ENGENHARIA DE COMPUTAÇÃO

Guilherme Alan Ritter

MODELO GENÉRICO DE PLATAFORMA ROBÓTICA OMNIDIRECIONAL EM CÓDIGO ABERTO

Santa Cruz do Sul, dezembro de 2016

ENGENHARIA DE COMPUTAÇÃO

Guilherme Alan Ritter

MODELO GENÉRICO DE PLATAFORMA ROBÓTICA OMNIDIRECIONAL EM CÓDIGO ABERTO

Professor Me. Taiser Tadeu Teixeira Barros
Orientador

Santa Cruz do Sul, dezembro de 2016

ENGENHARIA DE COMPUTAÇÃO

Guilherme Alan Ritter

MODELO GENÉRICO DE PLATAFORMA ROBÓTICA OMNIDIRECIONAL EM CÓDIGO ABERTO

Professor Dr. Leonel Pablo Tedesco
Avaliador

ENGENHARIA DE COMPUTAÇÃO

Guilherme Alan Ritter

MODELO GENÉRICO DE PLATAFORMA ROBÓTICA OMNIDIRECIONAL EM CÓDIGO ABERTO

Professor Me. Werner Haetinger
Avaliador

ENGENHARIA DE COMPUTAÇÃO
Guilherme Alan Ritter

MODELO GENÉRICO DE PLATAFORMA ROBÓTICA OMNIDIRECIONAL EM CÓDIGO ABERTO

Trabalho de Conclusão II apresentado ao Curso de Engenharia da Computação da Universidade de Santa Cruz do Sul, como requisito parcial para a obtenção do título de Bacharel em Engenharia da Computação.

Orientador: Professor Me. Taiser Tadeu Teixeira Barros

Santa Cruz do Sul, dezembro de 2016

ABSTRACT

Despite robotics being well established in industry since the 1960s, and being consolidated in other areas like defense, health and services, there still are many areas that can be explored. However, the research still presents many challenges. One way to encourage research is to increase the number or researchers, and one way to do that is to facilitate the access of students to the subject. Since robotics frequently involves complex math, the use of a tool with solved solutions can anticipate the interest of students for robotics. With the intention to strenghten research and to provide a base for more ambitious projects, the objective of this work is to develop an omnidirectional mobile robotic platform generic model that moves according to varied commands. To provide the robot for future researches, all of the documents necessary to its construction, including source code(s), will be shared on an on-line platform for the community in a free way. This work is compared to similar works, justifying it's relevance. Several tests are realized and their results are discussed, resulting in the validation of the control program. Future improvements for the work are mentioned.

Keywords: Mobile robotics, Omnidirectional, Free software, Open source.

RESUMO

Apesar de a robótica ter presença garantida na indústria desde os anos 60, e de estar se consolidando em outras áreas como defesa, saúde e serviços, ainda há muitas áreas que podem ser exploradas. Porém, a pesquisa ainda apresenta muitos desafios. Uma maneira de incentivar a pesquisa é aumentar o número de pesquisadores, e uma maneira de fazer isso é facilitar o acesso de estudantes ao assunto. Como a robótica frequentemente envolve cálculos complexos, o uso de uma ferramenta com soluções prontas (como um robô previamente projetado, construído e testado e a matemática que realiza o controle de movimentação) pode adiantar o interesse de alunos pela robótica. Com o intuito de fortalecer a pesquisa e de fornecer uma base para projetos maiores, o objetivo deste trabalho é desenvolver um modelo genérico de plataforma robótica móvel omnidirecional que se move a partir de comandos variados. Para disponibilizar o robô para pesquisas futuras, todos os documentos necessários para a sua construção, incluindo código(s)-fonte, serão compartilhados em uma plataforma *on-line* para a comunidade de forma livre. Este trabalho é comparado com trabalhos similares, justificando sua relevância. Vários testes são realizados e seus resultados discutidos, resultando na validação do programa de controle. Melhorias futuras para o trabalho são mencionadas.

Palavras-chave: Robótica móvel, Omnidirecional, Software livre, Código fonte aberto.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS

2D	duas dimensões
3D	três dimensões
ACM	Association for Computing Machinery
CAD	Computer Aided Design
CC0	Creative Commons Zero
CC	corrente contínua
CD	Compact Disk
CMU	Carnegie Mellon University
CSI	Camera Serial Interface
DEE/UFCG	Departamento de Engenharia Elétrica da Universidade de Campina Grande
DMT	Département de microtechnique
DSI	Digital Serial Interface
EaD	Ensino à Distância
EPFL	École Polytechnique Fédérale de Lausanne
GNU	GNU's Not Unix!
PID	Proportional–Integral–Derivative
PCB	printed circuit board
ROS	Robot Operating System
SDF	Simulation Description Format
SMT	surface-mount technology
STEP	Standard for the Exchange of Product model data
TUM	Technische Universität München

SUMÁRIO

1 INTRODUÇÃO

GARCIA et al. (2007) escreve sobre a evolução da robótica:

> Durante os últimos 45 anos, a pesquisa em robótica esteve focada em descobrir soluções para as necessidades técnicas de robótica aplicada. A evolução dos campos de aplicação e de sua sofisticação influenciou tópicos de pesquisa na comunidade robótica. Esta evolução tem sido dominada por necessidades humanas. No início dos anos 1960, a revolução industrial colocou robôs industriais na fábrica para liberar o operador humano de tarefas arriscadas e nocivas. A incorporação posterior de robôs industriais em outros tipos de processos de produção adicionou novos requerimentos que exigiam mais flexibilidade e inteligência em robôs industriais. Atualmente, a criação de novas necessidades e mercados exteriores ao mercado robótico de manufatura tradicional (por exemplo limpeza, desminagem, construção, construção de navios, agricultura) e o mundo de envelhecimento em que vivemos está demandando robôs de campo e de serviço para atender o novo mercado e as necessidades sociais humanas.

Especificamente sobre a robótica industrial, SIEGWART; NOURBAKHSH (2004) escreve:

> A robótica atingiu seu maior sucesso até hoje no mundo da manufatura industrial. Braços robóticos, ou *manipuladores*, compreendem uma indústria de 2 bilhões de dólares. Parafusado pelo seu ombro a uma posição específica da linha de montagem, o braço robótico pode se mover com grande velocidade e exatidão para desempenhar tarefas repetitivas como solda ponto e pintura. Na indústria eletrônica, manipuladores colocam componentes SMT com precisão sobre-humana, fazendo possíveis o telefone portátil e o computador *laptop*.

É possível perceber que a robótica industrial teve um bom tempo de desenvolvimento e que o seu desempenho hoje está em um nível mais do que aceitável, enquanto que as outras áreas da robótica estão em crescente demanda e, consequentemente, maior necessidade de pesquisa. Uma dessas áreas é a da robótica móvel que, de acordo com SIEGWART; NOURBAKHSH (2004), "[...] é um campo jovem"e, de acordo com DU-

DEK; JENKIN (2010), "[...] o desenvolvimento de sistemas que exibem mobilidade é um obstáculo chave". Apesar da robótica móvel ser o contexto deste trabalho, ele pode ser usado para desenvolver outras áreas da robótica, além de outras áreas além da robótica. GHOSH; XI; TARN (1999) escreve sobre outro assunto relacionado à pesquisa (relativamente) recente envolvendo robótica:

> Nos últimos anos, tem havido um interesse crescente na necessidade da fusão de sensores para resolver problemas em planejamento e controle para sistemas robóticos. A aplicação de tais sistemas iria variar de tarefas de montagem em automação industrial até manipulação de materiais em ambientes perigosos e tarefas de serviço no espaço. Dentro do arcabouço de uma abordagem orientada a eventos, a robótica achou novas aplicações em automação, como cirurgia assistida por robô e microfabricação, que apresentam novos desafios para as comunidades de controle, automação e manufatura.

KURFESS (2005) também fala sobre aplicações da robótica que necessitam de mais pesquisa:

> Olhando para a frente há muitas fronteiras na robótica. Muitas das aplicações apresentadas aqui estão em sua infância e terão crescimento considerável. Outras áreas mais maduras terão desenvolvimento continuado, como tem sido o caso desde o *boom* tecnológico que seguiu a Segunda Guerra Mundial. Muitas áreas teóricas mantém possibilidades infindas para expansão — controle não linear, álgebra computacional, geometria computacional, inteligência em ambientes não estruturados, e muito mais. [...]

A organização deste trabalho é apresentada a seguir. O capítulo 2 apresenta os conceitos base do trabalho, como a teoria que fundamenta o trabalho, até os componentes que o formarão, passando por um pouco de contextualização sobre a área do trabalho. O capítulo 3 apresenta os trabalhos que foram considerados relacionados com este, por quais critérios foram assim considerados, e comparações entre os trabalhos a partir dos critérios. O capítulo 4 apresenta o que foi desenvolvido a partir deste trabalho. O capítulo 5 apresenta testes e seus resultados que foram usados para validar o trabalho desenvolvido. O capítulo 6 apresenta as conclusões do autor sobre o trabalho. O capítulo 7 apresenta melhorias a serem desenvolvidas futuramente que foram considerados por agregarem valor significativo ao trabalho.

1.1 Motivação

Conceitos matemáticos que hoje são conhecidos pela maioria das pessoas antigamente eram reservados para a elite. A revolução que mudou essa situação contou, entre outras coisas, com a *Aritmética de Treviso*, livro que ensinava matemática comercial de maneira mais acessível, lançado em 1478. Uma revolução similar começou a partir dos anos 1970, com relação à programação de computadores, que começou a ser ensinada em algumas escolas de ensino fundamental e médio e é considerada por outras (BLIKSTEIN, 2013).

Outra revolução começou por volta de 1999, quando se percebeu que o ensino de tecnologia em geral, como engenharia e projeto, era necessário para desenvolver habilidades fundacionais e adaptativas frente à rápida evolução da tecnologia. Esse ensino teve um período de moderado sucesso, decaiu e ultimamente tem ganhado força novamente, com o auxílio de máquinas de corte a laser e impressoras 3D (BLIKSTEIN, 2013).

Apesar disso, a evasão de alunos de cursos de áreas da tecnologia, tanto em nível técnico como em nível superior, inclusive em cursos oferecidos na modalidade EaD, é um tema atual mas que não recebe a devida atenção no Brasil, ao contrário de outros países e da evasão no ensino fundamental e médio. Vários índices de evasão no Brasil giram em torno de 20%. Esse problema resulta em desperdício de recursos que não geram retorno (LOBO E SILVA FILHO et al., 2007) (JORGE et al., 2010) (CRAVO, 2012). O estudo da evasão em uma instituição de ensino superior privada revela que o segundo principal motivo pela evasão seria o "Atendimento de minhas aptidões e interesses"(DE ASSIS, 2013).

Este trabalho é motivado não somente pelo apoio à pesquisa mais avançada de robótica, como também pela atração de alunos de ensino fundamental e médio para as ciências tecnológicas, e poderá facilmente ser modificado para esse e outros fins, além de ter a intenção de aumentar o interesse de alunos de cursos técnicos e superiores pela robótica.

1.2 Objetivos

O objetivo deste trabalho é desenvolver um modelo genérico de plataforma robótica omnidirecional, tanto a nível de simulação, consistindo de programa a ser executado em simulador para prover processamentos e cálculos necessários para o controle da plataforma, além do modelo gráfico a ser exibido pelo simulador, quanto em nível físico, consistindo de desenhos técnicos, esquemáticos eletrônicos e código fonte para a construção de um protótipo físico, ambos possuindo manuais contendo informações necessárias para suas realizações.

Espera-se que este trabalho possa ser utilizado para promover a pesquisa na área da robótica e incentivar a formação de pesquisadores através da livre disponibilidade de soluções prontas. Através desta plataforma será possível agilizar a implementação de robôs mais complexos, através da existência de uma plataforma móvel pronta, e facilitar o desenvolvimento de melhores soluções de movimentação, através do fornecimento de uma modelo prontamente expansível.

1.2.1 Objetivos específicos

- Estudar a cinemática direta e inversa de robôs omnidirecionais e a odometria dos mesmos.
- Implementar o programa que recebe comandos de movimentação, lê os sensores e atua os motores.
- Desenvolver um modelo do robô utilizando a ferramenta Gazebo.
- Utilizar a simulação par validar a programação.
- Construir um protótipo físico.
- Realizar testes e analisar os resultados.
- Disponibilizar o conteúdo para a comunidade.

2 REFERENCIAL TEÓRICO

Neste capítulo os conceitos que servirão de base para o trabalho serão introduzidos: a robótica móvel, contexto deste trabalho; conceitos matemáticos como cinemática e odometria; a geometria do robô omnidirecional; os componentes físicos do protótipo; as ferramentas para validação e como disponibilizar o trabalho para a comunidade.

2.1 Robótica móvel

Como o fruto do trabalho será uma plataforma robótica omnidirecional, muitos dos seus desafios se encontram no contexto da robótica móvel. Esta seção faz uma breve introdução à mesma.

Robôs podem ser classificados sob vários pontos de vista. Em um desses, eles podem ser divididos em três categorias: industriais, médicos e móveis. Os robôs industriais são formados por uma estrutura mecânica articulada e que podem mover cargas pesadas a grandes velocidade e com grande exatidão. Robôs médicos, de cooperação ou de reabilitação são próteses inteligentes para as pessoas com necessidades físicas especiais, procuram ter a aparência das extremidades humanas e realizar suas funções, e são controlados por sinais nervosos ou musculares. Os robôs móveis são dispositivos de transporte automático, ou seja, são plataformas mecânicas dotadas de um sistema de locomoção capazes de navegar através de um determinado ambiente de trabalho, dotados de certo nível de autonomia para sua locomoção, portando cargas. Suas aplicações podem ser muito variadas e estão sempre relacionadas com tarefas que normalmente são arriscadas ou nocivas para a saúde humana, em áreas como a agricultura, no transporte

de cargas perigosas ou em tarefas de exploração solitárias ou cooperativas junto a outros veículos não tripulados (CALDERÓN ESTEVEZ, 1989 apud SECCHI, 2008).

De acordo com JONES; FLYNN; SEIGER (1999),

> [...] há uma grande variedade de maneiras de se mover sobre uma superfície sólida. Dentre os robôs, os três sistemas mais comuns usam rodas, lagartas/esteiras e pernas. Veículos com rodas são de longe os mais populares por vários motivos práticos. Robôs com rodas são mecanicamente mais simples e fáceis de construir. A razão entre a massa da carga pela do mecanismo também é favorável. Tanto sistemas com pernas como com lagartas/esteiras geralmente requerem *hardware* mais complexo e pesado do que sistemas com rodas projetados para levar a mesma carga. Além disso, uma ampla variedade de dispositivos com rodas, como brinquedos, podem ser modificados para uso robótico. A principal desvantagem das rodas é que, em terreno irregular, elas podem desempenhar mal. Por via de regra, um veículo com rodas tem dificuldade se a altura do objeto que ele deve superar se aproxima do raio das rodas. Uma solução é simplesmente usar rodas que são mais largas comparadas com todas as obstruções prováveis. Em muitas instâncias, porém, isso não é prático.

A figura 1 apresenta os tipos de rodas básicos. (a) Rodas padrão: dois graus de liberdade; rotação ao redor do eixo (motorizado) da roda e do ponto de contato. (b) Roda caster (SICILIANO; SCIAVICCO; ORIOLO, 2009) ou castor (SECCHI, 2008): dois graus de liberdade; rotação ao redor de uma junta de esterço deslocada. (c) Roda sueca: três graus de liberdade; rotação ao redor do eixo (motorizado) da roda, ao redor dos rolantes e ao redor do ponto de contato. (d) Roda esférica: realização tecnicamente difícil (SIEGWART; NOURBAKHSH, 2004).

A escolha de tipos de rodas para um robô móvel é fortemente ligada com a escolha de arranjo das rodas, ou geometria das rodas. Essas duas questões devem ser consideradas simultaneamente no projeto de robôs móveis, pois as características fundamentais de um robô móvel são governadas por essas escolhas: manobrabilidade, controlabilidade e estabilidade. Há uma variedade de configurações de rodas para robôs móveis pelos mesmos se depararem com uma variedade de ambientes, ao contrário de carros, que são feitos para um ambiente padronizado (ruas, estradas, rodovias, etc.) (SIEGWART; NOURBAKHSH, 2004). As tabelas 5 e 6 no anexo A dão uma visão geral sobre geometrias das rodas ordenadas por quantidade.

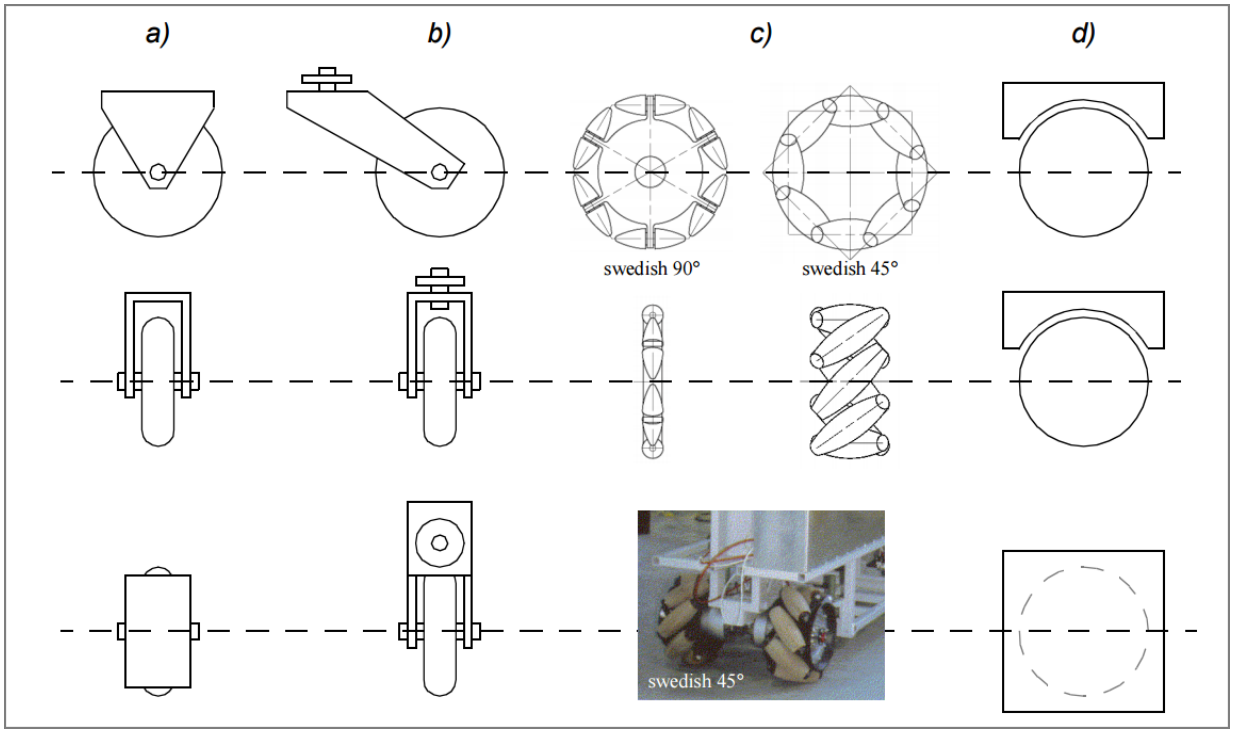


Figura 1: Tipos de rodas básicos usadas por robôs móveis.
Fonte: (SIEGWART; NOURBAKHSH, 2004).

2.1.1 Cinemática

Como o conceito de movimento é ubíquo na área da robótica móvel, é importante definir o termo "cinemática". De acordo com CRAIG (2005),

> A cinemática é a ciência que trata do movimento, sem considerar as forças que causam. Dentro da cinemática, se estudam a posição, velocidade, aceleração e todas as derivadas de maior ordem das variáveis de posição (em relação ao tempo ou a qualquer outra variável).

e com GARCíA (2009),

> [...] Existem duas abordagens dentro da cinemática, chamadas de cinemática direta e cinemática inversa.
>
> O problema da cinemática direta consiste em calcular a posição e orientação do veículo em relação a um sistema de coordenadas fixo, que sem perda de generalidade é a origem do plano XY. [...] em robótica móvel cinemática direta pode ser visto como uma transformação que mapeia pares de trajetórias de velocidade [...] do veículo em trajetórias percorridas pelo veículo no espaço cartesiano, [...]
>
> [...]
>
> Além disso, dada uma trajetória nominal ou desejada [...], pode-se dizer que o problema da cinemática inversa é encontrar um perfil de velocidade [...] que gera a posição e a orientação desejada do veículo, [...]

2.1.2 Roda omnidirecional

De acordo com SIEGWART; NOURBAKHSH (2004),

> A roda Sueca e a roda esférica são ambas modelos que são menos restritas por direcionalidade do que as rodas padrões convencionais. A roda Sueca funciona como uma roda normal, mas provém baixa resistência em outra direção também, as vezes perpendicular à direção convencional, [...] e as vezes a um ângulo intermediário [...]. Os rolantes pequenos conectados ao redor da circunferência da roda são passivos e o eixo principal da roda serve como a única junta atuada ativamente. A vantagem chave deste modelo é que, apesar de a rotação da roda ser atuada apenas ao longo do único eixo principal [...], a roda pode se mover cineticamente com fricção muito baixa ao longo de várias trajetórias possíveis, e não simplesmente para frente e para trás.

A figura 2 mostra dois exemplos de rodas omnidirecionais. Desvantagens das rodas omnidirecionais podem ser causadas pelos seus rolantes, que aumentam os graus de liberdade da roda, resultando em acúmulo de derrapagem, redução da precisão da odometria, aumento na complexidade do projeto (SIEGWART; NOURBAKHSH, 2004), vibrações verticais e posicionamento impreciso causado pela descontinuidade de contato (CHUNG et al., 2010) (HOLMBERG; KHATIB, 2000).

Figura 2: Exemplos de rodas omnidirecionais.
Fonte: (GARCíA, 2009).

Foi escolhido o uso para este trabalho de rodas omnidirecionais com dois conjuntos de roletes. Motivos para essa escolha incluem: menor ruído e oscilação, comparadas com rodas com apenas um conjunto de roletes; o fato de rodas Mecanum causarem forças desnecessárias na estrutura e serem patenteadas, além de possuir menor benefício/custo neste caso. (BEMIS, 2009).

2.2 Robô omnidirecional

Esta sessão apresenta a geometria selecionada para a plataforma. Esta seleção implica no uso de rodas omnidirecionais, apresentadas na seção 2.1. A geometria das rodas do robô omnidirecional é a apresentada pela tabela 6 no anexo A, 5ª linha dentre as de 3 rodas. A figura 3 mostra um exemplo de robô omnidirecional.

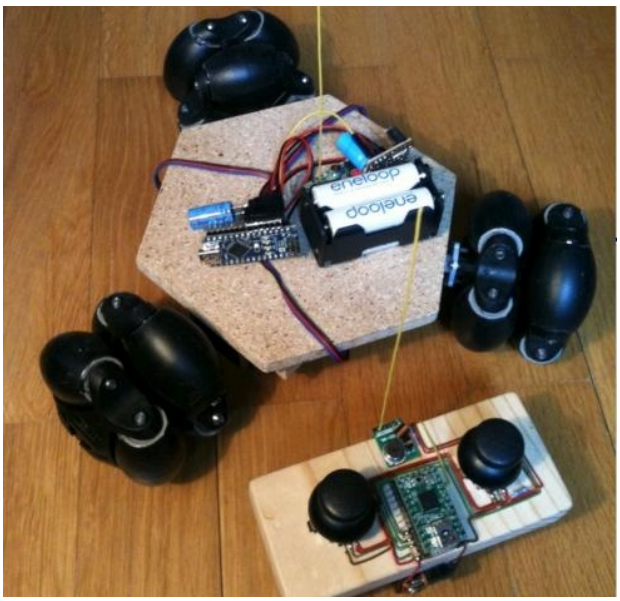


Figura 3: Exemplo de robô omnidirecional de três rodas.
Fonte: (GRITSCH, 2011).

O robô omnidirecional tem "[...] total mobilidade no plano o que significa que ele pode se mover a cada instante em qualquer direção sem qualquer reorientação."(CAMPION; BASTIN; D' ANDRéA-NOVEL, 1996). "Uma plataforma móvel omnidirecional holonômica provê simplicidade de controle em várias aplicações, como em navegação autônoma, manipulação móvel e controle de reboque."(CHUNG et al., 2010).

2.2.1 Holonomia

O termo "holonômico"possui ampla aplicabilidade para várias áreas da matemática, mas, para a robótica móvel, o termo se refere especificamente às restrições cinemáticas do chassi do robô. Um robô holonômico é um robô com zero restrições cinemáticas não holonômicas. Inversamente, um robô não holonômico é um robô com uma ou mais restrições cinemáticas não holonômicas (SIEGWART; NOURBAKHSH, 2004).

Uma restrição cinemática holonômica pode ser expressa como uma função explícita exclusivamente de variáveis de posição, sem usar derivadas dessas variáveis. Uma restrição cinemática não holonômica requer uma relação diferencial, como uma derivada de uma variável de posição. Além disso, ela não pode ser integrada para prover uma restrição em termos de variáveis de posição exclusivamente. Por causa disso, sistemas não holonômicos também são chamados de sistemas não integráveis(SIEGWART; NOURBAKHSH, 2004).

Outra maneira de descrever um robô holonômico é pela relação entre os graus de liberdade de seus atuadores e os graus de liberdade do seu espaço de trabalho ("número de coordenadas necessárias para especificar a configuração do sistema"(HOLMBERG; KHATIB, 2000)). Um robô só é holonômico quando esses dois valores forem iguais. Na robótica móvel, chassis úteis geralmente tem de atingir poses em um espaço de trabalho com dimensionalidade 3 (x, y, θ). Considerando as habilidades "holonômicas"de manobrar ao redor de obstáculos sem afetar a orientação e de se manter apontado para um alvo ao longo de um caminho arbitrário, a forma de holonomia mais relevante para a robótica móvel é a de três graus de liberdade (dos atuadores e do espaço de trabalho). Um robô omnidirecional é, portanto, um robô holonômico com três graus de liberdade nos atuadores (SIEGWART; NOURBAKHSH, 2004). Esta definição desconsidera a composição física do mecanismo, que pode ser criado de diversas maneiras diferentes para atingir a holonomia (HOLMBERG; KHATIB, 2000).

2.3 Cinemática da geometria omnidirecional

Esta seção explica a cinemática aplicada à geometria escolhida. BORENSTEIN; EVERETT; FENG (1996) mostra as equações da cinemática inversa em relação ao sistema de coordenadas da plataforma, enquanto que GONçALVES; LIMA; COSTA (2008) mostra as equações da cinemática inversa em relação ao sistema de coordenadas do mundo. É possível deduzir que a única diferença entre cada sistema de coordenadas é o ângulo entre elas, chamado de θ. Por isso, basta aplicar uma rotação em θ para converter de um sistema de coordenadas para outro. Para comprovar isso, basta realizar a conversão do sistema de coordenadas do mundo para o da plataforma, aplicar essas equações nas equações de BORENSTEIN; EVERETT; FENG (1996), e as equações resultantes serão

análogas às equações de GONçALVES; LIMA; COSTA (2008). O inverso também é válido, ressaltando que o θ deve ser substituído por 0, dado que não há ângulo entre a plataforma e o seu sistema de coordenadas. As equações serão análogas ao invés de iguais pois cada autor utiliza uma convenção para a posição dos vetores das rodas.

Considerando isso, o autor deste trabalho estabeleceu suas convenções, mostradas na figura 4, que ilustra todas as relações geométricas necessárias para a derivação das equações. Nessa figura, $V^m_{x|y}$ são as velocidades da plataforma no plano em relação ao sistema de coordenadas da plataforma, cuja origem coincide com o centro da plataforma, em metros por segundo; $V^w_{x|y}$ são as velocidades da plataforma no plano em relação a um sistema de coordenadas fixo no mundo, em metros por segundo; $V_{1|2|3}$ são as velocidades lineares das rodas, em metros por segundo; ω_p é a velocidade de rotação, em radianos por segundo; L é a distância entre o centro das rodas e o centro da plataforma, em metros; θ é o ângulo entre o sistema de coordenadas do mundo e o sistema de coordenadas da plataforma, em radianos.

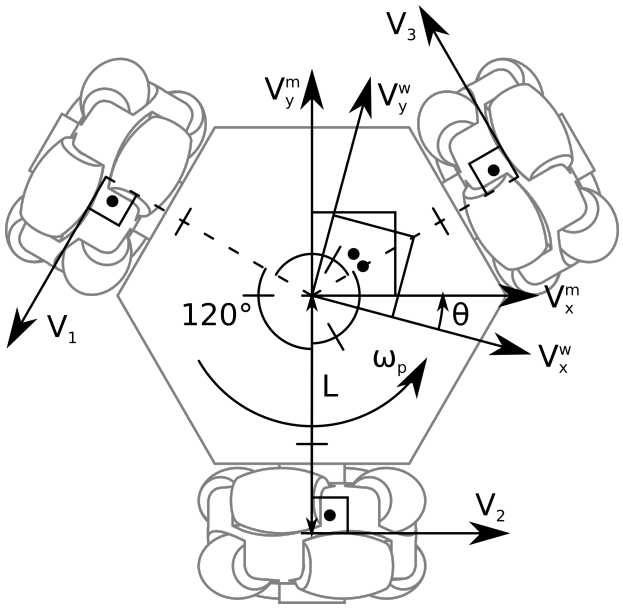


Figura 4: Geometria da plataforma omnidirecional.
Fonte: autor.

As equações 2.4 realizam a cinemática inversa em relação ao sistema de coordenadas da plataforma. Para realizar a cinemática inversa em relação ao sistema de coordenadas do mundo, primeiro as equações 2.3 devem ser usadas, que convertem as velocidades do sistema de coordenadas do mundo para o da plataforma. Como a plataforma móvel rotaciona da mesma maneira em qualquer sistema de coordenadas,

ω_p não precisa ser convertido. Para a cinemática direta em relação ao sistema de coordenadas da plataforma, basta isolar as variáveis de velocidade de translação e rotação das equações 2.4, resultando nas equações 2.1. Para realizar a cinemática direta em relação ao sistema de coordenadas do mundo, basta isolar as variáveis de velocidade de translação das equações 2.3, o que resulta nas equações 2.2, e aplicar essas equações depois da cinemática direta. Foi decidido utilizar as conversões ao invés das equações das cinemáticas em relação ao sistema de coordenadas do mundo porque as equações apresentadas aqui são consideravelmente mais compactas.

$$\begin{aligned} V_x^m &= \frac{2V_2 - V_1 - V_3}{3} \\ V_y^m &= \frac{\sqrt{3}V_3 - \sqrt{3}V_1}{3} \\ \omega_p &= \frac{V_1 + V_2 + V_3}{3L} \end{aligned} \tag{2.1}$$

$$\begin{aligned} V_x^w &= \cos(\theta)\, V_x^m - \sin(\theta)\, V_y^m \\ V_y^w &= \sin(\theta)\, V_x^m + \cos(\theta)\, V_y^m \end{aligned} \tag{2.2}$$

$$\begin{aligned} V_x^m &= \cos(\theta)\, V_x^w + \sin(\theta)\, V_y^w \\ V_y^m &= -\sin(\theta)\, V_x^w + \cos(\theta)\, V_y^w \end{aligned} \tag{2.3}$$

$$\begin{aligned} V_1 &= -\frac{V_x^m}{2} - \frac{\sqrt{3}V_y^m}{2} + L\omega_p \\ V_2 &= V_x^m + L\omega_p \\ V_3 &= -\frac{V_x^m}{2} + \frac{\sqrt{3}V_y^m}{2} + L\omega_p \end{aligned} \tag{2.4}$$

2.4 Curva de Bézier

Esta seção descreve a curva de Bézier (figura 5), que será usada como mais uma maneira de informar à plataforma trajetórias (e velocidades) desejadas. De acordo com ROGERS; ADAMS (1990), a curva de Bézier é uma curva no plano gerada através de pontos de controle. A equação 2.5 descreve a curva de Bézier, onde t é a variável de

tempo normalizada, $P\,(t)$ é o ponto no plano em determinado tempo, n é a quantidade de pontos de controle menos 1, $\binom{n}{i}$ é o coeficiente binomial (GALLIER, 1999) (ver equação 2.6) e B_i são os pontos de controle, com o primeiro em B_0.

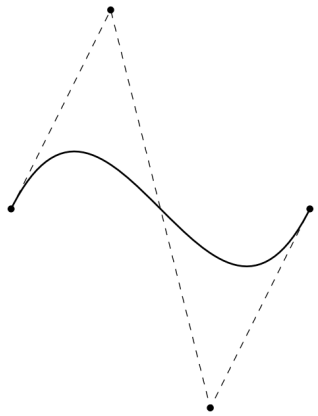


Figura 5: Curva de Bézier com 4 pontos de controle.
Fonte: (GALLIER, 1999).

$$P\,(t) = \sum_{i=0}^{n} \binom{n}{i} t^i\,(1-t)^{n-1}\,B_i, \quad 0 \leq t \leq 1 \tag{2.5}$$

$$\binom{n}{i} = \frac{n!}{i!\,(n-i)!} \tag{2.6}$$

2.5 Motor de corrente contínua

Os componentes que compõem a plataforma física serão apresentados nas próximas seções, começando pelo motor de corrente contínua, que serão um para cada eixo. De acordo com KURFESS (2005),

> Motores de escovas de corrente contínua (CC) convertem corrente elétrica aplicada em movimento mecânico. Eles são um tipo de servomotor, e o torque de saída é diretamente proporcional à corrente aplicada. Comparados com motores de passo, motores CC requerem um sinal de retroalimentação para operação estável. Eles devem ser usados em malha fechada, e isso pode fazer o sistema mais complexo do que um usando motores de passo. Eles também usam escovas condutivas para comutação mecânica ao invés de elétrica e portanto podem ter custos de manutenção mais altos devido ao desgaste. Porém, motores CC tem movimento suave e torque de pico mais alto e podem ser usados em velocidades mais altas do que motores de passo.

O exterior fixo dos motores CC é chamado de estator e pode incorporar imãs

permanentes ou enrolamentos, enquanto que o eixo central rotativo é chamada de rotor e incorpora enrolamentos. Todos os motores CC incluem escovas condutivas que fazem contato deslizante com um comutador na medida em que o rotor gira. O comutador é anexado ao eixo do rotor e os enrolamentos do rotor são conectados às seções individuais do comutador. A quantidade de seções do comutador é a mesma que a quantidade de polos do rotor (KURFESS, 2005).

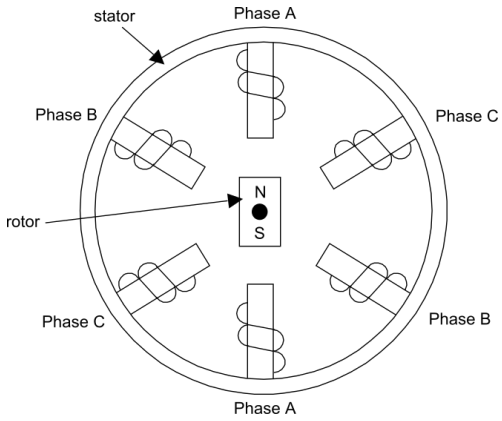


Figura 6: Exemplo de motor CC de dois polos.
Fonte: (KURFESS, 2005).

A figura 6 mostra um exemplo de motor CC com dois polos no rotor. Quando tensão elétrica é aplicada às escovas, a corrente flui das escovas para ao comutador e, em seguida, para os enrolamentos. Isso cria um campo magnético no rotor. A interação entre esse campo e o do estator faz o rotor girar até estar alinhado com o campo magnético do estator. Assim que o rotor estiver alinhado, as escovas conectam com a próxima seção do comutador e o campo magnético do rotor é invertido, reiniciando o ciclo. Motores CC tipicamente incluem múltiplos polos no rotor para suavizar o movimento e aumentar o torque (KURFESS, 2005).

2.6 *Encoder*

Esta seção apresenta os *encoders*, que serão utilizados um em cada eixo para prover odometria (mais na seção 2.7). BORENSTEIN; EVERETT; FENG (1996) escreve sobre os *encoders* óticos:

> [...] Os dispositivos correspondentes de hoje basicamente incorporam uma versão miniaturizada do sensor de proximidade de feixe interrompido. Um feixe de luz focado apontado para um fotodetector corres-

> pondente é periodicamente interrompido por um padrão opaco/transparente codificado em um disco intermediário rotativo anexado ao eixo de interesse. O disco rotativo pode assumir a forma de cromado em vidro, metal gravado, ou *photoplast* como Mylar. Relativo aos resolvers de corrente alternada mais complexos, o esquema de codificação direto e saída digital inerente do *encoder* ótico resulta em um pacote confiável de baixo custo com boa imunidade a ruído.
>
> Há dois tipos básicos de *encoders*: incremental e absoluto. A versão incremental mede velocidade rotacional e pode inferir posição relativa, enquanto que modelos absolutos medem diretamente a posição angular e inferem velocidade. Se a informação de posição não volátil não for uma consideração, *encoders* incrementais são mais fáceis de interfacear e provêm resolução equivalente a um custo muito mais baixo do que *encoders* óticos absolutos.

O tipo mais simples de *encoder* incremental é o *encoder* tacômetro de um canal, que utiliza um disco dividido angularmente em seções que permitem e bloqueiam a luz. Quando a luz passa, ela gera um pulso. Os *encoders* geram um número determinado de pulsos por cada volta do eixo. Modelos com mais pulsos implicam em maior resolução e custo. Esses dispositivos relativamente baratos são bem adequados para sensores de velocidade em sistemas de controle de média e alta velocidade, mas apresentam problemas de ruído e estabilidade em velocidades extremamente pequenas por causa de erros de quantização. *Encoders* tacômetros não são capazes de detectar a direção de rotação e não podem ser utilizados como sensor de posição. Para superar esses problemas, os *encoders* incrementais de fase-quadratura possuem mais um canal, defasado do primeiro em 90°, o que também causa um aumento na resolução (BORENSTEIN; EVERETT; FENG, 1996). A figura 7 ilustra a operação de um *encoder* incremental de fase-quadratura.

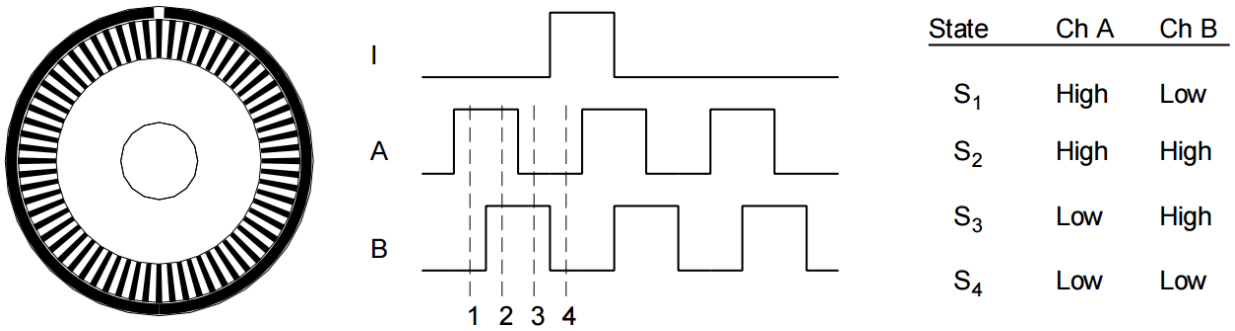


Figura 7: *Encoder* incremental de fase-quadratura.
Fonte: (BORENSTEIN; EVERETT; FENG, 1996).

2.7 Odometria

Esta seção introduz a odometria, que será utilizada no trabalho para possibilitar alguns dos comandos de movimentação. De acordo com BORENSTEIN; EVERETT; FENG (1996),

> *Dead reckoning* (derivado de *deduced reckoning* dos dias de navegação à vela), é um procedimento matemático simples para determinar a localização atual de uma embarcação pelo avanço de uma posição anterior através de informações de curso e velocidade conhecidos sobre um dado período de tempo. A vasta maioria de sistemas robóticos terrestres em uso hoje dependem de *dead reckoning* para formar a base de seus sistemas de navegação e, como seus equivalentes náuticos, periodicamente anulam erros acumulados com ajustes recorrentes a partir de dicas navegacionais sortidas.
>
> A implementação mais simplista de *dead reckoning* é as vezes chamada de odometria; o termo implica que o deslocamento de veículo ao longo do caminho do movimento é diretamente derivado de algum "odômetro"a bordo. Maneiras comuns de implementação de odometria envolve *encoders* óticos diretamente acoplados à armação do motor ou eixos das rodas.

Assim, para se saber a posição x e y da plataforma no instante K atual, basta medir e calcular a velocidade em x e y (V_x e V_y) a cada tempo de amostragem T, saber x e y no instante anterior $K-1$, e utilizar as fórmulas 2.7 (GONçALVES; LIMA; COSTA, 2008). O mesmo vale para a orientação da plataforma.

$$\begin{aligned} x(K) &= x(K-1) + V_x T \\ y(K) &= y(K-1) + V_y T \end{aligned} \tag{2.7}$$

2.8 Raspberry Pi

Esta seção apresenta o Raspberry Pi, que é o processador (na verdade, computador) que será utilizado para fazer a interface entre a comunicação de controle e a movimentação da plataforma, através da execução do programa que recebe os comandos de comunicação através de uma conexão cabeada, lê os dados dos *encoders*, realiza os processamentos e cálculos necessários e envia os sinais de controle para os motores. De acordo com RASPBERRY PI FOUNDATION (2016a),

> O Raspberry Pi é um computador do tamanho de um cartão de crédito

> que conecta na sua TV e um teclado. É um pequeno computador capaz que pode ser usado em projetos de eletrônica e para muitas coisas que o seu computador de mesa faz, como planilhas, processamento de texto, navegar na *internet* e jogos. Ele também reproduz vídeo em alta definição. Nós queremos ver ele ser usado por crianças ao redor do mundo para aprender programação.

A versão 1 Model B possui as seguintes características (RASPBERRY PI WIKI, 2016):

- CPU ARM A11 700 MHz
- 256 ou 512 MiB
- 2 portas USB
- 26 pinos de entrada/saída genéricos
- Porta Full HDMI
- Porta Ethernet
- Conector RCA de vídeo composto
- Conector TRS 3.5 mm de áudio
- Interface de câmera (CSI)
- Interface de exibição (DSI)
- *Slot* para cartão Micro SD
- Núcleo de gráficos 3D VideoCore IV

A figura 8 mostra um dos modelos do Rasbperry Pi. Pelo seu processador, ele é capaz de executar todas as distribuições GNU/Linux para ARM, assim como Windows 10 (RASPBERRY PI FOUNDATION, 2016b).

2.9 Controle

Esta seção descreve como será feito o controle da plataforma. Primeiro, alguns termos devem ser introduzidos. A *variável controlada* ou *variável do processo* é a variável

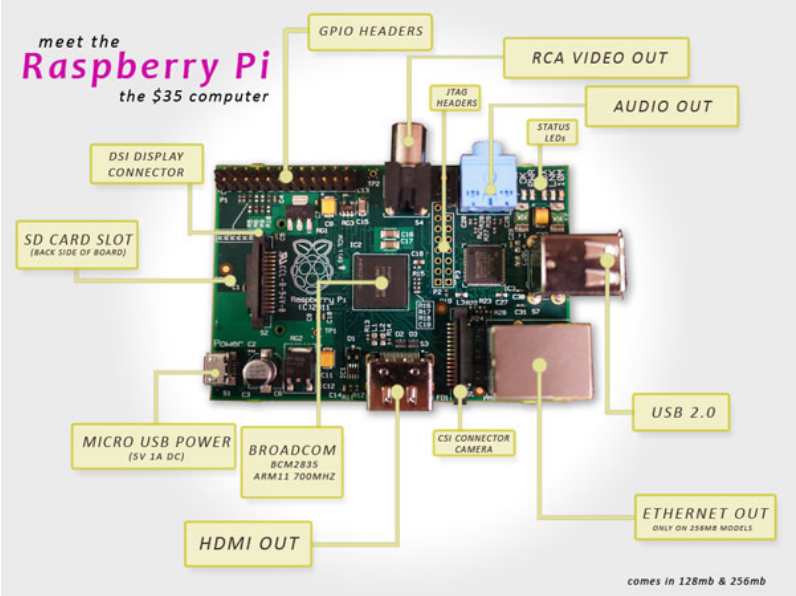


Figura 8: Um modelo do Raspberry Pi.
Fonte: (WORDMAN, 2012).

a ser mantida em um valor desejado. O *ponto de ajuste* é o valor desejado da variável controlada. A *variável manipulada* é a variável usada para manter a variável controlada no ponto de ajuste. Variáveis que impedem que a variável controlada se mantenha no ponto de ajuste são chamadas de *distúrbios* ou *perturbações*. O *erro* é a diferença entre a variável controlada e o ponto de ajuste (SMITH; CORRIPIO, 1997).

Assim, este trabalho usará um *controle em malha fechada*, onde o controlador está conectado ao processo comparando a variável controlada e o ponto de ajuste e determinando a variável manipulada de acordo. O controlador será do tipo PID, cuja saída é a soma de três termos: a entrada multiplicada por uma constante, a integral da entrada multiplicada por uma constante e a derivada da entrada multiplicada por uma constante. Construir um controlador PID significa determinar essas três constantes (SMITH; CORRIPIO, 1997).

De acordo com ANG; CHONG; LI (2005), o controle PID "[...] oferece a solução mais simples contudo a mais eficiente para muitos problemas de controle do mundo real"e "[...] nenhum outro controlador se equipara à simplicidade, funcionalidade clara, aplicabilidade e facilidade de uso oferecido pelo controlador PID".

2.10 Gazebo

Antes de construir o protótipo físico, o mesmo será simulado para validação, de forma a evitar perda de tempo e materiais na construção e testes da plataforma física e da sua programação. Esta seção introduz a ferramenta que será utilizada para isso, o simulador Gazebo.

O Gazebo é um simulador livre e de código aberto de alta fidelidade com a capacidade de simular populações de robôs em ambientes complexos internos e externos. Trabalha com um dentre quatro motores de física, possui gráficos de boa qualidade e também pode ser controlado via linha de comando. Suporta a simulação de sensores, opcionalmente adicionando ruído na leitura, como telêmetro a laser, câmeras 2D/3D, *scanners* de luz estruturada, sensores de contato, de força e torque, etc. As simulações podem ser executadas e controladas remotamente através de um protocolo específico e podem ser executadas em três serviços de armazenamento em nuvem (OPEN SOURCE ROBOTICS FOUNDATION, 2014).

Ele vem com uma biblioteca *online* de robôs e objetos, que são descritos por um arquivo de texto em um formato chamado de SDF, pelo qual é possível descrever novos robôs e objetos. A interface gráfica também possui um modo que auxilia a construção de novos modelos, permitindo a inserção de elos, juntas, sensores, etc, e esses modelos podem ser contribuídos para a biblioteca. Robôs, objetos, o mundo simulado e o próprio Gazebo podem ser controlados através de *plugins*, que são escritos em C++ e compilados com os compiladores padrões, através de um CMakeLists.txt e Makefile customizado. Também pode ser integrado com ROS (OPEN SOURCE ROBOTICS FOUNDATION, 2014).

A simulação executada pelo Gazebo simula automaticamente vários fenômenos físicos, como gravidade, colisão, atrito e escorregamento, etc. A simulação é capaz de apresentar resultados realistas na medida em que os parâmetros de cada entidade são atribuídos, como massa, momento de inércia, etc.

2.11 SOLIDWORKS

Esta seção introduz o SOLIDWORKS, ferramenta que será utilizada para gerar os modelos 3D e calcular os parâmetros de massa, volume e momento de inércia, requeridos pela ferramenta Gazebo (introduzida na seção 2.10). O programa SOLIDWORKS é uma ferramenta de CAD (projeto assistido por computador), capaz de modelar peças e montagens sofisticadas. Possui ferramentas para aplicações em outras áreas, como desenho, análise de projeto, estimativa de custo, renderização, animação, e gerenciamento de arquivos (DASSAULT SYSTEMES, 2016).

O SOLIDWORKS também é capaz de importar geometrias geradas por várias ferramentas diferentes, como arquivos em formato STEP. É provável que algumas das peças adquiridas para montar o protótipo de validação tenham a sua geometria disponibilizada desta maneira.

2.12 *Software* livre e código fonte aberto

Como o trabalho será disponibilizado para a comunidade de forma livre, é relevante mencionar dois termos que representam essa comunidade: *software* livre e código fonte aberto. De acordo com STALLMAN (2016), "Quando nós nos referimos a software livre, nós queremos dizer que ele respeita as liberdades básicas dos usuários: a liberdade de rodá-lo, estudá-lo e modificá-lo, e de redistribuir cópias com ou sem mudanças". E, de acordo com OPEN SOURCE INITIATIVE (2016), "Geralmente, *software* de Código Fonte Aberto é *software* que pode ser livremente acessado, usado, modificado e compartilhado (de forma modificada ou não) por qualquer um". STALLMAN (2016) explica a diferença entre os dois termos que aparentemente representam os mesmos conceitos:

> Os dois termos descrevem quase a mesma categoria de software, mas eles representam visões baseadas em valores fundamentalmente diferentes. Código fonte aberto é uma metodologia de desenvolvimento; software livre é um movimento social. Para o movimento do software livre, software livre é uma imperativa ética, respeito essencial pelas liberdades do usuário. Em contraste, a filosofia do código fonte aberto considera questões em termos de como fazer um software 'melhor'–num sentido prático somente.

2.13 Direitos autorais

Existem várias maneiras de disponibilizar o trabalho para a comunidade de forma livre. Como os direitos autorais são tratados é uma das coisas que diferem entre essas maneiras. Essa seção introduz algumas delas.

De acordo com CREATIVE COMMONS (2016), "Direitos autorais concedem direitos exclusivos a criadores de obras originais de autoria. Leis nacionais geralmente estendem a proteção a tais obras automaticamente assim que fixadas em um meio tangível, proibindo a cópia sem a permissão do possuidor dos direitos, entre outras coisas". KENNEDY (2005) escreve sobre um uso alternativo dos direitos autorais (neste caso, para código fonte):

> *Software* de código fonte aberto tem recentemente capturado a atenção pública ambos por causa da atratividade e da crescente fatia de mercado de programas desenvolvidos sobre o modelo de código fonte aberto e por causa da sua abordagem única para licenciamento de *software* e programação baseada em comunidade. [...] O movimento do código fonte aberto reflete a intenção dos seus fundadores de virar de ponta cabeça noções tradicionais de direitos autorais, licenciamento de *software*, distribuição, desenvolvimento e até propriedade, até o ponto de criar o termo "esquerdo de cópia"para descrever a abordagem alternativa a essas questões.

O criador de um trabalho também pode optar por renunciar aos seus direitos de cópia, fazendo com que o trabalho entre em domínio público. BRANCO (2011) escreve sobre ele:

> Em uma concepção introdutória do instituto, podemos dizer que o domínio público representa o fim dos direitos patrimoniais do autor, normalmente em razão de ter sido atingido o prazo previsto em lei. Em outras palavras, as obras podem ser utilizadas por toda a sociedade independentemente de licença por parte de seus autores originais, seus sucessores ou outros titulares de direitos autorais. Isso inclui o uso comercial e não há qualquer distinção legal quanto ao uso que se pretenda dar às obras até então protegidas.

3 TRABALHOS RELACIONADOS

Este capítulo compara este trabalho com alguns trabalhos de variada semelhança. Foram considerados critérios de semelhança: movimento omnidirecional; facilidade de expansão; interface de controle; disponibilidade de códigos fonte/projetos de PCB/desenhos técnicos para reprodução; e instruções para reprodução. Não foi encontrado um trabalho que se assemelha a este em todos esses critérios.

Dois trabalhos da academia com um bom nível de semelhança serão discutidos. Dois trabalhos encontrados pela *internet* também serão discutidos, a fim de comparação. A tabela 1 resume as características de cada trabalho de acordo com os critérios citados. Onde a tabela menciona "parcial", significa que alguns dados estão disponíveis, e provavelmente seria possível extrapolar o resto dos dados.

Tabela 1: Resumo da comparação dos trabalhos relacionados

Trabalho	Geometria	Controle	*Software*	Eletrônica	Mecânica	Instruções
1	3 omni	serial	citado; indisponível	disponível	parcial	disponível
2	4 omni	manual *wireless*	disponível	parcial	citado	disponível
3	3 omni	programa; *wireless*	citado; indisponível	disponível	disponível	disponível
4	4 Mecanum	manual *wireless*	citado; indisponível	disponível	disponível	disponível
5	3 omni	serial	disponível	disponível	disponível	disponível

Fonte: Autor

1. Monografia de INUZUKA; SILVA LIMA (2005) para a obtenção do título de Engenheiro de Mecatrônica. O controle é o mesmo que é proposto por este trabalho (mensagens com comandos), com menos mensagens disponíveis. O código está

disponível em um CD em anexo. Porém, esse CD não está disponível na *internet*, ao contrário do texto da monografia. Pode ser facilmente utilizado em um projeto maior.

2. Tese de BEMIS (2009) para a obtenção do título de Mestre em Ciência Aplicada em Engenharia Mecânica. O controle é feito através de um controle remoto, que é conectado a um computador de mesa executando dois nós ROS, que envia comandos a um computador portátil montado no robô e que também executa um nó ROS, que envia comandos serialmente ao microcontrolador. A movimentação pode ser feita através do computador de mesa, que emula a operação do controle remoto. A posição e orientação do robô são controladas pelos três eixos do controle remoto. Não possui a mesma correspondência de mensagens deste trabalho, mas pode ser facilmente expandido pela troca do nó ROS. A parte mecânica é baseada em outro trabalho que não pôde ser encontrado, portanto não se sabe quanto de documentação há disponível. Pode ser facilmente utilizado em um projeto maior.

3. Projeto de KNOLL (2005), professor doutor da TUM. Executa uma versão do Debian GNU/Linux. Controle é feito através da programação de aplicativos, que podem controlar a velocidade das rodas ou determinar velocidades desejadas de translação e rotação. Possui interface sem fio. Pode ser facilmente utilizado em um projeto maior. *Links* para o código quebraram durante a realização deste trabalho.

4. Projeto de DIEGEL (2011). Plataforma feita sem um controlador específico, mas um é usado de exemplo. Controle é feito a partir de um computador com conexão sem fio ou telefone celular, determinando velocidades de translação ou de rotação. Parte mecânica pode ser construída com impressora 3D, incluindo rodas. Difícil de ser utilizado em um projeto maior. *Links* para a página *web* quebraram durante a realização deste trabalho.

5. Este trabalho

A nível de controle, o autor acredita que qualquer um destes trabalhos possa ser facilmente adaptado para atingir os mesmos objetivos que este trabalho. O único código que pôde ser analisado foi o de BEMIS (2009), e que foi considerado bem comentado e modularizado. Os trabalhos de INUZUKA; SILVA LIMA (2005) e de BEMIS (2009)

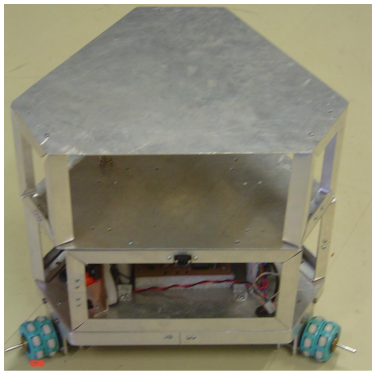


Figura 9: Trabalho de INUZUKA; SILVA LIMA (2005).
Fonte: INUZUKA; SILVA LIMA (2005).

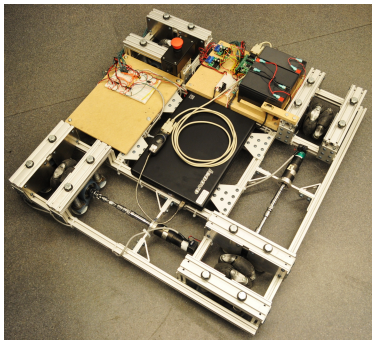


Figura 10: Trabalho de BEMIS (2009).
Fonte: BEMIS (2009).

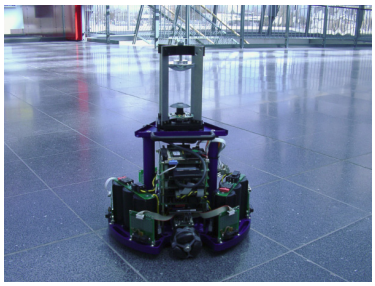


Figura 11: Trabalho de KNOLL (2005).
Fonte: KNOLL (2005).

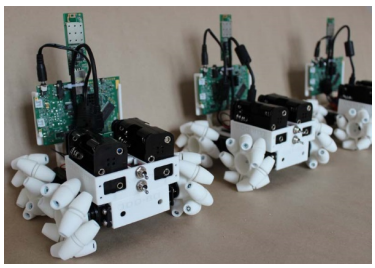


Figura 12: Trabalho de DIEGEL (2011).
Fonte: DIEGEL (2011).

poderiam ser facilmente utilizados em robôs mais complexos; o trabalho de DIEGEL (2011) precisaria de mais ajustes por ter um tamanho mínimo necessário e o trabalho de KNOLL (2005) incorpora várias funções, portanto precisaria de vários ajustes para atender a outros objetivos.

Foi considerado que os trabalho de INUZUKA; SILVA LIMA (2005) e BEMIS (2009) foram os que mais se aproximaram deste trabalho no quesito de atender os mesmos objetivos, e o trabalho de KNOLL (2005) no quesito de disponibilidade de informações, pelo menos até o momento em que os *links* ainda funcionavam. Portanto, este trabalho se destaca pela garantia da disponibilidade e livre uso das informações necessárias para a sua replicação, além da proposta genérica de oferecer mobilidade para projetos de robôs e da oferta de vários comandos de movimentação.

4 TRABALHO DESENVOLVIDO

Este capítulo apresenta o resultado do trabalho. Foi desenvolvido um modelo fictício de plataforma omnidirecional para ser usado com o simulador Gazebo (figura 13). As rodas foram modeladas a partir de modelos de rodas omnidirecionais reais, de 38,1 mm de diâmetro e 8 roletes emborrachados (ROBOTS-R-US, 2011). Os roletes são simulados independentemente das rodas no Gazebo. Por motivos de desempenho, detalhes menores da roda não foram modelados, como os parafusos e os eixos dos roletes.

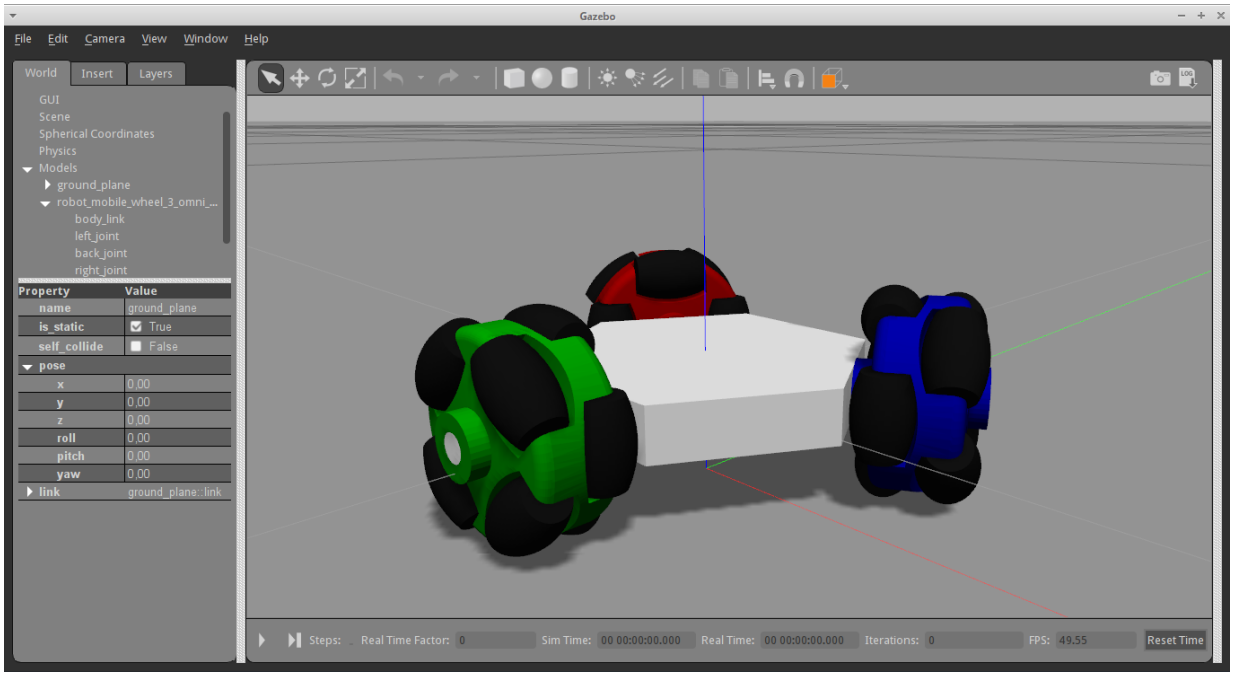


Figura 13: Interface do Gazebo com o modelo fictício desenvolvido.
Fonte: Autor.

O centro das rodas fica distante do centro do corpo por 0,04 m. O corpo do robô é somente um bloco de alumínio maciço, sem motores ou qualquer outro componente,

dado que a plataforma simulada só serve para a validação do programa de controle. Como a única alteração necessária no programa é o valor de L se houver alteração na geometria da plataforma (e o valor de r se forem usadas outras rodas), o modelo pode ser facilmente alterado com as dimensões e massa de um modelo proposto para tornar a sua simulação realista.

Foi desenvolvido um programa para controlar o modelo simulado. Ele foi escrito em C++, sendo compilado como um *plugin* do Gazebo e adicionado ao modelo, o que já vem pronto. O programa usa o modelo cinemático da plataforma omnidirecional para controlar as juntas que representam os motores, e os valores de velocidade dessas juntas são usados no lugar dos *encoders* para prover a odometria. É usado um controlador PID entre a cinemática e o controle dos motores para gerenciar a velocidade de movimento do robô, com $P = 5$ e $I = 0,000\,5$.

Foram desenvolvidos vários comandos que geram movimentação da plataforma. Esses comandos podem ser usados imediatamente no programa sem nenhuma configuração prévia, tornando possível o uso da plataforma como base pronta para um robô mais complexo. Todos os comandos podem ser executados em dois modos, onde em cada um é usado um referencial (sistema de coordenadas) diferente para definir as posições/ângulos/velocidades necessárias: ou o referencial da plataforma móvel, ou o referencial fixo do mundo.

4.1 Comandos de movimento

Esta seção descreve os comandos de movimento que foram desenvolvidos para permitir que a plataforma descreva várias trajetórias diferentes.

4.1.1 Comandos para movimento direto

Permite ao usuário controlar diretamente as velocidades das rodas e do robô. A tabela 2 explica esses comandos.

Tabela 2: Comandos para movimento direto

Comando	Descrição
`fireMovementDirectWheel`	Faz com que cada roda gire de acordo com a velocidade inserida.
`fireMovementDirectMobile`	Faz com que a plataforma se mova de acordo com as velocidades inseridas, com relação ao referencial da plataforma móvel.
`fireMovementDirectWorld`	Faz com que a plataforma se mova de acordo com as velocidades inseridas, com relação ao referencial fixo do mundo.
`fireMovementDirectHybrid`	Traduz as velocidades inseridas do referencial da plataforma móvel para o referencial do mundo e faz com que as plataforma se mova com essas velocidades em relação ao referencial do mundo.

Fonte: Autor

4.1.2 Movimento de pose

Os métodos `fireMovementAbsoluteM|W` e `fireMovementRelativeM|W` controlam a pose final da plataforma (posição e ângulo). Usam o controlador PID para garantir que o movimento cesse quando a plataforma estiver aproximadamente no destino. É possível configurar a velocidade máxima da plataforma e a distância (em metros e radianos) de tolerância para considerar se a plataforma chegou no destino. Inicialmente, é considerado que a plataforma chegou ao destino se a sua distância até o mesmo for menor ou igual a 1 mm e a diferença de ângulo for menor ou igual a $\frac{\pi}{30}$ rad = 6°. A pose pode ser fixa ou relativa à pose atual. Por exemplo, a plataforma pode se mover para $x = 1$ m ou se mover 1 m em x.

Como mencionado antes, pode usar qualquer um dos referenciais. Uma menção deve ser feita sobre os métodos `fireMovementAbsoluteM` e `fireMovementAbsoluteMRaw`. Se só fosse considerado que a plataforma se movesse para uma pose no seu referencial, seria fácil garantir a pose da plataforma no seu referencial, mas extremamente complexo de fazê-lo no referencial do mundo, dado que a velocidade de transição do ângulo entre os valores inicial e final influenciam na posição da plataforma em relação ao mundo. Por esse motivo, o método `fireMovementAbsoluteM` faz uma transformação da posição desejada no referencial da plataforma para o referencial do mundo, e em seguida executa o movimento em relação ao referencial do mundo. O método `fireMovementAbsoluteMRaw` faz o movimento sem a transformação e diretamente no referencial da plataforma. Não

foi possível definir uma utilidade para esse movimento, mas como ele é extensão natural do `fireMovementAbsoluteW`, ele foi mantido.

4.1.3 Movimento curva de Bézier

Os métodos `fireMovementBezierM|W` fazem a plataforma descrever uma trajetória de acordo com uma curva de Bézier definida através de pontos de controle. Os pontos de controle podem ser tratados como posições absolutas em qualquer referencial ou como posições relativas. No último caso, o primeiro ponto é transladado para a posição atual da plataforma, e os demais pontos são transladados (e rotacionados, no caso do referencial da plataforma) de acordo a manter o desenho original da curva.

O ângulo da plataforma também pode ser controlado por uma curva de Bézier. Foi percebido que o algoritmo para a geração da curva de Bézier trata cada dimensão independentemente. Assim, foi desenvolvido um algoritmo unidimensional para controlar o ângulo da plataforma, que recebe como entrada uma lista de valores ao invés de uma lista de pontos. Para a visualização do resultado, pode-se usar qualquer *software* que desenhe curvas de Bézier e, por exemplo, considerar somente os valores de x, que são resultado da lista formada pelos valores de x dos pontos de controle. O ângulo da plataforma também pode ser relativo. Nesse caso, o primeiro valor de controle é considerado o ângulo inicial da plataforma. Uma lista deve ser passada por parâmetro com pelo menos um valor. Para nenhuma rotação, pode ser usado o valor 0 com movimento relativo.

Como o algoritmo para geração da curva de Bézier gera cada ponto em sequência, do inicial ao final, a plataforma segue o último ponto gerado até chegar ao destino. Como o algoritmo deve ser configurado com um valor que define o intervalo de iteração, que influencia na distância entre cada ponto gerado, e como cada iteração do algoritmo da curva de Bézier é executada em uma iteração do programa da plataforma, que geralmente ocorre em um período de tempo conhecido ou medido, o intervalo de iteração do algoritmo da curva de Bézier define o tempo necessário para geração da curva. Como a plataforma se movimenta em velocidade máxima para alcançar o ponto sendo gerado, se o tempo de iteração do algoritmo da curva de Bézier for muito curto e a curva for muito

longa, a plataforma não será capaz de acompanhar o ponto sendo gerado e, portanto, não será capaz de percorrer a trajetória completamente. Assim, cuidado deve ser usado para escolher o tempo de iteração do algoritmo da curva de Bézier.

5 RESULTADOS

5.1 Cinemática

Este capítulo demonstra os testes e seus resultados que foram usados para a validação do trabalho. Os testes foram iniciados pela parte mais fundamental da plataforma, que é a cinemática. Foram escolhidas combinações de velocidades para as rodas e foram calculadas as velocidades lineares e rotacional usando as fórmulas da cinemática direta em relação ao referencial da plataforma. Algumas dessas combinações podem ser comprovadas visualmente, como é mostrado na figura 14: como a velocidade de cada roda é um vetor, desde que não haja rotação, a soma dos vetores mostra qual será o vetor velocidade da plataforma. Decompondo esse vetor, se obtém a velocidades em x e y.

A tabela 3 mostra os valores de entrada da cinemática direta (velocidade de cada roda, em radianos por segundo) em relação ao referencial da plataforma e os valores de saída (velocidades da plataforma) (valores arredondados para três casas decimais e em metros por segundo). Na figura, os vetores não estão desenhados em escala, mas a informação do sinal dos valores é necessária para a comprovação do teste. Círculos representam velocidade zero.

Tabela 3: Testes de cálculos da cinemática direta em relação ao referencial da plataforma

V_3	V_2	V_1	V_y	V_x
$-0,5$	0	$0,5$	$-0,577$	0
$-0,5$	$0,5$	0	$-0,289$	$0,5$
0	$0,5$	$-0,5$	$0,289$	$0,5$
0	$-0,5$	$0,5$	$-0,289$	$-0,5$
$0,5$	$-0,5$	0	$0,289$	$-0,5$
$0,5$	0	$-0,5$	$0,577$	0

Fonte: Autor

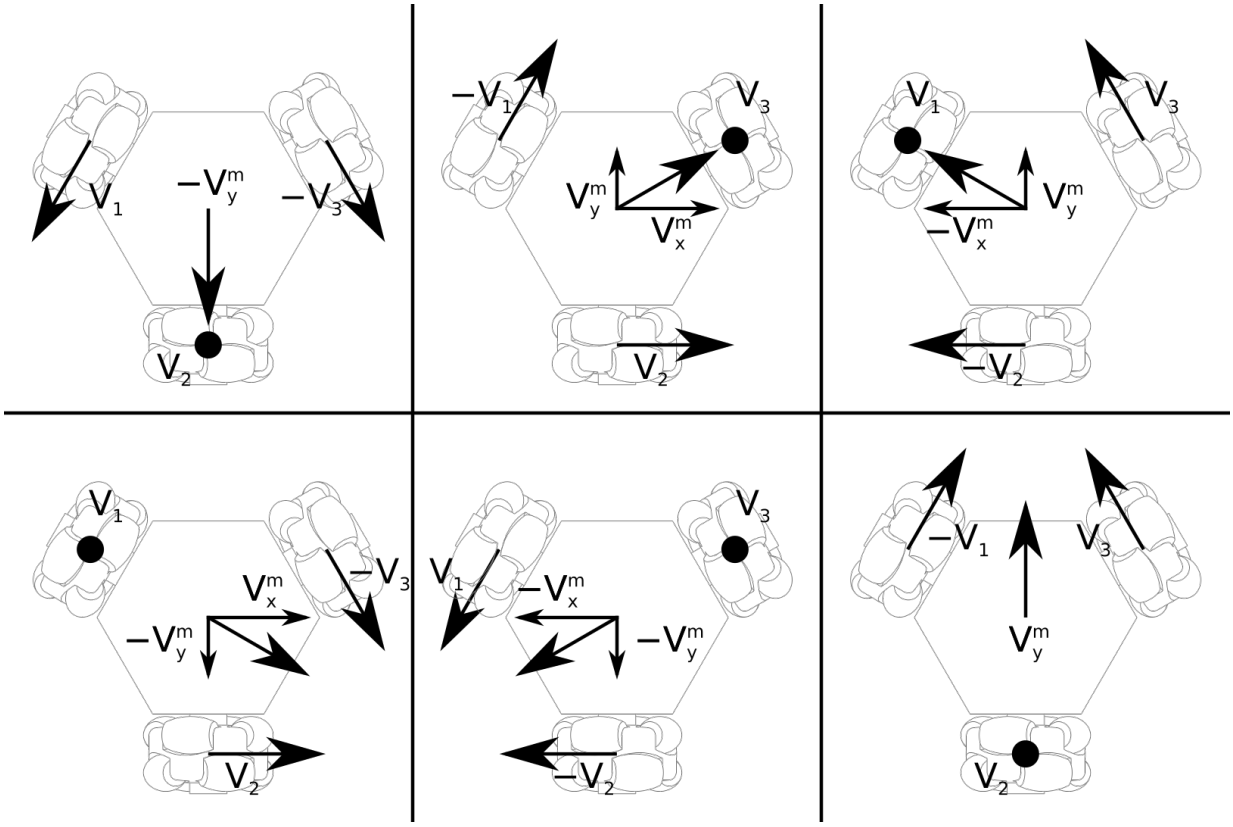


Figura 14: Combinações de valores para testes da cinemática.
Fonte: Autor.

É possível perceber que as equações da cinemática condizem com um teste rápido manual. Em seguida, esses testes foram repetidos no modelo simulado, para verificar se a plataforma se move como esperado e se as suas velocidades medidas condizem com as esperadas pelas equações. As figuras 15 mostram as velocidades da plataforma, medidas pela cinemática, onde o eixo x representa a amostragem dos dados, que ocorreu durante 2 segundos, os traços vermelhos e verdes representam as velocidades lineares em x e y, respectivamente, cujos valores são representados em metros por segundo no eixo y da direita, e o traço azul representa a velocidade rotacional representada em radianos por segundo no eixo y da esquerda.

Esses valores foram obtidos através da leitura da velocidade rotacional das rodas, que não são iguais às velocidades desejadas pois há distúrbios como aceleração e atrito que desviam o valor real do valor desejado. Essas velocidades são multiplicada pelo raio das rodas para se obter as velocidades lineares, que são alimentadas na cinemática direta para se obter as velocidades da plataforma.

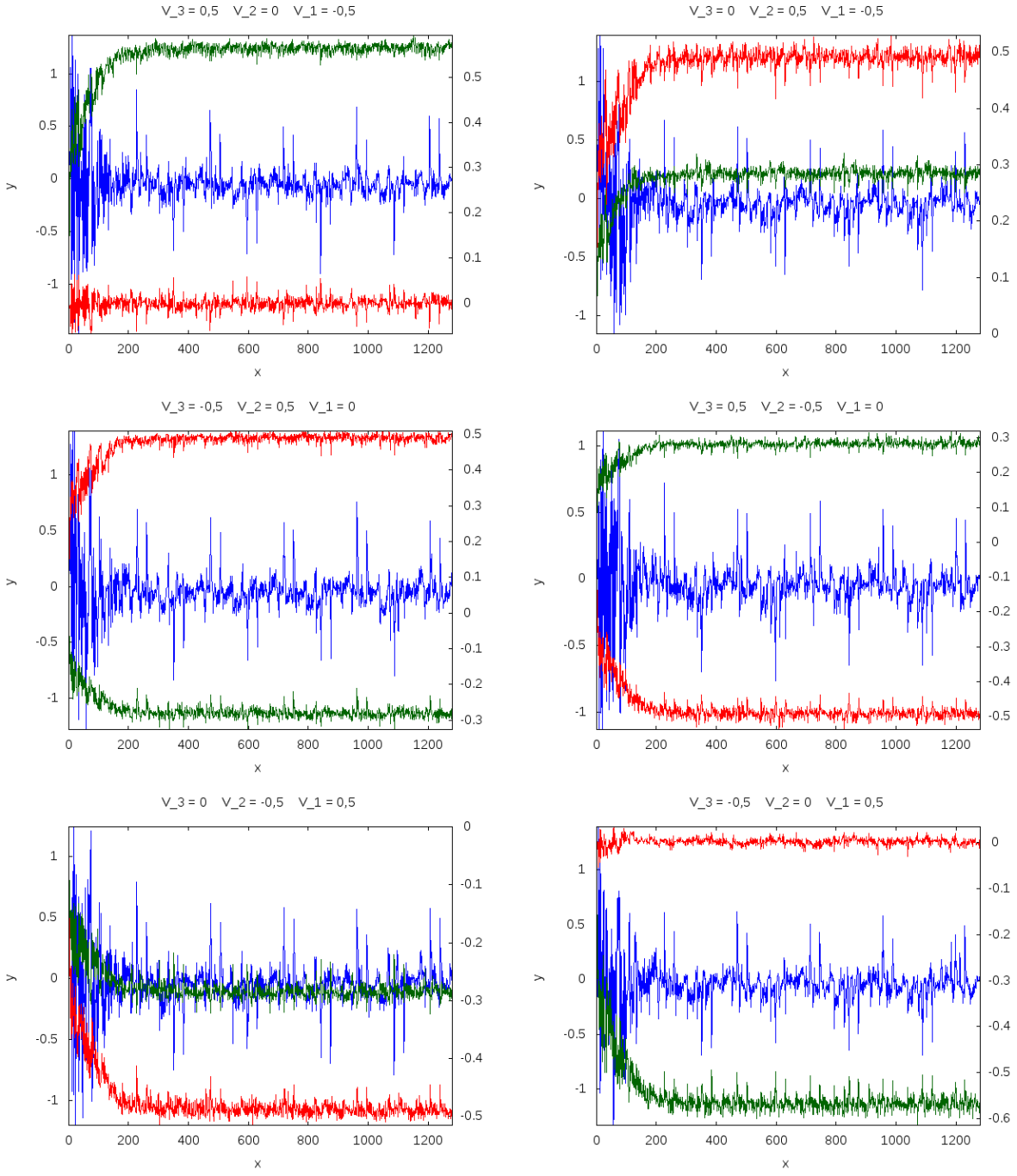


Figura 15: Medição de velocidades da plataforma para as combinações de velocidade das rodas.

Fonte: Autor.

Inicialmente é possível perceber que cada roda demora para acelerar até a velocidade desejada, exemplando o grau de realismo oferecido pela ferramenta de simulação. A velocidade se estabiliza após aproximadamente $0,6$ s (1281 amostras, portanto, aproximadamente a partir da amostra 385). A tabela 4 mostra a média dos valores das velocidades a partir desse ponto. É possível perceber que, apesar da variação constante das velocidades, elas se mantém em uma média bem próxima ao valor calculado/desejado.

Tabela 4: Comparação entre as velocidades da plataforma calculadas e médias das medidas para algumas combinações de velocidades lineares das rodas.

V_3	V_2	V_1	V_y		V_x	
			Calculado	Medido	Calculado	Medido
$-0,5$	0	$0,5$	$-0,577$	$-0,567$	0	$0,001$
$-0,5$	$0,5$	0	$-0,289$	$-0,289$	$0,5$	$0,490$
0	$0,5$	$-0,5$	$0,289$	$0,285$	$0,5$	$0,491$
0	$-0,5$	$0,5$	$-0,289$	$-0,285$	$-0,5$	$-0,488$
$0,5$	$-0,5$	0	$0,289$	$0,283$	$-0,5$	$-0,491$
$0,5$	0	$-0,5$	$0,577$	$0,565$	0	$-0,001$

Fonte: Autor

5.2 Odometria

Estando a cinemática funcionando como esperado, a próxima parte a ser testada é a odometria. Como é possível obter a posição real exata da plataforma através do simulador, basta que a plataforma se movimente para comparar a posição obtida pela odometria com a posição real. Foram gerados 10 combinações de valores aleatórios de velocidade para as rodas, com duração de 1 segundo cada, e as posições real e estimada pela odometria são apresentadas no gráfico da figura 16. Foram usados os valores da odometria em relação ao referencial do mundo, pois é nesse referencial que estão os valores lidos da simulação. O eixo x representa amostras. A escala y da esquerda corresponde à x e a da direita à y, ambas em metros.

É possível observar que a diferença entre os valores reais e medidos se altera quando há alterações de movimento, permanecendo constante com o movimento constante, portanto, pelo menos até 10 segundos, não há um aumento constante no erro da odometria como é esperado da mesma. O maior erro em x foi de $0,116\,558$ m e em y foi de $0,141\,205$ m. Considerando que o diâmetro da plataforma é de pouco mais de 10 cm,

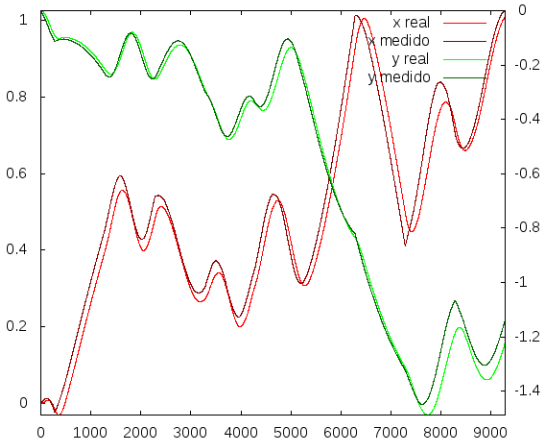


Figura 16: Comparação entre a posição da plataforma real e medida pela odometria após movimentos aleatórios.

Fonte: Autor.

que esse diâmetro é menor do que seria se fosse empregado para uma tarefa real por ser um modelo abstrato, e que a plataforma utiliza apenas um método de odometria, sendo que outros podem ser empregados para melhorar esse resultado, tanto absolutos como relativos, esse resultado foi considerado aceitável.

5.3 PID

A odometria permite que a plataforma "saiba"onde ela está, o que permite que ela se movimente para poses desejadas. Porém, para garantir o mínimo de desempenho nesses movimentos, é necessário um controle em malha fechada que seja alimentado pela distância entre a pose atual e a desejada da plataforma, chamada de *erro*. O resultado do controle seria o trio de velocidades da plataforma, que resultam na velocidade de cada roda através da cinemática. Além de trabalhar para reduzir o erro, o controle também seria capaz de lidar com perturbações, como aceleração e atrito, diminuindo ou eliminando o impacto destes no movimento. Esse controle foi fornecido pelo controlador PID que foi implementado.

O controlador PID foi implementado gradualmente. Para ajustar os seus parâmetros, foi definido um teste cujo objetivo é o de levar a plataforma da pose $(x, y, \theta) = (0, 0, 0)$ à pose $(x, y, \theta) = (0, 1, 0)$ no menor tempo possível. Durante esses testes, foi descoberto que a plataforma fica difícil de controlar acima de 1 m/s de translação, por-

tanto, foi implementado um limite de velocidade na plataforma. Para o protótipo real, o limite de velocidade seria o atingível pelos motores, provavelmente reduzido pela massa que esses motores carregariam.

Como o erro inicial nesse teste era de 1 m, e usando $P = 1$ e $I = 0$ no PID resultaria em 1 m/s inicialmente, esse valor foi usado para iniciar o ajuste. No gráfico respectivo da figura 17, é possível ver que a distância diminui em função do tempo, o que é esperado, e que a velocidade também diminui em função do tempo, o que também é esperado. O problema é que a velocidade diminui exponencialmente em função do tempo, dado que a velocidade é proporcional à distância. Na medida em que a velocidade diminui, a distância passa a diminuir menos. A 4 mm do objetivo, a velocidade é tão baixa que um erro do Gazebo faz com que as velocidades variem aleatoriamente entre valores bem baixos, fazendo com que a plataforma não se mova mais em direção ao alvo. Na realidade, é provável que a velocidade ficasse tão baixa a ponto da plataforma não se mover mais, nunca chegando ao objetivo.

A plataforma usa um parâmetro que especifica que a ela atingiu o alvo quando estiver a 1 mm do mesmo ou menos. Esse parâmetro poderia ser aumentado para fazer com que seja considerado que a plataforma tenha atingido o seu alvo. Porém, foi considerado inaceitável 7 s para atingir o destino, dado que, à velocidade máxima e desconsiderando aceleração, esse tempo poderia ser de 1 s (distância do teste sendo percorrida na velocidade limite). Portanto, P precisa ser melhorado. Em seguida, foi tentado com $P = 2$. No gráfico respectivo da figura 17, é possível perceber que, além do tempo ter sido cortado pela metade, o gráfico está com um formato menos exponencial, o que é uma melhoria significativa. Porém, o tempo total do movimento e, especialmente o seu final, ainda demora muito tempo. Portanto, foi feito novo teste com $P = 5$.

No gráfico respectivo da figura 17, é possível perceber que a velocidade se mantém constante, provavelmente no limite mencionado anteriormente, durante a maioria do percurso, o gráfico só ficando exponencial no final do movimento. O tempo total do movimento também está mais satisfatório. Para esse movimento, pode ser que essas configurações do PID resolvam. Porém, como a velocidade é proporcional à distância, com uma distância maior é provável que o comportamento fosse o mesmo do primeiro

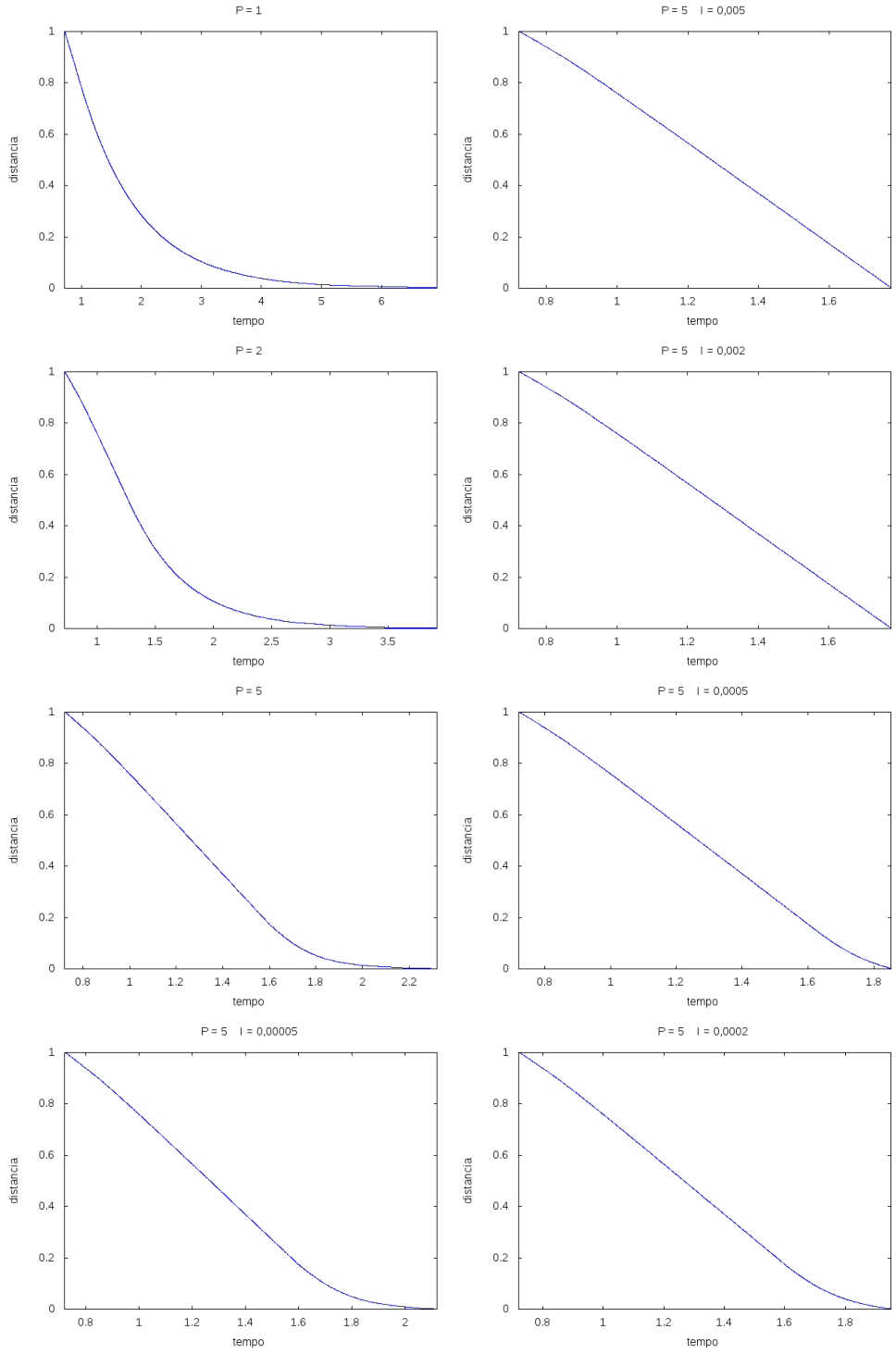


Figura 17: Distância em função do tempo de um movimento para várias combinações de parâmetros do controlador PID.

Fonte: Autor.

teste. Portanto, é necessária a inclusão do parâmetro I para também definir a velocidade proporcional ao tempo.

Como o parâmetro I acumula o erro a cada iteração, e são $1\,000$ iterações por segundo, o primeiro valor de I foi ajustado para $\frac{P}{1\,000} = \frac{5}{1\,000} = 0,005$. No gráfico respectivo da figura 17, é possível perceber que a velocidade se manteve constante do início ao fim do movimento, o que significa que o movimento foi executado no menor tempo possível. Porém, usar valores para I muito altos podem causar com que o PID acumule excesso de valor para a velocidade. No caso da plataforma não passar exatamente pelo destino, esse acúmulo vai fazer com que a plataforma se movimente por uma distância considerável até que o acúmulo negativo do erro possa neutralizar o acúmulo positivo, e quando isso ocorrer, a mesma situação pode se repetir em escala menor, com a plataforma oscilando ao redor do destino até que o acúmulo do erro se estabilize e não impessa a plataforma de chegar ao destino.

Por isso, I será ajustado até que haja quantidade considerável de desaceleração na chegada ao destino. Caso a plataforma erre o alvo, ela já estará quase parando, e sua pose poderá ser corrigida em menos tempo. No gráfico respectivo da figura 17 para $I = 0,002$, é possível perceber que não há diferença perceptível entre os resultados para $I = 0,005$. No gráfico respectivo da figura 17 para $I = 0,000\,5$, é possível perceber uma diminuição significativa da velocidade no final do movimento, enquanto que o tempo total do mesmo só foi acrescido de 1 ou 2 décimos de segundo. No gráfico respectivo da figura 17 para $I = 0,000\,2$, é possível perceber ainda mais diminuição da velocidade e aumento de mais alguns décimos no tempo total do movimento.

No gráfico respectivo da figura 17 para $I = 0,000\,05$, é possível perceber um resultado comparável com $P = 5$ e $I = 0$. Portanto, $I = 0,000\,5$ ou $I = 0,000\,2$ seria o mais ideal para o controlador. Para decidir entre os dois, foi feito outro teste. A plataforma se movimentou para 10 poses aleatórias diferentes, seguindo para a próxima assim que detectado que tinha atingido a anterior. Foi adicionado um ângulo aleatório em cada pose para adicionar perturbações, entre 0 e 2π. Esse teste foi feito para $P = 5$ e os dois valores de I mencionados anteriormente. A figura 18 mostra o gráfico para cada valor de I.

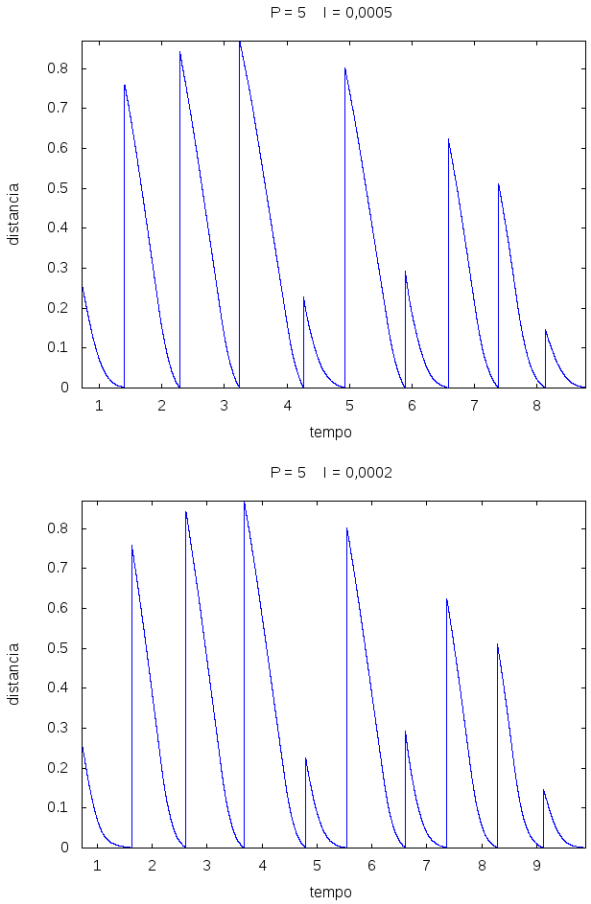


Figura 18: Distância em função do tempo de 10 movimentos aleatórios para duas combinações de valores de I do controlador PID.

Fonte: Autor.

Além do acréscimo de 1 s no tempo total do teste, não há grande diferença no desempenho em cada teste. Não houve ameaça de ultrapassagem em nenhum momento. Assim, pelo melhor desempenho, foi elegido $I = 0,000\,5$. O próximo a ser ajustado seria D, porém, de acordo com ANG; CHONG; LI (2005), muitos usuários de PID desistem de usar o termo derivativo por frustração ao tentar ajustar um valor para o mesmo, dado que o uso desse termo costuma gerar resultados contrários ao esperado, especialmente quando há um atraso de transporte. Além disso, os resultados foram considerados satisfatórios o suficiente, dado o tamanho da plataforma e a sua baixa complexidade. O trabalho de INUZUKA; SILVA LIMA (2005) também utiliza somente um controlador PI.

Também é possível perceber a relação existente entre a escolha de valores e o resultado do controlador PID, de forma que um simples ajuste empírico é suficiente para prover um ganho significativo de desempenho. Se a plataforma fosse mais complexa para justificar ainda mais desempenho, um ajuste mais fino desses parâmetros ainda seria possível de ser realizado.

5.4 Outros testes

O tempo de iteração do algoritmo foi medido durante a execução de um movimento de curva de Bézier com 6 pontos de controle. Após pouco mais de 5 000 iterações, o tempo mínimo foi de $11,088$ µs, o tempo máximo foi de $1,140\,925$ ms e a média aritmética dos tempos foi de $28,483\,595\,681$ µs.

O trabalho de GONçALVES; LIMA; COSTA (2008) foi usado de base para um teste para comparação de resultados entre aquele e este trabalho. Para melhorar a comparação, o parâmetro que define a partir de que distância que é considerado que a plataforma atingiu o seu destino foi alterado para o mesmo valor do trabalho mencionado, que é de 2 cm. A figura 19 mostra os resultados do teste. Nos dois últimos gráficos, a escala da esquerda representa o erro da odometria em metros, enquanto que a escala da direita representa a posição real da plataforma, também em metros.

Na figura 19a, é possível perceber que a trajetória percorrida por este trabalho apresentou distância da trajetória alvo entre aproximadamente $0,03$ m e $0,06$ m, sempre

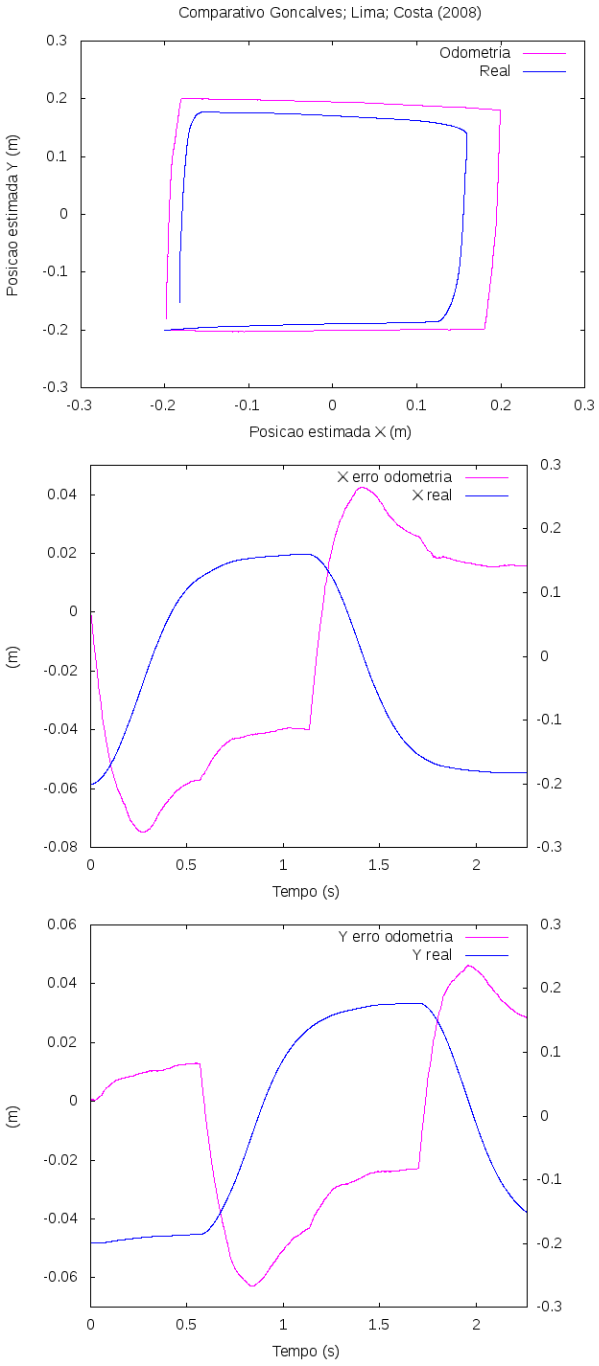


Figura 19: Resultados de teste similar aos realizados por GONçALVES; LIMA; COSTA (2008).

Fonte: Autor.

abaixo de $0,2$ m, enquanto que o robô do trabalho de GONçALVES; LIMA; COSTA (2008) apresentou distância entre aproximadamente 0 m e $0,06$ m, sempre acima de $0,2$ m. Na média, a distância apresentada por este trabalho foi menor, além da trajetória ser mais retilínea, o que significa que este trabalho tem maior capacidade de mudar de direção. Porém, o primeiro quarto da trajetória apresentada pelo trabalho de GONçALVES; LIMA; COSTA (2008) apresentou quase nenhum desvio, provavelmente devido ao uso de odometria assistida por visão computacional.

Nas figuras 19b e 19c, é possível perceber que as trajetórias em x e y executadas por este trabalho apresentaram um formato de "platô" em determinados segmentos, enquanto que as do trabalho de GONçALVES; LIMA; COSTA (2008) apresentaram maior curvatura. Isso indica, para este trabalho, estabilização no movimento ao atingir um valor próximo ao desejado, que foi atingido com mais rapidez.

Com relação à velocidade, a plataforma deste trabalho executou o movimento em $\sim 2,2$ s, enquanto que o robô de GONçALVES; LIMA; COSTA (2008) levou ~ 20 s. A velocidade menor é devida ao uso de odometria visual, pois à medida que a velocidade aumenta, o robô percorre uma distância maior a cada aquisição de uma imagem, reduzindo a precisão da odometria (GONçALVES; LIMA; COSTA, 2008). Enfim, é possível concluir que ambos os trabalhos possuem níveis similares de desempenho, cada um com algumas características que apresentam melhor desempenho do que as do outro trabalho.

6 CONCLUSÃO

O trabalho foi disponibilizado na plataforma GitHub, sob licensa MIT/Expat. Essa plataforma permite que qualquer pessoa possa livremente ter acesso aos arquivos, reportar problemas, sugerir melhorias, e interagir socialmente com os desenvolvedores (GITHUB INC., 2016). O trabalho foi disponibilizado com alguns tutoriais, tanto para quem somente modificará a programação de controle, como para quem quiser construir um robô novo usando essa plataforma como base móvel. A licensa MIT/Expat foi escolhida por ser uma das mais permissivas, além de ser nativamente suportada pelo GitHub e ser a mais usada lá (GITHUB INC., 2015), (FREE SOFTWARE FOUNDATION, 2016). O trabalho foi disponibilizado em RITTER (2016).

Foi possível concluir que a necessidade da sociedade atual pela robótica móvel, diferentemente da robótica industrial, está em estágios mais iniciais mas vem crescendo, o que justifica maior necessidade de pesquisa neste campo. Também foi possível perceber a falta de trabalhos que se proponham a atendam aos mesmos objetivos que este, justificando a relevância do mesmo.

Com os resultados dos testes, foi possível perceber a facilidade de ganho de desempenho sem a necessidade de um grande trabalho de ajuste com o uso de PID. Foi reproduzido um teste feito em um trabalho similar, resultando em desempenho melhor em alguns aspectos e pior em outros. Considerando o propósito genérico e a baixa complexidade da plataforma, os resultados foram considerados satisfatórios.

Foi possível perceber, apesar da proposta genérica deste trabalho, a existência de várias possíveis melhorias para o mesmo. Estas possibilitariam que seus objetivos

pudessem ser cumpridos em uma quantidade maior de situações.

Enfim, é possível concluir que, exceto pelo desenvolvimento do protótipo físico, este trabalho atingiu os seus objetivos com sucesso, contribuindo para o campo da robótica móvel com um ferramenta capaz de facilitar desenvolvimentos e agregar valor à pesquisas na área. Também, o autor se considera satisfeito com o ganho de novos conhecimentos e habilidades em diversos assuntos e com poder ter contribuído com sua área de maior interesse.

7 TRABALHOS FUTUROS

Este capítulo descreve o que pode ser desenvolvido futuramente no trabalho. Primeiramente, dado que não foi possível desenvolver o protótipo físico, pois diversos motivos impediram a obtenção dos materiais, todo o trabalho relacionado ao protótipo físico fica para ser desenvolvido futuramente.

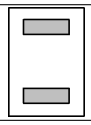
Além disso, o óbvio seria aumentar as capacidades da plataforma para que a mesma fique mais próxima do estado da arte, aumentando a sua utilidade para cumprir os seus objetivos. Um avanço significativo viria com o uso de um modelo dinâmico ao invés de um modelo cinemático, que provê melhor performance com variado custo computacional (KALMáR-NAGY; D'ANDREA; GANGULY, 2002) (KUANG; LIU; LIN, 2008) (FERNANDES; CERQUEIRA; LIMA, 2012).

Outro desenvolvimento seria a adição de mais sensores para melhorar o desempenho da odometria. Dado à natureza genérica da plataforma, seria preferível sensores que dependessem de menos informações do mundo externo, como acelerômetros e câmeras que capturassem imagens da superfície sendo percorrida pelo robô, como no trabalho de KILLPACK et al. (2010).

Acredita-se que os comandos de movimento ainda possam ser significativamente melhorados. Além do movimento de curva de Bézier necessitar de controle de velocidade ao longo da curva, acredita-se que seja possível controlar cada velocidade do robô com um comando diferente. Por exemplo, V_x ser controlado por um x alvo, V_y ser constante e ω_p ser controlado por uma curva de Bézier.

ANEXOS A TABELA DE CONFIGURAÇÕES DE RODAS PARA VEÍCULOS ROLANTES

Tabela 5: Configurações de rodas para veículos rolantes.

Número de rodas	Arranjo	Descrição	Exemplos típicos
2		Uma roda esterçável na frente, uma roda tracionável atrás	Bicicleta, motocicleta
		Unidade diferencial de duas rodas, com o centro de massa embaixo do eixo	Robô pessoal Cye
3		Unidade diferencial centralizada de duas rodas com um terceiro ponto de contato	Nomad Scout smartRob EPFL
		Duas rodas atrás/na frente atuadas independentemente, 1 roda omnidirecional livre na frente/atrás	Muitos robôs internos incluindo os robôs da EPFL Pygmalion e Alice
		Duas rodas de tração conectadas (diferencial) atrás, 1 roda livre esterçável na frente	Minicaminhões Piaggio
		Duas rodas livres atrás, 1 roda esterçável de tração na frente	Nepture (CMU), Hero-1
		Tres rodas suecas ou esféricas motorizadas dispostas em um triângulo; movimento omnidirecional é possível	Stanford wheel Tribolo EPFL, Conjunto robótico Palm Pilot (CMU)
		Três rodas motorizadas e esterçáveis sincronamente; a orientação não é controlável	"Synchro drive" Denning MRV-2, Georgia Institute of Technology, I-Robot B24, Nomad 200

Fonte: adaptado de (SIEGWART; NOURBAKHSH, 2004).

Tabela 6: Configurações de rodas para veículos rolantes (continuação).

Número de rodas	Arranjo	Descrição	Exemplos típicos
4		Duas rodas motorizadas atrás, 2 rodas esterçáveis na frente; esterço tem que ser diferente para as 2 rodas para evitar deslizamento/derrapagem.	Carro com tração traseira
		Duas rodas motorizadas e esterçáveis atrás, 2 rodas livres na frente; esterço tem que ser diferente para as 2 rodas para evitar deslizamento/derrapagem.	Carro com tração dianteira
		Quatro rodas motorizadas e esterçáveis	Carro 4×4, Hyperion de esterço nas 4 rodas (CMU)
		Duas rodas de tração (diferencial) atrás/na frente, 2 rodas omnidirecionais na frente/atrás	Charlie (DMT-EPFL)
		Quatro rodas omnidirecionais	Uranus (CMU)
		Unidade diferencial de duas rodas, com 2 pontos de contato adicionais	Khepera da EPFL, Hyperbot Chip
		Quatro rodas castor motorizadas e esterçáveis	Nomad XR4000
6		Duas rodas motorizadas e esterçáveis alinhadas no centro, 1 roda omnidirecional em cada canto	First
		Duas rodas tracionadas (diferencial) no centro, 1 roda omnidirecional em cada canto	Terregator (CMU)

Ícones para cada tipo de roda:

	Roda omnidirecional livre (esférica, castor, sueca);
	roda sueca motorizada (roda Stanford);
	roda padrão livre;
	roda padrão motorizada;
	roda castor motorizada e esterçável;
	roda padrão esterçável;
	rodas conectadas

Fonte: adaptado de (SIEGWART; NOURBAKHSH, 2004).

REFERÊNCIAS

ANG, K. H.; CHONG, G.; LI, Y. PID Control System Analysis, Design, and Technology. *IEEE Transactions on Control Systems Technology*, [S.l.], v.13, n.4, p.559–576, 2005.

BEMIS, Steven. *Design and development of a novel omni-directional platform*. 2009. 191p. Tese (Faculty of Engineering and Applied Science) — University of Ontario Institute of Technology, Oshawa, ON. (2009).

BLIKSTEIN, Paulo. Digital Fabrication and 'Making' in Education: the democratization of invention. In: WALTER-HERRMANN, J.; BüCHING, C. (Ed.). *FabLabs*: of machines, makers and inventors. Bielefeld: Transcript Publishers, 2013. p.1–22.

BORENSTEIN, J.; EVERETT, H. R.; FENG, L. *"Where am I?" Sensors and Methods for Mobile Robot Positioning*. Ann Arbor: University of Michigan, 1996. CD-ROM.

BRANCO, Sérgio. *O domínio público no direito autoral brasileiro*. Rio de Janeiro, RJ: Editora Lumen Juris, 2011.

CAMPION, G.; BASTIN, G.; D' ANDRéA-NOVEL, B. Structural Properties and Classification of Kinematic and Dynamic Models of Wheeled Mobile Robots. *IEEE Transactions on Robotics and Automation*, [S.l.], v.12, n.1, p.47–62, 1996.

CHUNG, W.; MOON, C.-b.; JUNG, C.; JIN, J. Design of the Dual Offset Active Caster Wheel for Holonomic Omni-directional Mobile Robots. *International Journal of Advanced Robotic Systems*, [S.l.], v.7, n.4, p.105–110, 2010.

CRAIG, John J. *Introduction to Robotics*. 3.ed. Upper Saddle River: Pearson Education, Inc., 2005. 408p.

CRAVO, Ana Cristina. Análise das causas da evasão escolar do curso técnico de informática em uma faculdade de tecnologia de Florianópolis. *Gestão Universitária na América Latina*, Florianópolis, v.5, n.2, p.238–250, 2012.

CREATIVE COMMONS. *Frequently Asked Questions*. Disponível em <https://creativecommons.org/faq/>. Acesso em: maio de 2016.

DASSAULT SYSTEMES. *SOLIDWORKS 3D CAD*. Disponível em <http://www.solidworks.com/sw/products/3d-cad/solidworks-3d-cad.htm>. Acesso em: maio de 2016.

DE ASSIS, Cristiano Ferreira. *Estudo dos fatores que influenciam a evasão dos alunos nos Cursos Superiores de Tecnologia de uma Instituição de Ensino Superior Privada*. 2013. 102p. Mestre em Administração — Faculdade Pedro Leopoldo, Pedro Leopoldo.

DIEGEL, Olaf. *OddBot omni-directional mecanum wheeled robot*. Disponível em <http://www.odd.org.nz/oddbot.html>. Acesso em: maio de 2016.

DUDEK, G.; JENKIN, M. *Computational Principles of Mobile Robotics*. 2.ed. Nova Iorque, NY: Cambridge University Press, 2010. 391p.

DUKE UNIVERSITY. *The Incredible Shrinking Public Domain*. Disponível em <http://web.law.duke.edu/cspd/publicdomainday/2013/shrinking>. Acesso em: maio de 2016.

FERNANDES, C. A. P.; CERQUEIRA, J. d. J. F.; LIMA, A. M. N. Dinâmica não linear do escorregamento de um robô móvel omnidirecional com restrição de rolamento. In: XIX CONGRESSO BRASILEIRO DE AUTOMáTICA, CBA 2012, 2012, Campina Grande. *Anais...* DEE/UFCG, 2012. p.687–694.

FREE SOFTWARE FOUNDATION. *O Que é "Esquerdo de Cópia"(Copyleft)?* Disponível em <https://www.gnu.org/server/select-language.html?goto=/copyleft/;language=pt-br>. Acesso em: maio de 2016.

FREE SOFTWARE FOUNDATION. *Various Licenses and Comments about Them*. Disponível em <https://www.gnu.org/licenses/license-list.en.html#Expat>. Acesso em: novembro de 2016.

GALLIER, Jean. *Curves and Surfaces In Geometric Modeling*: theory and algorithms. Burlington, MA: Morgan Kaufmann, 1999. 492p. Disponível em <http://www.cis.upenn.edu/jean/geomcs-v2.pdf>. Acesso em: junho de 2016.

GARCIA, E.; JIMENEZ, M. A.; SANTOS, P. G. D.; ARMADA, M. The Evolution of Robotics Research. *IEEE Robotics & Automation Magazine*, [S.l.], v.14, n.1, p.90–103, 2007.

GARCíA, José Felipe Camarena. *Análisis Cinemático, Dinámico y Control en Tiempo Real de un Vehículo Guiado Automáticamente*. 2009. 137p. Maestría en Ciencias en Ingeniería Mecatrónica — Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca.

GHOSH, B. K.; XI, N.; TARN, T.-J. *Control in Robotics and Automation*. San Diego, CA: Academic Press, 1999. 428p.

GITHUB INC. *Open source license usage on GitHub.com*. Disponível em <https://github.com/blog/1964-license-usage-on-github-com>. Acesso em: novembro de 2016.

GITHUB INC. *GitHub Help*. Disponível em <https://help.github.com/>. Acesso em: novembro de 2016.

GONçALVES, J.; LIMA, J.; COSTA, P. REAL TIME TRACKING OF AN OMNIDIRECTIONAL ROBOT - An Extended Kalman Filter Approach. In: ICINCO 2008, PROCEEDINGS OF THE FIFTH INTERNATIONAL CONFERENCE ON INFORMATICS IN CONTROL, AUTOMATION AND ROBOTICS, ROBOTICS AND AUTOMATION 2, FUNCHAL, MADEIRA, PORTUGAL, MAY 11-15, 2008, 2008, Funchal. *Anais...* INSTICC Press, 2008. p.5–10.

GRITSCH, Markus. *Omnidirectional Remote Controlled Robot*. Disponível em <http://dangerousprototypes.com/forum/viewtopic.php?f=56&t=2019>. Acesso em: maio de 2016.

HANWELL, Marcus D. *Should I use a permissive license? Copyleft? Or something in the middle?* Disponível em <https://opensource.com/business/14/1/what-license-should-i-use-open-source-project>. Acesso em: maio de 2016.

HOLMBERG, R.; KHATIB, O. Development and Control of a Holonomic Mobile Robot for Mobile Manipulation Tasks. *International Journal of Robotics Research*, Salt Lake City, v.19, n.11, p.1066–1074, 2000.

INUZUKA, H.; SILVA LIMA, L. da. *Desenvolvimento de um Robô Móvel Omnidirecional*: o robô rohl. 2005. 56p. Monografia (Departamento de Engenharia Elétrica) — Universidade de Brasília, Brasília. (2005).

JONES, J. L.; FLYNN, A. M.; SEIGER, B. A. *Mobile robots*: inspiration to implementation. 2.ed. Natick, MA: A K Peters, Ltd, 1999. 486p.

JORGE, B. G.; MARTINS, C. Z.; CARNIEL, F.; LAZILHA, F. R.; VIEIRA, M. C.; GOI, V. M. Evasão na Educação a Distância: um estudo sobre a evasão em uma instituição de ensino superior. In: CIAED - CONGRESSO INTERNACIONAL ABED DE EDUCAçãO A DISTâNCIA, 16., 2010, Foz do Iguaçu. *Anais...* [S.l.: s.n.], 2010. p.1–10. CD-ROM.

KALMáR-NAGY, T.; D'ANDREA, R.; GANGULY, P. Near-Optimal Dynamic Trajectory Generation and Control of an Omnidirectional Vehicle. *Robotics and Autonomous Systems*, Países Baixos, v.46, n.1, p.47–64, 2002.

KENNEDY, Dennis M. *A Primer on Open Source Licensing Legal Issues*: copyright, copyleft and copyfuture. Disponível em <http://www.cs.miami.edu/home/burt/learning/Csc322.052/docs/opensourcedmk.pdf>. Acesso em: maio de 2016.

KILLPACK, M.; DEYLE, T.; ANDERSON, C.; KEMP, C. C. Visual Odometry and Control for an Omnidirectional Mobile Robot with a Downward-Facing Camera. In: IEEE/RSJ INTERNATIONAL CONFERENCE ON INTELLIGENT ROBOTS AND SYSTEMS (IROS), 2010., 2010, Taipei. *Anais...* IEEE, 2010. p.139–146.

KNOLL, Alois Christian. *OpenRobertino.org*. Disponível em <http://www.openrobertino.org/doc/index.html>. Acesso em: maio de 2016.

KUANG, J. M.; LIU, M.; LIN, X. Moving Target Tracking of Omnidirectional Robot with Stereo Cameras. In: BHATTI, Asim (Ed.). *Stereo Vision*. Vienna, Austria: InTech, 2008. p.197–220.

KURFESS, Thomas R. *Robotics and automation handbook*. Boca Raton: CRC Press LLC, 2005. 579p.

LOBO E SILVA FILHO, R. L.; MOTEJUNAS, P. R.; HIPóLITO, O.; CARVALHO MELO LOBO, M. B. de. A Evasão no Ensino Superior Brasileiro. *Cadernos de Pesquisa*, [S.l.], v.37, n.132, p.1–18, 2007.

OPEN SOURCE INITIATIVE. *Frequently Answered Questions*. Disponível em <https://opensource.org/faq>. Acesso em: maio de 2016.

OPEN SOURCE ROBOTICS FOUNDATION. *Gazebo*. Disponível em <http://gazebosim.org/>. Acesso em: agosto de 2016.

RASPBERRY PI FOUNDATION. *Raspberry Pi FAQs - Frequently Asked Questions*. Disponível em <https://www.raspberrypi.org/help/faqs/>. Acesso em: maio de 2016.

RASPBERRY PI FOUNDATION. *Raspberry Pi 2 Model B*. Disponível em <https://www.raspberrypi.org/products/raspberry-pi-2-model-b/>. Acesso em: maio de 2016.

RASPBERRY PI WIKI. *RPi Hardware*. Disponível em <http://elinux.org/RPi_Hardware>. Acesso em: maio de 2016.

RITTER, Guilherme Alan. *GuiRitter/OpenBase*: an omnidirectional mobile platform with a 3 omnidirectional wheels layout. Disponível em <https://github.com/GuiRitter/OpenBase>. Acesso em: novembro de 2016.

ROBOTS-R-US. *38MM DOUBLE PLASTIC OMNI WHEEL (COMPATIBLE WITH SERVO MOTOR) | Wheel - Omni | Robot R Us*. Disponível em <https://www.robot-r-us.com/wheel-omni/38mm-double-plastic-omni-wheel-compatible-with-servo-motor.html>. Acesso em: outubro de 2016.

ROGERS, D. F.; ADAMS, J. A. *Mathematical Elements for Computer Graphics*. 2.ed. New York City: McGraw-Hill, Inc., 1990. 611p.

SECCHI, Humberto Alejandro. *Una Introducción a los Robots Móviles*. 2008. 78p. Dissertação (Instituto de Automática) — Universidade Nacional de San Juan, San Juan, San Juan, Argentina. (2008).

SICILIANO, B.; SCIAVICCO, L.; ORIOLO, L. V. G. *Robotics*. 1.ed. Londres, Inglaterra: Springer-Verlag London Limited, 2009. 632p.

SIEGWART, R.; NOURBAKHSH, I. R. *Introduction to Autonomous Mobile Robots*. Cambridge, MA: The MIT Press, 2004. 335p.

SMITH, C. A.; CORRIPIO, A. B. *Principles and Practices of Automatic Process Control*. 2.ed. Hoboken, NJ: John Wiley & Sons, Inc., 1997. 580p.

STALLMAN, Richard. *Why Open Source misses the point of Free Software*. Disponível em <https://www.gnu.org/philosophy/open-source-misses-the-point.html.en>. Acesso em: mar. de 2016.

WORDMAN, Paul B. *$35 Raspberry Pi Computer is No Fruitcake*. Disponível em <http://blog.parts-people.com/2012/01/05/35-raspberry-pi-computer-is-no-fruitcake/>. Acesso em: maio de 2016.