

UNIVERSIDADE FEDERAL DO RIO GRANDE
CENTRO DE CIÊNCIAS COMPUTACIONAIS
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO
CURSO DE MESTRADO EM ENGENHARIA DE COMPUTAÇÃO

Dissertação de Mestrado

Modelagem de um Robô Móvel para Navegação em Tubos de Seções Transversais Circulares e Retangulares

Vinicius Leite de Garcia

Dissertação de Mestrado apresentado ao Programa de Pós-Graduação em Computação da Universidade Federal do Rio Grande, como requisito parcial para a obtenção do grau de Mestre em Engenharia de Computação

Orientador: Prof. Dr. Emanuel da Silva Diaz Estrada
Co-orientador: Prof. Dr. Rodrigo Zelir Azzolin

Rio Grande, 2023

Ficha Catalográfica

G216m Garcia, Vinicius Leite de.

Modelagem de um robô móvel para navegação em tubos de seções transversais circulares e retangulares / Vinicius Leite de Garcia. – 2023.

91 f.

Dissertação (mestrado) – Universidade Federal do Rio Grande – FURG, Programa de Pós-Graduação em Computação, Rio Grande/RS, 2023.

Orientador: Dr. Emanuel da Silva Diaz Estrada.

Coorientador: Dr. Rodrigo Zelir Azzolin.

1. Controlador PID 2. Modelagem Cinemática 3. Navegação
4. Robótica Móvel 5. ROS I. Estrada, Emanuel da Silva Diaz
II. Azzolin, Rodrigo Zelir III. Título.

CDU 004.896

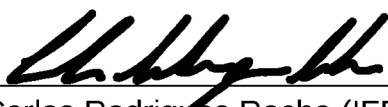
Catálogo na Fonte: Bibliotecário José Paulo dos Santos CRB 10/2344

DISSERTAÇÃO DE MESTRADO

Projeto de um robô móvel para navegação em tubos de seções transversais circulares e retangulares

Vinicius Leite de Garcia

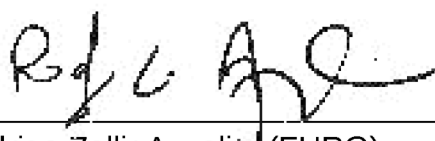
Banca examinadora:



Prof. Dr. Carlos Rodrigues Rocha (IFRS)



Prof. Dr. Eder Mateus Nunes Gonçalves (FURG)

Prof. Dr. Emanuel da Silva Diaz Estrada (FURG)
OrientadorProf. Dr. Rodrigo Zellir Azzolin (FURG)
Coorientador

RESUMO

GARCIA, Vinicius Leite de. **Modelagem de um Robô Móvel para Navegação em Tubos de Seções Transversais Circulares e Retangulares**. 2023. 91 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande, Rio Grande.

Uma das formas de detectar falhas nos sistemas de tubulações na indústria foi inspecionar o interior dos tubos, normalmente estes ambientes possuem diferentes tipos de seções transversais: circular, retangular e quadrada. Devido a variedade de seções, este trabalho teve como intuito apresentar um projeto conceitual de uma modelagem de um robô móvel para navegação em dutos de seções transversais circulares e retangulares, permitindo um sistema híbrido de locomoção. Esta modelagem foi composta por estudo dos cenários circular e retangular; determinação do robô para estes dois ambientes; construção dos ambientes e da solução robótica de forma simulada; desenvolvimento dos modelos cinemáticos para o robô em cada um dos casos; e implementação de um controlador PID (Proporcional-Integral-Derivado) para controlar o dispositivo robótico em cada um dos cenários. Os ambientes de atuação e o robô foram desenvolvidos através de softwares de simulação. O modelo cinemático foi para estudar como se comportam as variáveis controladas, ou seja, os movimentos de avanço e rotação no duto circular, e os movimentos de avanço, subida e descida no tubo retangular. A implementação da estratégia de controle PID foi a partir da análise das equações cinemáticas obtidas. Os resultados deste experimento ilustram o comportamento do modelo em diferentes locais, no cenário circular e retangular, no qual o equipamento robótico foi controlado pela estratégia de controle PID. Validado a pesquisa proposta quanto a navegação do robô nos dois cenários estabelecidos na modelagem do robô apresentada através da simulação computacional, viabilizando futuras testagens em ambientes reais.

Palavras-chave: Controlador PID, Modelagem Cinemática, Navegação, Robótica Móvel, ROS.

ABSTRACT

GARCIA, Vinicius Leite de. **Modeling a Mobile Robot for Navigating Tubes with Circular and Rectangular Cross Sections..** 2023. 91 f. Dissertação (Mestrado) – Programa de Pós-Graduação em Computação. Universidade Federal do Rio Grande, Rio Grande.

One of the ways to detect failures in piping systems in the industry was to inspect the inside of the tubes, normally these environments have different types of cross sections: circular, rectangular and square. Due to the variety of sections, this work aimed to present a conceptual design of a mobile robot modeling for navigation in ducts with circular and rectangular cross sections, allowing a hybrid locomotion system. This modeling was composed by studying the circular and rectangular scenarios; determination of the robot for these two environments; construction of environments and the robotic solution in a simulated way; development of kinematic models for the robot in each case; and implementing a PID (Proportional-Integral-Derivative) controller to control the robotic device in each of the scenarios. The performance environments and the robot were developed using simulation software. The kinematic model was used to study how the controlled variables behave, that is, the advance and rotation movements in the circular duct, and the advance, ascent and descent movements in the rectangular tube. The implementation of the PID control strategy was based on the analysis of the obtained kinematic equations. The results of this experiment illustrate the behavior of the model in different locations, in the circular and rectangular scenario, in which the robotic equipment was controlled by the PID control strategy. The proposed research was validated regarding robot navigation in the two scenarios established in the robot modeling presented through computer simulation, enabling future tests in real environments.

Keywords: PID Controller, Kinematic Modeling, Navigation, Mobile Robotics, ROS.

LISTA DE FIGURAS

2.1	(a) representa um robô do tipo PIG. Esses tipos de robôs são utilizados para inspecionar tubulações em que o próprio fluido serve de força motriz para deslocamento do robô. Normalmente utilizado em oleodutos; (b) representa um robô com rodas. Esses robôs possuem motores acoplados as rodas. A principal característica desse tipo é a facilidade de construção; (c) representa um robô que utiliza o método de esteira para locomoção. A vantagem desse tipo de locomoção se deve a maior área de contato da esteira com a tubulação; (d) representa um robô que pressiona as paredes da tubulação, adotando mecanismos de adaptação a variação de diâmetro interno da mesma; (e) representa um robô com acessórios análogos as pernas. Esse tipo de locomoção permite o deslocamento em variadas condições de operação. Entretanto, são de difícil construção e controle; (f) representa um robô do tipo minhoca; e (g) um robô do tipo parafuso. . .	16
2.2	Robô de inspeção de tubos internos.	17
2.3	Esquemático do robô de inspeção de tubos internos.	17
2.4	Robô para inspecionar e limpar dutos com grandes diâmetros de concreto.	18
2.5	Robô magnético.	18
2.6	Robô Kantaro.	19
2.7	Robô de solução automatizada.	19
2.8	Um estudo do robô de limpeza e inspeção de tubulações.	20
2.9	Robô PCAMS RadPiper.	20
2.10	Sistema robótico pneumático para inspeção de dutos.	21
2.11	Esquemático frontal do robô no interior do duto.	21
2.12	Esquemático lateral do robô no interior do duto.	21
2.13	Robô proposto para inspecionar os cabos elétricos dentro da tubulação.	22
2.14	Robô NPK proposto para limpar os dutos de transporte de NPK.	22
2.15	Robô de locomoção em curvas de 90°.	23
2.16	Mini robô proposto para realizar tarefas de inspeção nas tubulações.	23
2.17	Protótipo de um robô em um tubo e seus componentes.	24
2.18	Robô FURO.	25
2.19	Sistema robótico de inspeção de gasoduto.	25
2.20	Micro robô de inspeção para tubulações de 1 pol.	26
2.21	Robô de inspeção para dutos.	26
2.22	Robô bio-inspirado de inspeção.	26
2.23	Robô de inspeção semi autônomo.	27

2.24	Robô de inspeção de tubo adaptável.	27
2.25	Robô de inspeção de tubulações.	28
2.26	Robô de inspeção de tubos.	28
2.27	Robô de inspeção de tubos (PIKo).	29
2.28	Robô de inspeção para tubulações de água.	29
2.29	Configuração do robô.	30
2.30	Robô de inspeção de tubos.	30
2.31	Estrutura típica de um pacote ROS em C++.	32
2.32	Exemplo de objeto no formato STL.	35
2.33	Interface do Gazebo.	35
2.34	Exemplo de esboço de um robô no gazebo.	36
3.1	CAD do robô.	39
3.2	(a) representa o robô na vista isométrica, onde ele poder se mover tanto para frente quanto trás; (b) representa o modelo robótico na vista frontal rotacionando no sentido anti-horário; e (c) representa o dispositivo na vista frontal girando no sentido horário.	40
3.3	(a) o robô está no plano isométrico podendo se locomover tanto para frente quanto para trás; (b) o equipamento está no plano frontal e realiza o movimento para a parte superior do tubo; (c) o dispositivo também está no plano frontal só que está se dirigindo para a parte inferior do tubo; (d) o robô está na vista lateral esquerda rotacionando no sentido anti-horário, com centro nas rodas traseiras, e se dirigindo para a parte superior; e (e) o sistema robótico está na vista lateral esquerda e rotaciona no sentido horário, com centro nas rodas traseiras, e se locomovendo para a parte inferior.	41
3.4	Sistema de coordenadas cartesianos xyz vista frontal	42
3.5	Sistema de coordenadas cartesianos xyz vista superior	43
3.6	Sistema de coordenadas cartesianos xyz vista lateral esquerda	43
3.7	Esquemático das rodas frontais do robô	43
3.8	Esquemático da vista frontal do robô.	44
3.9	Diagrama de blocos para o controle no tubo circular.	46
3.10	Fluxograma de deslocamento de tarefa no tubo circular.	48
3.11	Sistema de coordenadas cartesianos xyz vista frontal	50
3.12	Esquemático das rodas do robô.	50
3.13	Esquemático do robô no plano yz - Etapa A	51
3.14	Esquemático do robô no plano yz - Etapa B	52
3.15	Esquemático do robô no plano yz - Etapa C	52
3.16	Esquemático do robô no plano yz - Etapa D	53
3.17	Diagrama de blocos para o controle no tubo retangular.	55
3.18	Fluxograma de deslocamento de tarefa no tubo retangular.	56
4.1	Imagem frontal e isométrica do tubo com seção circular.	57
4.2	Imagem frontal e isométrica do tubo com seção retangular.	57
4.3	(a) Robô proposto por Mello-Jr et al. (2008) e (b) visão isométrica robô no simulador Gazebo.	58
4.4	Imagens do robô navegando ao longo do duto circular.	59

4.5	(a) o robô posicionado na origem do ambiente ($y = 0$ m e $\theta = 0^\circ$); (b) o sistema robótico está à 90° em relação a origem; (c) o equipamento está à 180° em relação a origem; (d) o robô está à 270° em relação a posição inicial; e (e) o robô executou uma volta completa ($\theta = 360^\circ$).	59
4.6	Imagem frontal e isométrica do robô no tubo retangular.	60
4.7	(a) o sistema robótico na posição inicial ($y = 0$ m, $z = 0,18$ m e $\phi = 0^\circ$); (b) o robô se desloca para a parte superior do tubo através da variação dos ângulos de ataque das rodas; (c) o equipamento se locomove na parte superior do ambiente de atuação; (d) o dispositivo se move para a parte inferior do tubo retangular; e (e) o robô é apresentado no seu estágio final do ambiente, onde se encontra na parte inferior.	60
4.8	Teste 1 ($y^* = 5$ m e $\theta^* = 0^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	62
4.9	Teste 2 ($y^* = 0$ m e $\theta^* = 0^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	62
4.10	Teste 3 ($y^* = 5,5$ m e $\theta^* = 30^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	63
4.11	Teste 4 ($y^* = 5,5$ m e $\theta^* = -45^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	64
4.12	Teste 5 ($y^* = 0,75$ m e $\theta^* = 90^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	65
4.13	Teste 6 ($y^* = 0,75$ m e $\theta^* = -180^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	65
4.14	Teste 7 ($y^* = 0$ m e $\theta^* = 0^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	66
4.15	Teste 8 ($y^* = 0$ m e $\theta^* = 0^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	67
4.16	Teste 9 ($y^* = 0$ m e $\theta^* = 180^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	67
4.17	Teste 10 ($y^* = 9$ m e $\theta^* = 360^\circ$) - (a) Deslocamento Linear em y ;(b) Deslocamento Angular em y (θ);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	68
4.18	Teste 1 ($y^* = 4$ m e $z^* = 0,16$ m) - (a) Deslocamento Linear em y ;(b) Deslocamento Linear em z (z);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	69
4.19	Teste 2 ($y^* = 4$ m e $z^* = 0,20$ m) - (a) Deslocamento Linear em y ;(b) Deslocamento Linear em z (z);(c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	70

4.20	Teste 3 ($y^* = 6$ m e $z^* = 0,12$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	71
4.21	Teste 4 ($y^* = 6$ m e $z^* = 0,22$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	71
4.22	Teste 5 ($y^* = 9$ m e $z^* = 0,25$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	72
4.23	Teste 6 ($y^* = 9$ m e $z^* = 0,05$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	73
4.24	Teste 7 ($y^* = 0$ m e $z^* = 0,24$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	74
4.25	Teste 8 ($y^* = 2$ m e $z^* = 0,14$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	74
4.26	Teste 9 ($y^* = 1$ m e $z^* = 0,25$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).	75
A.1	Mensagem do terminal de confirmação da criação do pacote ROS. . .	85
A.2	(a) Imagem CAD do tubo circular no formato STL; e (b) Imagem CAD do tubo retangular no formato STL.)	87
A.3	Barra de ferramenta <i>Insert</i> para adicionar os ambientes de atuação no Gazebo.	87
A.4	Imagem do robô feita em CAD.	88
A.5	(a) Chassi do robô em formato STL; (b) Braço direito do robô em formato STL; (c) Braço esquerdo do robô em formato STL; (d) Roda frontal direita do robô em formato STL; (e) Roda frontal direita e esquerda do robô em formato STL; (f) Roda traseira direita do robô em formato STL; e (g) Roda traseira esquerda do robô em formato STL.	88
A.6	Inércia do robô no Gazebo	89

SUMÁRIO

1	INTRODUÇÃO	12
1.1	MOTIVAÇÕES	12
1.2	ESTADO DA ARTE	13
1.3	OBJETIVOS	14
1.3.1	OBJETIVO GERAL	14
1.3.2	OBJETIVOS ESPECÍFICOS	14
1.4	ORGANIZAÇÃO DO TRABALHO	15
2	FUNDAMENTAÇÃO TEÓRICA	16
2.1	TRABALHOS RELACIONADOS	16
2.1.1	SISTEMA DE LOCOMOÇÃO POR RODAS	17
2.1.2	SISTEMA DE LOCOMOÇÃO POR ESTEIRA	19
2.1.3	SISTEMA DE LOCOMOÇÃO DO TIPO QUE PRESSIONA AS PAREDES DOS TUBOS	20
2.1.4	SISTEMA DE LOCOMOÇÃO TIPO MINHOCA	28
2.2	O SISTEMA ROS	31
2.2.1	CATKIN E WORKSPACE	31
	2.2.1.1 CATKIN	31
	2.2.1.2 WORKSPACE	31
2.2.2	PACOTES ROS	32
2.2.3	TÓPICOS	33
2.2.4	ARQUIVOS URDF E XACRO	34
2.2.5	FORMATO STL E MESHES	34
2.3	GAZEBO	35
2.4	GROUND TRUTH	36
2.5	CINEMÁTICA	36
2.6	CONTROLADOR PID	37
3	MODELAGEM DO ROBÔ MÓVEL PARA NAVEGAÇÃO EM DUTOS DE SEÇÕES CIRCULARES E RETANGULARES	38
3.1	CRIAÇÃO DOS AMBIENTES DE NAVEGAÇÃO E DA SOLUÇÃO ROBÓTICA	38
3.2	ASPECTOS GERAIS DO MODELO CINEMÁTICO E ESTRATÉGIA DE CONTROLE	40
3.3	MODELAGEM CINEMÁTICA E ESTRATÉGIA DE CONTROLE DO ROBÔ PARA OS AMBIENTES CIRCULAR	41
3.3.1	MODELO CINEMÁTICO	41

3.3.2	CONTROLADOR PID	46
3.4	MODELAGEM CINEMÁTICA E ESTRATEGIA DE CONTROLE PID DO ROBÔ PARA OS AMBIENTES RETANGULAR	49
3.4.1	MODELO CINEMÁTICO	49
3.4.2	CONTROLADOR PID	54
4	RESULTADOS	57
4.1	SIMULADOR	57
4.1.1	AMBIENTES DE LOCOMOÇÃO E PROPOSTA ROBÓTICA	57
4.1.2	PROJETO SIMULADO	58
4.2	RESULTADOS DO CONTROLADOR PID PARA O ROBÔ NO INTE- RIOR DOS DUTOS CIRCULAR E RETANGULAR	60
4.2.1	PARA A SOLUÇÃO ROBÓTICA NO TUBO CIRCULAR	61
4.2.2	PARA A SOLUÇÃO ROBÓTICA NO TUBO RETANGULAR	68
5	CONCLUSÕES	76
	REFERÊNCIAS	78
	APÊNDICE A CRIAÇÃO DA SIMULAÇÃO COMPUTACIONAL	83
A.1	CONFIGURANDO O <i>WORKSPACE</i>	83
A.2	CRIANDO OS PACOTE ROS	84
A.3	CRIAÇÃO DOS ARQUIVOS PARA OS DOIS AMBIENTES DE ATUAÇÃO	85
A.4	CONSTRUÇÃO DO ARQUIVO DO ROBÔ MÓVEL PARA NAVEGAÇÃO EM DUTOS COM SEÇÕES TRANSVERSAIS CIRCULARES E RETANGULARES	87
A.5	CRIAÇÃO DOS ARQUIVOS LAUNCH E CONFIG	89
A.6	CRIAÇÃO DOS ARQUIVOS MY_ODOM_PUBLISHER E SIM- PLE_CONTROLLER	91

1 INTRODUÇÃO

1.1 MOTIVAÇÕES

A utilização de tubulações por seres humanos antecede provavelmente a história escrita (TELLES, 1996). Segundo Telles (1996), foram encontrados vestígios de sistemas de dutos em locais como as ruínas da Babilônia, da China antiga e de Pompéia. O autor ainda destaca que os primeiros tubos metálicos foram produzidos de chumbo, com uma estimativa de terem sido construídos séculos atrás, contendo instalações com esse material observadas nas termas da Roma Antiga.

As tubulações possuem um papel importante na humanidade, segundo Sherwood e Whistance (1991), elas são usadas para fins industriais, marítimos e comerciais, e tem como função transportar líquidos entre tanques de armazenamento e unidades de processamento, e também transportar vapor, ar, água (estado líquido), entre outros fluidos.

Segundo Affonso (2012), os processos industriais se utilizam de muitos sistemas de tubulações. A ocorrência de falhas nestas instalações são inevitáveis e podem ter das mais variadas consequências. O autor ainda salienta que, em determinados casos, o prejuízo final não ultrapassa o custo de manutenção; em outros, pode comprometer o lucro da empresa, devido as perdas de produção, acidentes e agressões ao meio ambiente.

Isto posto, falhas nos sistemas de tubulações industriais geram despesas e suas implicações são variáveis, sendo necessárias operações específicas para mitigá-la (AFFONSO, 2012). Segundo Ismail et al. (2012), o processo de inspeção em tubulações pode produzir relevantes melhorias nas questões de segurança e eficiência. O autor ainda destaca que as atividades de limpeza, manutenção, e outras, acarretam em custos altos. Assim, é visto como solução atraente a aplicação de sistemas robóticos para executar serviços dessa natureza.

A robótica tem como preocupação o estudo de máquinas que visam substituir a mão-de-obra humana na realização de tarefas, tanto em relação à atividade física quanto à tomada de decisão (SICILIANO et al., 2010). Segundo Niku (2010), os equipamentos robóticos consistem não apenas em robôs, mas também em outros dispositivos e sistemas utilizados em conjunto com eles. A principal característica desses componentes é a sua

estrutura mecânica, sendo que eles podem ser classificados como robôs manipuladores (possuem a base fixa) e robôs móveis (possuem a base móvel) (SICILIANO et al., 2010).

Segundo Sandler (1999), os robôs móveis são constituídos por uma plataforma móvel e possuem um campo vasto de aplicações. O autor classifica os dispositivos robóticos de acordo com os seus principais meios de propulsão.

Na literatura encontra-se uma vasta quantidade de artigos sobre robótica móvel. Particularmente no interesse deste trabalho, encontram-se soluções robóticas descrevendo processos de navegação e inspeção de dutos de seções circulares e retangulares, pois segundo McGregor et al. (2016), estes dois ambientes são os tipos de dutos mais usados na indústria. O autor ainda destaca que as seções transversais mais utilizadas são as circulares, as quadradas e as retangulares. Devido estas instalações serem muito utilizadas, a robótica se torna necessária para a inspeção dos tubos em vez de trabalhadores humanos, pois segundo Li et al. (2020), o emprego de robôs para estas tarefas de inspecionamento é mais eficiente e segura que o trabalho humano, impedindo assim risco de acidentes.

1.2 ESTADO DA ARTE

Como anteriormente apresentado, encontra-se diversos trabalhos associados à ideia do uso de robôs para navegação e inspeção em sistemas de tubulações. Os estudos mais recentes nessa área destacam que esses equipamentos são capazes de se locomover no interior do tubos circulares ou retangular.

Em Kakogawa, Murata e Ma (2022) foi proposto um método de deslocamento automático de bifurcação de um robô articulado de rodas para inspeção em tubos, chamado de AIRO-5.2, sua estrutura física foi composta por duas juntas elásticas passivas com molas de torção e uma junta ativa, que utiliza de um atuador elástico de série à base de poliuretano (SEA). O modelo robótico pode se locomover tipo uma minhoca e ainda pode passar por bifurcações verticais, porém foi necessário treinar os operadores para executar está determinada função.

Já no trabalho apresentado em Kazeminasab e Banks (2021) foi descrito um equipamento robótico para inspeção e monitoramento da qualidade da água em sistemas de distribuição. Este dispositivo foi composto por três módulos de braço que são conectados ao processador central em ângulos de 120° e tornam o diâmetro externo do robô adaptável ao diâmetro do tubo (228,6 à 558,8 mm), fazendo que o robô pressione as rodas nas paredes e assim se mova dentro do tubo.

A proposta de Tourajizadeh et al. (2020) também foi um modelo robótico usado para inspecionar e investigar o interior de tubulações com taxa de passo variável. Este robô possui atributos que possibilitam contornar obstáculos, pressionar as paredes do duto, e foi capaz de trabalhar na presença de fluido com o auxílio de seu controle robusto projetado.

Já o trabalho proposto por Hadi et al. (2020) foi um projeto conceitual de um robô de locomoção por rodas, feito para detectar as falhas existentes nas paredes internas das tubulações e dos tubos estreitos na indústria, de forma que este dispositivo possa operar de forma adaptável e livre em trajetórias verticais inclinadas e dobradas. O robô utilizou um mecanismo simples baseado em atuadores de liga com memória de forma, ajustando a força de contato entre o equipamento e a parede do tubo.

O robô proposto por Roussialian et al. (2019) foi capaz de executar a navegação dos tubos e ainda se adaptar às diferentes seções circulares, suas rodas inferiores e superiores foram usadas para obter a estabilidade e movimento ao longo do duto.

O trabalho apresentado por Chablat, Venkateswaran e Boyer (2019) consistiu em um equipamento robótico bio-inspirado para inspeção de tubulações que proporciona o movimento de uma lagarta. Esse dispositivo contém três módulos que permitiu o alongamento e a fixação do mesmo na tubulação, proporcionando a navegação do sistema no ambiente.

Outro modelo de inspeção de tubos de seções circulares foi o desenvolvido por Jones et al. (2019). O dispositivo, chamado de PCAMS RadPiper, também foi usado para inspecionar e obter informações no interior de tubulações para difusão gasosa. Este robô possui atributos que possibilitam o seu movimento horizontal sobre a base de um duto circular.

Observa-se que os modelos mostrados anteriormente possuem atributos eficientes, porém insatisfatórios para a navegação para os dois ambientes circular e retangular. É necessário que o robô trabalhe em diferentes locais internos em ambos tubos, especificamente na parte inferior e superior. Estudos apontados na literatura apresentam dispositivos robóticos capazes de realizar deslocamentos em sistemas de tubulações com seções transversais circulares ou que se movimentam na parte inferior do tubo retangular.

Portanto, o estado da arte apresenta uma necessidade de uma solução robótica para navegação em diferentes tipos de perfis de seções, capaz de transladar e rotacionar no mesmo eixo no tubo circular, e também transladar na parte superior ou inferior do duto retangular. Assim, o robô realiza a avaliação em qualquer interna da tubulação.

1.3 OBJETIVOS

1.3.1 OBJETIVO GERAL

O objetivo do trabalho foi modelar o projeto de Mello-Jr et al. (2008) em um robô para navegação no interior de dutos capaz de atuar em seções circulares e retangulares.

1.3.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos para o desenvolvimento deste trabalho foram:

- estudar os ambientes de atuação circular e retangular;

- propor uma solução robótica para esses dois cenários de atuação;
- construir os ambientes e o robô de forma simulada;
- desenvolver modelos cinemáticos para o robô atuar em cada um dos ambientes;
- implementar estratégias de controle para controlar o robô em cada um dos cenários.

1.4 ORGANIZAÇÃO DO TRABALHO

O trabalho está organizado em cinco capítulos. Afora este capítulo introdutório, o Capítulo 2 apresenta uma fundamentação teórica contendo os trabalhos relacionados a inspeção e navegação de dutos em seção transversais circulares e retangulares e suas ferramentas computacionais. O Capítulo 3 aborda o projeto conceitual da modelagem, destacando os ambientes, o robô, modelos cinemáticos e a estratégia de controle PID. No Capítulo 4, apresentam-se os resultados obtidos ao longo do desenvolvimento e análise dos estudos realizados. No Capítulo 5, apresentam-se as conclusões. Por fim, apresenta-se o Apêndice.

2 FUNDAMENTAÇÃO TEÓRICA

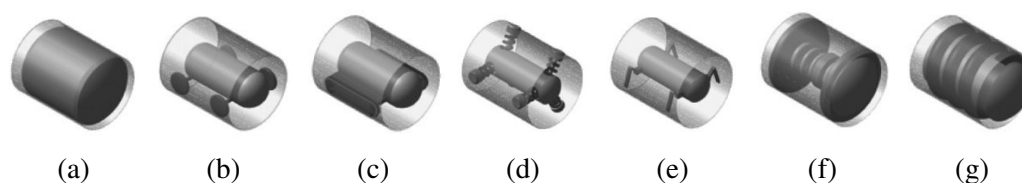
Para o entendimento do trabalho proposto, torna-se necessário apresentar uma breve revisão de sua plataforma robótica no que diz respeito aos trabalhos relacionados e ao conceito das ferramentas computacionais. Este capítulo irá detalhar os robôs de navegação, o sistema ROS, o Gazebo e o *Ground Truth*.

2.1 TRABALHOS RELACIONADOS

Na literatura são encontrados diversos equipamentos automáticos que se deslocam na parte interna de tubulações tanto de formato cilíndrico como de seção retangular. A seguir serão referenciados trabalhos que poderiam, através de modificações, serem também utilizados para navegação de dutos como aqui proposto.

Os equipamentos robóticos utilizados para inspeção interna de tubulações podem ser classificados de acordo com o método de locomoção adotado, segundo Roh e Choi (2005), conforme pode ser visualizado na Figura 2.1.

Figura 2.1: (a) representa um robô do tipo PIG. Esses tipos de robôs são utilizados para inspecionar tubulações em que o próprio fluído serve de força motriz para deslocamento do robô. Normalmente utilizado em oleodutos; (b) representa um robô com rodas. Esses robôs possuem motores acoplados as rodas. A principal característica desse tipo é a facilidade de construção; (c) representa um robô que utiliza o método de esteira para locomoção. A vantagem desse tipo de locomoção se deve a maior área de contato da esteira com a tubulação; (d) representa um robô que pressiona as paredes da tubulação, adotando mecanismos de adaptação a variação de diâmetro interno da mesma; (e) representa um robô com acessórios análogos as pernas. Esse tipo de locomoção permite o deslocamento em variadas condições de operação. Entretanto, são de difícil construção e controle; (f) representa um robô do tipo minhoca; e (g) um robô do tipo parafuso.

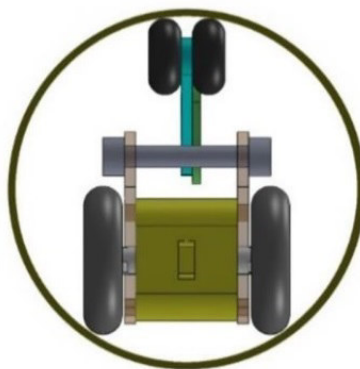


Fonte: Adaptado de Roh e Choi (2005, p. 1)

2.1.1 SISTEMA DE LOCOMOÇÃO POR RODAS

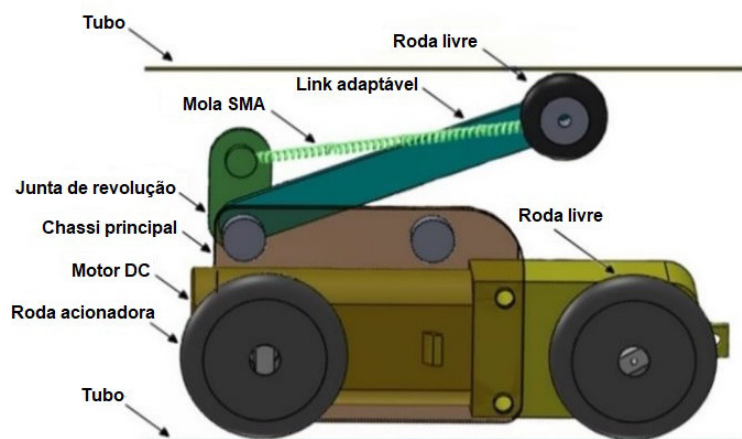
No trabalho proposto por Hadi et al. (2020) foi apresentado um projeto conceitual de um sistema robótico em miniatura feito para inspecionar as falhas existentes nas paredes internas das tubulações e dos tubos estreitos na indústria, de forma que este dispositivo possa operar de forma adaptável e livre em trajetórias verticais inclinadas e dobradas. O robô utiliza um mecanismo simples baseado em atuadores de liga com memória de forma, assim ajusta a força de contato entre o dispositivo e a parede do tubo. O uso destes materiais possibilita que o robô seja menor e mais leve, como consequência proporciona uma maior mobilidade em dutos mais complexos. As Figuras 2.2 e 2.3 mostram esta ferramenta.

Figura 2.2: Robô de inspeção de tubos internos.



Fonte: Adaptado de Hadi et al. (2020, p. 2)

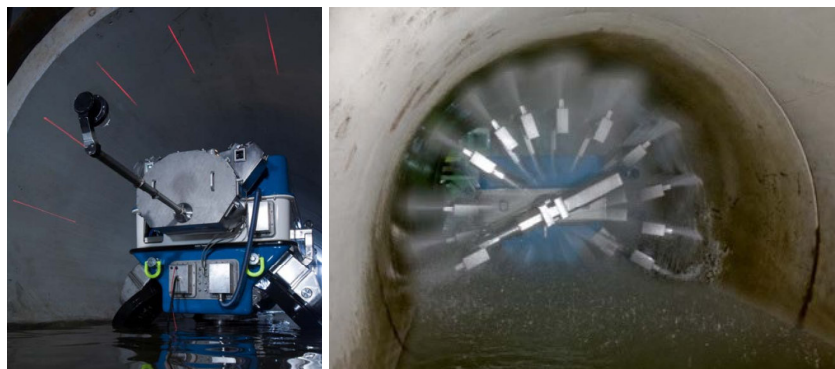
Figura 2.3: Esquemático do robô de inspeção de tubos internos.



Fonte: Adaptado de Hadi et al. (2020, p. 2)

Um outro sistema robótico com rodas, ilustrado na Figura 2.4, foi utilizado para inspecionar e limpar a superfície interna de dutos de concreto (SAENZ et al., 2010). A tubulação utilizada em testes práticos foi de 2300 mm de diâmetro.

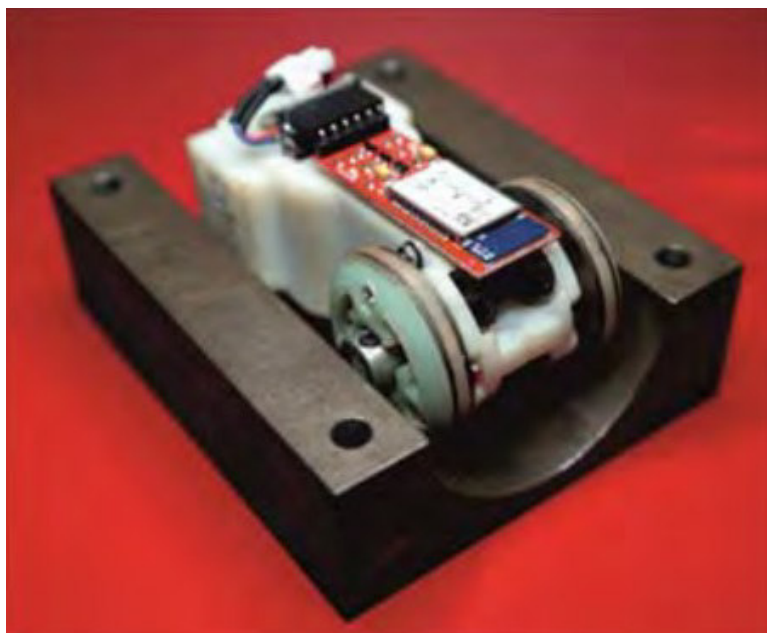
Figura 2.4: Robô para inspecionar e limpar dutos com grandes diâmetros de concreto.



Fonte: Adaptado de Saenz et al. (2010, p. 2 e 3)

Um outro trabalho eficiente foi o de Mills et al. (2018), o qual consiste em um projeto robótico miniaturizado em paredes magnéticas, conforme mostra a Figura 2.5. O intuito desse robô foi para solucionar a inspeção de tubos de diâmetro de 50 mm, podendo escalar obstáculos na tubulação para acessar conexões de tubos maiores. Segundo o autor, o mesmo pode usar as matrizes magnéticas dirigidas, através das placas de fluxo de aço dentro das rodas, fazendo o equipamento aderir ao tubo. E o seu controle foi feito através de rádio comunicação *Bluetooth*, da frequência de 2,4 GHz, até a distância de 100 m.

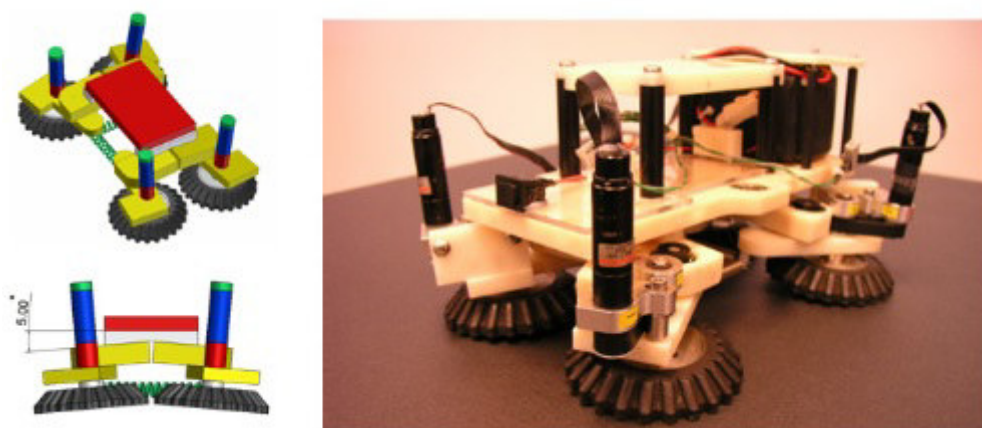
Figura 2.5: Robô magnético.



Fonte: Adaptado de Mills et al. (2018, p. 10)

Já o Kantaro foi um robô utilizado para inspecionar esgotos no Japão (NASSIRAEI et al., 2007). Esse robô, ilustrado na Figura 2.6, foi capaz de inspecionar dutos retos de 200 à 300 mm. Ele contém uma bateria de lítio que suporta até quatro horas de funcionamento.

Figura 2.6: Robô Kantaro.

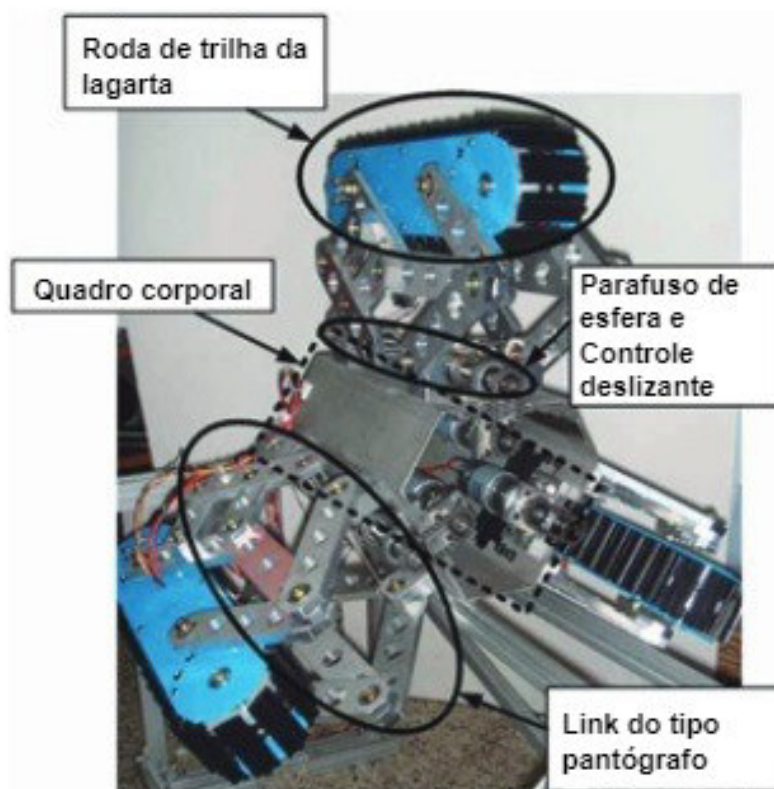


Fonte: Adaptado de Nassiraei et al. (2007, p. 5)

2.1.2 SISTEMA DE LOCOMOÇÃO POR ESTEIRA

Um robô do tipo esteira foi proposto por Kim et al. (2009). Foi inicialmente desenvolvido para inspecionar tubulações de água do mar. Esse robô, ilustrado na Figura 2.7, foi projetado para percorrer longas distâncias. Uma característica foi a sua capacidade de adaptação a seções circulares de 600 a 800 mm de diâmetro.

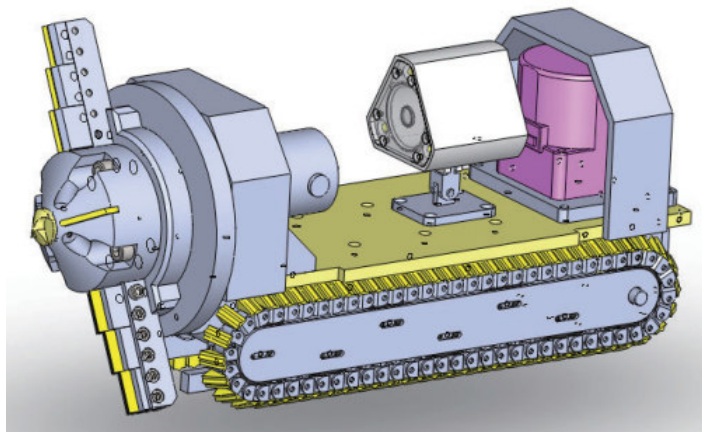
Figura 2.7: Robô de solução automatizada.



Fonte: Adaptado de Kim et al. (2009, p. 2)

Um robô móvel do tipo lagarta foi proposto para limpar as tubulações de esgoto no Vietnã (TRUONG-THINH; NGOC-PHUONG; PHUOC-THO, 2011). Esse equipamento, mostrado na Figura 2.8, foi utilizado para inspecionar tubulações com diâmetros menores que 800 mm. Ele possui acesso remoto e consegue ultrapassar obstáculos de até 100 mm.

Figura 2.8: Um estudo do robô de limpeza e inspeção de tubulações.



Fonte: Adaptado de Truong-Thinh, Ngoc-Phuong e Phuoc-Tho (2011, p. 1)

O robô desenvolvido por Jones et al. (2019) chamado de PCAMS RadPiper, demonstrado na Figura 2.9, foi usado para executar a navegação e obter informações no interior das tubulações de difusão gasosa. Esse sistema robótico possui atributos eficientes como medição precisa de distância, detecção de obstáculos no tubo, capacidade de transformação para dois tamanhos de tubo e robustez na operação autônoma.

Figura 2.9: Robô PCAMS RadPiper.

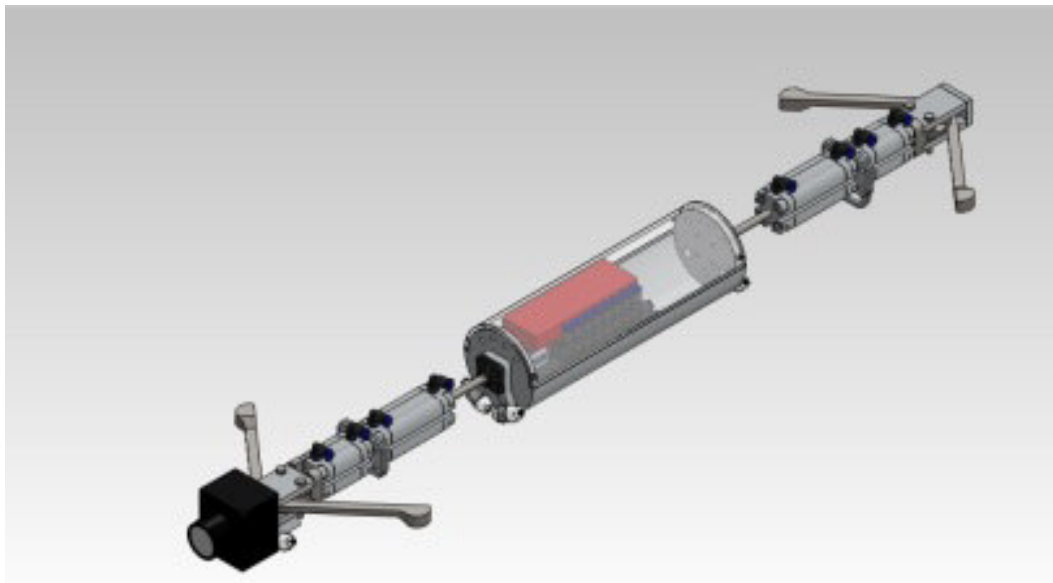


Fonte: Adaptado de Jones et al. (2019, p. 2 e 6)

2.1.3 SISTEMA DE LOCOMOÇÃO DO TIPO QUE PRESSIONA AS PAREDES DOS TUBOS

O trabalho de Núñez et al. (2013) apresentou um sistema robótico pneumático que tem por objetivo dar suporte ao processo de inspeção de eletrodutos da Companhia Estadual de Energia Elétrica do Rio Grande do Sul (CEEE). Esse robô foi baseado no movimento de uma lagarta, foram utilizados garras que possibilitam os braços se fixarem nas ranhuras internas do duto de paredes corrugadas (ondulações em um padrão helicoidal). A Figura 2.10 mostra este dispositivo.

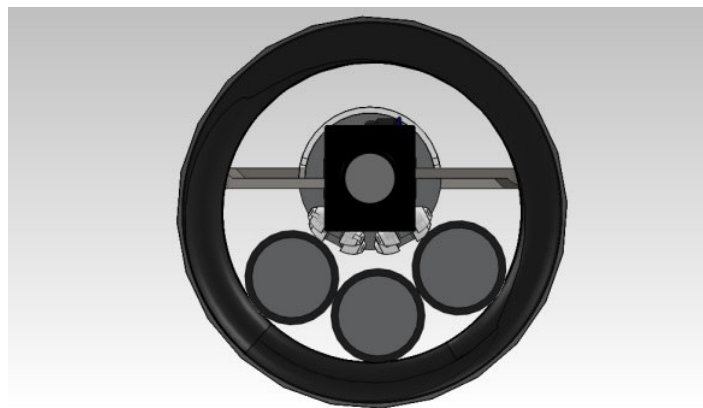
Figura 2.10: Sistema robótico pneumático para inspeção de dutos.



Fonte: Adaptado de Núñez et al. (2013, p. 5)

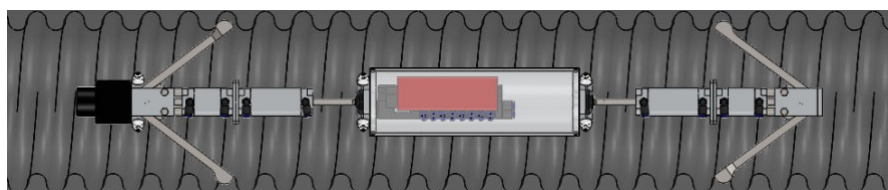
O tubo por onde este sistema robótico se locomove, possui um diâmetro de 150 mm e passam três cabos em seu interior, sendo cada cabo de até 50 mm de diâmetro. As Figuras 2.11 e 2.12 demonstram como esse equipamento robótico foi inserido no interior do duto.

Figura 2.11: Esquemático frontal do robô no interior do duto.



Fonte: Adaptado de Núñez et al. (2013, p. 5)

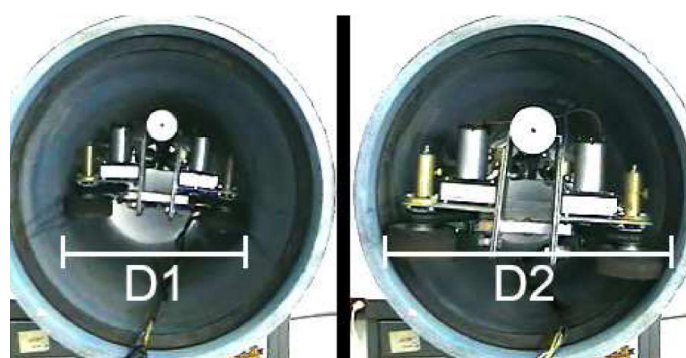
Figura 2.12: Esquemático lateral do robô no interior do duto.



Fonte: Adaptado de Núñez et al. (2013, p. 5)

Um outro sistema robótico usado para inspecionar dutos foi o proposto por Mello-Jr et al. (2008) chamado de TATUBOT. Esse robô, mostrado na Figura 2.13, possui rodas acionadas e tracionadas individualmente por motores, e por um sistema de molas que pressionam as rodas contra as paredes dos dutos circulares. Para que este equipamento mantenha-se alinhado no tubo em qualquer direção radial, as rodas dianteiras e os seus motores de propulsão serão fixados de forma a proporcionar a estabilidade do dispositivo. Referido protótipo foi construído originalmente para inspecionar os cabos de energia elétrica no interior de eletrodutos com 300 mm de diâmetro.

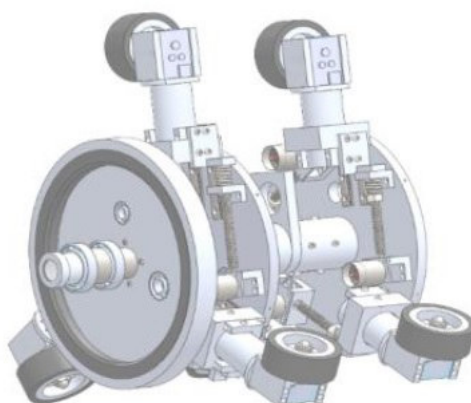
Figura 2.13: Robô proposto para inspecionar os cabos elétricos dentro da tubulação.



Fonte: Adaptado de Mello-Jr et al. (2008, p. 4)

Um dos trabalhos encontrados para limpeza de dutos em empresas de fertilizante apresentou o robô NPK (TRAN et al., 2014). Esse robô, conforme pode ser visualizado na Figura 2.14, utiliza rodas para locomoção, forçando as mesmas em direção a superfície interna do tubo. Assim, o robô foi capaz de se adaptar a variações de diâmetro da tubulação (320-440 mm). O robô NPK foi composto por três hastes distanciadas em 120°, o que permite a centralização da ferramenta de limpeza acoplada a parte frontal do robô. Este modelo, sendo adaptado, poderá também ser usado para inspecionar dutos devido a sua mobilidade.

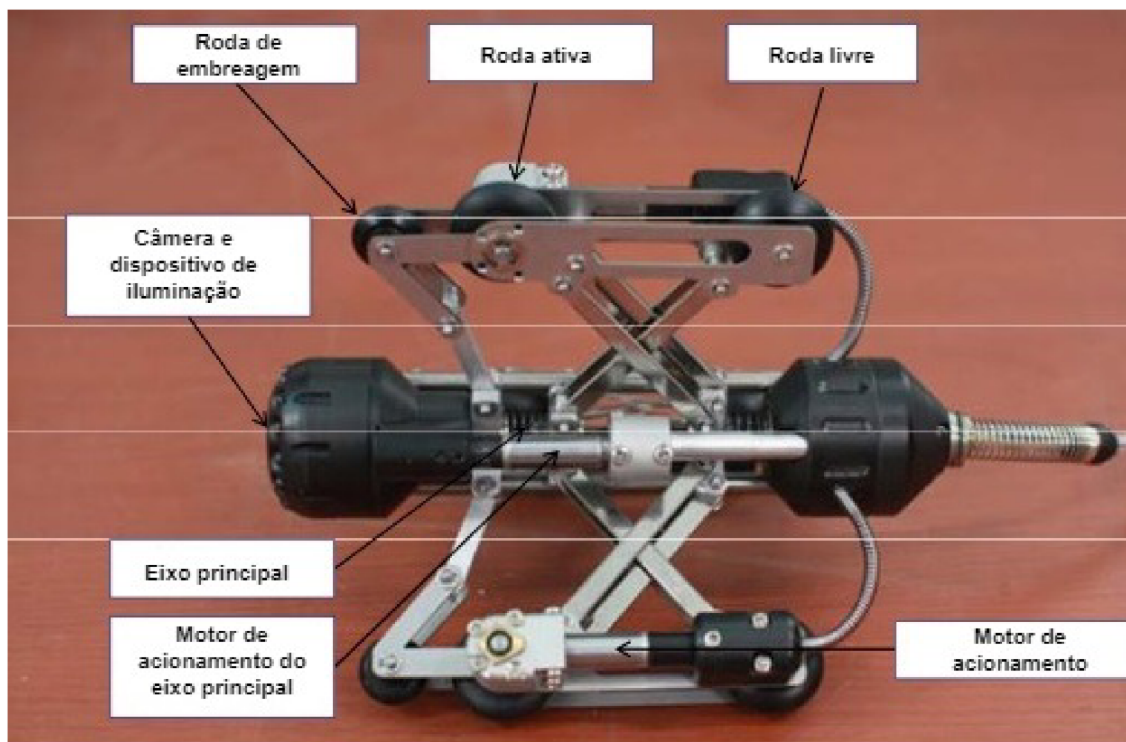
Figura 2.14: Robô NPK proposto para limpar os dutos de transporte de NPK.



Fonte: Adaptado de Tran et al. (2014, p. 1)

Um outro modelo robótico para a problemática em questão foi o proposta por Kwon et al. (2010), onde apresenta um robô que foi capaz de se locomover em curvas de 90° e em conexões de tubo em formato T, sendo desenvolvido para inspecionar tubulações que podem variar de 90 a 110 mm. A Figura 2.15 mostra este equipamento.

Figura 2.15: Robô de locomoção em curvas de 90°.



Fonte: Adaptado de Kwon et al. (2010, p. 2)

Uma plataforma móvel foi proposta para inspecionar dutos de diâmetros de 140 a 200 mm (TĂȚAR; MANDRU; ARDELEAN, 2007). Foi um robô do tipo duas rodas adaptativo com características eficientes. Esse modelo foi proposto para deslocar-se em curvas de 90° e em curvas no formato de T. Uma das ideias foi a utilização do mesmo protótipo para torná-lo modular para transportar o equipamento necessário. A Figura 2.16 mostra este dispositivo.

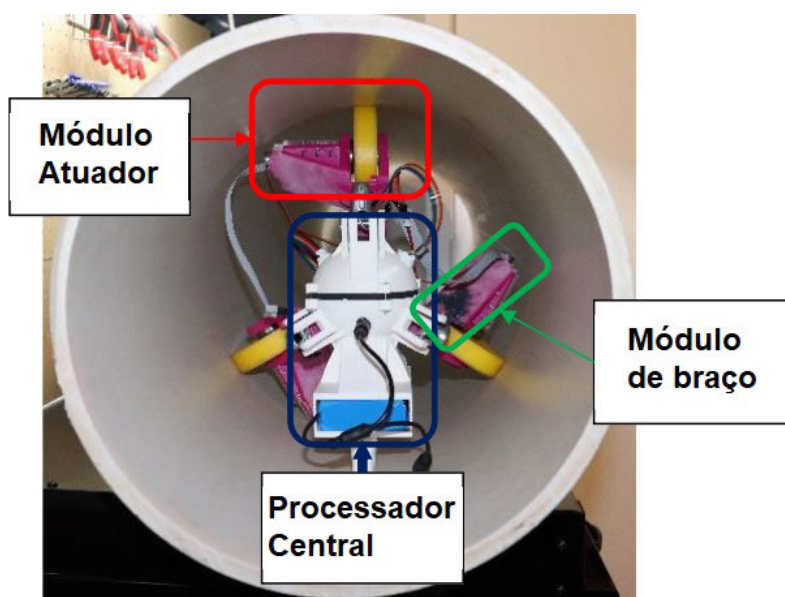
Figura 2.16: Mini robô proposto para realizar tarefas de inspeção nas tubulações.



Fonte: Adaptado de Tătar, Mandru e Ardelean (2007, p. 1)

Kazeminasab e Banks (2021) apresenta um equipamento robótico para inspeção e monitoramento da qualidade da água em sistemas de distribuição, como mostra a Figura 2.17. Este robô usa um procedimento de operação sincronizado de um sistema de comunicação sem fio desenvolvido para ambientes agressivos. A sua estrutura se baseia em três módulos de braço que são conectados ao processador central, em ângulos de 120° , e tornam o diâmetro externo do robô adaptável ao diâmetro do tubo (228,6 à 558,8 mm), fazendo que o robô pressione as rodas nas paredes e assim se mova dentro do tubo.

Figura 2.17: Protótipo de um robô em um tubo e seus componentes.



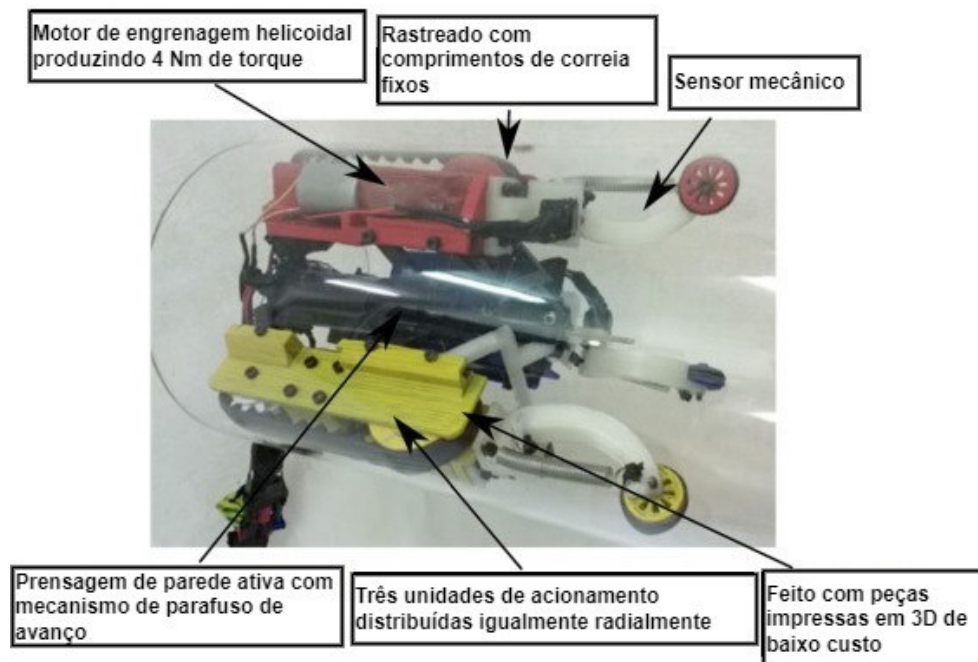
Fonte: Adaptado de Kazeminasab e Banks (2021, p. 2)

Já o robô FURO, desenvolvido para detecção de cotovelos em tubos desconhecidos, foi projetado por Brown et al. (2018), seguindo alguns requisitos como operar dentro de um tubo de 150 mm de diâmetro sem fluidos e também em dutos tanto no sentido horizontal como vertical; navegar por junções de tubos; e carregar uma carga útil de 0,5 kg. O autor ainda destaca que após testes e verificações desse protótipo, ele foi miniaturizado para trabalhar numa faixa de diâmetro entre 50 a 150 mm.

O exemplar FURO, apresentado na Figura 2.18, trabalha nominalmente com um tubo de 150 mm, contendo um mecanismo de parafuso de avanço ao longo do centro do robô, que atribui capacidade de expandir para tamanhos variáveis de 126 a 175 mm, também seus braços são espaçados de 120° entre si.

O trabalho desenvolvido por Zhang e Yan (2007) consiste em um sistema com três conjuntos de mecanismos de pernas espaçadas circunferencialmente em 120° e simétricas. Sendo assim, esse robô, ilustrado na Figura 2.19, tem a capacidade de se adaptar a diâmetros de tubos numa faixa de 400 mm até 650 mm, com ajuste de força de tração estável.

Figura 2.18: Robô FURO.



Fonte: Adaptado de Brown et al. (2018, p. 5)

Figura 2.19: Sistema robótico de inspeção de gasoduto.

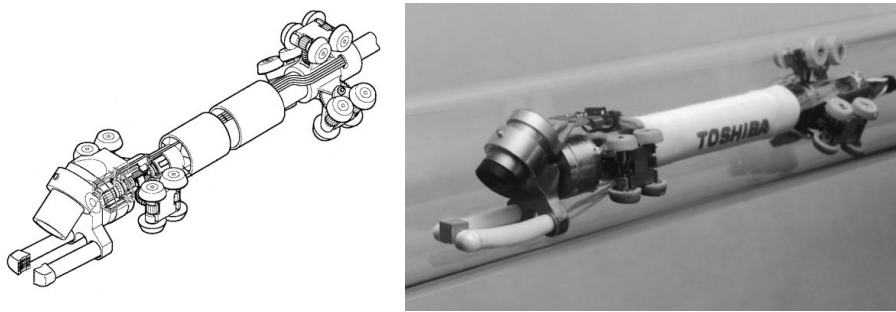


Fonte: Adaptado de Zhang e Yan (2007, p. 12)

Outro sistema robótico encontrado na literatura foi o desenvolvido por Suzumori et al. (1999). Esse micro robô de inspeção foi projetado para seções circulares de uma polegada de diâmetro, sendo equipado com uma câmera CCD (micro *charge-coupled device*) de alta qualidade, e possui uma garra dupla para manipular objetos encontrados nos dutos. A Figura 2.20 ilustra este equipamento. Tal robô possui diâmetro de 23 mm, comprimento de 110 mm, taxa de deslocamento de 6 mm/s, e poder de carga de 1 N.

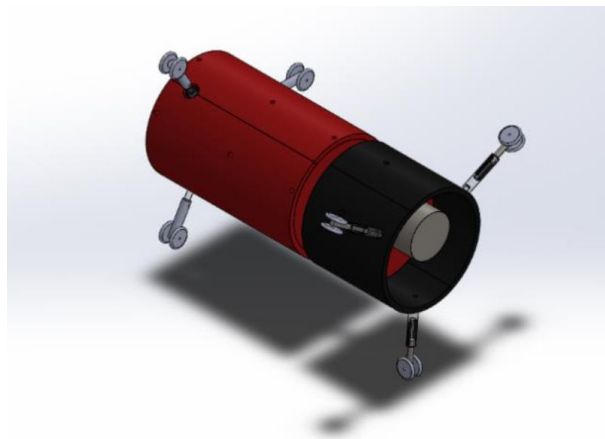
O trabalho apresentado por Tourajizadeh et al. (2020) tem como objetivo de mostrar um modelo robótico usado para inspecionar e investigar o interior de tubulações com taxa de passo variável. Assim, com um novo mecanismo o equipamento proposto consegue contornar obstáculos, pressionar as paredes do duto, e também foi capaz de trabalhar na presença de fluido com o auxílio de seu controle robusto projetado. A Figura 2.21 mostra este dispositivo.

Figura 2.20: Micro robô de inspeção para tubulações de 1 pol.



Fonte: Adaptado de Suzumori et al. (1999, p. 1 e 2)

Figura 2.21: Robô de inspeção para dutos.



Fonte: Adaptado de Tourajizadeh et al. (2020, p. 4)

O projeto elaborado por Chablat, Venkateswaran e Boyer (2019) apresenta um robô bio-inspirado para inspeção de tubulações, que apresenta o movimento de uma lagarta. Esse equipamento robótico possui três módulos, sendo um módulo central para alongamento e dois outros localizados na parte frontal e traseira para fixação. Também foi constituído um mecanismo fenda-deslizante (escolhido para as pernas do robô). A ilustração deste design encontra-se na Figura 2.22.

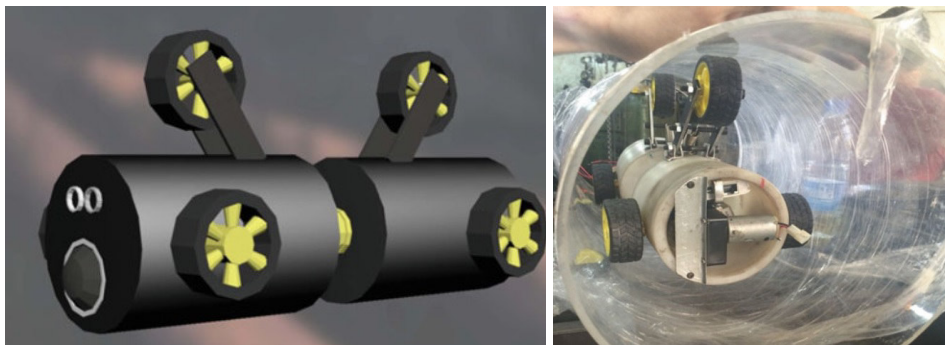
Figura 2.22: Robô bio-inspirado de inspeção.



Fonte: Adaptado de Chablat, Venkateswaran e Boyer (2019, p. 3)

Roussialian et al. (2019) propôs um sistema robótico de inspeção semi autônomo capaz de executar a navegação por áreas estreitas e ao mesmo tempo se adaptar a diferentes seções cilíndricas das tubulações. O seu design e sua interface de controle possibilitam acesso mais fácil e seguro para regiões complexas e distantes de uma tubulação. Este dispositivo projetado pode ser observado na Figura 2.23.

Figura 2.23: Robô de inspeção semi autônomo.



Fonte: Adaptado de Roussialian et al. (2019, p. 6 e 9)

Os autores destacam que esse equipamento acima possui várias características principais, dentre elas: locomoção em tubulações horizontais e verticais; monitoramento e controle sem fio; capacidade de medição da distância percorrida; e também aplicação em diferentes faixas de diâmetros de tubos. Entretanto, apesar de possuir diversas qualidades, o mesmo terá algumas limitações como: inadequação para canos de água e de tubos de óleo; gama limitada de diâmetros entre 250 a 500 mm; e comunicação sem fio limitada a uma distância de no máximo 120 metros.

O projeto elaborado por Nayak e Pradhan (2014) se constitui de um dispositivo robótico de inspeção de tubos adaptável, com parafuso de acionamento de parede. Ele tem a capacidade de se mover tanto horizontal como verticalmente e pode passar por regiões onde se encontram os cotovelos da tubulação. Foi composto por três módulos: o rotor, o estator e a unidade de controle. A Figura 2.24 mostra este equipamento.

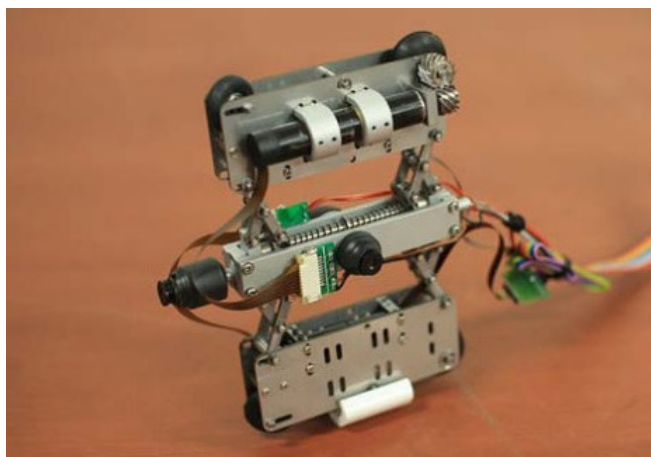
Figura 2.24: Robô de inspeção de tubo adaptável.



Fonte: Adaptado de Nayak e Pradhan (2014, p. 4 e 10)

O trabalho de Kwon et al. (2011) apresentou um sistema robótico de inspeção de tubulação, a qual possui vários sensores para inspecionamento de tubos de 80 a 100 mm. Esse robô possui como característica especial a capacidade de condução e direção usando apenas duas correntes de roda. Também permite o controle simples e fácil interface com usuário, especialmente em tubos de formatos T. Na sua forma plana proporciona a montagem de sensores adicionais nos dois lados do equipamento. A Figura 2.25 ilustra esse dispositivo.

Figura 2.25: Robô de inspeção de tubulações.



Fonte: Adaptado de Kwon et al. (2011, p. 2)

2.1.4 SISTEMA DE LOCOMOÇÃO TIPO MINHOCAS

Kakogawa, Murata e Ma (2022) apresentou um método de deslocamento automático de ramificação em T de um robô articulado de rodas para inspeção em tubos, chamado de AIRO-5.2. Onde equipamento robótico possui uma estrutura física composta por duas juntas elásticas passivas com molas de torção e uma junta ativa, que utiliza de um atuador elástico de série à base de poliuretano (SEA). Este equipamento, como mostra a Figura 2.26, pode passar por ramais em T verticais ao alternar do controle de torque para o controle de ângulo deste SEA, porém, foi necessário treinar os operadores.

Figura 2.26: Robô de inspeção de tubos.



Fonte: Adaptado de Kakogawa, Murata e Ma (2022, p. 2).

O robô proposto por Fjerdingen, Liljebäck e Transeth (2009), mostrado na Figura 2.27, foi projetado para ser capaz de navegar em lugares complexos. Isso inclui tantos tubos horizontais (através de um esquema que se assemelha a um trem) como verticais (alcançado através da extensão do duto, de modo alternado). A constituição deste modelo foi feita por uma série de módulos idênticos interconectados por duas juntas ativas de dois graus de liberdade, e ainda há um conjunto de rodas ativas em cada módulo que permitem a propulsão.

Figura 2.27: Robô de inspeção de tubos (PIKo).



Fonte: Adaptado de Fjerdingen, Liljebäck e Transeth (2009, p. 1).

No trabalho de Adams et al. (2018) foi projetado um sistema robótico flexível para ser utilizado na inspeção de serviços de água, conforme ilustrado na Figura 2.28. Esse dispositivo consiste de atuadores macios infláveis com raio ajustável. Quando pressurizados, podem movimentar o dispositivo no interior do tubo ou fixá-lo em um determinado lugar. Ao mesmo tempo, possui um sistema de câmera que permite capturar o movimento. Para controle deste equipamento foi utilizado um algoritmo que faz com que o robô manobre, no duto, de forma que o movimento seja similar ao de uma minhoca.

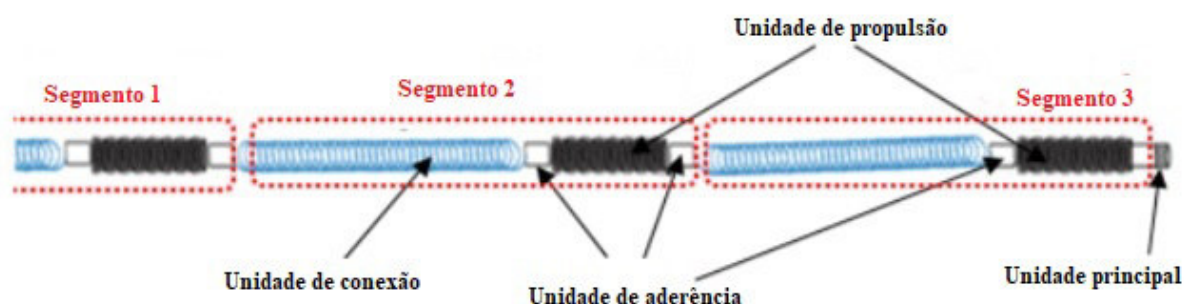
Figura 2.28: Robô de inspeção para tubulações de água.



Fonte: Adaptado de Adams et al. (2018, p. 1).

O trabalho produzido por Miyasaka, Kawano e Tsukagoshi (2018) consiste de uma ferramenta robótica de inspecionar tubulações, com diâmetro interno de 50 a 100 mm, sendo do tipo tubo flexível, acionado por pneumática. Tem como objetivo transportar uma câmera com fio para inspeção e um bocal para limpeza. É capaz de percorrer uma distância de até 50 m. Os autores propuseram uma configuração capaz de permitir que o robô, conforme ilustrado nas Figuras 2.29 e 2.30, realize a inspeção usando unidades de aderência para duas linhas diferentes e unidades de propulsão para outra linha, sendo todas conectadas em série.

Figura 2.29: Configuração do robô.



Fonte: Adaptado de Miyasaka, Kawano e Tsukagoshi (2018, p. 2).

Figura 2.30: Robô de inspeção de tubos.



Fonte: Adaptado de Miyasaka, Kawano e Tsukagoshi (2018, p. 5).

2.2 O SISTEMA ROS

Para o desenvolvimento do protótipo simulado deste trabalho, foi utilizado o ROS versão 1 (*Robot Operating System*). Segundo Quigley, Gerkey e Smart (2015), o ROS é uma infraestrutura de software livre para construção de sistemas robóticos. Essa plataforma permite que os usuários compartilhem código e ideias, de forma rápida, economizando tempo de desenvolvimento de software comuns aos sistemas robóticos. Além de ser um *framework* flexível, oferece várias ferramentas e bibliotecas para escrever comandos robóticos (JOSEPH; CACACE, 2018).

Especificamente para realizar as simulações necessárias a este trabalho, foram usados o Gazebo, e também as ferramentas *catkin*, *workspace*¹ e *ROS packages* (Pacotes ROS), que, entre outras características de software a serem adotadas, descritas em maior detalhe nas próximas seções.

2.2.1 CATKIN E WORKSPACE

2.2.1.1 CATKIN

Catkin é o sistema de construção do ROS, composto por um conjunto de instrumentos que o ROS pode usar para gerar programas executáveis, bibliotecas, scripts e interfaces que outros códigos podem também adotar (QUIGLEY; GERKEY; SMART, 2015). Segundo Quigley, Gerkey e Smart (2015), este sistema abrange um conjunto de macros do *CMake* (sistema de criação de código aberto) e *scripts Python* personalizados para proporcionar mais funcionalidades sobre o fluxo de trabalho normal do *CMake*. O autor ainda afirma que os usuários que utilizarem o *catkin*, precisam dominar apenas as informações contidas nos arquivos: *CMakeLists.txt* e *package.xml*.

2.2.1.2 WORKSPACE

O *workspace* é uma pasta que contém pacotes. Esses diretórios contêm os arquivos de origem e o ambiente ou espaço de trabalho que possibilitam compilar esses pacotes (MAHTANI et al., 2016). Segundo Quigley, Gerkey e Smart (2015), antes de desenvolver qualquer código ROS, se faz necessário configurar um *workspace* para o pleno funcionamento desse código. Este espaço de trabalho é simplesmente um conjunto de pacotes nos quais há um conjunto relacionado de código ROS (JOSEPH; CACACE, 2018).

Mahtani et al. (2016) destaca que um *workspace* é onde cada pasta é um espaço distinto com uma função diferente. O autor descreve estes espaços da seguinte maneira:

- *The source space*²: no espaço de origem (a pasta *src*), se colocam os pacotes, projetos, entre outros. Um dos arquivos mais importantes nesse espaço é o *CMakeLists.txt*. A pasta *src* possui esse arquivo porque é chamado pelo *cmake* quando

¹Na tradução em português é espaço de trabalho

²Na tradução em português é O espaço de origem

you configure the packages in the workspace. This file is created with the command `catkin_init_workspace`;

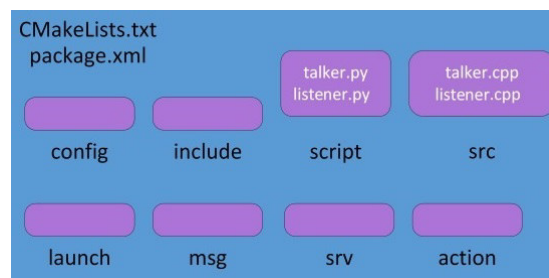
- *The build space*³: In the build folder, `cmake` and `catkin` maintain the cache, configuration and other intermediary files relative to the packages and projects;
- *Development (devel) space*⁴: the `devel` folder is used to keep the compiled programs. It is used to test the programs without the installation step. After the programs are tested, you can install or export the package to share it with other developers.

2.2.2 PACOTES ROS

ROS is organized in packages, where each one contains some combination of code, data and documentation. This system includes thousands of packages publicly available in open repositories (QUIGLEY; GERKEY; SMART, 2015).

A typical structure of a ROS package is visualized in Figure 2.31.

Figure 2.31: Typical structure of a ROS package in C++.



Fonte: Adaptado de Joseph e Cacace (2018, p. 13).

According to Joseph and Cacace (2018), the use of each folder is described in the following way:

- *config*: All the configuration files used in this ROS package are kept in this folder, created by the user. It is normally in this folder that the configuration files are kept;
- *include / package-name*: this folder consists of headers and libraries necessary to the package;
- *script*: this folder keeps *Python scripts* executable, which can be observed in the block diagram, where two *scripts* are shown as an example in Figure 2.31;
- *src*: this folder has the function of storing the C++ source code;

³Na tradução em português é O espaço de construção

⁴Na tradução em português é Espaço de desenvolvimento (devel)

- *launch*: essa pasta tem como objetivo manter os arquivos utilizados para inicializar um ou mais *nodos* ROS;
- *msg*: a pasta *msg* contém definições de mensagens personalizadas;
- *srv*: diferente da *msg*, a pasta *srv* contém as definições de serviço;
- *action*: esta pasta contém os arquivos de ação;
- *package.xml*: este acessório é o arquivo de manifesto do pacote;
- *CMakeLists.txt*: esse arquivo contém as diretrizes para compilar o pacote;

O autor ainda ressalta a necessidade de conhecer alguns comandos para criar, modificar e operar os pacotes descritos acima. Segundo ele, os comandos são:

- *catkin_create_pkg*: utilizado para criar novos pacotes;
- *rospack*: usado para obter informações sobre o pacote no conjunto de arquivos;
- *catkin_make*: utilizado para construir os pacotes *workspace*;
- *roscdep*: realiza a instalação das dependências dos sistemas necessários a este pacote.

2.2.3 TÓPICOS

O ROS consiste de vários *nodos* independentes, que são processos que executam computação (ROS.ORG, 2022). Segundo Quigley, Gerkey e Smart (2015), estes processos ficam interessantes quando eles se comunicam entre si, ocasionando troca de informações e dados, sendo comum mais tópicos para isso.

Segundo Mahtani et al. (2016), um tópico é o nome para um fluxo de mensagens com um tipo definido. No caso de um *nodo* enviar dados, podemos afirmar que ele está publicando um tópico, podendo receber tópicos de outros *nodos* (MAHTANI et al., 2016). Os tópicos ainda implementam um mecanismo de comunicação *publish/subscribe*, sendo umas das maneiras mais comuns de trocar de dados em um sistema distribuído (QUIGLEY; GERKEY; SMART, 2015). O autor ainda salienta que os *nodos*, antes de transmitir os dados sobre determinado tópico, devem primeiro anunciar o nome do tópico e o tipo de mensagem que serão enviadas, para em seguida, começarem a enviar ou publicar os dados reais dos tópicos (MAHTANI et al., 2016).

Para a proposta do trabalho foram utilizados cinco tópicos para implementação do robô na simulação computacional.

2.2.4 ARQUIVOS URDF E XACRO

A maneira que o ROS descreve o modelo de um robô, seja para simulá-lo ou simplesmente para ajudar os desenvolvedores em seu trabalho diário, é por meio dos arquivos URDF. Segundo Mahtani et al. (2016), este arquivo, cuja sigla significa *Unified Robot Description Format*, possui formato XML e descreve as características de um robô.

Os arquivos URDF são muito utilizados no ROS para caracterizar todos os elementos do robô. No entanto, não são capazes de identificar a posição do equipamento robótico em um ambiente, a força de atrito ou as uniões entre as juntas (ANDRADE, 2017). A autora ressalta a necessidade de adicionar algumas *tags* ao modelo URDF, para uma adequada simulação no Gazebo.

Segundo Koubaa (2016), o URDF adota uma estrutura semelhante a uma árvore para descrever a cinemática do sistema robótico. Por outro lado, o Gazebo usa o SDF (*Simulation Description Format*) que é uma especificação XML para descrever modelos robóticos, capaz de lidar com a representação gráfica da cinemática (KOUBAA, 2016).

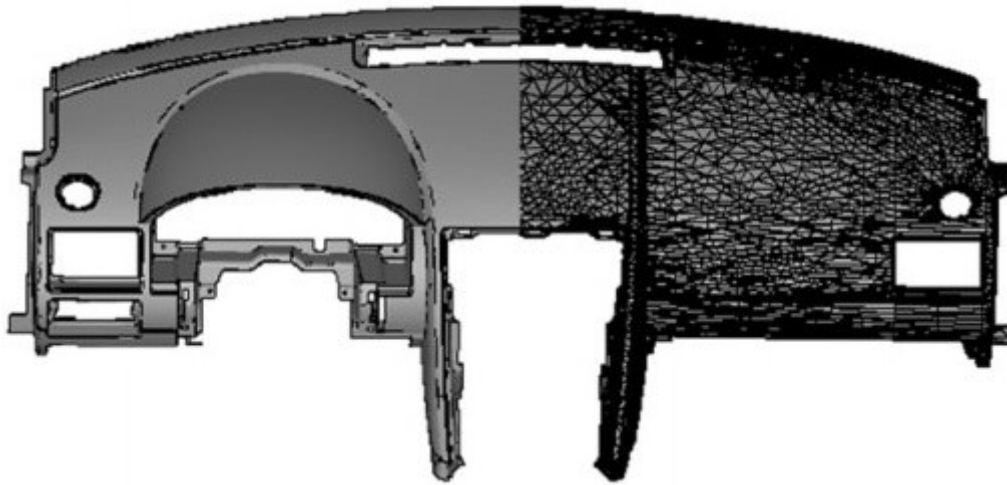
Apesar disso, para a elaboração deste trabalho, o URDF foi usado para a simulação do robô, e também o Xacro, um interpretador de macros. Esse interpretador executa todas as macros, gerando os resultados (ROS.ORG, 2018). Xacro é a abreviação de XML Macros. Ele ajuda a reduzir o tamanho geral do arquivo URDF, facilitando a leitura e a manutenção do mesmo. Permite criar módulos e reutilizá-los para descrever estruturas repetidas (MAHTANI et al., 2016). Apesar das sintaxes similares entre o URDF e o SDF, eles são distintos. As montagens dos ambientes propostos neste trabalho serão implementadas no Gazebo em formato SDF por suas vantagens em relação a outra extensão.

2.2.5 FORMATO STL E MESHES

Os métodos de apresentação usados para mostrar a geometria CAD (*Computer-Aided Design*) variam de um sistema para outro, tornando-se útil adotar uma interface padrão para transmitir descrições geométricas de vários pacotes CAD para sistemas de prototipagem rápida (CHUA; LEONG; LIM, 2003). Para a representação da geometria dos componentes do robô aqui proposto foi adotado o formato STL. Segundo Grimm (2004), o STL é um formato de arquivo binário ou ASCII utilizado como entrada para sistemas de prototipagem rápida, consistindo em uma lista não ordenada de facetas triangulares que representam superfície de um objeto (CHUA; LEONG; LIM, 2003). A Figura 2.32 ilustra a interpretação de um arquivo STL.

Na realização da simulação do equipamento robótico foram utilizados os arquivos STL, inseridos na pasta *meshes*. Esta pasta foi escolhida porque, segundo Joseph e Cacace (2018), ela mantém as malhas que precisamos incluir nos arquivos *URDF*. Um *mesh* é uma coleção de superfícies poligonais que nos fornecem uma forma mais realista para um objeto 3D (FAIRCHILD; HARMAN, 2016).

Figura 2.32: Exemplo de objeto no formato STL.

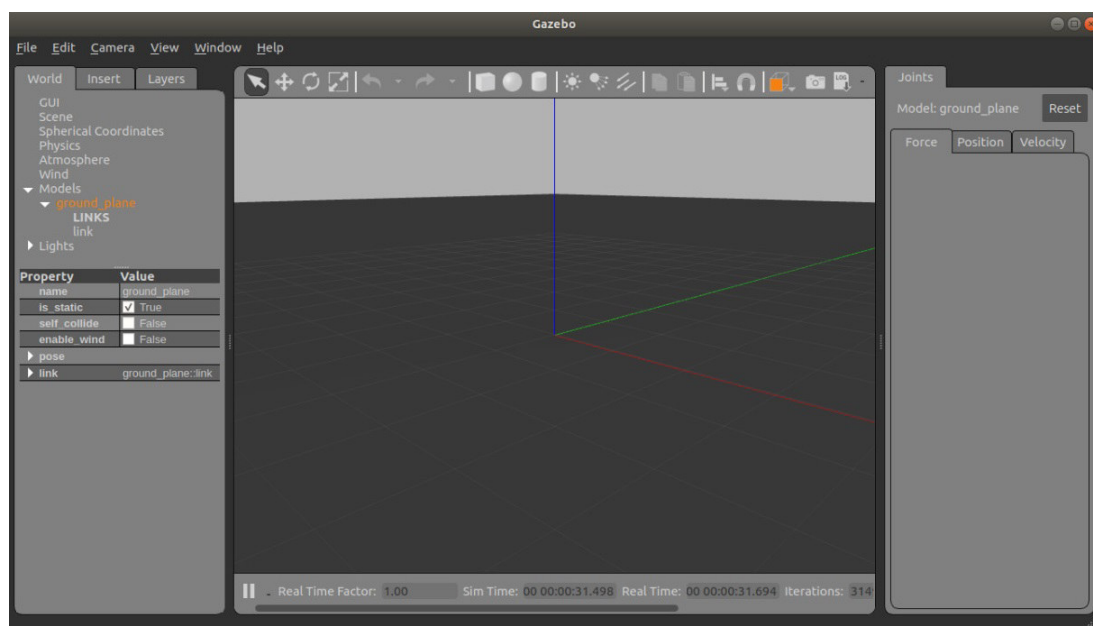


Fonte: Adaptado de Grimm (2004, p. 55).

2.3 GAZEBO

Para as simulações computacionais deste protótipo foi utilizado o software Gazebo, ambiente de simulação de robôs gratuita e de código aberto, fornecido pelo Willow Garage como um instrumento multifuncional para desenvolvedores de sistemas robóticos ROS (FAIRCHILD; HARMAN, 2016). O Gazebo é também um simulador multi-robótico para simulação interna e externa complexas, com possibilidades de simular robôs, sensores robóticos e uma variedade de objetos 3D (JOSEPH; CACACE, 2018). A interface deste programa pode ser visualizada na Figura 2.33.

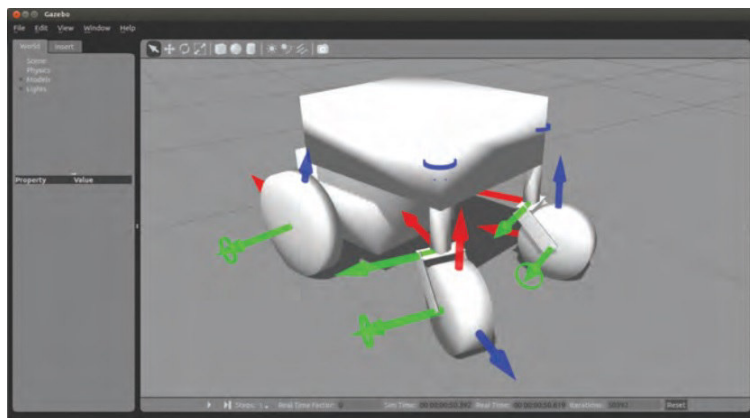
Figura 2.33: Interface do Gazebo.



Fonte: Autor.

O Gazebo, além de ser uma plataforma gratuita e de fornecer ferramentas de fácil construção de protótipos, também abre a possibilidade da interação ser realizada de forma simples com o ROS (KOU BAA, 2016). O esboço de um robô no Gazebo pode ser visualizado na Figura 2.34.

Figura 2.34: Exemplo de esboço de um robô no gazebo.



Fonte: Adaptado de Newman (2017, p. 142).

2.4 GROUND TRUTH

Primeiramente, para o entendimento do *ground truth*⁵ é importante compreender um pouco da odometria do robô. Segundo Mahtani et al. (2016), a odometria nada mais é do que a distância de uma posição do modelo robótico em relação à sua posição inicial. Estes dados são obtidos através da mensagem *nav_msgs/Odometry* no ROS (KOU BAA, 2016). Sendo assim, ela não é calculada através de equações complexas, mas apenas de utilização de posições fornecidas pelo simulador. Estas informações são chamadas de *ground truth* (EZQUERRO, 2017).

Na área da robótica, o *ground truth* é a medida mais precisa disponível, podendo vir direto de um simulador ou de um sensor de alta precisão (PETERSEN, 2022). Segundo Petersen (2022), no caso de robôs simulados, o *ground truth* é a posição exata do modelo robótico dentro da simulação computacional.

Para a proposta deste trabalho foi utilizado do *tópico ground truth*. Este proporciona a realização das medições das posições lineares e angulares do robô dentro do software Gazebo.

2.5 CINEMÁTICA

Segundo Craig (1989), a cinemática nada mais é do que a ciência do movimento que desconsidera análise de forças. Neste estudo são consideradas posição, velocidade,

⁵Na tradução em português é Verdade Fundamental

aceleração e todas as derivadas de ordem superior das variáveis de posição (em relação ao tempo ou a quaisquer outras variáveis) (CRAIG, 1989). Na robótica móvel, é importante entender o comportamento mecânico dos dispositivos com a finalidade de projetar equipamentos apropriados a realizar tarefas e o perfeito entendimento dos seus mecanismos de controle (SIEGWART; NOURBAKHSH; SCARAMUZZA, 2011). Esta pesquisa utilizou cinemática para obtenção das equações a fim de estudar o comportamento das variáveis para a aplicação da estratégia de controle.

2.6 CONTROLADOR PID

O controlador PID significa Proporcional-Integral-Derivativo, refere-se a soma de três parâmetros chamado de proporcional (K_p), de integral (K_i) e de derivativo (K_d), estas três variáveis dependem do erro e entre os parâmetros de entrada e saída (PHU et al., 2020). A sintonização correta destas variáveis irá melhorar a reação dinâmica de um sistema, erradicar o erro de estado estacionário, diminuir o *overshoot*⁶ e proporcionar uma melhor estabilidade do processo (SAAD; JAMALUDDIN; DARUS, 2012). O controlador PID possui algumas vantagens como: o ajuste fácil dos ganhos, a simplicidade de projetar e de possuir boa robustez (ZULU; JOHN, 2016).

A utilização da estratégia de controle PID em malha fechada consiste, segundo Ogata (2010) em um sistema de controle em malha fechada onde o sinal de erro atuante realimenta o controlador, de modo a minimizar o erro e acertar a saída do sistema ao valor determinado. Neste trabalho, o controlador PID recebeu as informações das posições e dos ângulos atualizados do robô ao longo do tempo, fornecidas pelo simulador Gazebo.

⁶Na tradução em português é Ultrapassar

3 MODELAGEM DO ROBÔ MÓVEL PARA NAVEGAÇÃO EM DUTOS DE SEÇÕES CIRCULARES E RETANGULARES

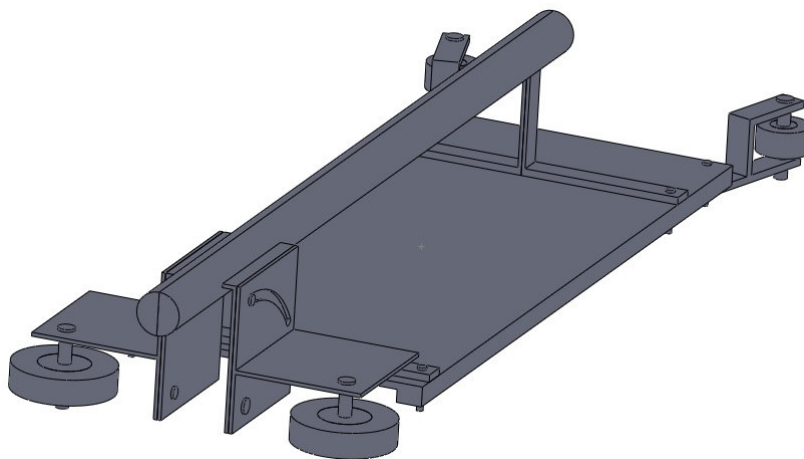
Para o entendimento detalhado do modelo proposto torna-se necessário apresentar uma descrição da modelagem do robô no que diz respeito aos aspectos dos ambientes de atuação, do modelo matemático dos dutos, e da estratégia de um controlador PID no controle das velocidades lineares e dos ângulos de ataque das rodas frontais, para cada um dos cenários de atuação. Todas essas características implicam nas definições de quais elementos e restrições devem-se levar em conta no desenvolvimento e simulação do robô nos locais de atuação determinados.

3.1 CRIAÇÃO DOS AMBIENTES DE NAVEGAÇÃO E DA SOLUÇÃO ROBÓTICA

Para o desenvolvimento do modelo computacional do ambiente de navegação foram realizados e considerados os seguintes procedimentos: primeiro, definiu-se que os testes foram executados em seções transversais circular e retangular; em seguida, foi admitido que para manter o mesmo robô nos dois locais de atuação, se fez necessário que o diâmetro do tubo circular tenha a mesma dimensão de um dos lados do tubo de seção retangular; após isso, foi determinado que estas seções transversais são sempre constantes ao longo do duto retilíneo de comprimento de 10 m; e por último se baseou nos métodos de criação detalhados na seção A.3 do Apêndice A.

Já no caso do modelo robótico, foi modelado o projeto TATUBOT para este sistema realizar movimento de translação e navegar nas partes inferiores e superiores dos tubos retangulares, e manter sua mobilidade em dutos circulares, podendo transladar e rotacionar. O dispositivo realiza navegação nestes dois ambientes de atuação devido ao ângulo de ataque ter o mesmo valor em módulo em ambas as rodas. Assim, de forma controlada, o robô poderá ser mantido em qualquer inclinação desejada, tanto na seção circular como na retangular. A Figura 3.1 mostra estas características.

Figura 3.1: CAD do robô.



Fonte: Autor.

Para o desenvolvimento da simulação computacional do robô, o trabalho se baseou nas etapas de construção da seção A.4 do Apêndice A, e na literatura de Mello-Jr et al. (2008).

Após as etapas de criação dos ambientes e do robô separados, parte-se então para a fase de ativar o sistema dentro dos ambientes de atuação e ainda permitir seu controle. O procedimento segue os seguintes passos: primeiro, se criou o diretório *catkin_workspace* para configurar os Pacotes ROS, permitindo assim iniciar a parte computacional do espaço de trabalho. Esta etapa foi baseada nas seções A.1 e A.2 do Apêndice A. Em seguida, se inseriu o espaço de estrutura físicas e visuais do robô, utilizando dos arquivos *urdf* e *meshes*, conforme detalha a seção A.3 do Apêndice A. E por último foram construídos os espaços de configurações (*config*¹) e de ativação (*launch*²) do pacote, seguindo os métodos da seção A.5 do Apêndice A. No caso destes dois últimos arquivos é onde ocorre a definição dos *tópicos* que serão usados para manusear o sistema robótico, eles são: das variáveis controladas (ω e γ), e do *ground_truth*. Após todos estes parâmetros definidos é gerado o primeiro pacote chamado de *tatubot.ni*.

O pacote *tatubot.ni* é a parte de estrutura do robô e dos ambientes de atuação. Então, junto com este diretório, foram criados mais dois pacotes: o primeiro é *simple_controller*, desenvolvido para o algoritmo de etapas, determinar os comandos para os alvos do equipamento robótico nos cenários estabelecidos; e o segundo pacote criado foi para localização do robô no Gazebo na qual é chamado de *my_odom_publisher*. *My_odom_publisher* é para gera o tópico *ground_truth* que atualizará as posições do modelo robótico no espaço. Estes dois pacotes citados foram construídos baseados na seção A.6 do Apêndice A.

Deste modo, com todas estas características determinadas dos ambientes e do robô, e o desenvolvimento da sua simulação já é possível apresentar os modelos cinemáticos e

¹Na tradução em português é Configuração

²Na tradução em português é Ativação

suas estratégias de controle para cada um dos casos citados anteriormente.

3.2 ASPECTOS GERAIS DO MODELO CINEMÁTICO E ESTRATÉGIA DE CONTROLE

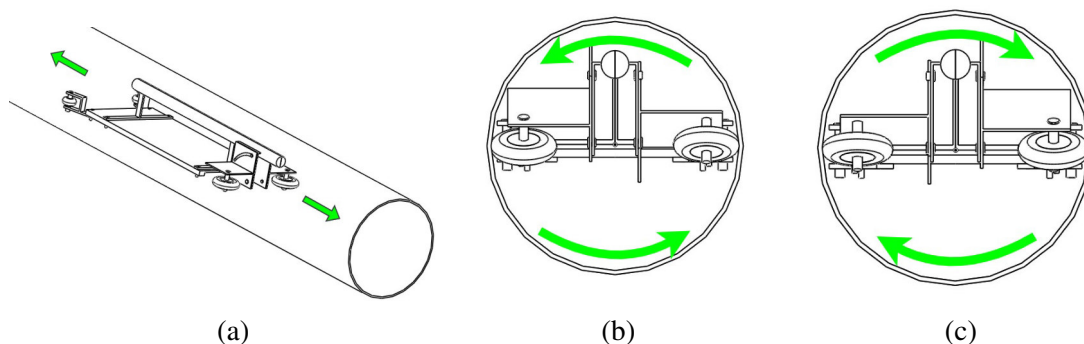
Para analisar a navegação do robô no interior dos tubos, foram desenvolvidos modelos matemáticos e ações de controle PID para entender o funcionamento dos dispositivos. O modelo foi baseado nas representações apresentadas nas Figuras 3.2 e 3.3, que ilustram o sistema de coordenadas convencional e também o robô no interior dos tubos circular e retangular. O equacionamento será realizado utilizando os princípios da cinemática.

O modelo cinemático tem como objetivo o entendimento das variáveis controladas (no caso das velocidades angulares e dos ângulos de ataque das rodas frontais). Assim, é permitido realizar simulações computacionais, a fim de analisar o comportamento deste sistema robótico quando a estratégia de controle for utilizada.

Seguindo o raciocínio anterior, no caso da navegação em dutos de seção circular, as rodas da frente terão os ângulos de ataque com o mesmo valor em módulo, porém com sentidos opostos. Isso faz o robô avançar e rotacionar em relação ao eixo estabelecido y . Já na situação da seção retangular, os ângulos de ataque são iguais e deverão ter o mesmo sentido, propiciando que o equipamento se desloque no plano desejado. As Figuras 3.2 e 3.3 mostram a vista frontal deste modelo robótico nos dois cenários propostos.

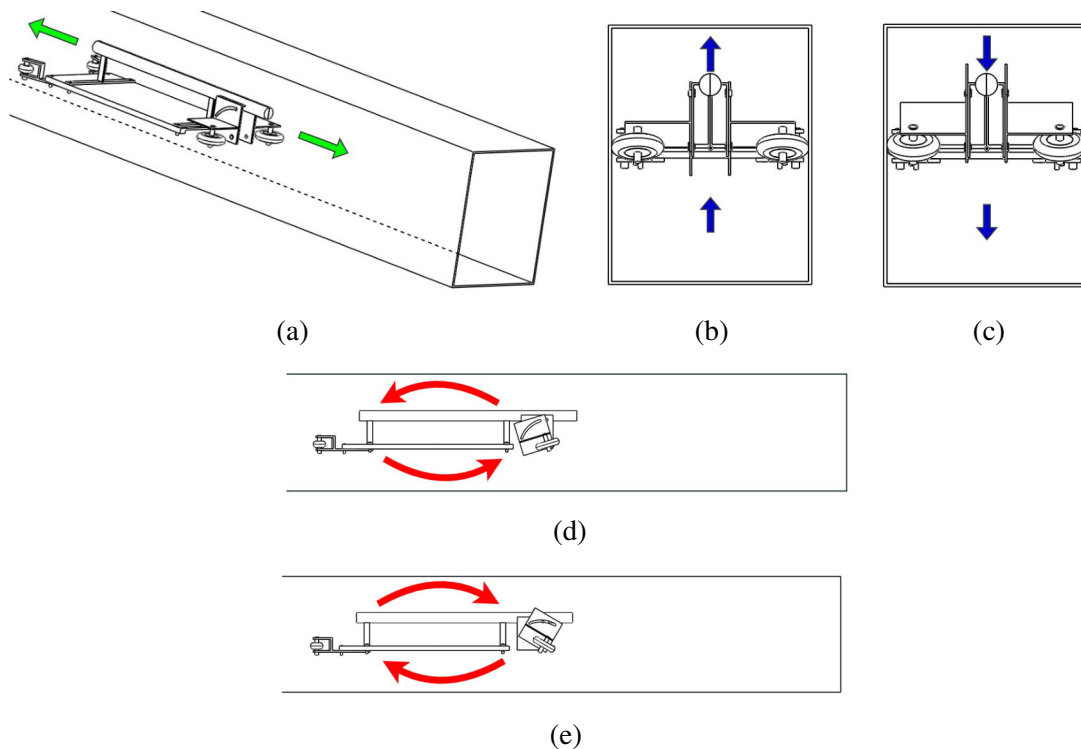
Nos estudos realizados foi usado o plano yz para ambos os ambientes, pois usar este plano faz com que tanto tubo circular quanto retangular atendam o sistema convencional de sinais onde se utiliza: direita (na horizontal), cima (na vertical) e a regra da mão direita como positivo). No caso do duto de seção retangular, o robô irá se deslocar nos eixos y e z , e rotacionar no eixo x .

Figura 3.2: (a) representa o robô na vista isométrica, onde ele poder se mover tanto para frente quanto trás; (b) representa o modelo robótico na vista frontal rotacionando no sentido anti-horário; e (c) representa o dispositivo na vista frontal girando no sentido horário.



Fonte: Autor.

Figura 3.3: (a) o robô está no plano isométrico podendo se locomover tanto para frente quanto para trás; (b) o equipamento está no plano frontal e realiza o movimento para a parte superior do tubo; (c) o dispositivo também está no plano frontal só que está se dirigindo para a parte inferior do tubo; (d) o robô está na vista lateral esquerda rotacionando no sentido anti-horário, com centro nas rodas traseiras, e se dirigindo para a parte superior; e (e) o sistema robótico está na vista lateral esquerda e rotaciona no sentido horário, com centro nas rodas traseiras, e se locomovendo para a parte inferior.



Fonte: Autor.

Para os dois ambientes são levadas em conta algumas hipóteses simplificadoras como:

- Admite-se um dispositivo de corpo rígido;
- Rodas que não possuem deformações;
- No tubo circular, deslocamento linear e rotação em relação ao eixo y .
- No ambiente retangular, deslocamentos lineares em y e z , e rotação em relação ao eixo x .

3.3 MODELAGEM CINEMÁTICA E ESTRATÉGIA DE CONTROLE DO ROBÔ PARA OS AMBIENTES CIRCULAR

3.3.1 MODELO CINEMÁTICO

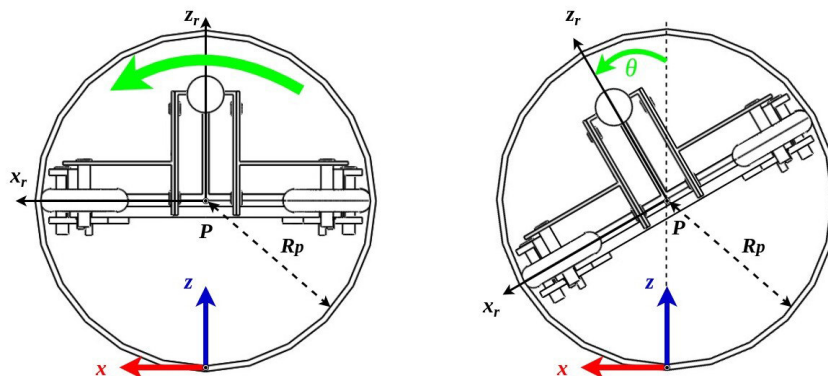
O cenário de tubos circulares baseia-se nos referenciais apresentados nas Figuras 3.4 (representa o sistema de coordenadas fixo no corpo do robô em movimento de rotação, da

vista frontal); 3.5 (representa o sistema de referência da parte superior do dispositivo no duto); 3.6 (apresenta as coordenadas do equipamento robótico na vista lateral esquerda); e 3.7 (apresenta o esquemático das rodas frontais do robô no plano yz); e no trabalho de Kwon et al. (2011).

As Figuras 3.4, 3.5 e 3.6 apresentam o robô no tubo e o sistema de coordenadas estabelecido. Já a Figura 3.7 descreve e detalha a variação do ângulo de ataque da roda frontal, onde as variáveis do sistema são:

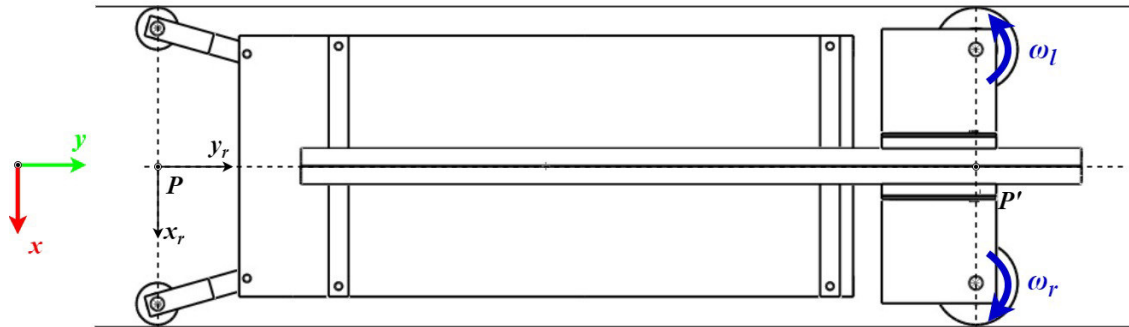
- l - distância entre o ponto P e P' .
- P - centro do referencial do robô, localizado nas rodas traseiras;
- P' - ponto de referência localizado nas rodas frontais;
- R_p - raio do duto;
- r_w - raio da roda do robô;
- v - velocidade linear do robô;
- v_y - velocidade na componente y ;
- (x, y, z) - posição do referencial fixo no plano;
- (x_r, y_r, z_r) - posição do referencial fixo no corpo do robô;
- γ - ângulo de ataque das rodas frontais;
- θ - ângulo de orientação do robô em relação ao eixo y ;
- ω_r - velocidade angular da roda direita frontal;
- ω_l - velocidade angular da roda esquerda frontal;

Figura 3.4: Sistema de coordenadas cartesianos xyz vista frontal



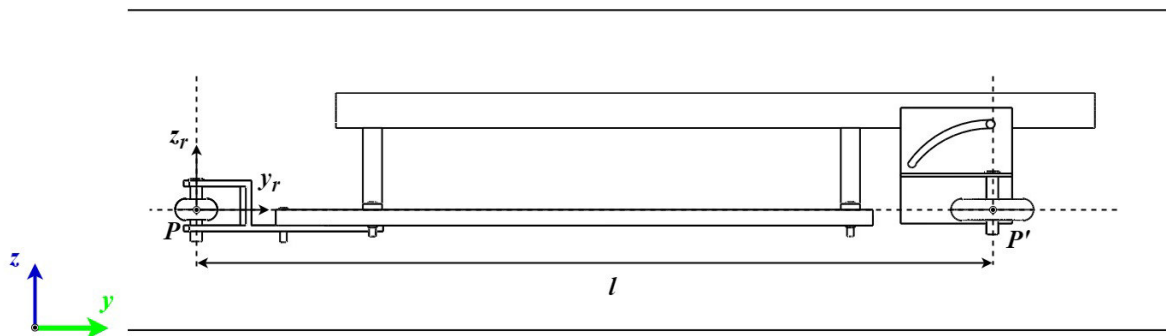
Fonte: Autor.

Figura 3.5: Sistema de coordenadas cartesianos xyz vista superior



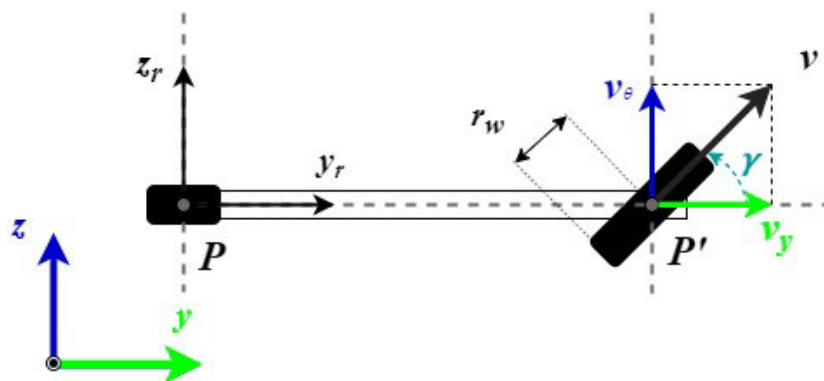
Fonte: Autor.

Figura 3.6: Sistema de coordenadas cartesianos xyz vista lateral esquerda



Fonte: Autor.

Figura 3.7: Esquemático das rodas frontais do robô

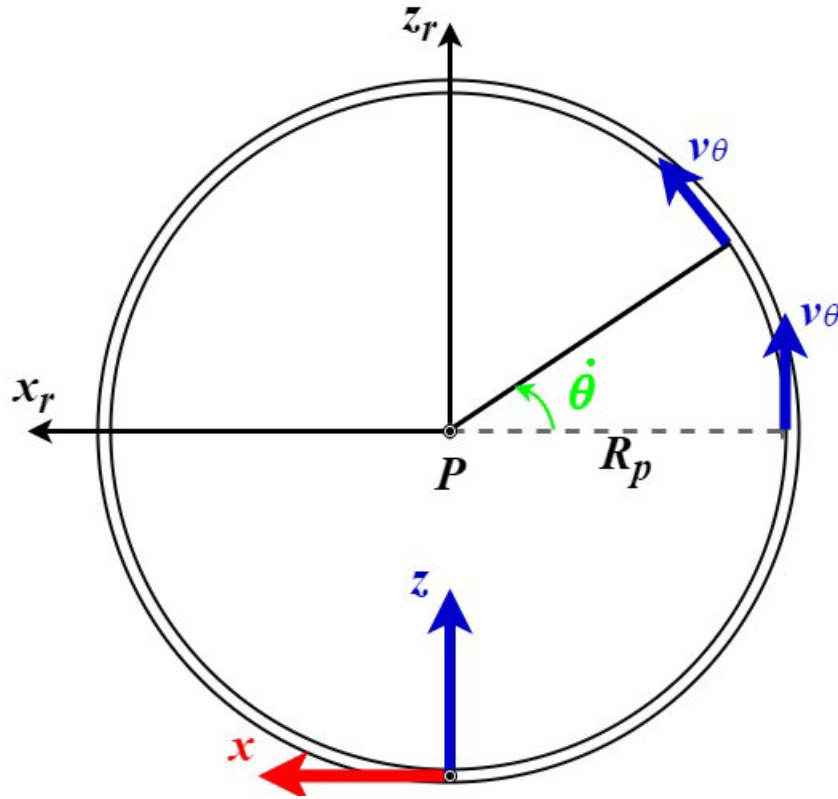


Fonte: Autor.

Determinado os sistemas de referenciais do robô, agora será mostrado outro esquemático da modelagem cinemática, ilustrado na Figura 3.8, onde as variáveis são:

- $\dot{\theta}$ - velocidade angular do robô;
- v_{θ} - vetor auxiliar.

Figura 3.8: Esquemático da vista frontal do robô.



Fonte: Autor.

O sistema robótico representado nas Figuras 3.5 e 3.6 possui um conjunto de rodas pressionadas por sistema de molas contra a parede permitindo que o mesmo esteja sempre alinhado a uma corda imaginária, que cruza o centro do duto. As rodas frontais terão sempre o ângulo de ataque (γ) de forma simétrica ($\gamma_r = -\gamma_l$). Assim, os movimentos realizados pelo dispositivo robótico serão de translação e rotação no eixo y , os movimentos nos demais eixos serão desconsiderados.

Segundo Kwon et al. (2011), para garantir os movimentos, as velocidades lineares das rodas frontais terão suas magnitudes sempre iguais ($|v_{wr}| = |v_{wl}|$). Então suas Equações são dadas como:

$$v_{wr} = r_w \omega_r = v_{wl} = r_w \omega_l \quad (1)$$

Onde o raio (r_w) para ambas as rodas é o mesmo, então temos:

$$v = \frac{v_{wr} + v_{wl}}{2} \quad (2)$$

$$v = v_{wr} = v_{wl} \quad (3)$$

$$\omega = \omega_r = \omega_l \quad (4)$$

A componente da velocidade em y é ilustrada na Figura 3.7, definida pelas Equações:

$$v_y = v \cos \gamma \quad (5)$$

Para determinar a componente em z foi usado o vetor auxiliar v_θ , mostrado na Figura 3.7. Assim, temos:

$$v_\theta = v \sin \gamma \quad (6)$$

Analisando a Figura 3.8, v_θ também pode ser determinado por:

$$v_\theta = \dot{\theta} R_p \quad (7)$$

Substituindo (7) em (6), temos:

$$\dot{\theta} = \frac{v \sin \gamma}{R_p} \quad (8)$$

Considerando um curto intervalo de tempo, as Equações (5) e (8) podem ser reescritas como:

$$\frac{\Delta y}{\Delta t} = v \cos \gamma \quad (9)$$

$$\frac{\Delta \theta}{\Delta t} = \frac{v \sin \gamma}{R_p} \quad (10)$$

Substituindo (1) em (9) e (10), obtém-se:

$$\frac{\Delta y}{\Delta t} = \omega r_w \cos \gamma \quad (11)$$

$$\frac{\Delta \theta}{\Delta t} = \frac{\omega r_w \sin \gamma}{R_p} \quad (12)$$

Isolando Δy na Equação (11) e $\Delta \theta$ na Equação (12), resulta em:

$$\Delta y = \omega r_w \Delta t \cos \gamma \quad (13)$$

$$\Delta \theta = \frac{\omega r_w \Delta t \sin \gamma}{R_p} \quad (14)$$

3.3.2 CONTROLADOR PID

Está proposta foi baseada na literatura de Ogata (2010), e definida a partir da determinação das variáveis controladas que são: a velocidade angular das rodas frontais (ω) e o ângulo de ataque (γ). Após isso, é feito um estudo do comportamento das variáveis (y , θ) em relação as variáveis controladas, através das Equações (13) e (14), observando que r_w e R_p são constantes.

Analisando a Equação (13) pode-se observar que a variação no deslocamento linear em y ocorre, principalmente, devido a variação da velocidade angular das rodas (ω). Variações com o cosseno do ângulo de ataque γ também a afetam, mas de forma menos significativa pois o γ varia lentamente. Dessa forma, pode-se concluir que a malha de controle do deslocamento linear do robô possui fraco acoplamento com a variável controlada γ . Logo, uma lei de controle PID clássica pode ser utilizada para o controle do deslocamento linear em y .

Na Equação (14), pode-se observar que a variação no deslocamento angular em θ é ocasionada devido as variações do ângulo de ataque das rodas frontais (γ) e da velocidade angular (ω). Assim, considerando uma velocidade angular das rodas (ω) constante, a variação do ângulo (θ) dependerá apenas da variação do ângulo de ataque das rodas (γ). Logo, uma lei de controle PID clássica pode ser utilizada para controle do deslocamento angular em θ .

A partir da análise das Equações (13) e (14), é possível considerar que o sistema possui fraco acoplamento entre as variáveis a serem controladas. Isso permite simplificar o sistema e tratar o controle das variáveis y e θ por dois controladores independentes. A Figura 3.9 é apresentado um diagrama de blocos para o controle das variáveis y e θ , utilizando controladores PID, para o caso do tubo circular.

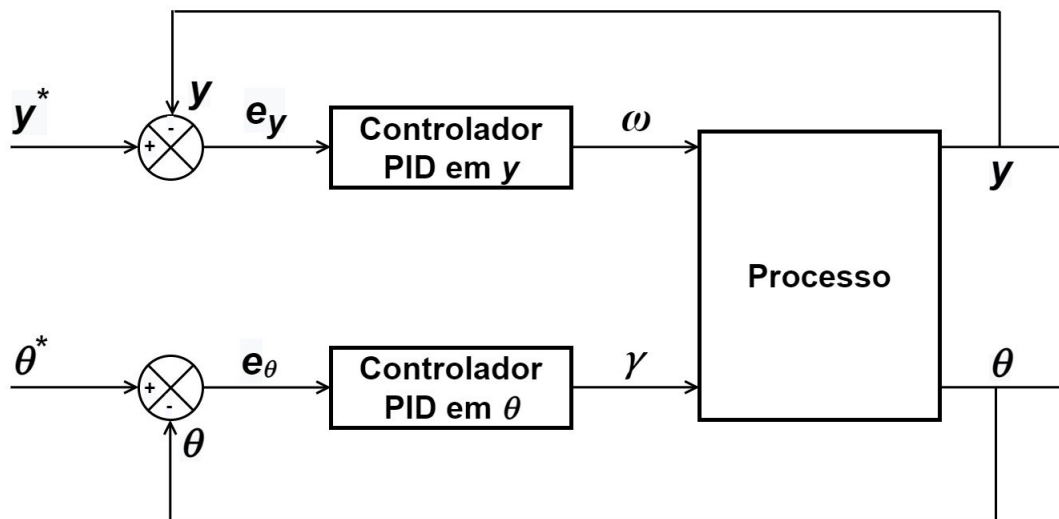


Figura 3.9: Diagrama de blocos para o controle no tubo circular.

Fonte: Autor

No tubo circular o ângulo de ataque (γ) varia em mesmo módulo e sentido oposto, respeitando um intervalo de -14° à 14° (estes valores de γ e o intervalo de $-3,5$ rad/s à $3,5$ rad/s de ω foram determinados a partir de testagens das rodas anteriores a simulação, e neste valores o robô apresentou os seus melhores desempenhos).

Segundo Rossomando e Soria (2015), os blocos dos controladores são definidos pela ação de controle do PID discretizado representado através da Equação 15:

$$u_{(k)} = u_{(k-1)} + K_P[e_{(k)} - e_{(k-1)}] + K_i T_s e_{(k-1)} + \frac{K_D}{T_s} [e_{(k)} - 2e_{(k-1)} + e_{(k-2)}] \quad (15)$$

As variáveis são: u (sinal de saída do controlador); e (sinal de erro atuante); K_p (ganho proporcional); K_i (ganho integrativo); K_d (ganho derivativo); T_s (tempo de amostragem); e k (tempo discreto).

Com base na Equação (15), as equações para as variáveis controladas assumem estas expressões:

$$\omega_{(k)} = \omega_{(k-1)} + K_{py}[e_{y(k)} - e_{y(k-1)}] + K_{iy} T_s e_{y(k-1)} + \frac{K_{dy}}{T_s} [e_{y(k)} - 2e_{y(k-1)} + e_{y(k-2)}] \quad (16)$$

$$e_{y(k)} = y^* - y_{(k)} \quad (17)$$

$$\gamma_{(k)} = \gamma_{(k-1)} + K_{p\theta}[e_{\theta(k)} - e_{\theta(k-1)}] + K_{i\theta} T_s e_{\theta(k-1)} + \frac{K_{d\theta}}{T_s} [e_{\theta(k)} - 2e_{\theta(k-1)} + e_{\theta(k-2)}] \quad (18)$$

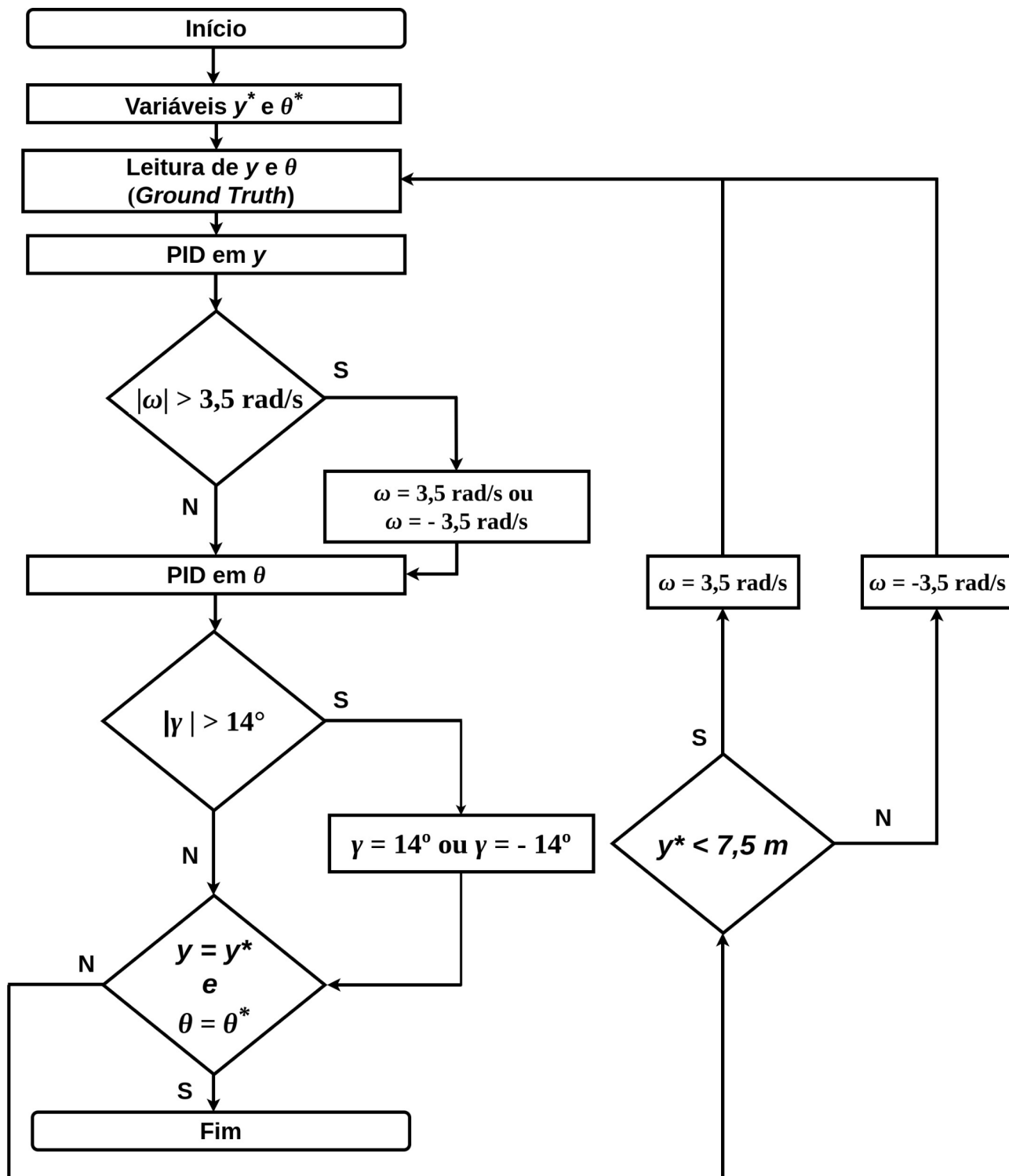
$$e_{\theta(k)} = \theta^* - \theta_{(k)} \quad (19)$$

As Equações (17) e (19) apresentam os erros do sistema de cada caso, onde y^* e θ^* são as variáveis de referência que terão seus valores determinados pelo usuário. Por sua vez, as variáveis y e θ serão atualizadas instantaneamente, ocasionando ações no processo, como mostram as Equações (16) e (18).

Com estes estudos realizados, é possível entender como deve ser o funcionamento do fluxograma de deslocamento de tarefas aplicado para a navegação do robô no tubo circular, representado na forma de fluxograma na Figura 3.10. O algoritmo inicia pela leitura das variáveis e dos erros, gerando um novo valor para a velocidade angular e o ângulo de ataque das rodas frontais. Após um instante de tempo é calculado um novo valor para estas variáveis. Foi estabelecida uma restrição de segurança: quando o alvo estiver no intervalo de 0 à 7,5m o robô irá executar o movimento de avanço, e quando for

valor superior à 7,5m, o modelo robótico realizará o movimento de ré no duto, porque as leituras da variável y são feitas nos sistemas de coordenadas $x_r y_r z_r$ (como é mostrado na Figura 3.7), assim valores inferiores à 0 m e superiores à 9 m fazem o equipamento robótica sair para fora do tubo.

Figura 3.10: Fluxograma de deslocamento de tarefa no tubo circular.



Fonte: Autor

3.4 MODELAGEM CINEMÁTICA E ESTRATEGIA DE CONTROLE PID DO ROBÔ PARA OS AMBIENTES RETANGULAR

3.4.1 MODELO CINEMÁTICO

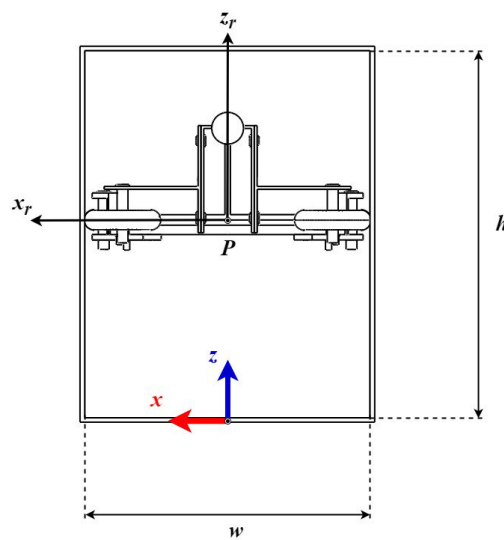
Para a cinemática do robô no interior do tubo de seção retangular, serão levadas em conta as mesmas considerações simplificadoras adotadas acima. Para este caso, o dispositivo também se utilizará das suas rodas tracionadas para manter estabilidade e executar o movimento em linha reta, podendo se deslocar na parte superior ou inferior do duto, dentro do plano yz , possibilitando uma rotação em torno do eixo x (ϕ). Este movimento só é permitido quando os dois ângulos de ataque das rodas frontais (γ) forem sempre iguais ($\gamma_r = \gamma_l$).

Com todos estes dados determinados, a modelagem se baseou nos esquemáticos das Figuras 3.5 e 3.6 e 3.11, e na adaptação da obra de Zheng e Li (2022). Para especificar a posição do robô no plano se estabelecem o eixo de coordenadas fixo xyz e o sistema de referência $x_r y_r z_r$, este localizado nas rodas traseiras (Ponto P) do dispositivo robótico, como ilustram as Figuras 3.5 e 3.6 e 3.11. Assim, escrevemos as variáveis:

- C - centro de rotação;
- h - altura da seção retangular;
- l - distância entre o ponto P e P' ;
- P - centro do referencial do robô, localizado nas rodas traseiras;
- P' - ponto de referência localizado nas rodas frontais;
- R - raio de rotação, medido entre o ponto C até o ponto P ;
- r_w - raio da roda do robô;
- v - velocidade linear do robô;
- v_y - componente da velocidade em y do robô;
- v_z - componente da velocidade em z do robô;
- v_w - velocidade da componente em y da roda do robô;
- v_{wy} - componente da velocidade em y da roda do robô;
- v_{wz} - componente da velocidade em z da roda do robô;
- w - largura da seção retangular;

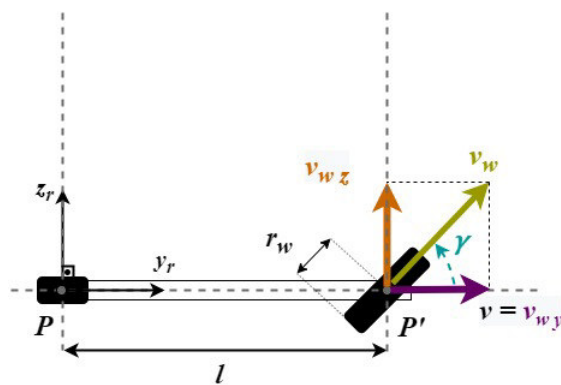
- (x, y, z) – posição do referencial fixo no plano;
- (x_r, y_r, z_r) – posição do referencial fixo no corpo do robô;
- γ - ângulo de ataque das rodas frontais;
- ϕ - ângulo de orientação do robô em relação ao eixo x ;
- $\dot{\phi}$ - velocidade angular do robô;
- ω_r - velocidade angular da roda direita frontal;
- ω_l - velocidade angular da roda esquerda frontal;

Figura 3.11: Sistema de coordenadas cartesianos xyz vista frontal



Fonte: Autor.

Figura 3.12: Esquemático das rodas do robô.



Fonte: Autor

Para os movimentos no tubo é necessário que as velocidades lineares das rodas frontais possuam magnitudes iguais ($|v_{wr}| = |v_{wl}|$). E de acordo com a Figura 3.12, a relação entre v e as velocidades das rodas (v_{wr} e v_{wl}) são:

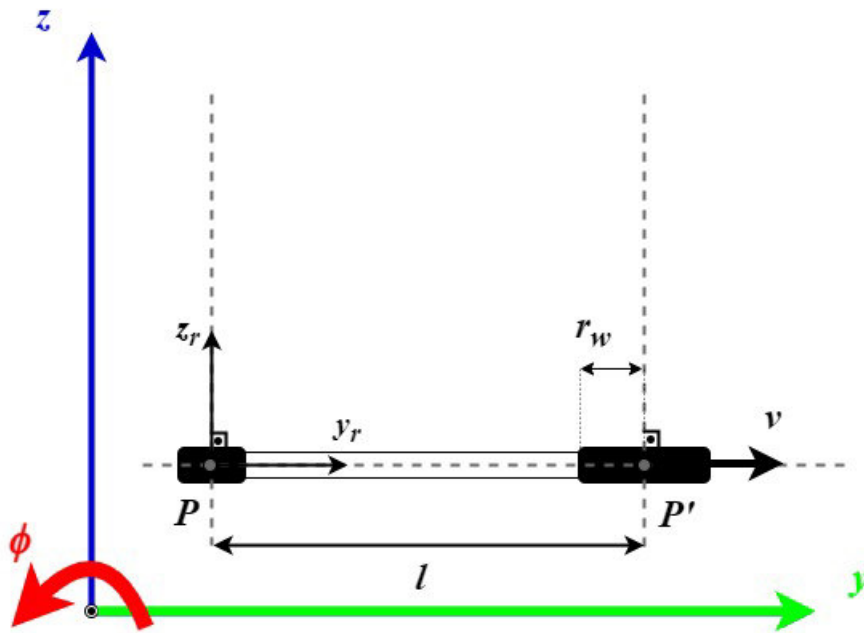
$$v_{wr} = v_{wl} = v_w = \omega_r r_w = \omega_l r_w = \omega r_w \quad (20)$$

$$v = v_{wy} = v_w \cos \gamma \quad (21)$$

Assume-se que o equipamento robótico representado na Figura 3.6 e 3.12 possa se locomover de forma linear (nos eixos y e z) ou rotacionar em relação ao eixo de coordenadas x . As etapas para o equacionamento são:

1. Determinado os sistemas de coordenadas do robô, traçam-se linhas verticais e perpendiculares as rodas, e também uma linha horizontal nos pontos P e P' , como mostra o esquemático ilustrado na Figura 3.13:

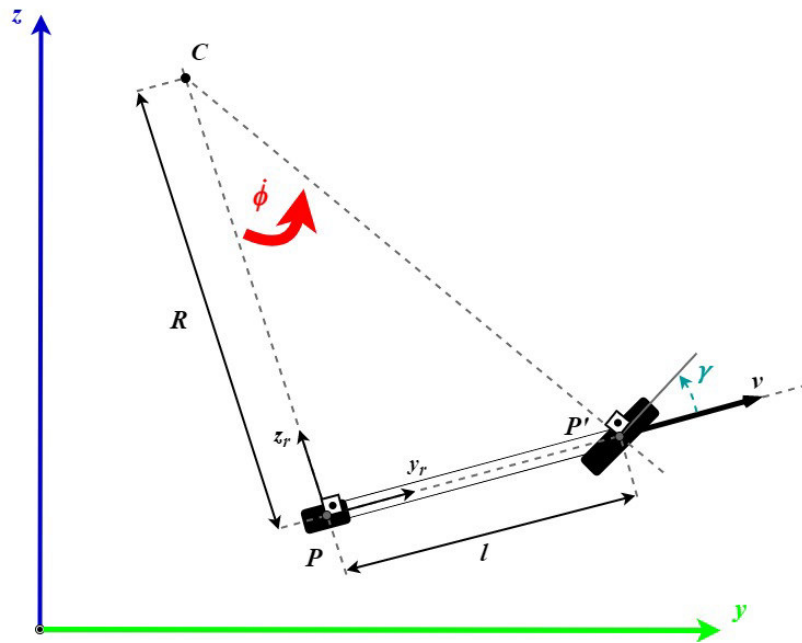
Figura 3.13: Esquemático do robô no plano yz - Etapa A



Fonte: Autor

2. Após este procedimento, gera-se um ângulo de ataque nas rodas frontais (γ , medido a partir do sistema de coordenadas fixo no robô $x_r y_r z_r$). O que ocasiona uma variação do ângulo de rotação (ϕ , medido a partir da horizontal e pelo sistema de referências fixo global xyz), em torno do eixo x . Assim, as retas perpendiculares das rodas se cruzam, e formam o centro de rotação (C) e o raio (R) entre os pontos C e P . A Figura 3.14 ilustra isso:

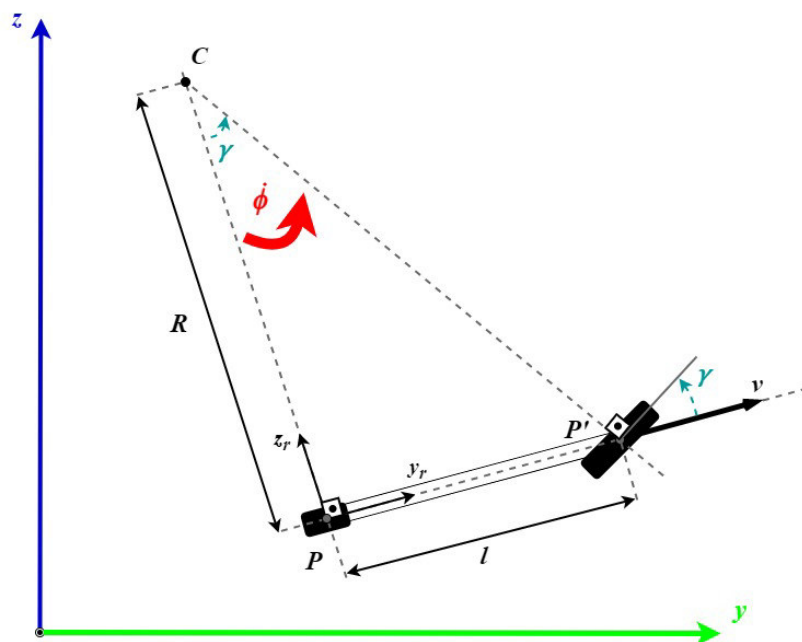
Figura 3.14: Esquemático do robô no plano yz - Etapa B



Fonte: Autor

3. O ângulo (localizado no vértice C) é igual ao ângulo de ataque (γ). Para comprovar isto usamos o triângulo retângulo PCP' , demonstrado na Figura 3.15. O ângulo (do vértice P') é $90^\circ - \gamma$, porque é formado um ângulo raso (180°) no prolongamento do eixo y_r . A soma dos ângulos internos de um triângulo é de 180° , assim o ângulo (do vértice C) é igual ao ângulo de ataque das rodas frontais (γ).

Figura 3.15: Esquemático do robô no plano yz - Etapa C



Fonte: Autor

A velocidade linear v de um objeto que realiza movimento circular, conforme mostra a Figura 3.15, é definido como:

$$v = \dot{\phi} R \quad (22)$$

Observando a Figura 3.15 pode-se verificar que é formado um triângulo retângulo CPP' . Deste triângulo, podemos encontrar a relação entre γ , l e R :

$$\tan \gamma = \frac{l}{R} \quad (23)$$

Da relação obtida na Equação (23) isolamos a variável R ..

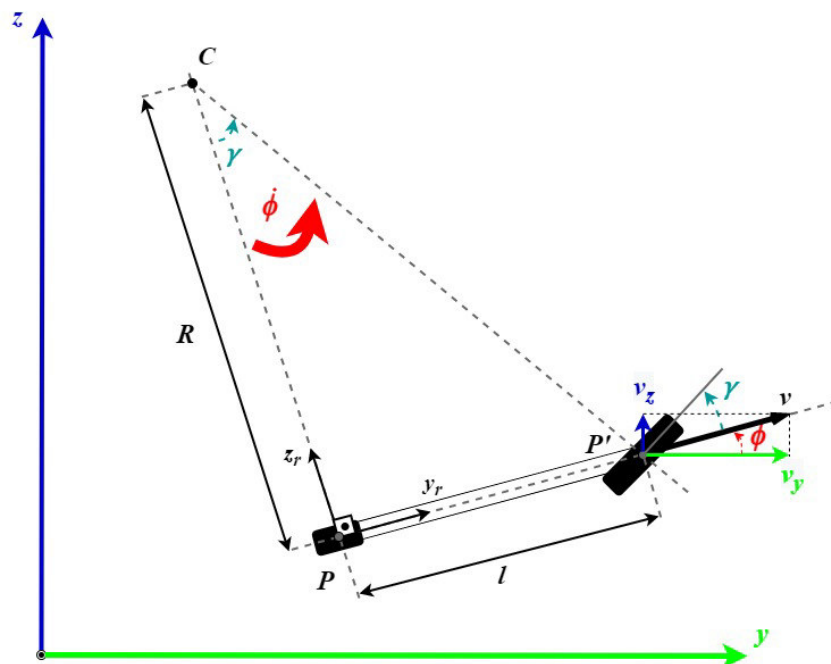
$$R = \frac{l}{\tan \gamma} \quad (24)$$

Assim, a velocidade angular pode ser expressa como:

$$\dot{\phi} = \frac{v \tan \gamma}{l} \quad (25)$$

4. O ângulo de rotação do robô (ϕ) é medido com os sistemas coordenadas fixo no plano xyz . Com sua variação, gera-se duas componentes da velocidade linear do robô (v), uma em relação à y e outra em relação à z . Como detalha a Figura 3.16:

Figura 3.16: Esquemático do robô no plano yz - Etapa D



Fonte: Autor

Segundo a Figura 3.16 as componentes das velocidades v em y e z são:

$$v_y = v \cos \phi \quad (26)$$

$$v_z = v \sin \phi \quad (27)$$

Para um estudo seguinte, as Equações (26) e (27) precisam estar em um curto intervalo de tempo. Então, considerando um curto intervalo de tempo estas Equações, ficam:

$$\frac{\Delta y}{\Delta t} = v \cos \phi \quad (28)$$

$$\frac{\Delta z}{\Delta t} = v \sin \phi \quad (29)$$

Lembrando que a Equação (21) é $v = \omega r_w \cos \gamma$, pode-se então substituir nas Equações (28) e (29), onde tem-se:

$$\frac{\Delta y}{\Delta t} = \omega r_w \cos \gamma \cos \phi \quad (30)$$

$$\frac{\Delta z}{\Delta t} = \omega r_w \cos \gamma \sin \phi \quad (31)$$

Isolando Δy e Δz nas equações acima, resultam em:

$$\Delta y = \omega r_w \Delta t \cos \gamma \cos \phi \quad (32)$$

$$\Delta z = \omega r_w \Delta t \cos \gamma \sin \phi \quad (33)$$

3.4.2 CONTROLADOR PID

A partir da análise da Equação (32) pode-se concluir que variações em y ocorrem, basicamente, devido a variações em ω . Variações em γ e ϕ podem ser consideradas como distúrbios pois variam lentamente. Da mesma forma, analisando a Equação (33), para a variável z . Assim, considerando que as dinâmicas de γ e ϕ são mais lentas que as dinâmicas controladas, o sistema pode ser considerado fracamente acoplado e passível de utilização de controladores do tipo PID. Assim, o diagrama de blocos que representa o controle do sistema em um tubo retangular é apresentado na Figura 3.17. As variáveis controladas são as posições em y e z . As ações de controle são sobre as variáveis ω e γ , respectivamente.

Com estas análises feitas, é possível utilizar uma estratégia de controle PID independente para cada variável. E também, como no caso do duto circular, a variável γ (no tubo retangular) nunca vai ultrapassar o intervalo -14° à 14° . Porém diferente do cenário cir-

cular, a variável γ possui módulos e sentidos iguais. Já a variável ω tem um intervalo de $-3,5$ rad/s à $3,5$ rad/s. E por fim, uma outra observação a ser feita foi em relação a variável ϕ que foi utilizado como ângulo de projeto contendo um intervalo de -5° à 5° , pois o robô neste trabalho não foi modelado para se locomover em direções verticais.

A lei de controle discretizada será a mesma da Equação (15), abordada no capítulo anterior. Então, as equações para as variáveis controladas assumem as expressões:

$$\omega(k) = \omega(k-1) + K_{py}[e_y(k) - e_y(k-1)] + K_{iy}T_s e_y(k-1) + \frac{K_{dy}}{T_s}[e_y(k) - 2e_y(k-1) + e_y(k-2)] \quad (34)$$

$$e_y(k) = y^* - y(k) \quad (35)$$

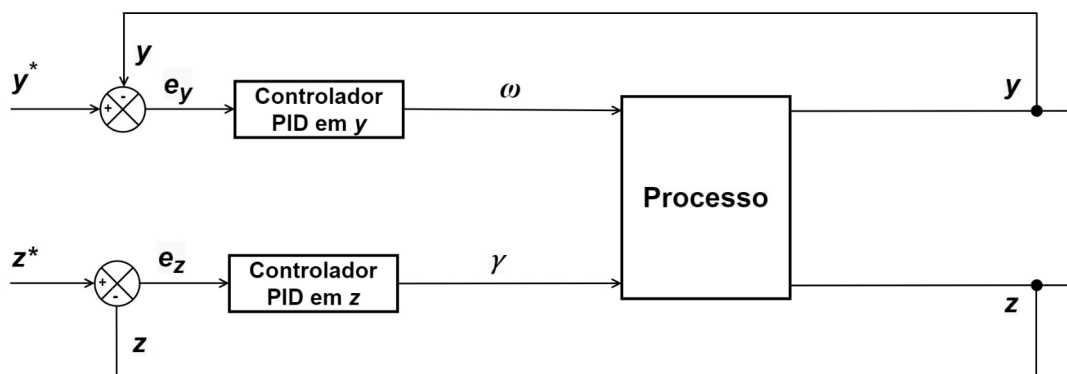
$$\gamma(k) = \gamma(k-1) + K_{pz}[e_z(k) - e_z(k-1)] + K_{iz}T_s e_z(k-1) + \frac{K_{dz}}{T_s}[e_z(k) - 2e_z(k-1) + e_z(k-2)] \quad (36)$$

$$e_z(k) = z^* - z(k) \quad (37)$$

Fazendo um breve resumo, observamos que as Equações (35) e (37) apresentam os erros do sistema para cada variável, onde y^* e z^* são as variáveis de referência do alvo determinado, y e z são as variáveis que terão suas informações atualizadas no mesmo instante de tempo, gerando ações no processo, como demostram as Equações (34) e (36). Um diagrama de blocos representando o controlador é apresentada na Figura 3.17

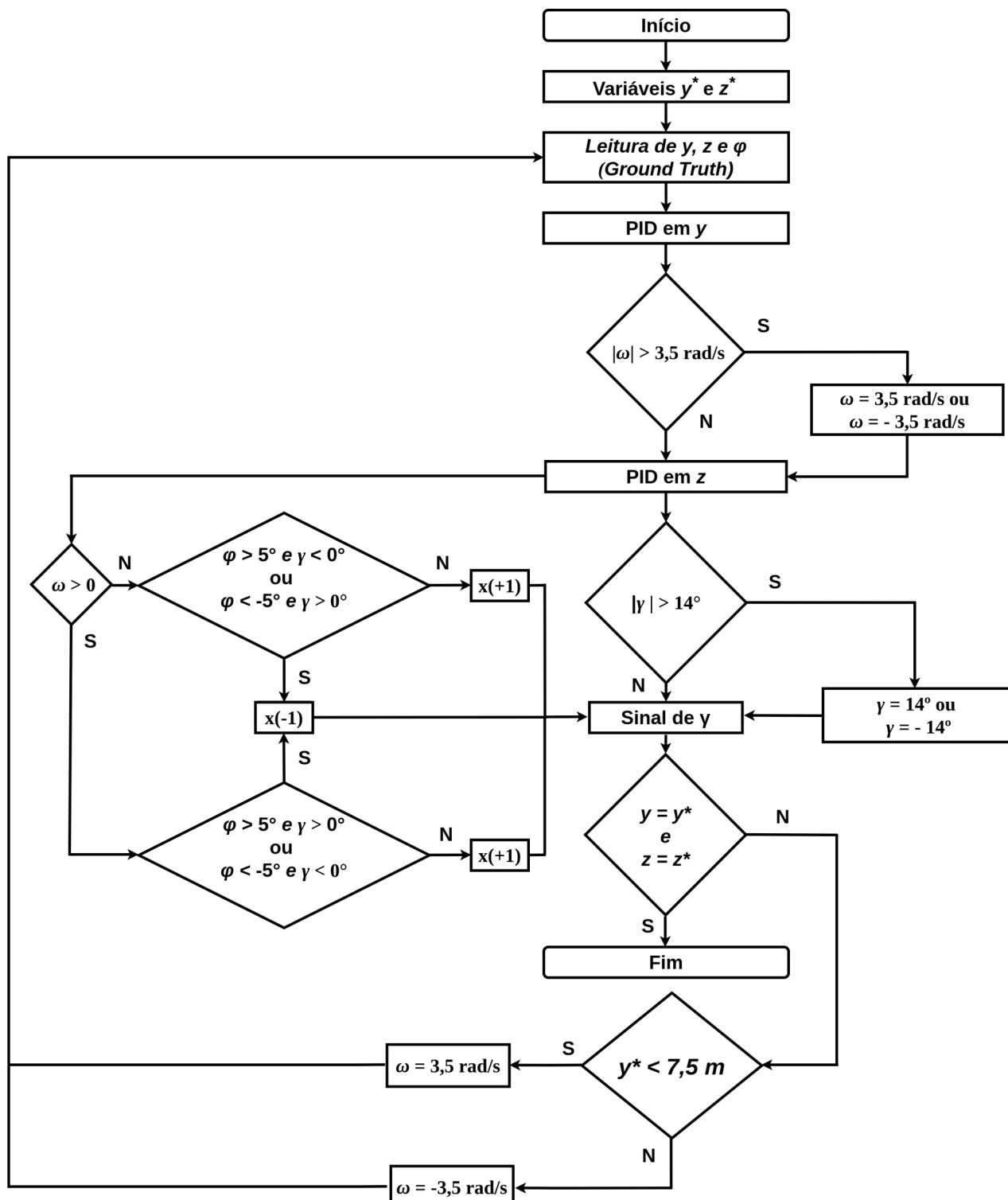
O fluxograma de deslocamento de tarefa para este ambiente é descrito pelo fluxograma na Figura 3.18. Inicia-se pelas leituras das variáveis e dos erros, fornecendo assim um novo valor para ω , para γ , e para y^* . Todos estes valores e critérios são os mesmos do controlador PID para ambiente circular.

Figura 3.17: Diagrama de blocos para o controle no tubo retangular.



Fonte: Autor

Figura 3.18: Fluxograma de deslocamento de tarefa no tubo retangular.



Fonte: Autor

4 RESULTADOS

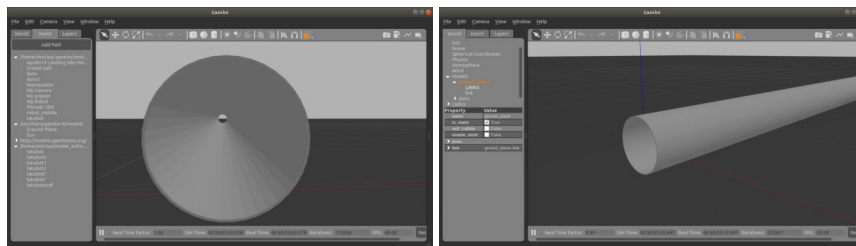
Neste capítulo serão discutidos os resultados, detalhando o simulador com os ambientes de atuação e o robô móvel proposto separados. A seguir, discuti-se o experimento dentro dos cenários de navegação determinados, e por último as análises e estudos da estratégia de controle PID utilizada no trabalho.

4.1 SIMULADOR

4.1.1 AMBIENTES DE LOCOMOÇÃO E PROPOSTA ROBÓTICA

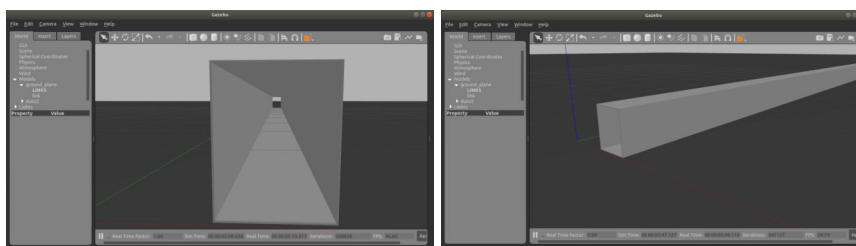
Os ambientes de atuação foram criados com o auxílio do software Gazebo, na versão 9.11.0. Uma visão geral dos dois tubos é ilustrada nas Figuras 4.1 e 4.2, onde se pode visualizar os ambientes com suas seções transversais propostas.

Figura 4.1: Imagem frontal e isométrica do tubo com seção circular.



Fonte: Autor.

Figura 4.2: Imagem frontal e isométrica do tubo com seção retangular.

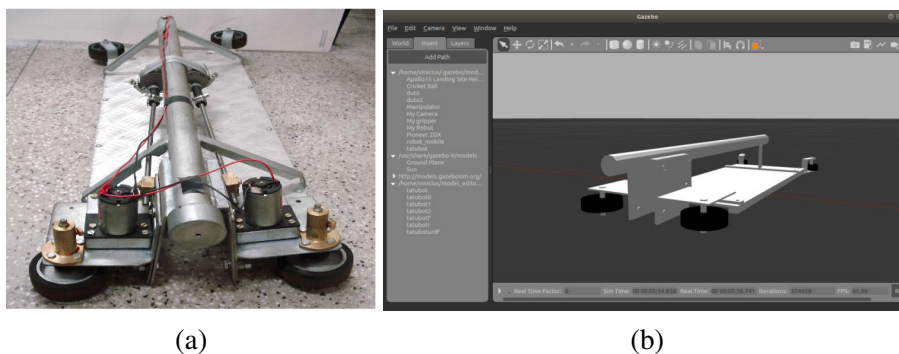


Fonte: Autor.

Os tubos retilíneos possuem um comprimento de 10 m e suas seções transversais são constantes (ao longo dos dutos). Tendo em vista as dimensões operacionais do protótipo da Figura 4.3, foi escolhido um diâmetro de 304 mm, bem como a largura do duto retangular, também de 304 mm, com uma altura determinada de 500 mm.

O sistema robótico foi modelado e baseado na obra de Mello-Jr et al. (2008), ao qual foi mantida a sua mobilidade de navegação em ambiente circulares e adicionada a capacidade de locomoção para cenários com seções transversais retangulares. A Figura 4.3 detalha melhor estes equipamentos.

Figura 4.3: (a) Robô proposto por Mello-Jr et al. (2008) e (b) visão isométrica robô no simulador Gazebo.



Fonte: Adaptado de Mello-Jr et al. (2008, p. 2) e Autor.

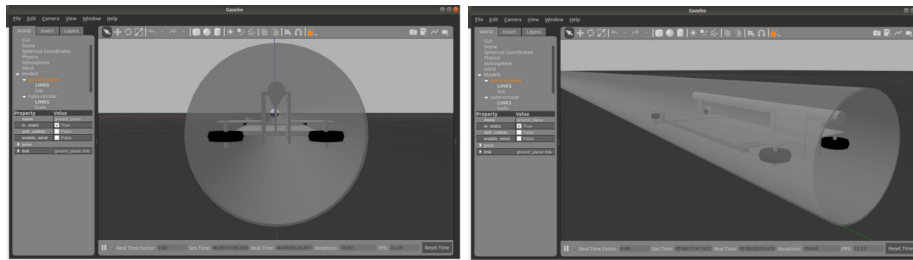
4.1.2 PROJETO SIMULADO

O simulador Gazebo possibilitou a criação e visualização dos ambientes de atuação e do modelo robótico de modo separados. O sistema ROS (versão 1) permitiu criar mecanismos de controle das variáveis controladas (ω - velocidades angulares e γ - ângulos de ataque) e das informações das posições do sistema robótico no simulador. Estes mecanismos de comunicações são chamadas de *tópicos* e foram criados cinco para o trabalho:

- */tatubotni/right_front_wheel_controller/command* - tópico para a velocidade angular da roda frontal direita;
- */tatubotni/left_front_wheel_controller/command* - tópico para a velocidade angular da roda frontal esquerda;
- */tatubotni/right_front_hinge_controller/command* - tópico para o ângulo de ataque da roda frontal direita;
- */tatubotni/left_front_hinge_controller/command* - tópico para o ângulo de ataque da roda frontal esquerda;
- */ground_truth* - tópico para a odometria do robô.

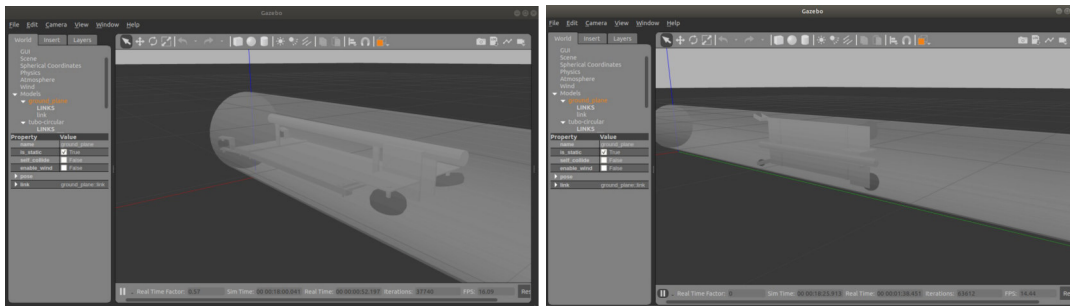
As Figuras 4.4, 4.5, 4.6, e 4.7 mostram a forma simulada do robô nos tubos.

Figura 4.4: Imagens do robô navegando ao longo do duto circular.



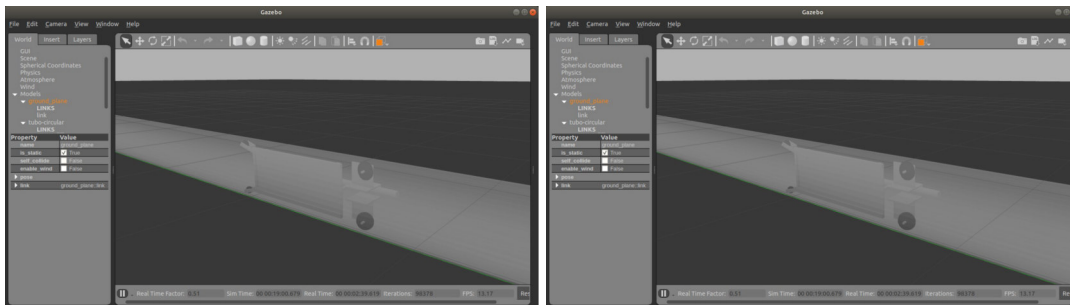
Fonte: Autor.

Figura 4.5: (a) o robô posicionado na origem do ambiente ($y = 0$ m e $\theta = 0^\circ$); (b) o sistema robótico está à 90° em relação a origem; (c) o equipamento está à 180° em relação a origem; (d) o robô está à 270° em relação a posição inicial; e (e) o robô executou uma volta completa ($\theta = 360^\circ$).



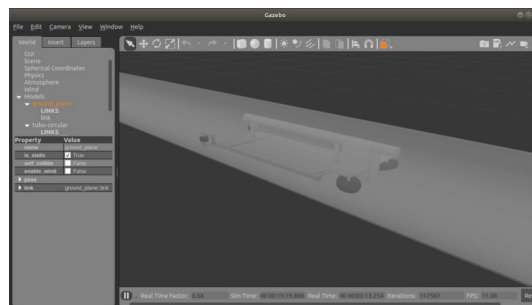
(a)

(b)



(c)

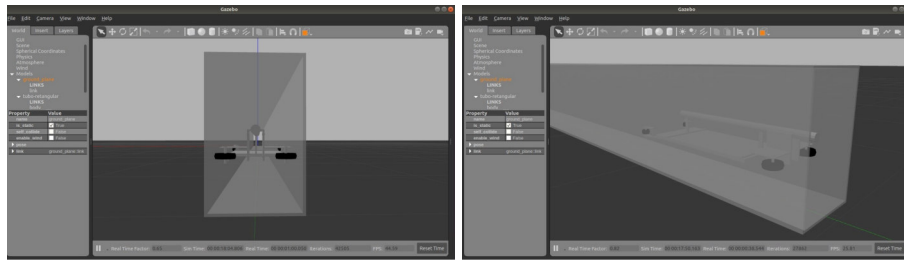
(d)



(e)

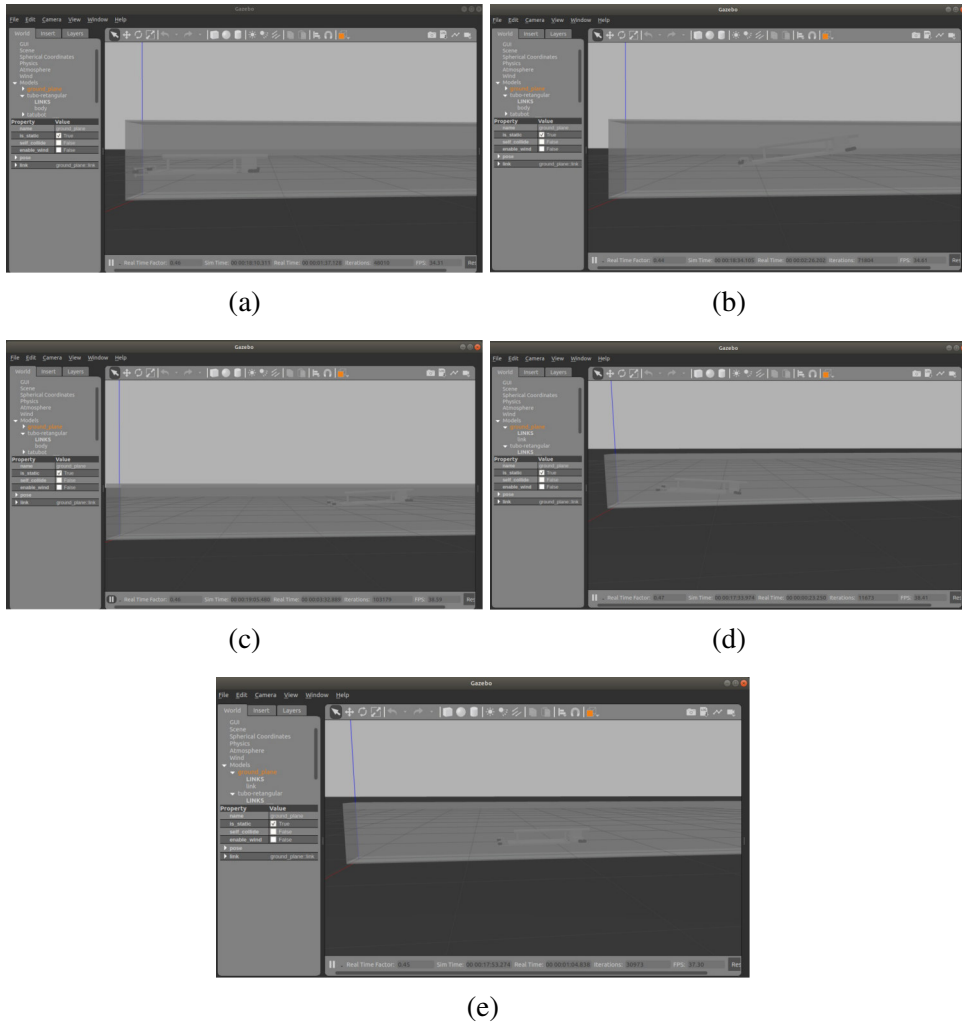
Fonte: Autor.

Figura 4.6: Imagem frontal e isométrica do robô no tubo retangular.



Fonte: Autor.

Figura 4.7: (a) o sistema robótico na posição inicial ($y = 0\text{m}$, $z = 0,18\text{ m}$ e $\phi = 0^\circ$); (b) o robô se desloca para a parte superior do tubo através da variação dos ângulos de ataque das rodas; (c) o equipamento se locomove na parte superior do ambiente de atuação; (d) o dispositivo se move para a parte inferior do tubo retangular; e (e) o robô é apresentado no seu estágio final do ambiente, onde se encontra na parte inferior.



Fonte: Autor.

4.2 RESULTADOS DO CONTROLADOR PID PARA O ROBÔ NO INTERIOR DOS DUTOS CIRCULAR E RETANGULAR

O robô para os tubos circular e retangular foi testado com a estratégia de controle PID. A lei de controle do PID discretizado foi determinado pela Equação 15 com um T_s (tempo

de amostragem) de 0,001s (frequência de amostragem de 1000 Hz). Os testes foram feitos com um critério de erro de mais ou menos 5%.

Para a obtenção dos valores dos coeficientes dos ganhos foi utilizado o método empírico. Inicialmente se fez um processo de tentativa e erro dos seus valores através da planilha gratuita *LibreOffice*, onde se considerou que o deslocamento em y e θ variasse de forma constante. Desse modo, foi possível aumentar, de forma lenta, os ganhos proporcionais em escala 10, e mantendo os demais coeficientes nulos. Após ajustar o valor para o K_p , foram encontrados os demais ganhos.

4.2.1 PARA A SOLUÇÃO ROBÓTICA NO TUBO CIRCULAR

Para todos os casos testados com o robô no ambiente circular. O controlador PI em y utilizou os seguintes ganhos:

- $K_{py} = 1,5$;
- $K_{iy} = 0,375$;
- $K_{dy} = 0$.

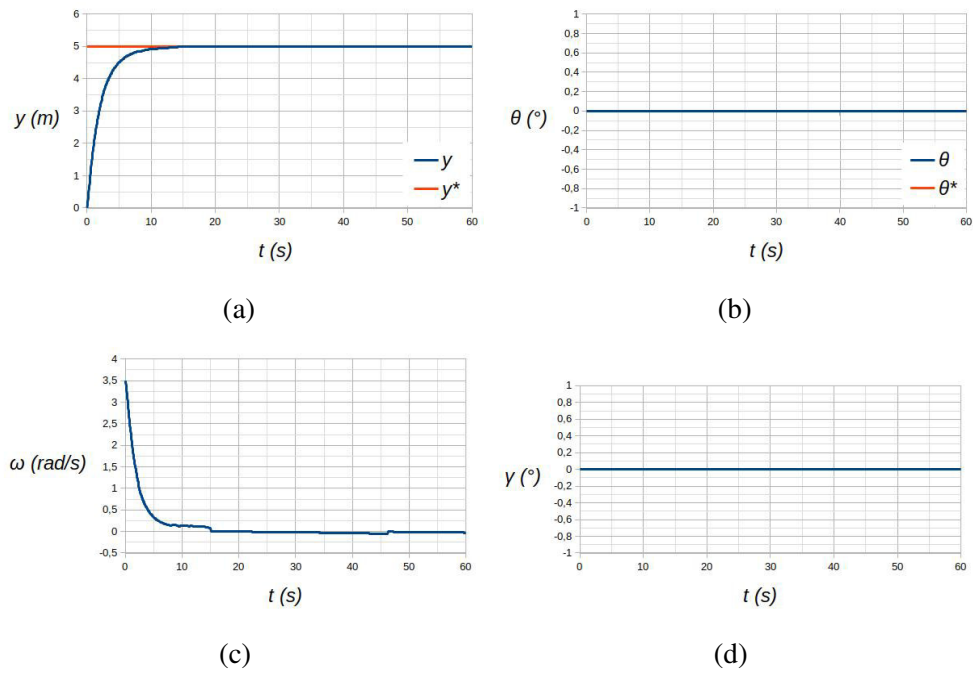
E o controlador PI em θ teve os seguintes valores:

- $K_{p\theta} = 0,775$;
- $K_{i\theta} = 1$;
- $K_{d\theta} = 0$.

No primeiro teste realizado, como ilustra a Figura 4.8, o robô realizou o deslocamento sem giro partindo da sua origem ($y_{inicial} = 0$ m e $\theta_{inicial} = 0^\circ$) até os alvos estipulados ($y^* = 5$ m e $\theta^* = 0^\circ$). A Figura 4.8, mostra que o sistema atingiu a estabilidade na referência de forma a ficar dentro da margem de erro de 5% tanto em y e como em θ . A sua variável controlada ω , decrescente conforme o deslocamento linear em y se aproxima do alvo, quando o sistema robótico chega ao destino, ω atinge o valor zero. A saída do controlador PI em θ (γ) não sofre alterações porque não há deslocamento angular em y (θ). Assim seu valor ao longo do tempo é zero.

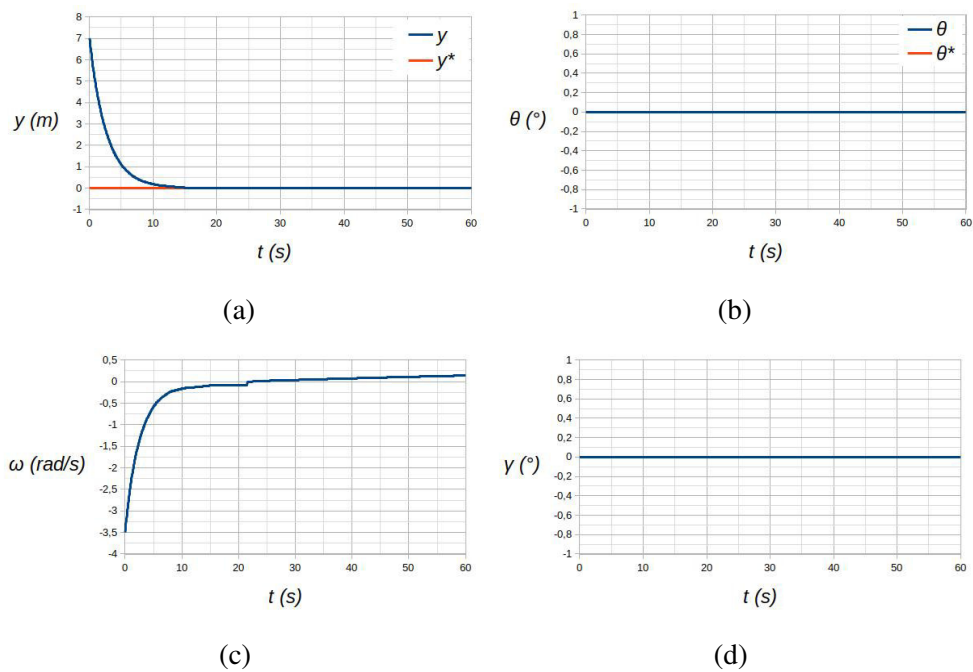
No segundo caso (teste 2), como é mostrado na Figura 4.9, o equipamento robótico executou também deslocamento linear em y sem rotacionar, mas neste teste foi de uma posição de $y_{inicial} = 7$ m e $\theta_{inicial} = 0^\circ$ até a sua origem ($y^* = 0$ m e $\theta^* = 0^\circ$). O gráfico do deslocamento angular em y (da Figura 4.9) mostra que novamente o dispositivo se moveu sem rotacionar proporcionando uma saída com ângulo de ataque igual à zero. Já no caso do controlador em y , o robô alcançou a referência de modo estável e a sua variável de saída (ω) foi lentamente se aproximando do zero. Ambos os controladores atenderam o critério.

Figura 4.8: Teste 1 ($y^* = 5 \text{ m}$ e $\theta^* = 0^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

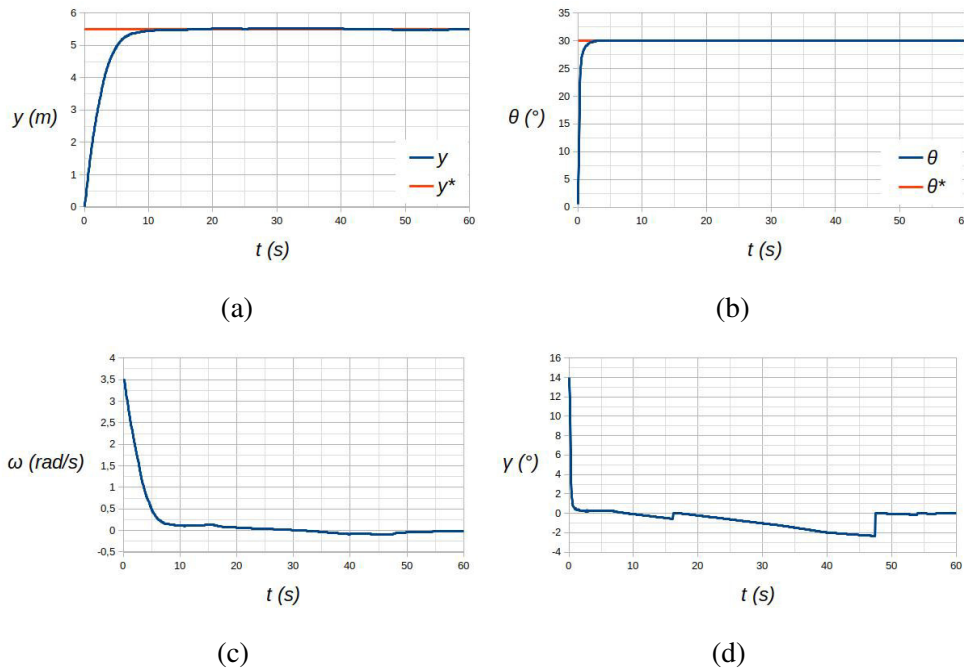
Figura 4.9: Teste 2 ($y^* = 0 \text{ m}$ e $\theta^* = 0^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

Nos testes 3 (Figura 4.10) e 4 (Figura 4.11), o robô partiu da mesma posição inicial ($y_{inicial} = 0$ m e $\theta_{inicial} = 0^\circ$) até o mesmo objetivo em y ($y^* = 5,5$ m), porém os seus deslocamentos angulares em y (θ) são diferentes. Em ambos os casos, o sistema robótico realizou o deslocamento necessário, em y , de forma estável na referência. Todavia no caso 4 ele atingiu a referência em menos tempo em relação ao 3. Em relação a variável ω , a saída do teste 3 (Figura 4.10) apresentou uma pequena variação próxima do valor nulo, pois pode ter ocorrido alguns distúrbios ao longo do trajeto. No caso 4, da Figura 4.11, o dispositivo apresentou um ω estável na saída. Já para o deslocamento angular em $y(\theta)$, em ambos os testes, o robô atingiu a referência de modo a respeitar a margem de erro e antes dos deslocamentos lineares em y . No teste 3, o equipamento girou 30° , e no 4 girou -45° . Porém, o comportamento de ambas as saídas γ , os controladores apresentaram algumas perturbações na variável γ até atingirem o valor zero mas dentro dos 5%.

Figura 4.10: Teste 3 ($y^* = 5,5$ m e $\theta^* = 30^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).

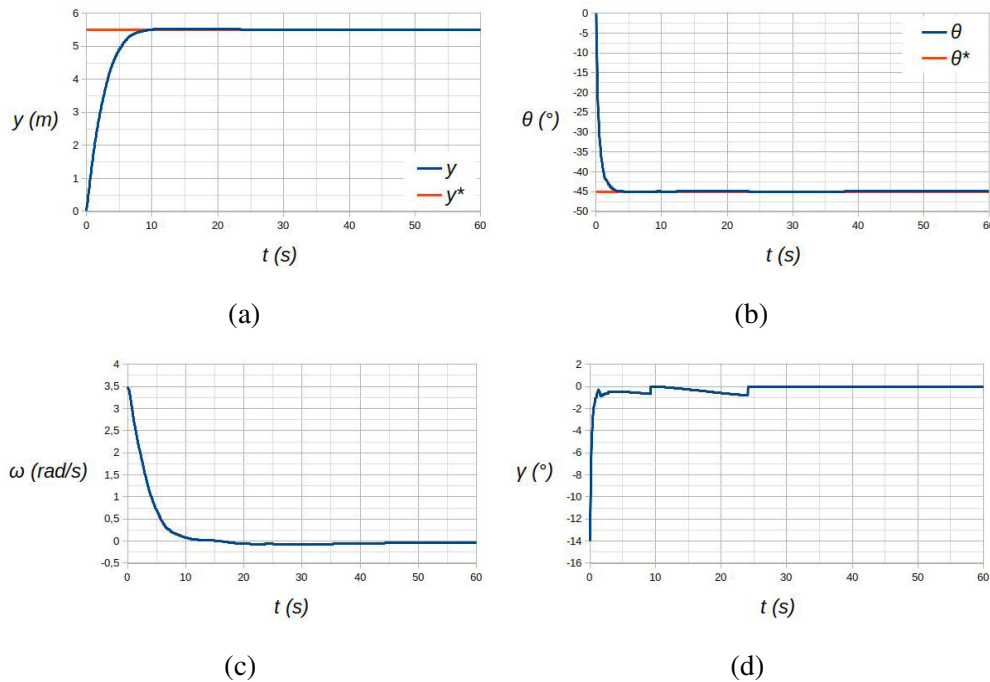


Fonte: Autor.

Para o teste 5 (Figura 4.12) e 6 (Figura 4.13), o dispositivo robótico realizou rotação e mais um deslocamento inferior para realizar o giro, o alvo determinado é $y^* = 0,75$ m partindo da origem ($y_{inicial} = 0$ m e $\theta_{inicial} = 0^\circ$). Na Figura 4.12 mostra o comportamento do robô quando o mesmo alcança $y^* = 0,75$ m, e ao mesmo não ter concluído todo o giro. Para este caso, foi dado um estímulo ($\omega = 3,5$ rad/s) para o equipamento poder avançar no duto e continuar rotacionando. O sistema robótico completou o giro de 90° e ainda voltou ao objetivo em y , podendo ser observado que ao concluir os alvos as suas variáveis controladas (ω e γ) estabilizaram no valor zero após o pulso dado e ficando abaixo do erro

de 5%.

Figura 4.11: Teste 4 ($y^* = 5,5$ m e $\theta^* = -45^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).

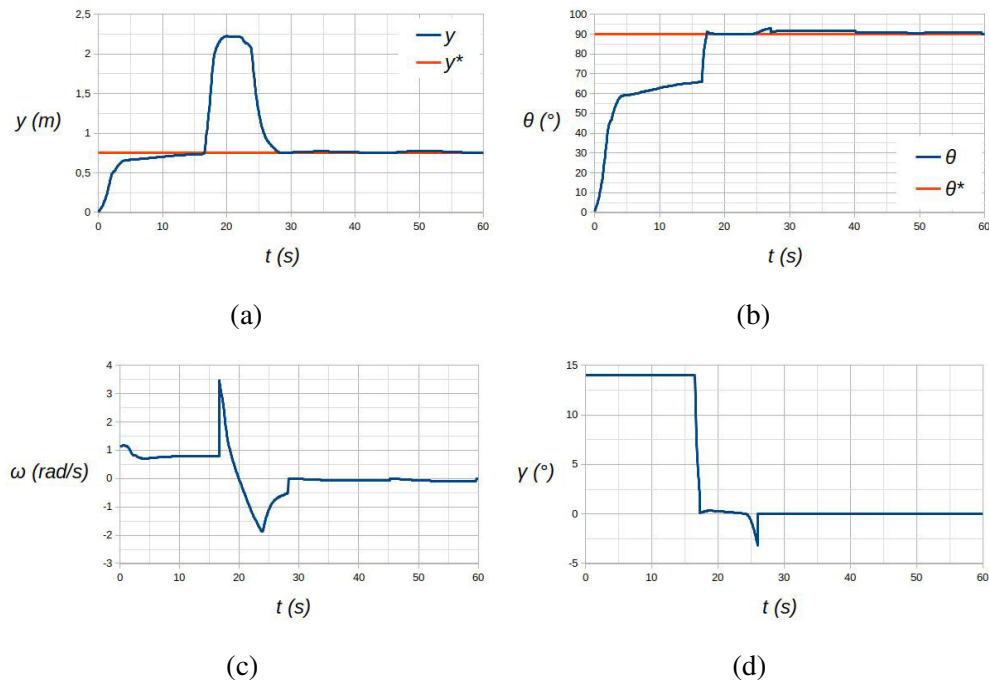


Fonte: Autor.

No caso 6, ilustrado na Figura 4.13, o robô executou o giro de -180° ($\theta^* = -180^\circ$) com um deslocamento linear em y de $0,75$ m ($y^* = 0,75$ m). O equipamento saiu de $y_{inicial} = 0$ m e $\theta_{inicial} = 0^\circ$. Neste caso, o sistema robótico realizou o deslocamento em y antes de completar a rotação (θ), e novamente foi dado um pulso de $\omega = 3,5$ rad/s, permitindo assim que o robô continuasse o giro e atingisse o alvo em θ de -180° , e assim recuou até o alvo novamente em $y = 0,75$ m. Já no controlador PI em θ , este apresentou uma entrada e uma saída ficou dentro do erro de 5% como mostrar a Figura 4.13. Já no caso do controlador em y , o dispositivo atingiu o alvo e se estabilizou, porém ocorreram pequenos ruídos na sua saída, todos dentro da margem de erro.

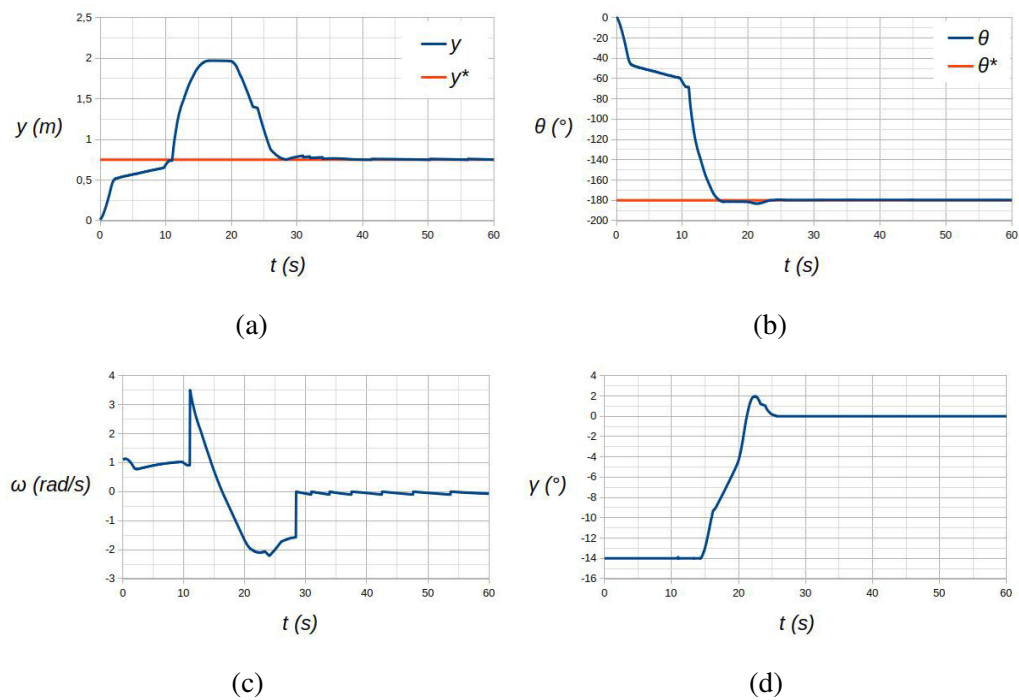
O dispositivo robótico, no teste 7, realizou o retorno para a origem do ambiente circular ($y^* = 0$ m e $\theta^* = 0^\circ$) a partir de $y_{inicial} = 5,5$ m e $\theta_{inicial} = 30^\circ$. Neste teste, o robô alcançou o objetivo desejado, no deslocamento em y , mesma a sua saída (ω) ter apresentado um leve distúrbio. Já no caso do deslocamento angular em y (θ), a entrada apresentou um *overshoot*, mas logo em seguida se estabilizou na referência. Na Figura 4.14, pode ser observado que a saída γ (do controlador PI em θ) teve um breve momento em que o ângulo de ataque atingiu um valor positivo, e logo em seguida assumiu o valor zero. Neste instante em que a saída atingiu este valor positivo se deve ao *overshoot* da sua entrada. Ambos os controladores ficaram dentro do critério utilizado.

Figura 4.12: Teste 5 ($y^* = 0,75$ m e $\theta^* = 90^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



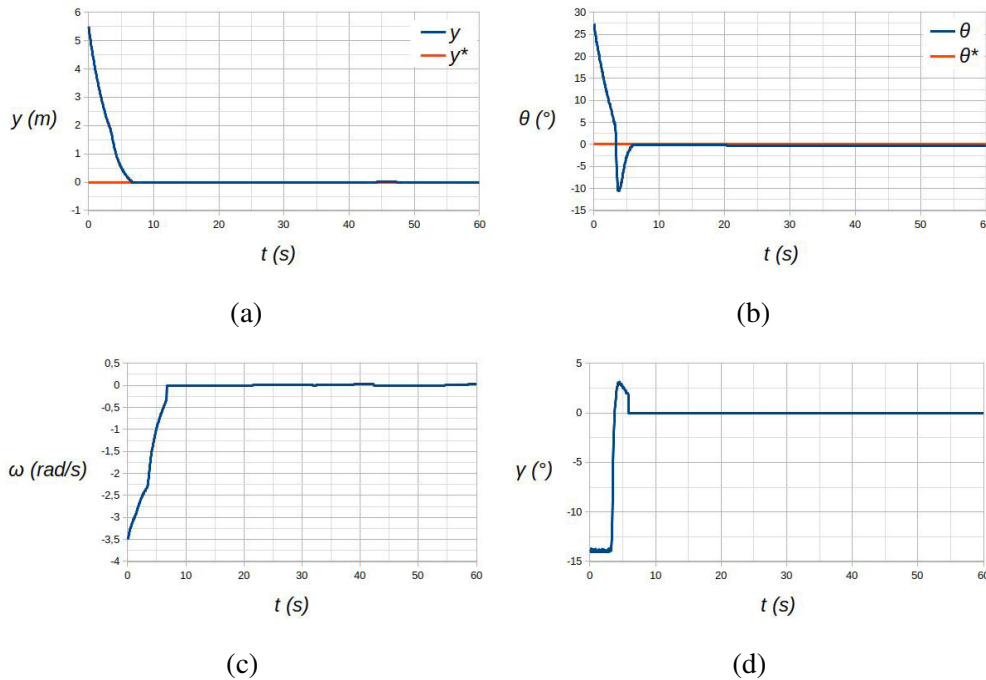
Fonte: Autor.

Figura 4.13: Teste 6 ($y^* = 0,75$ m e $\theta^* = -180^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

Figura 4.14: Teste 7 ($y^* = 0$ m e $\theta^* = 0^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).

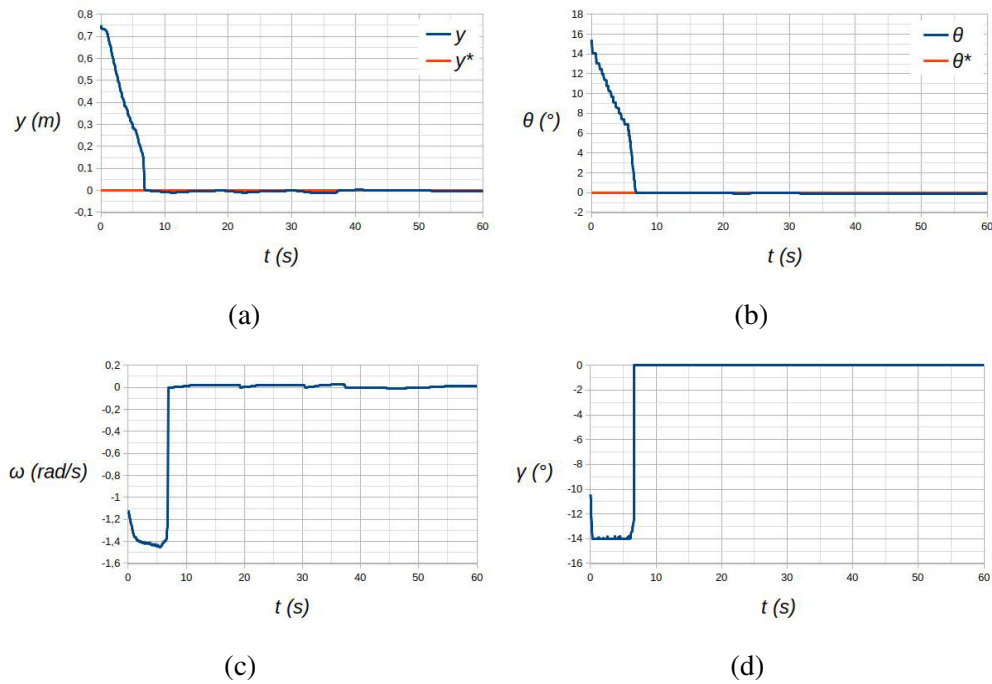


Fonte: Autor.

No caso 8, o robô também executou o processo de retorno para $y^* = 0$ m e $\theta^* = 0^\circ$ (origem do ambiente) só que a partir de $y_{inicial} = 0,75$ m e $\theta_{inicial} = 15^\circ$. Neste alvo, tanto a distância em y como a rotação em θ são menores que as do teste 7. A Figura 4.15 mostra as entradas e as saídas do teste 8. No controlador PI em y , pode ser analisado que a entrada (y) alcançou a referência, mas existiram pequenas oscilações no seu objetivo ocasionados pelo leve ruído também ocorrido na saída (ω). Já para controlador em θ , a entrada apresentou uma demora nos primeiros instantes ($t = 5$ s) e depois se estabilizou rápido, tendo um resultado dentro na margem de erro tanto na sua entrada θ quanto na sua saída γ . Mesmo que a saída tendo saturado no valor mínimo de γ que é de -14° .

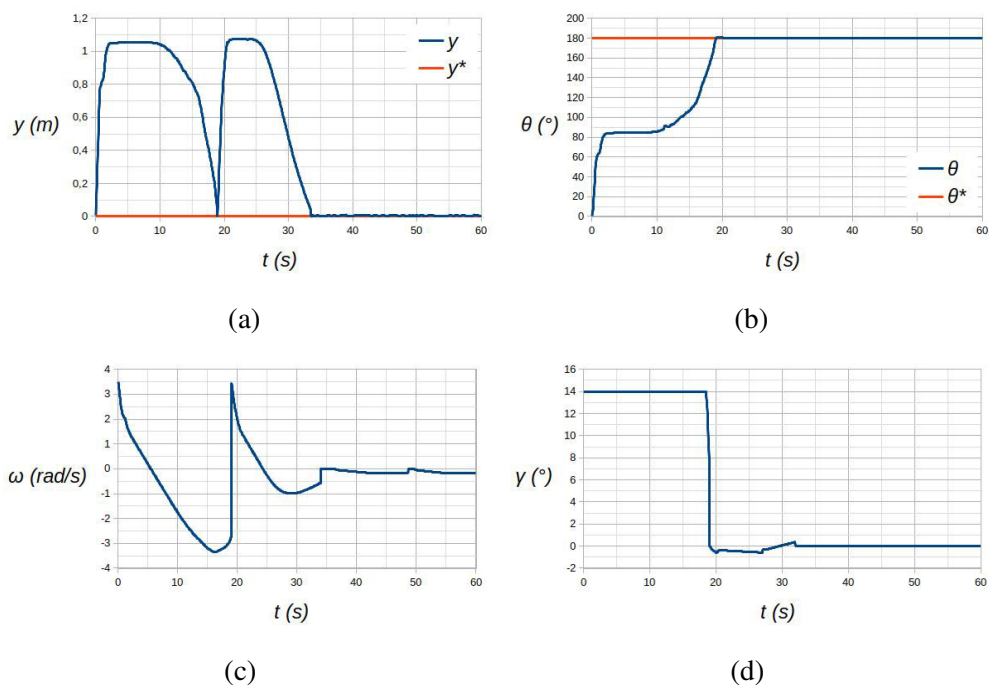
No teste 9, mostrado na Figura 4.16, o robô partiu da origem ($y_{inicial} = 0$ m e $\theta_{inicial} = 0^\circ$) e foi até os alvos $y^* = 0$ m e $\theta^* = 180^\circ$. Neste teste específico, o dispositivo precisou receber dois estímulos ($\omega = 3,5$ rad/s) para atingir o objetivo em θ , e só após isso concluir $y^* = 0$ m. A entrada alcançou a referência porém sofreu um pequeno distúrbio após receber dois pulsos. O mesmo vale para a variável controlada ω mas nesse caso o ruído na saída foi maior. Já no caso do deslocamento angular em y (θ), a entrada apresentou uma subida rápida e depois subiu lentamente até se estabilizar na referência. A saída do controlador em θ ficou na zona de saturação (atingindo o valor máximo do ângulo de ataque de 14°), e no instante $t = 18$ s caiu bruscamente para um valor levemente negativo até se estabilizar no valor zero. Todos dentro da margem de erro.

Figura 4.15: Teste 8 ($y^* = 0$ m e $\theta^* = 0^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

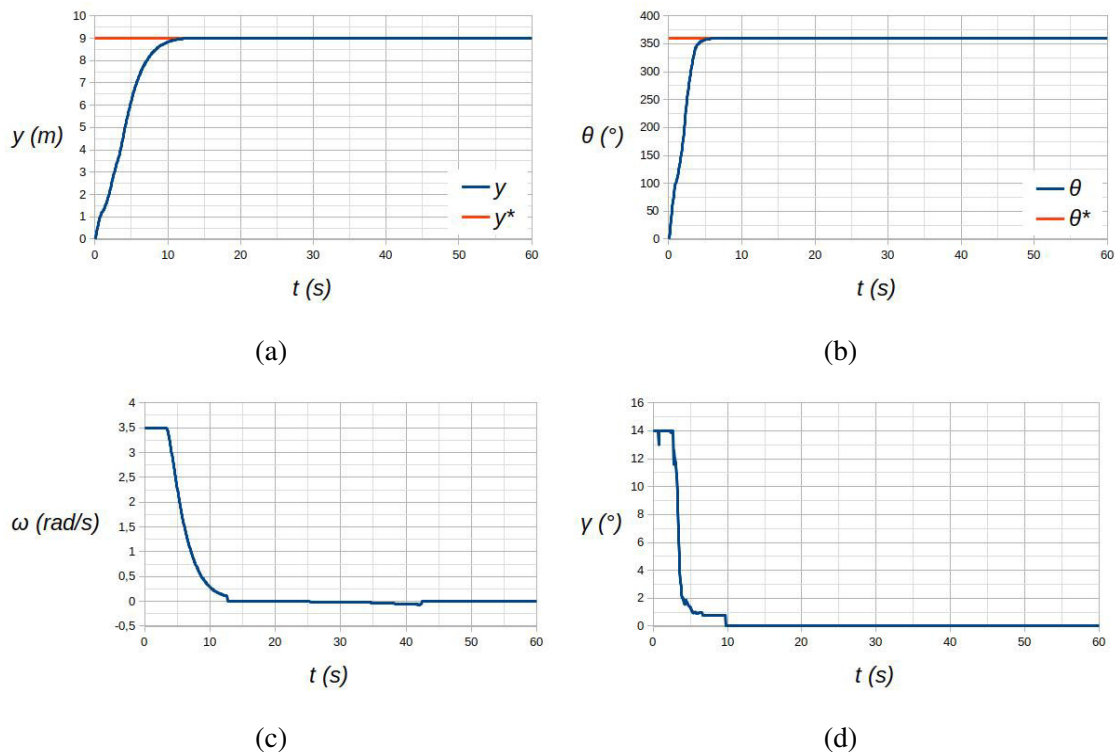
Figura 4.16: Teste 9 ($y^* = 0$ m e $\theta^* = 180^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

Para o teste 10 foi dado como objetivo realizar uma rotação completa ($\theta^* = 360^\circ$) e ainda chegar no limite do duto circular ($y^* = 9$ m). Neste caso, a estratégia de controle PI em θ atingiu a referência no instante de $t = 5$ s e sua saída γ ficou igual à zero no instante $t = 10$ s. Este atraso se deve ao pequeno ruído ocasionado durante a navegação. Já no controlador PI em y , a entrada atingiu o alvo um pouco depois da entrada do controlador PI em θ . E a sua saída ω apresentou um certo atraso para ficar igual a zero, e também teve algumas leves perturbações em alguns momentos. A Figura 4.17 mostra o comportamento destas variáveis com um erro inferior à 5%.

Figura 4.17: Teste 10 ($y^* = 9$ m e $\theta^* = 360^\circ$) - (a) Deslocamento Linear em y ; (b) Deslocamento Angular em y (θ); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

4.2.2 PARA A SOLUÇÃO ROBÓTICA NO TUBO RETANGULAR

Para todos os experimentos simulados com o robô no duto retangular, os controladores PI e P em y e em z , respectivamente, apresentaram os seguintes coeficientes para os seus ganhos, no caso em y :

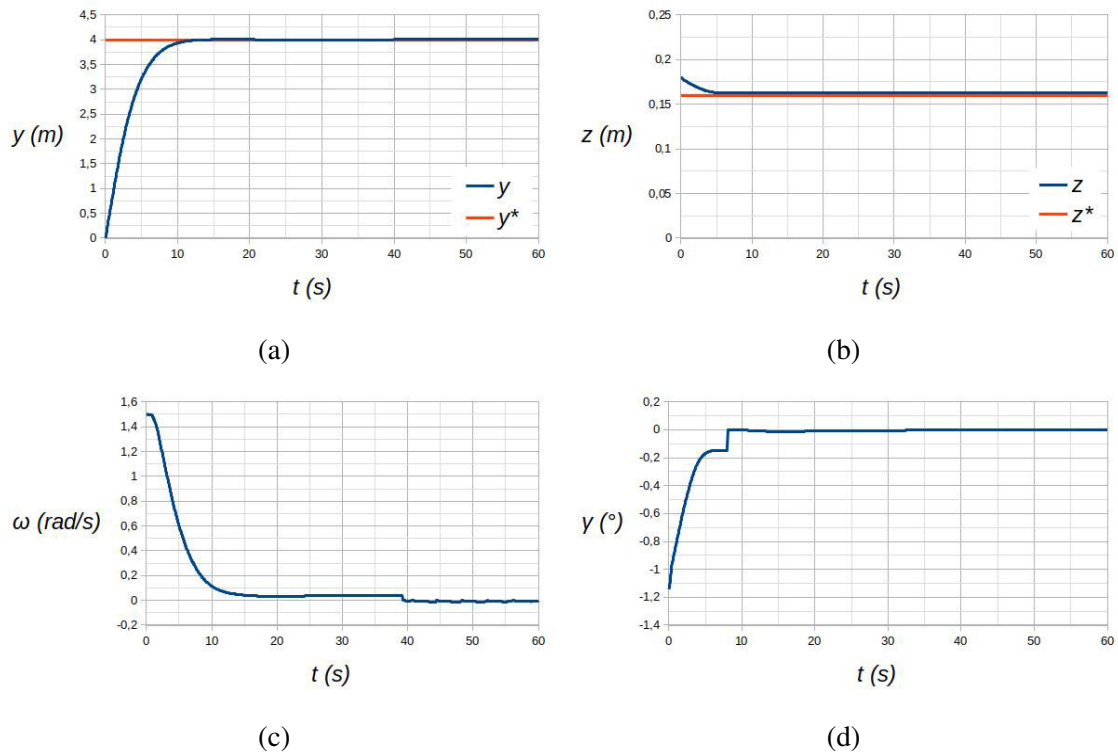
- $K_{py} = 1,5$;
- $K_{iy} = 0,375$;
- $K_{dy} = 0,0$.

Para a estratégia de controle em z :

- $K_{pz} = 1,0$;
- $K_{iz} = 0,0$;
- $K_{dz} = 0,0$.

No teste 1 (como é mostrado na Figura 4.18) o dispositivo robótico realizou o deslocamento linear em y e z a partir da origem ($y_{inicial} = 0$ m e $z_{inicial} = 0,18$ m) até os objetivos determinados ($y^* = 4$ m e $z^* = 0,16$ m). Neste caso, o robô se dirigiu para a parte inferior do tubo retangular. A Figura 4.18, ilustra que o sistema atingiu a estabilidade na referência de forma estável em y , porém na entrada de z ficou levemente em cima da referência cometendo um erro inferior a 5%. Isso se deve ao pequeno ruído na saída do controlador PI em y . A variável de saída ω sofreu um distúrbio provocado pelo ambiente. Já a saída no controlador em z teve sua estabilidade em zero mesmo tendo uma subida lenta. Em ambos os cenários os resultados estão dentro do critério.

Figura 4.18: Teste 1 ($y^* = 4$ m e $z^* = 0,16$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).

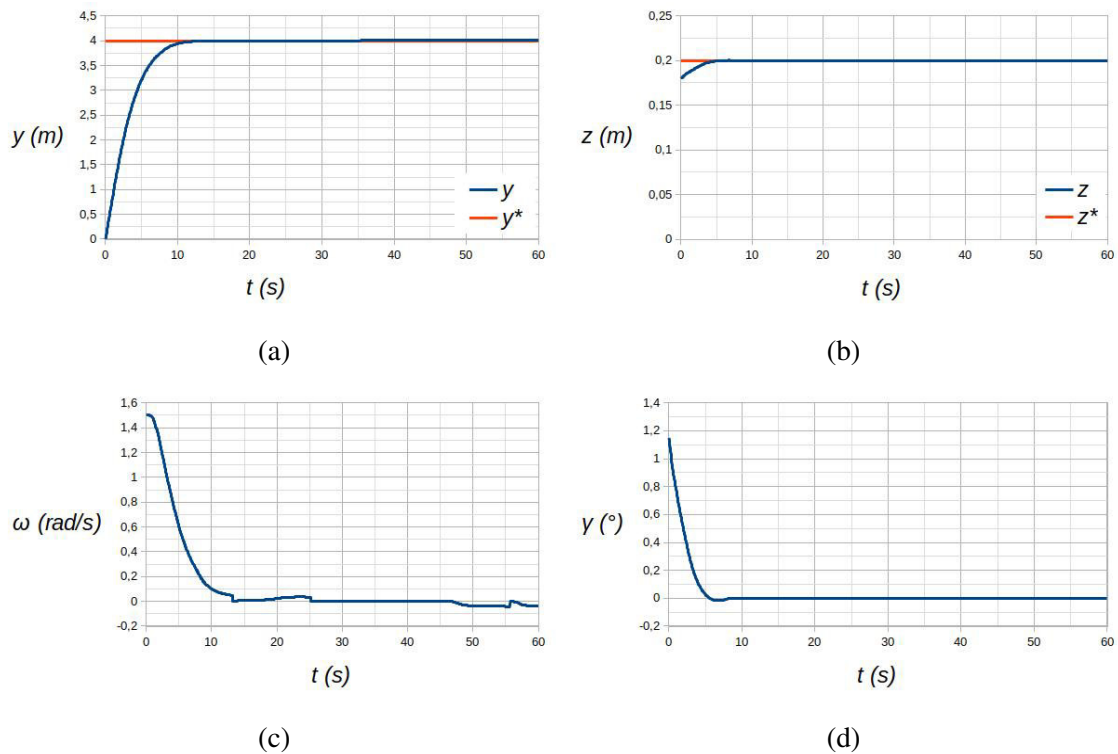


Fonte: Autor.

No segundo teste, como é visualizado na Figura 4.19, o robô teve o mesmo alvo em y de $y^* = 4$ m, mas desta vez se locomoveu para a parte superior do duto retangular (z^*

= 0,20 m). A posição inicial é a origem ($y_{inicial} = 0$ m e $z_{inicial} = 0,18$ m). Os gráficos dos deslocamentos lineares tanto em y como em z atingiram a estabilidade sem grandes problemas apresentando resultados dentro da margem de erro de 5%. A saída da estratégia de controle P em z obteve uma estabilidade. Todavia, o controlador em y sofreu com as perturbações do ambiente e oscilou um pouco em cima do zero. O gráfico da velocidade angular das rodas frontais (ω), da Figura 4.19, ilustra estas oscilações provocadas pelo ambiente.

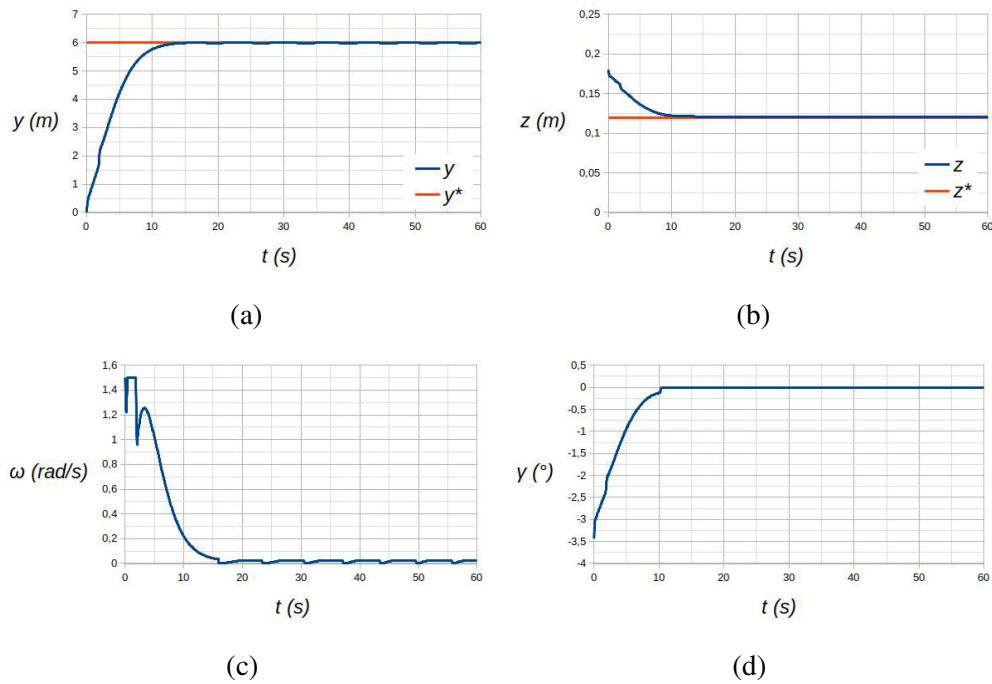
Figura 4.19: Teste 2 ($y^* = 4$ m e $z^* = 0,20$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

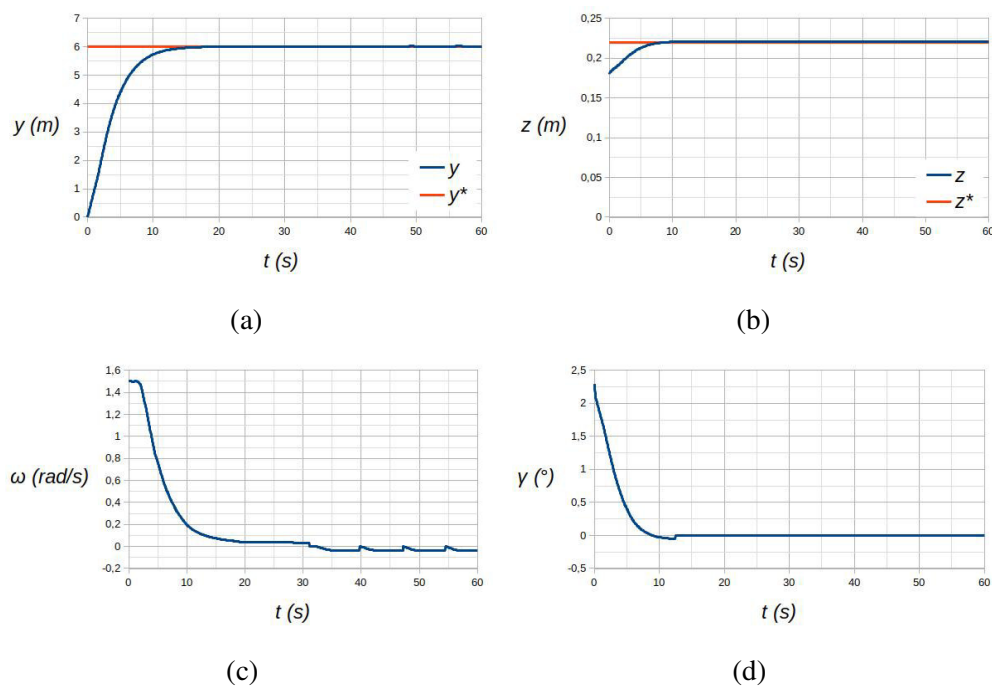
Nos casos 3 e 4, das Figuras 4.20 e 4.21, respectivamente, o robô saiu da mesma posição inicial, como nos testes anteriores ($y_{inicial} = 0$ m e $z_{inicial} = 0,18$ m). Porém, os dois possuem um alvo de $y^* = 6$ m. A diferença é que enquanto o teste 3 se desloca para a parte inferior, o teste 4 se locomove para parte superior do ambiente retangular. Em ambos os casos, tanto o caso 3 como 4 atingiram a referência em suas entradas (y e z) de modo estável e com erro inferior à 5%. Já as saídas dos controladores em z (teste 3 e 4), as variáveis γ , alcançaram o zero sem apresentar problemas ou perturbações. Já nas saídas (ω) das estratégias de controle PI em y ocorreram ruídos e isso fez ambos os testes oscilarem no zero. As Figuras 4.20 e 4.21 mostram estes resultados dos controladores PI e P dos experimentos 3 e 4.

Figura 4.20: Teste 3 ($y^* = 6$ m e $z^* = 0,12$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

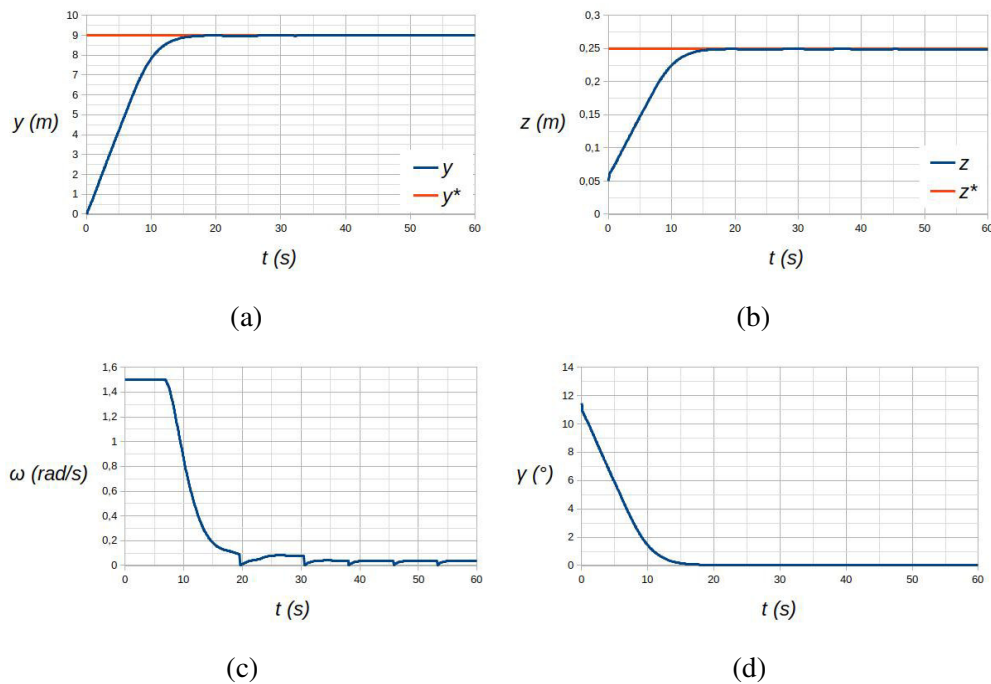
Figura 4.21: Teste 4 ($y^* = 6$ m e $z^* = 0,22$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

Para os testes 5 (Figura 4.22) e 6 (Figura 4.23), o sistema robótico realizou o mesmo deslocamento em y ($y^* = 9$ m). Mas, em ambos os casos, as posições iniciais e objetivos em z foram diferentes. No teste 5, o robô iniciou da posição $y_{inicial} = 0$ m e $z_{inicial} = 0,05$ m, e saiu da parte inferior do ambiente até a parte superior em $z^* = 0,25$ m. Neste caso, o controlador PI em y apresentou uma entrada com uma leve oscilação em torno da referência, proporcionando um distúrbio no valor zero, da saída ω . Já na estratégia de controle P em z , o mesmo apresentou erro inferior de 5% na entrada e na saída.

Figura 4.22: Teste 5 ($y^* = 9$ m e $z^* = 0,25$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z ; (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



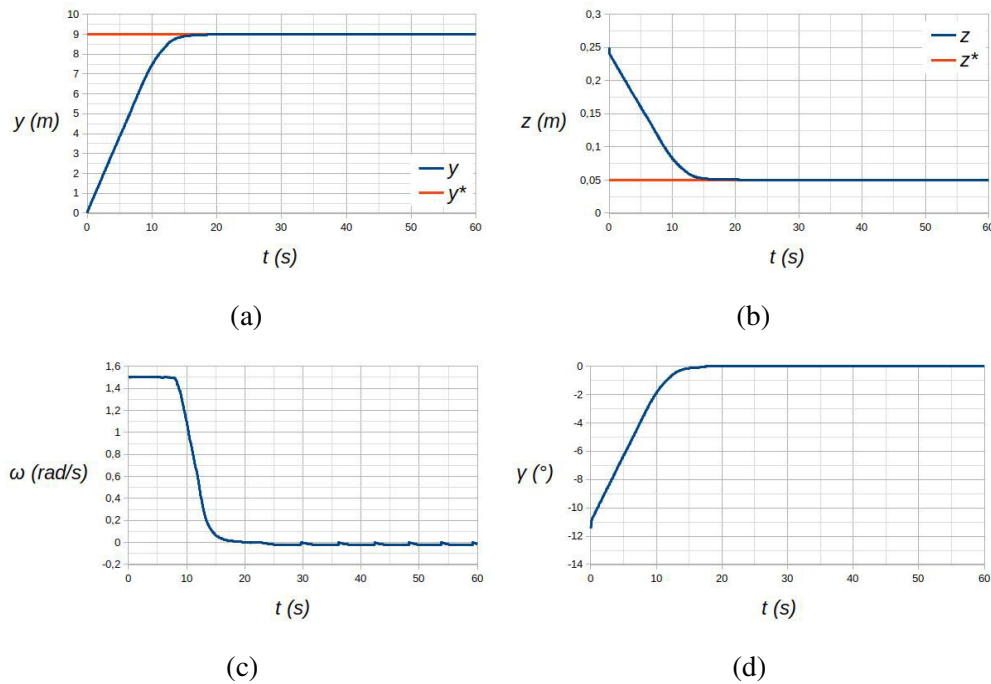
Fonte: Autor.

No experimento 6, o equipamento robótico iniciou da posição $y_{inicial} = 0$ m e $z_{inicial} = 0,25$ m e foi até a posição final $y^* = 9$ m e $z^* = 0,05$ m. Nesta simulação, o robô se deslocou para a parte inferior do duto. Como no teste 5, o controlador P em z se estabilizou na referência na entrada e na saída γ atingindo o valor zero. No controlador PI em y , a entrada obteve estabilidade, porém a sua saída ω teve um pequeno ruído. Em ambos controladores ficaram dentro da margem de erro de 5%.

No experimento 7, visualizado na Figura 4.24, mostra que o robô se locomoveu de uma posição inicial ($y_{inicial} = 4$ m e $z_{inicial} = 0,14$ m) até o alvo ($y^* = 0$ m e $z^* = 0,24$ m). Neste teste, o dispositivo robótico realizou movimento de ré, no eixo y , enquanto sobe para a parte superior do tubo retangular. Na estratégia de controle PI em y , o equipamento oscilou um pouco em torno de $y^* = 0$ m, provocados pelos ruídos na entrada proporcionando por pequenos pulsos na saída ω do controlador, com erro inferior à 5%.

No caso do controlador P em z , o resultado na entrada (z) e na saída (γ) foram dentro da margem de erro nas referências.

Figura 4.23: Teste 6 ($y^* = 9$ m e $z^* = 0,05$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z ; (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).

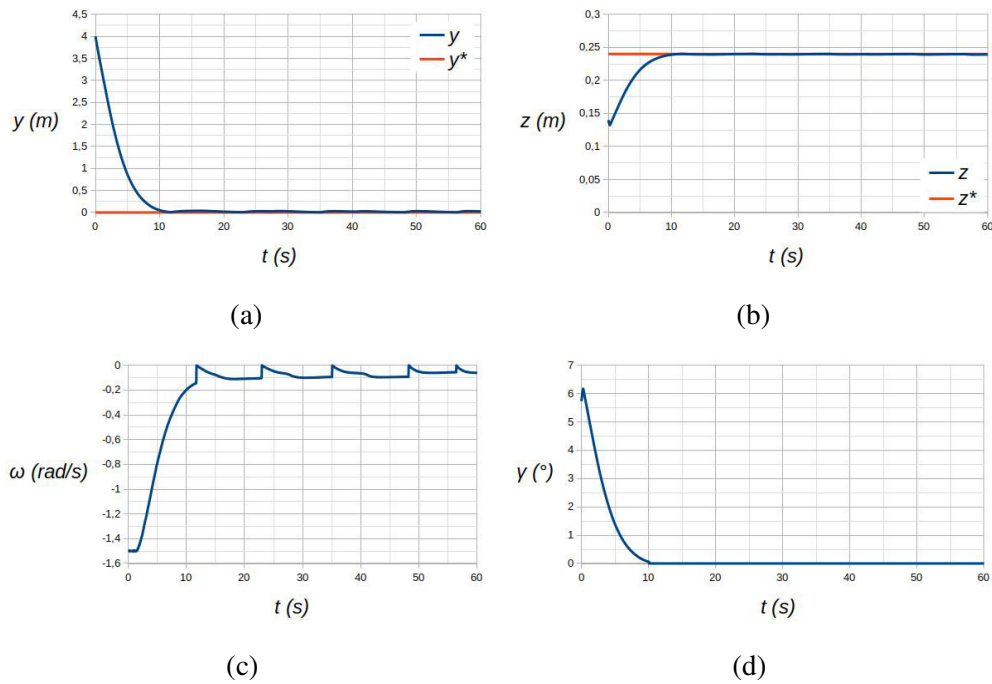


Fonte: Autor.

No teste 8, ilustrado na Figura 4.25, o sistema robótico também executou movimento de ré mas desta vez saiu da parte superior para a inferior do duto. O robô começou em $y_{inicial} = 9$ m e $z_{inicial} = 0,24$ m, e se locomoveu até o destino $y^* = 2$ m e $z^* = 0,14$ m. A Figura 4.25, mostra que o controlador PI em y teve uma entrada estável na referência porém continuou, como no teste 7, com perturbações na saída ω , ocasionadas pelo ambiente. Já controlador em z obteve resultados na entrada e saída dentro do critério de 5%, porque z e γ se estabilizaram sem provocarem distúrbios.

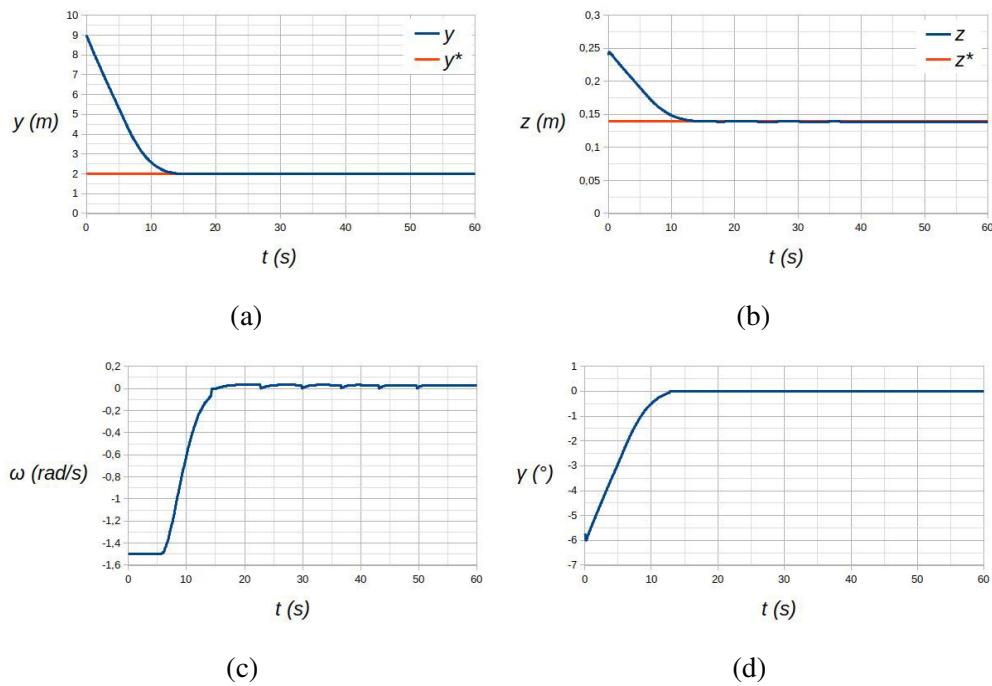
No teste 9, mostrado na Figura 4.26, o robô possui como alvos: $y^* = 1$ m e $z^* = 0,25$ m. Neste caso, se encurtou a distância em y com a finalidade do dispositivo sair de $y_{inicial} = 0$ m e $z_{inicial} = 0,14$ m, e chegar nos objetivos. Conforme pode ser visto na Figura 4.26, o robô atingiu primeiro o alvo em y sem ter concluído o $z^* = 0,25$ m, assim foi necessário realizar um pulso de $\omega = 3,5$ rad/s para corrigir altura do equipamento no tubo. O gráfico (do deslocamento linear em y) mostra que o controlador PI se estabilizou na referência após o pulso, e que a saída ω atingiu o zero. Já no controlador P em z , a entrada z atingiu a referência com algumas oscilações. Já na variável γ ocorreu dois degraus em torno do valor zero apresentando pelos ruídos. Estes erros estão dentro da faixa determinada que é de 5%.

Figura 4.24: Teste 7 ($y^* = 0$ m e $z^* = 0,24$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z ; (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



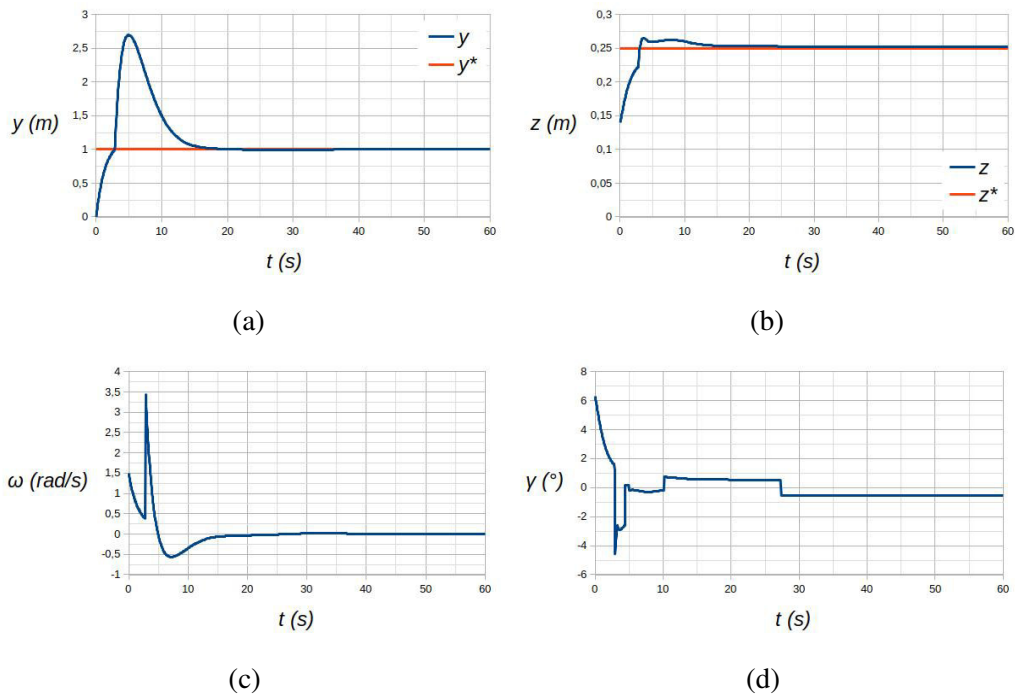
Fonte: Autor.

Figura 4.25: Teste 8 ($y^* = 2$ m e $z^* = 0,14$ m) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z ; (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

Figura 4.26: Teste 9 ($y^* = 1 \text{ m}$ e $z^* = 0,25 \text{ m}$) - (a) Deslocamento Linear em y ; (b) Deslocamento Linear em z (z); (c) Velocidade Angular das Rodas Frontais (ω); e (d) Ângulo de Ataque das Rodas Frontais (γ).



Fonte: Autor.

5 CONCLUSÕES

O ramo da robótica de navegação e movimentação vem tendo poucas soluções disponíveis especificamente para um mesmo robô navegar nos tubos circulares e retangulares. Inovações tecnológicas que permitam a locomoção em ambos os tipos de dutos, são altamente desejadas. No caso de indústrias que possuem estes dois ambientes, estes equipamentos robóticos possibilitam melhorias nas etapas de inspeção e manutenção, ocasionando menores tempos de paradas e redução de custos.

Desta forma, foram utilizados um conjunto de objetivos para realizar uma modelagem de um sistema robótico para navegar em dois cenários distintos. Estes objetivos específicos foram: estudar os tubos de atuação circular e retangular; modelar o projeto TATUBOT para estes dois ambientes; construir os cenários e o robô de forma simulada; desenvolver modelos cinemáticos para o sistema robótico em cada um dos ambientes para analisar o entendimento das variáveis de saída (ω e γ) em relação as variáveis de entrada (y , z e θ); e ainda implementar os controladores PI e P para controlar o robô em cada um dos dutos. Ao longo desta dissertação foi dedicado capítulos para a explicação de cada uma das fases.

O comportamento do dispositivo robótico navegando nos tubos circulares e retangulares foi verificado a partir de diversas simulações. Observou-se que variações de posição linear e angular do robô, utilizando das estratégias de controle PI e P adotadas para cada cenário, foram dentro do critério de erro de 5% estabelecido. As simulações com o uso dos controladores apresentaram erros inferiores à 5% e ficando de acordo com o modelo real.

Entretanto, na realização das simulações foram encontrados algumas limitações no ambiente retangular, para casos que os deslocamentos lineares em y foram pequenos, o robô não conseguiu concluir os deslocamentos em z grandes. Isso aconteceu pois, o ângulo ϕ de projeto nunca ultrapassou o intervalo de -5° à 5° , e o comprimento l (distância entre o ponto P e P') é grande. E como consequência dificultou a mobilidade do equipamento no ambiente retangular.

Os resultados da pesquisa referente a navegação do robô nos cenários circular e retangular que foram analisados através da simulação computacional possibilitam desenvolver

trabalhos futuros. O protótipo TATUBOT poderá ser testado de forma experimental com a finalidade do robô navegar no ambiente de atuação retangular e manter sua mobilidade em tubos circulares.

REFERÊNCIAS

- ADAMS, W. et al. Water pipe robot utilizing soft inflatable actuators. In: IEEE. *2018 IEEE International Conference on Soft Robotics (RoboSoft)*. [S.l.], 2018. p. 321–326.
- AFFONSO, L. O. A. *Equipamentos Mecânicos. Análise de Falhas e Solução de Problemas*. 3. ed. [S.l.]: São Paulo: Qualitymark, 2012.
- ANDRADE, M. R. de. *SIMULADOR PARA O ROV LUMA UTILIZANDO GAZEBO*. 2017. <http://monografias.poli.ufrj.br/monografias/monopoli10020423.pdf>. Acesso em: 11 de novembro de 2019.
- BROWN, L. et al. Elbow detection in pipes for autonomous navigation of inspection robots. *Journal of Intelligent & Robotic Systems*, Springer, p. 1–15, 2018.
- CHABLAT, D.; VENKATESWARAN, S.; BOYER, F. Dynamic model of a bio-inspired robot for piping inspection. In: *ROMANSY 22–Robot Design, Dynamics and Control*. [S.l.]: Springer, 2019. p. 42–51.
- CHUA, C. K.; LEONG, K. F.; LIM, C. S. *Rapid Prototyping: Principles and Applications*. 2. ed. [S.l.: s.n.], 2003. ISBN 9812381171,9789812381170,9789812388834.
- CRAIG, J. J. *Introduction to Robotics: Mechanics and Control*. 2. ed. [S.l.]: Prentice Hall, 1989. ISBN 0-201-09528-9,9780201095289.
- EZQUERRO, A. *ROS Q&A — How to know the Pose of a robot (Python) ?* 2017. Disponível em: <https://www.theconstructsim.com/ros-qa-know-pose-robot-python/>. Acesso em: 01 de novembro 2022.
- FAIRCHILD, C.; HARMAN, T. L. *ROS Robotics By Example*. [S.l.]: Packt Publishing Ltd, 2016. ISBN 978-1782175193.
- FJERDINGEN, S. A.; LILJEBÄCK, P.; TRANSETH, A. A. A snake-like robot for internal inspection of complex pipe structures (piko). In: IEEE. *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. [S.l.], 2009. p. 5665–5671.
- GAZEBO. *Inertial parameters of triangle meshes*. 2014. Disponível em: https://classic.gazebosim.org/tutorials?cat=build_robot&tut=inertia. Acesso em: 30 de novembro 2022.
- GAZEBO. *Tutorial: Make a Mobile Robot - Gazebo*. 2014. Disponível em: http://gazebosim.org/tutorials/?tut=build_robot. Acesso em: 29 de novembro 2021.

GRIMM, T. *User's Guide to Rapid Prototyping*. [S.l.]: Society of Manufacturing Engineers (SME), 2004. ISBN 978-1-61344-986-8, 978-0-9823843-8-1, 0-87263-697-6.

HADI, A. et al. Developing an adaptable pipe inspection robot using shape memory alloy actuators. *Journal of Intelligent Material Systems and Structures*, SAGE Publications Sage UK: London, England, v. 31, n. 4, p. 632–647, 2020.

ISMAIL, I. N. et al. Development of in-pipe inspection robot: A review. In: IEEE. *2012 IEEE Conference on Sustainable Utilization and Development in Engineering and Technology (STUDENT)*. [S.l.], 2012. p. 310–315.

JONES, H. et al. A robot for nondestructive assay of holdup deposits in gaseous diffusion piping. *arXiv preprint arXiv:1901.10341*, 2019.

JOSEPH, L. *Robot Operating System (ROS) for Absolute Beginners: Robotics Programming Made Easy*. 1. ed. [S.l.]: Apress, 2018. ISBN 1484234049, 978-1484234044.

JOSEPH, L.; CACACE, J. *Mastering ROS for Robotics Programming: Design, build, and simulate complex robots using the Robot Operating System*. 2. ed. [S.l.]: Packt Publishing Ltd, 2018. ISBN 178847452X, 9781788474528, 1788478959, 9781788478953.

KAKOGAWA, A.; MURATA, K.; MA, S. Automatic t-branch travel of an articulated wheeled in-pipe inspection robot using joint angle response to environmental changes. *IEEE Transactions on Industrial Electronics*, IEEE, 2022.

KAZEMINASAB, S.; BANKS, M. K. A localization and navigation method for an in-pipe robot in water distribution system through wireless control towards long-distance inspection. *IEEE Access*, IEEE, v. 9, p. 117496–117511, 2021.

KIM, D.-W. et al. Force adjustment of an active pipe inspection robot. In: IEEE. *ICCAS-SICE, 2009*. [S.l.], 2009. p. 3792–3797.

KOUBAA, A. *Robot Operating System (ROS)*. 1. ed. [S.l.]: Springer International Publishing, 2016. (Studies in Computational Intelligence 625). ISBN 978-3-319-26052-5, 978-3-319-26054-9.

KWON, Y.-S. et al. A pipeline inspection robot with a linkage type mechanical clutch. In: IEEE. *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. [S.l.], 2010. p. 2850–2855.

KWON, Y.-S. et al. A flat pipeline inspection robot with two wheel chains. In: IEEE. *2011 IEEE International Conference on Robotics and Automation*. [S.l.], 2011. p. 5141–5146.

LI, H. et al. Development of a pipeline inspection robot for the standard oil pipeline of china national petroleum corporation. *Applied Sciences*, MDPI, v. 10, n. 8, p. 2853, 2020.

MAHTANI, A. et al. *Effective Robotics Programming with ROS*. 3. ed. [S.l.]: Packt Publishing Ltd, 2016. ISBN 978-1786463654.

- MAZZARI, V. *Robotic simulation scenarios with Gazebo and ROS. Génération Robots*. 2015. Disponível em: <https://www.generationrobots.com/blog/en/robotic-simulation-scenarios-with-gazebo-and-ros/>. Acesso em: 05 de dezembro 2021.
- MCGREGOR, C. et al. Axial crushing of triaxially braided composite tubes at quasi-static and dynamic rates. *Composite Structures*, Elsevier, v. 157, p. 197–206, 2016.
- MELLO-JR, C. et al. Tatubot–robotic system for inspection of underground cable system. In: IEEE. *Robotic Symposium, 2008. LARS'08. IEEE Latin American*. [S.l.], 2008. p. 170–175.
- MILLS, G. H. et al. Miniature magnetic robots for in-pipe locomotion. In: *Memorias de Congresos UTP*. [S.l.: s.n.], 2018. p. 289–300.
- MIYASAKA, K.; KAWANO, G.; TSUKAGOSHI, H. Long-mover: Flexible tube in-pipe inspection robot for long distance and complex piping. In: IEEE. *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. [S.l.], 2018. p. 1075–1080.
- NASSIRAEI, A. A. et al. Concept and design of a fully autonomous sewer pipe inspection mobile robot “kantaro”. In: IEEE. *Robotics and Automation, 2007 IEEE International Conference on*. [S.l.], 2007. p. 136–143.
- NAYAK, A.; PRADHAN, S. Design of a new in-pipe inspection robot. *Procedia Engineering*, Elsevier, v. 97, p. 2081–2091, 2014.
- NEWMAN, W. *A Systematic Approach to Learning Robot Programming with ROS*. [S.l.]: Chapman and Hall/CRC, 2017.
- NIKU, S. B. *Introduction to Robotics: Analysis, Control, Applications*. 2nd. ed. [S.l.]: Wiley, 2010. ISBN 0470604468, 9780470604465.
- NÚÑEZ, A. et al. A pneumatic robotic system for inspection of underground electrical conduit. In: IEEE. *2013 Symposium on Computing and Automation for Offshore Shipbuilding*. [S.l.], 2013. p. 57–62.
- OGATA, K. *Engenharia de Controle Moderno*. 5. ed. [S.l.]: Pearson / Prentice Hall, 2010.
- PETERSEN, K. *ECE 4960/5960: Fast Robots*. 2022. Disponível em: <https://cei-lab.github.io/ECE4960-2022/Lab10.html>. Acesso em: 17 de novembro 2022.
- PHU, N. D. et al. A new fuzzy pid control system based on fuzzy pid controller and fuzzy control process. *International Journal of Fuzzy Systems*, Springer, v. 22, n. 7, p. 2163–2187, 2020.
- QUIGLEY, M.; GERKEY, B.; SMART, W. D. *Programming Robots with ROS: a practical introduction to the Robot Operating System*. [S.l.]: “O’Reilly Media, Inc.”, 2015.
- ROH, S.-g.; CHOI, H. R. Differential-drive in-pipe robot for moving inside urban gas pipelines. *IEEE transactions on robotics*, IEEE, v. 21, n. 1, p. 1–17, 2005.

ROS.ORG, R. W. *tf*. 2017. Disponível em: <http://wiki.ros.org/tf>. Acesso em: 30 de novembro 2021.

ROS.ORG, R. W. *Using Xacro to Clean Up a URDF File*. 2018. <http://wiki.ros.org/urdf/Tutorials/Using%20Xacro%20to%20Clean%20Up%20a%20URDF%20File>. Acesso em: 11 de novembro de 2019.

ROS.ORG, R. W. *ROS/ Concepts*. 2022. Disponível em: <https://wiki.ros.org/ROS/Concepts>. Acesso em: 30 de novembro 2022.

ROSSOMANDO, F. G.; SORIA, C. M. Identification and control of nonlinear dynamics of a mobile robot in discrete time using an adaptive technique based on neural pid. *Neural Computing and Applications*, Springer, v. 26, n. 5, p. 1179–1191, 2015.

ROUSSIALIAN, M. et al. Design and development of a pipeline inspection robot. In: *Mechanism, Machine, Robotics and Mechatronics Sciences*. [S.l.]: Springer, 2019. p. 43–52.

SAAD, M. S.; JAMALUDDIN, H.; DARUS, I. Z. Pid controller tuning using evolutionary algorithms. *Wseas transactions on Systems and Control*, WSEAS, v. 7, n. 4, p. 139–149, 2012.

SAENZ, J. et al. Robotic systems for cleaning and inspection of large concrete pipes. In: IEEE. *Applied Robotics for the Power Industry (CARPI), 2010 1st International Conference on*. [S.l.], 2010. p. 1–7.

SANDLER, B.-Z. *Robotics: Designing the Mechanisms for Automated Machinery*. 2. ed. [S.l.]: Academic Press, 1999. ISBN 0126185204,9780126185201.

SHERWOOD, D. R.; WHISTANCE, D. J. *The piping guide: For the design and drafting of industrial piping systems*. [S.l.]: Syntek, 1991.

SICILIANO, B. et al. *Robotics: modelling, planning and control*. [S.l.]: Springer Science & Business Media, 2010.

SIEGWART, R.; NOURBAKHSI, I. R.; SCARAMUZZA, D. *Introduction to Autonomous Mobile Robots*. 2nd. ed. [S.l.]: The MIT Press, 2011. (Intelligent Robotics and Autonomous Agents series). ISBN 0262015358,9780262015356.

SUZUMORI, K. et al. Micro inspection robot for 1-in pipes. *IEEE/ASME transactions on mechatronics*, IEEE, v. 4, n. 3, p. 286–292, 1999.

TĂȚAR, O.; MANDRU, D.; ARDELEAN, I. Development of mobile minirobots for in pipe inspection tasks. *Mechanika*, v. 68, n. 6, 2007.

TELLES, P. C. da S. *Tubulações Industriais. Materiais, Projeto, Montagem*. Engenharia. [S.l.]: LTC - GRUPO GEN, 1996. (Em Português do Brasil). ISBN 8521612893,9788521612896.

TELLEZ, R. *[ROS Q&A] 053 - How to Move a Robot to a Certain Point Using Twist*. 2017. Disponível em: <https://www.theconstructsim.com/ros-qa-053-how-to-move-a-robot-to-a-certain-point-using-twist/>. Acesso em: 01 de novembro 2022.

TOURAJIZADEH, H. et al. Design of a new steerable in-pipe inspection robot and its robust control in presence of pipeline flow. *Journal of Mechanical Engineering and Sciences*, v. 14, n. 3, p. 6993–7016, 2020.

TRAN, N. C. T. et al. Study on control of cleaning robot for fertilizer pipeline. 2014.

TRUONG-THINH, N.; NGOC-PHUONG, N.; PHUOC-THO, T. A study of pipe-cleaning and inspection robot. In: IEEE. *Robotics and Biomimetics (ROBIO), 2011 IEEE International Conference on*. [S.l.], 2011. p. 2593–2598.

ZHANG, Y.; YAN, G. In-pipe inspection robot with active pipe-diameter adaptability and automatic tractive force adjusting. *Mechanism and Machine Theory*, Elsevier, v. 42, n. 12, p. 1618–1631, 2007.

ZHENG, J.; LI, M. Path tracking control of patrol unmanned vehicle based on gwo-pid. 2022.

ZULU, A.; JOHN, S. A review of control algorithms for autonomous quadrotors. *arXiv preprint arXiv:1602.02622*, 2016.

APÊNDICE A CRIAÇÃO DA SIMULAÇÃO COMPUTACIONAL

A.1 CONFIGURANDO O *WORKSPACE*

Para a configuração do *Workspace* teremos como base e adaptação a literatura de Fairchild e Harman (2016). O primeiro passo é entender o *catkin_worksapce*, que trata de um diretório (pasta) na qual pode-se criar ou modificar pacotes *catkin* existentes. Esta, estrutura *catkin*, simplifica o processo de compilação e instalação dos seus pacotes ROS. O sistema ROS usado foi da primeira versão e o sistema operacional usado foi o Ubuntu 18.04.

Como foi visto, sobre a definição de *catkin_workspace* que pode conter uma pasta, como também pode conter até três ou mais subdiretórios diferentes como: */build*, */dev* e */src*¹. Onde cada um tem uma função distinta no processo de desenvolvimento e aprimoramento do software.

Após o entendimento conceitual do *catkin_workspace*, passamos para a criação do mesmo e para sua construção inicial, chamamos este arquivo de *catkin_ws*, usando o seguinte comando no terminal:

```
$ mkdir -p ~/ catkin_ws / src
```

Para compilar o *workspace*, deve-se originar o ambiente ROS a fim de obter acesso às funções ROS:

```
$ source /opt/ros/melodic/setup.bash
```

Alterna-se para o diretório *src* de origem criado anteriormente.

```
$ cd ~/ catkin_ws / src
```

Inicia-se um novo *catkin_workspace*, usando:

```
$ catkin_init_workspace
```

Após realizar os passos acima, mesmo que ainda o *workspace* esteja vazio e não há pacotes na pasta *src* (apenas um único link CMakeLists.txt), pode-se construir o *catkin_workspace* utilizando os seguintes comandos:

¹ A pasta *src* é onde os pacotes são mantidos, se é desejado criar ou compilar um pacote, deve-se copiar esses para a pasta *src* (JOSEPH, 2018)

```
$ cd ~/ catkin_ws /
```

```
$ catkin_make /
```

O comando *catkin_make* cria o *catkin workspace*. Se for visualizado o conteúdo atual da pasta agora, deverá ter as pastas de *build* e *devel*. No caso do diretório *devel*, há várias arquivos *setup.*Sh*. Logo após analisar isso, falta fornecer o arquivo *setup.bash* para sobrepor esse *workspace* sobre o ambiente ROS que está trabalhando:

```
$ source ~/ catkin_ws / devel / setup.bash
```

Realizado todos as etapas anteriores, e também de adicionar o comando acima ao arquivo *.bashrc* utilizando: **\$ echo "fonte ~/ catkin_ws / devel / setup.bash" >> ~/ .bashrc**.

Desenvolvido este passo parte-se para a criação dos Pacotes ROS, que será apresentado no próximo seção.

A.2 CRIANDO OS PACOTE ROS

Na seção A.1, foi definido o *catkin workspace*. Agora, podemos desenvolver e criar os pacotes ROS. Este passo é adaptado de Joseph e Cacace (2018).

Estes pacotes, citados acima, possuem *nodos* de amostra para demonstrar o funcionamento dos tópicos, mensagens, serviços, entre outros. O comando *catkin_create_pkg* foi usado para criar um pacote ROS.

Primeiramente, para o desenvolvimento do pacote ROS foi criado a pasta *src* do *catkin workspace*, então se inicia com o seguinte comando:

```
$ cd ~/catkin_ws/src
```

Realizado a etapa anterior, cria-se o pacote, utilizando o seguinte comando:

```
$ catkin_create_pkg package_name [dependência1] [dependência2]
```

O nome usado foi *tatubot_ni* (*tatubot* é referente ao projeto Mello-Jr et al. (2008), e *ni* são as iniciais de navegação e inspeção.), com isso o comando para a criação deste pacote foi o seguinte:

```
$ catkin_create_pkg tatubot_ni roscpp rospy urdf rviz xacro std_msgs tf
```

Realizado o comando, se observa os pacotes *urdf*, *xacro* e *rviz*. Eles devem ser criados, porque segundo Joseph e Cacace (2018), o pacote depende principalmente destas dependências para o funcionamento do sistema. As demais dependências nos pacotes são descritas:

- **roscpp**: esta dependência foi incluída porque é a implementação do ROS em C++, onde é uma biblioteca do cliente ROS que fornece APIs para desenvolvedores de C++ para desenvolver *nodos* ROS com tópicos, serviços, parâmetros e outros (JOSEPH; CACACE, 2018);
- **rospy**: esta é a biblioteca do cliente ROS para *Python*. Sua vantagem é a economia de tempo de desenvolvimento. Neste pacote, ainda pode-se criar um *nodo* ROS em menos tempo do que com o *roscpp*. Segundo Joseph (2018), é a forma ideal para

aplicações de prototipagem rápida, porém o desempenho é inferior comparado ao *roscpp*. O autor ainda ressalta, que grande parte das ferramentas de linha de comando, no ROS, são codificadas usando *rospy* com *roslaunch* e *roscore*;

- **std_msgs**: este pacote contém tipos de dados primitivos do ROS básico, como por exemplo: número inteiro, número flutuante, *string*, matriz e entre outros. Pode-se ainda utilizar de forma direta esses tipos de dados em *nodos* sem definir uma nova mensagem ROS (JOSEPH; CACACE, 2018);

- **tf**: este pacote é o que permite ao usuário acompanhar vários quadros de coordenadas ao longo do tempo, ele também mantém a ligação entre os quadros de coordenadas em uma estrutura de árvore armazenada em *buffer* no tempo. (ROS.ORG, 2017). Segundo ROS.org (2017), o *tf* ainda permite ao usuário transformar pontos, vetores, outros., entre quaisquer dois quadros de coordenadas em qualquer momento a qual se deseja;

Seguindo a literatura de Joseph e Cacace (2018), após realizar e desenvolver a criação do pacote, pode ainda ser adicionado manualmente dependências adicionais, as quais deverão ser editadas nos arquivos *CMakeLists.txt* e *package.xml*. Executando os passos anteriores corretamente a seguinte mensagem, indicando que o pacote foi criado, é apresentado na Figura A.1:

Figura A.1: Mensagem do terminal de confirmação da criação do pacote ROS.

```
Created file tatubot_ni/CMakeLists.txt
Created file tatubot_ni/package.xml
Created folder tatubot_ni/include/tatubot_ni
Created folder tatubot_ni/src
Successfully created files in /home/vinicius/catkin_ws/src/tatubot_ni. Please adjust the values in package.xml.
```

Fonte: Autor.

Confirmado a verificação da mensagem da Figura A.1 indicando a criação do pacote, constrói-se o pacote sem adicionar *nodos*, utilizando o comando *catkin_make*, este deve ser executado a partir do caminho do *catkin_workspace*. Este comando a seguir mostra como criar o pacote ROS vazio:

```
~/catkin_ws $ catkin_make
```

Concluído as etapas anteriores, pode-se dar início ao desenvolvimento e adição de *nodos* a pasta *src* deste pacote, que terá seguimento nas próximas seções.

A.3 CRIAÇÃO DOS ARQUIVOS PARA OS DOIS AMBIENTES DE ATUAÇÃO

Nas seções anteriores, foi definido o *catkin_workspace*, sendo a pasta de possíveis modificações e criações de pacotes *catkin* existentes, e os pacotes ROS, as quais foram desenvolvidos na seção anterior. Com todas estas ferramentas definidas, parte-se para a criação dos ambientes de atuação, onde o robô proposto visa navegar no interior de dutos cuja suas seções transversais podem ser circulares ou retangulares. Para a construção

destes locais de trabalho foi baseado e adaptado a obra de Gazebo (2014b).

Conforme o parágrafo acima, se teve um breve entendimento dos ambientes de atuação, para iniciar esta construção devemos procurar a pasta *.gazebo*, que é um arquivo oculto, após a instalação da plataforma ROS, localizado na área de trabalho do sistema Ubuntu 18.04. A pasta *.gazebo* contém alguns arquivos e documentos que são gerados após a instalação, para este trabalho foi utilizado a pasta *models*.

Após esta análise, podemos configurar o diretório do modelo e passamos para a construção dos dois tubos de navegação. Estes ambientes são chamados de: tubo-circular e tubo-retangular. Assim, os comandos utilizados para criar estes arquivos são:

```
$ mkdir -p ~/ .gazebo / models / tubo-circular
```

```
$ mkdir -p ~/ .gazebo / models / tubo-retangular
```

Em seguida, é criado os arquivos de configuração para o caso do tubo de seção circular e retangular.

```
$ ~/ .gazebo / models / tubo-circular / model.config
```

```
$ ~/ .gazebo / models / tubo-retangular / model.config
```

Criado o documento de configuração, ainda foi necessário construir as características físicas e visuais dos dutos. Os comandos para gerar estes arquivos são descritos a seguir:

```
$ mkdir -p ~/ .gazebo / models / tubo-circular / model.sdf
```

```
$ mkdir -p ~/ .gazebo / models / tubo-retangular / model.sdf
```

Nas etapas anteriores foi criado arquivos que configuram o ambiente e caracterizam as aparências físicas do mesmo. Para gerar estas características visuais foi necessário desenvolver a pasta *meshes*, que irá contém as imagens 3D, no formato STL. O comando para a criação deste diretório tanto no caso do tubo circular como do retangular, são apresentados:

```
$ mkdir -p ~/ .gazebo / models / tubo-circular / meshes
```

```
$ mkdir -p ~/ .gazebo / models / tubo-retangular / meshes
```

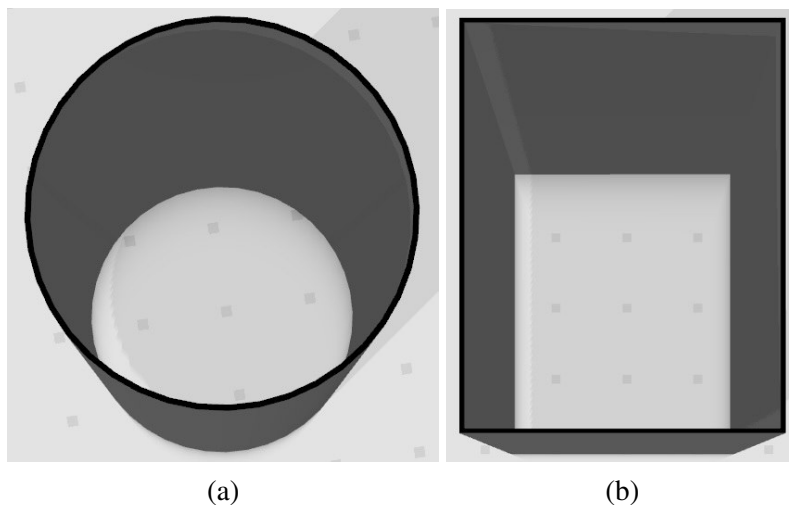
Conforme mencionado anteriormente, as pastas *meshes*, em ambos ambientes, terão os seus diretórios, de formato STL, inseridos.

E as imagens 3D dos ambientes de atuação são visualizados na Figura A.2:

Realizado todos os passos, agora foi possível inserir os ambientes no simulador seguindo dois procedimentos: o primeiro foi estar com gazebo aberto, e em seguida ir na aba *Insert* e adicionar os tubos no software. A Figura A.3 mostrar esta barra de ferramenta do gazebo que permite inserir os dutos de seções transversais circulares e retangulares:

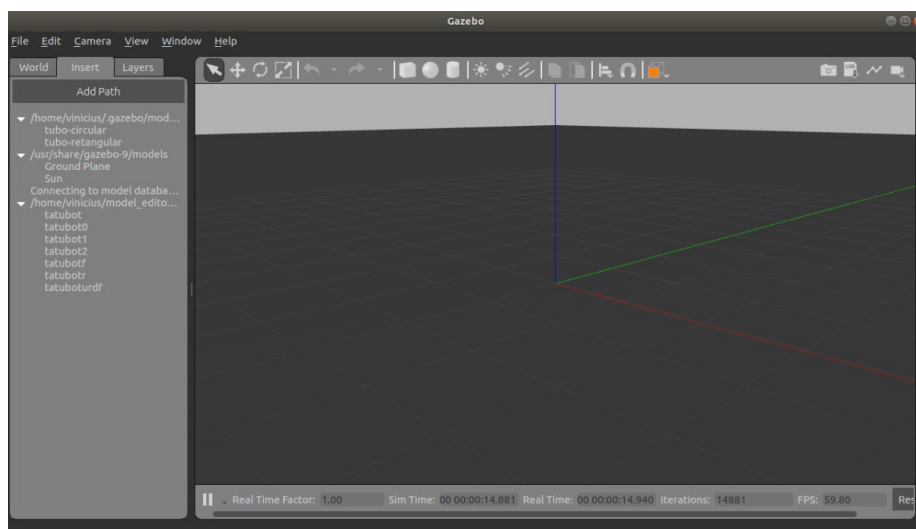
Finalizado a criação dos dois arquivos para o robô executar o processo de navegação nos ambientes de atuação, pode-se dar início ao desenvolvimento do espaço do equipamento robótico, que terá seguimento na seção seguinte.

Figura A.2: (a) Imagem CAD do tubo circular no formato STL; e (b) Imagem CAD do tubo retangular no formato STL.)



Fonte: Autor.

Figura A.3: Barra de ferramenta *Insert* para adicionar os ambientes de atuação no Gazebo.

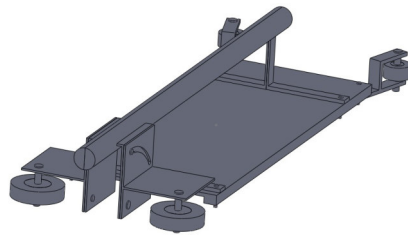


Fonte: Autor.

A.4 CONSTRUÇÃO DO ARQUIVO DO ROBÔ MÓVEL PARA NAVEGAÇÃO EM DUTOS COM SEÇÕES TRANSVERSAIS CIRCULARES E RETANGULARES

O robô construído para este trabalho foi um equipamento robótico móvel do tipo que se prende nas paredes do tubo com um conjunto de 4 rodas, as rodas traseiras possuem um sistema de molas que possibilitam o mesmo estar sempre tracionado na parede, e as rodas frontais que proporcionam o movimento linear e o de rotação, ocasionados pelo ângulo de ataque das rodas frontais. A Figura A.4 mostra o dispositivo robótico (feito em CAD), o qual foi desenvolvido no software Gazebo:

Figura A.4: Imagem do robô feita em CAD.



Fonte: Autor.

Para o desenvolvimento deste protótipo no simulador teremos como base e adaptação a literatura de Mazzari (2015).

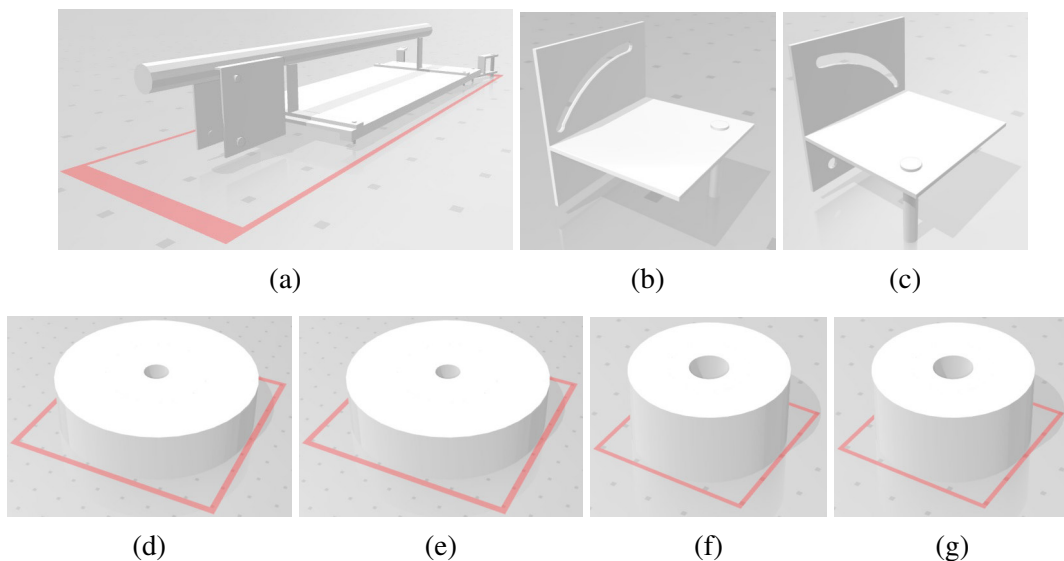
Para iniciar a criação deste sistema robótico, se deve primeiramente ter o *catkin_workspace* definido e os pacotes ROS configurados, como foi abordado nas seções A.1 e A.2. Com estes procedimentos estabelecidos, foi criado a pasta *tatubot_ni*, a qual contém os arquivos gerados pelos pacotes ROS.

Com todas estas ferramentas disponíveis, parti-se para a criação do robô, no diretório *tatubot_ni*, onde podemos criar os arquivos *meshes* e *urdf* para a construção do dispositivo. Nesta primeira etapa foi gerado a pasta *meshes* utilizando o comando:

```
$ mkdir -p ~/catkin_ws/src/tatubot_ni/meshes
```

Criado o diretório *meshes*, inserimos as imagens 3D com os componentes do equipamento robótico na pasta *meshes*, as peças de formato STL é mostrada na Figura A.5:

Figura A.5: (a) Chassi do robô em formato STL; (b) Braço direito do robô em formato STL; (c) Braço esquerdo do robô em formato STL; (d) Roda frontal direita do robô em formato STL; (e) Roda frontal direita e esquerda do robô em formato STL; (f) Roda traseira direita do robô em formato STL; e (g) Roda traseira esquerda do robô em formato STL.



Fonte: Autor.

Finalizado o arquivo *meshes*, o qual foi usado para gerar as componentes visuais do sistema robótico. Precisamos ainda, dentro do pacote de descrição (*tatubot_ni*), criar a pasta *urdf*, com o seguinte comando:

```
$ mkdir -p ~/catkin_ws/src/tatubot_ni/urdf
```

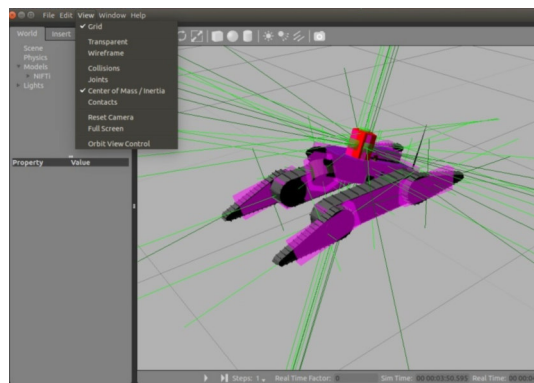
Criado este diretório *urdf*, inserimos os dois arquivos de descrição que possuem: o formato urdf usado para descrever todos os elementos do robô, e o formato xacro, que foi utilizado por ser útil em realizar descrições mais curtas e claras da estrutura. Os comandos para gerar estes documentos são mostrados a seguir:

```
$ ~/catkin_ws/src/tatubot_ni/urdf/01-tatubot_ni.urdf
```

```
$ ~/catkin_ws/src/tatubot_ni/urdf/02-tatubot_ni.urdf.xacro
```

Destes arquivos criados, o mais importante foi o *02-tatubot_ni.urdf.xacro*, porque este ajuda a simplificar e facilitar a leitura dos arquivos URDF. E também outra observação a ser levada em conta, são as componentes do modelo proposto, os modelos visuais são os mostrado na Figura A.5, e para suas estruturas físicas devemos, segundo Gazebo (2014a), simplificar o modelo utilizando formas geométricas regulares na sua composição e a sua inércia deve ter mais ou menos o mesmo tamanho destas formas geométricas, pois, o autor ainda ressalta que corpos não homogêneos podem não funcionar gerando inconsistências no comportamento do robô. Na Figura A.6 é mostrado um exemplo de como deve ser feito.

Figura A.6: Inércia do robô no Gazebo



Fonte: Adaptado de Gazebo (2014a, p. 1)

A.5 CRIAÇÃO DOS ARQUIVOS LAUNCH E CONFIG

As seções anteriores trataram da criação dos arquivos: do *catkin_workspace*, dos pacotes ROS, dos ambientes de atuação e do robô. Determinado e construído estes acessórios, parte-se para o desenvolvimento dos arquivos de configuração e de ativação da simulação. Nesta seção se utilizou da adaptação do trabalho de Fairchild e Harman (2016).

Primeiramente, para a construção destes arquivos, inicia-se à criação: da pasta de configuração chamada *config*, e do diretório de ativação, cujo o nome é *launch*. Definido

isto, se inicia as construções destes a partir da utilização dos seguinte comandos:

```
$ mkdir -p ~/ catkin_ws / src / tatubot_ni / config
```

```
$ mkdir -p ~/ catkin_ws / src / tatubot_ni / launch
```

Após a construção de ambas as pastas, no pacote *tatubot_ni*, podemos dar início a elaboração do documento, do diretório *config*. Neste ambiente, o arquivo gerado possui a extensão YAML, que é uma linguagem de marcação comumente usada para parâmetros ROS, sendo útil utilizar os arquivos codificados em YAML para poder definir as propriedades ROS no servidor de parâmetros. Para o caso do equipamento robótico, o arquivo YAML foi desenvolvido para manter as configurações do controlador conjunto e também carregar por meio do arquivo de inicialização do controle. Para este robô foram definidos quatro controladores do tipo PID. Os ganhos do PID foram ajustados para o controle de cada velocidade angular e de cada ângulo de ataque. Então, para esta pasta teremos o arquivo *controllers.yaml*, para ajustar os valores dos quatro controladores das rodas do robô. O comando para gerar este documentos foi o seguinte:

```
$ ~/ catkin_ws / src / tatubot_ni / config / controllers.yaml
```

Criado e codificado o documento, este mesmo foi salvo na pasta *config* do pacote *tatubot_ni*.

Concluído a etapa de configuração, ainda foi necessário construir arquivos que permitam a inicialização da simulação. E a melhor forma de iniciar os *tópicos* das velocidades angulares e os ângulos de ataque das rodas frontais do robô, foi criando um arquivo *launch* para carregar as propriedades no servidor de parâmetros, e começar todos os *nodos*. Este documento contém instruções que carregam as configurações do controlador no servidor de parâmetros a partir do arquivo: *controllers.yaml*. E ainda possui *nodos* que possibilitam criar os *tópicos* a partir dos pacotes. Neste trabalho foram programados e criados dois arquivos, que forma salvos na pasta *launch*, estes são: 01-tatubot.launch e world.launch. Os comandos para desenvolver estes documentos:

```
$ ~/ catkin_ws / src / tatubot_ni / launch / 01-tatubot_ni.launch
```

```
$ ~/ catkin_ws / src / tatubot_ni / launch / world.launch
```

Estes dois documentos foram salvos na pasta *launch*, também do pacote *tatubot_ni*.

Foi criado os arquivos de inicialização na pasta *launch*, onde o documento *01-tatubot_ni.launch*, carrega os cinco *tópicos*:

- */tatubotni/right_front_wheel_controller/command;*
- */tatubotni/left_front_wheel_controller/command;*
- */tatubotni/right_front_hinge_controller/command;*
- */tatubotni/left_front_hinge_controller/command;*
- */ground_truth.*

E também o *world.launch*, a qual este permitiu que o usuário possa escolher em qual dos cenários de atuação que o robô navegará dentro do gazebo

Para podermos iniciar o arquivo de extensão *.launch*, devemos utilizar o comandos abaixo:

```
$ roslaunch tatubot_ni 01-tatubo_ni.launch
```

A.6 CRIAÇÃO DOS ARQUIVOS MY_ODOM_PUBLISHER E SIMPLE_CONTROLLER

Nesta seção, foi tratado os procedimentos de criação dos arquivos que permitiu a utilização do *ground_truth* e da estratégia de controle. Para a realização deste Apêndice será adaptado as literaturas de Ezquerro (2017) e Tellez (2017).

Inicialmente, foi preciso criar dois pacotes para estes dois códigos. Ambos diretórios foram construídos dentro do *catkin_workspace*, sendo adicionadas no arquivo *src*. Os comandos para estes pacotes são os seguintes:

```
$ catkin_create_pkg my_odom_publisher rospy
```

```
$ catkin_create_pkg simple_controller rospy
```

Após a criação destes dois pacotes, se parte para a visualização do interior de cada um deles. Para em seguida, criar os arquivos *odom.py* na pasta *src* (do pacote *my_odom_publisher*), e os arquivos *controller_tc* e *controller_tr* (localizados no pacote *simple_controller*).

O arquivo *odom.py* foi inicializado a partir da ativação do arquivo *01-tatubot.launch*, gerando o tópico */ground_truth*. Este irá informar as informações das posições lineares e angulares do robô. Já os arquivos *controller_tc.py* e *controller_tr.py* iniciam os programas a partir de um destes comandos:

```
$ rosrn simple_controller controller_tc.py
```

```
$ rosrn simple_controller controller_tr.py
```

O comando **rosrn simple_controller controller_tc.py** foi para o caso do sistema robótico navegar no interior do duto circular e o comando **rosrn simple_controller controller_tr.py** foi para o caso do usuário escolher o cenário retangular para o robô navegar. Vale ressaltar que em ambas circunstâncias só é possível realizar um destes comandos após o ambiente e o robô estiverem carregados no simulador Gazebo.