# Simulador Gráfico para Controle de Robôs Móveis Omnidirecionais

GREYCE NOGUEIRA SCHROEDER[1]
DANÚBIA BUENO ESPÍNDOLA[2]
SILVIA SILVA DA COSTA BOTELHO[3]
ALESSANDRO DE LIMA BICHO[4]
VINÍCIUS MENEZES DE OLIVEIRA[5]

[1]Engenharia de Computação – Fundação Universidade Federal de Rio Grande
Av. Itália, km 8 - Pav. M - CEP 96201-900 Rio Grande, RS
[2]DCA – FEEC – Unicamp
Rua Albert Einstein, 400 – 13083-970 Campinas, SP
[1]sgreyce@dca.fee.unicamp.br
[2]dmtdbe@furg.br
[3]silviacb@ee.furg.br
[4]bicho@dca.fee.unicamp.br
[5]vinicius@ieee.org

**Resumo.** O desenvolvimento de um simulador gráfico para robôs móveis visa fornecer uma ferramenta gráfica computacional que auxilie, através de gráficos e animação, o projeto e a análise de leis de controle para robôs móveis a rodas, tanto em relação à definição de que tipo de técnica utilizar, quanto ao ajuste de parâmetros do controlador. As interfaces desenvolvidas permitem ao usuário definir, dentre os diferentes tipos de robôs com rodas, parâmetros de controle e de simulação, fornecendo a visualização de forma separada, de todas as informações necessárias ao usuário, no período de execução da tarefa. A modelagem cinemática e dinâmica concentrou seu foco na classe de robôs omnidirecionais, porém a estrutura do simulador foi desenvolvida para dar suporte às demais classes de robôs móveis com rodas existentes. A implementação de tal ferramenta se apresenta como uma interessante forma de ensino e de aprendizado em disciplinas que envolvam robótica móvel e técnicas de controle para tais sistemas.

**Palavras-Chave:** robôs móveis omnidirecionais, simulação computacional, leis de controle, análise gráfica.

# Graphical Simulator for Omnidirectional Mobile Robots Control

**Abstract.** The development of this graphical simulator for mobile robots aims to supply a computational graphical tool that assists, through graphs and animation, the project and analysis of control laws for wheeled mobile robots, as much concerning definiton of control technique as as to the tunning of controller's parameters. The developed interfaces allow the user to define among all 5 different classes of wheele mobile robots, simulation and control parameters, supplying the independent visualization of all necessary information to the designer, while the task execution is execute. The kinematics and dynamics models developed are for the omnidirectional wheeled mobile robots, however, the structure of the simulator was developed to support all classes of wheeled mobile robots. The implementation of such tool is presented as an interesting form of education and knowledge acquisition in disciplines that involve mobile robotics and techniques of control for such systems.

**Keywords:** Omnidirectional Mobile Robots, Computational Simulator, Control.

(Received June 10, 2005 / Accepted August 12, 2005)

## 1 Introdução

Esse trabalho tem por objetivo a implementação de uma ferramenta computacional gráfica que auxilie no projeto de leis de controle para robôs móveis. Várias ferramentas já foram desenvolvidas com essa mesma finalidade, entretanto nenhuma satisfaz completamente as nossas expectativas, motivo pelo qual se propõe esse trabalho. Algumas questões pertinentes às funcionalidades dos diversos simuladores analisados foram elaboradas, a fim de se identificar o que cada ferramenta oferece, como por exemplo, se o simulador possibilita ajustar os parâmetros de controle e quais as leis de controle disponibilizadas E se há possibilidade de incluir novas técnicas de controle, se a interface permite a escolha da lei de controle e se há como se escolher diferentes tipos de robôs.

Entre os simuladores encontrados destacamos os mais relevantes para o nosso estudo. BotControler [1] é um simulador gráfico 2D para ambiente Windows© que serve para simular robôs desenvolvidos pela K-Team [8], tais como Khepera, Koala e Hemisson. Nesse simulador não são apresentados parâmetros de controle nas interfaces e, assim como os parâmetros, as leis de controle devem ser programadas pelo usuário. As leis de controle podem ser alteradas diretamente no código, que deve ser recompilado a cada alteração na lei de controle. A estrutura física deste simulador abrange os 3 robôs produzidos pela K-Team e não há opção para modificá-los.

O Cyberbotics [3] é um simulador 3D para robôs como Khepera, Koala e demais que suportem o mesmo tipo de *driver*, estando disponível para os ambientes Linux e Windows©. Não são apresentados os parâmetros de controle na interface nem as leis de controle, as quais devem ser editadas e alteradas pelo usuário no código fonte, sendo C++ a linguagem de programação utilizada [2]. As estruturas físicas são limitadas por robôs do tipo Khepera e Koala.

O SIM1 é um simulador gráfico para robôs manipuladores desenvolvido pelo Núcleo de Matemática Aplicada da Fundação Universidade Federal do Rio Grande [9]. O usuário pode escolher a técnica de controle, ajustar os parâmetros do controlador e parâmetros da simulação. Também é possível se definir diferentes leis de controle, importando o código na linguagem Pascal. Embora esse simulador seja para robôs manipuladores, destaca-se por apresentar as características desejadas para o nosso simulador.

O RoboWorks [11] é um ambiente 3D desenvolvido somente para plataforma Windows ©, que simula e anima qualquer sistema físico, permitindo a implementacao do modelo desejado. Os parâmetros de controle apresentados são utilizados praticamente para todas classes de robôs. As leis de controle usadas são: dinâmica e estática, não sendo possível modificá-las. O usuário pode modificar alguns parâmetros das leis de controle.

SIMRobot [12] permite simular o comportamento de um ou mais robôs móveis em um ambiente virtual, onde cada robô pode ser equipado com diversos sensores virtuais e dirigido por um algoritmo de controle (editável pelo usuário), utilizando lógica fuzzy e rede neural. O usuário pode criar modelos para diferentes robôs.

No simulador Juice [7] os parâmetros de controle são definidos em uma janela para cada parte do robô e não foi possível determinar se há a implementação de leis de controle ou não.

Nota-se claramente a dificuldade de se encontrar ferramentas de simulação para a análise e o desenvolvimento de sistemas robóticos móveis em suas diversas classes. Os sistemas existentes não abrangem todas as características necessárias para o estudo completo da robótica móvel. Tais sistemas estão direcionados apenas a uma classe com determinadas variáveis, mas nenhum se preocupa com todos os aspectos das diversas classes dos robôs móveis com rodas. Em nenhum dos sistemas existentes encontra-se uma junção completa das características necessárias para o estudo, tais como parâmetros de controle, do modelo e da simulação do sistema, que pudessem ser modificados e visualizados facilmente pelo usuário, no período durante o qual o robô realiza sua tarefa. Também não há recursos para que o usuário possa escolher e testar leis de controle que melhor se adaptem às classes de robôs móveis em questão no simulador.

Visando ocupar essa lacuna é que se propôs desenvolver um simulador com interface amigável, de caráter didático, que permita ao usuário escolher a classe do robô móvel a estudar e, a partir desta, definir seu controle para que o usuário possa analisar, através dos gráficos e da animação, como se comportaria o robô com os parâmetros indicados. Este simulador poderá ser utilizado tanto por estudantes quanto por pesquisadores e até mesmo curiosos da área de robótica móvel, que queiram aprender ou aplicar seus conhecimentos em estudo ou experimentos da área.

Dentre as diversas aplicações, a implementação de tal ferramenta se apresenta como uma interessante forma de ensino e aprendizado em disciplinas que envolvem Robótica Móvel e o controle de tais sistemas. Nessa primeira etapa do desenvolvimento do simulador, se implementa somente a classe de robôs omnidirecionais, deixando como proposta futura a implementação das demais classes existentes na robótica móvel.

## 2 Robótica Móvel

A robótica móvel é uma área de pesquisa que possui ênfase no estudo de técnicas de controle de robôs autônomos e semi-autônomos na sua locomoção em ambientes dinamicamente instáveis. Nestes ambientes, podem existir obstáculos móveis e fixos e, portanto, o robô deve ser capaz de perceber, se localizar e se mover em ambientes complexos desse tipo, tomando decisões a partir de informações provenientes de mecanismos de sensoreamento. Tais veículos que puderem executar tarefas dessa maneira são objetos de estudo da robótica móvel.

Os temas de pesquisa na área da robótica móvel vão desde modelagem e estratégia de controle até tipos de sistemas de locomoção e técnicas de inteligência artificial. O controle em baixo nível concentra-se nos atuadores, enquanto que o controle de alto nível projeta arquiteturas para gerenciamento de tarefas e planejamento de trajetórias. Neste projeto preocupou-se inicialmente com controle em baixo nível, deixando, porém, suporte para posterior inclusão de todos os temas de pesquisa no simulador, para que este fique o mais completo possível na área de robôs móveis com rodas.

Robôs móveis com rodas podem ser divididos em cinco diferentes classes, de acordo com sua mobilidade e manobrabilidade. Nesse trabalho, motivados pelo desenvolvimento dos robôs no projeto Furgbol [4], primeiramente foi implementada a classe de robôs móveis omnidirecionais (classe (3,0)). Logo, um dos objetivos é estudar aspectos de modelagem cinemática e dinâmica desse tipo de robô, para implementacao no simulador.

### 2.1 Descrição do Robô

Um robô móvel omnidirecional tem mobilidade completa no plano, ou seja, pode se movimentar em qualquer direção sem a necessidade de se reorientar. Tal mobilidade não acontece nos robôs mais comuns, com duas rodas, onde para se locomover em uma dada direção é necessário que o robô esteja na orientação apropiada. A omnidirecionalidade no robô Furgbol é obtida através do controle individual de três rodas na base do robô [15]. A estrutura da base do robô pode ser visualizada na Figura 1.

Nota-se que pela combinação da força aplicada em cada roda é possível compor uma velocidade linear e uma velocidade angular resultantes . Assim, controlando-se individualmente cada motor, é possível movimentar o robô em qualquer direção e sentido. Porém, o uso de rodas comuns, em forma de disco, geraria um excessivo atrito, já que nem sempre estas movimentam-se na sua direção normal de giro. Para se transpor tal limitação

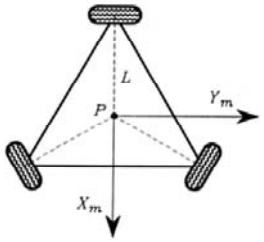


**Figura 1:** Estrutura da base do robô.

das rodas comuns, utilizam-se rodas especiais com dois graus de liberdade, conforme ilustra a Figura 2.

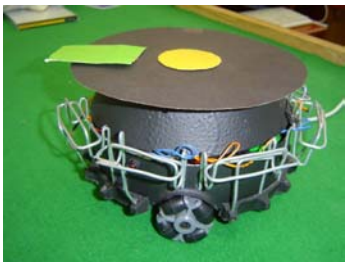

**Figura 2:** Protótipo FurgBol.

Os robôs utilizados nesse trabalho são feitos de uma estrutura rígida com rodas não deformáveis que se movem em um plano horizontal. A posição do robô pode ser descrita em termos de duas coordenadas $x$, $y$ do ponto $P$, (conforme Figura 1) e pela orientação $\vartheta$ da base móvel. Assim, a postura do robô é dada por um vetor $(3 \times 1)$:

$$\xi = \begin{bmatrix} x \\ y \\ \vartheta \end{bmatrix} \tag{1}$$

e a matriz de rotação que expressa a orientação do sistema de referência fixo à base do robô em relação ao sistema de referência inercial é:

$$R(\vartheta) = \begin{bmatrix} cos(\vartheta) & sin(\vartheta) & 0 \\ -sin(\vartheta) & cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

Foi assumido que, durante a realização do movimento, o plano de cada roda permanece na vertical, as rodas giram na horizontal e existe um único ponto de contato entre a roda e o solo. Para as rodas convencionais, o contato entre a mesma e o solo satisfaz as condições de puro rolamento e não derrapagem durante a locomoção. Isso significa que a velocidade no ponto de contato é igual a zero, implicando que as componentes dessa velocidade, paralelas ao plano de cada roda e ortogonais a esse plano, são iguais a zero.

Para as rodas omnidirecionais, somente uma componente da velocidade do ponto de contato da roda com o solo é suposto ser igual a zero ao longo da locomoção. A direção da componente zero da velocidade é, *a priori* arbitrária, sendo fixa com a respectiva orientação da roda.

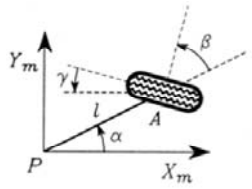


**Figura 3:** Roda omnidirecional.

A posição das rodas omnidirecionais é descrita como uma roda fixa, por três parâmetros constantes: $\alpha$, $\beta$ e $l$ . Um parâmetro adicional é requerido para caracterizar a direção com o respectivo plano da roda, da componente zero da velocidade e do ponto de contato representado pelo ângulo $\gamma$ (Figura 3).

A configuração do robô é descrita pelas coordenadas de postura $\xi(t) = \begin{bmatrix} x(t) & y(t) & \vartheta(t) \end{bmatrix}^T$, orientação $\beta(t) = 0$ e rotação $\varphi(t) = \varphi_{sw}(t)$. As restrições de locomoção deste tipo de roda são expressas como:

$$J_{sw}R(\vartheta)\dot{\xi} + J_2\dot{\varphi} = 0 \tag{3}$$

$$C_1(\beta_s, \beta_c)R(\vartheta)\dot{\xi} + C_2\dot{\beta}_c = 0 \tag{4}$$

onde $J_{sw}$ é uma matriz constante de dimensão $3 \times 3$, obtida a partir da Equação 3 e $J_2$ também é uma matriz constante de dimensão $3 \times 3$, dadas por:

$$J_1 = J_{1sw} = \begin{bmatrix} -\frac{\sqrt{3}}{2} & \frac{1}{2} & L \\ 0 & -1 & L \\ -\frac{\sqrt{3}}{2} & \frac{1}{2} & L \end{bmatrix} \tag{5}$$

$$J_2 = \begin{bmatrix} r & 0 & 0 \\ 0 & r & 0 \\ 0 & 0 & r \end{bmatrix} \tag{6}$$

## 3 Modelos Cinemático e Dinâmico de Robôs Móveis com Rodas

O modelo cinemático relaciona deslocamento, velocidade, aceleração e tempo, sem se preocupar com a causa do movimento. Já o modelo dinâmico relaciona as forças generalizadas provida aos atuadores, considerando a energia aplicada ao sistema [16].

### 3.1 Modelo Cinemático

O robô Furgbol possui três rodas dispostas como se estivessem nas extremidades de um triângulo eqüilátero, conforme pode ser visto na Figura 1. As constantes são especificadas na tabela 1, de acordo com a Figura 3.

| $roda$ | $\alpha$ | $\beta$ | $\gamma$ | $l$ |
|---|---|---|---|---|
| 1 | $\pi/3$ | 0 | 0 | L |
| 2 | $\pi$ | 0 | 0 | L |
| 3 | $5\pi/4$ | 0 | 0 | L |

**Tabela 1:** Constantes da classe de robos (3,0) Swedish rodas.

De acordo com [16], para todo $t$ existe um vetor variante no tempo $\eta$ tal que:

$$\dot{\xi} = R^T(\vartheta)\Sigma(\beta_s)\eta \tag{7}$$

A dimensão do vetor $\eta$ são os graus de mobilidade $\delta_m$ do robô. Obviamente que nesse caso onde o robô não tem rodas de direção ($\delta_s = 0$), a matriz $\Sigma$ é constante e a expressão reduz para:

$$\dot{\xi} = R^T(\vartheta)\eta \tag{8}$$

O modelo cinemático de robôs omnidirecionais, tendo o ponto $P$ e o eixo $X_m$ e $Y_m$ selecionados arbitrariamente, é então:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\vartheta} \end{bmatrix} = \begin{bmatrix} cos(\vartheta) & sin(\vartheta) & 0 \\ -sin(\vartheta) & cos(\vartheta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \end{bmatrix} \tag{9}$$

sendo $\eta_1$ e $\eta_2$ as componentes de velocidade de $X_m$ e $Y_m$ (Figura 1), respectivamente, enquanto que $\eta_3$ é a velocidade angular do robô.

A Equação 9 é a Equação que foi implementada no modelo cinemático. Esta implementação será mostrada na seção 4.

### 3.2 Modelo Dinâmico

O modelo dinâmico do espaço de estados de robôs móveis descreve as relações dinâmicas entre as coordenadas de configuração $\xi$, $\varphi$ e os torques desenvolvidos pelos atuadores. Os torques providos pelos atuadores são denotados por $\tau_\varphi$ para a rotação das rodas.

Para a determinação do modelo dinâmico adotou-se o formalismo de Euler-Lagrange, visto que é simples e de fácil desenvolvimento das Equações, apresentado a seguir:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right)^T - \left(\frac{\partial L}{\partial q}\right)^T = \Sigma Q \tag{10}$$

onde $L$ é o Lagrangeano do sistema, dado pela diferença entre a energia cinética e a energia potencial do sistema. O termo $\Sigma Q$ é o somatório dos esforços externos ou não conservativos e $q$ são as coordenadas generalizadas de configuração, dadas por:

$$q = (\ x \quad y \quad \vartheta \quad \varphi_1 \quad \varphi_2 \quad \varphi_3\ )^T \tag{11}$$

onde $x$ é coordenada ao longo do eixo $X$ do robô, $y$ é coordenada ao longo do eixo $Y$ do robô, $\vartheta$ é ângulo de orientação, $\varphi_1$ é ângulo de orientação da roda 1, $\varphi_2$ é ângulo de orientação da roda 2, e $\varphi_3$ é ângulo de orientação da roda 3. A energia cinética é constituída pela energia cinética da base e de cada uma das rodas. A energia potencial é igual a zero, pois adotou-se que o robô está situado em uma superficie horizontal plana.

A energia cinética total do robô, como apresentado em [15], pode ser expressa por:

$$T = \dot{\xi}^T R^T(\vartheta) M R(\vartheta) \dot{\xi} + \dot{\varphi}^T I_\varphi \dot{\varphi} \tag{12}$$

sendo a matriz $M$ uma matriz diagonal contendo a massa da base e dos motores e $I_\varphi$ uma matriz diagonal com a inércia do motor. Os torques provido dos atuadores são denotados por $\tau_\varphi$ para a rotação de cada uma das rodas.

Aplicando o formalismo de Euler-Lagrange obtém-se o modelo dinâmico como segue:

$$\begin{aligned} \frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\xi}}\right)^T - \left(\frac{\partial T}{\partial \xi}\right)^T &= R^T(\vartheta) J_1^T \lambda + R^T(\vartheta) C_1^T \mu \\ \frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\varphi}}\right)^T - \left(\frac{\partial T}{\partial \varphi}\right)^T &= J_2^T \lambda + \tau_\varphi \end{aligned} \tag{13}$$

onde $T$ é a energia cinética e, $\lambda$ e $\mu$ são os multiplicadores de Lagrange.

Para eliminar os multiplicadores de Lagrange, a Equação 13 é multiplicada pelas matrizes $R(\vartheta)$ e $E^T$, sendo, após, somadas. Isto leva a seguinte Equação, onde os multiplicadores de Lagrange desaparecem:

$$\Sigma^T \left(R(\vartheta)[T]_\xi + E^T [T]_\varphi\right) = \Sigma^T \left(E \tau_\varphi\right) \tag{14}$$

A seguinte notação mais compacta do modelo dinâmico pode ser utilizada:

$$[T]_\psi = \frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\psi}}\right)^T - \left(\frac{\partial T}{\partial \psi}\right)^T \tag{15}$$

A seguir, substituindo a generalização utilizada na Equacao 15 por $\xi$ e $\varphi$, e, desenvolvendo as respectivas derivadas parcias tem-se:

$$[T]_\xi = \frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\xi}}\right)^T - \left(\frac{\partial T}{\partial \xi}\right)^T \tag{16}$$

$$[T]_\xi = 2M\ddot{\xi} \tag{17}$$

$$[T]_\varphi = \frac{d}{dt}\left(\frac{\partial T}{\partial \dot{\varphi}}\right)^T - \left(\frac{\partial T}{\partial \varphi}\right)^T \tag{18}$$

$$[T]_\varphi = 2I_\varphi \ddot{\varphi} \tag{19}$$

Então, substituindo 17 e 19 na Equação 13:

$$\begin{cases} 2M\ddot{\xi} = R^T J_1^T \lambda + R^T C_1^T \mu \\ 2I_\varphi \ddot{\varphi} = J_2^T \lambda + \tau_\varphi \end{cases} \tag{20}$$

E, substituindo 17 e 19 na Equação 14:

$$\Sigma^T \left(2R(\vartheta) M \ddot{\xi} + 2E^T I_\varphi \ddot{\varphi}\right) = \Sigma^T E \tau_\varphi \tag{21}$$

$$2R(\vartheta) M \ddot{\xi} + 2E^T I_\varphi \ddot{\varphi} = E\tau_\varphi \tag{22}$$

Entretanto, segundo [16], a configuração do modelo dinâmico tem a seguinte forma geral:

$$\dot{\xi} = R^T(\vartheta)\eta \tag{23}$$

$$\dot{\varphi} = E\eta \tag{24}$$

$$H_1 \dot{\eta} + f_1 = BP\tau_\varphi \tag{25}$$

mas, derivadando-se parcialmente as Equações 23 e 24:

$$\ddot{\xi} = R^T(\vartheta)\dot{\eta} + \dot{R^T}(\vartheta)\eta \tag{26}$$

$$\ddot{\varphi} = E\dot{\eta} + \dot{E}\eta \tag{27}$$

e, substituindo em 22:

$$RMR^T\dot{\eta} + RM\dot{R^T}\eta + E^T I_\varphi E\dot{\eta} = E^T\tau_\varphi \tag{28}$$

$$\dot{\eta}\left(RMR^T + E^T I_\varphi E\right) + RM\dot{R^T}\eta = E^T\tau_\varphi \tag{29}$$

$$\dot{\eta}\left(M + I_\varphi\right) + RM\dot{R^T}\eta = E^T\tau_\varphi \tag{30}$$

Então pode-se dizer que o modelo dinâmico é:

$$\left(M + I_\varphi\right)\dot{\eta} + RMR^T\dot{\eta} = E^T\tau_\varphi \tag{31}$$

$$H\dot{\eta} + f = BP\tau_m \tag{32}$$

sendo:

$$B = E^T = -J_2^{-1} J_1 \tag{33}$$

$$(\tau_\varphi) = P\tau_m \tag{34}$$

A configuração do modelo dinâmico de robôs móveis omnidirecionais pode ser reescrita em uma forma mais compacta:

$$\dot{q} = S(\dot{q})\eta \tag{35}$$

$$H\dot{\eta} + f(\eta) = F\tau_0 \tag{36}$$

sendo:

$$\tau_0 = F^{-1}\left(H\nu + f(\eta)\right) \tag{37}$$

$$F = B \tag{38}$$

### 3.3 Lei de Controle

Para validação do modelo dinâmico desenvolvido, utilizou-se uma linearização por realimentação estática de estado, a fim de facilitar o projeto da lei de controle por torque computado [16]. Diferentes técnicas de controle podem ser desenvolvidas, de acordo com a tarefa que se deseja realizar com o robô [17]. Sendo o modelo dinâmico descrito pela Equação 35 obtém-se a seguinte realimentação linearizante:

$$\tau_0 = F^{-1}(H\nu + f(\eta)) \tag{39}$$

Subtituindo a 39 em 35 resulta:

$$\dot{q} = S(q)u \tag{40}$$

$$\dot{u} = \nu \tag{41}$$

Para demonstrar a validade do modelo dinâmico desenvolvido, utiliza-se a seguinte lei de controle:

$$\nu = R^{-T}\left(-\dot{R}^T\eta + \ddot{\xi}_r - (\Lambda_1 + \Lambda_2)\dot{\tilde{\xi}} - \Lambda_1\Lambda_2\tilde{\xi}\right) \tag{42}$$

sendo $\tilde{\xi} = \xi - \xi_r$ e, $\Lambda_1$ e $\Lambda_2$ são matrizes diagonais positivas definidas.

## 4 Implementação do Modelo Cinemático e Dinamico

A implementação do modelo cinemático do robô omnidirecional foi feita a partir da Equação 10 e, a do modelo dinâmico, pelas Equações 35 e 36. A Figura 4 mostra um fluxograma do algoritmo de implementação do modelo cinemático e a Figura 5 a implementação do modelo dinâmico.

Para o modelo cinemático, os valores de *t, x, y, PHI, ni1, ni2* e *ni3* são salvos no arquivo *omnikin.dat*, que conterá as informacões necessárias para plotar os gráficos; já no modelo dinâmico, os valores das coordenadas de postura da posição onde o robô deveria estar e, os valores dos torques de cada uma das rodas no tempo em questão são salvos no arquivo *omnidin.dat*.

Os gráficos gerados pelo simulador são:

- Posição X ao longo do tempo;
- Posição Y ao longo do tempo;
- Ângulo de orientação ao longo do tempo;
- Trajetória realizada ao longo do tempo.

A simulação do modelo dinâmico, assim como do cinemático, será realizada no instante em que as novas posições do robô são integradas pelo método de Runge Kutta [18], esta funcão é responsável pelos cálculos dos valores de posição do robô apartir da equação do modelo dinâmico, dos valores de postura ($\xi$), dos valores dos torques das rodas e do valor do tempo.

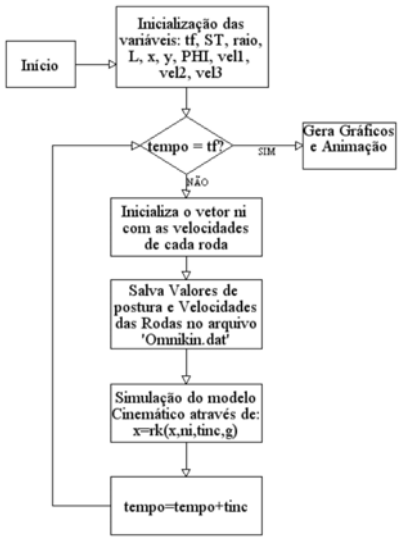


**Figura 4:** Fluxograma do Modelo Cinemático.

Enquanto, o valor de $t$ não for igual ao tempo de simulação desejado $tf$ pelo usuário, novos valores de posição são gerados. Quando o tempo for atingido, são gerados os gráficos a partir do GnuPlot [5].

## 5 Ambiente Gráfico

### 5.1 Ferramentas para desenvolvimento

Para desenvolvimento do simulador optou-se pela linguagem C++, devido às suas potencialidades para implementações das mais diversas aplicações. O fato dessa linguagem possuir tanto características de baixo nível quanto de alto nível, aliada ao paradigma da orientação a objetos, foram fundamentais para escolha, além da portabilidade do código escrito nesta linguagem, possibilitando, a partir de compiladores apropriados, gerar executáveis para máquinas de diferentes portes e diferentes sistemas operacionais.

Devido à necessidade em se utilizar funções gráficas no ambiente Unix para o desenvolvimento do simulador escolheu-se a biblioteca Qt [13]. A Qt é uma classe da biblioteca C++ e uma caixa de ferramentas GUI para o sistema Unix e X11. Para o desenvolvimento de uma interface gráfica no sistema Unix, poderia ser utilizada a GTK [6], porém escolheu-se a Qt por ser portátil e poderosa, em termos de execução, já que a linguagem de desenvolvimento é C++. A portabilidade foi o diferencial na escolha, pois uma vez desenvolvendo o programa, para implementação de tal interface, torna-se

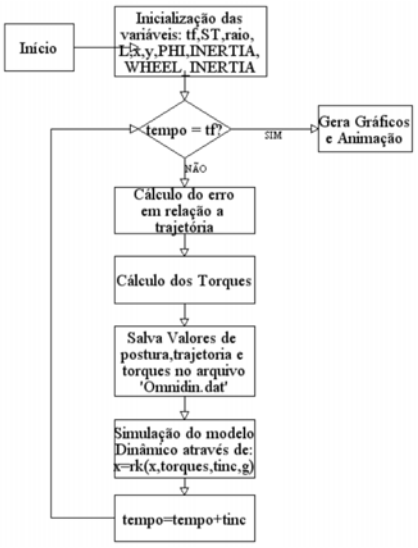


**Figura 5:** Fluxograma do Modelo Dinâmico.

praticável a recompilação para outras plataformas aumentando assim o público alvo.

Em relação às interfaces, o Qt emula tanto o aspecto do Motif no ambiente Unix, quanto o aspecto do sistema de janelas do ambiente Windows©. Portanto, podemos prover aos usuários interfaces de aspecto original ao ambiente em que estão acostumados a trabalhar. Além das classes de interface com o usuário, a Qt representa suporte portável ao sistemas de arquivos, trabalhando com valores de tempo, data e programação de redes.

Para a animação, foi feita a escolha pela biblioteca gráfica OpenGL (OpenGL Graphics Library) [10]. Esta é uma biblioteca de rotinas gráficas e de modelagem, 2D e 3D, extremamente portável e rápida. Sua maior vantagem é a rapidez no processamento, uma vez que usa algoritmos desenvolvidos e otimizados pela Silicon Graphics, Inc. [14], que é uma das líderes mundiais em Computação Gráfica e em Animação, fato este que influenciou sua escolha. Seu funcionamento é semelhante ao de uma biblioteca C, uma vez que fornece uma série de funcionalidades. As aplicações implementadas utilizando a OpenGL variam de ferramentas CAD a programas de modelagem usados para criar dinossauros para o cinema. Além do desenho de primitivas gráficas, tais como linhas e polígonos, a OpenGL dá suporte a iluminação, ao sombreamento, ao mapeamento de texturas, de transparência, de animação e de muitos outros efeitos especiais. Atualmente, a OpenGL é reconhecida e aceita como um padrão API (Application Programmer's Interface) para desenvolvimento de aplicações gráficas 3D em tempo-real.

Como a OpenGL fornece apenas um pequeno conjunto de primitivas gráficas para construção de modelos geométricos(pontos, linhas e polígonos), utiliza-se a biblioteca GLU (que faz parte da implementação OpenGL) para modelagem de objetos mais complexos, tais como superfícies quádricas, curvas e demais superfícies. A OpenGL não possui funções para gerenciamento de janelas para interação com o usuário ou para arquivos de entrada/saída. Nesse sentido, a solução encontrada foi a utilização da Qt.

### 5.2 Aspectos gerais

O Simulador Gráfico para Robôs Móveis disponibiliza no menu principal as seguintes ações: *simulação*, *parâmetros*, *animação* e *ajuda*. Sua barra de ferramentas padrão apresenta os ícones para atalho de todas as funções referentes ao menu principal. A tela principal contém três grupos de ações: o primeiro indica qual modelo será simulado (*dinâmico* ou *cinemático*); o segundo produz os frames (quadros) da *animação*, enquanto uma *barra de progresso* indica este processamento, e ao final, apresenta a animação resultante, de acordo com uma certa taxa (frames/segundo); e o terceiro permite a visualização dos *gráficos* gerados pela simulação de um dos modelos de controle (Figura 6).

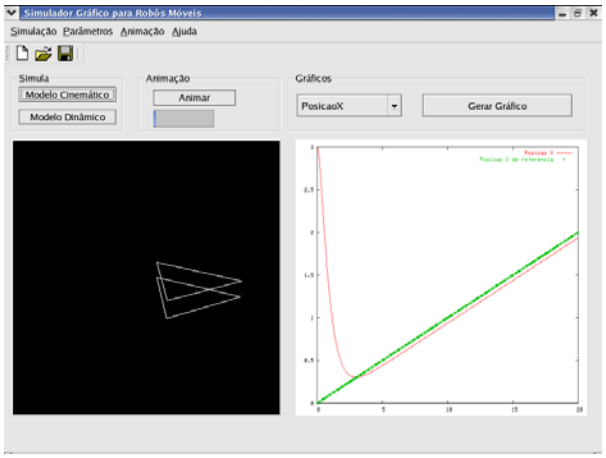


**Figura 6:** Interface principal do simulador.

O *menu Simulação* possui as seguintes funções: *Nova*, *Abrir*, *Salvar*, *Salvar Como*, e *Fechar*, todas estas manipulando os arquivos de simulação gerados pelas ações dos *botões Modelo Cinemático* e *Modelo Dinâmico*. O menu *Sair* fecha o programa.

O *menu Parâmetros* apresenta duas funcionalidades: *Modelo* e *Simulação*. Quando acionados, disparam duas janelas: *Parâmetros do Modelo* e *Parâmetros de Simulação* conforme figuras a seguir:

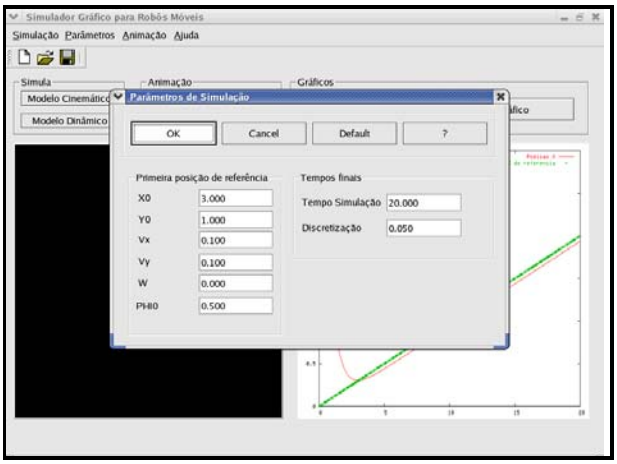


**Figura 7:** Interface Parâmetros da Simulação.

São editados na interface de paraâmetros de simulacao (Figura 7), os parâmetros de simulação do robô: a posição inicial dada pelas coordenadas *X0 e Y0*, as velocidades iniciais *(Vx ,Vy)* em X e Y, a velocidade angular *W* e o ângulo de rotação *PHI0*.

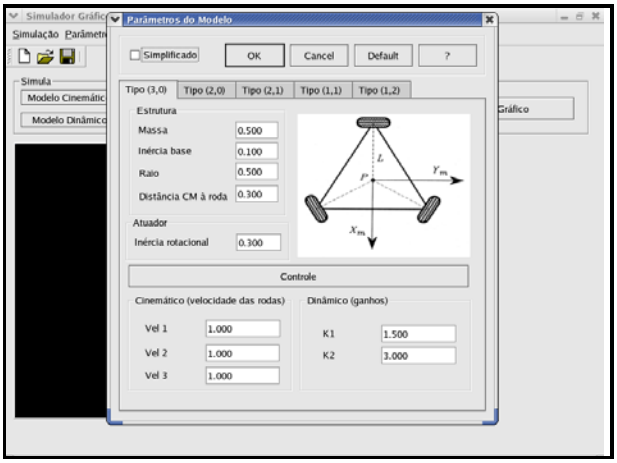


**Figura 8:** Interface Parâmetros do Modelo.

São disponibilizados na interface da Figura 8 cinco classes de robôs móveis a rodas, cada uma contém parâmetros de modelo e de controle próprios. Os parâmetros de modelo para classe (3,0) - omnidirecionais (foco deste projeto) - informados na interface são: *massa, inércia da base, raio e distância CM (centro de massa) à roda*. O parâmetro do atuador informado nesta mesma janela é a *inércia rotacional* do atuador. Os parâmetros de controle informados são as *velocidades (Vel) das rodas 1,2 e 3* para o modelo cinemático e os ganhos *K1 e K2* para o modelo dinâmico. Existe ainda a possibilidade do usuário informar sua própria lei de controle, através do *botão Controle* desta interface.

O *menu animação* possui a mesma ação do *botão Animar* da tela principal do simulador. O *menu ajuda* inclui um manual de utilização do simulador (no momento ainda não está disponível).

### 5.3 Estrutura do Simulador

A seqüência de funcionamento do simulador segue a seguinte ordem:

- Informar os parâmetros do modelo e da simulação (ou vice versa) - presentes no *menu principal*.
- Escolher qual modelo irá animar, cinemático ou dinâmico - presentes na tela pr incipal do grupo *simula*.
- Visualizar os *resultados* em forma de *animação* e/ou *geração de gráficos* (ou vice versa) - presentes na tela principal do simulador.

Primeiramente, o usuário irá definir o modelo do robô que deseja simular, deverá escolher entre as cinco classes de robôs móveis existentes no simulador e deverá informar os parâmetros do modelo e do controle da classe escolhida. Em seguida, deverá informar os parâmetros de simulação do robô que funcionam independente da classe escolhida. Portanto, estes parâmetros podem ser informados antes ou depois dos parâmetros do modelo, pois servem para todas as classes. Caso o usuário não informe os parâmetros do modelo e da simulação antes de simulá-los, o simulador utilizará valores *default* já inseridos na interface.

Após informar os parâmetros do modelo e do controle, o usuário deverá definir o modelo a ser simulado através dos botões do grupo Simula da tela principal. Clicando nos botões de *modelo cinemático* ou de *modelo dinâmico* (grupo simula), será processado os arquivos de simulação *omnidin.dat* (dinâmico) e *omnikin.dat* (cinemático) que conterão os resultados obtidos a partir dos parâmetros informados anteriormente pelo usuário ou dos valores *default* do sistema.

Finalmente, após o arquivo de simulação ter sido gerado através do acionamento dos botões Modelo Cinemático ou Modelo Dinâmico, será possível visualizar os resultados através de gráficos de uma animação. O usuário deve clicar em Animar para visualizar a animação na tela. Para visualizar os gráficos, basta escolher o tipo de gráfico *(posição X, posiçãoY, velocidade, torque, orientação ou trajetória)* que deseja visualizar, e clicar em *Gerar Gráfico*.

## 6 Resultados

Os resultados obtidos com o simulador mostraram-se satisfatórios. Isso comprova que o desenvolvimento dos modelos cinemático e dinâmico da lei de controle implementada estão corretos. Logo, o simulador permite que o usuário possa visualizar os movimentos do robô através da animação (formada pela execução dos frames) e verificar através dos gráficos de posição, velocidade, aceleração, trajetória e orientação se os parâmetros de controle informados no modelo da classe escolhida condizem com o que está sendo mostrado nos gráficos e na animação. Podemos comprovar tais resultados com o estudo de caso a seguir, que simula a validação do modelo dinâmico.

Foi definida uma trajetória padrão para o robô percorrer e devido à análise dos gráficos a seguir, percebe-se que o robô seguiu a trajetória conforme o esperado, validando a implementação do modelo dinâmico de configuração. Os gráficos gerados são os seguintes:

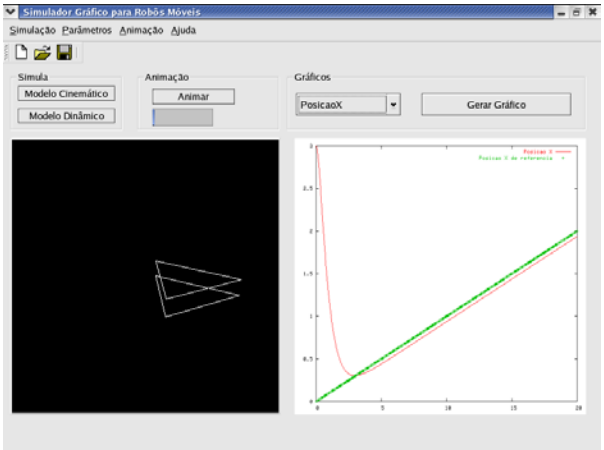


**Figura 9:** Posição X.

Na Figura 9 percebe-se que o robô parte da posição inicial na coordenada X e segue a trajetória definida. Pode-se perceber o mesmo com a Figura 10, no entanto, o robô parte da posição inicial na coordenada Y definida pelo usuário.

Com base na Figuras 11 e 12, percebe-se claramente que o robô partiu das coordenadas de postura iniciais definidas pelo usuário e seguiu a trajetória de referências devido aos ganhos nos torques ajustados pelo usuário.

## 7 Conclusões e proposta para trabalhos futuros

O desenvolvimento desta primeira etapa do simulador mostrou-se de caráter fundamentalmente didático, po-

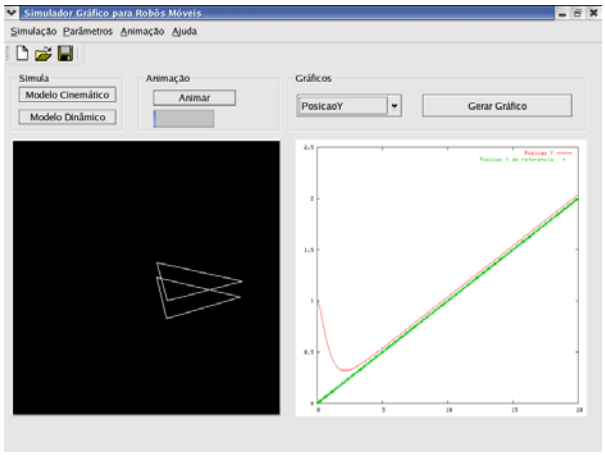


**Figura 10:** Posição Y.

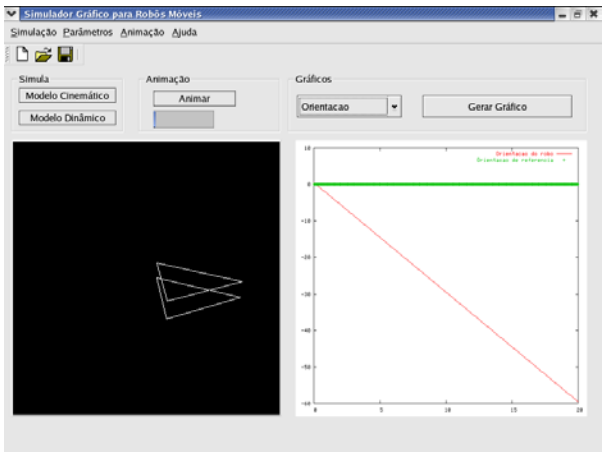


**Figura 11:** Orientação do Robô.

dendo vir a ser objeto de estudo de pesquisadores e estudantes que se interessam por robótica móvel. A implementação em C++ com a biblioteca gráfica Qt foi uma experiência inovadora, satisfatória e de fácil entendimento para posteriores mudanças.

O projeto cumpriu parte de seus objetivos, de maneira didática e fácil, dispôs ao usuário uma interface amigável onde uma vez escolhida a classe do robô, ele pode testar a lei de controle com base nos parâmetros informados, visualizando os resultados de forma clara através de gráficos e animação, suprindo assim, a ausência de softwares deste tipo na área. Possui ainda um código de fácil entendimento, claro o suficiente para que a continuidade se desenvolva de maneira ágil.

O grande ganho do projeto mostrou ser o interfaceamento gráfico para o controle de robôs, já que permite que o usuário utilize a interfce sem ter que possuir um conhecimento prévio dos complexos códigos

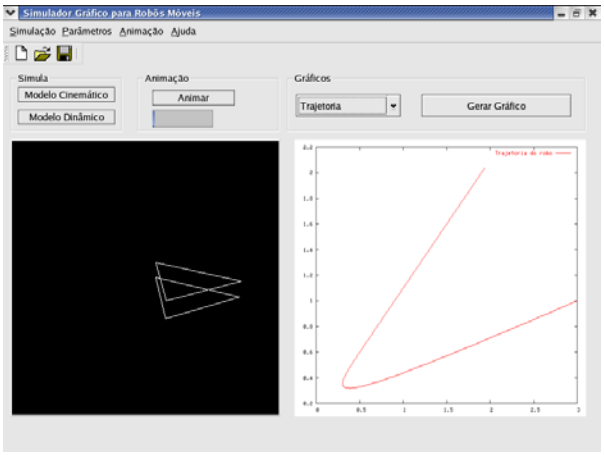


**Figura 12:** Trajetória.

de programa das modelagens de robôs desenvolvidos atualmente. Além disso, permite a análise de resultados gráficos onde se pode fazer um comparativo entre os gráficos gerados automaticamente pelo programa e a animação gerada em paralelo com estes resultados.

Porém, a implementação completa do simulador não foi desenvolvida e algumas funcionalidades ainda não estão implementadas, ficando então como proposta para trabalhos futuros.

Como implementação para trabalhos futuros devem ser desenvolvidos ainda:

- As demais classes de robôs móveis, bem como seus parâmetros de modelo e controle.
- Incorporar o controle desenvolvido pelo usuário para todas as classes.
- Desenvolver o menu de ajuda do simulador.

Espera-se, em breve, disponibilizar na *WEB* o simulador, tanto para execução *online* quanto para *download*, a partir do endereço http://www.ecomp.furg.br/sgrm

## Referências


[1] BotController. http://www.mobotsoft.com/botcontroller.htm

[2] cplusplus.com - The C++ resources network. `http://www.cplusplus.com`.

[3] Cyberbotics. `http://www.cyberbotics.com`.

[4] Furgbol. `http://www.ee.furg.br/furgbol`.

[5] Gnuplot Homepage. `http://www.gnuplot.info/`.

[6] GTK+ The Gimp Toolkit `http://www.gtk.org`.

[7] Juice. `http://www.natew.com/juice`.

[8] K-Team Group. `http://www.k-team.com`.

[9] NuMA - Núcleo de Matematica Aplicada. `http://www.numa.furg.br`.

[10] OpenGL - The Industry Standard for High Performance Graphics. `http://www.opengl.org/`.

[11] Robotic Simulator: RoboWorks. `http://www.newtonium.com/`.

[12] SimRobot - 3D-Robotiksimulator. `http://www.informatik.uni-bremen.de/simrobot`.

[13] Trolltech - Cross-plataform C++ GUI Development. `http://www.trolltech.com`.

[14] Welcome to SGI. `http://www.sgi.com/`.

[15] B. Carter et al *Mechanical Design and Modelling of an Omnidiretional RoboCup Player*. Proceedings RoboCup 2001 INternation Symposium, Seatle - WA, April 2003.

[16] Carlos, Canudas de Wit and Bruno Siciliano and Georges Bastin. *Theory of Robot Control*. Springer, Great Britain, 1997.

[17] Oliveira, V. M. de Oliveira. *Técnicas de Controle de Robôs Móveis, Dissertação de Mestrado*. DAS/UFSC, fevereiro 2001.

[18] William Press and Brian Flannery and Saul Teukolsky and William Vetterling. *Numerical Recipes in C- The Art of Scientific Computing*. Cambridge University Press, 1990.