fixas. Estas funções simplesmente

precisam da extensão do arquivo e lidam automaticamente com toda a decodificação ou codificação que é necessária.

A terceira parte do HighGUI é o sistema de janelas (ou GUI). A biblioteca fornece algumas funções simples que nos permitem abrir uma janela e jogar uma imagem nesta janela. Ela também nos permite registrar e responder a eventos do mouse e do teclado nesta janela. [20]

Detecção de Cores

O primeiro desafio foi detectar a presença de um ou mais robôs, sem preocupação com suas orientações.

Para tal, foi decidida a utilização de marcadores sobre os robôs que facilitassem sua localização.

Os marcadores adotados são feitos de discos de borracha EVA (Espuma Vinílica Acetinada), de aproximadamente 4 centímetros de diâmetro e de cores que sejam facilmente distinguíveis em relação ao fundo preto.

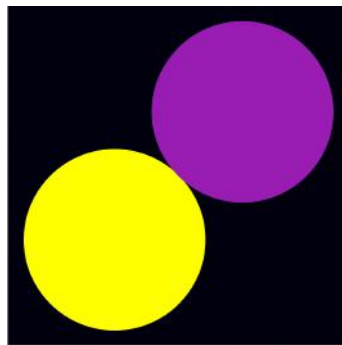


Figura 3.9: *Padrão de cores no topo do robô*

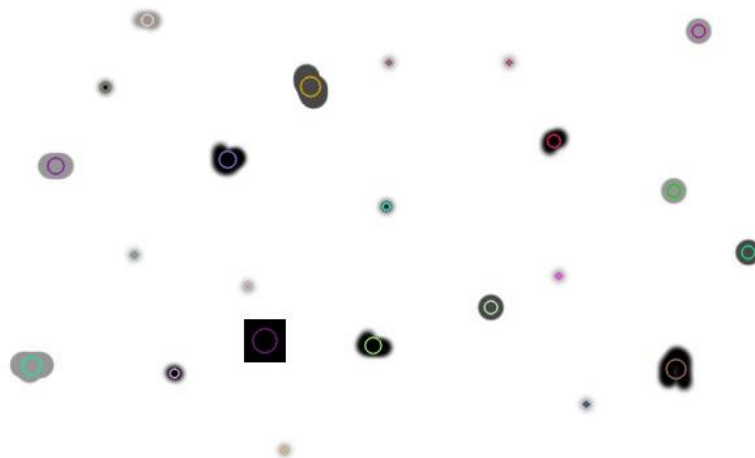


Figura 3.10: *Blobs*

Para a identificação dos padrões de cores foi utilizado o conceito de blobs. Um Blob é um grupo de pixels conectados em uma imagem que partilham alguma propriedade comum (por exemplo valor tons de cinza). Na imagem acima, as regiões escuras conectadas são bolhas e a meta de detecção de blob é identificar e marcar estas regiões. O OpenCV fornece uma maneira conveniente para detectar manchas/blobs e filtrá-los com base em características diferentes. Por exemplo pode-se filtrar os blobs por cores, tamanhos e formas.

A figura 3.11 mostra a imagem capturada pela câmera na interface do software em tempo real.



Figura 3.11: *Imagem da câmera*

Pode-se observar na figura 3.12 os filtros de cores utilizados na captura das imagens e reconhecimento dos padrões de cores pela câmera. Na caixa superior pode-se escolher qual cor será configurada, bem como adicionar uma nova cor. Na caixa seguinte é possível editar o nome da cor. Os *sliders* permitem escolher parâmetros mínimos e máximos para a captura dos espectros de cada cor, bem como uma área mínima e máxima para o reconhecimento de um blob. Pode-se também estabelecer uma distância mínima entre dois pixels para que estes sejam considerados como uma área a ser captada. É possível também selecionar o time a ser controlado de acordo com o padrão de cor no topo do robô. O botão "*save this color*" salva essas configurações para serem utilizadas pelo software.

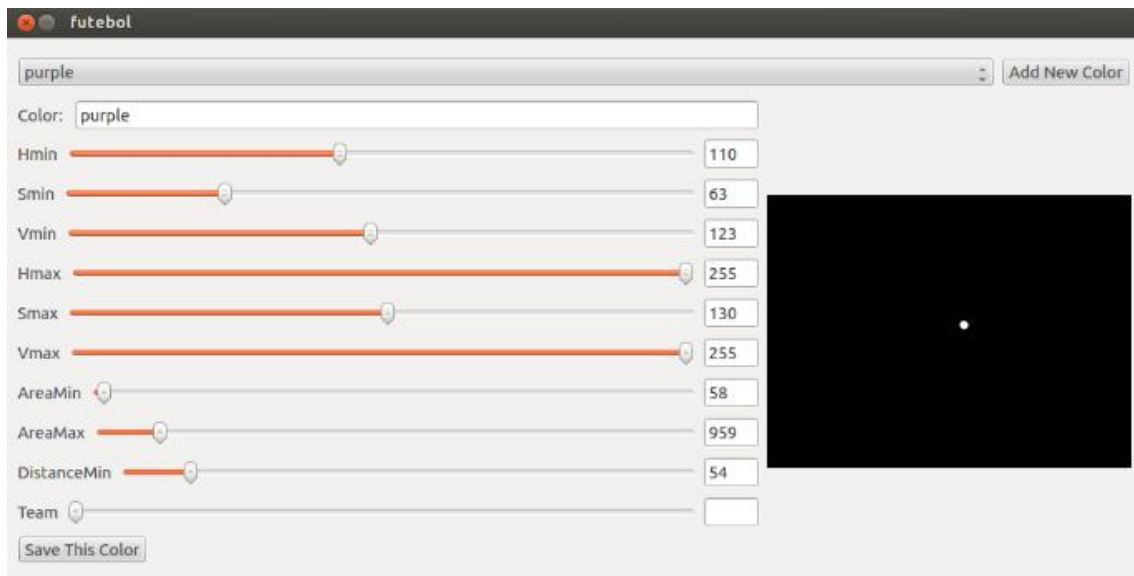


Figura 3.12: *Deteção De Cores*

3.3 Comunicação

A comunicação do sistema se dá de maneira distribuída entre o computador que executa o programa da interface gráfica e o microcontrolador embarcado em cada robô.

3.3.1 Comunicação no Software

Após receber e processar todas as informações do sistema, o controle envia uma mensagem com as informações sobre qual robô deve receber a mensagem, orientação na qual devem girar as rodas e sinais de controle. Essa mensagem serial é enviada através do XBee configurado como emissor conectado à porta USB do computador que gerencia toda a aplicação.

A mensagem então recebida pelo XBee configurado como receptor do robô destino, é passada ao microcontrolador que codifica o sinal de controle em sinais PWM (modulados por largura de pulso) que são mapeados de 0 à 255 de acordo com a intensidade do sinal de controle enviado.

O PWM definido pelo microcontrolador é então passado ao motor, juntamente com a informação de direção em que a roda deve girar, fazendo assim com que o robô se movimente.

Capítulo 4

Modelagem

A relação entre um vetor livre \mathcal{V} no plano representado no sistema de coordenadas inerciais $(\mathcal{V})^A$ e o mesmo vetor representado no sistema de coordenadas do corpo $(\mathcal{V})^B$ é governada pela matriz de rotação em torno do eixo perpendicular ao plano de movimento, da seguinte maneira:

$$(\mathcal{V})^A = R(\theta)(\mathcal{V})^B, \quad \text{onde} \quad R(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix},$$

onde θ é a orientação do robô.

Por isso verifica-se que a velocidade angular ω satisfaz

$$\dot{\theta} = \omega.$$

Definindo o vetor de coordenadas generalizadas $q := \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$, onde $\begin{bmatrix} x \\ y \end{bmatrix}$ é o vetor

posição do centro de massa G representado nas coordenadas inerciais. Nota-se que

$(a_G)^A = \begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix}$ e que o vetor velocidade do centro de massa representado nas coordenadas inerciais é dado por $(v_G)^A := \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}$. Pode-se escrever:

$$\dot{q} = \begin{bmatrix} (v_G)^A \\ \omega \end{bmatrix} = \mathcal{R}(\theta) \begin{bmatrix} (v_G)^B \\ \omega \end{bmatrix}.$$

Considerando a forças externas representadas no sistema de coordenadas do corpo (sistema móvel), a dinâmica do movimento de translação do centro de massa pode ser descrita por:

$$m(a_G)^A = R(\theta) \sum (F_{ext})^B, \quad (4.1)$$

Assim, a dinâmica do robô pode ser reescrita nas coordenadas inerciais da seguinte maneira:

$$M\ddot{q} = \begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} (F_{ext})^B \\ \Sigma \mathcal{M}_{extG} \end{bmatrix},$$

sendo

$$M := \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & \mathcal{I} \end{bmatrix},$$

e $\begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} (F_{ext})^B \\ \Sigma \mathcal{M}_{extG} \end{bmatrix}$ o vetor de forças generalizadas.

Considerando-se que apenas as forças de atrito compõem a resultante de forças externas que determinam o movimento do uniciclo, nota-se que, as componentes das forças de atrito das rodas tracionadas no eixo $(x)^B$ resultam da reação das forças de atuação nas rodas (forças de controle devido aos atuadores do robô). São desprezadas as forças devido a distúrbios como cabos presos ao robô e outras forças externas. Assim, o movimento de translação do centro de massa pode ser descrito por:

$$m(a_G)^A = R(\theta)(F_{at})^B, \quad (F_{at})^B = \begin{bmatrix} (f_{atx})^B \\ (f_{aty})^B \end{bmatrix}. \quad (4.2)$$

Assume-se também que o torque em torno do centro de massa gerado pelas componentes no eixo $(y)^B$ das forças de atrito das rodas traseiras é nulo ou desprezível¹. Portanto, o movimento de rotação baricêntrica pode ser descrito por:

$$\mathcal{I} \dot{\omega} = \tau, \quad (4.3)$$

onde $\tau = (f_1 - f_2)l$. A dinâmica do robô pode ser reescrita nas coordenadas inerciais como:

$$M\ddot{q} = \begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} (f_{atx})^B \\ (f_{aty})^B \\ \tau \end{bmatrix}, \quad M := \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & \mathcal{I} \end{bmatrix}, \quad (4.4)$$

ou, de forma equivalente,

$$M\ddot{q} = B(q)\mathcal{F} + J^T(q)(f_{aty})^B, \quad (4.5)$$

onde

$$\mathcal{F} := \begin{bmatrix} (f_{atx})^B \\ \tau \end{bmatrix}, \quad \text{e} \quad J^T(q) := \begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}.$$

¹ Assume-se que o centro de massa esteja sobre o eixo $(y)^B$ ou próximo dele.

Pode-se utilizar a restrição cinemática juntamente com a dinâmica do robô (4.5) para permitir expressar de forma mais simples a dinâmica do robô em um sistema de coordenadas mais adequado. Definindo $v := \begin{bmatrix} u \\ \omega \end{bmatrix}$, tem-se que:

$$\dot{q} = B(q)v, \quad (4.6)$$

Portanto, derivando (4.6) tem-se $\ddot{q} = B(q)\dot{v} + \dot{B}(q)v$, ou seja, $\ddot{q} = \begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} \dot{v} + \begin{bmatrix} R'(\theta) & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} v\omega = \begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{u} \\ \omega u \\ \dot{\omega} \end{bmatrix}$.

Desta forma, $M\ddot{q} = M \begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{u} \\ \omega u \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix} M \begin{bmatrix} \dot{u} \\ \omega u \\ \dot{\omega} \end{bmatrix}$ e de (4.4)

pode-se concluir que

$$M \begin{bmatrix} \dot{u} \\ \omega u \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} (f_{atx})^B \\ (f_{aty})^B \\ \tau \end{bmatrix},$$

pois $\begin{bmatrix} R(\theta) & 0 \\ 0 & 1 \end{bmatrix}$ é invertível. Logo a dinâmica do unicyclo pode ser descrita da seguinte forma reduzida:

$$\mathcal{M}\dot{v} = \begin{bmatrix} (f_{atx})^B \\ \tau \end{bmatrix} := \mathcal{F}, \quad \mathcal{M} := B^T M B = \begin{bmatrix} m & 0 \\ 0 & \mathcal{I} \end{bmatrix}, \quad (4.7)$$

onde \mathcal{F} passa a ser denominado vetor de forças generalizadas aplicadas associado às novas coordenadas v e \mathcal{M} é denominado o momento de inércia generalizado. Nota-se que, a componente da força de atrito $(f_{aty})^B$ que assegura o movimento restrito do unicyclo é dada nas novas coordenadas por $(f_{aty})^B = m\omega u$. Ressaltando que de (4.6)

$$J(q)\dot{q} = 0 \quad \text{pois} \quad J(q)B(q) = \begin{bmatrix} 0 & 0 \end{bmatrix}.$$

4.1 Descrição Expandida

Por meio da equação 4.7, pode-se concluir que a dinâmica do robô é dada por:

$$\begin{aligned} m\dot{u} &= (f_{atx})^B = f_1 + f_2, \\ \dot{x} &= \cos(\theta)u, \\ \dot{y} &= \text{sen}(\theta)u, \\ \mathcal{I}\dot{\omega} &= \tau = (f_1 - f_2)l, \\ \dot{\theta} &= \omega. \end{aligned}$$

Considerando ainda a dinâmica do conjunto motor/roda 1 (o mesmo para o conjunto motor/roda 2), tem-se ainda:

$$\begin{aligned} L_1\dot{I}_1 + R_1I_1 &= v_1 - k_1\dot{\theta}_1, \\ k_1I_1 - f_1r_1 &= J_1\ddot{\theta}_1, \end{aligned}$$

sendo L_1 a indutância da armadura, R_1 a resistência da armadura, I_1 a corrente da armadura, k_1 a constante de torque e da força contra eletromotriz (são numericamente iguais porém com unidades diferentes), k_1I_1 é torque induzido, r_1 é o raio da roda, f_1r_1 é o torque devido ao atrito, J_1 é o momento de inércia do conjunto eixo/roda e θ_1 é a posição angular da roda. Com isso a força de atrito f_1 pode ser escrita como:

$$f_1 = \frac{k_1/r_1}{R_1 + L_1s}v_1 - \frac{k_1^2/r_1}{R_1 + L_1s}\dot{\theta}_1 - \frac{J_1}{r_1}\ddot{\theta}_1,$$

e para $L_1/R_1 \approx 0$, tem-se:

$$f_1 = \frac{k_1}{r_1R_1}v_1 - \frac{k_1^2}{r_1R_1}\dot{\theta}_1 - \frac{J_1}{r_1}\ddot{\theta}_1.$$

Fazendo o mesmo para o conjunto motor/roda 2, a dinâmica do veículo é descrita como:

$$\begin{aligned} m\dot{u} &= \frac{k_1}{r_1R_1}v_1 - \frac{k_1^2}{r_1R_1}\dot{\theta}_1 - \frac{J_1}{r_1}\ddot{\theta}_1 + \frac{k_2}{r_2R_2}v_2 - \frac{k_2^2}{r_2R_2}\dot{\theta}_2 - \frac{J_2}{r_2}\ddot{\theta}_2, \\ \dot{x} &= \cos(\theta)u, \\ \dot{y} &= \text{sen}(\theta)u, \\ \frac{\mathcal{I}}{l}\dot{\omega} &= \frac{k_1}{r_1R_1}v_1 - \frac{k_1^2}{r_1R_1}\dot{\theta}_1 - \frac{J_1}{r_1}\ddot{\theta}_1 - \frac{k_2}{r_2R_2}v_2 + \frac{k_2^2}{r_2R_2}\dot{\theta}_2 + \frac{J_2}{r_2}\ddot{\theta}_2, \\ \dot{\theta} &= \omega. \end{aligned}$$

Considerando $k_1 = k_2 = k$, $r_1 = r_2 = r$, $J_1 = J_2 = J$ e $R_1 = R_2 = R$, tem-se que a

dinâmica do veículo pode ser dada por:

$$\begin{aligned}
m\dot{u} &= \frac{k}{rR}(v_1 + v_2) - \frac{k^2}{rR}(\dot{\theta}_1 + \dot{\theta}_2) - \frac{J}{r}(\ddot{\theta}_1 + \ddot{\theta}_2), \\
\dot{x} &= \cos(\theta)u, \\
\dot{y} &= \text{sen}(\theta)u, \\
\frac{\mathcal{J}}{l}\dot{\omega} &= \frac{k}{rR}(v_1 - v_2) - \frac{k^2}{rR}(\dot{\theta}_1 - \dot{\theta}_2) - \frac{J}{r}(\ddot{\theta}_1 - \ddot{\theta}_2), \\
\dot{\theta} &= \omega.
\end{aligned}$$

Seja p_1 (p_2) o vetor posição do ponto de contato da roda 1 (2). Então, $p_g = (p_1 + p_2)/2$, sendo p_g o vetor posição do C.G.. Logo, $\dot{p}_g = (\dot{p}_1 + \dot{p}_2)/2$. Mas, \dot{p}_1, \dot{p}_2 possuem a mesma direção, logo $u = (u_1 + u_2)/2$, sendo u_1 (u_2) componente da velocidade linear do ponto de contato da roda 1 (2) com a superfície na direção da roda e u a componente da velocidade linear do C.G. nesta mesma direção (eixo x do sistema do corpo). Além disso, $u_1 = \omega_1 r_1 = \dot{\theta}_1 r_1$ e $u_2 = \omega_2 r_2 = \dot{\theta}_2 r_2$. Com isso, tem-se:

$$\begin{aligned}
m\dot{u} &= \frac{k}{rR}(v_1 + v_2) - \frac{2k^2}{r^2R}u - \frac{J}{r}(\ddot{\theta}_1 + \ddot{\theta}_2), \\
\dot{x} &= \cos(\theta)u, \\
\dot{y} &= \text{sen}(\theta)u, \\
\frac{\mathcal{J}}{l}\dot{\omega} &= \frac{k}{rR}(v_1 - v_2) - \frac{k^2}{rR}(\dot{\theta}_1 - \dot{\theta}_2) - \frac{J}{r}(\ddot{\theta}_1 - \ddot{\theta}_2), \\
\dot{\theta} &= \omega.
\end{aligned}$$

Além disso, o vetor $p_1 - p_2$ possui módulo constante ($|p_1 - p_2| = 2l$) e $|\dot{p}_1 - \dot{p}_2| = |\omega||p_1 - p_2| = 2|\omega|l$. Mas, \dot{p}_1, \dot{p}_2 possuem a mesma direção, logo $\dot{p}_1 - \dot{p}_2 = u_1 - u_2 = \dot{\theta}_1 r_1 - \dot{\theta}_2 r_2 = (\dot{\theta}_1 - \dot{\theta}_2)r$. Com isso, $|(\dot{\theta}_1 - \dot{\theta}_2)r| = 2|\omega|l$. Para considerar sinal, pode-se escrever que: $(\dot{\theta}_1 - \dot{\theta}_2)r = 2\omega l$ e, conseqüentemente, tem-se o modelo completo considerando apenas simetria:

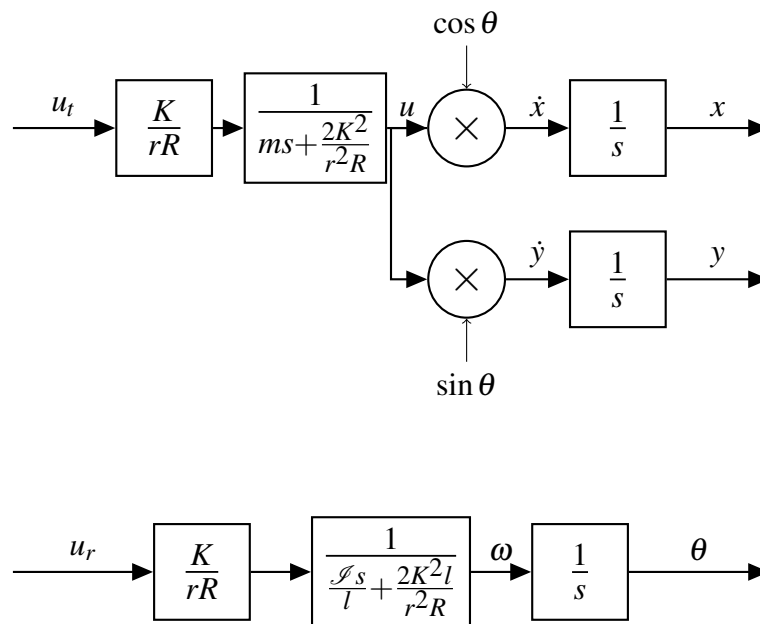
$$\begin{aligned}
m\dot{u} &= -\frac{2k^2}{r^2R}u + \frac{k}{rR}u_t - \frac{J}{r}(\ddot{\theta}_1 + \ddot{\theta}_2), \\
\dot{x} &= \cos(\theta)u, \\
\dot{y} &= \text{sen}(\theta)u, \\
\frac{\mathcal{J}}{l}\dot{\omega} &= -\frac{2k^2l}{r^2R}\omega + \frac{k}{rR}u_r - \frac{J}{r}(\ddot{\theta}_1 - \ddot{\theta}_2), \\
\dot{\theta} &= \omega.
\end{aligned}$$

sendo $u_t = v_1 + v_2$ e $u_r = v_1 - v_2$ os sinais de controle de translação e rotação, respectivamente. Para o caso em que $\ddot{\theta}_1, \ddot{\theta}_2 \approx 0$ e para J pequeno tem-se o modelo considerando

simetria e/ou aceleração pequena nas rodas ou inércia pequena das rodas:

$$\begin{aligned} m\dot{u} &= -\frac{2k^2}{r^2R}u + \frac{k}{rR}u_t, \\ \dot{x} &= \cos(\theta)u, \\ \dot{y} &= \text{sen}(\theta)u, \\ \frac{\mathcal{I}}{l}\dot{\omega} &= -\frac{2k^2l}{r^2R}\omega + \frac{k}{rR}u_r, \\ \dot{\theta} &= \omega. \end{aligned}$$

Definindo o sistema em diagrama de blocos:



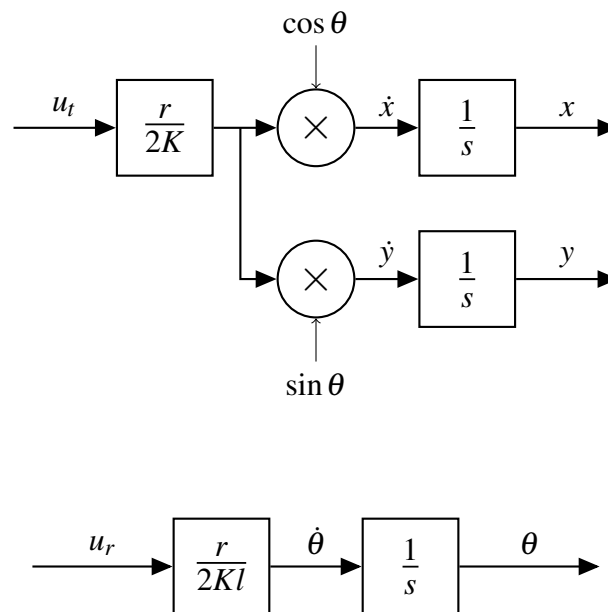
Para o caso em que a massa do robô e a inércia são desprezíveis ($m, \mathcal{I} \approx 0$) tem-se o modelo cinemático:

$$\begin{aligned} \frac{2k^2}{r^2R}u &= \frac{k}{rR}u_t, \\ \dot{x} &= \cos(\theta)\frac{r}{2k}u_t, \\ \dot{y} &= \text{sen}(\theta)\frac{r}{2k}u_t, \\ \frac{2k^2l}{r^2R}\omega &= \frac{k}{rR}u_r, \\ \dot{\theta} &= \frac{r}{2kl}u_r. \end{aligned}$$

ou ainda,

$$\begin{aligned}\dot{x} &= \cos(\theta) \frac{r}{2k} u_t, \\ \dot{y} &= \sin(\theta) \frac{r}{2k} u_t, \\ \dot{\theta} &= \frac{r}{2kl} u_r.\end{aligned}$$

Pode-se então definir em diagrama de blocos o sistema:



Capítulo 5

Controle

Este capítulo tem como intuito o desenvolvimento de uma estratégia de controle eficiente aplicada ao futebol de robôs pelo uso de técnicas clássicas em um robô holonômico a ser controlado, neste caso um robô unicycle. Propõe-se avaliar configurações diferentes de controladores, tanto em composições padrão (PID) quanto por meio da manipulação das variáveis de modelagem do robô através de equações para um controle de linearização por realimentação.

5.1 Objetivo de Controle: Movimentos Usuais no Âmbito de Futebol de Robôs

Os controles PID e de linearização por realimentação foram desenvolvidos para aplicação na equipe de futebol de robôs da UFRJ Hefestos. E para tal tarefa, alguns parâmetros deveriam ser respeitados:

- a) O controle deve dar ao robô a capacidade de movimentar-se tanto para frente quanto para trás da mesma forma.
- b) O controle deve receber as informações de posição da visão e com isso definir seus parâmetros para a movimentação.
- c) O controle deve ser capaz de definir a velocidade individual de cada roda bem como a direção de rotação da mesma, que posteriormente serão enviados ao robô.

Dessa maneira, tanto nas simulações do Capítulo 6 quanto nos resultados do Capítulo 7 foram estudadas trajetórias que fossem aplicáveis às situações de jogo. Essas situações são explicitadas à seguir.

Referência Degrau de Posição: Bola Parada

Durante saídas de bola, cobranças de falta e pênaltis a bola fica parada em determinada posição e o robô deve avançar em sua direção. Dessa maneira foi simulada uma entrada de grau para o robô que determina um ponto ao qual este deve avançar.

Referência Senoidal: Defesa

O goleiro e o zagueiro devem movimentar-se lateralmente para exercerem proteção tanto da faixa defensiva do campo quanto no gol. Dessa forma, uma entrada senoidal na posição vertical com posição horizontal fixa, simula a orientação da estratégia para essas situações.

Seguimento de Trajetórias Gerais: Atacante e Zagueiro

O atacante deve procurar direcionar a bola para o gol enquanto o zagueiro (além de fazer seu movimento lateral padrão) pode mover-se para marcar o atacante adversário se este estiver com a bola. Para essas situações de movimentação mais genérica, foram testadas as trajetórias como a curva de Lissajous e uma trajetória circular.

5.2 Controle Desacoplado de Posição (Translação e Rotação)

O controle de velocidade de translação é o responsável por fazer o robô se locomover para frente e para trás com os parâmetros previamente descritos.

Ao receber os dados processados de distância relativa ao próximo ponto, o controle se subdivide em duas partes: translação e rotação.

Isto posto, os parâmetros foram atendidos da seguinte forma:

- a) Uma variável relativa à direção foi aplicada ao fator proporcional do controle, informando assim se o robô deve movimentar-se para frente ($g = 1$), ou para trás ($g = -1$). Nota-se que ao fazer isso, o ganho proporcional automaticamente é invertido, bem como toda a movimentação do robô. Como o robô aqui descrito é simétrico, a movimentação deve manter-se análoga.
- b) Com as informações da visão, posição e orientação atual (obtida na câmera) e posição e orientação desejada, o controle executa as seguintes ações:
- c) Se a posição desejada de X e/ou de Y estiver numa distância maior que 26px, essa distância é saturada para que não ocorra sobrecarga no modelo.
- d) O módulo da distância ao próximo ponto (D) é definido pela raiz quadrada do quadrado das diferenças entre a posição desejada e atual:

$$\epsilon_D = \sqrt{(x_d - x)^2 + (y_d - y)^2} \quad (5.1)$$

onde $\begin{bmatrix} x_d \\ y_d \end{bmatrix}$ = vetor de posição desejada.

- e) A orientação desejada é definida e dada pela diferença angular entre a orientação atual do robô e a orientação desejada.

$$\theta_d = \text{atan} \left(\frac{y_d - y}{x_d - x} \right) \quad (5.2)$$

- f) O controle então calcula os ganhos dos controladores de translação e rotação, que serão mostrados mais à frente. Por meio de uma matriz de desacoplamento, são obtidas as velocidades individuais de cada motor.

Tanto o controle de rotação quanto o de translação são compostos por um fator proporcional e um derivativo como definidos abaixo:

- Proporcional:

$$\omega_p = Kp_\omega \times \Delta\theta \rightarrow \text{Rotação} \quad (5.3)$$

$$u_p = Kp_u \times D \rightarrow \text{Translação} \quad (5.4)$$

- Derivativo:

$$\omega_d = Kd_\omega \times \dot{\theta} \rightarrow \text{Rotação} \quad (5.5)$$

$$u_d = Kd_u \times \dot{D} \rightarrow \text{Translação} \quad (5.6)$$

onde Kd_ω , Kd_u , Kp_ω e Kp_u são os ganhos do controle.

5.2.1 Controle PD para Translação

Aqui nesta seção será abordada a parte de translação do controlador.

Na composição do controle de translação, são utilizados os fatores P, I e D.

- Para a componente proporcional do controlador, a seguinte lógica foi elaborada:

$$Kp_u = g \times KP_{Uig} \quad (5.7)$$

$$u_p = Kp_u \times \varepsilon_D \quad (5.8)$$

Onde KP_{Uig} é dado pelo controle da interface gráfica, g é a variável relativa ao movimento desejado do robô, u_p é a componente proporcional do controle de translação e ε_D é a distância ao próximo ponto desejado como já discutido anteriormente.

- A componente integral foi implementada da seguinte maneira:

$$KI_u = g \times KI_{Uig} \quad (5.9)$$

$$u_{i+1} = u_i + h \times KI_u \times \varepsilon_D \quad (5.10)$$

Onde KI_{Uig} é dado pelo controle da interface gráfica, u_i é a componente integral do controle de translação no momento t , u_{i+1} é a componente integral do controle de translação no momento $t + 1$, h é a componente relativa à taxa de frames por segundo que a câmera consegue gerar, g e ε_D são as mesmas da componente de translação anterior.

- E finalmente a componente derivativa foi construída da seguinte forma:

$$KD_u = g \times KD_{Uig} \quad (5.11)$$

$$u_d = KD_u \times deriv_{\varepsilon_D} \quad (5.12)$$

Onde é definida e calculada a derivada da distância $deriv_{\varepsilon_D}$ através da técnica chamada derivada suja, KD_{Uig} é dado pelo controle da interface gráfica, u_d é a componente derivativa do controle de translação e g é a mesma da componente de translação anterior.

- Uma vez obtidos estes componentes do controle linear, pode-se simplesmente somá-los para obter os parâmetros do controle linear u :

$$u = u_p + u_i + u_d \quad (5.13)$$

Nesta equação u_p , u_i e u_d são os componentes de u que é, finalmente, o sinal de controle de translação que será enviado ao robô.

Dessa maneira elaborou-se a primeira parte do controlador, porém, ainda é necessário o outro fator presente na modelagem descrita no capítulo anterior. A componente ω relativa ao controle de rotação.

5.2.2 Controle PD para Rotação

Uma vez que já estão calculados o ângulo desejado para o próximo ponto da trajetória e a diferença entre os ângulos atual e o desejado, pode-se elaborar o controle de rotação para esses parâmetros.

O controle de rotação elaborado, de forma análoga ao de translação, possui componentes P, I e D.

- A primeira componente do controlador a ser definida é a componente proporcional, que é obtida segundo:

$$KP_w = g \times KP_{Wig} \quad (5.14)$$

$$w_p = KP_w \times dif_\theta \quad (5.15)$$

Onde KP_{Wig} é dado pelo controle da interface gráfica, g é a variável relativa ao movimento desejado do robô, w_p é a componente proporcional do controle de rotação e dif_θ é a diferença entre o ângulo desejado e o atual.

- Após obter a componente proporcional, a componente integral é calculada da seguinte maneira:

$$KI_w = g \times KI_{Wig} \quad (5.16)$$

$$w_{i+1} = w_i + h \times KI_w \times dif_\theta \quad (5.17)$$

Onde KI_{Wig} é dado pelo controle da interface gráfica, w_i é a componente integral do controle de rotação no instante t , w_{i+1} é a componente integral do controle de rotação no instante $t + 1$, h é a componente relativa à taxa de frames por segundo que a câmera consegue gerar, g e dif_θ são as mesmas da componente de rotação anterior.

- A componente derivativa foi implementada da seguinte maneira:

$$KD_w = g \times KD_{Wig} \quad (5.18)$$

$$w_d = KD_w \times deriv_{dif_\theta} \quad (5.19)$$

Onde calculou-se a derivada da diferença entre o ângulo atual e o desejado $deriv_{dif_\theta}$ através da técnica chamada derivada suja. KD_{Wig} é dado pelo controle da interface gráfica, u_d é a componente integral do controle de rotação e g e D são as mesmas das componentes de rotação anteriores.

- Assim, compondo todas essas equações, foi obtido o controlador de rotação ω .

$$w = w_p + w_i + w_d \quad (5.20)$$

Nesta equação w_p , w_i e w_d são as componentes de w que é, finalmente, o sinal de controle de rotação que será enviado ao modelo.

Dessa maneira, são conhecidas as duas componentes necessárias para compor o sinal de controle que será enviado ao robô.

5.3 Controle de linearização por realimentação

O controle Desacoplado de Posição é a maneira clássica de controlar um robô móvel. Porém, analisando as equações que compõem a descrição da modelagem cinemática do robô uniclo, pode-se abordar o controle de posição do robô de maneira diferente.

Assim como na seção anterior, este controle atende aos parâmetros do projeto da seguinte maneira:

- a) Analogamente ao controle anterior, este possui uma variável relativa à direção que informa se o robô deve movimentar-se para frente ($g = 1$) ou para trás ($g = -1$) aproveitando a simetria do robô.
- b) O controle de linearização por realimentação recebe as informações de posição horizontal(x) e vertical(y) da câmera e as posições desejadas do simulador de trajetória.
- c) Com as informações, o controle aproveita-se das equações cinemáticas do modelo para definir os sinais de controle a serem enviados para que o ponto objetivo seja alcançado.

5.3.1 Equações do controle de linearização por realimentação

Modelo Cinemático

Como vimos no Capítulo 4, a descrição cinemática do modelo é dada pelas equações:

$$\dot{x} = \cos(\theta) \frac{r}{2k} u_t, \quad (5.21)$$

$$\dot{y} = \sin(\theta) \frac{r}{2k} u_t, \quad (5.22)$$

$$\dot{\theta} = \frac{r}{2kl} u_r. \quad (5.23)$$

O que permite controlar a variável u_t apenas com o erro da posição horizontal(x):

1. Controle do erro em x:

$$\dot{x} = \frac{r}{2K} \cos \theta \times u_t \quad (5.24)$$

$$u_t = \frac{2K}{r \cos \theta} \times u_x \quad (5.25)$$

$$\dot{x} = u_x \quad (5.26)$$

$$u_x = K_p e_x + \dot{x}_d \quad (5.27)$$

$$\dot{x} = g \times K_p e_x + \dot{x}_d \quad (5.28)$$

$$u_t = \frac{2K}{r \cos \theta} \times (K_p e_x + \dot{x}_d) \quad (5.29)$$

Onde e_x é a diferença entre a posição horizontal atual do robô e a desejada, g é a variável relativa à direção da frente do robô em relação ao ponto desejado e K_p é o ganho proporcional definido através do software.

Para o controle do segundo sinal de controle u_r , primeiro será controlado o seno do ângulo desejado através do erro de posição vertical(y) como demonstram as equações à seguir.

2. Controle do erro de y:

$$\dot{y} = \frac{r}{2K} \sin \theta u_t \quad (5.30)$$

$$\sin \theta = u_y = \frac{2K}{r u_t} (g \times K_p e_y + \dot{y}_d) \quad (5.31)$$

$$\theta_d = \arcsin\left(\frac{2K}{r u_t} (g \times K_p e_y + \dot{y}_d)\right) \quad (5.32)$$

Onde e_y é a diferença entre a posição vertical atual do robô e a desejada, g é a variável relativa à direção da frente do robô em relação ao ponto desejado e K_p é o ganho proporcional definido através do software.

Assim, possuindo o θ desejado, pode-se obter o sinal u_r através do erro de θ como mostrado nas equações:

3. Controle do erro de θ :

$$\dot{\theta} = \frac{r}{2Kl} \times u_r \quad (5.33)$$

$$u_r = (\dot{\theta}_d - \theta) \times K_p \quad (5.34)$$

$$u_r = \frac{2Kl}{r} \dot{\theta}_d + g \times K_p e_\theta \quad (5.35)$$

Onde e_θ é a diferença entre o ângulo atual do robô e o desejado, g é a variável relativa à direção da frente do robô em relação ao ponto desejado e K_p é o ganho proporcional definido através do software.

Dessa forma são obtidos os sinais de controle u_t e u_r que serão enviados ao robô. O que resta agora é provar, por meio de simulação, que a aproximação do modelo dinâmico para o modelo cinemático é suficientemente boa para que se tenha certeza sobre a aplicabilidade desta proposta para o controle. Assim será feito no próximo capítulo.

Capítulo 6

Simulações

Este capítulo abordará os simuladores desenvolvidos para o controlador do tipo PD e o controle de linearização por realimentação apresentados no Capítulo 5 baseado nas observações da equações do Capítulo 4 e os resultados obtidos.

6.1 Simulador para o controle PD

Para este controlador, foi desenvolvido o simulador apresentado à seguir.

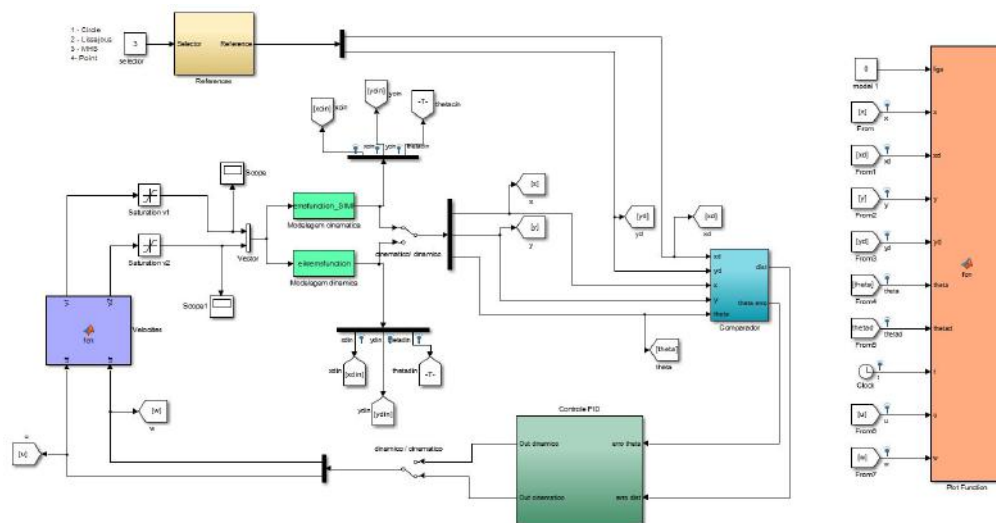


Figura 6.1: Simulink simulador PD

Neste simulador é possível selecionar a trajetória desejada através do seletor de referência apresentado na figura 6.2. As referências podem ser escolhidas entre: Ponto de referência para o qual o robô deve se direcionar, referência circular, curva de Lissajous e variação senoidal da referência vertical com referência horizontal fixa.

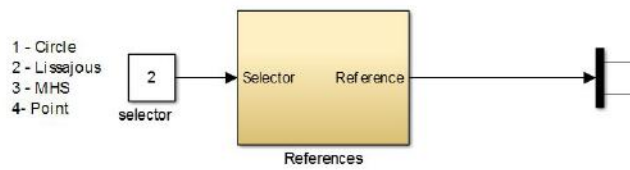


Figura 6.2: *Seletor de referência do simulador PD*

Os dados da referência definida e os dados de posição atuais do robô são passados ao comparador apresentado na figura A.13 que define a distância entre o ponto atual do robô e o ponto desejado bem como a diferença do ângulo atual do robô para o ângulo desejado.

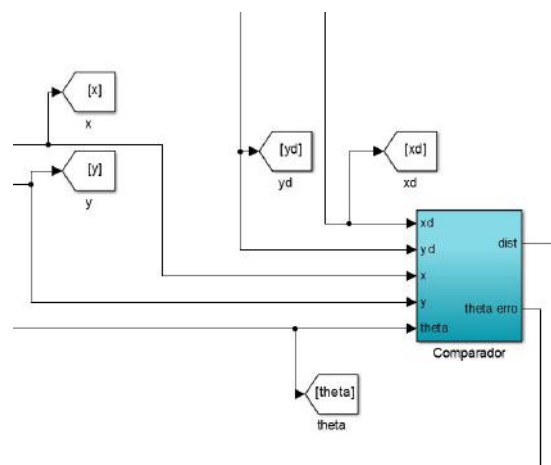


Figura 6.3: *Comparador de posição do simulador PD*

Com a distância e o erro do ângulo definidos, o controlador PD apresentado na figura A.14, utiliza os ganhos previamente definidos para estabelecer o sinal de controle que será enviado à planta.

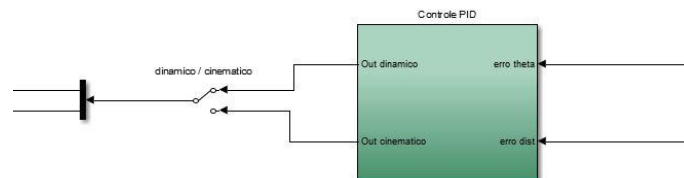


Figura 6.4: *Controlador PD do simulador PD*

Por fim, é simulado o microcontrolador na figura 6.5 que traduz e satura o sinal de controle para os motores do robô que se movimenta.

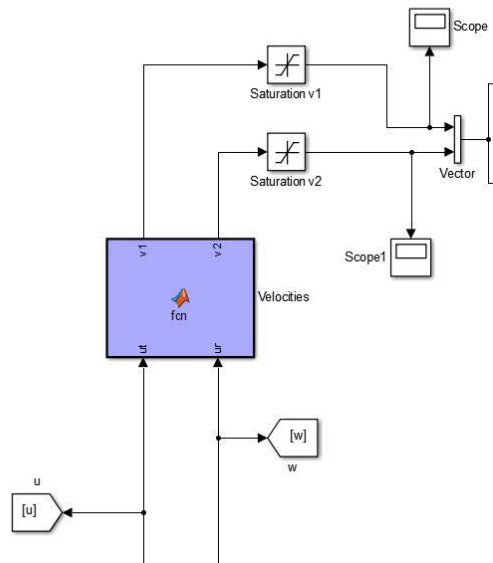


Figura 6.5: *Conversor de sinal do simulador PD*

É possível ainda seleccionar entre os modelos cinemático ou dinâmico do robô para realizar a simulação, como mostrado na figura 6.6

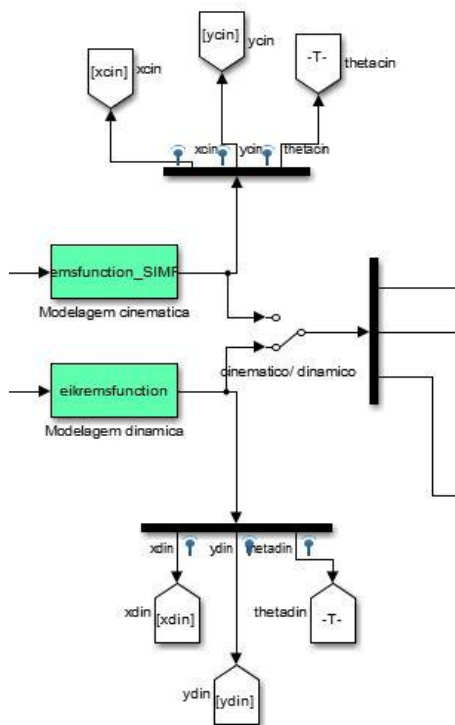


Figura 6.6: *Modelos cinemático e dinâmico do simulador PD*

Foi implementada no simulador uma função para guardar os dados da simulação e posteriormente exibí-los em uma figura 6.7, tanto em tempo real durante a simulação

quando após a mesma.

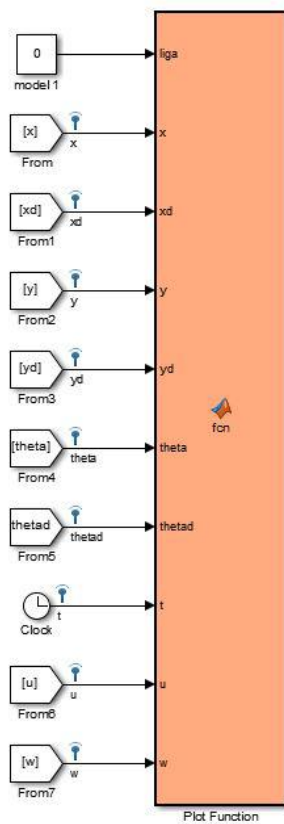


Figura 6.7: Função de exibição de dados do simulador PD

6.2 Simulador para controle de linearização por realimentação

Para o controle de linearização por realimentação no capítulo anterior, foi desenvolvido o simulador apresentado à seguir.

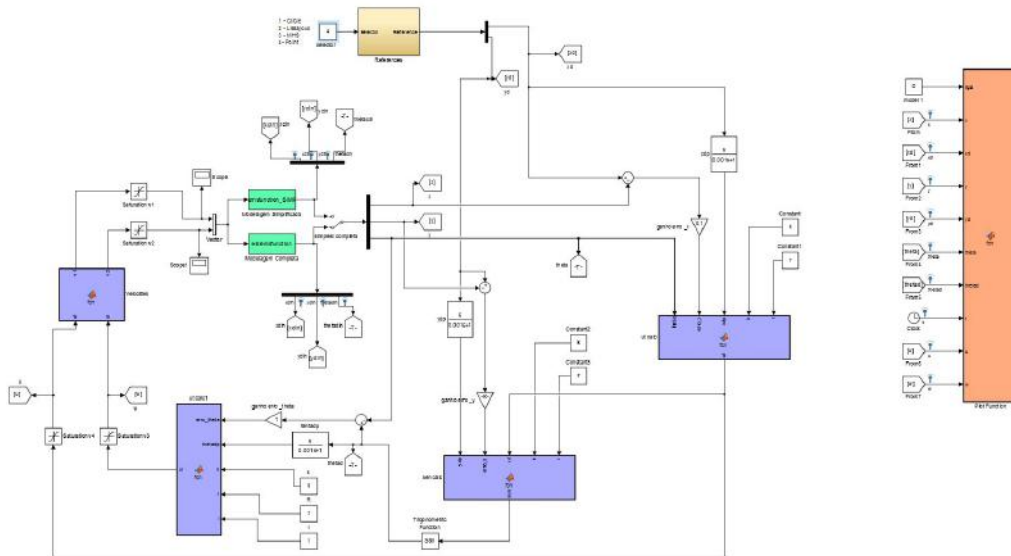


Figura 6.8: Simulink simulador para controle de linearização por realimentação

Neste simulador foram mantidas as funções de seleção de trajetória, simulador do microcontrolador, exibição dos dados e seleção de modelo do simulador anterior. A grande diferença entre este simulador e o anterior fica por parte dos comparadores. Ao invés de apenas um comparador para distância e ângulo, são utilizados três comparadores. À começar pelo comparador da posição horizontal x na figura 6.9.

O comparador da posição horizontal recebe a diferença entre as posições atual e desejada do robô com relação à trajetória definida, a derivada da posição desejada, o ângulo atual do robô e as constantes relativas à construção do robô como apresentado no Capítulo 5. Dessa maneira, um dos sinais de controle é obtido.

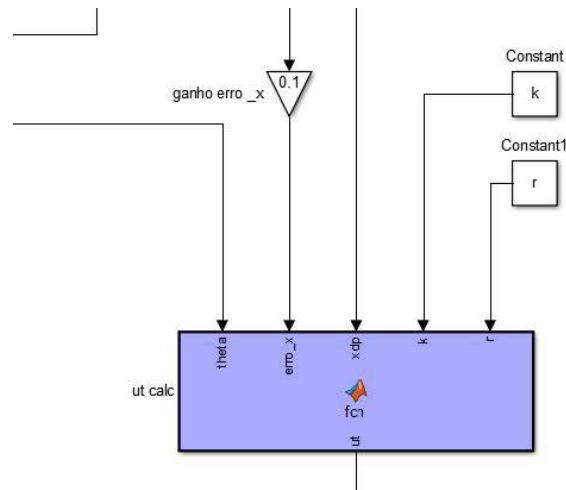


Figura 6.9: Comparador de posição horizontal do simulador para controle de linearização por realimentação

O comparador de posição vertical recebe dados análogos aos do controlador horizontal com exceção do ângulo do robô, que é substituído pelo sinal da saída do comparador anterior como mostra a figura 6.10. Assim obtém-se o seno do ângulo desejado para o robô.

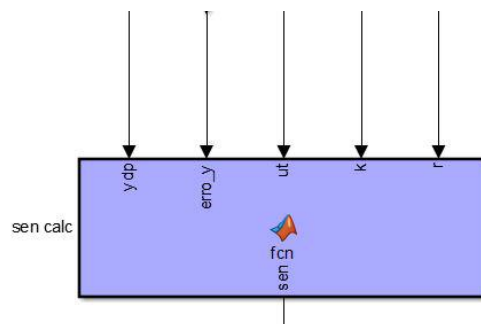


Figura 6.10: Comparador de posição vertical do simulador para controle de linearização por realimentação

Por fim, para compor o segundo sinal de controle, utiliza-se o comparador do ângulo do robô na figura 6.11. Este comparador recebe a diferença entre o ângulo atual do robô e o arco cujo seno foi definido pelo comparador de posição vertical. Este arco também é derivado e passado ao controlador, que também recebe parâmetros da construção do robô como definido nas equações do Capítulo 5.

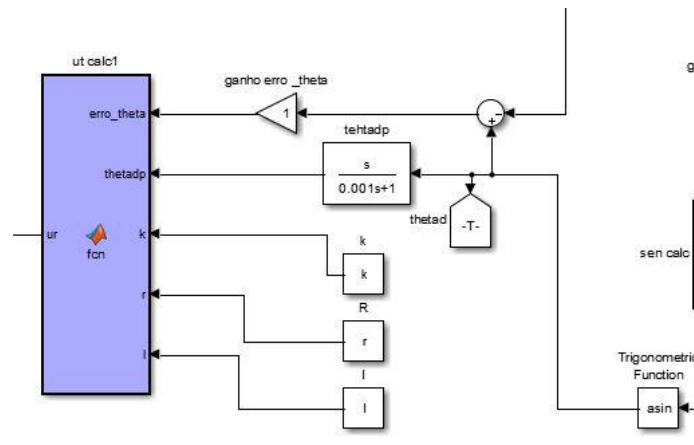


Figura 6.11: Comparador angular do simulador para controle de linearização por realimentação

Dessa maneira, são obtidos os dois sinais que são passados ao simulador microcontrolador e utilizados no controle do sistema.

6.3 Movimentos com Bola Parada: Pênalti, Cobrança de Faltas e Início de Jogo

Nesta seção serão mostrados os resultados obtidos na simulação de uma trajetória retilínea emulando uma cobrança de pênalti, que é um dos movimentos importantes com bola parada e pode ser expandido para cobranças de falta em diferentes posições do campo e início de jogo no centro do campo, utilizando o controle PD e o controle de linearização por realimentação.

Controle PD

Abaixo estão os resultados utilizando o controle PD para os modelos cinemático e dinâmico.

1. Modelo Cinemático

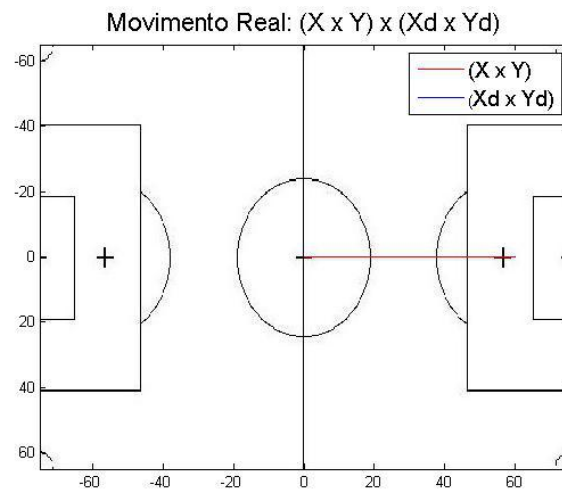


Figura 6.12: *Movimento Real da simulação de cobrança de pênalti utilizando o controle PD para o modelo cinemático.*

A Figura 6.12 mostra que a simulação de uma cobrança de pênalti utilizando o controle PD para o modelo cinemático apresenta valores iguais aos do movimento desejado.

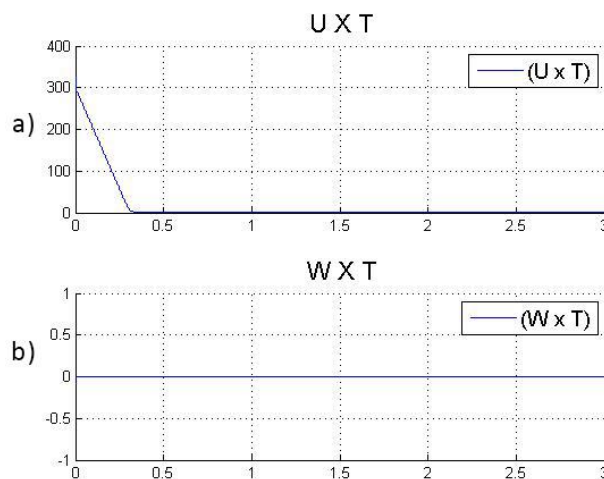


Figura 6.13: *Sinais de Controle da simulação de cobrança de pênalti utilizando o controle PD para o modelo cinemático.*

A Figura 6.13a, mostra o transitório que varia de 300 à 0 em aproximadamente 0,3 segundos, onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado.

A Figura 6.13b, não mostra nenhuma variação no sinal de controle angular visto que o ponto desejado apresenta mesma orientação do ponto inicial.

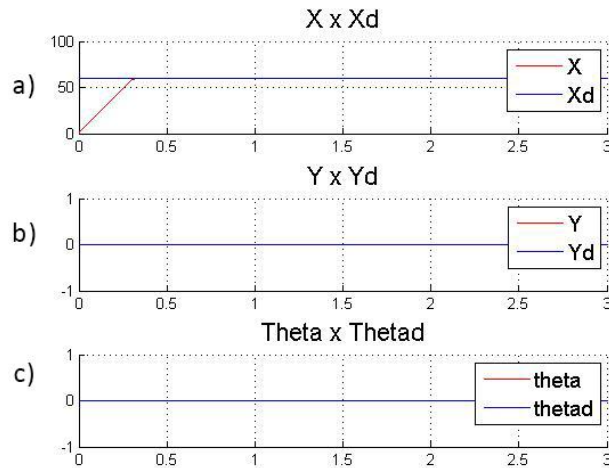


Figura 6.14: Posições linear e angular da simulação de cobrança de pênalti utilizando o controle PD para o modelo cinemático.

A Figura 6.14a, mostra um transitório que varia da posição inicial 0 à posição desejada 60 em aproximadamente 0.3 segundos na posição linear x.

A Figura 6.14b, não mostra nenhuma variação na posição linear y, visto que não há variação em y entre os pontos inicial e desejado.

A Figura 6.14c, não mostra nenhuma variação na posição angular theta, visto que o ponto desejado apresenta mesma orientação do ponto inicial.

2. Modelo Dinâmico

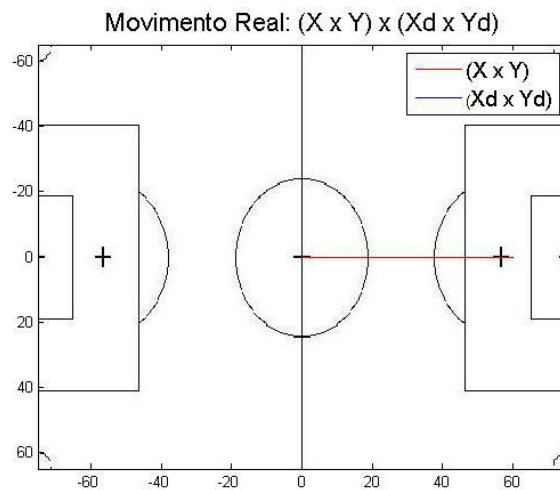


Figura 6.15: Movimento Real da simulação de cobrança de pênalti utilizando o controle PD para o modelo dinâmico.

A Figura 6.15 mostra que a simulação de uma cobrança de pênalti utilizando o controle PD para o modelo dinâmico apresenta valores iguais aos do movimento

desejado.

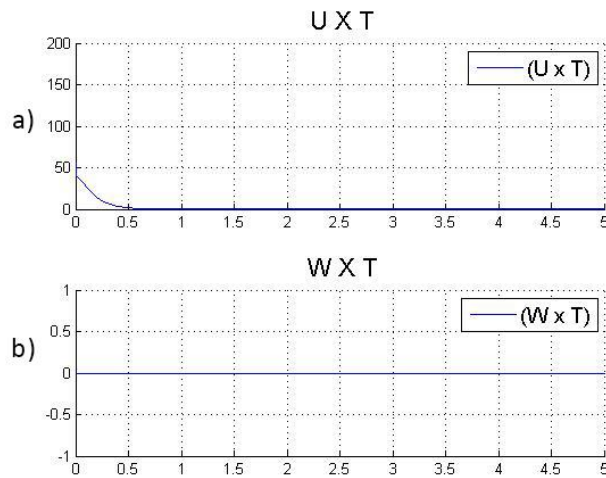


Figura 6.16: Sinais de Controle da simulação de cobrança de pênalti utilizando o controle PD para o modelo dinâmico.

A Figura 6.16a, mostra o transitório que varia de 40 à 0 em aproximadamente 0.5 segundos, onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado.

A Figura 6.16b, não mostra nenhuma variação no sinal de controle angular visto que o ponto desejado apresenta mesma orientação do ponto inicial.

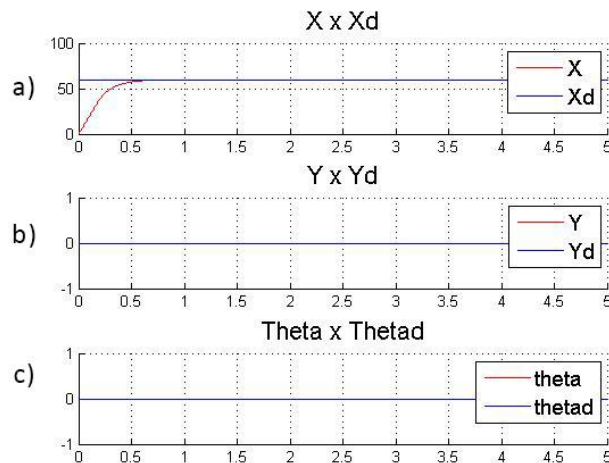


Figura 6.17: Posições linear e angular da simulação de cobrança de pênalti utilizando o controle PD para o modelo dinâmico.

A Figura 6.17a, mostra um transitório que varia da posição inicial 0 à posição desejada 60 em aproximadamente 0.6 segundos na posição linear x.

A Figura 6.17b, não mostra nenhuma variação na posição linear y , visto que não há variação em y entre os pontos inicial e desejado.

A Figura 6.17c, não mostra nenhuma variação na posição angular θ , visto que o ponto desejado apresenta mesma orientação do ponto inicial.

- Comparação entre os modelos Cinemático e Dinâmico

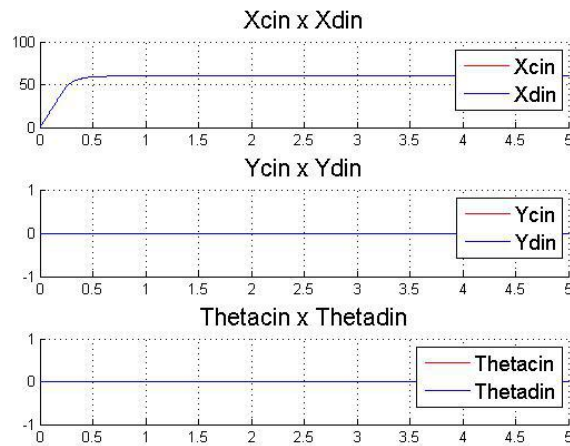


Figura 6.18: *Comparação entre os modelos cinemático e dinâmico para a cobrança de pênalti utilizando o controle PD*

A Figura 6.18 compara os dois modelos ao controlar o modelo cinemático em malha fechada e o modelo dinâmico em malha aberta e mostra que não houve variação entre as posições linear e angular dos mesmos, ou seja, apenas controlando através das equações do modelo cinemático, o modelo dinâmico se comportou de maneira satisfatória. Comprovando assim o que já era esperado por conta das velocidades atingidas serem baixas.

Controle de linearização por realimentação

Abaixo estão os resultados utilizando o controle de linearização por realimentação para o modelo cinemático.

1. Modelo Cinemático

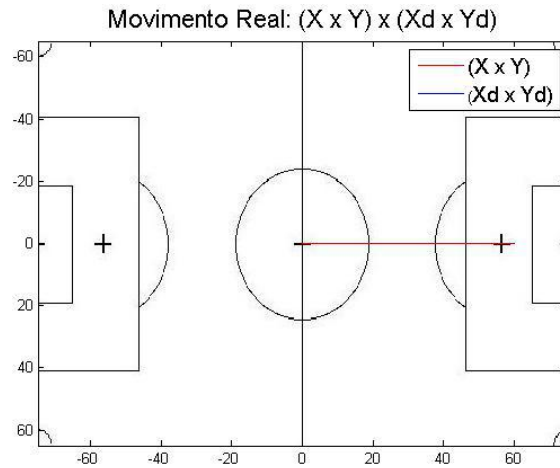


Figura 6.19: *Movimento Real da simulação de cobrança de pênalti utilizando o controle de linearização por realimentação para o modelo cinemático.*

A Figura 6.19 mostra que a simulação de uma cobrança de pênalti utilizando o controle de linearização por realimentação para o modelo cinemático apresenta valores iguais aos do movimento desejado.

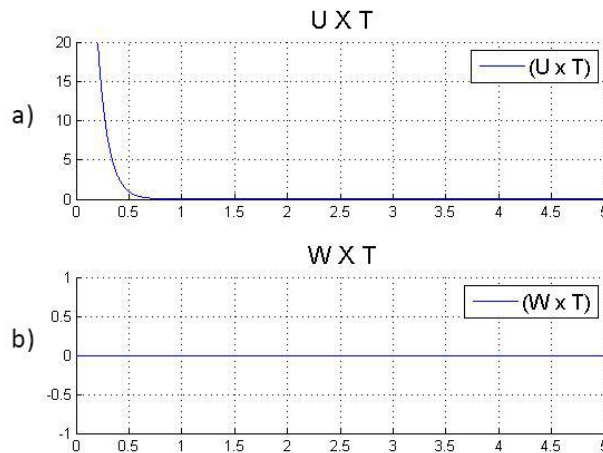


Figura 6.20: *Sinais de Controle da simulação de cobrança de pênalti utilizando o controle de linearização por realimentação para o modelo cinemático.*

A Figura 6.20a, mostra o transitório que varia de 20 à 0 em aproximadamente 0.6 segundos, onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado.

A Figura 6.20b, não mostra nenhuma variação no sinal de controle angular visto que o ponto desejado apresenta mesma orientação do ponto inicial.

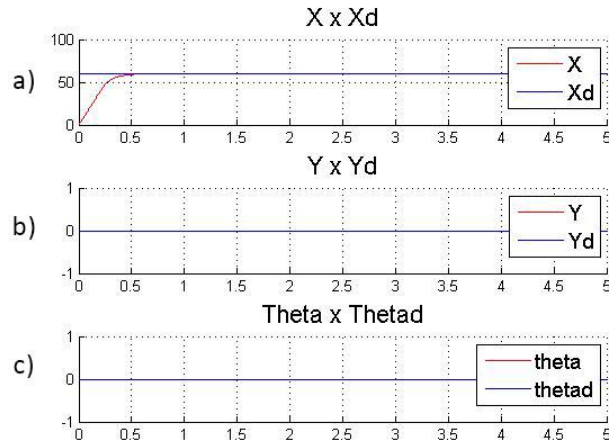


Figura 6.21: Posições linear e angular da simulação de cobrança de pênalti utilizando o controle de linearização por realimentação para o modelo cinemático.

A Figura 6.21a, mostra um transitório que varia da posição inicial 0 à posição desejada 60 em aproximadamente 0.5 segundos na posição linear x.

A Figura 6.21b, não mostra nenhuma variação na posição linear y, visto que não há variação em y entre os pontos inicial e desejado.

A Figura 6.21c, não mostra nenhuma variação na posição angular theta, visto que o ponto desejado apresenta mesma orientação do ponto inicial.

Conclusão dos Movimentos com Bola Parada

Como pode-se observar nas figuras 6.14, 6.17 e 6.21, o objetivo de atingir o ponto desejado foi alcançado levando aproximadamente o mesmo tempo para ambos simuladores. É importante notar também que, conforme comentado na figura 6.18, a aproximação que foi feita da modelagem dinâmica para a cinemática é suficiente para o controle.

6.4 Movimentos de Defesa

Nesta seção serão mostrados os resultados obtidos na simulação para o principal movimento de defesa, que consiste em uma trajetória retilínea dentro da área do gol, utilizando o controle PD e o controle de linearização por realimentação.

Controle PD

Abaixo estão os resultados utilizando o controle PD para os modelos cinemático e dinâmico.

1. Modelo Cinemático

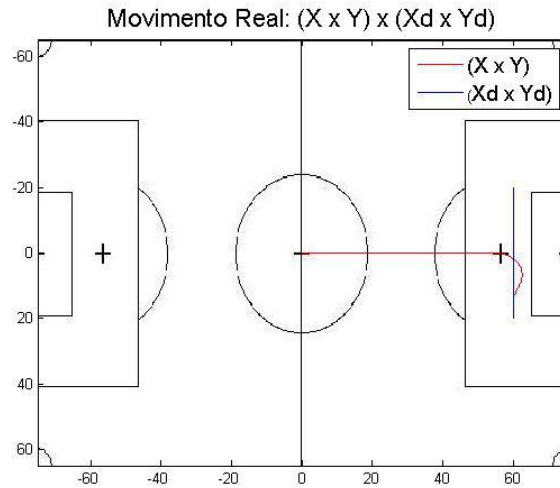


Figura 6.22: *Movimento Real da simulação de movimento de defesa utilizando o controle PD para o modelo cinemático.*

A Figura 6.22 mostra que a simulação de um movimento de defesa utilizando o controle PD para o modelo cinemático apresenta valores aproximados aos do movimento desejado.

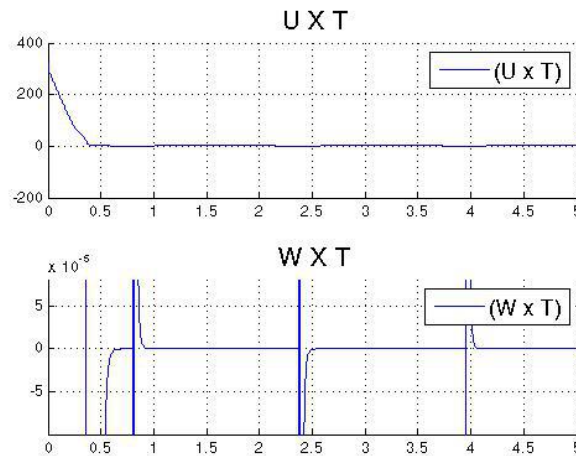


Figura 6.23: *Sinais de Controle da simulação de movimento de defesa utilizando o controle PD para o modelo cinemático.*

A Figura 6.23a, mostra o transitório que varia de 300 à 0 em aproximadamente 0.4 segundos, onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado.

A Figura 6.23b, mostra alguns picos no sinal de controle angular devido às singularidades do arctan como mostra a equação 5.2, que são tão rápidos que não interferem na performance do robô.

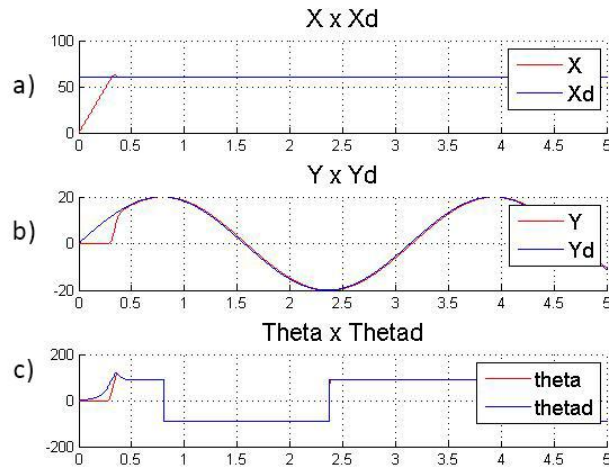


Figura 6.24: Posições linear e angular da simulação de movimento de defesa utilizando o controle PD para o modelo cinemático.

A Figura 6.24a, mostra um transitório que varia da posição inicial 0 à posição desejada 60 em aproximadamente 0.3 segundos na posição linear x.

A Figura 6.24b, mostra um transitório de valor 15 em aproximadamente 0.4 segundos na posição linear y, que depois se mantém igual a posição desejada durante o resto da trajetória.

A Figura 6.24c, mostra uma pequena diferença inicial na orientação do robô durante os primeiros 0.4 segundos, que depois se mantém igual à orientação desejada durante o resto da trajetória.

2. Modelo Dinâmico

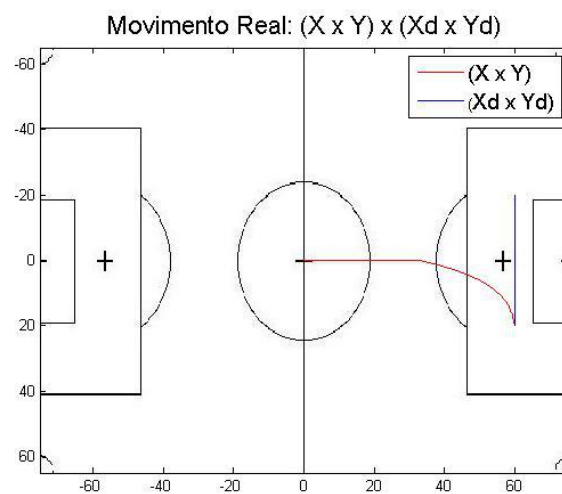


Figura 6.25: Movimento Real da simulação de movimento de defesa utilizando o controle PD para o modelo dinâmico.

A Figura 6.25 mostra a simulação de um movimento de defesa utilizando o controle PD para o modelo dinâmico, onde o mesmo segue a trajetória desejada.

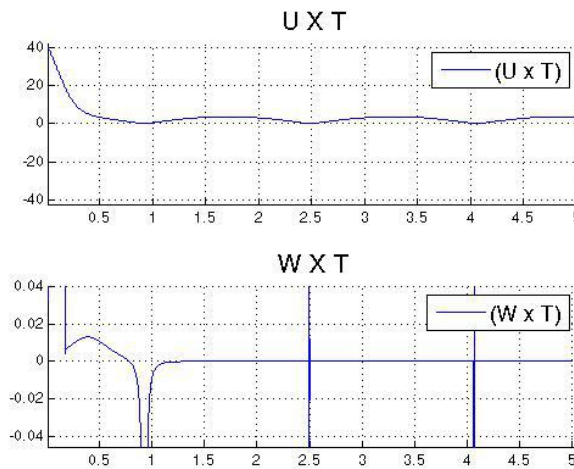


Figura 6.26: *Sinais de Controle da simulação de movimento de defesa utilizando o controle PD para o modelo dinâmico.*

A Figura 6.26a, mostra que o sinal de controle linear varia de 50 à 0 em aproximadamente 0.4 segundos e depois se mantém constante próximo à 0.

A Figura 6.26b, mostra o sinal de controle angular constante em zero, com 2 picos nos tempos 2.5 e 4.1 segundos devido à singularidade do arctan como mostra a equação 5.2, que são tão rápidos que não interferem na performance do robô.

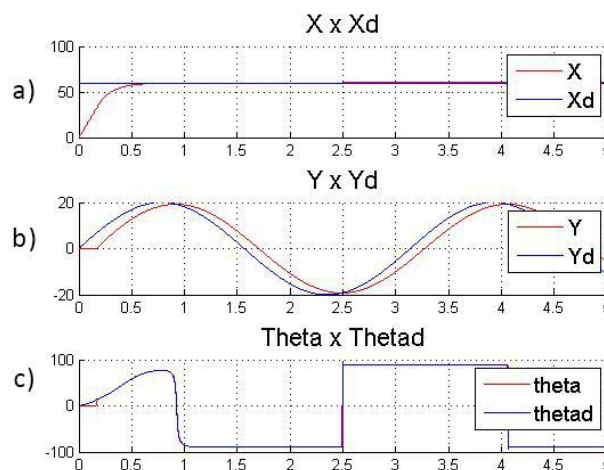


Figura 6.27: *Posições linear e angular da simulação de movimento de defesa utilizando o controle PD para o modelo dinâmico.*

A Figura 6.27a, mostra um transitório que varia da posição inicial 0 à posição desejada 60 em aproximadamente 0.5 segundos na posição linear x.

A Figura 6.27b, mostra uma diferença e atraso muito pequenos da posição linear y.

A Figura 6.27c, mostra a orientação do robô igual à orientação desejada.

- Comparação entre os modelos Cinemático e Dinâmico

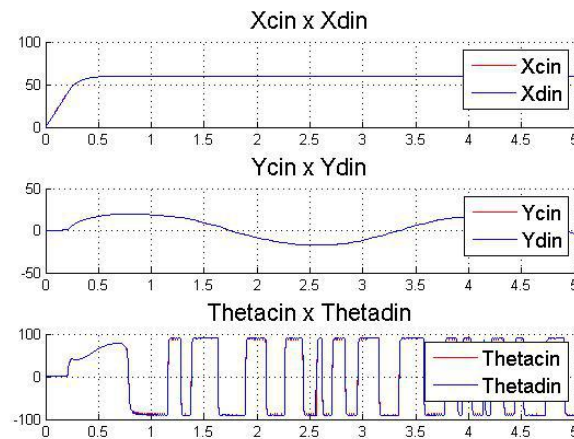


Figura 6.28: Comparação entre os modelos cinemático e dinâmico para o movimento de defesa utilizando o controle PD.

A Figura 6.28 compara novamente os dois modelos ao controlar o modelo cinemático em malha fechada e o modelo dinâmico em malha aberta e mostra que não houve variação entre as posições linear e angular dos mesmos, ou seja, apenas controlando através das equações do modelo cinemático, o modelo dinâmico se comportou de maneira satisfatória. Comprovando assim o que já era esperado por conta das velocidades atingidas serem baixas.

Controle de linearização por realimentação

Abaixo estão os resultados utilizando o controle de linearização por realimentação para o modelo cinemático.

1. Modelo Cinemático

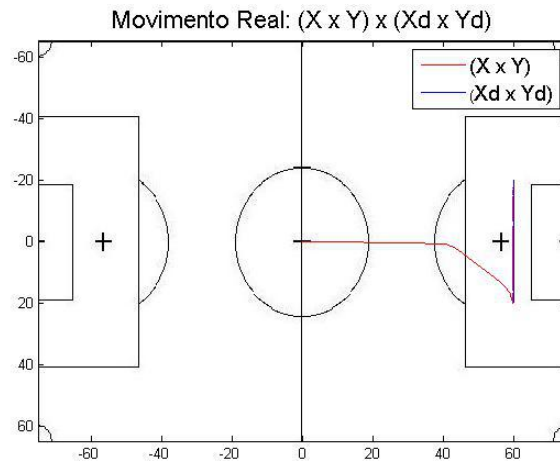


Figura 6.29: *Movimento Real da simulação de movimento de defesa utilizando o controle de linearização por realimentação para o modelo cinemático.*

A Figura 6.29 mostra a simulação de um movimento de defesa utilizando o controle de linearização por realimentação para o modelo cinemático, onde o mesmo segue a trajetória desejada.

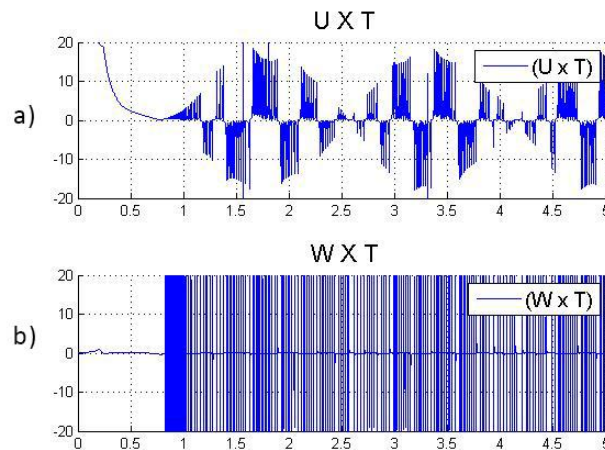


Figura 6.30: *Sinais de Controle da simulação de movimento de defesa utilizando o controle de linearização por realimentação para o modelo cinemático.*

A Figura 6.30a, mostra que o sinal de controle linear varia entre valores positivos e negativos devido à ida e à volta do movimento retilíneo e valores máximos de 20 e -20.

A Figura 6.30b, mostra o sinal de controle angular varia entre 20 e -20 durante toda a trajetória.

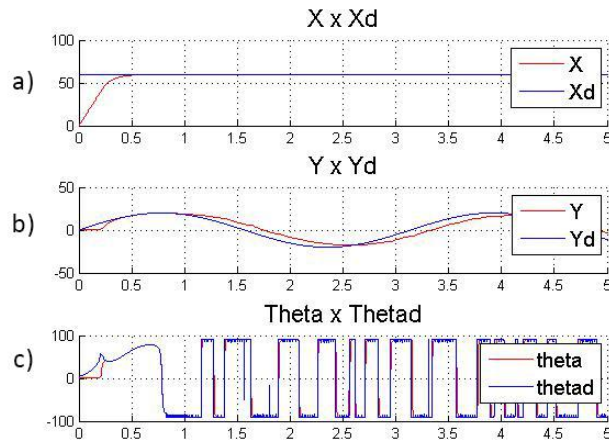


Figura 6.31: *Posições linear e angular da simulação de movimento de defesa utilizando o controle de linearização por realimentação para o modelo cinemático.*

A Figura 6.31a, mostra um transitório que varia da posição inicial 0 à posição desejada 60 em aproximadamente 0.5 segundos na posição linear x.

A Figura 6.31b, mostra uma diferença e atraso muito pequenos da posição linear y.

A Figura 6.31c, mostra a orientação do robô igual à orientação desejada.

Conclusão dos Movimentos de Defesa

As figuras 6.22, 6.25 e 6.29, mostram que o movimento realizado pelo robô na simulação foi bem próximo do desejado. Porém, é importante notar que na figura 6.30, os sinais do controle de linearização por realimentação sofrem muita interferência devido às singularidades de suas equações. Essas singularidades não afetam a movimentação do robô pois encontram-se saturadas antes de serem enviadas ao modelo. Os dois controladores desempenharam de maneira suficiente seu papel para essa trajetória.

6.5 Movimentos de Zagueiro e Atacante

Nesta seção serão mostrados os resultados obtidos na simulação para as trajetórias circular e de Lissajous que emulam uma possível estratégia de ataque e defesa, utilizando o controle PD e o controle de linearização por realimentação.

Círculo

Controle PD

Abaixo estão os resultados utilizando o controle PD para os modelos cinemático e dinâmico.

1. Modelo Cinemático

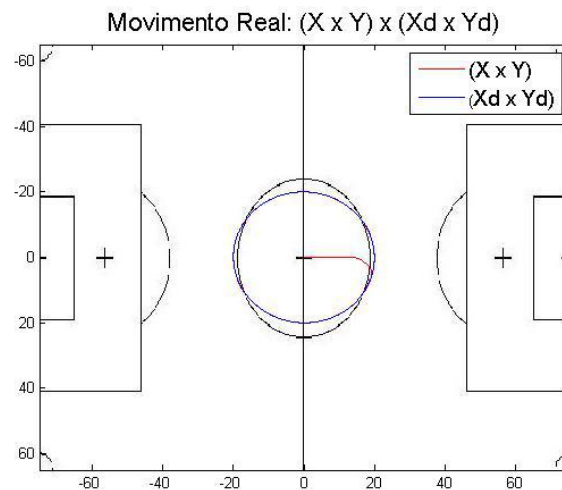


Figura 6.32: *Movimento Real da simulação de trajetória circular utilizando o controle PD para o modelo cinemático.*

A Figura 6.32 mostra a simulação de uma trajetória circular utilizando o controle PD para o modelo cinemático.

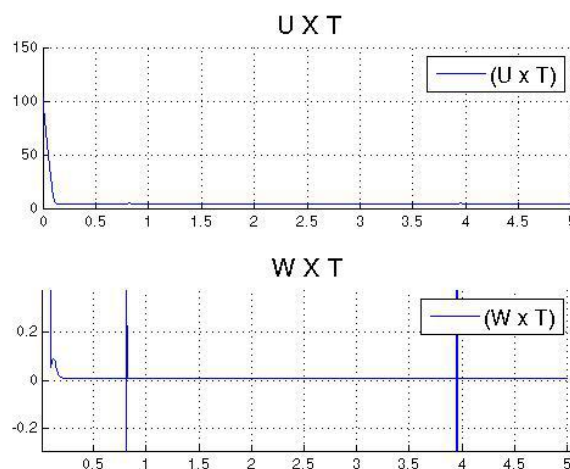


Figura 6.33: *Sinais de Controle da simulação de trajetória circular utilizando o controle PD para o modelo cinemático.*

A Figura 6.33a, mostra que o sinal de controle linear varia de 100 à ≈ 0 em aproximadamente 0.15 segundos e depois se mantém constante próximo à 0.

A Figura 6.33b, mostra o sinal de controle angular constante em zero, com 2 picos nos tempos 0.7 e 3.9 segundos devido à singularidade do arctan como mostra a equação 5.2, que são tão rápidos que não interferem na performance do robô.

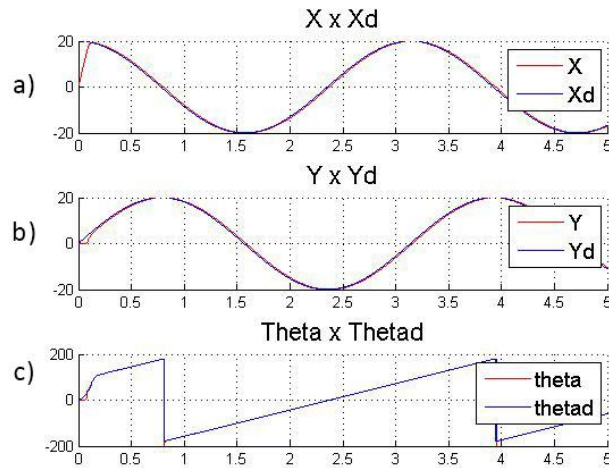


Figura 6.34: Posições linear e angular da simulação de trajetória circular utilizando o controle PD para o modelo cinemático.

A Figura 6.34a, mostra um transitório que varia da posição inicial 0 à posição desejada 20 em aproximadamente 0.1 segundos na posição linear x.

A Figura 6.34b, mostra que a posição linear em y se manteve igual à posição desejada.

A Figura 6.34c, mostra a orientação do robô igual à orientação desejada.

2. Modelo Dinâmico

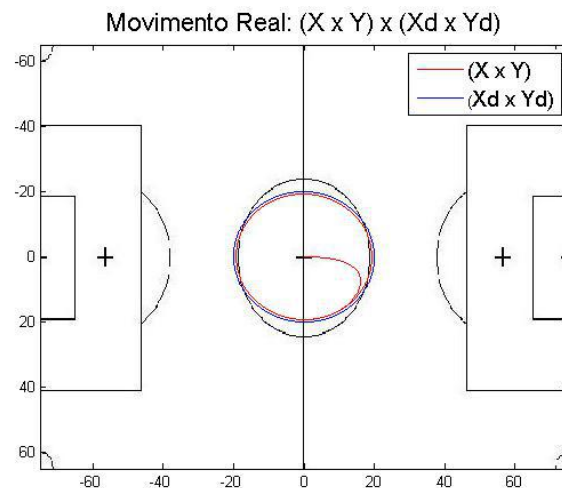


Figura 6.35: Movimento Real da simulação de trajetória circular utilizando o controle PD para o modelo dinâmico.

A Figura 6.35 mostra a simulação de uma trajetória circular utilizando o controle PD para o modelo dinâmico.

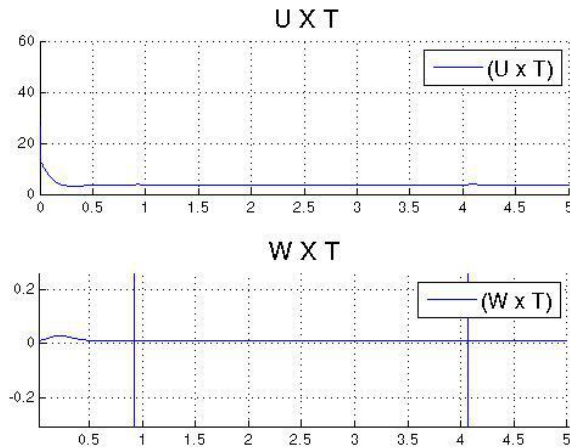


Figura 6.36: Sinais de Controle da simulação de trajetória circular utilizando o controle PD para o modelo dinâmico.

A Figura 6.36a, mostra que o sinal de controle linear varia de 10 à 5 em aproximadamente 0.2 segundos e depois se mantém constante.

A Figura 6.36b, mostra o sinal de controle angular constante em zero, com 2 picos nos tempos 1.9 e 4.1 segundos devido à singularidade do arctan como mostra a equação 5.2, que são tão rápidos que não interferem na performance do robô.

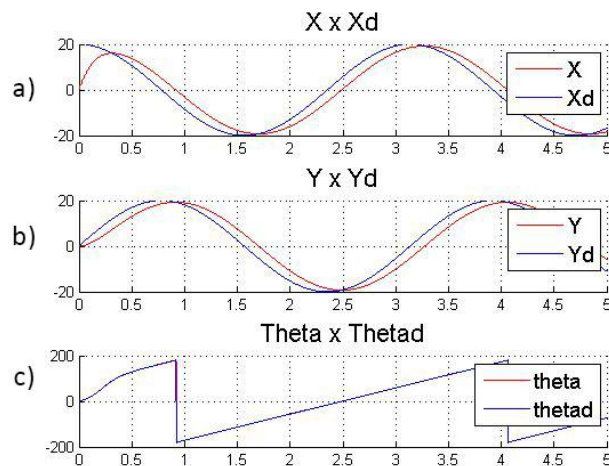


Figura 6.37: Posições linear e angular da simulação de trajetória circular utilizando o controle PD para o modelo dinâmico.

A Figura 6.37a, mostra um transitório que varia da posição inicial 0 à posição desejada 60 em aproximadamente 0.5 segundos na posição linear x.

A Figura 6.37b, mostra uma diferença e atraso muito pequenos da posição linear y.

A Figura 6.37c, mostra a orientação do robô igual à orientação desejada.

- Comparação entre os modelos Cinemático e Dinâmico

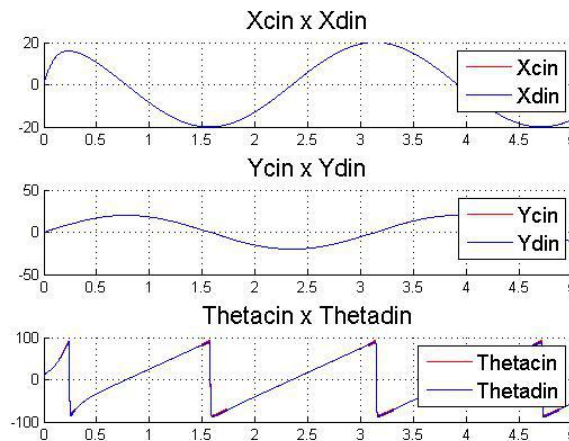


Figura 6.38: Comparação entre os modelos cinemático e dinâmico para trajetória circular utilizando o controle PD.

A Figura 6.38 compara novamente os dois modelos ao controlar o modelo cinemático em malha fechada e o modelo dinâmico em malha aberta e mostra que não houve variação entre as posições linear e angular dos mesmos, ou seja, apenas controlando através das equações do modelo cinemático, o modelo dinâmico se comportou de maneira satisfatória. Comprovando assim o que já era esperado por conta das velocidades atingidas serem baixas.

Controle de linearização por realimentação

Abaixo estão os resultados utilizando o controle de linearização por realimentação para o modelo cinemático.

1. Modelo Cinemático

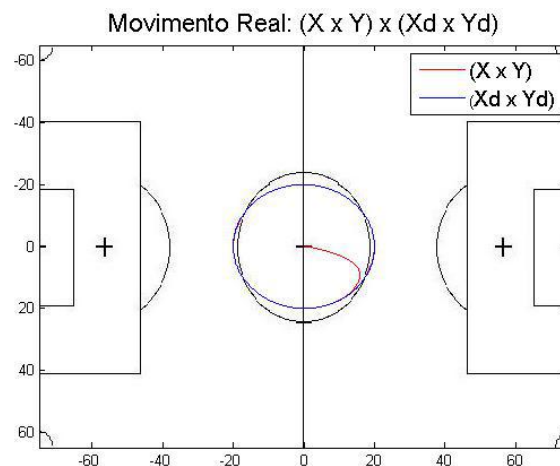


Figura 6.39: Movimento Real da simulação de trajetória circular utilizando o controle de linearização por realimentação para o modelo cinemático.

A Figura 6.39 mostra a simulação de uma trajetória circular utilizando o controle de linearização por realimentação para o modelo cinemático.

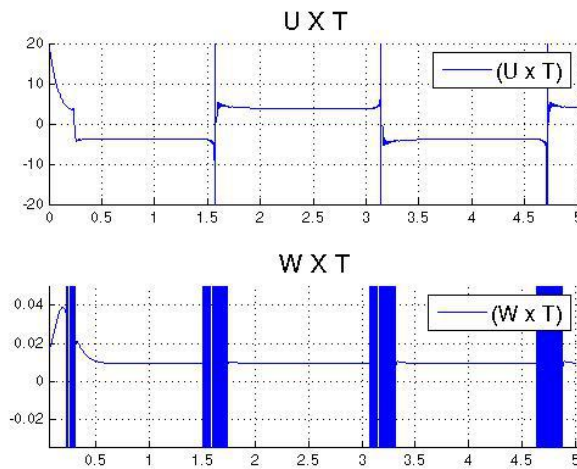


Figura 6.40: Sinais de Controle da simulação de trajetória circular utilizando o controle de linearização por realimentação para o modelo cinemático.

A Figura 6.40a, mostra que o sinal de controle linear varia de 20 à 5 em aproximadamente 0.25 segundos e depois se mantém constante em 5. Pode-se observar algumas singularidades nos tempos 1.6, 3.2 e 4.4, que ocorrem quando o $\cos(\theta)$ (equação 5.29) que aparece no denominador está muito próximo do valor 0, porém é tão rápida que não interfere na performance do robô.

A Figura 6.40b, mostra o sinal de controle angular constante em zero, com picos que ocorrem quando o $\cos(\theta)$, que interfere além da equação 5.29 nas equações 5.32 e 5.35 está próximo do valor 0. Porém são tão rápidos que não interferem na performance do robô.

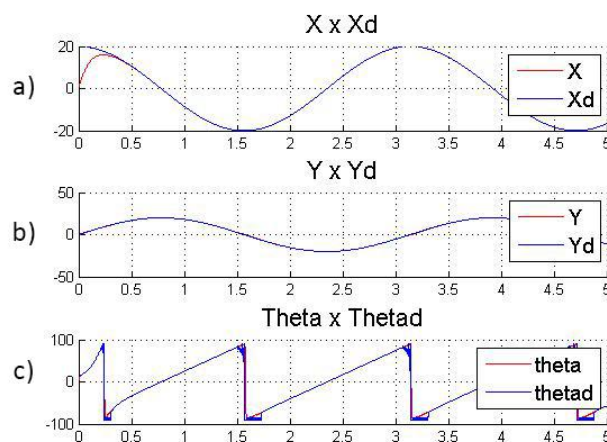


Figura 6.41: Posições linear e angular da simulação de trajetória circular utilizando o controle de linearização por realimentação para o modelo cinemático.

A Figura 6.41a, mostra um transitório que varia da posição inicial 0 à posição desejada em aproximadamente 0.25 segundos na posição linear x.

A Figura 6.41b, mostra a posição linear y igual a posição desejada durante toda a trajetória.

A Figura 6.41c, mostra a orientação do robô igual à orientação desejada.

Lissajous

Controle PD

Abaixo estão os resultados utilizando o controle PD para os modelos cinemático e dinâmico.

1. Modelo Cinemático

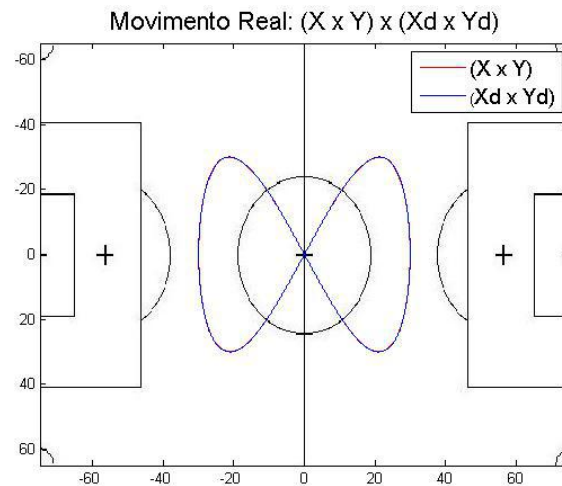


Figura 6.42: *Movimento Real da simulação de curva de Lissajous utilizando o controle PD para o modelo cinemático.*

A Figura 6.42 mostra a simulação de uma curva de Lissajous utilizando o controle PD para o modelo cinemático.

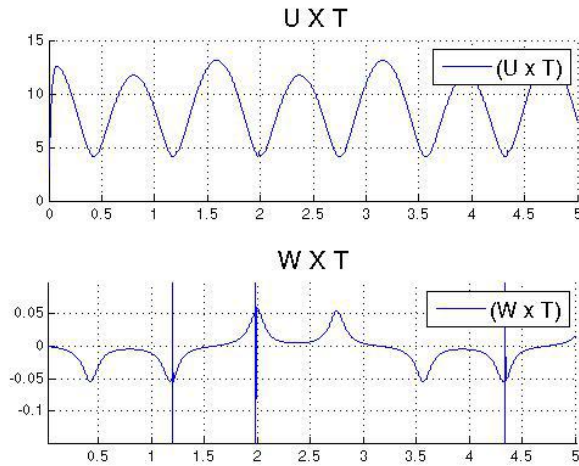


Figura 6.43: *Sinais de Controle da simulação de curva de Lissajous utilizando o controle PD para o modelo cinemático.*

A Figura 6.43a, mostra que o sinal de controle linear varia de 12 à 4 durante toda a trajetória.

A Figura 6.43b, mostra o sinal de controle angular constante em zero, com picos devido à singularidade do arctan como mostra a equação 5.2, que são tão rápidos que não interferem na performance do robô.

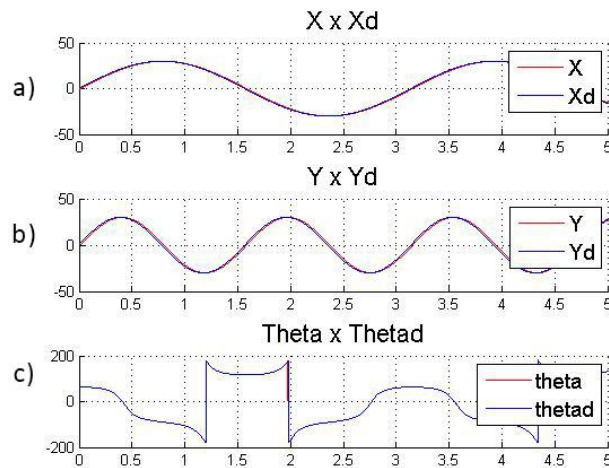


Figura 6.44: *Posições linear e angular da simulação de curva de Lissajous utilizando o controle PD para o modelo cinemático.*

A Figura 6.44a, mostra a posição linear x igual a posição desejada.

A Figura 6.44b, mostra a posição linear y igual a posição desejada.

A Figura 6.44c, mostra a orientação do robô igual à orientação desejada.

2. Modelo Dinâmico

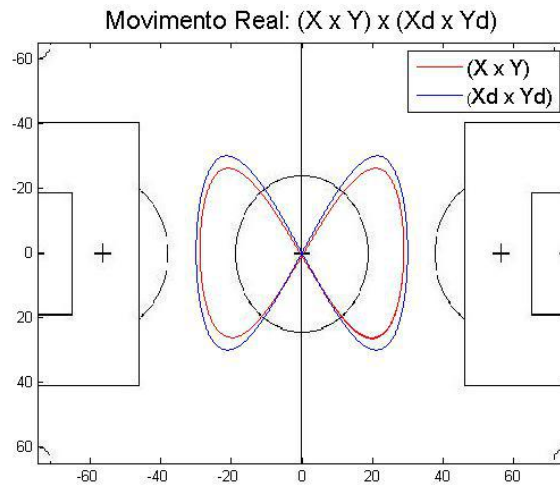


Figura 6.45: *Movimento Real da simulação de curva de Lissajous utilizando o controle PD para o modelo dinâmico.*

A Figura 6.45 mostra a simulação de uma curva de Lissajous utilizando o controle PD para o modelo dinâmico.

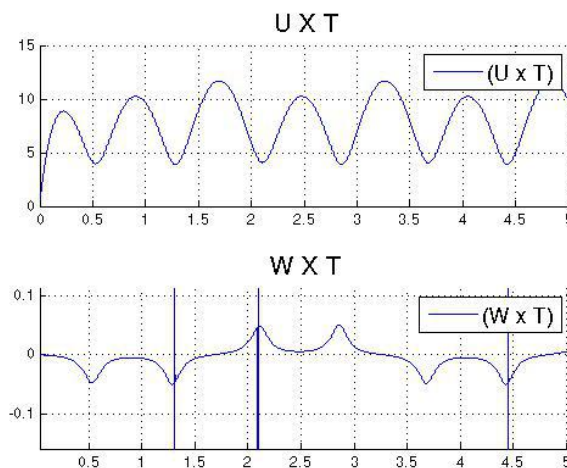


Figura 6.46: *Sinais de Controle da simulação de curva de Lissajous utilizando o controle PD para o modelo dinâmico.*

A Figura 6.46a, mostra que o sinal de controle linear varia de 2 à 12 durante toda a trajetória.

A Figura 6.46b, mostra o sinal de controle angular constante em zero, com picos nos tempos devido à singularidade do arctan como mostra a equação 5.2, que são tão rápidos que não interferem na performance do robô.

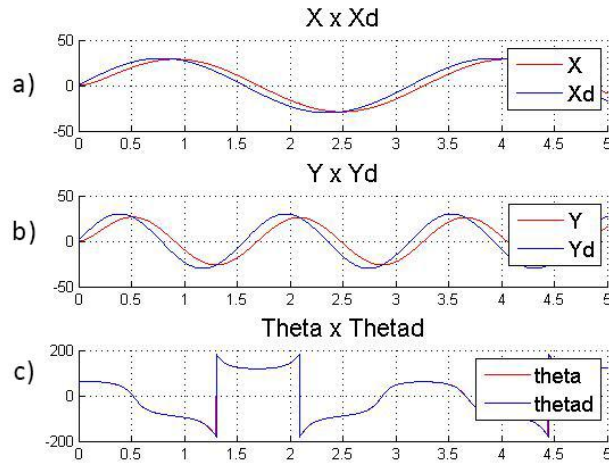


Figura 6.47: Posições linear e angular da simulação de curva de Lissajous utilizando o controle PD para o modelo dinâmico.

A Figura 6.47a, mostra o valor da posição linear x muito próximo da posição desejada.

A Figura 6.47b, mostra uma pequena diferença e atraso de 0.1 segundos na posição linear y.

A Figura 6.47c, mostra a orientação do robô igual à orientação desejada.

- Comparação entre os modelos Cinemático e Dinâmico

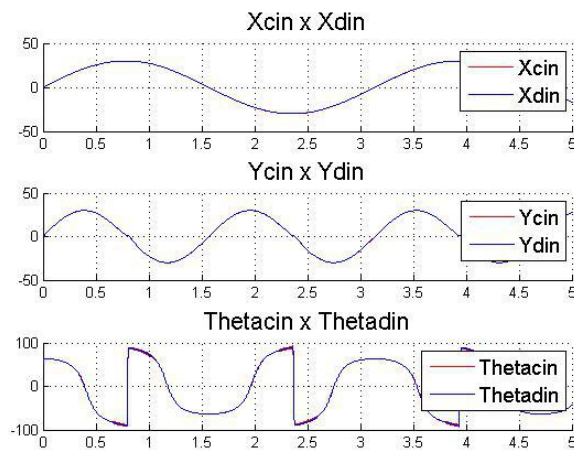


Figura 6.48: Comparação entre os modelos cinemático e dinâmico para curva de Lissajous utilizando o controle PD.

A Figura 6.48 compara novamente os dois modelos ao controlar o modelo cinemático em malha fechada e o modelo dinâmico em malha aberta e mostra que não

houve variação entre as posições linear e angular dos mesmos, ou seja, apenas controlando através das equações do modelo cinemático, o modelo dinâmico se comportou de maneira satisfatória. Comprovando assim o que já era esperado por conta das velocidades atingidas serem baixas.

Controle de linearização por realimentação

Abaixo estão os resultados utilizando o controle de linearização por realimentação para o modelo cinemático.

1. Modelo Cinemático

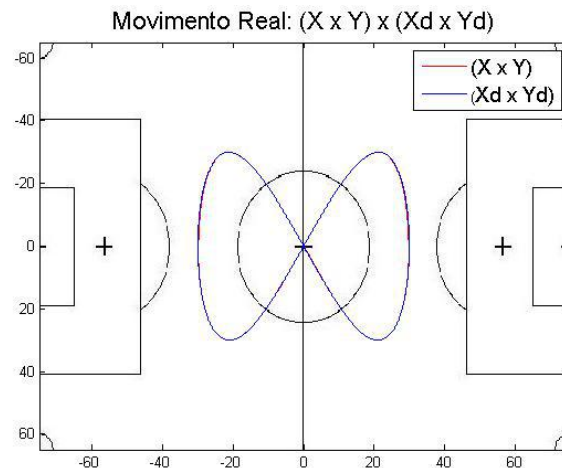


Figura 6.49: Movimento Real da simulação de curva de Lissajous utilizando o controle de linearização por realimentação para o modelo cinemático.

A Figura 6.49 mostra a simulação de uma curva de Lissajous utilizando o controle de linearização por realimentação para o modelo cinemático.

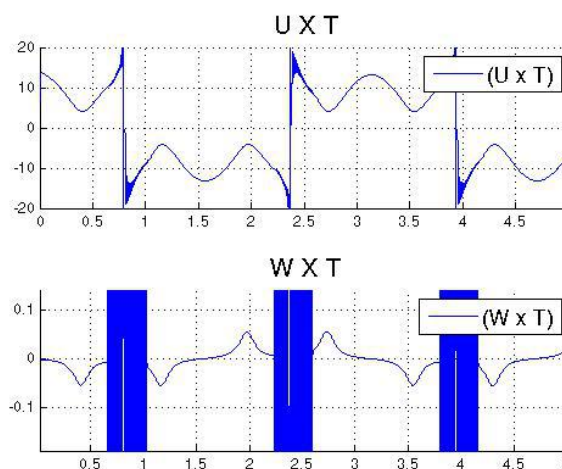


Figura 6.50: Sinais de Controle da simulação de curva de Lissajous utilizando o controle de linearização por realimentação para o modelo cinemático.

A Figura 6.50a, mostra que o sinal de controle linear varia de 20 à -20 em aproximadamente 3 segundos. Pode-se observar algumas singularidades nos tempos 0.7, 2.4 e 3.9, que ocorrem quando o $\cos(\theta)$ (equação 5.29) que aparece no denominador está muito próximo do valor 0. Porém, não interfere na performance do robô.

A Figura 6.50b, mostra o sinal de controle angular constante em zero, com picos que ocorrem quando o $\cos(\theta)$, que interfere além da equação 5.29 nas equações 5.32 e 5.35 está próximo do valor 0. Porém, não interferem na performance do robô.

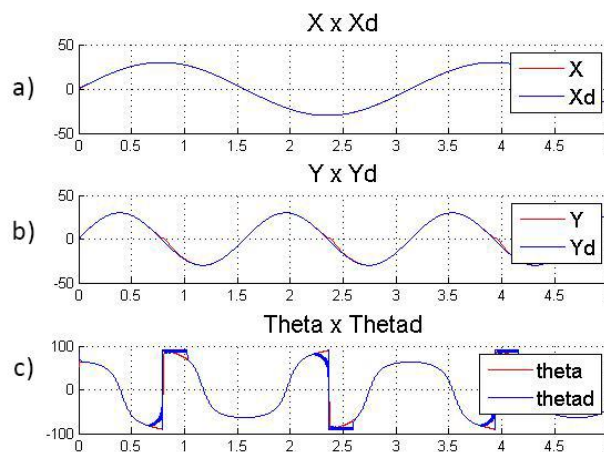


Figura 6.51: Posições linear e angular da simulação de curva de Lissajous utilizando o controle de linearização por realimentação para o modelo cinemático.

A Figura 6.51a, mostra a posição linear x igual a posição desejada.

A Figura 6.51b, mostra a posição linear y igual a posição desejada.

A Figura 6.51c, mostra a orientação do robô igual à orientação desejada.

Conclusão dos Movimentos de Zagueiro e Atacante

Os dois controladores atingiram o objetivo de realizar as trajetórias definidas como pode-se observar nas figuras de movimentação real do robô. Entretanto, novamente verificaram-se singularidades no controle de linearização por realimentação que precisou ter sua saída saturada apesar da singularidade ter um período de duração muito curto para causar qualquer efeito negativo no sistema.

Conclusões gerais de movimentação dos simuladores

Pode-se afirmar que, através das simulações, tanto o controle PD quanto o controle de linearização por realimentação são abordagens válidas e eficientes para o problema de rastreamento de trajetória em futebol de robôs. Porém, em mais de um caso a singularidades do controlador de linearização por realimentação se revelaram um problema que precisou ser contornado com saturação.

Capítulo 7

Experimentos e Resultados

Nesta seção serão mostrados os resultados obtidos utilizando as diferentes configurações dos controles P, PD e PID para o robô desenvolvido e adquirido de forma a verificar qual é a combinação ideal para o ambiente em questão. Foi analisada também a performance quanto à precisão e velocidade de resposta dos dois robôs de forma a testar o projeto desenvolvido contra uma solução de mercado específica para o objetivo de rastreamento de trajetória.

7.1 Estrutura da Bancada

A bancada montada para o desenvolvimento e testes do projeto foi composta de:

- Campo desmontável feito em madeira.

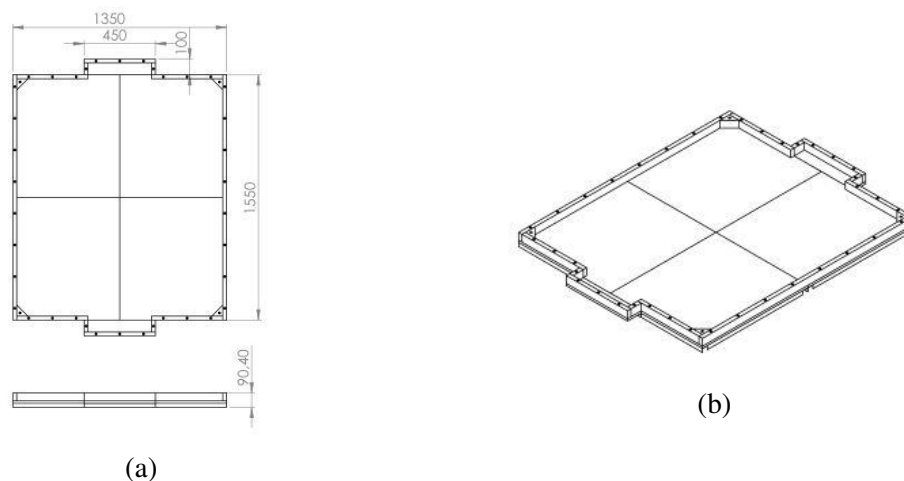


Figura 7.1: *Campo Desmontável*

- Estrutura em pvc para suporte à câmera utilizada na captura das imagens processadas na visão e às luzes.

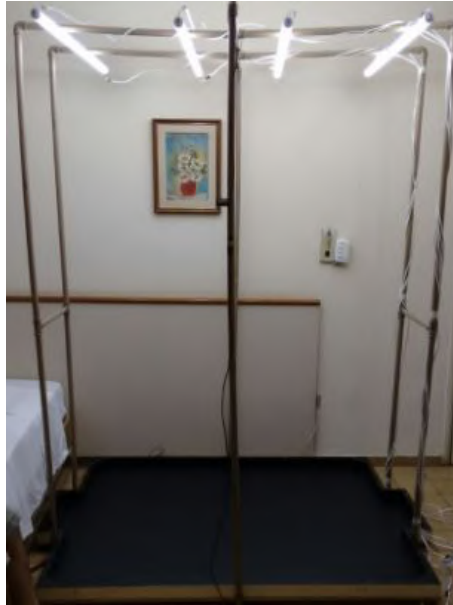


Figura 7.2: *Suporte para a câmera e luzes.*

- Câmera: Foi utilizada a *webcam* Microsoft LifeCam HD-3000.
- Computador.

7.2 Movimentos com Bola Parada: Pênalti, Cobrança de Faltas e Início de Jogo

Esta seção considera uma trajetória retilínea emulando uma cobrança de pênalti, que é um dos movimentos importantes com bola parada e pode ser expandido para cobranças de falta em diferentes posições do campo e início de jogo no centro do campo.

Controle P

1. Robô Desenvolvido

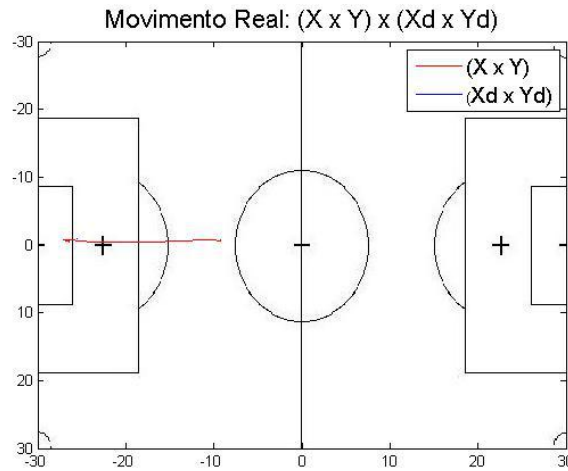


Figura 7.3: *Movimento Real do robô desenvolvido na cobrança de pênalti utilizando o controle P.*

A Figura 7.3 mostra o resultado obtido para uma cobrança de pênalti utilizando o controle P para o robô desenvolvido, onde o mesmo atinge a posição desejada.

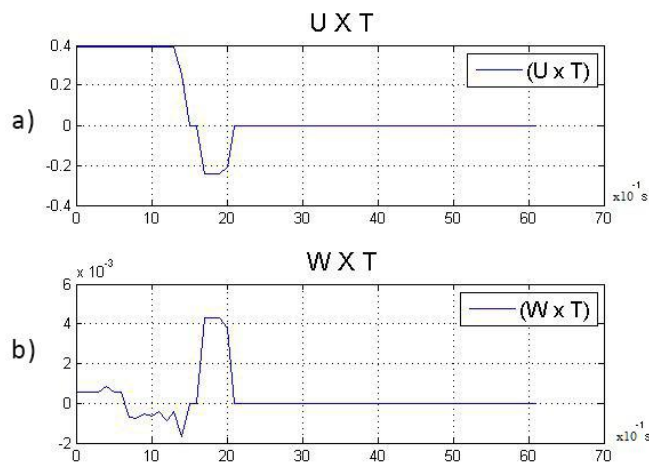


Figura 7.4: *Sinais de Controle do robô desenvolvido na cobrança de pênalti utilizando o controle P.*

A Figura 7.4a, mostra o transitório que varia de 0.4 à 0 em aproximadamente 1.5 segundos onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado e um *undershoot* de aproximadamente 0.25 em 0.5 segundos, onde o robô ultrapassou um pouco o ponto desejado e depois retornou.

A Figura 7.4b, mostra uma variação entre aproximadamente 1×10^{-3} e -2×10^{-3} no sinal de controle angular durante os 1.5 segundos em que o robô se aproximava do ponto desejado e um *overshoot* de $\approx 4 \times 10^{-3}$ durante os 0.5 segundos em que o robô ultrapassou um pouco o ponto desejado e retornou.

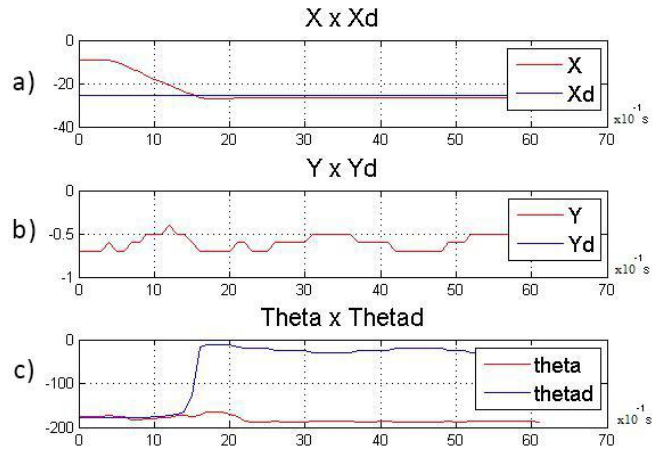


Figura 7.5: Posições linear e angular do robô desenvolvido na cobrança de pênalti utilizando o controle P.

A Figura 7.5a, mostra um transitório que varia da posição inicial -10cm à posição desejada -25cm em aproximadamente 1.5 segundos na posição linear x.

A Figura 7.5b, mostra uma pequena variação de $\approx 0.3cm$ na posição linear y em 1.5 segundos até chegar posição desejada. A variação após esses 1.5 segundos se deve ao erro da camera.

A Figura 7.5c, não mostra variação na posição angular theta nos 1.5 segundos em que o robô leva para chegar à posição desejada. A variação após esses 1.5 segundos se deve ao fato da orientação desejada final ir para zero, porém como o robô já estava no ponto, ele permaneceu com a orientação inicial.

2. Robô Adquirido

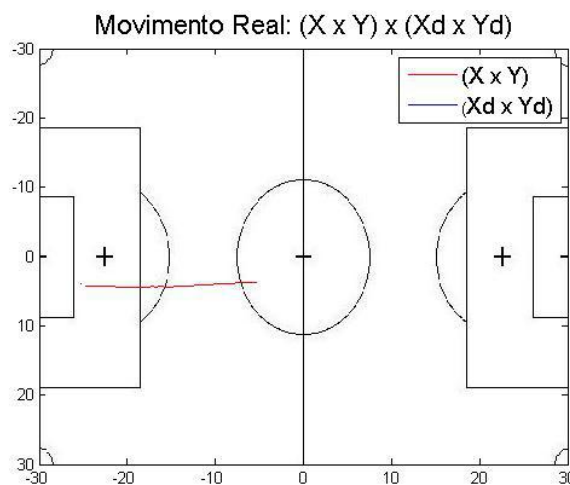


Figura 7.6: Movimento Real do robô adquirido na cobrança de pênalti utilizando o controle P.

A Figura 7.6 mostra o resultado obtido para uma cobrança de pênalti utilizando o controle P para o robô adquirido, onde o mesmo atinge a posição desejada.

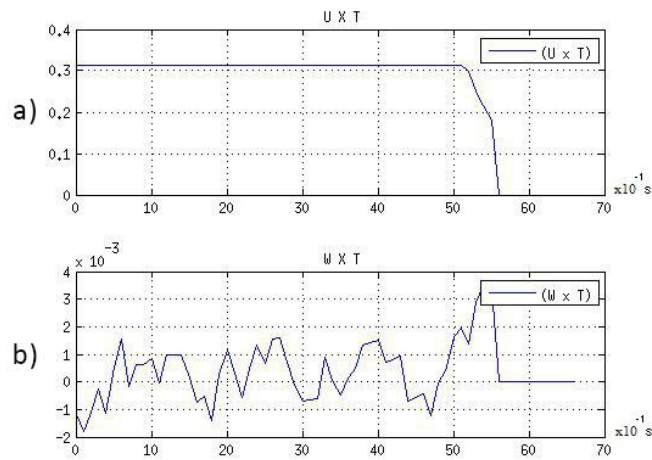


Figura 7.7: Sinais de Controle do robô adquirido na cobrança de pênalti utilizando o controle P.

A Figura 7.7a, mostra o transitório que varia de 0.3 à 0 em aproximadamente 5.5 segundos onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado.

A Figura 7.7b, mostra uma variação entre aproximadamente -2×10^{-3} e 4×10^{-3} no sinal de controle angular durante os 5.5 segundos em que o robô se aproximava do ponto desejado.

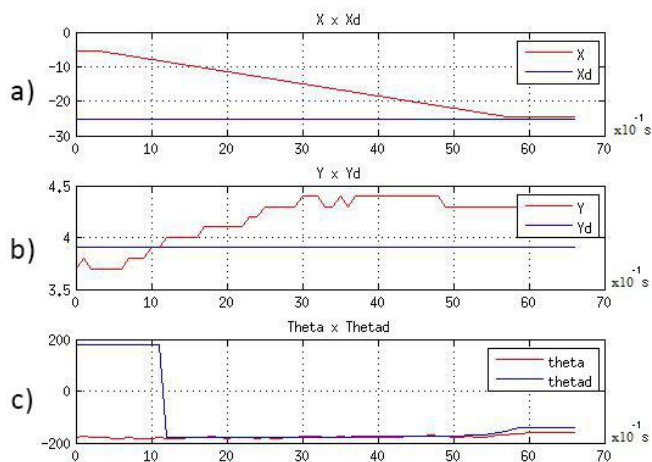


Figura 7.8: Posições linear e angular do robô adquirido na cobrança de pênalti utilizando o controle P.

A Figura 7.8a, mostra um transitório que varia da posição inicial -5cm à posição desejada -25cm em aproximadamente 5.5 segundos na posição linear x.

A Figura 7.8b, mostra uma pequena variação de $\approx 0.7cm$ na posição linear y em 5.5 segundos até chegar posição desejada. A diferença observada após esses 5.5 segundos se deve ao fato da posição de chegada ser considerada suficientemente próxima do ponto final.

A Figura 7.8c, mostra um θ desejado inicial errado no primeiro segundo devido à uma variação de iluminação que fez a câmera capturar uma orientação equivocada. Porém em 1.1 segundos, a orientação desejada é corrigida e o robô segue a mesma até chegar ao ponto final.

Controle PD

1. Robô Desenvolvido

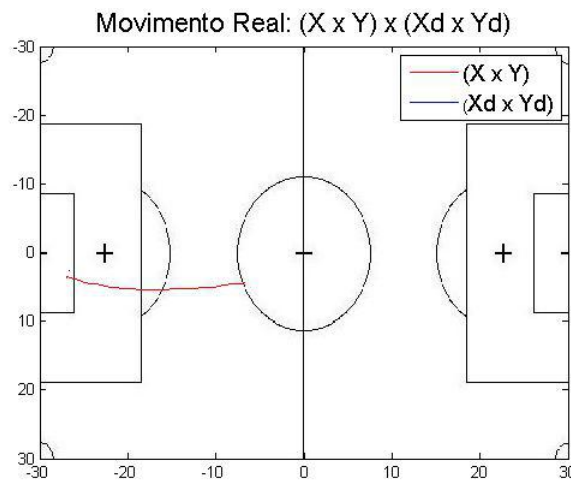


Figura 7.9: *Movimento Real do robô desenvolvido na cobrança de pênalti utilizando o controle PD.*

A Figura 7.9 mostra o resultado obtido para uma cobrança de pênalti utilizando o controle PD para o robô desenvolvido, onde o mesmo atinge a posição desejada.

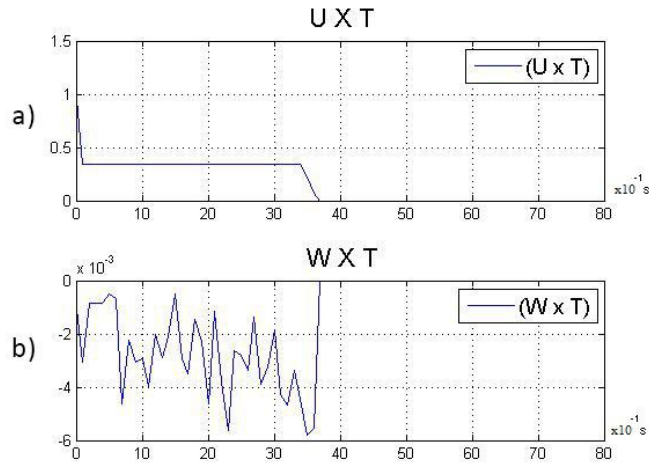


Figura 7.10: Sinais de Controle do robô desenvolvido na cobrança de pênalti utilizando o controle PD.

A Figura 7.10a, mostra o transitório que varia de 1 à 0 em aproximadamente 3.7 segundos onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado.

A Figura 7.10b, mostra uma variação entre aproximadamente -1×10^{-3} e 52×10^{-3} no sinal de controle angular durante os 3.7 segundos em que o robô se aproximava do ponto desejado.

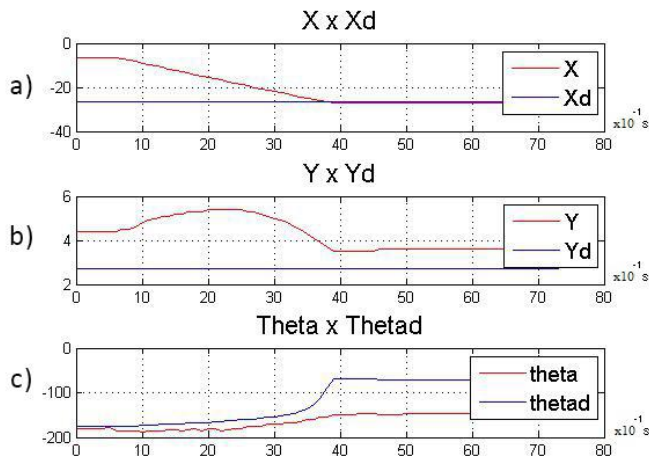


Figura 7.11: Posições linear e angular do robô desenvolvido na cobrança de pênalti utilizando o controle PD.

A Figura 7.11a, mostra um transitório que varia da posição inicial -10cm à posição desejada -25cm em aproximadamente 3.7 segundos na posição linear x.

A Figura 7.11b, mostra uma variação de $\approx 3\text{cm}$ na posição linear y em 3.7 segundos até chegar posição desejada. A diferença após esses 3.7 segundos se deve ao fato

da posição de chegada ser considerada suficientemente próxima do ponto final.

A Figura 7.11c, mostra uma pequena diferença na posição angular θ nos 3.7 segundos em que o robô leva para chegar à posição desejada. A diferença após esses 3.7 segundos se deve ao fato do robô já estar no ponto.

2. Robô Adquirido

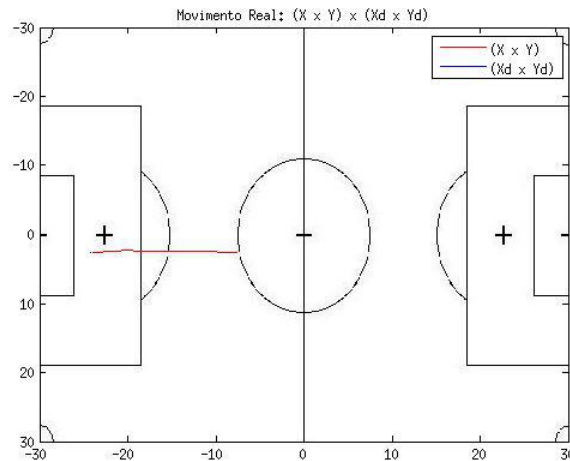


Figura 7.12: *Movimento Real do robô adquirido na cobrança de pênalti utilizando o controle PD.*

A Figura 7.12 mostra o resultado obtido para uma cobrança de pênalti utilizando o controle PD para o robô adquirido, onde o mesmo atinge a posição desejada.

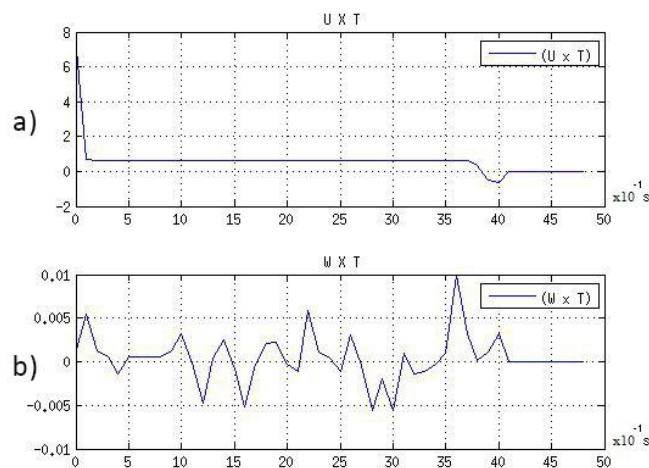


Figura 7.13: *Sinais de Controle do robô adquirido na cobrança de pênalti utilizando o controle PD.*

A Figura 7.13a, mostra o transiente que varia de 6 à 0 em aproximadamente 4 segundos onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado.

A Figura 7.13b, mostra uma variação entre aproximadamente -0.005 e 0.01 no sinal de controle angular durante os 4 segundos em que o robô se aproximava do ponto desejado.

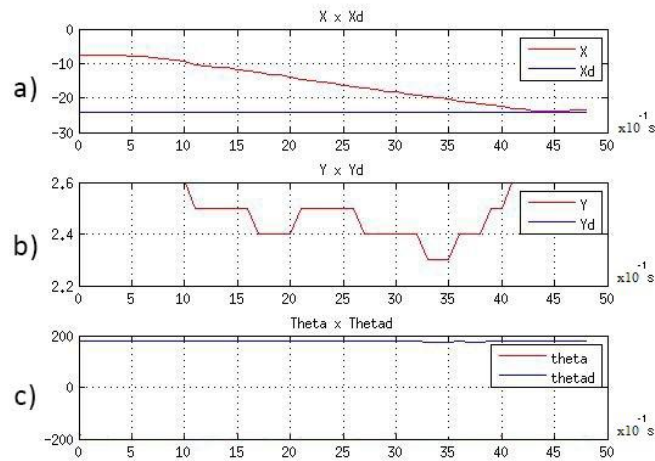


Figura 7.14: Posições linear e angular do robô adquirido na cobrança de pênalti utilizando o controle PD.

A Figura 7.14a, mostra um transitório que varia da posição inicial -10cm à posição desejada -25cm em aproximadamente 4 segundos na posição linear x.

A Figura 7.14b, mostra uma pequena variação de $\approx 0.3cm$ na posição linear y e em 4 segundos até chegar posição desejada.

A Figura 7.14c, não mostra diferença entre posição angular theta e a orientação desejada.

Controle PID

1. Robô Desenvolvido

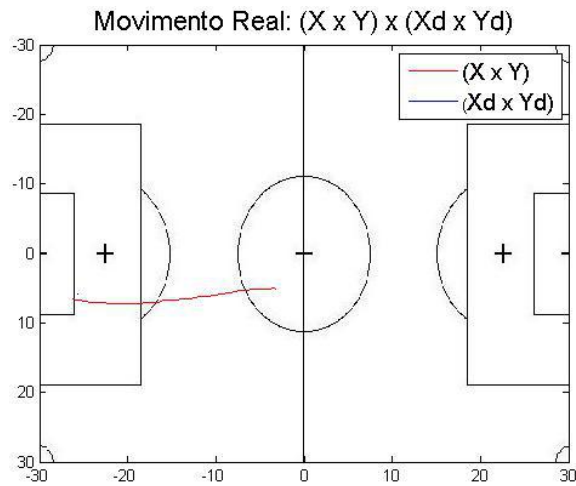


Figura 7.15: *Movimento Real do robô desenvolvido na cobrança de pênalti utilizando o controle PID.*

A Figura 7.15 mostra o resultado obtido para uma cobrança de pênalti utilizando o controle PID para o robô desenvolvido, onde o mesmo atinge a posição desejada.

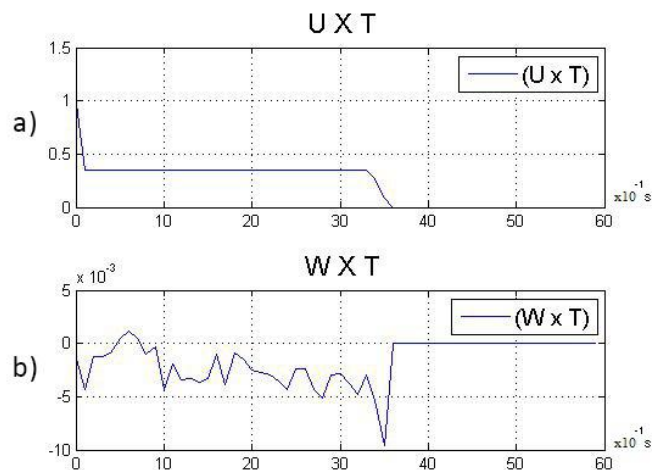


Figura 7.16: *Sinais de Control do robô desenvolvido na cobrança de pênalti utilizando o controle PID.*

A Figura 7.16a, mostra o transitório que varia de 1 à 0 em aproximadamente 3.5 segundos onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado.

A Figura 7.16b, mostra uma variação de aproximadamente 10×10^{-3} no sinal de controle angular durante os 3.5 segundos em que o robô se aproximava do ponto desejado.

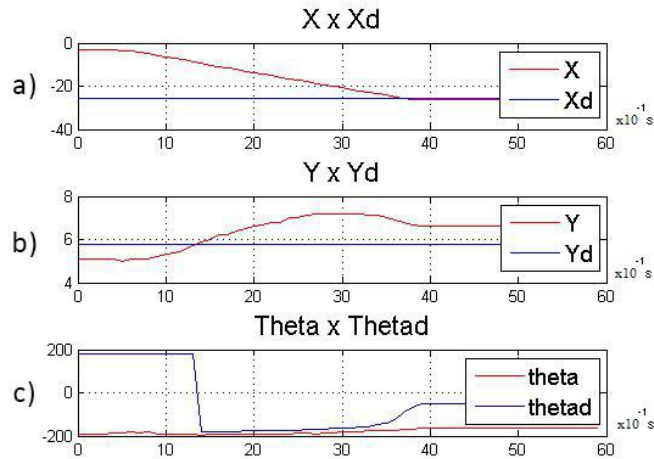


Figura 7.17: Posições linear e angular do robô desenvolvido na cobrança de pênalti utilizando o controle PID.

A Figura 7.17a, mostra um transitório que varia da posição inicial -5cm à posição desejada -25cm em aproximadamente 3.5 segundos na posição linear x.

A Figura 7.17b, mostra uma pequena variação de $\approx 1\text{cm}$ na posição linear y em 3.5 segundos até chegar posição desejada. A diferença após esses 15 segundos se deve ao fato da posição de chegada ser considerada suficientemente próxima do ponto final.

A Figura 7.17c, mostra um theta desejado inicial errado no primeiro segundo devido à uma variação de iluminação que fez a câmera capturar uma orientação equivocada. Porém em 1.3 segundos, a orientação desejada é corrigida e o robô segue a mesma até chegar ao ponto final. A diferença após esses 3.5 segundos se deve ao fato da orientação desejada final ir para zero, porém como o robô já estava no ponto, ele permaneceu com a orientação inicial.

2. Robô Adquirido

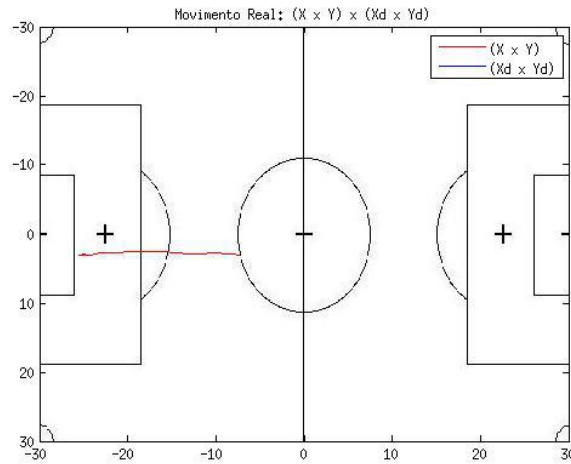


Figura 7.18: *Movimento Real do robô adquirido na cobrança de pênalti utilizando o controle PID.*

A Figura 7.18 mostra o resultado obtido para uma cobrança de pênalti utilizando o controle PID para o robô adquirido, onde o mesmo atinge a posição desejada.

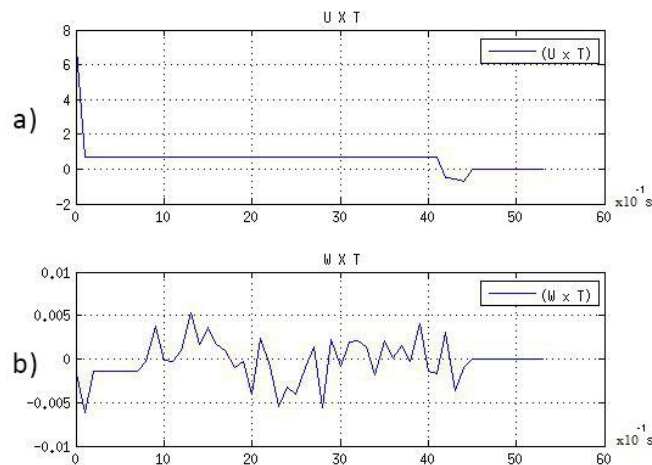


Figura 7.19: *Sinais de Controle do robô adquirido na cobrança de pênalti utilizando o controle PID.*

A Figura 7.19a, mostra o transitório que varia de 6 à 0 em aproximadamente 4.2 segundos onde o sinal de controle linear foi diminuindo conforme o robô se aproximava do ponto desejado e um *undershoot* de aproximadamente 0.5 em 0.3 segundos, onde o robô ultrapassou um pouco o ponto desejado e depois retornou.

A Figura 7.19b, mostra uma variação entre aproximadamente 0.01 no sinal de controle angular durante os 4.5 segundos em que o robô se aproximava do ponto desejado.

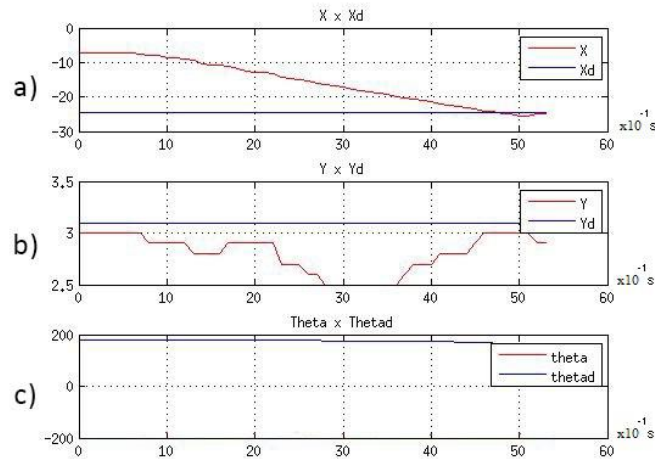


Figura 7.20: Posições linear e angular do robô adquirido na cobrança de pênalti utilizando o controle PID.

A Figura 7.20a, mostra um transitório que varia da posição inicial -10cm à posição desejada -25cm em aproximadamente 4.5 segundos na posição linear x.

A Figura 7.20b, mostra uma pequena variação de $\approx 0.5\text{cm}$ na posição linear y em 4.5 segundos até chegar posição desejada. A diferença após esses 4.5 segundos se deve ao fato da posição de chegada ser considerada suficientemente próxima do ponto final.

A Figura 7.20c, não mostra diferença entre as orientações desejada e real.

Conclusões sobre Movimento com Bola Parada

Observou-se que, entre as diversas configurações aqui apresentadas, não houve grande variação quanto à atingir o objetivo. Porém, durante os experimentos, podemos provar que o robô adquirido se mostra mais lento do que o desenvolvido para alcançar a posição desejada. O robô adquirido chega na posição com uma média de 6 segundos contra uma média de 3.5 segundos do robô desenvolvido.

Notou-se também que o robô desenvolvido precisa de sinais de controle mais elevados para realizar sua movimentação.

7.3 Movimentos de Defesa

O principal movimento de defesa consiste em uma trajetória retilínea dentro da área de defesa (*área do gol*).

Controle P

1. Robô Desenvolvido

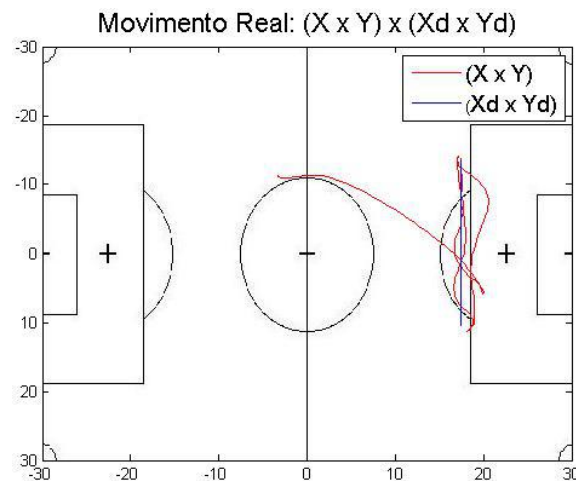


Figura 7.21: *Movimento Real do robô desenvolvido no movimento de defesa utilizando o controle P.*

A Figura 7.21 mostra o resultado obtido para um movimento de defesa utilizando o controle P para o robô desenvolvido, onde o mesmo segue a trajetória desejada.

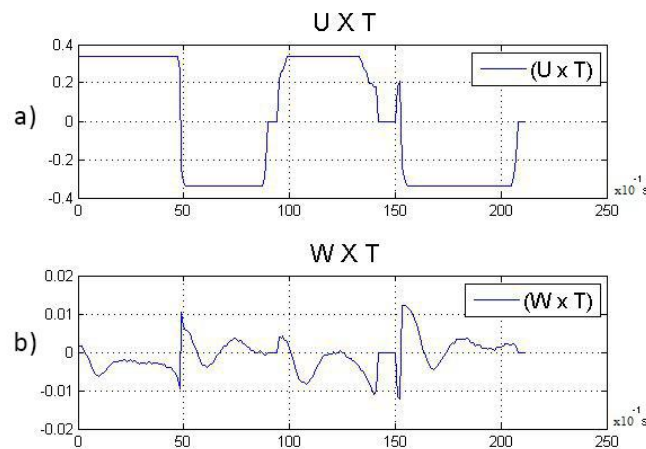


Figura 7.22: *Sinais de Controle do robô desenvolvido no movimento de defesa utilizando o controle P.*

A Figura 7.22a, mostra a variação de aproximadamente 0.7 a cada 5 segundos do sinal de controle linear, período de tempo para cada ida e volta da trajetória retilínea.

A Figura 7.22b, mostra uma variação de aproximadamente 0.02 no sinal de controle angular durante os 20 segundos em torno da orientação desejada para a trajetória.

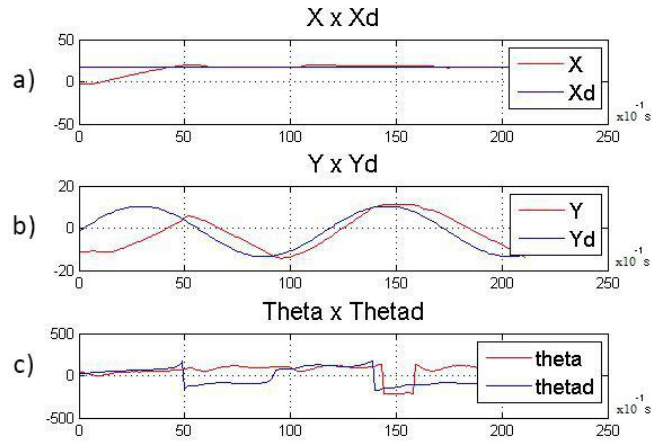


Figura 7.23: Posições linear e angular do robô desenvolvido no movimento de defesa utilizando o controle P.

A Figura 7.23a, mostra um diferença de aproximadamente 20cm durante os 4 segundos até que o robô atinge a posição linear x desejada e permanece nela até o fim da trajetória.

A Figura 7.23b, mostra uma diferença máxima de aproximadamente 15cm durante os primeiros segundos até chegar à posição inicial da trajetória e depois uma diferença muito pequena em torno da posição linear y da mesma.

A Figura 7.23c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

2. Robô Adquirido

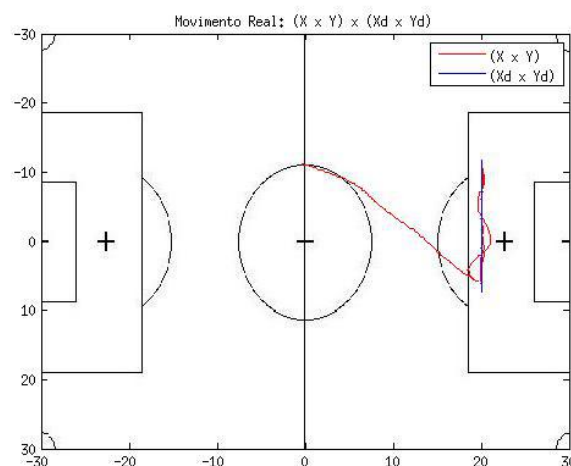


Figura 7.24: Movimento Real do robô adquirido no movimento de defesa utilizando o controle P.

A Figura 7.24 mostra o resultado obtido para um movimento de defesa utilizando o

controle P para o robô adquirido, onde o mesmo segue a trajetória desejada.

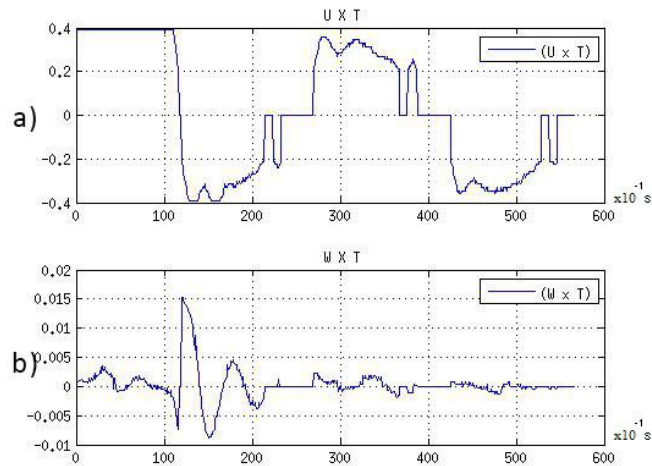


Figura 7.25: Sinais de Controle do robô adquirido no movimento de defesa utilizando o controle P.

A Figura 7.25a, mostra a variação de aproximadamente 0.8 a cada 12 segundos do sinal de controle linear, período de tempo para cada ida e volta da trajetória retilínea.

A Figura 7.25b, mostra uma variação de aproximadamente 0.025 no sinal de controle angular durante os 50 segundos em torno da orientação desejada para a trajetória.

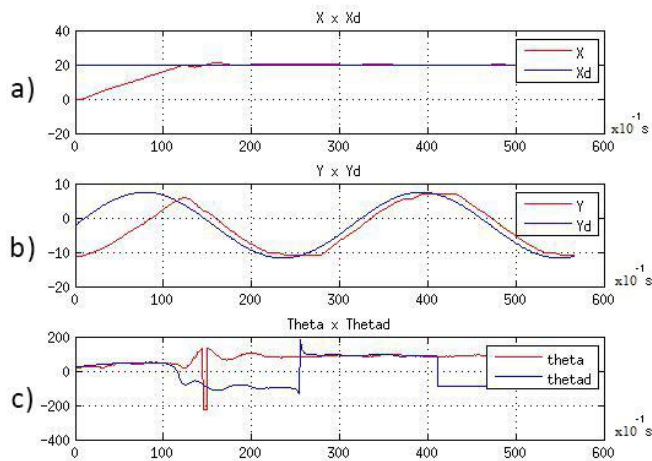


Figura 7.26: Posições linear e angular do robô adquirido no movimento de defesa utilizando o controle P.

A Figura 7.26a, mostra um diferença de 20cm durante os 12 segundos até que o robô atinge a posição linear x desejada e permanece nela até o fim da trajetória.

A Figura 7.26b, mostra uma diferença máxima de aproximadamente 11cm durante

os primeiros segundos até chegar à posição inicial da trajetória e depois uma diferença muito pequena em torno da posição linear y da mesma.

A Figura 7.26c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória, não apresentando valores negativos, visto que o robô anda de frente e de costas.

Controle PD

1. Robô Desenvolvido

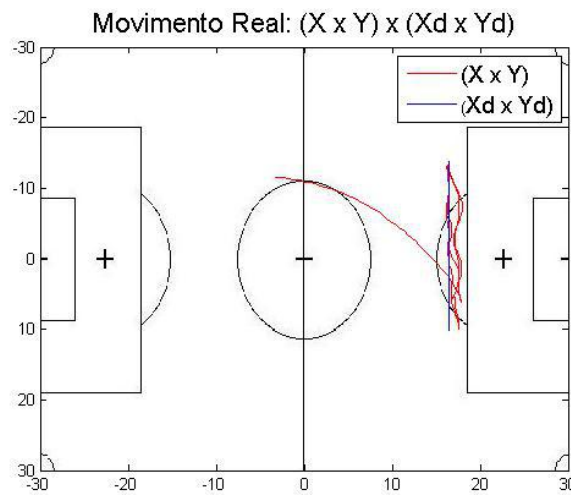


Figura 7.27: *Movimento Real do robô desenvolvido no movimento de defesa utilizando o controle PD.*

A Figura 7.27 mostra o resultado obtido para um movimento de defesa utilizando o controle PD para o robô desenvolvido, onde o mesmo segue a trajetória desejada.

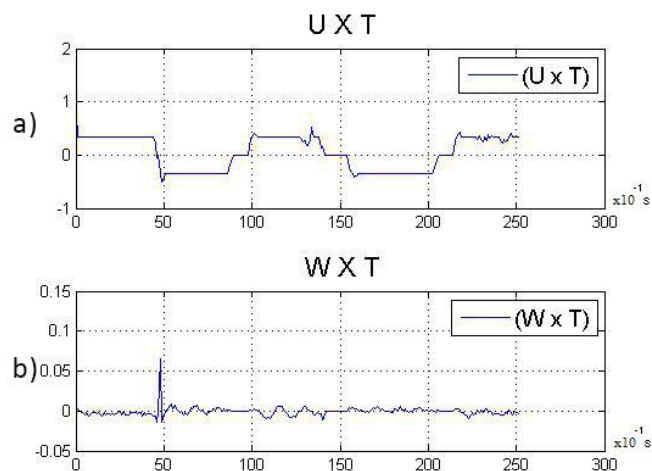


Figura 7.28: *Sinais de Controle do robô desenvolvido no movimento de defesa utilizando o controle PD.*

A Figura 7.28a, mostra a variação de aproximadamente 0.6 a cada 5 segundos do sinal de controle linear, período de tempo para cada ida e volta da trajetória retilínea. A Figura 7.28b, mostra uma variação muito pequena do controle angular em torno da orientação desejada para a trajetória, com apenas um *overshoot* de 0.06 na primeira troca de sinal.

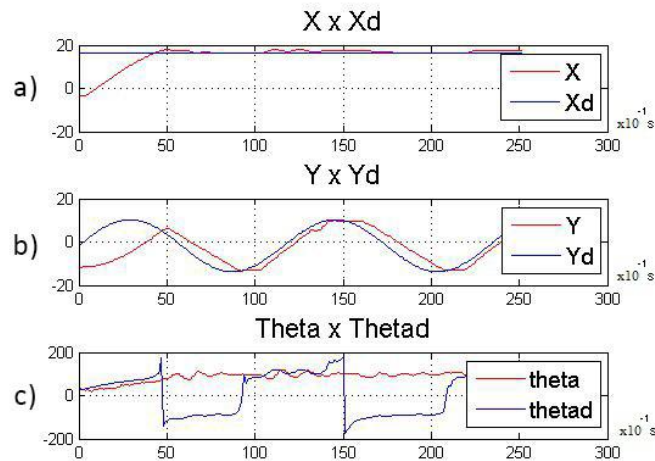


Figura 7.29: Posições linear e angular do robô desenvolvido no movimento de defesa utilizando o controle PD.

A Figura 7.29a, mostra um diferença de aproximadamente 20cm durante os 5 segundos até que o robô atinge a posição linear x desejada e permanece nela até o fim da trajetória.

A Figura 7.29b, mostra uma diferença máxima de aproximadamente 10cm durante os primeiros segundos até chegar à posição inicial da trajetória e depois uma diferença muito pequena em torno da posição linear y da mesma.

A Figura 7.29c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória, não apresentando valores negativos, visto que o robô anda de frente e de costas.

2. Robô Adquirido

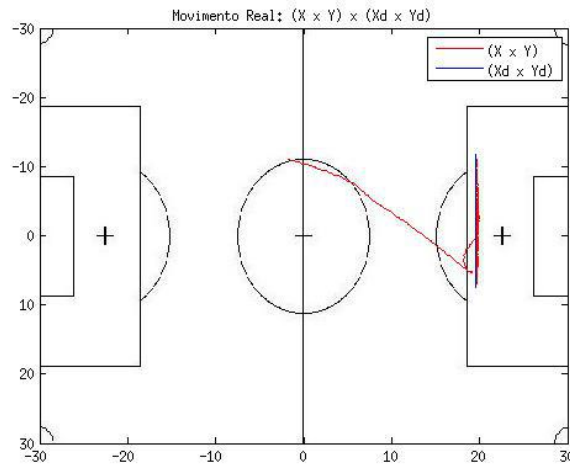


Figura 7.30: *Movimento Real do robô adquirido no movimento de defesa utilizando o controle PD.*

A Figura 7.30 mostra o resultado obtido para um movimento de defesa utilizando o controle PD para o robô adquirido, onde o mesmo segue a trajetória desejada.

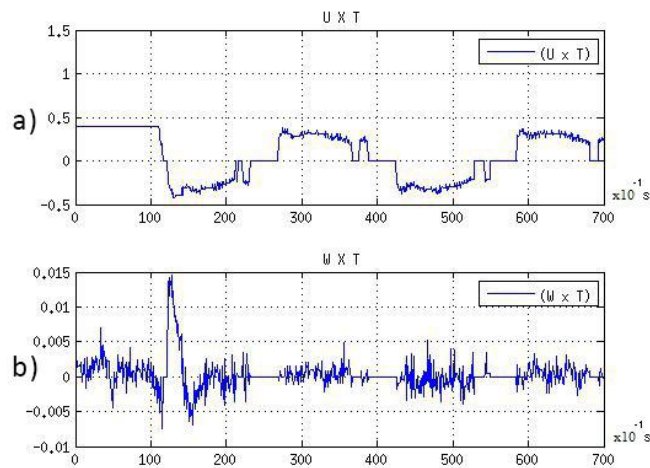


Figura 7.31: *Sinais de Controle do robô adquirido no movimento de defesa utilizando o controle PD.*

A Figura 7.31a, mostra a variação de aproximadamente 0.8 a cada 12 segundos do sinal de controle linear, período de tempo para cada ida e volta da trajetória retilínea.

A Figura 7.31b, mostra uma variação de aproximadamente 0.01 no sinal de controle angular em torno da orientação desejada para a trajetória, com apenas um *overshoot* de 0.015 na primeira troca de sinal.

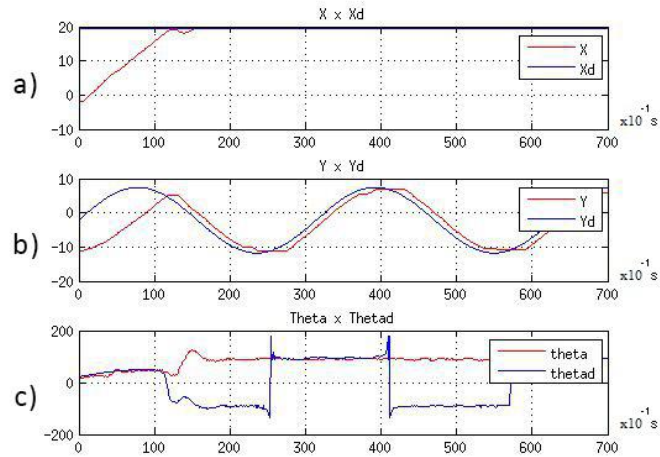


Figura 7.32: Posições linear e angular do robô adquirido no movimento de defesa utilizando o controle PD.

A Figura 7.32a, mostra um diferença de aproximadamente 20cm durante os 12 segundos até que o robô atinge a posição linear x desejada e permanece nela até o fim da trajetória.

A Figura 7.32b, mostra uma diferença máxima de aproximadamente 10cm durante os primeiros segundos até chegar à posição inicial da trajetória e depois uma diferença muito pequena em torno da posição linear y da mesma.

A Figura 7.32c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória, não apresentando valores negativos, visto que o robô anda de frente e de costas..

Controle PID

1. Robô Desenvolvido

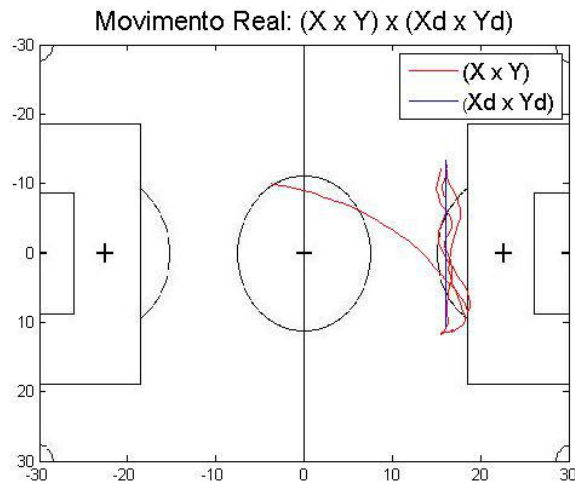


Figura 7.33: *Movimento Real do robô desenvolvido no movimento de defesa utilizando o controle PID.*

A Figura 7.33 mostra o resultado obtido para um movimento de defesa utilizando o controle PID para o robô desenvolvido, onde o mesmo segue a trajetória desejada.

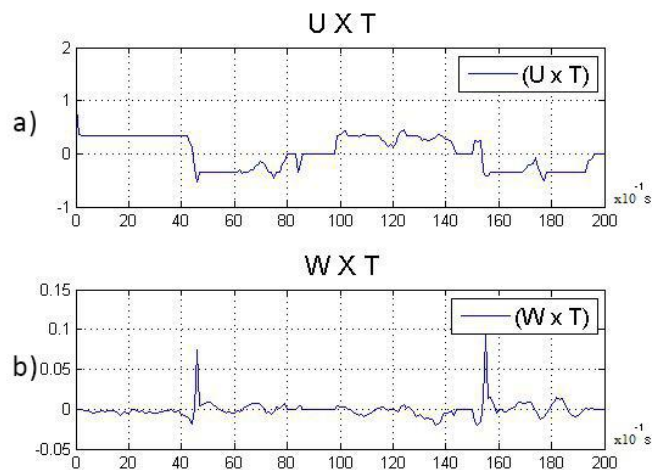


Figura 7.34: *Sinais de Controle do robô desenvolvido no movimento de defesa utilizando o controle PID.*

A Figura 7.34a, mostra a variação de aproximadamente 0.7 a cada 5 segundos do sinal de controle linear, período de tempo para cada ida e volta da trajetória retilínea.

A Figura 7.34b, mostra uma variação de aproximadamente 0.01 no sinal de controle angular em torno da orientação desejada para a trajetória, com apenas 2 *overshoots* de 0.06 em duas trocas de sinal..

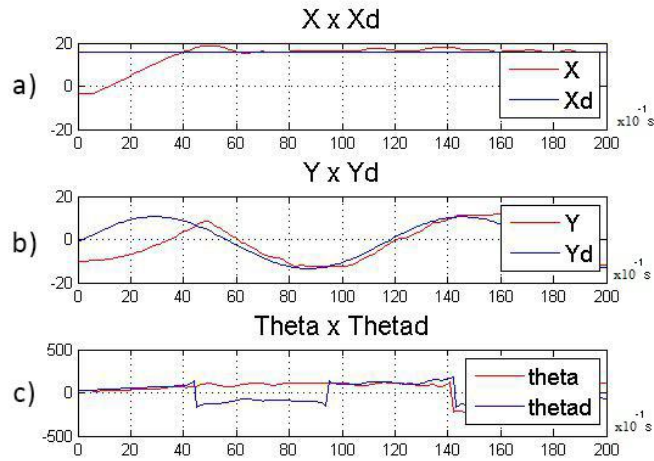


Figura 7.35: Posições linear e angular do robô desenvolvido no movimento de defesa utilizando o controle PID.

A Figura 7.35a, mostra uma diferença de aproximadamente 20cm durante os 4 segundos até que o robô atinge a posição linear x desejada e permanece nela até o fim da trajetória.

A Figura 7.35b, mostra uma diferença máxima de aproximadamente 10cm durante os primeiros segundos até chegar à posição inicial da trajetória e depois uma diferença muito pequena em torno da posição linear y da mesma.

A Figura 7.35c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória, não apresentando valores negativos, visto que o robô anda de frente e de costas.

2. Robô Adquirido

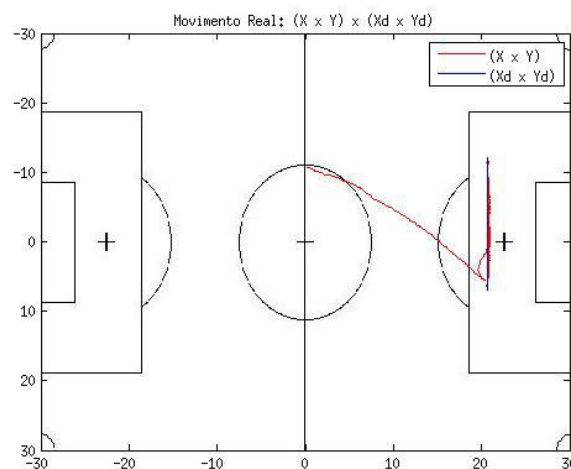


Figura 7.36: Movimento Real do robô adquirido no movimento de defesa utilizando o controle PID.

A Figura 7.36 mostra o resultado obtido para um movimento de defesa utilizando o controle PID para o robô adquirido, onde o mesmo segue a trajetória desejada.

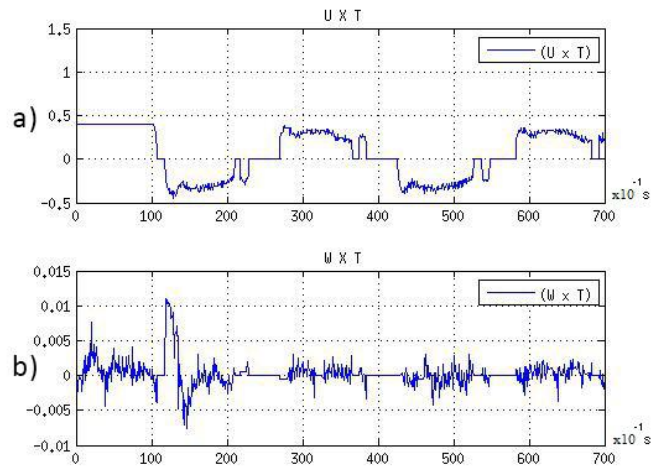


Figura 7.37: Sinais de Controle do robô adquirido no movimento de defesa utilizando o controle PID.

A Figura 7.37a, mostra a variação de aproximadamente 0.8 a cada 12 segundos do sinal de controle linear, período de tempo para cada ida e volta da trajetória retilínea.

A Figura 7.37b, mostra uma variação de aproximadamente 0.005 no sinal de controle angular em torno da orientação desejada para a trajetória com apenas um *overshoot* de 0.01 na primeira troca de sinal.

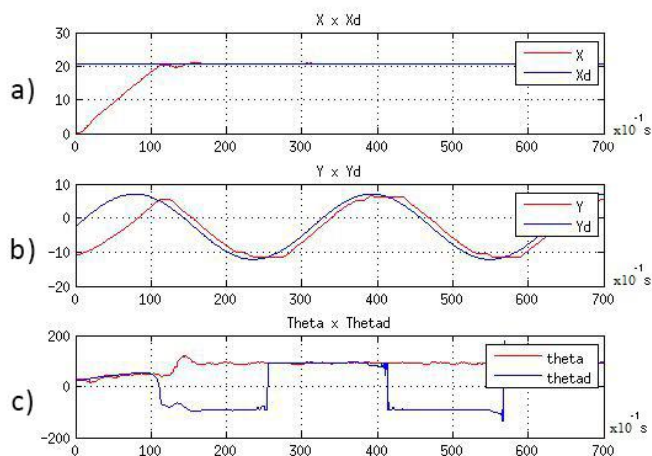


Figura 7.38: Posições linear e angular do robô adquirido no movimento de defesa utilizando o controle PID.

A Figura 7.38a, mostra um diferença de aproximadamente 20cm durante os 12 segundos até que o robô atinge a posição linear x desejada e permanece nela até o fim da trajetória.

A Figura 7.38b, mostra uma diferença máxima de aproximadamente 10cm durante os primeiros segundos até chegar à posição inicial da trajetória e depois uma diferença muito pequena em torno da posição linear y da mesma.

A Figura 7.38c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória, não apresentando valores negativos, visto que o robô anda de frente e de costas.

Conclusões sobre Movimentos de Defesa

Pode-se concluir que o controle puramente proporcional tanto para o robô desenvolvido quanto para o robô adquirido não atinge o objetivo de maneira eficiente, pois permite muita oscilação em torno do eixo vertical. Ao introduzir o elemento derivativo, ambos respondem de maneira a diminuir essa oscilação mas o robô adquirido tem resultados mais próximos do desejado. A introdução do fator integral não produz grandes efeitos em ambos os sistemas.

Verificou-se também que, ao introduzir o fator integral no sistema, foi necessário aplicar técnicas para que o efeito integral não atingisse níveis muito altos (causando instabilidade do sistema). Dessa forma, concluímos que o controle do tipo PD é o mais eficiente tanto para o sistema desenvolvido quanto para o adquirido.

7.4 Movimentos de Zagueiro e Atacante

Nesta seção serão consideradas as trajetórias circular e de Lissajous emulando uma possível estratégia de ataque e defesa, que não é o foco deste trabalho.

Círculo

Controle P

1. Robô Desenvolvido

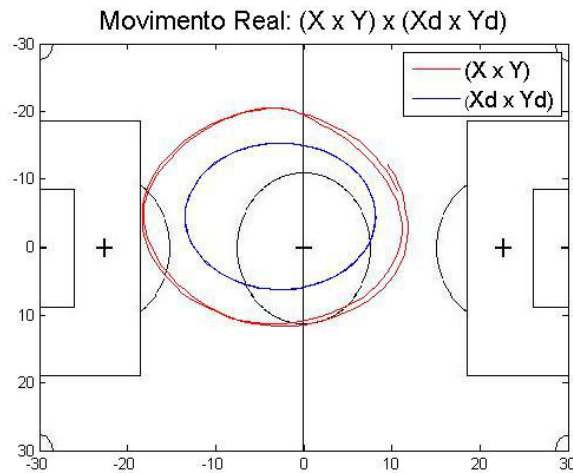


Figura 7.39: *Movimento Real do robô desenvolvido na trajetória circular utilizando o controle P.*

A Figura 7.39 mostra o resultado obtido para uma trajetória circular utilizando o controle P para o robô desenvolvido. Lembrando que este robô possui 10cm de diâmetro e sua posição é calculada a partir de seu centro. Logo, ele seguiu muito próximo à esta trajetória .

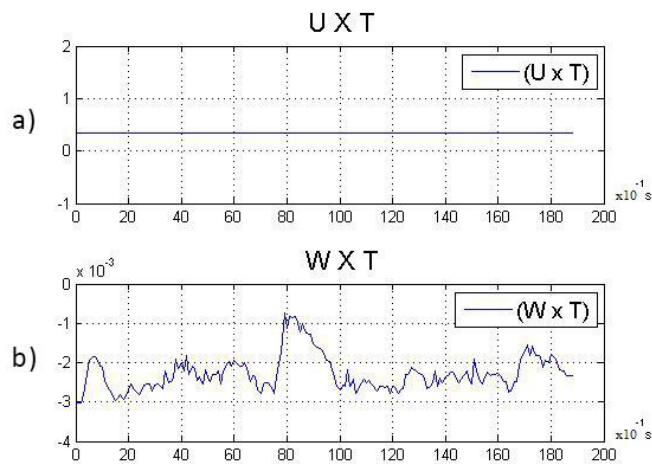


Figura 7.40: *Sinais de Controle do robô desenvolvido na trajetória circular utilizando o controle P.*

A Figura 7.40a, mostra que o sinal de controle linear se manteve constante com valor de aproximadamente 3 durante toda a trajetória.

A Figura 7.40b, mostra uma variação de aproximadamente 2×10^{-3} no sinal de controle angular.

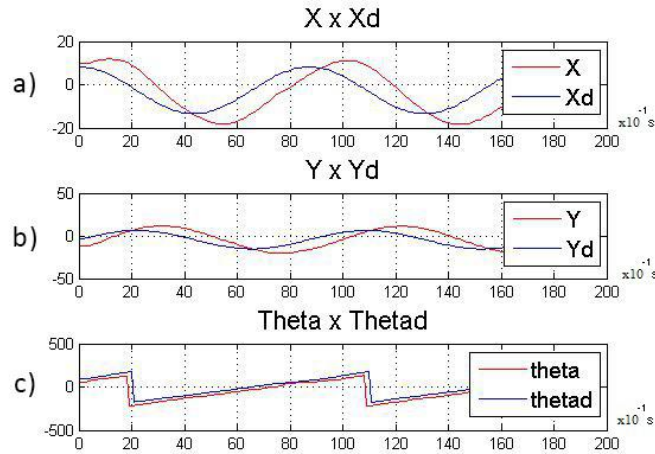


Figura 7.41: Posições linear e angular do robô desenvolvido na trajetória circular utilizando o controle P.

A Figura 7.41a, mostra um diferença de aproximadamente 5cm na posição linear x desejada com um atraso de menos de 2 segundos.

A Figura 7.41b, mostra a posição linear em y muito próxima da desejada com um atraso de menos de 1 segundo.

A Figura 7.41c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

2. Robô Adquirido

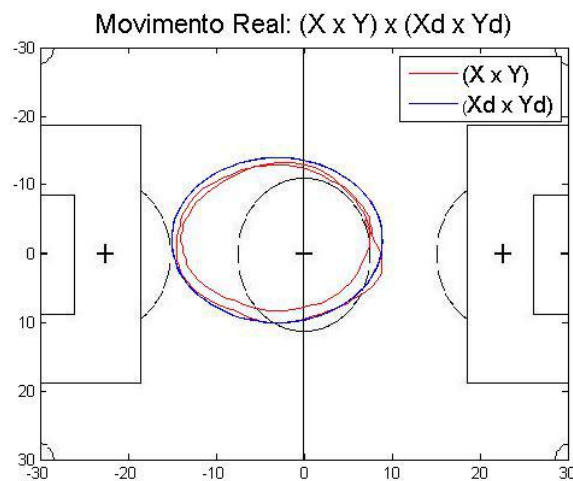


Figura 7.42: Movimento Real do robô adquirido na trajetória circular utilizando o controle P.

A Figura 7.42 mostra o resultado obtido para uma trajetória circular utilizando o controle P para o robô adquirido. Lembrando que este robô possui 8cm de diâmetro

e sua posição é calculada a partir de seu centro. Logo, ele seguiu muito próximo à esta trajetória.

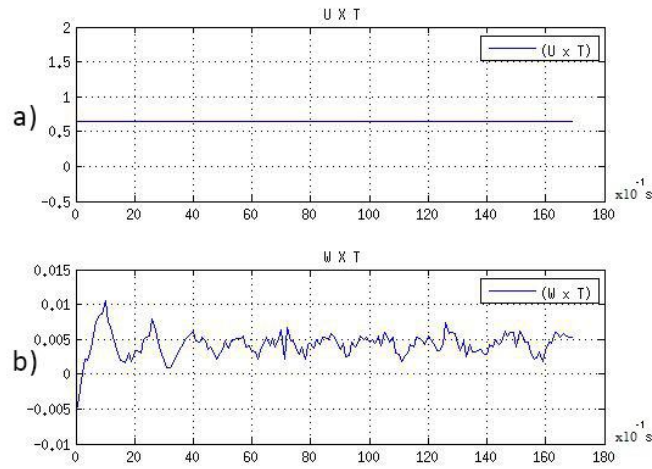


Figura 7.43: Sinais de Controle do robô adquirido na trajetória circular utilizando o controle P .

A Figura 7.43a, mostra que o sinal de controle linear se manteve constante com valor de aproximadamente 0.6 durante toda a trajetória.

A Figura 7.43b, mostra uma variação de aproximadamente 0.01 no sinal de controle angular.

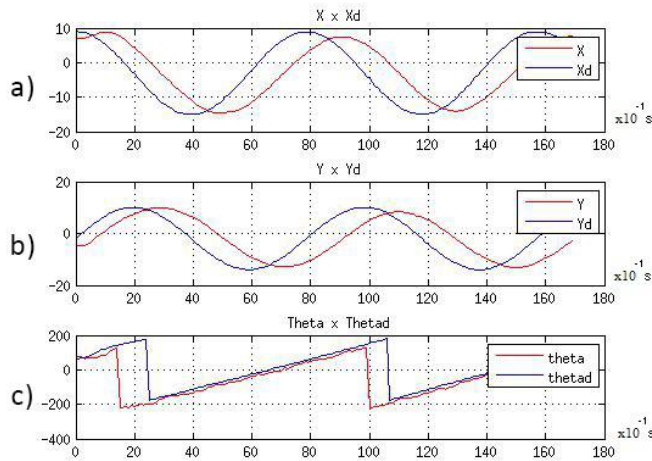


Figura 7.44: Posições linear e angular do robô adquirido na trajetória circular utilizando o controle P .

A Figura 7.44a, mostra um diferença de aproximadamente 5cm na posição linear x desejada com um atraso de menos de 2 segundos.

A Figura 7.44b, mostra a posição linear em y muito próxima da desejada com um atraso de aproximadamente 1 segundo.

A Figura 7.44c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória com um atraso de menos de 1 segundo.

Controle PD

1. Robô Desenvolvido

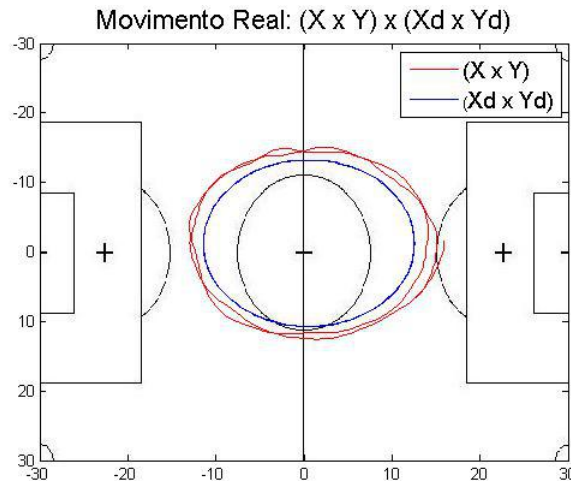


Figura 7.45: Movimento Real do robô desenvolvido na trajetória circular utilizando o controle PD.

A Figura 7.45 mostra o resultado obtido para uma trajetória circular utilizando o controle PD para o robô desenvolvido. Lembrando que este robô possui 10cm de diâmetro e sua posição é calculada a partir de seu centro. Logo, ele seguiu muito próximo à esta trajetória.

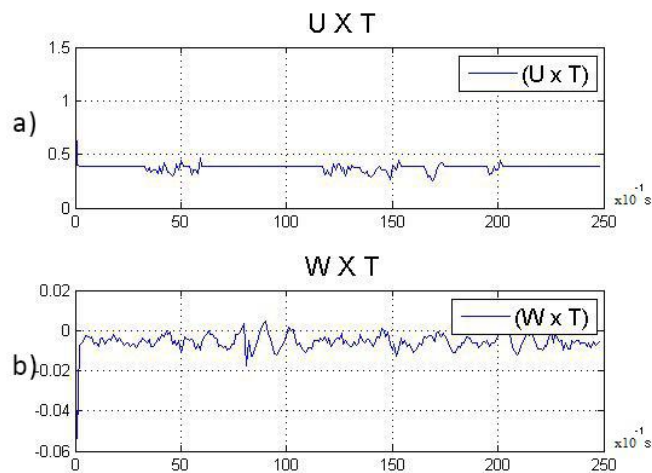


Figura 7.46: Sinais de Controle do robô desenvolvido na trajetória circular utilizando o controle PD.

A Figura 7.46a, mostra que o sinal de controle linear se manteve com variações muito pequenas ao redor do valor 0.4 durante toda a trajetória.

A Figura 7.46b, mostra uma variação de aproximadamente 0.02 no sinal de controle angular.

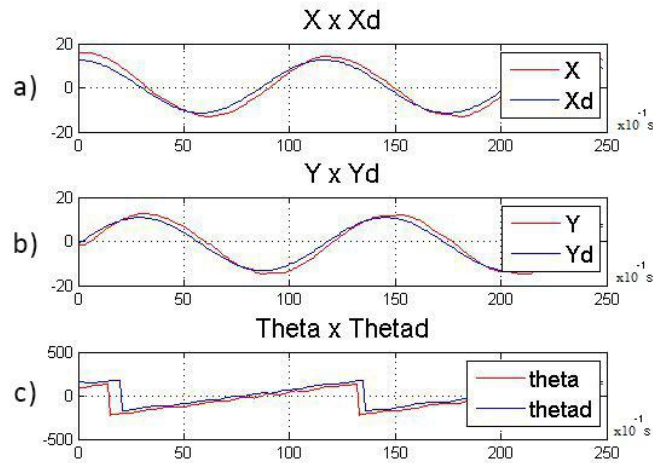


Figura 7.47: Posições linear e angular do robô desenvolvido na trajetória circular utilizando o controle PD.

A Figura 7.47a, mostra uma diferença muito pequena na posição linear x desejada.

A Figura 7.47b, mostra a posição linear em y muito próxima da desejada.

A Figura 7.47c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

2. Robô Adquirido

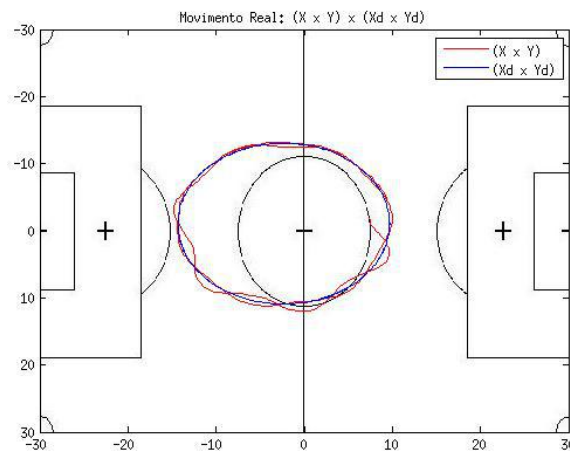


Figura 7.48: Movimento Real do robô adquirido na trajetória circular utilizando o controle PD.

A Figura 7.48 mostra o resultado obtido para uma trajetória circular utilizando o controle PD para o robô adquirido. Lembrando que este robô possui 8cm de diâmetro e sua posição é calculada a partir de seu centro. Logo, ele seguiu muito próximo à esta trajetória.

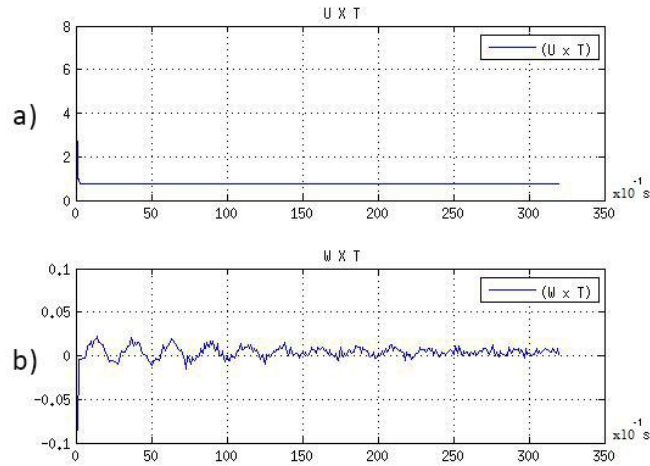


Figura 7.49: Sinais de Controle do robô adquirido na trajetória circular utilizando o controle PD.

A Figura 7.49a, mostra que o sinal de controle linear se manteve constante com valor de aproximadamente 1 durante toda a trajetória.

A Figura 7.49b, mostra uma variação de aproximadamente 0.02 no sinal de controle angular.

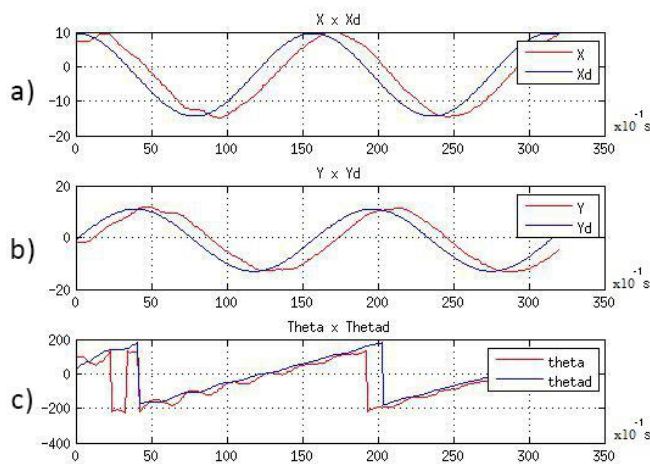


Figura 7.50: Posições linear e angular do robô adquirido na trajetória circular utilizando o controle PD.

A Figura 7.50a, mostra um diferença de aproximadamente 5cm na posição linear x desejada com um atraso de menos de 2 segundos.

A Figura 7.50b, mostra um diferença de aproximadamente 5cm na posição linear y desejada com um atraso de menos de 2 segundos.

A Figura 7.50c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

Controle PID

1. Robô Desenvolvido

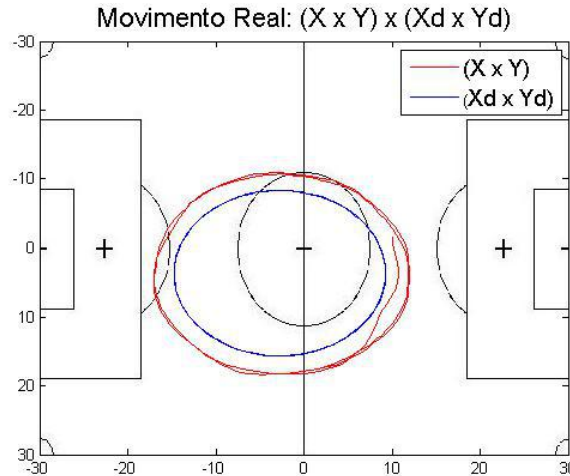


Figura 7.51: *Movimento Real do robô desenvolvido na trajetória circular utilizando o controle PID.*

A Figura 7.51 mostra o resultado obtido para uma trajetória circular utilizando o controle PID para o robô desenvolvido. Lembrando que este robô possui 10cm de diâmetro e sua posição é calculada a partir de seu centro. Logo, ele seguiu muito próximo à esta trajetória.

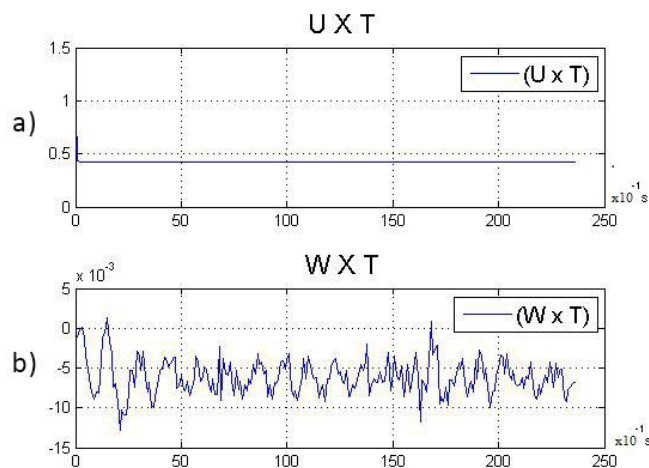


Figura 7.52: *Sinais de Controle do robô desenvolvido na trajetória circular utilizando o controle PID.*

A Figura 7.52a, mostra que o sinal de controle linear se manteve constante com valor de aproximadamente 0.4 durante toda a trajetória.

A Figura 7.52b, mostra uma variação de aproximadamente 6×10^{-3} no sinal de controle angular.

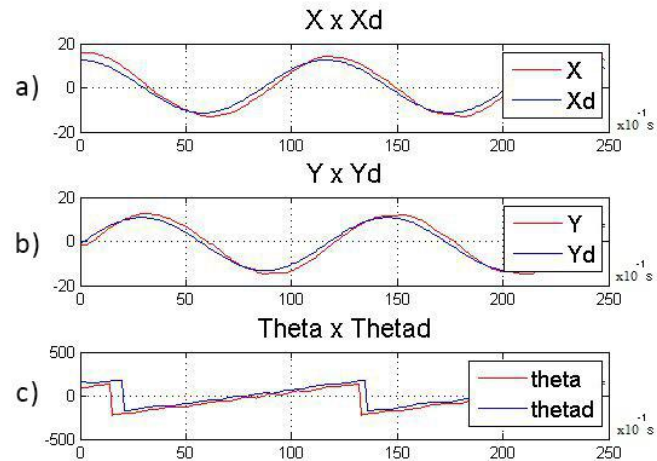


Figura 7.53: Posições linear e angular do robô desenvolvido na trajetória circular utilizando o controle PID.

A Figura 7.53a, mostra uma diferença e atraso muito pequenos para a posição desejada x.

A Figura 7.53b, mostra uma diferença e atraso muito pequenos para a posição desejada y.

A Figura 7.53c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

2. Robô Adquirido

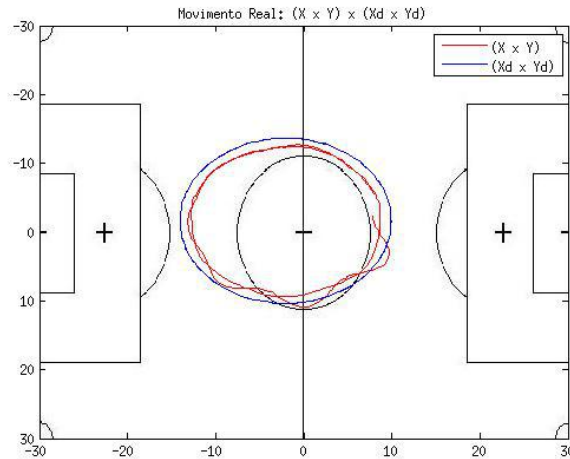


Figura 7.54: *Movimento Real do robô adquirido na trajetória circular utilizando o controle PID.*

A Figura 7.54 mostra o resultado obtido para uma trajetória circular utilizando o controle PID para o robô adquirido. Lembrando que este robô possui 8cm de diâmetro e sua posição é calculada a partir de seu centro. Logo, ele seguiu muito próximo à esta trajetória.

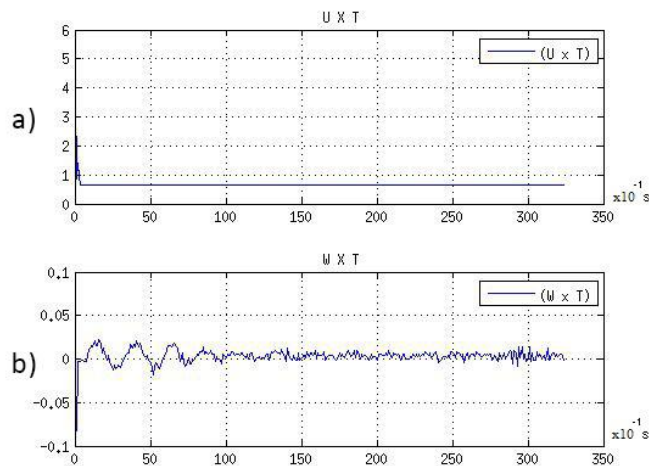


Figura 7.55: *Sinais de Controle do robô adquirido na trajetória circular utilizando o controle PID.*

A Figura 7.55a, mostra que o sinal de controle linear se manteve constante com valor de aproximadamente 0.9 durante toda a trajetória.

A Figura 7.55b, mostra uma variação de aproximadamente 0.02 no sinal de controle angular.

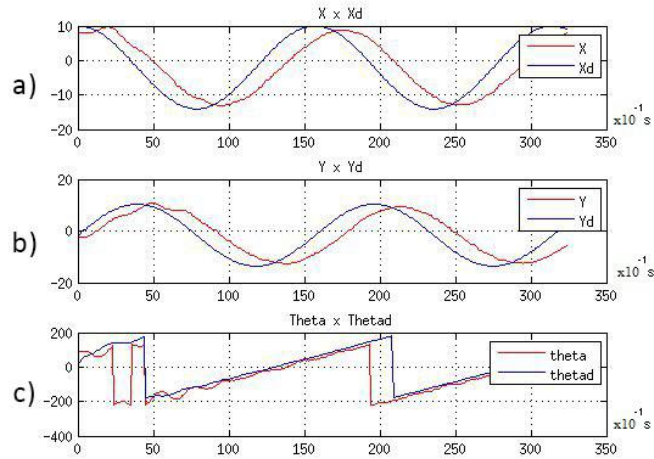


Figura 7.56: Posições linear e angular do robô adquirido na trajetória circular utilizando o controle PID.

A Figura 7.56a, mostra um diferença de aproximadamente 5cm na posição linear x desejada com um atraso de menos de 2 segundos.

A Figura 7.56b, mostra um diferença de aproximadamente 5cm na posição linear y desejada com um atraso de menos de 2 segundos.

A Figura 7.56c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

Lissajous

Controle P

1. Robô Desenvolvido

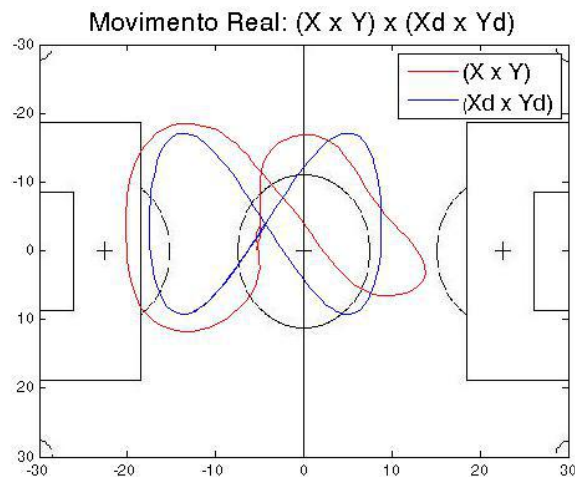


Figura 7.57: Movimento Real do robô desenvolvido na curva de Lissajous utilizando o controle P.

A Figura 7.21 mostra o resultado obtido para uma curva de Lissajous utilizando o controle P para o robô desenvolvido. Lembrando que este robô possui 10cm de diâmetro e sua posição é calculada a partir de seu centro.

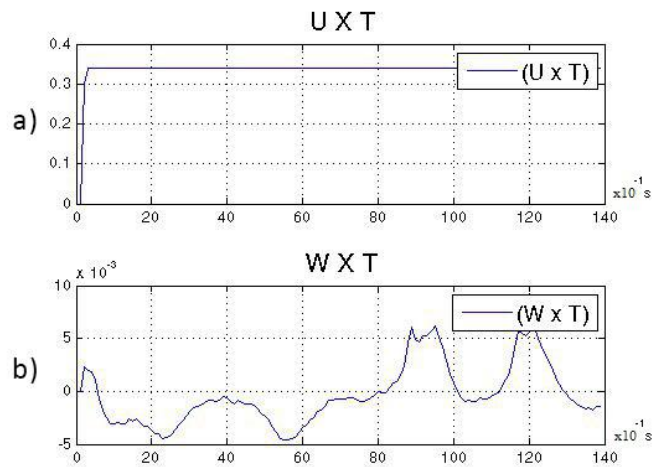


Figura 7.58: Sinais de Controle do robô desenvolvido na curva de Lissajous utilizando o controle P.

A Figura 7.58a, mostra que o sinal de controle linear variou de 0 à 0.35 e depois se manteve constante com valor de aproximadamente 0.35 durante o resto da trajetória.

A Figura 7.58b, mostra uma variação de aproximadamente 10×10^{-3} no sinal de controle angular.

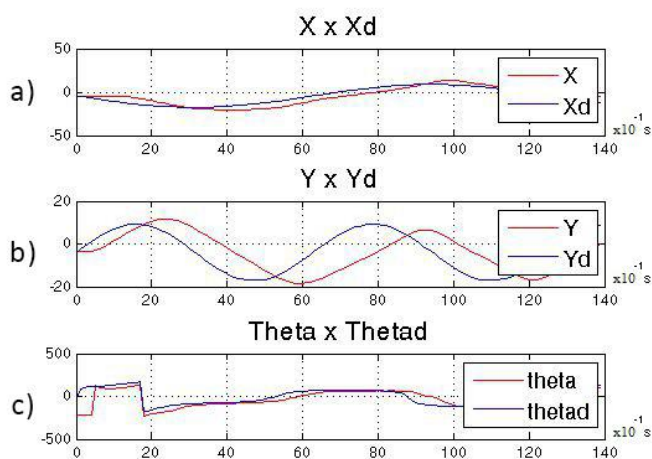


Figura 7.59: Posições linear e angular do robô desenvolvido na curva de Lissajous utilizando o controle P.

A Figura 7.59a, mostra uma diferença e atraso muito pequenos para a posição x desejada.

A Figura 7.59b, mostra um diferença de menos de 10cm na posição linear y desejada com um atraso de menos de 2 segundos.

A Figura 7.59c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

2. Robô Adquirido

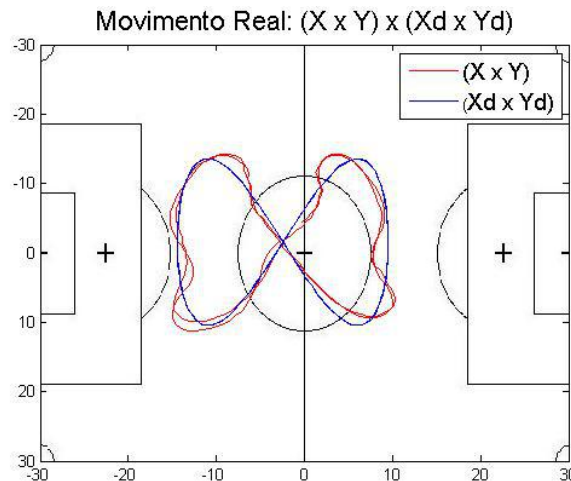


Figura 7.60: *Movimento Real do robô adquirido na curva de Lissajous utilizando o controle P.*

A Figura 7.60 mostra o resultado obtido para uma curva de Lissajous utilizando o controle P para o robô adquirido. Lembrando que este robô possui 8cm de diâmetro e sua posição é calculada a partir de seu centro.

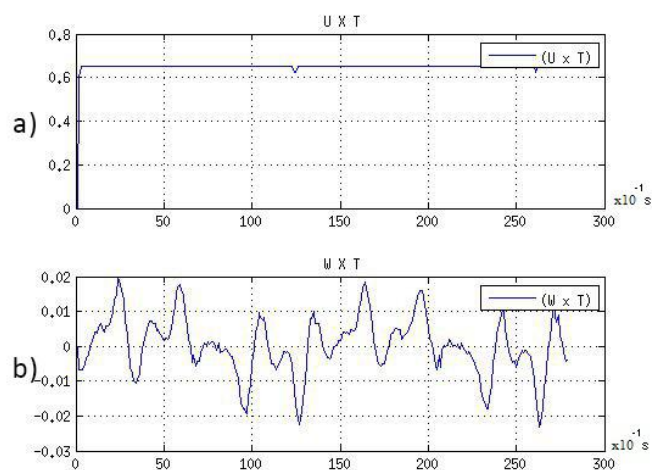


Figura 7.61: *Sinais de Controle do robô adquirido na curva de Lissajous utilizando o controle P.*

A Figura 7.61a, mostra que o sinal de controle linear variou de 0 à 0,65 e depois se

mantve constante com valor de aproximadamente 0.65 durante o resto da trajetória. A Figura 7.61b, mostra uma variação de aproximadamente 0.04 no sinal de controle angular.

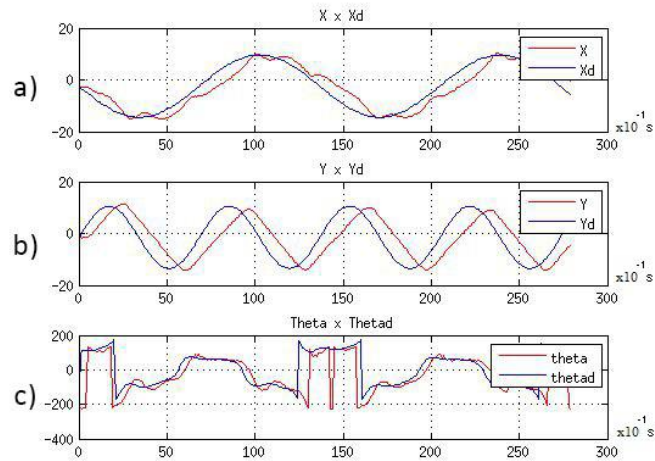


Figura 7.62: Posições linear e angular do robô adquirido na curva de Lissajous utilizando o controle P.

A Figura 7.62a, mostra uma diferença e atraso muito pequenos para a posição x desejada.

A Figura 7.62b, mostra uma diferença muito pequena da posição linear y desejada com um atraso de menos de 2 segundos.

A Figura 7.62c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

Controle PD

1. Robô Desenvolvido

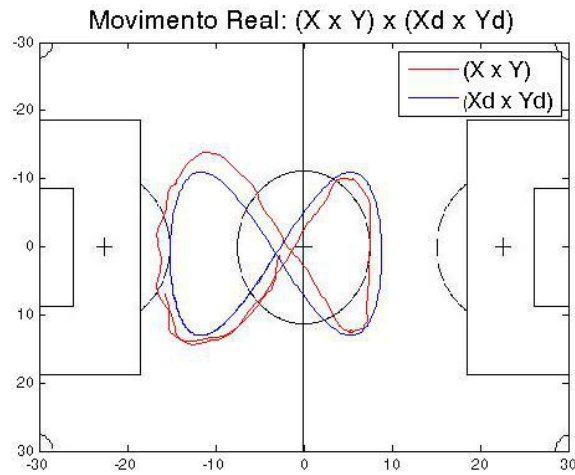


Figura 7.63: *Movimento Real do robô desenvolvido na curva de Lissajous utilizando o controle PD.*

A Figura 7.63 mostra o resultado obtido para uma curva de Lissajous utilizando o controle PD para o robô desenvolvido. Lembrando que este robô possui 10cm de diâmetro e sua posição é calculada a partir de seu centro.

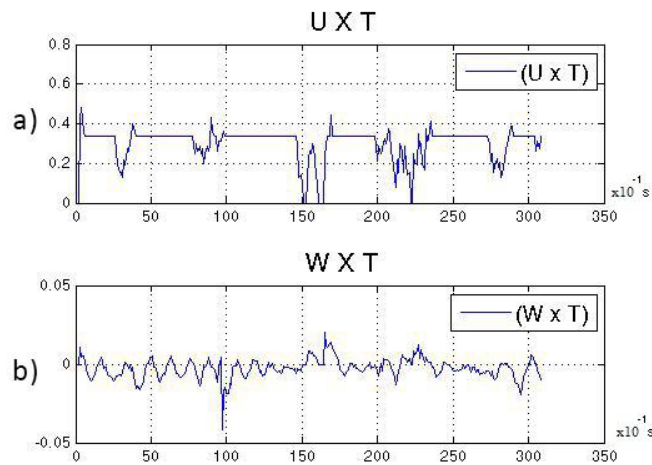


Figura 7.64: *Sinais de Controle do robô desenvolvido na curva de Lissajous utilizando o controle PD.*

A Figura 7.64a, mostra que o sinal de controle linear variou de 0 à 0.4 durante toda a trajetória.

A Figura 7.64b, mostra uma variação de aproximadamente 0.01 no sinal de controle angular.

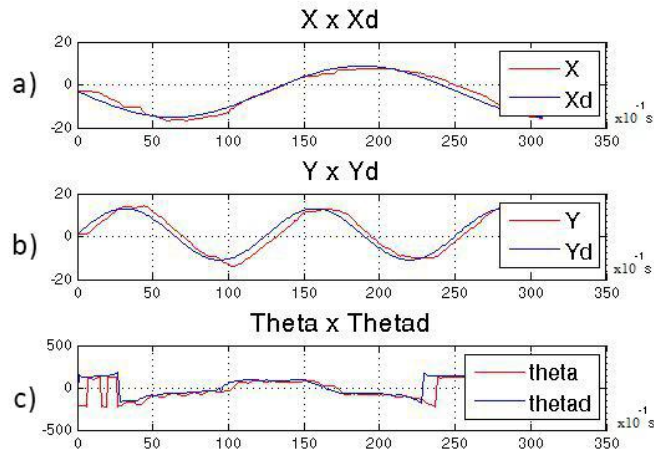


Figura 7.65: Posições linear e angular do robô desenvolvido na curva de Lissajous utilizando o controle PD.

A Figura 7.65a, mostra uma diferença e atraso muito pequenos para a posição x desejada.

A Figura 7.65b, mostra uma diferença e atraso muito pequenos para a posição y desejada.

A Figura 7.65c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

2. Robô Adquirido

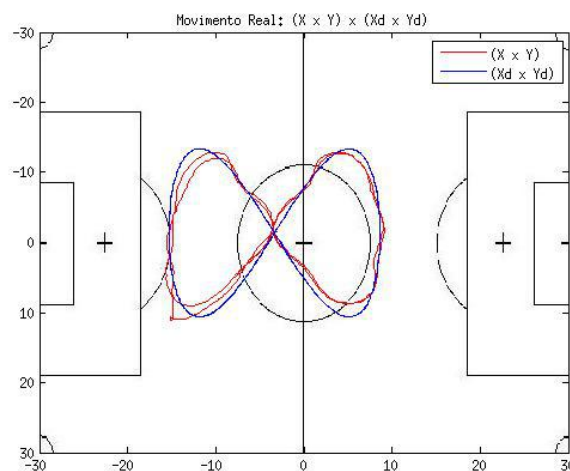


Figura 7.66: Movimento Real do robô adquirido na curva de Lissajous utilizando o controle PD.

A Figura 7.66 mostra o resultado obtido para uma curva de Lissajous utilizando o controle PD para o robô adquirido. Lembrando que este robô possui 8cm de diâmetro e sua posição é calculada a partir de seu centro.

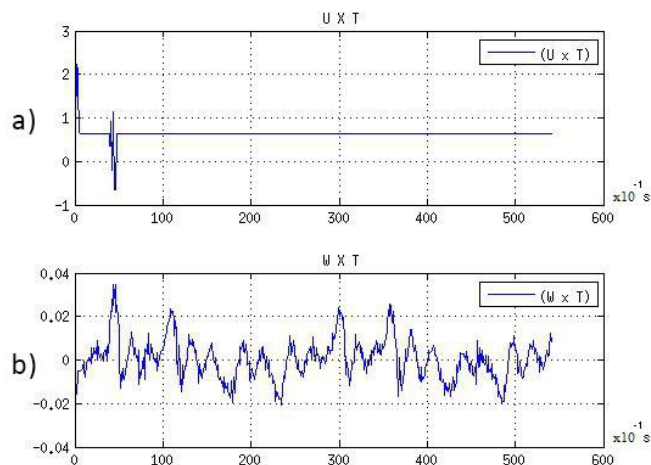


Figura 7.67: *Sinais de Controle do robô adquirido na curva de Lissajous utilizando o controle PD.*

A Figura 7.67a, mostra que o sinal de controle linear variou de 2 à 0.8, uma perturbação no quinto segundo e depois se manteve constante em 0.8.

A Figura 7.67b, mostra uma variação de aproximadamente 0.06 no sinal de controle angular.

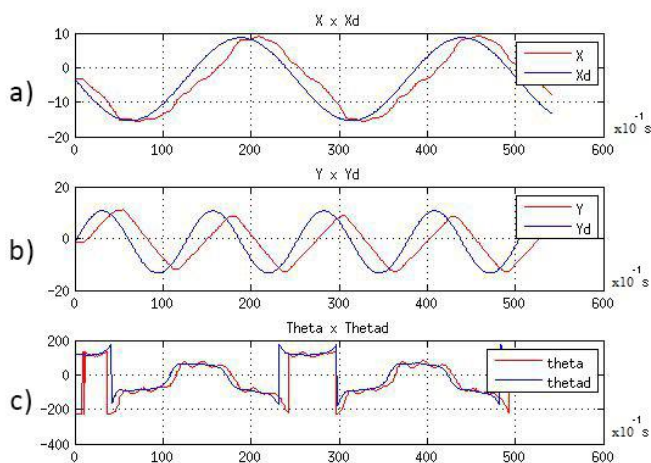


Figura 7.68: *Posições linear e angular do robô adquirido na curva de Lissajous utilizando o controle PD.*

A Figura 7.68a, mostra uma diferença e atraso muito pequenos para a posição x desejada.

A Figura 7.68b, mostra um diferença de menos de 10cm na posição linear y desejada com um atraso de menos de 2 segundos.

A Figura 7.68c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

Controle PID

1. Robô Desenvolvido

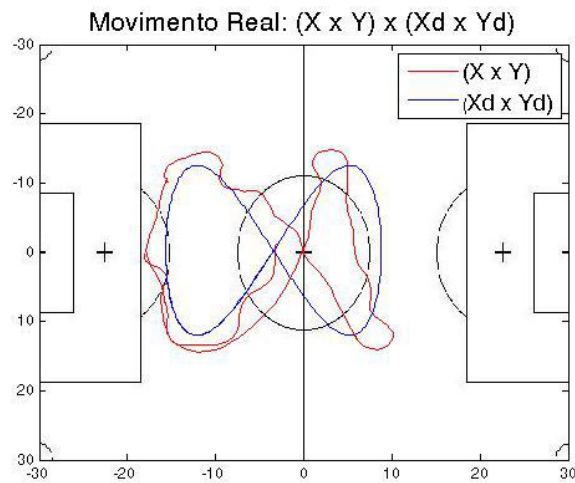


Figura 7.69: *Movimento Real do robô desenvolvido na curva de Lissajous utilizando o controle PID.*

A Figura 7.69 mostra o resultado obtido para uma curva de Lissajous utilizando o controle PID para o robô desenvolvido. Lembrando que este robô possui 10cm de diâmetro e sua posição é calculada a partir de seu centro.

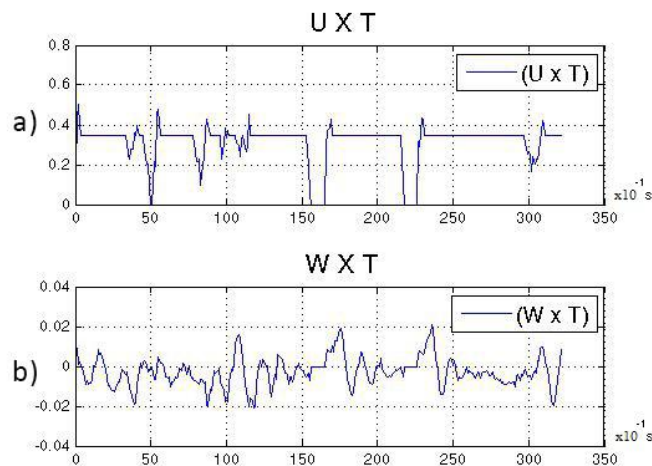


Figura 7.70: *Sinais de Controle do robô desenvolvido na curva de Lissajous utilizando o controle PID.*

A Figura 7.70a, mostra que o sinal de controle linear variou entre 0.5 e 0 durante toda a trajetória.

A Figura 7.70b, mostra uma variação de aproximadamente 0.04 no sinal de controle angular.

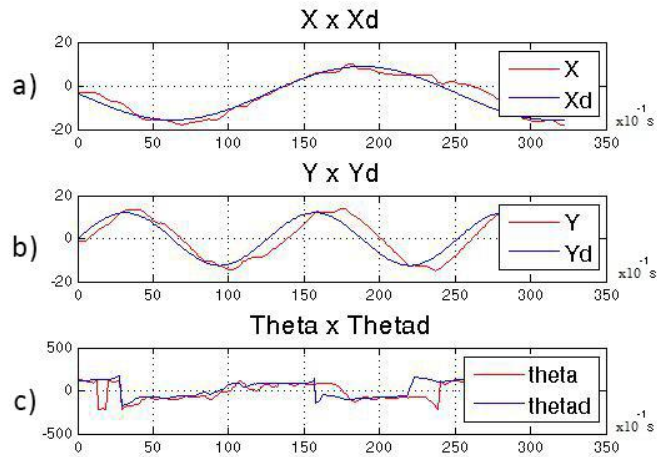


Figura 7.71: Posições linear e angular do robô desenvolvido na curva de Lissajous utilizando o controle PID.

A Figura 7.71a, mostra uma diferença e atraso muito pequenos para a posição x desejada.

A Figura 7.71b, mostra um diferença de menos de 5cm na posição linear y desejada com um atraso de menos de 1 segundo.

A Figura 7.71c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

2. Robô Adquirido

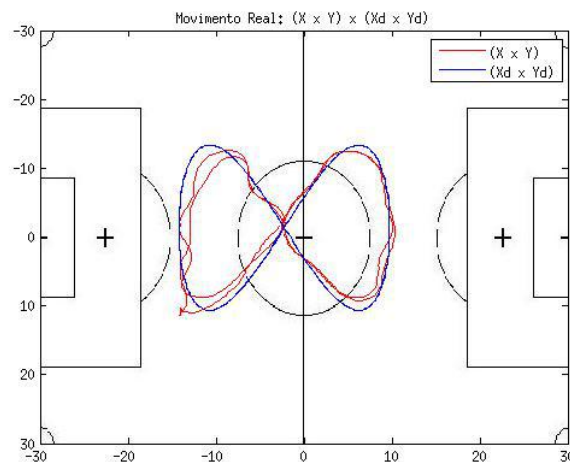


Figura 7.72: Movimento Real do robô adquirido na curva de Lissajous utilizando o controle PID.

A Figura 7.72 mostra o resultado obtido para uma curva de Lissajous utilizando o controle PID para o robô adquirido. Lembrando que este robô possui 10cm de diâmetro e sua posição é calculada a partir de seu centro.

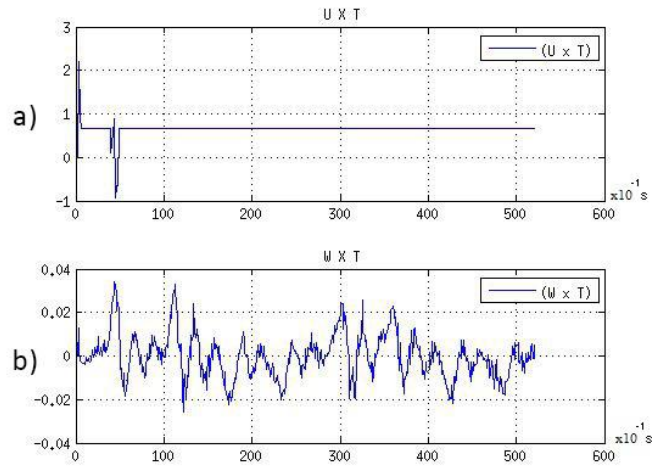


Figura 7.73: Sinais de Controle do robô adquirido na curva de Lissajous utilizando o controle PID.

A Figura 7.73a, mostra que o sinal de controle linear variou de 2 à 0.8, apresentou uma perturbação no quinto segundo e depois se manteve constante com valor de aproximadamente 0.8 durante o resto da trajetória.

A Figura 7.73b, mostra uma variação de aproximadamente 0.06 no sinal de controle angular.

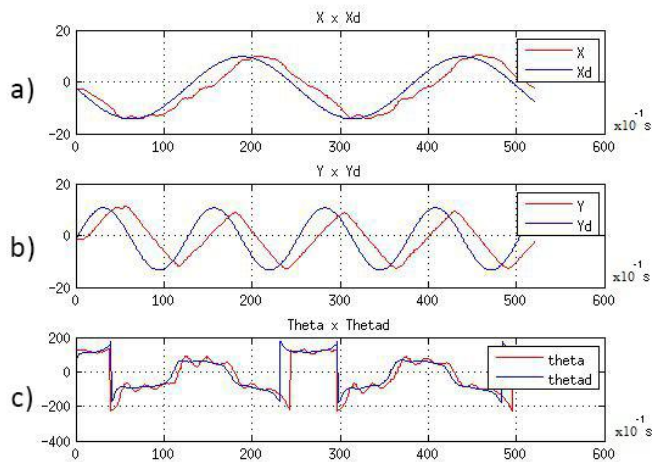


Figura 7.74: Posições linear e angular do robô adquirido na curva de Lissajous utilizando o controle PID.

A Figura 7.74a, mostra uma diferença e atraso muito pequenos para a posição x desejada.

A Figura 7.74b, mostra um diferença de menos de 10cm na posição linear y desejada com um atraso de menos de 2 segundos.

A Figura 7.74c, mostra uma pequena diferença em torno da orientação desejada durante a trajetória.

Conclusões sobre Movimentos de Zagueiro e Atacante

Observando os resultados, pode-se concluir novamente que o controle puramente proporcional não atinge de maneira satisfatória o objetivo de seguir a trajetória. Isso fica claro principalmente nas figuras 7.39 e 7.57 onde o robô desenvolvido fica bem longe da trajetória desejada.

Ao introduzir o elemento derivativo, a trajetória passa a ser seguida de maneira mais próxima como visto principalmente nas figuras 7.45 e 7.63 para o robô desenvolvido e nas figuras 7.48 e 7.66 para o robô adquirido.

O elemento integral novamente provoca mais problemas ao sistema do que ajustes de trajetória de maneira similar ao acontecido na movimentação de defesa, sendo este um dos motivos pelos quais pode-se considerar o controlador PD o mais eficiente novamente.

Conclusões Gerais sobre o Controle de Movimentação

Levando em consideração as três conclusões parciais, observamos que:

1. O robô adquirido, apesar de possibilitar uma acurácia maior para o seguimento de trajetórias em relação ao desenvolvido, é mais lento
2. O robô adquirido necessita de sinais de controle mais elevados para realizar sua movimentação quando comparado ao desenvolvido
3. O controlador do tipo PD se mostrou o mais eficiente dentre todas as configurações testadas
4. O fator integral em diversas situações introduziu instabilidades no sistema que tiveram que ser contornadas com técnicas para que o fator não ultrapassasse certos níveis
5. O controle desenvolvido atinge de maneira satisfatória o objetivo de seguir trajetórias tanto com o robô desenvolvido quanto com o robô adquirido

Capítulo 8

Conclusões, Contribuições e Trabalhos Futuros

Conclusões

Em relação aos simuladores desenvolvidos, observou-se que tanto o controle PD quanto o controle de linearização por realimentação são abordagens válidas e eficientes para o de rastreamento de trajetórias em futebol de robôs. Porém, constatou-se que as singularidades do controlador de linearização por realimentação precisaram ser contornadas com saturação para não desestabilizar o sistema, embora ele possa apresentar resultados superiores em faixas onde não ocorre singularidade.

Já para os resultados experimentais obtidos para as diferentes combinações de controle PID, observou-se que o robô adquirido é mais preciso para seguir as trajetórias, porém é mais lento. Os sinais de controle são mais elevados para o robô adquirido quando comparado ao desenvolvido, devido à diferença de dimensionamento dos motores e do peso dos robôs. Entre as combinações de PID testadas, o controlador PD apresentou os melhores resultados para as diversas situações propostas, além de se observar que o fator integral introduziu instabilidades no sistema.

Pode-se concluir ainda, que o robô desenvolvido tem resultados suficientemente bons mesmo quando comparado à um robô adquirido com aplicações específicas para o objetivo de rastreamento de trajetória. Este fato mostra que, com algumas melhorias no projeto, o robô teria grandes chances de sucesso em ambientes competitivos.

Contribuições

- Foi realizada uma revisão bibliográfica sobre o futebol de robôs, assim como diferentes métodos de controle de robôs móveis que podem ser aplicados ao futebol.
- Foram descritos não só métodos de controle que é o foco deste trabalho, mas também todas as áreas relacionadas ao funcionamento do robô, como visão e comunicação

- Foi montada uma bancada experimental que poderá ser utilizada para projetos futuros no laboratório.
- Foi projetado e desenvolvido o robô.
- Foram desenvolvidos dois simuladores para o controle de robôs móveis que também poderão ser utilizados no laboratório.
- Foi desenvolvido um sistema de aquisição e armazenamento de dados entre o software e o Matlab.

Trabalhos Futuros

Um dos trabalhos futuros seria a implementação do controle de linearização por realimentação abordado neste projeto tanto para o robô desenvolvido quanto para o adquirido, de forma a obter mais uma opção de controle para robôs móveis e comparar sua resposta aos resultados existentes.

Implementar no programa a possibilidade de controlar mais de um robô ao mesmo tempo, podendo assim utilizar formações específicas para cada jogador durante uma partida.

Implementação da visualização dos níveis de bateria de cada robô para que se possa acompanhar em tempo real e saber quando será necessário recarregar o robô.

Construção de mais robôs para testes e simulações em formações de robôs em uma partida futebol.

Aplicar em ambiente competitivo as técnicas e controles desenvolvidos neste projeto.

Referências Bibliográficas

- [1] <http://www.cbrobotica.org>. [Online; acessado 10-Outubro-2016].
- [2] <http://www.robocup.org.br>. [Online; acessado 10-Outubro-2016].
- [3] <http://www.fira.net>. [Online; acessado 10-Outubro-2016].
- [4] BALLARD, D. H., BROWN, C. M. *Computer Vision*. New Jersey: Prentice Hall, 1982.
- [5] LUCCHESI, L., MITRA, S. K. “Color image segmentation: A state-of-the-art survey”, 2001.
- [6] SAPIRO, G. “Color snakes”, .
- [7] ITOH, S., MATSUDA, I. “Segmentation of colour still images using Voronoi diagrams”, *Proc. Of the 8th European signal processing conference (eusipco-96)*, 1996.
- [8] HARRIS, C., STEPHENS, M. “A combined corner and edge detection”. In: *Proc. of the Fourth Alvey Vision Conference*, pp. pp. 147–151, 1988.
- [9] KADIR, T., BRADY, M. . “Saliency, scale and image description”, *International Journal of Computer Vision*, v. 45, 2, pp. pp. 83–105, 2001.
- [10] LOWE, D. “Distinctive image features from scale-invariant keypoints”, *International Journal of Computer Vision*, v. 60, 2, pp. pp. 91–110, 2004.
- [11] BAY, H., TUYTELAARS, T., VAN GOOL, L. “SURF : Speeded-Up Robust Features”. In: *European Conference on Computer Vision (ECCV)*, 2006.
- [12] PHAN, T., SOHONI, S., LARSON, E. C., CHANDLER, D. M. “Performance-analysis-based acceleration of image quality assessment”, *IEEE Southwest Symposium on Image Analysis and Interpretation*, 2012.
- [13] SHAFAIT, F., KEYSERS, D., M. BREUEL, T. “Efficient implementation of local adaptive thresholding techniques using integral images”, *Electronic Imaging*, 2008.

- [14] ITTI, L., KOCH, C. “Computational modeling of visual attention”, *Nature Reviews Neuroscience*, v. 2, 3, pp. pp.194–203, 2001.
- [15] ITTI, L., KOCH, C., NIEBUR, E. “A model of saliency-based visual attention for rapid scene analysis”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 20, 11, pp. pp. 1254–1259., 1998.
- [16] BUTKO, N. J., ZHANG, L., COTTRELL, G. W., MOVELLAN, J. R. “Visual Saliency Model for Robot Cameras”, *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. pp. 2398–2403, 2008.
- [17] ZHANG, L., TONG, M. H., MARKS, T. K., SHAN, H., COTTRELL, G. W. “SUN: A Bayesian Framework for Saliency Using Natural Statistics”, *Journal of vision*, 2008.
- [18] RUSU, R. B., BLODOW, N., MARTON, Z. C., BEETZ, M. “Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments”, *IEEE/RSJ Intelligent Robots and Systems (IROS)*, pp. pp. 1–6, 2009.
- [19] FILLIAT, D., BATTESTI, E., BAZEILLE, S., DUCEUX, G., GEPPERTH, A., HARRATH, L., JEBARI, I., PEREIRA, R., TAPUS, A., MEYER, C., IENG, S., BENOSMAN, R., CIZERON, E., MAMANNA, J.-C., POTHIER, B. “Rgbd object recognition and visual texture classification for indoor semantic mapping.” In: *Proceedings of the 4th International Conference on Technologies for Practical Robot Applications (TePRA)*, 2012.
- [20] BRADSKI, G., KAEBLER, A. *Learning OpenCV*. 1 ed. New York, O’Reilly Media, 2008.
- [21] IGOE, T. *Making Things Talk - Practical Methods for Connecting Physical Objects*. Maker Media, Inc, 2007.
- [22] C. CANUDAS DE WIT, H. B., NIJMEIJER, H. “Practical stabilization of nonlinear systems in chained form”, *IEEE Conference on Decision and Control*, v. 4, pp. pp. 3475–3480, 1994.
- [23] W. OELEN, H. BERGHUIS, H. N., WIT, C. C. D. “Hybrid stabilizing control on a real mobile robot.” *IEEE Robotics and Automation Magazine*, v. 2,2, pp. pp. 16–23, 1995.
- [24] T. FUKAO, H. N., ADACHI, N. “Adaptive tracking control of a nonholonomic mobile robot”, *IEEE Transactions on Automatic Control*, v. 16,5, pp. pp. 609–615, 2000.

- [25] T. C. LEE, C. H. L., TENG, C. C. “Adaptive tracking control of nonholonomic mobile robots by computed torque.” *In Proceedings of the 38th IEEE American Conference on Decision and Control*, pp. pp. 1254–1259, 1999.
- [26] M. OYA, C. H. Y., KATOH., R. “Robust adaptive motion/force tracking control of uncertain nonholonomic mechanical systems.” *IEEE Transactions on Robotics and Automation*, v. 19,1, pp. pp. 175–181, 2003.
- [27] SUN., S. “Designing approach on trajectory-tracking control of mobile robot.” *Robotics and Computer Integrated Manufacturing*, v. 21,1, pp. pp. 81–85, 2005.
- [28] N. MARTINS, F., C. CELESTE, W., CARELLI, R., SARCINELLI-FILHO, M., F. BASTOS-FILHO, T. “An adaptive dynamic controller for autonomous mobile robot trajectory tracking”, *Elsevier*, 2007.
- [29] CHIH-YANG, C., TZUU-HSENG S., L., YING-CHIEH, Y., CHA-CHENG, C. “Design and Implementation of an Adaptive sliding-mode dynamic controller for wheeled mobile robots”, *Elsevier*, 2007.
- [30] KLANCAR, G., SKRJANC, I. “Tracking-error model-based predictive control for mobile robots in real time”, *Elsevier*, 2007.
- [31] G. ORIOLO, A. D. L., VENDITTELLI., M. “Wmr control via dynamic feedback linearization: design, implementation and experimental validation.” *IEEE Transactions on Control Systems Technology*, v. 10,6, pp. pp. 835–852, 2002.
- [32] BORENSTEIN, J., E. H., FENG, L. *Where am I ? Sensors and Methods for Mobile Robot Positioning*. Universidad de Michigan, USAI, 1996.
- [33] GONZÁLEZ JIMÉNEZ, J Y OLLERO BATURONE, A. *Estimación de la posición de un robot móvil*. Asociación Española de Informática y Automática, 1996.
- [34] <https://www.pololu.com/product/975>, 2016. [Online; acessado em 10-Outubro-2016].
- [35] <http://www.atmel.com/Images/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-3285-Datasheet.pdf>. [Online; acessado 07-Junho-2016].

Apêndice A

Apêndice

A.1 Regras do Futebol de Robôs

Regras retiradas do site da CBR [1].

1. Campo e Bola

- (a) Especificações do Campo: é composto por uma chapa plana e rígida de madeira medindo $150\text{cm} \times 130\text{cm}$, em cor preta fosca (não reflexiva) com laterais medindo 5cm de altura por 2.5cm de largura. A parte superior das laterais deverá ser na cor preta (a mesma cor do campo) e, as paredes deverão ser na cor branca. Quatro triângulos isósceles sólidos com $7\text{cm} \times 7\text{cm}$ são fixados nos cantos do campo, para evitar que a bola fique presa nestas regiões. A textura do campo é similar a de uma mesa de ping-pong.
- (b) Marcações do Campo: O campo tem marcações como as que podem ser vistas na figura A.1.

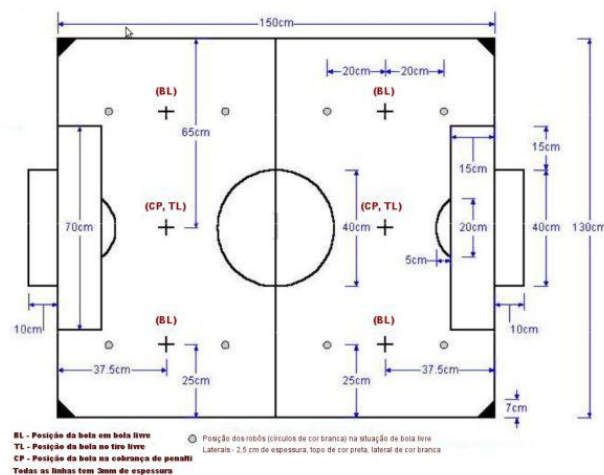


Figura A.1: Marcações do Campo

O círculo central tem 20cm de raio. O arco, que faz parte da área do gol, tem 20cm junto a linha do gol e 5cm na direção perpendicular. As linhas principais (linha central, bordas da área do gol e do círculo central) deverão ser na cor branca, com 3mm de espessura. As marcações de bola livre são na cor cinza.

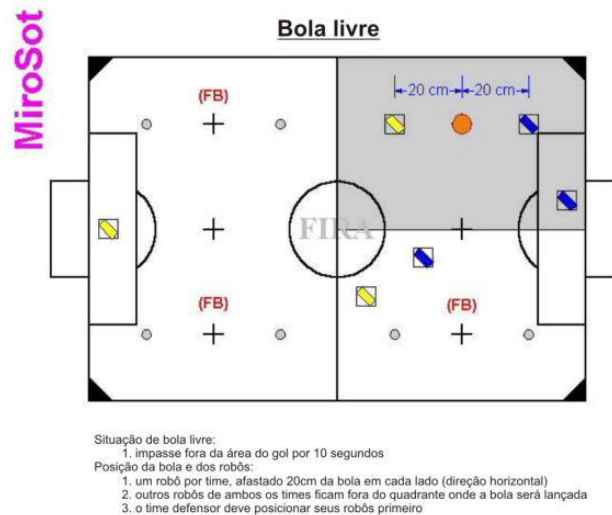


Figura A.2: *Bola Livre*

O árbitro declarará bola livre quando um impasse ocorrer por 10 segundos fora de qualquer uma das duas área de gol. Neste caso, os robôs poderão somente serem realocados por meios próprios ou pelo árbitro, e não por um integrante do time. Quando uma bola livre é declarada dentro de alguma quarta parte (quarto) do campo, os robôs devem parar e a bola será posicionada pelo árbitro em posição relevante para cobrança da bola livre(FB). O árbitro deve, também manualmente, realocar todos os robôs que estão a menos de 20cm de distância da marca de bola livre, movendo longitudinalmente os robôs em direção ao próprio campo, até que eles fiquem alinhados com o ponto de bola livre(círculos cinzas), localizados a 20cm da posição da bola na direção longitudinal no campo. Depois disso, um (ou nenhum) robô de cada time poderá automaticamente posicionar-se no ponto relevante para bola livre (círculo cinza) mais próximo possível do seu próprio gol. Os outros robôs (de ambos os times) deverão permanecer onde eles estiverem. O jogo deverá reiniciar quando o árbitro der o sinal, então todos os robôs podem mover-se livremente.

- (c) Gol: O gol tem 40cm × 10cm. Postes e redes não devem ser utilizadas.
- (d) Linha do Gol e Área do Gol: A linha do gol é a linha em frente ao gol e tem 40cm de largura. A área do gol compreende a área de um retângulo (medindo 70cm × 15cm, na frente do gol) e o arco associado (20cm junto a linha do gol

e 5cm na direção perpendicular).

- (e) Bola: Deverá ser utilizada uma bola de golfe laranja, com 42.7mm de diâmetro e 46g de massa.
- (f) Localização do Campo: O campo deve estar localizado em ambiente interno (i.e., coberto).
- (g) Condições Luminosas: A intensidade luminosa na competição deve ser fixa, em valores próximos a 1.000Lux.

2. Jogadores

- (a) Sistema Completo: Uma partida deve ser disputada por dois times, cada qual constituído por três robôs. Um dos robôs pode ser o goleiro. Somente três pessoas de cada equipe devem ficar no local da competição. Apenas um único computador por equipe pode ser usado, sobretudo dedicado ao processamento visual e para reconhecimento de posições.

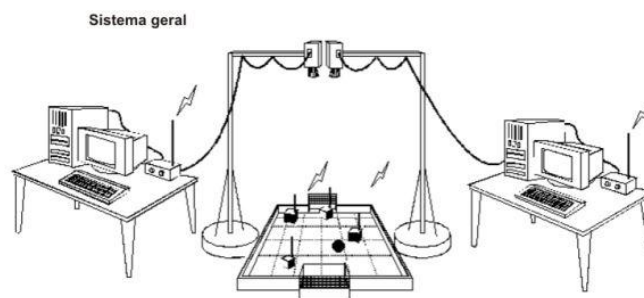


Figura A.3: *Sistema Geral*

- (b) Robôs: O tamanho de cada robô deve ser limitado a $7.5\text{cm} \times 7.5\text{cm} \times 7.5\text{cm}$. A altura da antena de radio frequência não deve ser considerada neste limite. A parte superior dos robôs não deve conter as cores laranja, branco ou cinza e também não deve ser colorida com mais de duas cores diferentes da cor preto. Uma cor, que pode ser azul ou amarelo, a ser definido pelos organizadores, identificará times distintos. Todos os robôs devem possuir uma região visível na parte superior. Uma etiqueta azul ou amarela, definida pelos organizadores, irá identificar os robôs de cada time. Todos os robôs devem ter visivelmente no topo, uma região sólida da etiqueta do time, seja ela na cor azul ou amarelo. Essa região pode ser de qualquer forma, mas deve ser capaz de conter (pelo menos) um quadrado com 3.5cm de lado ou um círculo com 4cm de diâmetro. A cor de identificação de cada time poderá mudar de partida a partida, portanto a etiqueta usada deverá ser destacável. Quando os times estiverem com as etiquetas com as cores (azul ou amarelo) definidas, os robôs não devem

ter nenhuma etiqueta da cor do time adversário. Observação: É recomendado que os times prepararem, no mínimo, 2 etiquetas azuis e 2 etiquetas amarelas. É recomendado, também, que os times prepararem no mínimo 6 etiquetas diferentes além da azul e amarela, para identificação individual de cada robô. Os robôs podem vestir uniformes, que deve ser limitado a $8\text{cm} \times 8\text{cm} \times 8\text{cm}$. Um robô dentro da área do próprio gol deverá ser considerado como "goleiro". O papel de goleiro pode ser trocado dinamicamente durante a disputa, assim um robô dentro de sua própria área sempre será considerado como o goleiro. Ao robô goleiro será permitido pegar ou segurar a bola somente dentro da área do próprio gol. Cada robô deve ser totalmente independente, com baterias e motores contidos em si mesmo. Somente comunicação sem fio deverá ser permitida para todos os tipos de interações entre o computador "host" e um robô. Os robôs podem ser equipados com braços, pernas, etc., mas devem respeitar as limitações de tamanho, mesmo depois dos equipamentos estarem totalmente expandidos. Nenhum dos robôs, exceto o designado como goleiro, serão permitidos a pegar ou segurar a bola como um goleiro de forma a encobrir mais do que 30% da bola, em vista superior.

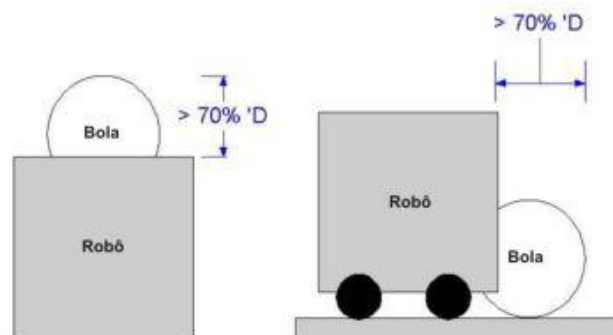


Figura A.4: Encobrimento da Bola

Enquanto uma partida estiver em progresso, a qualquer momento em que o árbitro apitar, o operador humano deverá parar todos os robôs usando a comunicação entre os robôs e o computador "host".

- (c) Substituições: Somente duas substituições são permitidas enquanto o jogo estiver em progresso. No meio tempo, qualquer quantidade de substituições podem ser feitas. Quando uma substituição é necessária enquanto o jogo estiver em progresso, um representante do time interessado deverá requisitar um intervalo ao árbitro e, este irá parar o jogo em um momento apropriado. O jogo irá recomeçar com todos os robôs e bola posicionados nas mesmas posições que estavam ocupando no momento da interrupção do jogo.
- (d) Intervalos: Um representante do time interessado (no intervalo) deverá requi-

sitar um intervalo ao árbitro. Cada time terá direito a 2 intervalos (durante todo o jogo), cada um com duração de máxima de 2 minutos.

3. Transmissão de Informação: Os integrantes humanos de um time podem transmitir certos comandos do computador "host" para seus robôs somente com autorização do árbitro ou quando o jogo não estiver em progresso. Não é permitido transmitir comandos tais como: sinais para parar qualquer um dos robôs ou, ainda, sinais de reinicialização, sem a permissão do árbitro. Qualquer outra informação, como estratégia de jogo, pode ser comunicada para os robôs somente quando um jogo não está em progresso. Um operador humano não deverá controlar diretamente o movimento de seus robôs por comandos via teclado, nem mesmo com um joystick, sob quaisquer circunstâncias. Enquanto o jogo estiver em progresso o computador "host" pode mandar qualquer informação autonomamente.
4. O Sistema de Visão: Para identificar o robô e a bola no cenário, um sistema de visão pode ser usado. A câmera (ou outro sistema de sensoriamento) deverá ser fixada acima do próprio meio do campo, incluindo sua linha central, de tal forma que não tenha que ser movida depois da troca de lado, após o meio tempo. Se ambos os times desejarem manter suas câmeras acima do círculo central do cenário, elas deverão ser posicionadas lado a lado, equidistante da linha central e o mais perto possível uma da outra. A câmera deverá estar suspensa a uma altura não inferior a 2 metros.
5. Duração do Jogo: Cada partida terá 2 períodos iguais, cada qual com 5 minutos, com um intervalo de meio tempo de 10 minutos. Um relógio oficial contará o tempo de substituições, de substituições de robôs danificados, do intervalo e durante situações que se considerem ser necessário registrar um período de tempo qualquer. Se um time não está pronto para voltar ao jogo depois do meio tempo, 5 minutos adicionais deverão ser permitidos. Se depois do tempo adicional permitido, um time não estiver pronto para começar o jogo, esse time será desqualificado do jogo.
6. Começo de Jogo: Antes de se começar um jogo, a cor do time ou a posse inicial de bola serão decididos utilizando-se uma moeda (cara ou coroa). O time que ganhar deverá escolher a cor de identificação ou a bola. No começo do jogo, ao time atacante (i.e., aquele que tem a posse de bola) é permitido posicionar seus robôs livremente em sua própria área e dentro do círculo central. O time defensor pode posicionar livremente seus robôs em sua própria área, exceto dentro do círculo central. No começo do primeiro e segundo tempo e depois de um gol ser feito, a bola deverá ser posicionada dentro do círculo central. Com um sinal do árbitro, o jogo deverá ser recomeçado e todos robôs podem mover-se livremente. No começo do jogo ou depois de um gol ser feito, o jogo deverá ser iniciado/continuado, com as

posições dos robôs descritas acima. Depois do meio tempo, os time devem trocar seus lados no campo.

7. Método de Pontuação:

- (a) O Vencedor Um gol deverá ser computado quando a bola inteira passar por cima da linha do gol. O vencedor de um jogo deverá ser escolhido com base no número de gols computados (saldo).
- (b) O Desempate: No caso de um empate depois do segundo tempo e se as regras da competição não permitem um empate acontecer nessa disputa específica, o ganhador será decidido no esquema de morte súbita. O jogo será continuado após um intervalo de 5 minutos, prorrogáveis por mais 3 minutos. O time que ganhar o primeiro jogo será declarado como ganhador. Se o empate continuar mesmo depois dos 3 minutos de prorrogação, o vencedor será decidido nos pênaltis. Cada time tem direito a três cobranças de pênaltis, que difere da Regra de Pênalti permitindo que somente um cobrador(robô que irá chutar) e um goleiro estejam presentes no campo. O goleiro deverá ser mantido dentro da área de seu gol e a posição do cobrador e da bola deverão estar em acordo com a Regra de Pênalti. Depois do apito do árbitro , o goleiro pode sair da área do gol. Em caso de empate depois das 3 cobranças de pênaltis de cada time, cobranças adicionais deverão ser feitas uma por uma, até que o vencedor seja decidido. Todas as cobranças de pênaltis devem ser feitas por um robô, depois do apito do árbitro. Um chute de pênalti será completado, quando uma das seguintes situações acontecer:
 - i. O goleiro pega a bola com algum de seus equipamentos(se tiver algum) na área do gol.
 - ii. A bola sai fora da área do gol.
 - iii. Depois de 30 segundos contados a partir do apito do árbitro.

8. Faltas: Uma falta pode acontecer nos seguintes casos: Colidindo com um robô do time adversário: o juiz registrará faltas que afetem diretamente o andamento da partida ou que aparentem ter potencial para danificar robôs do time adversário. Quando um robô defensor empurrar um robô oponente que esteja sem a bola, um tiro livre será dado para o time oponente se o juiz considerar que a falta é irrelevante para a partida. É permitido empurrar a bola e o robô adversário, contanto que a bola esteja entre ambos. É permitido empurrar o robô goleiro na área do gol, se a bola estiver entre o goleiro e o robô que o está empurrando. Entretanto, empurrar o goleiro em direção ao gol, com a bola, não é permitido. Se um robô atacante empurrar o goleiro com a bola em direção ao gol ou empurrar o goleiro diretamente, o juiz deve declarar tiro de meta. Um time atacando com mais de um robô, na área

do gol do time adversário, deve ser penalizado por um tiro de meta batido pelo time do goleiro. Um robô é considerado estar na área do gol se mais de 50% dele estiver dentro dela, com a confirmação do juiz. Um time defendendo com mais de um robô na área do gol quando a bola está dentro da área do gol, deve ser penalizado com um pênalti, exceto se o time atacante marcar um gol. Se a bola entrar na área do gol quando houver dois jogadores de defesa, ou se um segundo jogador de defesa entrar na área do gol onde já estão goleiro e bola, o juiz deve penalizar o time defensor com um pênalti para o time adversário. Se a bola não está na área do gol, não há penalizações para os times com dois robôs de defesa dentro da sua área de gol. É chamado de manipulação ("handling"), como dito pelo juiz, quando um robô diferente do goleiro retém a bola. A bola é considerada retida quando a bola se junta a um robô de tal forma que nenhum outro robô é capaz de mover a bola. O time responsável será penalizado com um tiro livre se a infração ocorrer fora de sua área de gol, ou com um pênalti caso ocorra dentro de sua área de gol. Quando o goleiro tem a posse de bola sem disputá-la, ele deve chutá-la para fora de sua área de gol dentro de, no máximo, 10 segundos. Se não o fizer, será penalizado com um pênalti dado para o time adversário. Se um atacante disputar a bola com o goleiro durante este tempo, a contagem de tempo é reiniciada. Bloquear o goleiro do time adversário na sua respectiva área, acarretará em tiro livre, a ser cobrado pelo time do goleiro. Somente o juiz e um dos integrantes do time devem tocar nos robôs. Tocar nos robôs sem permissão do juiz deve ser penalizado com um pênalti.

9. Interrupções Durante o Jogo: O jogo deverá ser interrompido e os robôs devem ser reposicionados quando:

- (a) Um robô tem que ser substituído.
- (b) Um robô caiu de maneira a bloquear o gol.
- (c) Foi marcado um gol ou uma falta.
- (d) Juiz ordena chute livre, pênalti, tiro de meta ou bola livre.

Os robôs podem ser reposicionados por um operador humano ou por seus próprios meios (autonomamente). Quando o árbitro declarar bola livre, o reposicionamento por humanos não é permitido.

10. Tiro Livre: Quando um robô defensor empurrar um robô adversário, um chute livre será dado para o time adversário. A bola será posicionada na posição relevante para tiro livre(CL) no campo. O robô que irá chutar deverá ser posicionado atrás da bola. O time atacante pode posicionar seus robôs livremente no seu lado do campo. Os robôs defensores devem ser posicionados em contato com a área do gol ou em

qualquer lado do arco. Com o apito do árbitro, todos os robôs podem começar a mover-se livremente.



Figura A.5: *Tiro Livre*

11. Pênalti: Um pênalti será declarado nas seguintes situações

- Defendendo com mais de um robô na área do gol.
- Falha por parte do goleiro ao chutar a bola para fora da sua área do gol dentro de 10 segundos.
- Quando qualquer um dos integrantes humanos tocarem os robôs sem permissão do árbitro, enquanto o jogo estiver em progresso.

Quando o árbitro declarar um pênalti, a bola será posicionada na posição relevante para cobrança de pênalti(PK) no campo. O robô que cobrar o pênalti deverá ser colocado atrás da bola. Durante a cobrança, um dos lados do goleiro deve estar em contato com a linha do gol. O goleiro pode ser orientado em qualquer direção. Outros robôs deverão ser posicionados livremente dentro do outro lado a partir da linha do meio, mas o time adversário tem a preferência em posicionar seus robôs. O jogo deverá recomeçar normalmente(todos os robôs podem se mover livremente) depois do apito do árbitro. O robô que cobrar o pênalti pode driblar a bola ou chuta-lá.



Figura A.6: *Tiro Livre*

12. Tiro de Meta: O tiro de meta deverá ser declarado sob as seguintes situações

- (a) Quando um robô atacante empurrar o goleiro dentro da área de gol do mesmo, o árbitro deverá chamar o tiro de meta que o goleiro irá cobrar.
- (b) Atacar com mais de 1 robô dentro da área do gol do time adversário deverá ser penalizado com um tiro de meta, cobrado pelo time adversário.
- (c) Quando um robô adversário bloquear o goleiro na área do gol deste.
- (d) Quando o goleiro pegar a bola com seus equipamentos (se tiver algum) dentro da área do próprio gol.
- (e) Quando um impasse ocorre na área do gol durante, pelo menos, 10 segundos.

Durante o tiro de meta, somente ao goleiro será permitido permanecer dentro da área do gol, e a bola pode ser colocada em qualquer lugar dentro da área do gol. Outros robôs do time devem ser posicionados fora da área do gol durante o tiro de meta. O time atacante terá preferencia em posicionar seus robôs em qualquer lugar no campo, obedecendo-se a Regra de Faltas. O time defensor pode, então, posicionar seus robôs dentro do seu próprio lado no campo. O jogo deverá recomeçar com o apito do árbitro.



Figura A.7: *Tiro de Meta*

A.2 Categorias das Competições

1. Categorias da CBR e LARC:

Descrição das categorias retiradas do site da CBR [1].

- RoboCup Small- Size (F180) Na RoboCup Small Size Futebol, também conhecida como RoboCup F-180, duas equipas de 6 robôs de até 150mm de altura e até 180mm de diâmetro jogam uma partida de futebol. Os robôs podem ser controlados remotamente por um computador, ou pode-se usar o processamento a bordo.
- RoboCup Simulation 2D Neste campeonato simulado, duas equipes de 11 agentes inteligentes autônomos disputam uma partida de futebol em duas dimensões. Cada agente que representa um jogador recebe informações limitadas sobre a situação do jogo e precisa decidir o seu pensamento ação na equipe como um todo.
- RoboCup Simulation 3D A competição de simulação 3D aumenta o realismo do ambiente simulado utilizado em outras ligas de simulação, adicionando uma dimensão extra e física mais complexa. Em 2009, o objetivo da competição de simulação 3D foi deslocado desde a concepção de comportamentos estratégicos de jogar futebol em relação ao controle de baixo nível de robôs humanóides e da criação de comportamentos básicos como caminhar, chutar, virar e levantar-se, entre outros. Neste campeonato simulado duas equipes de 11 robôs autônomos humanóides joga um jogo de futebol em 3 dimensões.
- RoboCup Humanoid and Standard Platform League (SPL)

No Humanoid League, robôs autônomos com um corpo humano joga uma partida de futebol uns contra os outros. Os robôs são divididos em três classes de tamanho: KidSize (altura 40-90cm), TeenSize (altura 80-140cm) e Adult-Size (130-180 cm de altura). Nas equipes de competição KidSizeSOCCER de quatro, robôs autônomos altamente dinâmicos competem uns com os outros. Curta dinâmica, correr e chutar a bola, mantendo o equilíbrio, percepção visual da bola, outros jogadores, e o campo, auto-localização, e jogo de equipe estão entre as muitas questões de pesquisa investigadas no Humanoid League. A RoboCup Standard Platform League é um campeonato de futebol de robôs da liga RoboCup, em que todas as equipes competem com robôs idênticos. Os robôs funcionam plenamente autônomos, ou seja, não há nenhum controle externo, nem por seres humanos, nem por computadores. A plataforma atual padrão utilizado é o NAO humanóide pela Aldebaran Robotics.

- RoboCup Rescue Simulation Agents A competição de agentes envolve pontuação competindo algoritmos de coordenação de agentes em diferentes mapas da plataforma de simulação Robocup Rescue. O desafio, neste caso, envolve o desenvolvimento de algoritmos de coordenação que permitem às equipes de ambulâncias, forças policiais, e de bombeiros para salvar tantos civis quanto possível e extinguir incêndios em uma cidade onde um terremoto que acabou de acontecer.
- RoboCup @Home A liga RoboCup @ Home visa desenvolver tecnologia de assistência robótica e de serviços com alta relevância para futuras aplicações domésticas pessoais. É a maior competição anual internacional de robôs de serviços autônomos e faz parte da iniciativa RoboCup. Um conjunto de testes de benchmark é usado para avaliar habilidades e desempenho dos robôs em um ambiente doméstico sem padronização realista. O foco recai sobre os seguintes domínios, mas não está limitado a: interação e cooperação robô-homem, de navegação e mapeamento em ambientes dinâmicos, visão computacional e reconhecimento de objetos sob condições de luz natural, manipulação de objetos, comportamentos adaptativos, comportamento de integração, inteligência de ambiente, normalização e integração de sistemas.

Todo mundo com um robô autônomo pode participar. A liga @Home consiste de alguns testes e um desafio aberto para demonstrar as habilidades de seu robô. Para participar do desafio aberto você tem que participar em pelo menos um teste.

Os testes devem:

incluir a interação homem-máquina ser socialmente relevante ser pedido dirigido / orientada ser cientificamente desafiador ser fácil de configurar e de

baixo custo ser simples e ter regras auto-explicativo ser interessante assistir ter uma pequena quantidade de tempo

- RoboCup Junior RoboCupJunior é uma iniciativa educacional orientada para o projeto que patrocina eventos robóticos locais, regionais e internacionais para jovens estudantes. Ele é projetado para introduzir RoboCup a crianças de escolas primárias e secundárias, bem como estudantes que não têm os recursos para se envolverem nas ligas superiores ainda. O foco da Liga Júnior encontra-se na educação. O torneio oferece aos participantes a oportunidade de participar em programas de intercâmbio internacional e de compartilhar a experiência do encontro com colegas do exterior. A Liga Júnior tem três competições diferentes: On Stage Soccer Rescue CoSpace Rescue
- RoboCup Festo Logistics RLC é uma competição com ênfase no uso de robôs móveis autônomos aplicados aos processos logísticos. Durante a competição, as equipes irão executar tarefas para mover peças de revistas de armazenamento, enfrentando obstáculos ao tentar alcançar seus objetivos.
- IEEE Standard Educational Kits (SEK) Um Kit Educacional IEEE Padrão, inicialmente conhecida Como IEEE LEGO, tem como objetivo apresentar um desafio estimulante para alunos de graduação que precisam montar robôs autônomos usando kits educacionais aprovados para a competição. Dois robôs cooperativos devem agir para executar uma tarefa que muda a cada período.
- IEEE Open A IEEE Open tem o objetivo de apresentar um desafio de alto nível para os alunos. Nesta competição qualquer equipamento pode ser usado para a montagem de um robô autônomo que deve realizar uma tarefa que é mudada a cada período. As tarefas da IEEE Open tentam reproduzir os desafios reais da robótica em uma escala menor.
- IEEE Very Small Size No futebol IEEE Very Small Size duas equipes de 3 robôs de até 7,5×7,5×7,5cm disputam uma partida de futebol. Os robôs são controlados remotamente por um computador, mas sem intervenção humana. O computador processa a imagem de uma câmera de vídeo colocada acima do campo e comanda os robôs.
- IEEE Humanoid Robot Racing Robôs humanóides atraem o interesse do público e pesquisadores de todo o mundo. No entanto, a coordenação e fluidez de movimento, mantendo o equilíbrio para realizar uma determinada tarefa, ainda é um desafio aberto para os humanóides. Em 2012, o RAS Conselho Robotic latino-americana IEEE introduz uma nova categoria robô competição, com a finalidade de desenvolver a coordenação robô humanóide. A competição é chamado Humanoid Racing Robot - IEEE HRR

2. Categorias RoboCup Mundial

Categorias retiradas do site da RoboCup [2].

- Simulation League
- Small-Size League
- Middle-Size League
- Standard Platform League
- Humanoid League
- Rescue Simulation League
- Rescue Robot League
- RoboCup Home
- RoboCup Junior

3. Categorias da FIRA

Categorias retiradas do site da FIRA [3].

- HuroCup
- AmireSot
- MiroSot
- NaroSot
- AndroSot
- RoboSot
- SimuroSot

A.3 Simulador

Seletor de Referências

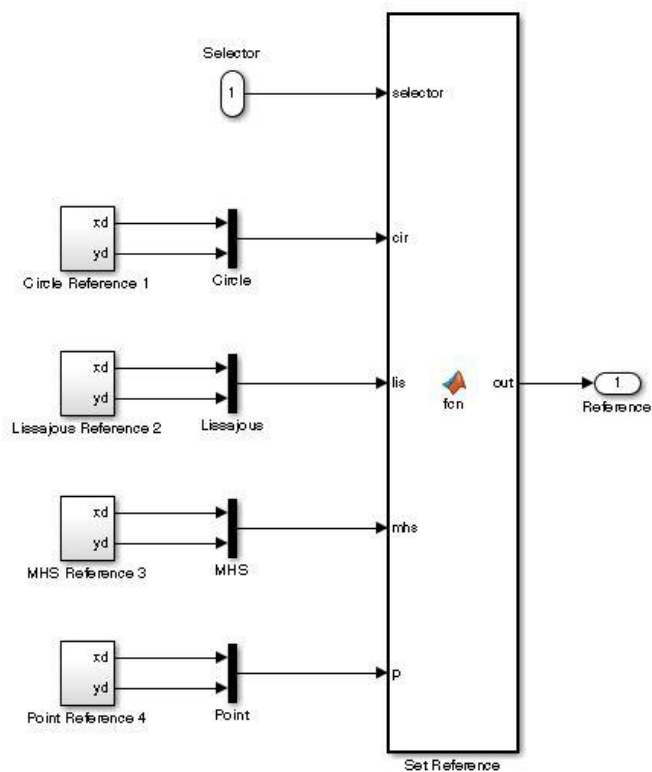


Figura A.8: Seletor de referência do simulador PD

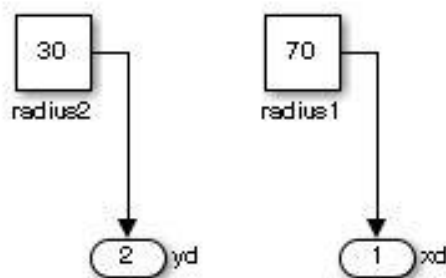


Figura A.9: Referência do pênalti para o simulador PD

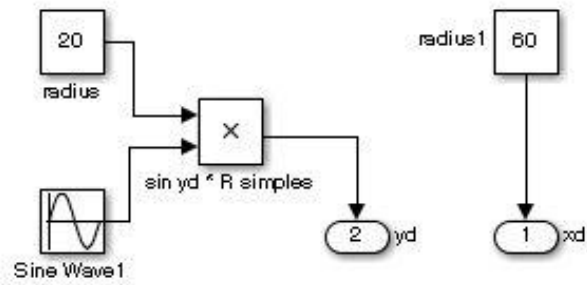


Figura A.10: Referência do movimento de defesa para o simulador PD

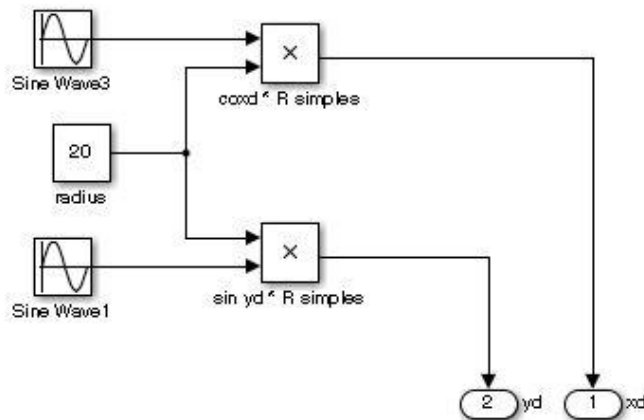


Figura A.11: Referência circular para o simulador PD

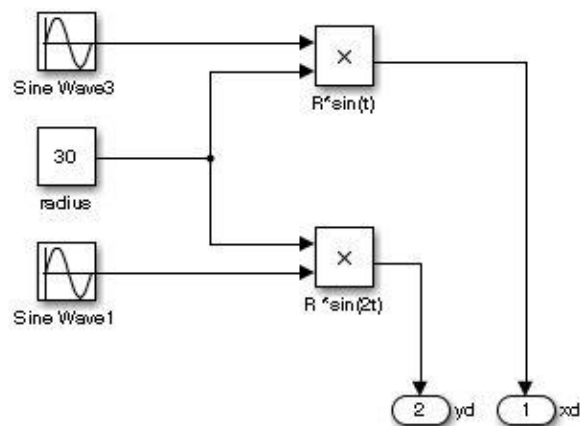


Figura A.12: Referência de uma curva de Lissajous para o simulador PD

Comparador

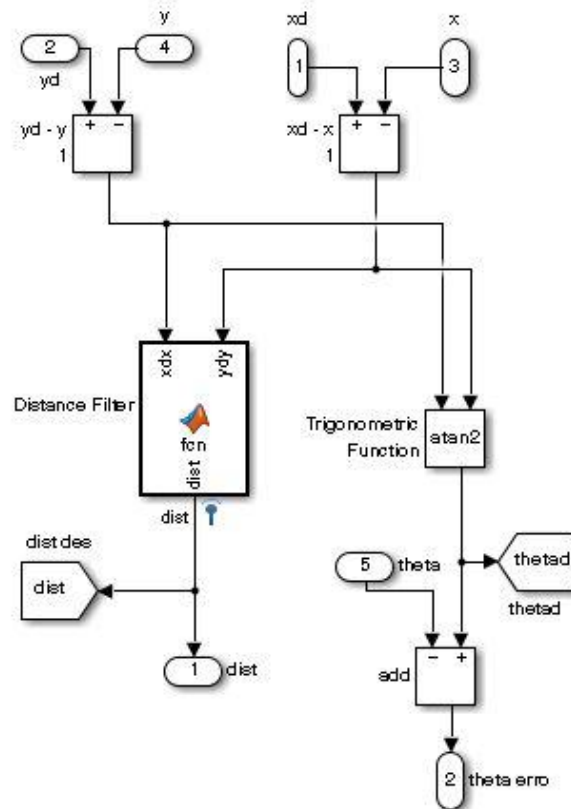


Figura A.13: Comparador de posição do simulador PD

```
1 function dist = fcn(xdx, ydy)
2 dist = sqrt((xdx^2) + (ydy^2));
```

Código A.1: Comparador

Controlador PD

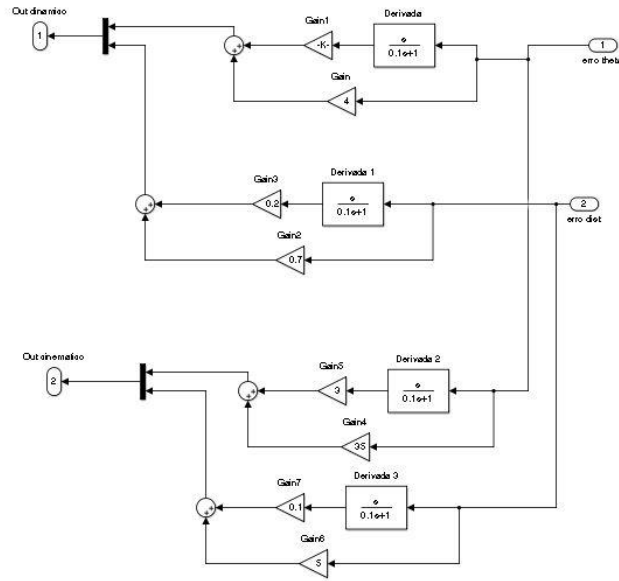


Figura A.14: Controlador PD do simulador PD

Conversor de Sinal

```

1 function [v1,v2]= fcn(ut , ur)
2
3 v1 = ((0.5)*ut + (ur*0.5));
4 v2 = ((0.5)*ut - (ur*0.5));

```

Código A.2: Conversor de Sinal

Modelo Cinemático

```

1 function [sys,x0,str,ts,simStateCompliance] = eikremfunction_SIMPLE(t,xp,up,flag,xi,k,r,1)
2
3 switch flag,
4
5 % Initialization %
6 case 0,
7     [sys,x0,str,ts,simStateCompliance]= mdlInitializeSizes(xi);
8
9 % Derivatives %
10 case 1,
11     sys=mdlDerivatives(t,xp,up,k,r,1);
12
13 % Update and Terminate %
14 case {2,4,9}
15     sys = []; % do nothing
16

```

```

17 % Outputs %
18 case 3,
19     sys=mdlOutputs(t, xp);
20
21 % Unexpected flags %
22 otherwise
23     DAStudio.error('Simulink: blocks: unhandledFlag', num2str(flag));
24
25 end
26
27 function [sys, x1, str, ts, simStateCompliance]=mdlInitializeSizes(xi)
28
29 sizes = simsizes;
30
31 sizes.NumContStates = 3;
32 sizes.NumDiscStates = 0;
33 sizes.NumOutputs = 3;
34 sizes.NumInputs = 2;
35 sizes.DirFeedthrough = 0;
36 sizes.NumSampleTimes = 1; % at least one sample time is needed
37
38 sys = simsizes(sizes);
39 str = [];
40 x1 = xi;
41 ts = [0 0]; % sample time: [period, offset]
42
43 function sys=mdlDerivatives(~, xp, up, k, r, l)
44
45 ut=up(1) + up(2);
46 ur=up(1) - up(2);
47
48 x=xp(1);
49 y=xp(2);
50 theta=xp(3);
51
52 dx=cos(theta)*r*ut/(2*k);
53 dy=sin(theta)*r*ut/(2*k);
54 dtheta= r*ur/(2*k*l);
55
56 dxp = zeros(3,1);
57
58 dxp(1) = dx;
59 dxp(2) = dy;
60 dxp(3) = dtheta;
61 sys = dxp;
62
63 function sys=mdlOutputs(~, xp)
64
65 x=xp(1);
66 theta=xp(3);
67 y=xp(2);
68
69 sys=[x y theta];

```

Código A.3: Modelo Cinemático

Modelo Dinâmico

```
1 function [sys,x0,str,ts,simStateCompliance] = eikremfunction(t, xp, up, flag, xi, k, r, R, m, l,
2     I)
3 switch flag,
4
5     % Initialization %
6     case 0,
7         [sys,x0,str,ts,simStateCompliance]=mdlInitializeSizes(xi);
8
9     % Derivatives %
10    case 1,
11        sys=mdlDerivatives(t,xp,up,k,r,R,m,l,I);
12
13    % Update and Terminate %
14    case {2,4,9}
15        sys = []; % do nothing
16
17    % Outputs %
18    case 3,
19        sys=mdlOutputs(t,xp);
20
21    % Unexpected flags %
22    otherwise
23        DAStudio.error('Simulink:blocks:unhandledFlag', num2str(flag));
24
25 end
26
27 function [sys,x0,str,ts,simStateCompliance]=mdlInitializeSizes(xi)
28
29 sizes = simsizes;
30
31 sizes.NumContStates = 5;
32 sizes.NumDiscStates = 0;
33 sizes.NumOutputs = 3;
34 sizes.NumInputs = 2;
35 sizes.DirFeedthrough = 0;
36 sizes.NumSampleTimes = 1; % at least one sample time is needed
37
38 sys = simsizes(sizes);
39 str = [];
40 x0 = xi;
41 ts = [0 0]; % sample time: [period, offset]
42
43 simStateCompliance = 'DefaultSimState';
44
45 function sys=mdlDerivatives(~,xp,up,k,r,R,m,l,I)
46
47
48 ut = up(1) + up(2);
49 ur = up(1) - up(2);
50
51 u=xp(1);
52 x=xp(2);
53 y=xp(3);
54 w=xp(4);
55 theta=xp(5);
```



```

56
57 du=(-2*k^2)/((r^2)*R)*(u/m) +(k/(r*R))*(ut/m);
58 dx=cos(theta)*u;
59 dy=sin(theta)*u;
60 dw=(-2*(k^2)*(l^2))/((r^2)*R*I)*w + (k*l*ur)/(r*R*I);
61 dtheta=w;
62
63 dxp = zeros(5,1);
64 dxp(1) = du;
65 dxp(2) = dx;
66 dxp(3) = dy;
67 dxp(4) = dw;
68 dxp(5) = dtheta;
69 sys = dxp;
70
71 function sys=mdlOutputs(~,xp)
72
73 x=xp(2);
74 theta=xp(5);
75 y=xp(3);
76
77 sys=[x y theta];

```

Código A.4: Modelo Dinâmico

Plotter

```

1
2 function fcn(liga,x,xd, y, yd,theta, thetad,t, u,w)
3 coder.extrinsic('scatter');
4
5 if liga == 1
6     if t == 0
7         close all
8         f1 = figure;
9         f2 = figure;
10        f3 = figure;
11
12        set(f1, 'tag', 'Plot1')
13        set(f2, 'tag', 'Plot2')
14        set(f3, 'tag', 'Plot3')
15
16    end
17
18    figure(findobj(0,'tag','Plot1'))
19    hold on
20    scatter(x,y,'r')
21    scatter(xd,yd,'b')
22    title('\fontsize{16}Movimento Real: (X x Y) x (Xd x Yd)')
23    axis([-75,75,-65,65])
24    hold on
25    figure(findobj(0,'tag','Plot2'))
26    subplot(3,1,1)
27    hold on
28    scatter(t,x,'r')
29    scatter(t,xd,'b')

```

```

30     title ('\fontsize{16}X x Xd')
31     grid on
32
33     subplot(3,1,2)
34     hold on
35     scatter(t,y, '.r')
36     scatter(t,yd, '.b')
37     title ('\fontsize{16}Y x Yd')
38     grid on
39
40     subplot(3,1,3)
41     hold on
42     scatter(t,theta * 180/3.14, '.r')
43     scatter(t,thetad*180/3.14, '.b')
44     title ('\fontsize{16}Theta x Thetad')
45     grid on
46
47
48     figure(findobj(0,'tag', 'Plot3'))
49     subplot(2,1,1)
50     hold on
51     scatter(t,u, '.b')
52     title ('\fontsize{16} U X T')
53     grid on
54
55     subplot(2,1,2)
56     hold on
57     scatter(t,w, '.b')
58     title ('\fontsize{16} W X T')
59     grid on
60 end

```

Código A.5: Ploter

Comparador Posição Horizontal

```

1 function ut= fcn( theta ,erro_x ,xdp, k, r)
2
3 ut = (2*k/(r*cos(theta)))*(erro_x + xdp);

```

Código A.6: Comparador Posição Horizontal

Comparador Posição Vertical

```

1 function sen= fcn( ydp,erro_y ,ut, k, r)
2 if ut ~= 0
3     sen = (2*k/(r*(ut)))*(erro_y + ydp);
4 else
5     sen = 0;
6 end;

```

Código A.7: Comparador Posição Vertical

Comparador Posição Angular

```
1 function ur = fcn(erro_theta , thetadp , k , r ,l)
2
3 ur = (2*k*l/r)*thetadp + erro_theta ;
```

Código A.8: Comparador Posição Angular

A.4 Experimentos

Foi utilizado o código abaixo para gerar os gráficos dos experimentos no Matlab.

```
1 M = csvread('C:\Users\dados.csv');
2
3 x = M(:,1);
4 xd = M(:,2);
5 y = M(:,3);
6 yd = M(:,4);
7 theta = M(:,5);
8 thetad = M(:,6);
9 t = M(:,7);
10
11 set(0,'defaultfigurecolor',[1 1 1])
12
13 a = imread('C:\Users\CAMPO.JPG');
14
15 figure
16
17 image(a);
18 hold on
19
20 plot(x*0.1,y*0.1,'-r', xd*0.1,yd*0.1,'-b')
21 legend('\fontsize{14}(X x Y)', '\fontsize{14}Xd x Yd')
22 title('\fontsize{16}Movimento Real: (X x Y) x (Xd x Yd)')
23
24 figure
25 subplot(3,1,1)
26 plot(t,x*0.1,'-r',t,xd*0.1,'-b',t,xf*0.1,'-g')
27 legend ('\fontsize{14}X', '\fontsize{14}Xd', '\fontsize{14}Xf')
28 title('\fontsize{16}X x Xd x Xf')
29 grid on
30
31 subplot(3,1,2)
32 plot(t,y*0.1,'-r',t,yd*0.1,'-b',t,yf*0.1,'-g')
33 legend ('\fontsize{14}Y', '\fontsize{14}Yd', '\fontsize{14}Yf')
34 title('\fontsize{16}Y x Yd x Yf')
35 grid on
36
37 subplot(3,1,3)
38 plot(t,theta * 180/3.14,'-r',t,thetad*180/3.14,'-b')
39 legend('\fontsize{14}theta', '\fontsize{14}thetad')
40 title('\fontsize{16}Theta x Thetad')
```

```
41 grid on
42
43 figure
44 subplot(2,1,1)
45 plot(t,u,'-b')
46 legend('\fontsize{14}(U x T)', '\fontsize{14}Y x Yd')
47 title('\fontsize{16} U X T')
48 grid on
49
50 subplot(2,1,2)
51 plot(t,w,'-b')
52 legend('\fontsize{14}(W x T)', '\fontsize{14}Y x Yd')
53 title('\fontsize{16} W X T')
54 grid on
```

Código A.9: Aquisição de Dados