Aplicação de Taekwondo em Robótica

Dalton M. Tavares[1]

[1]Departamento de Ciência da Computação – Universidade Federal de Goiás (UFG)
CEP: 75704-020 – Avenida Dr. Lamartine Pinto de Avelar, 1120, Setor Universitário
Catalão – GO – Brazil

dalton.tavares@catalao.ufg.br

Abstract. *Taekwondo is a korean martial art that uses hands and feet. The goal of this paper is to demonstrate the application of biomechanics to a Taekwondo practitioner, in order to program its movements in a personal robot. Thus, it would be possible to participate in fighting competitions between robots programmed via the movement mapping interface, a simpler interface compared to the manual insertion of each position, regarding a complex movement such as a kick or punch. Therefore, it was devised a case study intending to list the available hardware and software technologies that allow the operation of the robot.*

Resumo. *O Taekwondo é uma arte marcial coreana que utiliza os pés e as mãos. O objetivo deste artigo é demonstrar a aplicação de biomecânica a um praticante de Taekwondo, com o intuito de programar seus movimentos em um robô pessoal. Com isso, seria possível participar de competições de luta entre robôs programados via interface de mapeamento de movimentos de uma forma mais simples do que a inserção manual de cada posição, considerando um movimento tão complexo como um chute ou soco. Para tanto, foi realizado um estudo de caso com o intuito de elencar a disponibilidade de tecnologias de hardware e software que permitam a operação do robô.*

1. Introdução

Taekwondo é uma arte marcial Coreana e o esporte nacional na Coréia do Sul. Em Coreano, *tae* significa "bater ou quebrar com o pé", *kwon* significa "bater ou quebrar com o punho" e *do* significa "maneira", "método" ou "arte". Assim, Taekwondo pode ser traduzido informalmente como "a arte dos pés e das mãos" ou ainda "a arte de chutar e socar".

Este esporte combina técnicas de combate, auto-defesa, exercício e em alguns casos, meditação e filosofia. O *gyeorugi* ou luta, tem sido um evento olímpico desde o ano 2000 [Hee et al. 1989].

Embora existam diferenças quanto a doutrina e técnica entre os estilos esportivos (e entre as várias organizações), a arte, em geral, enfatiza chutes desferidos dinamicamente a partir de uma posição móvel, empregando o maior alcance e poder das pernas (quando comparado aos braços).

No contexto deste artigo destaca-se uma nova tendência do uso do Taekwondo, como inspiração nos campos de biomecânica e robótica. Biomecânica é o estudo de leis

mecânicas aplicadas ao movimento ou estrutura de organismos vivos[1]. Robótica, é o ramo da tecnologia que lida com o projeto, construção, operação e aplicação de robôs. Desta forma, pretende-se apresentar uma infra-estrutura básica, em termos de hardware e software, para a aplicação de biomecânica aos movimentos de um praticante de Taekwondo, com o intuito de replicá-los em um sistema robótico de computação pessoal.

Nas próximas seções este artigo discute os principais objetivos e metas pretendidos (seção 2), apresenta as demandas em termos de infra-estrutura de software (seção 3) e hardware (seção 4), discute as principais contribuições pretendidas (seção 5) e apresenta as conclusões e trabalhos futuros (seção 6).

2. Objetivos e Metas

O objetivo geral para este artigo é selecionar as tecnologias necessárias para a aplicação de biomecânica aos movimentos de um praticante de Taekwondo. Com isso, espera-se alcançar o mapeamento de movimentos de modo a replicá-los em uma plataforma robótica de computação pessoal. Os resultados desta pesquisa, podem ser aplicados, em um primeiro momento, em uma competição Coreana de Taekwondo para robôs, modalidade iniciada em 2010. Esta competição foi criada pelo *Korea Advanced Institute of Science and Technology* (KAIST), com o apoio da *World Taekwondo Federation* (WTF). Esta competição foi promovida durante o *International Robot Contest* no *Korea International Exhibition Center* (KINTEX), na província de Gyeonggi [Tae-gyu 2010].

Em seu atual formato, os movimentos são pré-programados nos robôs e estes se movem dependendo da resposta de seus sensores embarcados, os quais ajudam a predizer golpes (chutes ou socos) de seus oponentes. A programação de movimentos para o robô pode ser uma tarefa bastante laboriosa e cansativa, caso seja feita utilizando métodos tradicionais, baseados em leitura da posição de motores em uma tentativa de criar o movimento para um determinado golpe (chute ou soco). A participação de um praticante de Taekwondo para a criação e mapeamento dos movimentos utilizados pelo robô representa uma forma de programação mais natural, a qual pode permitir a popularização desta modalidade de competição.

Dessa forma, os objetivos específicos pretendidos podem ser enumerados da seguinte forma:

1. Determinar uma plataforma operacional para o controle de robôs pessoais em termos de hardware e software;
2. Implementação de controle baseado em gestos para o robô escolhido;
3. Produção de estudos de caso para as plataformas escolhidas.

3. Escolha da Plataforma de Software para Desenvolvimento

Existe uma grande quantidade de *frameworks* robóticos disponíveis (e.g. Robot Operating System [ROS.org 2011d] (ROS), Microsoft Robotics Studio [Microsoft 2011b], Webots [Zhang et al. 2011], Orocos [Bruyninckx et al. 2003], ORCA [Kaupp 2007], Player/Stage/Gazebo [Rusu et al. 2006], Aldebaran Coreographe etc), cada um com graus de sucesso e aceitação diferenciados. Pode-se dizer que dentre os *frameworks* existentes, cada um possui seu próprio critério de desenvolvimento/projeto.

[1]Definição do *Oxford English Dictionary*. Disponível em: http://oxforddictionaries.com/ Acesso: 25/08/2011.

Para a seleção de uma plataforma robótica levou-se em consideração os seguintes pré-requisitos:

- Plataforma de código aberto (*open source*);
- Plataforma flexível com relação ao desenvolvimento (multi-linguagem, possibilidade de distribuição de tarefas para balanceamento de carga);
- Possibilidade de uso de projetos de código aberto existentes, visando evitar retrabalho.

Considerando os pré-requisitos apresentados, o *framework* ROS é a solução mais indicada. Os critérios motivadores para o seu desenvolvimento foram definidos por [Quigley et al. 2009] e incluem uma arquitetura distribuída (*peer-to-peer*), independência de linguagem de programação (multi-linguagem com suporte a C++, Python, LISP de forma intercambiável), baseado em ferramentas existentes (em geral de código aberto), seus *drivers* e algoritmos ocorrem em bibliotecas *stand-alone*, as quais não possuem dependência com o ROS. Isso permite o *build* modular (cada módulo comunica-se com os demais via comunicação interprocessos) dentro de sua árvore de código fonte usando o $CMake$, além de ser livre e em código aberto (*open-source*). Uma listagem dos robôs pessoais controláveis por meio do *framework* ROS está disponível em [ROS.org 2011b].

Além do *framework* para controle do robô pessoal, é necessário mapear o movimento do ser humano, no caso, o praticante de Taekwondo. Este mapeamento pode ser usado para ensino de uma série de movimentos complexos (i.e. combinações de chutes e socos), os quais seriam cansativos de serem programados. Este é o caso de *poomsaes*, definidos como um conjunto de movimentos de ataque e defesa contra um oponente imaginário [Kim 1995]. Softwares como o FAAST (*The Flexible Action and Articulated Skeleton Toolkit* [Suma et al. 2011]) e dispositivos como o Microsoft Kinect [Microsoft 2011a] podem ser usados. O FAAST é um *middleware* cujo objetivo é facilitar a integração de controle de corpo inteiro a aplicações de realidade virtual, como é o caso de *video games*, usando sensores de profundidade compatíveis com o OpenNI [OpenNI 2011] (Figura 1). O *framework* OpenNI oferece uma interface para programação de aplicações (*Application Programming Interface* ou API), para a escrita de programas usando interação natural. Esta API permite a comunicação entre dispositivos de baixo nível (e.g. sensores de áudio e vídeo), assim como soluções de *middleware* (e.g. para rastreamento visual usando visão computacional).

4. Escolha do Hardware Pretendido para o Projeto

Considerando os objetivos propostos, os robôs mais indicados e suportados pelo *framework* ROS são o robô NAO da Aldeberan Robotics [Aldebaran Robotics 2011] e o robô Kondo KHR-3HV [RobotShop 2011].

O robô NAO, em sua versão mais completa, possui 25 graus de liberdade, conferindo uma precisão de movimentos suficiente para a execução de *poomsaes* ou *gyeorugi*. Exibições como a realizada na Shanghai 2010 world expo[2] demonstram o equilíbrio potencial deste robô pessoal para a execução de movimentos com os braços e pernas. Além disso, a existência de mãos articuladas permitiriam uma maior precisão na execução de movimentos com as mãos (abertas ou fechadas).

[2]Disponível em http://www.robotshop.com/blog/nao-performs-at-shanghai-682 Acesso: 16/08/2011.

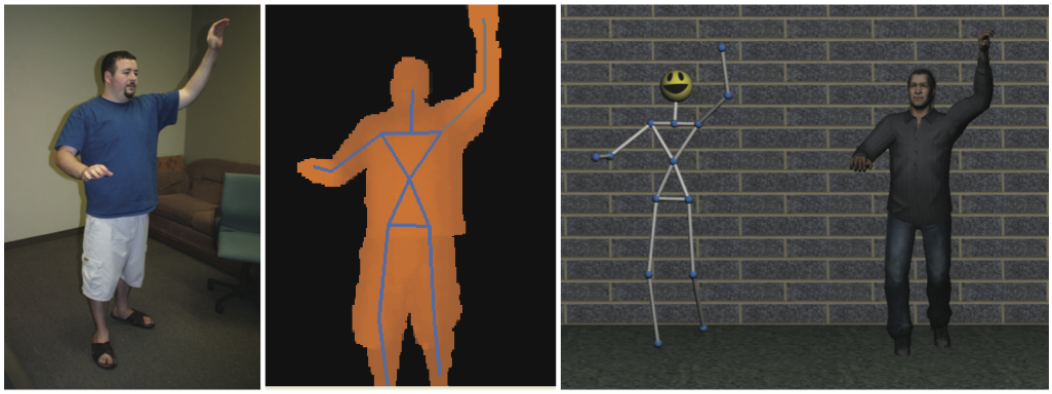

Figura 1. (a) Exemplo de usuário fazendo uma pose. (b) Software capaz de identificar e segmentar o usuário sem a imagem de fundo e inserir um esqueleto articulado à nuvem de pontos resultante. (c) Usando o FAAST, o usuário está apto a controlar um *avatar* e um personagem virtual animado em tempo real [Suma et al. 2011] pg 248.

O robô Kondo KHR-3HV possui 17 graus de liberdade com 5 servo motores opcionais, podendo totalizar 22 graus de liberdade. Comparado ao NAO, este robô não possui mãos articuladas, o que impossibilita a execução precisa de movimentos com as mãos em *poomsaes*. Além de seu software nativo, o robô Kondo pode ser controlado usando a extensão Veltrop ROS [ROS.org 2011c].

Estes sistemas geralmente operam de maneira a aceitar um programa a ser executado na plataforma alvo (no caso, o robô pessoal) e a partir deste ponto, eles executam limitados ao hardware disponível. Dependendo do hardware este procedimento pode impor limitações maiores ou menores. No caso do Kondo KHR-3HV, foi desenvolvida uma "mochila" ou *backpack*[3] contendo um processador Samsung de 32-bits, 200MHz baseado em ARM executando Linux. Este *backpack* mede 49,5 x 52 x 12mm e permite que se executem softwares adicionais, como por exemplo, o software para processamento de imagem na plataforma do robô.

Considerando o acesso a essas plataformas para a presente pesquisa, este mostra-se proibitivo. O custo do robô NAO gira em torno de dezesseis mil dólares, enquanto o robô Kondo é orçado em torno de dois mil dólares (sem o *backpack* de extensão). Pretende-se adquirir o robô NAO no contexto de outro projeto de pesquisa [Tavares 2011]; o que pode viabilizar, em um momento futuro, sua utilização no contexto desta pesquisa.

Dessa forma, neste primeiro momento, optou-se pelo uso de uma plataforma cujo custo benefício é mais acessível (em torno de mil reais) e flexível o suficiente para a geração dos primeiros resultados. A plataforma em questão é o robô pessoal LEGO® MINDSTORMS® NXT 2 com seu processador Atmel® de 32-bits ARM® (AT91SAM7S256), 256 KB de memória FLASH, 64 KB de memória RAM e 48 MHz.

Essa é uma configuração bastante limitada considerando as exigências de processamento de sensores e o processamento adicional imposto por softwares que pretendam conferir habilidades adicionais ao robô, como é o caso do reconhecimento de movimento

[3]http://www.plasticpals.com/?p=21309 Acesso: 03/08/2011.

pretendido para a execução/gravação de *poomsaes* e a realização de luta com o robô. Apesar de suas limitações, essa plataforma é flexível o suficiente, graças a disponibilidade de vários sensores presentes no kit padrão 8547.

Além disso, graças a possibilidade do uso do *framework* ROS com o robô LEGO® MINDSTORMS® NXT 2, é possível realizar o processamento adicional exigido por aplicações complexas em um computador PC ou *notebook*, o que possibilita a extensão de suas funcionalidades.

5. Principais Contribuições

Considerando as características do ROS (seção 1) e os critérios iniciais para o desenvolvimento de um sistema robótico aplicável ao Taekwondo nas modalidades de *gyeorugi* e *poomsae*, foi estabelecido um primeiro estudo de caso para verificar as potencialidades de robôs pessoais, utilizando o *framework* ROS. Além disso, optou-se, em termos de hardware, pelo uso da plataforma LEGO® MINDSTORMS® NXT 2 devido ao seu baixo custo e facilidade de reconfiguração dependendo da missão pretendida.

Para os testes experimentais realizados, foram utilizados os sensores padrão do kit LEGO® MINDSTORMS® NXT 2 (Figura 2). A plataforma de teste foi criada usando o sistema operacional (SO) Ubuntu 11.04, instalado em uma máquina virtual VMWare fusion em um Apple Macbook. Este ajuste inicial foi preferido com o intuito de inserir sobrecargas ao sistema e verificar a possível capacidade de tempo real do *framework* ROS.

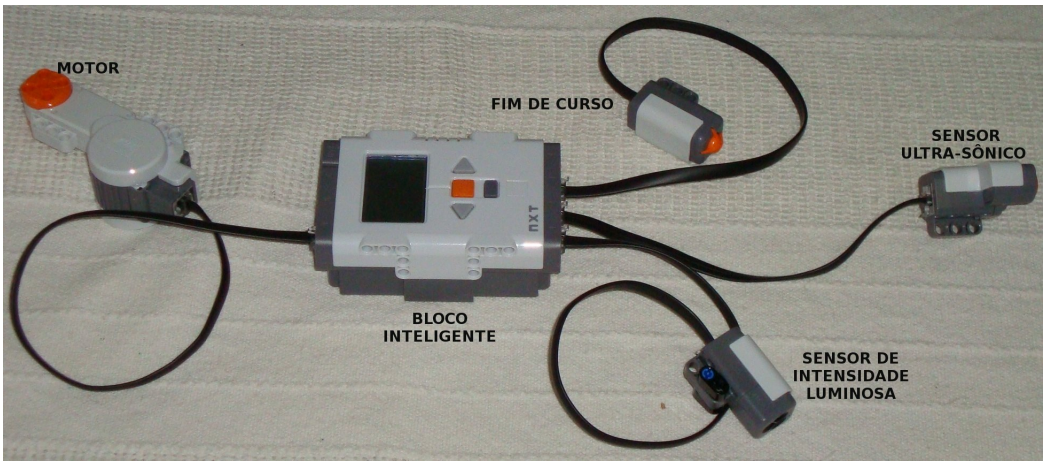

Figura 2. Bloco inteligentes e sensores do robô pessoal LEGO® MINDSTORMS® NXT 2.

Embora um cenário NXT-ROS possa ser executado com sensores lidos a uma frequência de 5-20 Hz [ROS.org 2011a], com esta configuração as leituras não puderam ocorrer em frequências superiores a 2 Hz. Outro ponto de sobrecarga foi o uso de comunicação bluetooth ao invés do uso de um cabo USB.

Para este primeiro estudo de caso, foi realizado o processo de instalação do *framework* robótico ROS, a criação de arquivos de configuração e o uso de um modelo usando o software *Lego Digital Designer*[4] (LDD). As leituras do sensor foram realizadas

[4]Software disponível gratuitamente em: http://ldd.lego.com/ Acesso: 04/08/2011.

usando a aplicação $rostopic$ e testada para cada sensor individualmente. Também foi realizada uma configuração mais complexa incluindo um motor, um sensor de fim de curso, um sensor ultrasônico e um sensor de intensidade luminosa. O esqueleto do sistema foi apresentado na Figura 2. Para a leitura individual, o ajuste de frequencia a 10 Hz funcionou bem. Usando os quatro dispositivos simultaneamente, foi necessário diminuir a frequência para que os sensores operassem adequadamente.

Considerando o modelo fornecido em [ROS.org 2011a] para o LDD, foi possível testar a importação deste para o ROS via o uso do script de conversão $lxf2urdf.py$[5]. Após a conversão, foi executada a aplicação $rviz$, a qual renderiza o modelo convertido e o conecta ao robô real. É possível visualizar o robô renderizado e gerenciá-lo por meio desta interface. Infelizmente, alguns problemas com relação ao mapeamento dos sensores ocorreram, posicionando-os de maneira incorreta (i.e. voltados para a origem, de frente para o bloco inteligente NXT). Este é um problema bem conhecido, porém ainda mal documentado. Todavia, foi possível conectar os sensores a simulação, por meio da alteração manual do nome do tópico na interface. O resultado da simulação até o momento é apresentado na Figura 3.

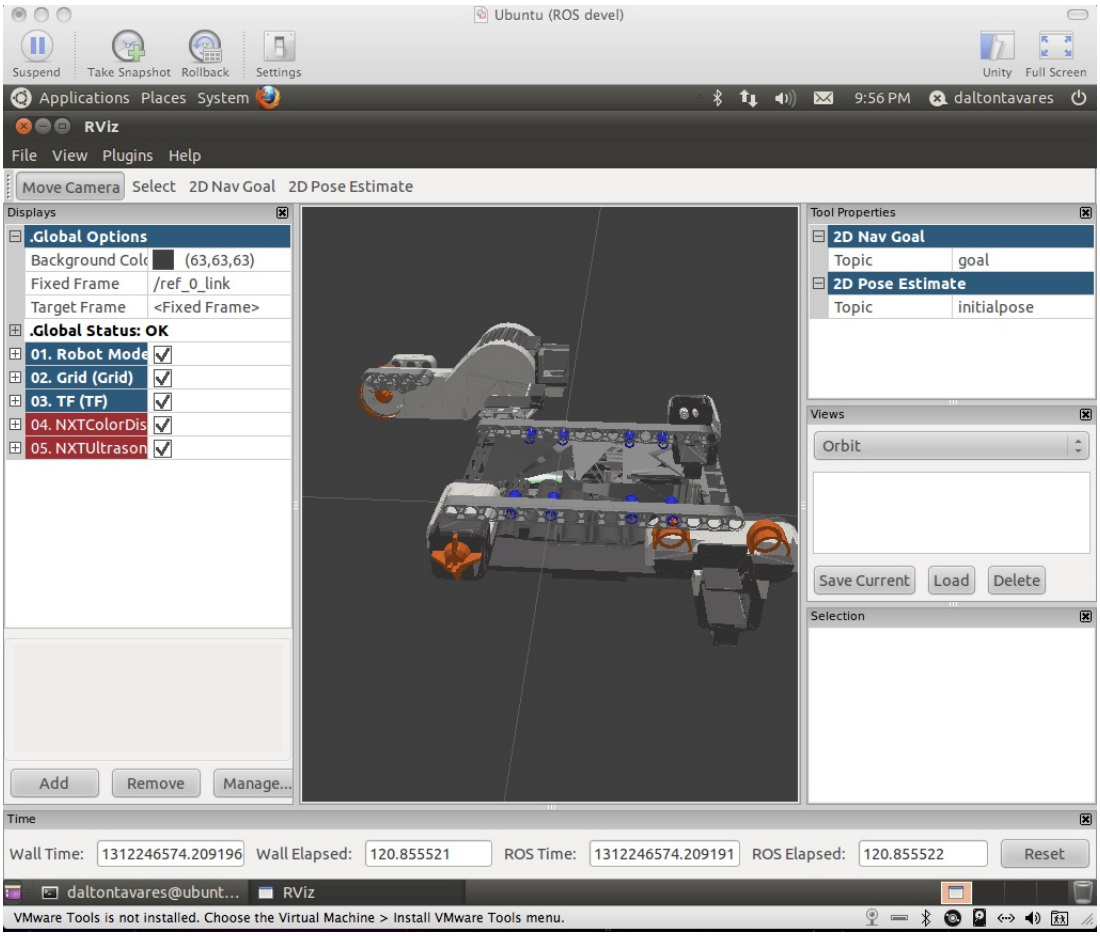

Figura 3. Interface *rviz* com erros conhecidos.

[5]http://www.ros.org/wiki/nxt_lxf2urdf/Tutorials/Creating%20a%20simple-%20robot%20model%20using%20lxf2urdf.py Acesso: 04/08/2011.

6. Conclusão e Proposta de Trabalhos Futuros

6.1. Conclusão

O objetivo deste artigo foi propor o uso de uma interface de programação mais simples considerando uma nova modalidade de competições em Taekwondo, envolvendo robôs pessoais. Com isso, espera-se reduzir as exigências em termos de conhecimentos computacionais para o praticante de Taekwondo, que neste contexto, se tornará o "programador" para os movimentos do robô pessoal.

Para tanto, foi proposta uma primeira aplicação envolvendo o mapeamento de movimentos de um praticante de Taekwondo e a posterior replicação destes movimentos por um avatar robótico. Como os movimentos em questão são mais complexos com relação a velocidade e execução dinâmica, apenas a execução destes movimentos pelo robô pessoal, dadas as suas limitações, já constitui um problema de pesquisa complexo. A execução destes movimentos pode ser realizada durante práticas de *gyeorugi* entre robôs operando de forma semi-autônoma, ou após a programação de movimentos, como no caso de aprendizado de *poomsaes* pelos robôs pessoais.

Nesse contexto, o *framework* ROS permite a possibilidade de agregação de funções em um computador externo, as quais, dependendo da plataforma de robótica pessoal, não poderiam ser migradas internamente para o hardware hospedeiro (e.g. o bloco inteligente NXT).

6.2. Trabalhos Futuros

Alguns desafios podem ser enumerados considerando o que foi proposto durante este artigo:

- Estudo dos movimentos complexos do Taekwondo e execução dos mesmos em uma plataforma de robótica pessoal;
- Usar a tecnologia FAAST e criar uma ponte para o controle e tradução das informações de posição capturadas pelo sensor de profundidade, no caso, o Microsoft Kinect, para o sistema de coordenadas do robô pessoal.
- Desenvolvimento de inteligência aplicada aos robôs pessoais, como forma de desenvolvimento tecnológico e teste para os atletas, similar às competições de xadrez entre humanos e robôs;
- Possibilidade de utilização do robô NAO da Aldebaran Robotics, empresa francesa de sucesso, cuja plataforma é usada como padrão atualmente para a RoboCup[6]. Este robô foi solicitado no contexto de outro projeto submetido ao CNPq [Tavares 2011]. Em caso de aprovação do projeto, pretende-se utilizar a plataforma NAO também para esta pesquisa.

Referências

Aldebaran Robotics (2011). NAO. http://www.aldebaran-robotics.com/ Acesso: 25/08/2011.

Bruyninckx, H., Soetens, P., and Koninckx, B. (2003). The real-time motion control core of the orocos project. In *Proceedings of the 2003 IEEE International Conference on Robotics & Automation*, pages 14–19, Taipei, Taiwan.

[6]http://www.robocup.org/robocup-soccer/standard-platform/ Acesso: 03/08/2011.

Hee, P. Y., Hwan, P. Y., and Gerrard, J. (1989). *Tae Kwon Do: The Ultimate Reference Guide to the World's Most Popular Martial Art*. Number 978-0816038398. Checkmark Books.

Kaupp, T. (2007). Orca robotics. On-line. Acesso em: 01/08/2011.

Kim, Y. J. (1995). *Arte Marcial Coreana Tae Kwon Do*, volume 1. 1 edition.

Microsoft (2011a). Apresentando Kinect para Xbox 360. http://www.xbox.com/pt-br/kinect Acesso: 25/08/2011.

Microsoft (2011b). Microsoft robotics. http://www.microsoft.com/robotics/ Acesso: 25/08/2011.

OpenNI (2011). Introducing OpenNI. http://www.openni.org/ Acesso: 25/08/2011.

Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, page 288.

RobotShop (2011). Kondo KHR-3HV Humanoid Robot Kit. http://www.robotshop.com/kondo-khr-3hv-humanoid-robot-kit-1.html Acesso: 25/08/2011.

ROS.org (2011a). Getting started with NXT-ROS. http://www.ros.org/wiki/nxt_ros/Tutorials/Getting%20started Acesso: 25/08/2011.

ROS.org (2011b). Robots Using ROS. http://www.ros.org/wiki/Robots Acesso: 25/08/2011.

ROS.org (2011c). ROS Repository for Roboard and Humanoids Now Available. http://www.ros.org/news/2010/10/ros-repository-for-roboard-and-humanoids-now-available.html Acesso: 25/08/2011.

ROS.org (2011d). ROS.org. http://www.ros.org/ Acesso: 25/08/2011.

Rusu, R., Robotin, R., Lazea, G., and Marcu, C. (2006). Towards open architectures for mobile robots: Zeero. In *Automation, Quality and Testing, Robotics, 2006 IEEE International Conference on*, volume 2, pages 260 –265.

Suma, E. A., Lange, B., Rizzo, A. S., Krum, D. M., and Bolas, M. (2011). FAAST: The Flexible Action and Articulated Skeleton Toolkit. In *Virtual Reality 2011*. IEEE.

Tae-gyu, K. (2010). Taekwondo robots to duke it out. http://www.koreatimes.co.kr/www/news/include/print.asp?newsIdx=67631 Acesso: 25/08/2011.

Tavares, D. M. (2011). Aplicação de Situation Awareness em robôs Pessoais (A.S.A.P). Edital universal - cnpq no 14/2011, Universidade Federal de Goiás, campus Catalão (UFG-CAC).

Zhang, X.-l., Weng, X.-d., and Yang, Z. (2011). Mobile robot simulation based on webots. In *Natural Computation (ICNC), 2011 Seventh International Conference on*, volume 3, pages 1750 –1752.