Universidade Federal do Rio de Janeiro

Escola Politécnica/COPPE

Sistema Web para controle e simulação de braços robóticos

Autor:

Igor do Valle Campbell

Orientador:

Prof. Ricardo Guerra Marroquim, D. Sc.

Examinador:

Prof. Claudio Esperança, Ph. D.

Examinador:

Prof. José Ferreira de Rezende, Ph. D.

Poli/COPPE

Abril de 2013

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
Escola Politécnica - Departamento de Eletrônica e de Computação
Centro de Tecnologia, bloco H, sala H-217, Cidade Universitária
Rio de Janeiro - RJ CEP 21949-900

Campbell, Igor do Valle

Sistema Web para controle e simulação de braços robóticos / Igor do Valle Campbell – Rio de Janeiro: UFRJ/POLI-COPPE, 2013.

xii, 50 p.: il.; 29,7 cm.

Orientador: Ricardo Guerra Marroquim

Projeto de Graduação – UFRJ/Escola Politécnica/Departamento de Eletrônica e de Computação – COPPE, 2013.

Referências Bibliográficas: p. 49-50.

1. Braços Robóticos. 2. Serviços Web. 3. WebGL. I. Marroquim, Ricardo Guerra. II. Universidade Federal do Rio de Janeiro, Poli/COPPE. III. Título.

DEDICATÓRIA

Dedico este trabalho ao meu pai, Carlos Henrique Gonçalves Campbell, exemplo de profissional que sempre me apoiou e despertou meu interesse pela engenharia.

AGRADECIMENTO

Gostaria de agradecer a todos que contribuíram para a realização deste trabalho. A minha família, aos meus amigos e aos professores, em especial ao professor Ricardo Marroquim, que me orientou neste trabalho e aos professores Claudio Esperança e José Ferreira de Rezende pela presença na banca examinadora.

RESUMO

Cada vez mais se fazem necessárias soluções virtuais que ofereçam conforto, praticidade e agilidade ao cliente. Na utilização de sistemas de controle de dispositivos à distância é difícil perceber, através de imagens de uma câmera, o espaço no qual o dispositivo se movimenta. Em função disso, alguns sistemas utilizam a simulação e visualização tridimensional para representar o dispositivo controlado, embora esta omita informações importantes do ambiente real obtidas pela câmera. A combinação de imagens de câmera e um ambiente virtual permite ao usuário entender melhor o espaço no qual o dispositivo se movimenta sem perder as informações do ambiente real. Contudo, a combinação das duas visualizações não é trivial.

O presente trabalho tem como objetivo desenvolver um sistema para controlar e visualizar, através de imagens de câmera e simulações em três dimensões, um braço robótico à distância. Através deste sistema, o usuário tem a capacidade de perceber com muito mais facilidade os elementos que está controlando, tornando fácil o treinamento dos usuários e aumentando a precisão na utilização do sistema.

O projeto foi desenvolvido através de três serviços Web que se comunicam e permitem o acesso do usuário através da Internet, utilizando apenas um navegador.

Palavras-Chave: braços robóticos, serviços Web, WebGL.

ABSTRACT

Virtual solutions have been increasingly used in order to provide comfort, practicity and agility to the customer. When using remote control systems for devices its hard to perceive, through camera frames, the space in which the device is embedded. As a result, some systems use tridimensional visualization and simulation to represent the controlled device, although this omits important information of the real environment obtained by the camera. The combination of the camera images and the virtual ambient allows the user to better understand the space in which the device acts without losing the information of the real environment. However, the combination of the two views is not trivial.

This work aims to develop a system to remotely control and visualize a robotic arm through camera frames and tridimensional simulation. Through this system, the user can understand more easily the elements being controlled, being useful for training sections as well as increasing the accuracy of the system.

This project was developed through three Web services that communicate with each other and allow user access via the Internet, using only a Web browser.

Key-words: robotic arms, Web services, WebGL.

SIGLAS

USB - *Universal Serial Bus*

COLLADA - *COLLaborative Design Activity*

WebGL - *Web Graphics Library*

CC - Circuito de Controle

MC - Módulo de Controle

MA - Módulo de Acesso

MV - Módulo de Visualização

REST - *REpresentational State Transfer*

SOAP - *Simple Object Access Protocol*

HTML - *HyperText Markup Language*

CSS - *Cascading Style Sheet*

AJAX - *Asynchronous Javascript And Xml*

XML - *Exchange Markup Language*

HTTP - *HyperText Transfer Protocol*

JSON - *Javascript Simple Object Notation*

JPEG - *Joint Photographic Experts Group*

WSDL - *Web Service Description Language*

WADL - *Web Application Description Language*

W3C - *World Wide Web Consortium*

OpenGL ES - *Open Graphics Library for Embedded Systems*

GPU - *Graphics Processing Unit*

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Tema

Este trabalho apresenta um software para controle e visualização de sistemas robóticos à distância. Com o objetivo de ser simples de utilizar, o projeto foi desenvolvido como um aplicativo Web, possibilitando a criação de uma interface de usuários bastante intuitiva e a utilização do sistema sem a necessidade de instalação de qualquer software, bastando acessar o endereço do sistema através de um navegador de Internet. O projeto também visa a fácil criação de novas funcionalidades e integração com outros sistemas, por isso foi desenvolvido através de serviços Web, de forma que as novas funcionalidades estejam em novos aplicativos que se comunicarão com os serviços Web do projeto.

1.2 Delimitação

Este trabalho é direcionado a profissionais da área de robótica e computação que necessitam de sistemas mais flexíveis e de fácil utilização para o controle de robôs, capazes de funcionar através da Internet.

1.3 Justificativa

A utilização de robôs nas indústrias proporciona grandes vantagens nas linhas de produção, reduzindo custos, aumentando a precisão na fabricação de peças e

aumentando a velocidade de produção. As vantagens são facilmente identificadas, porém o investimento necessário para a compra e instalação deles é uma barreira para a utilização em pequenas empresas ou em casos de prototipação de produtos ou produção de peças para provas de conceitos. Nestes casos, o aluguel de robôs poderia ser uma solução, mas os custos de transporte e instalação podem inviabilizar o aluguel, dependendo do tamanho dos robôs.

Uma alternativa interessante seria a instalação de vários robôs no mesmo local para que eles sejam controlados por usuários à distância, permitindo a montagem de uma linha de produção remota, capaz de construir produtos completos. Neste modelo de linha de produção remota, as horas de funcionamento de cada robô poderiam ser contratadas pelos usuários. Uma pequena empresa, por exemplo, poderia alugar robôs por um determinado período para fabricar seus produtos e requisitar a entrega dos produtos diretamente para seus clientes. Neste modelo, os custos de compra, transporte e instalação dos robôs ficam a cargo do dono da "fazenda de robôs". Este trabalho apresenta um sistema de controle de robôs que permite que este modelo de linha de produção remota seja utilizado.

1.4 Objetivos

O objetivo deste trabalho é o desenvolvimento de um sistema de controle de robôs que permita ao usuário comandar um robô, utilizando um navegador de Internet. O sistema possibilita a visualização através de imagens de uma câmera e de uma representação virtual do robô para que o usuário possa ter uma melhor percepção da movimentação do robô.

É importante que o usuário do sistema possa acessá-lo sem restrições de local e com o máximo de compatibilidade com os diferentes tipos de dispositivos, de navegadores de Internet e de sistemas operacionais.

1.5 Metodologia

Inicialmente foi projetado e desenvolvido um circuito eletrônico e um software simples para a comunicação entre o computador e o circuito através do protocolo USB 2.0. Após entender e configurar corretamente a comunicação entre as duas

partes, o software foi modificado e outro circuito foi desenvolvido para movimentar motores quando o circuito recebesse mensagens do computador.

Em seguida foi desenvolvido outro software para obter e interpretar imagens de webcams. Após a obtenção correta das imagens, foram desenvolvidos os recursos de calibração e de reconhecimento de posição de câmeras.

O software de comunicação com o circuito eletrônico e o software de leitura da câmera foram unidos formando a base do módulo de controle. A partir desta base, foi desenvolvido um sistema web utilizando a biblioteca Django, finalizando o módulo de controle.

Com o módulo de controle pronto, o módulo de acesso foi projetado e desenvolvido e a comunicação entre os dois módulos foi estabelecida.

Foi desenvolvida uma biblioteca para auxiliar a criação de serviços Web, facilitando a comunicação entre os módulos do sistema e a criação de extensões com novas funcionalidades.

Para o desenvolvimento do módulo de visualização foi feito um projeto inicial utilizando a biblioteca Django e a tecnologia WebGL. Com este projeto inicial foram realizados testes sobre os recursos da tecnologia WebGL. Em seguida foi projetado um formato de arquivos de modelos tridimensionais e foi desenvolvido um conversor de arquivos COLLADA para este novo formato. Com o conversor finalizado, foi desenvolvido um leitor para o novo formato para que um arquivo COLLADA pudesse ser convertido e interpretado pelo módulo de visualização e então desenhado no navegador do usuário. A comunicação entre o módulo de visualização e o módulo de acesso foi então desenvolvida e o módulo de visualização foi modificado para exibir as imagens da câmera do robô junto com a cena do robô virtual.

Com os três módulos funcionais, as interfaces de usuário foram reprojetadas e ajustadas e as comunicações entre os módulos foram novamente ajustadas.

1.6 Organização do Texto

O capítulo 2 abordará a base de sistemas robóticos e a forma pela qual o sistema de controle de robôs se comunica com os sistemas robóticos.

A arquitetura do sistema e o detalhamento de funcionalidades dos três sub-

sistemas do projeto serão apresentados no capítulo 3.

No capítulo 4 as tecnologias Web utilizadas no projeto serão descritas, assim como a forma de comunicação com cada um dos subsistemas do projeto.

O capítulo 5 apresenta as formas de visualização dos robôs, mostrando a forma de visualização das imagens das câmeras e a tecnologia utilizada para desenhar a representação virtual dos robôs.

No capítulo 6 são feitos testes de compatibilidades de navegadores de Internet com o projeto.

No capítulo 7 é feita a conclusão sobre o projeto desenvolvido e são apresentadas algumas possibilidades de melhorias para o sistema.

Capítulo 2

Sistemas Robóticos

2.1 O que é um robô?

Existem diversas definições do que é um robô. De acordo com a enciclopédia Britannica [2] um robô é "qualquer máquina operada automaticamente que substitui o esforço humano, mas pode não se assemelhar aos seres humanos em aparência ou na maneira de realizar as funções" ("Any automatically operated machine that replaces human effort, though it may not resemble human beings in appearance or perform functions in a humanlike manner"). Pela definição do dicionário Priberam [14], um robô é um "aparelho capaz de agir de maneira automática numa dada função". As definições apresentadas utilizam a ideia de funcionamento automático, porém há muitas divergências a respeito do que pode ser aceito como funcionamento automático. Portanto, não há uma definição comum de quais máquinas podem ser consideradas robôs, mas é aceito que um robô pode realizar algumas ou todas as seguintes tarefas: se mover, operar um membro mecânico, sentir e manipular o ambiente e exibir comportamentos inteligentes e/ou sociais que imitam humanos ou outros animais [4].

Uma classificação primária dos robôs de acordo com as necessidades e benefícios para as pessoas separa os tipos de robôs em 2 grupos, o grupo de robôs assistentes e o grupo de robôs de estímulos interativos. Os robôs do primeiro grupo possuem aparência de máquinas e executam movimentos básicos para prover atividades motoras ou sensoriais. Este grupo possui robôs industriais, para pesquisa, militares e de salvamento, médicos e de serviços domésticos. Os robôs do segundo

grupo possuem aparência que lembra seres vivos ou objetos animados e executam ações que se assemelham com expressões faciais ou ações com significado social ou emocional. Este grupo possui robôs sociais, para recreação, educativos, para reabilitação e com potencial terapêutico [10].

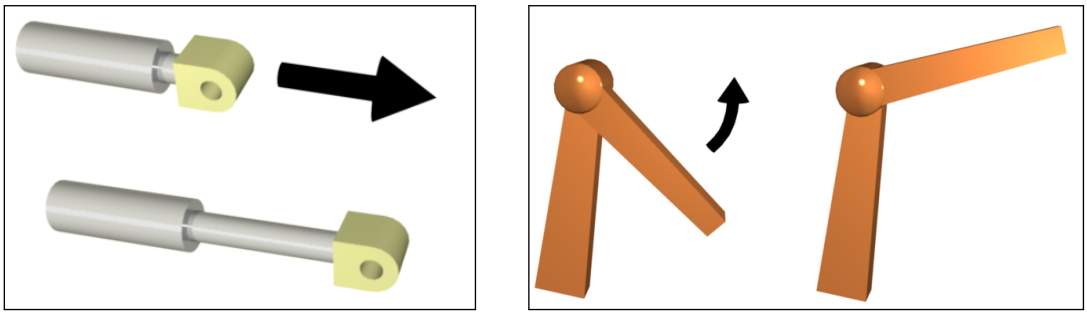
Os robôs podem ainda ser classificados de acordo com sua forma de operação, podendo ser autônomos, pré-programados ou controlados remotamente. Robôs autônomos são aqueles que executam tarefas sem o auxílio de um operador. Robôs pré-programados podem ser definidos como aqueles que executam uma sequência de ações, podendo executar sequências alternativas de ações de acordo com condições do ambiente, sendo a sequência principal e as alternativas definidas pelo operador. Os robôs controlados remotamente não possuem nenhuma sequência de ações pré-definida, executando as ações que o operador deseja em tempo real.

Os robôs a serem utilizados neste projeto enquadram-se no grupo de assistentes e são controlados remotamente. Porém, a estrutura extensível do projeto permite que sejam desenvolvidos módulos que armazenam sequências de ações ou que utilizam, por exemplo, elementos de inteligência artificial para controlar os robôs, formando sistemas que podem ser caracterizados como pré-programados ou autônomos.

2.2 Composição dos robôs

Para se mover ou operar membros mecânicos, os robôs devem utilizar elementos mecânicos. Os elementos mecânicos permitem que os robôs realizem transformações geométricas de suas partes, sendo as transformações mais comuns a translação e a rotação. A translação é descrita por uma movimentação das partes mecânicas em linha reta, como mostra a figura 2.1a. A rotação é descrita por movimentos circulares em torno de um eixo, como mostra a figura 2.1b.

Para a movimentação são utilizados os atuadores, elementos que realizam forças para que os elementos mecânicos se movimentem e alcancem as posições desejadas. Os atuadores podem ser pneumáticos, que utilizam a pressão do ar para gerar movimento, hidráulicos, que utilizam a pressão da água ou de óleos para gerar movimento, ou elétricos, que utilizam as forças eletromagnéticas para gerar



(a) Translação (b) Rotação

Figura 2.1: Transformações geométricas

movimento.

Os atuadores necessitam de um sistema de controle para que sejam ativados nos momentos certos e para que se movimentem no sentido correto e com a força ideal. O sistema de controle utiliza circuitos eletrônicos que geram sinais elétricos para ativar os compressores dos atuadores pneumáticos e hidráulicos ou para gerar os campos eletromagnéticos dos atuadores elétricos.

Para que os sistemas de controle identifiquem a posição que as estruturas mecânicas se encontram, são utilizados os sensores ou os temporizadores. Os temporizadores podem ser projetados com base no tempo de movimentação dos atuadores, de forma que o atuador se movimente somente durante o tempo necessário para que a posição desejada seja alcançada. Os sensores captam informações do ambiente e traduzem em impulsos elétricos, para serem interpretadas pelo sistema de controle.

Os sensores são utilizados para descobrir a posição dos elementos mecânicos. Eles podem captar informações de proximidade (chaves de toque, sensores indutivos, sensores capacitivos, sensores de ultrassom, etc), de rotação (tacógrafo, giroscópio, encoder), e muitas outras informações que permitem a localização dos elementos mecânicos do sistema robótico no espaço. Os sensores-limite são capazes de indicar se a estrutura mecânica está ou não em uma posição pré-definida, enquanto os sensores de curso são capazes de precisar a posição que o elemento mecânico se encontra em qualquer momento.

2.3 Sistema de Controle

As informações captadas pelos sensores possuem naturezas diversas e cada sensor pode traduzi-las em impulsos elétricos de maneiras diferentes. Os atuadores utilizados também podem ser controlados de diversas maneiras. As especificidades de cada sensor e atuador devem ser consideradas no projeto do sistema de controle do robô.

O sistema de controle utilizado neste projeto é dividido em duas partes, o circuito de controle e o módulo de controle. O circuito de controle é um circuito eletrônico que se conecta ao computador através do protocolo USB, interpreta os comandos recebidos pelo computador e transforma em impulsos elétricos para movimentar os atuadores do robô. O circuito de controle deve ser projetado para cada modelo de robô.

O módulo de controle é um software que identifica os circuitos de controle conectados no computador e envia comandos para este circuitos para movimentar os robôs. A comunicação entre o módulo de controle e o circuito de controle utiliza um protocolo de comunicação desenvolvido especificamente para este projeto.

2.3.1 Protocolo de comunicação de controle

A versão 2.0 do protocolo USB (Universal Serial Bus) possui quatro tipos de transferência de dados, isócrono, utilizado para transmissão ininterrupta de dados e sem verificação de erros; em massa, utilizado para a transferência de grandes blocos de dados; controle, utilizado para a transferência de informações de parâmetros de controle dos dispositivos; e interrupção, que possui verificação de erros, velocidade de transmissão garantida e é utilizado para pequenas quantidades de dados.

As comunicações realizadas entre módulo de controle e o circuito de controle utilizam a transferência de dados do tipo interrupção nos dois sentidos da comunicação. Os pacotes enviados pelo circuito de controle ou pelo módulo de controle possuem 4 bytes e seguem o formato apresentado na figura 2.2.

O byte 1 do pacote de dados representa a função que a mensagem deve desempenhar. Os valores possíveis para o byte 1 são apresentados na tabela 2.1. Os bytes 2, 3 e 4 contém os dados utilizados para cada uma das funções. O byte 4

Figura 2.2: Formato do pacote de dados

não é utilizado por nenhuma das funções listadas, mas foi reservado para permitir funcionalidades adicionais.

Valor (em hexadecimal)	**Função**
0x01	Ligar atuadores
0x02	Desligar atuadores
0x03	Obter atuadores ligados
0x30 até 0x37	Obter posição de um atuador
0x40 até 0x47	Mover um atuador

Tabela 2.1: Funções das mensagens

As funções das mensagens permitem que até oito atuadores sejam controlados. Cada uma das funções apresenta um conjunto de dados específico:

Ligar atuadores Esta função faz com que o circuito de controle ative alguns dos atuadores. Os atuadores que devem ser ligados são informados no byte 2 do pacote e os bytes 3 e 4 não são utilizados. Cada bit do byte 2 representa um dos atuadores, portanto, quando o bit que representa um atuador estiver ativo (valor lógico 1), o atuador correspondente será ativado, como mostra a figura 2.3. Desta forma, se o valor de byte 2 for 10100001, os atuadores 7, 5 e 0 serão ativados. Para a execução desta função, a mensagem é enviada apenas no sentido do módulo de controle para o circuito de controle.

Desligar atuadores Esta função faz com que o circuito de controle desative alguns dos atuadores. O formato desta função é semelhante à função de ligar atuadores, porém o byte 2 indicará quais atuadores devem ser desativados, por exemplo, se o valor de byte 2 for 10100001, os atuadores 7, 5 e 0 serão desativados. Para a execução desta função, a mensagem é enviada apenas no sentido do módulo de controle para o circuito de controle.

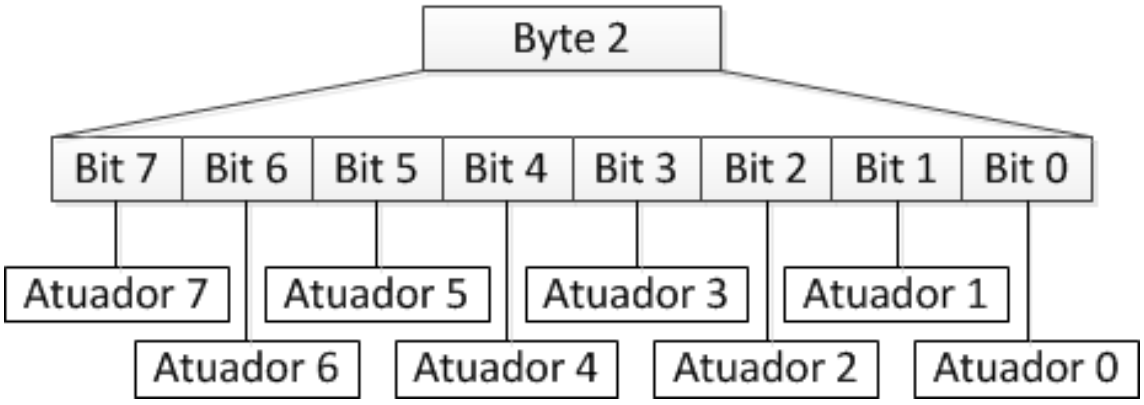


Figura 2.3: Correspondência da mensagem com os atuadores

Obter atuadores ligados Esta função faz com que o circuito de controle indique quais atuadores estão ativados. Nesta função são transmitidas duas mensagens, uma mensagem de requisição do módulo de controle para o circuito de controle e outra mensagem de resposta do circuito de controle para o módulo de controle. A mensagem de requisição não utiliza os bytes 2, 3 e 4. A mensagem de resposta não utiliza os bytes 3 e 4, mas o byte 2 informará quais atuadores estão ativados e quais estão desativados, utilizando a estrutura da figura 2.3.

Obter posição de um atuador Esta função faz com que o circuito de controle indique a posição atual do atuador desejado. Nesta função são transmitidas duas mensagens, uma mensagem de requisição do módulo de controle para o circuito de controle e outra mensagem de resposta do circuito de controle para o módulo de controle. Nesta função, o byte 1, além de indicar a função, indica a qual atuador a função se aplica, podendo variar do número hexadecimal 0x30 até 0x37, de forma que se o valor do byte 1 for o número hexadecimal 0x30, a função se aplica ao atuador 0, se o valor for 0x35, a função se aplica ao atuador 5. As mensagens de requisição e de resposta devem possuir o mesmo valor para o byte 1. A mensagem de requisição não utiliza os bytes 2, 3 e 4. A mensagem de resposta não utiliza o byte 4, mas utiliza os bytes 2 e 3 de forma que os dois bytes combinados formam um valor binário de 16 bits que representa a posição que o atuador referente à função está.

Alterar posição de um atuador Esta função faz com que o circuito de controle

movimente o atuador desejado para a posição indicada. Nesta função, o byte 1, além de indicar a função, indica a qual atuador a função se aplica, assim como a função de obter posição de um atuador, podendo variar do número hexadecimal 0x40 até 0x47. Esta função não utiliza o byte 4, mas utiliza os bytes 2 e 3, de forma que os dois bytes combinados formam um valor binário de 16 bits que representa a posição para a qual o atuador referente à função deve se movimentar. Para a execução desta função, a mensagem é enviada apenas no sentido do módulo de controle para o circuito de controle.

Capítulo 3

Subsistemas

3.1 Arquitetura do sistema

A figura 3.1 representa a as relações entre os subsistemas que integram o projeto. O Módulo de Controle (MC) conecta-se ao Circuito de Controle (CC) que é responsável por interpretar os comandos recebidos pelo MCs e transformá-los em movimentos no braço robótico. O MC também se conecta a uma Webcam que captura imagens do braço robótico. O MC conecta-se ao Módulo de Acesso (MA) que por sua vez conecta-se ao Módulo de Visualização (MV). O usuário do sistema conecta-se ao MV e transmite comandos para eles. Estes comandos são retransmitidos para o MA, depois para o MC e em seguida para o CC, que fará o braço robótico se mover. As imagens da webcam serão transmitidas para o MC que retransmitirá para o MA e em seguida para o MV que as exibirá para o usuários.

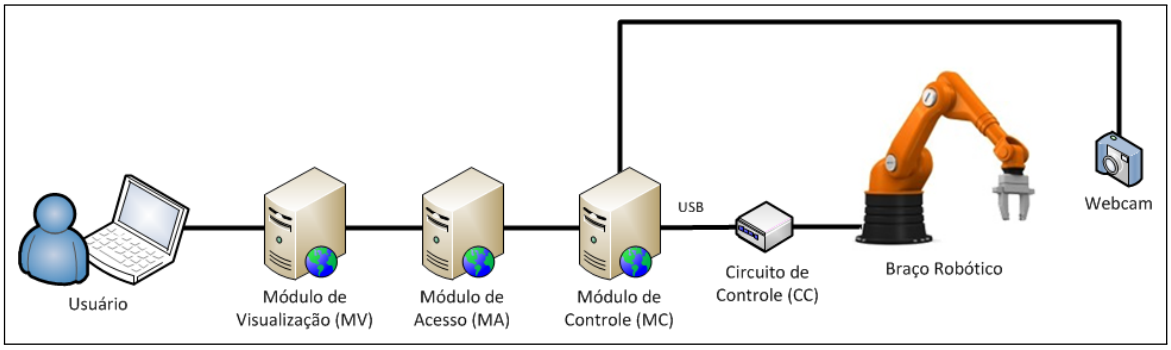


Figura 3.1: Arquitetura simples

A figura 3.1 demonstra a conexão entre os subsistemas de forma simples, porém a estrutura de conexão dos módulos do sistema é bastante flexível, devido ao

uso da tecnologia de sistemas e serviços Web, permitindo a criação de estruturas de conexão complexas sem grandes dificuldades, como pode ser exemplificado na figura 3.2.

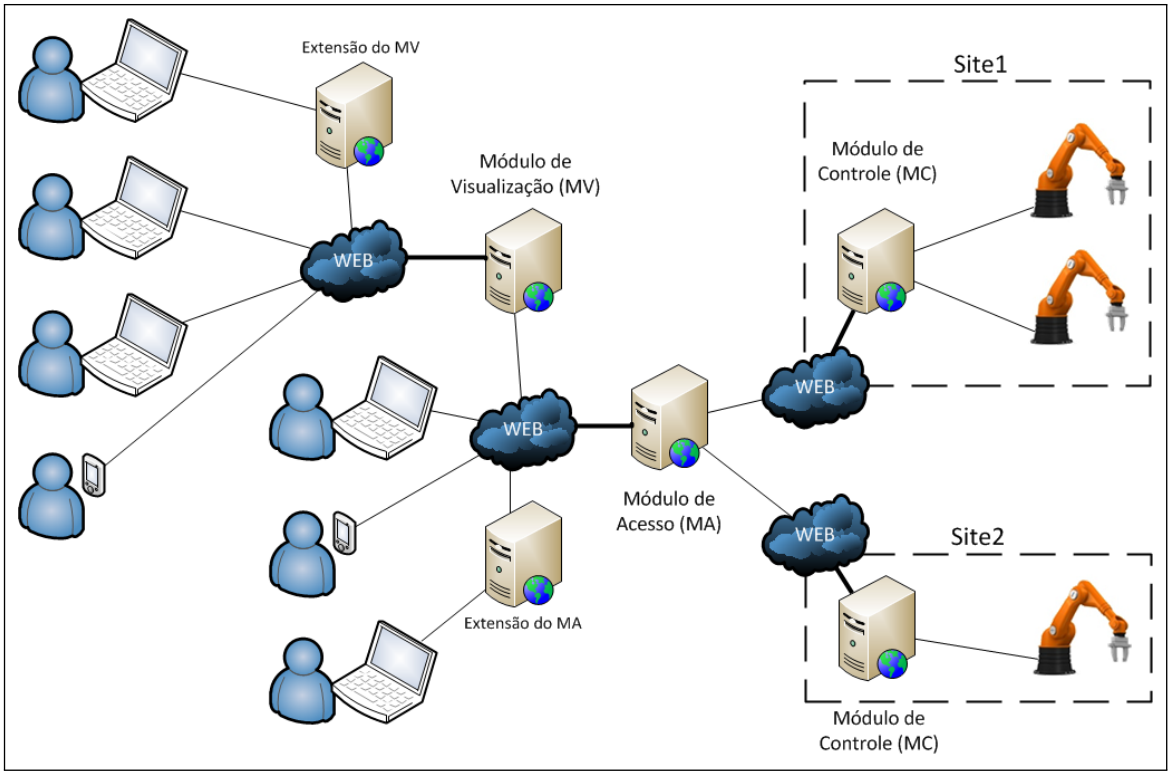


Figura 3.2: Arquitetura exemplo

Na figura 3.2 podemos ver que os usuários podem se conectar ao módulo de visualização para utilizar os recursos de representação de modelos virtuais ou podem se conectar diretamente ao módulo de acesso. Podemos observar também as extensões do módulo de acesso e do módulo visualização, utilizando-os como base para prover funcionalidades adicionais. Os módulos de controle permitem o controle de diversos robôs ao mesmo tempo, como demonstrado no Site 1 da figura 3.2.

3.2 Módulo de Controle

O módulo de controle (MC) é um subsistema responsável pelo controle dos robôs e pela captura de imagens das câmeras que acompanham os movimentos de cada robô. O controle dos robôs ocorre através da comunicação com um circuito de controle utilizando o protocolo de comunicação da seção 2.3.1. O processo de captura de imagens das câmeras dos robôs será detalhado no capítulo 5.

O módulo de controle possui uma interface de operação para que o administrador configure as conexões com os circuitos de controle de cada robô e as propriedades das câmeras conectadas. A interface de operação possui as seguintes funcionalidades:

Detectar circuitos de controle conectados Obtém a lista dos dispositivos conectados ao computador do módulo de controle e verifica se algum deles está na lista de dispositivos compatíveis com o módulo de controle.

Ver biblioteca de robôs Apresenta a biblioteca de modelos de robôs. A biblioteca apresenta os modelos compatíveis com o módulo de controle. Cada registro de modelo de robô na biblioteca possui as informações do identificador do circuito de controle, o número de atuadores que o robô possui, as posições mínimas e máximas alcançadas por cada atuador, o tipo de robô (cartesiano, articulado, cilíndrico, etc.) e o nome do robô.

Adicionar modelo de robô à biblioteca Cria um registro de robô na biblioteca de robôs compatíveis.

Controlar robô Apresenta uma tela com a imagem da câmera associada ao robô e uma barra para controlar a posição de cada atuador do robô.

Detectar câmeras conectadas Obtém a lista de câmeras conectadas ao computador do módulo de controle.

Calibrar câmera Ajusta as imperfeições da imagem da câmera. A calibração e o posicionamento de câmeras serão detalhados no capítulo 5.

Posicionar câmera Detecta a posição da câmera em relação ao robô.

Gravar câmera Grava a calibração e o posicionamento de uma câmera. Este registro é importante para que as câmeras não precisem ser calibradas e posicionadas toda vez que forem conectadas, basta o administrador selecionar o registro da câmera que foi conectada.

Selecionar registro de câmera Carrega as informações de calibração e posicionamento da câmera.

Ver imagem da câmera Apresenta a imagem atual de uma câmera.

Associar câmera a um robô O administrador deve selecionar qual câmera visualiza cada robô.

As figuras a seguir apresentam as interfaces de usuário do módulo de controle.

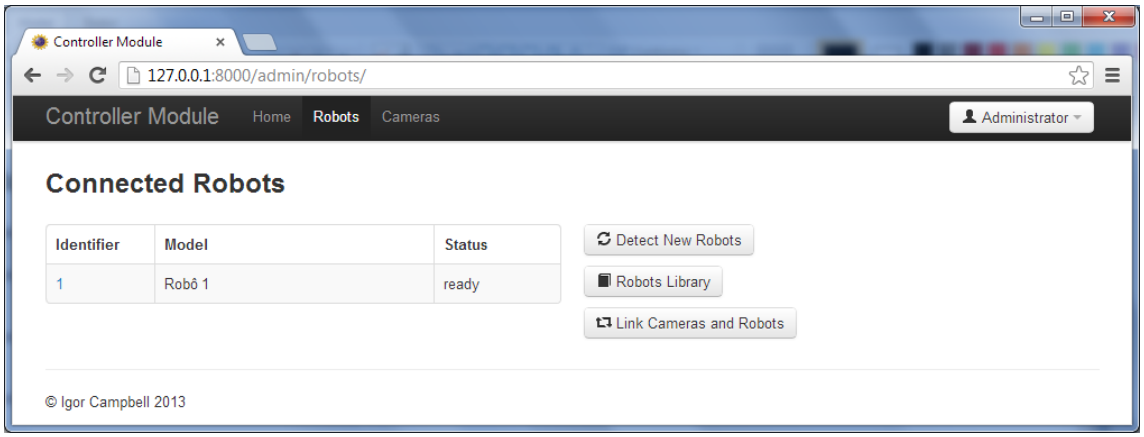


Figura 3.3: Tela com a lista de robôs conectados no MC

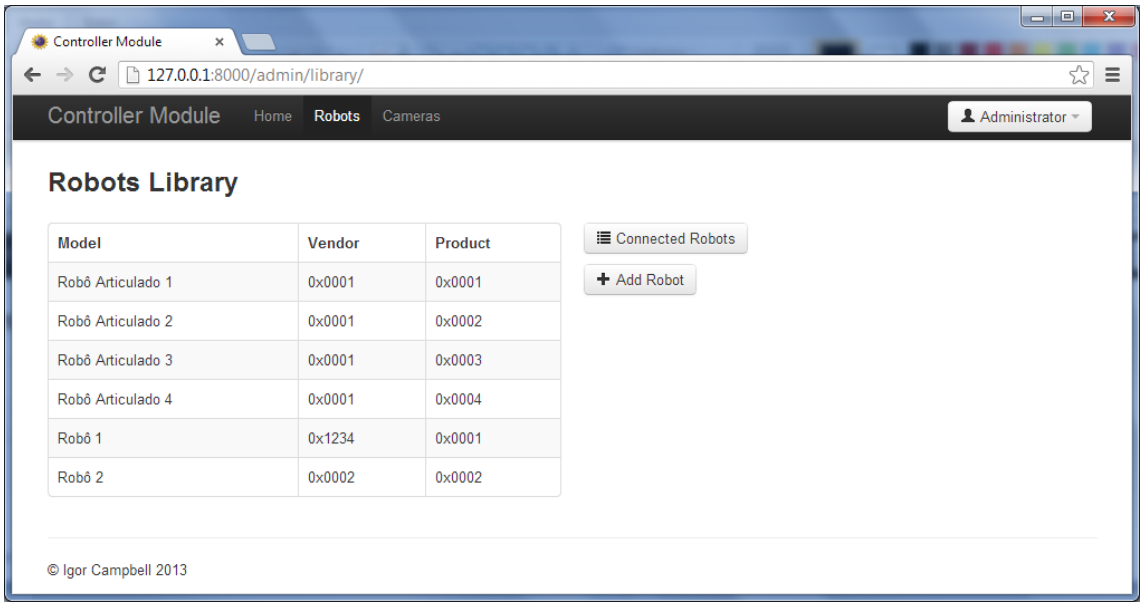


Figura 3.4: Tela com a biblioteca de robôs do MC

A figura 3.3 apresenta a tela com a lista de robôs conectados ao módulo de controle. Esta tela contém a lista dos robôs conectados, informando o identificador, o nome do modelo e o estado de cada robô, um botão para verificar se existe algum robô conectado que ainda não está na lista, um botão para visualizar a biblioteca de robôs (figura 3.4) e um botão para associação de robôs com as câmeras (figura 3.6). A lista de robôs conectados permite que o usuário clique no identificador de um robô para acessar a tela de controle (figura 3.7).

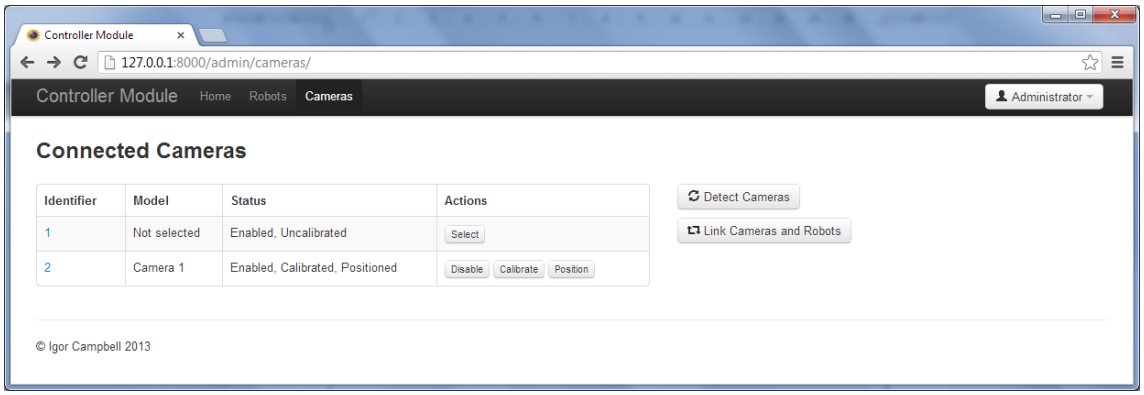


Figura 3.5: Tela com a lista de câmeras conectadas no MC

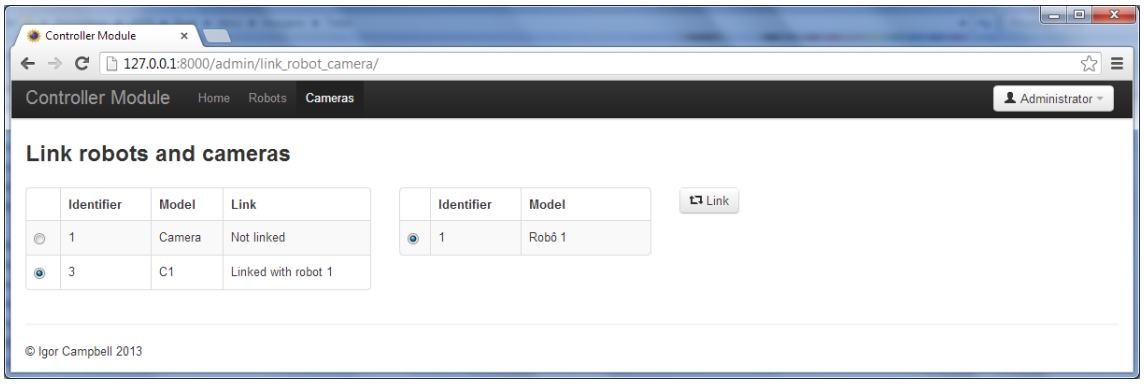


Figura 3.6: Tela com a associação entre câmeras e robôs no MC

A figura 3.4 apresenta a tela da biblioteca de robôs do módulo de controle. A biblioteca de robôs armazena as informações dos modelos de robô suportados pelos módulos de controle, permitindo acrescentar novos modelos à biblioteca e alterar informações dos modelos existentes. Esta tela possui a lista de modelos de robôs suportados informando o nome do modelo, o identificador de fabricante e o identificador de produto, para que o dispositivo seja identificado quando for conectado à porta USB do computador do módulo de controle. Esta tela também possui um botão para visualizar a lista de robôs conectados (figura 3.3) e um botão para adicionar um novo modelo de robô à biblioteca.

A figura 3.5 apresenta a lista de câmeras conectadas ao módulo de controle. A lista de câmeras conectadas permite que o usuário visualize as imagens da câmera em tempo real ao clicar no identificador da câmera desejada. Esta tela contém um botão para detectar câmeras conectadas e um botão para associar câmeras e robôs (figura 3.6). O botão para detectar câmeras conectadas verifica se existe alguma nova câmera conectada ao módulo de controle e a adiciona à lista. Quando uma

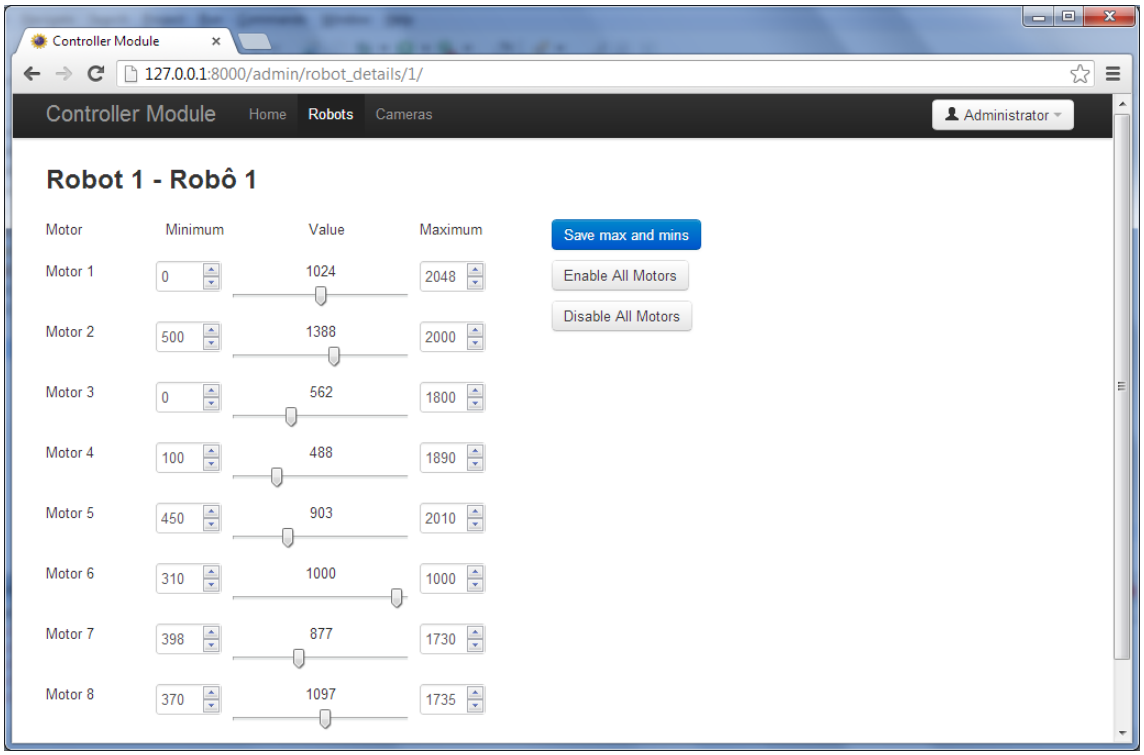


Figura 3.7: Tela de controle de um robô no MC

câmera é adicionada é necessário identificar qual é o modelo da câmera e selecionar na lista de câmeras conhecidas, através do botão que fica na coluna de ações das câmeras. Na coluna de ações também é possível desligar, calibrar e posicionar as câmeras.

A figura 3.6 apresenta a tela de associação de câmeras e robôs, que possui uma lista de robôs conectados, uma lista de câmeras conectadas e um botão para associação. O usuário deve selecionar o robô e a câmera que deseja associar e clicar no botão de associação para que as imagens da câmera selecionada sejam apresentadas na tela de controle do robô selecionado.

A figura 3.7 apresenta a tela de controle de um robô. A tela de controle possui a lista de atuadores do robô e controles para selecionar a posição atual de cada atuador e configurar os limites de posição mínimo e máximo que cada atuador pode alcançar. Os limites mínimo e máximo de cada atuador podem ser gravados na biblioteca do módulo de controle através do botão ao lado dos controles.

3.3 Módulo de Acesso

O Módulo de Acesso (MA) controla o acesso dos usuários ao sistema e comunica-se com os MCs para que os usuários possam controlar os robôs. A comunicação com os MCs ocorre através do serviço Web do MC, que será explicada na seção 4.1.1.

Quando o usuário se conectar ao MA, é apresentada uma tela com a lista de robôs disponíveis. O usuário pode escolher um dos robôs para controlar. Após escolher o robô, é apresentada ao usuário uma tela de controle do robô, com a imagem da câmera associada e barras de rolagem para o controle de cada um dos atuadores do robô. A tela do usuário requisita a atualização da imagem do robô em intervalos de 0,5 segundo. Esta requisição é encaminhada para o MC que envia as imagens atualizadas. As barras de rolagem, quando alteradas, enviam comandos para os MCs movimentarem os atuadores dos robôs.

O MA possui uma lista com o endereço de todos os MCs aos quais deve se conectar e organiza cada um deles por um identificador chamado *location_id*. A cada requisição de atualização de imagem ou comando de movimento do robô, deve ser enviado o *location_id*, para que o comando seja encaminhado para o MC correto.

A lista de robôs disponíveis no MA é formada pelo conjunto de robôs de todos os MCs que estejam conectados e marcados como disponíveis. Esta marcação é feita pelo MA para que somente um usuário obtenha o controle de cada robô de cada vez. Quando um usuário acessa um determinado robô, ele é marcado como ocupado até que o usuário deixe de utilizá-lo.

O MA possui apenas duas telas de interface com o usuário, a tela inicial (figura 3.8) e a tela de controle (figura 3.9). A tela inicial apresenta a lista de robôs disponíveis para controle, informando o identificador do MC de cada robô, o identificador que o robô possui em seu MC e o nome do modelo do robô. Ao clicar em um dos robôs disponíveis na lista, o usuário pode controla-lo através da tela de controle, que apresenta a lista de atuadores do robô com um controle de posição para cada, e exibe a imagem da câmera associada ao robô.

Figura 3.8: Tela de robôs conectados no MA

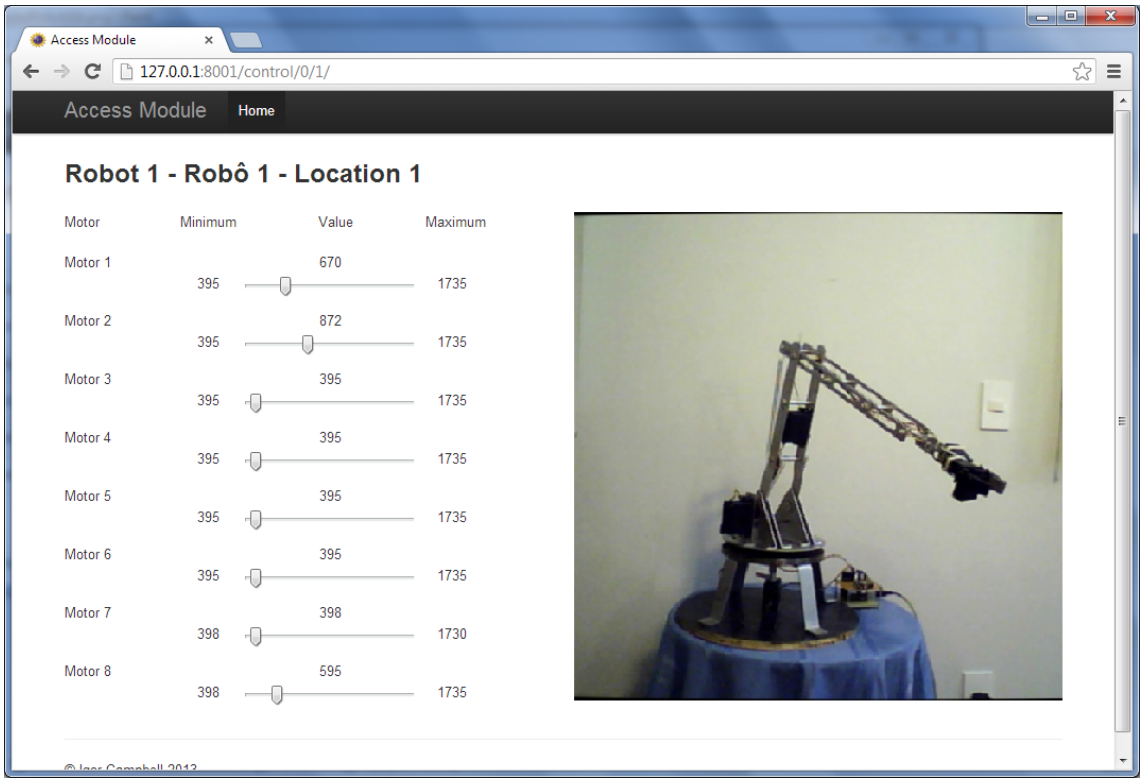


Figura 3.9: Tela de controle de um robô no MA

3.4 Módulo de Visualização

O módulo de visualização (MV) é um serviço Web do tipo REST que estende o MA adicionando a funcionalidade de visualização do robô virtual, portanto, o usuário pode acessar as funcionalidades do MA através dele.

O MV possui uma biblioteca de robôs virtuais que simulam modelos de robôs reais. Os usuários podem criar seus próprios modelos virtuais em softwares de modelagem 3D e salvá-los no formato de arquivo COLLADA (arquivos com extensão

*.dae) e enviar para o sistema para converter para o formato interno utilizado no projeto, ficando disponíveis a outros usuários.

O formato de arquivo utilizado para os modelos de projeto contém, além das informações visuais do objeto, as transformações dos objetos 3D para cada alteração de posição em um dos motores do robô. Estas transformações são ajustadas pelo usuário no conversor do arquivo COLLADA. Os ajustes são feitos com base nas informações de juntas e esqueletos dos arquivos COLLADA, aproveitando as posições dos pivôs de cada junta e as hierarquias dos esqueletos. O MV é um sistema Web com a parte do servidor desenvolvida na linguagem de programação Python com a biblioteca Django. A interface com o usuário foi desenvolvida utilizando o padrão HTML5 com a extensão WebGL (seção 5.2) para a apresentação dos modelos virtuais em três dimensões. As figuras 3.10, 3.11, 3.12, 3.13, 3.14, 3.15 e 3.16 apresentam as telas do MV.

A tela da figura 3.10 exibe a lista de robôs virtuais cadastrados no sistema. O usuário pode selecionar algum dos robôs cadastrados para movimentar a representação virtual do modelo de robô selecionado. O usuário também pode clicar no link na parte de baixo da tela para cadastrar seus próprios modelos virtuais de robôs no sistema, que o levará para a tela da figura 3.12.

A figura 3.11 mostra a tela de controle de um robô virtual. Nesta tela o usuário pode movimentar os controles dos atuadores do robô e o modelo virtual movimentará seus elementos para simular os movimentos do robô real. Para que o usuário possa perceber melhor como o robô está posicionado, é possível utilizar o mouse em conjunto com as teclas Alt, Shift e Ctrl do teclado para transladar, rotacionar e escalar o modelo virtual. O botão “Restore” desfaz as alterações no espaço tridimensional que o usuário fez.

As figuras 3.12, 3.13 e 3.14 mostram as três telas da conversão de arquivos COLLADA para o formato interno do sistema. Na primeira parte, o usuário deve inserir o nome que será exibido no sistema para o modelo virtual e deve selecionar um arquivo no formato COLLADA (*.dae) para enviar. Na segunda parte, o usuário deve inserir as relações entre cada atuador do robô real com os elementos do modelo virtual. Para cada atuador encontrado no arquivo do modelo virtual enviado, serão apresentados os campos *translate*, *rotate axis*, *rotation angle*, *initial angle* e *initial*

position. O campo *translate* indica quanto o atuador deve se mover nos três eixos cartesianos para cada valor alterado na barra de rolagem do sistema. Os campos *rotate axis* e *rotation axis* indicam em qual eixo e quantos graus o atuador deve girar quando a barra de rolagem for alterada, respectivamente. Os campos *initial angle* e *initial position* indicam o ângulo e a posição iniciais que o atuador deve possuir. Na terceira parte, o usuário pode visualizar e mover o modelo enviado para verificar se o arquivo enviado e as relações com os atuadores estão corretos para registrar o modelo virtual no sistema.

A tela da figura 3.15 exibe a lista de robôs disponíveis em todos os MCs conectados ao sistema. Nesta tela o usuário pode selecionar o robô que deseja controlar.

As figuras 3.16 e 3.16 representam a tela de controle do robô real. Nesta tela é possível controlar o robô real utilizando imagens da câmera em conjunto com seu respectivo modelo virtual. A figura 3.16 mostra o robô durante um movimento de rotação para a esquerda e a figura 3.17 mostra o robô após finalizar este movimento. É possível observar que o robô virtual se movimenta antes do robô real. Isto ocorre devido ao tempo que o motor necessita para movimentar a estrutura do robô, que varia de acordo com a força e a velocidade de rotação do motor e o peso da estrutura.

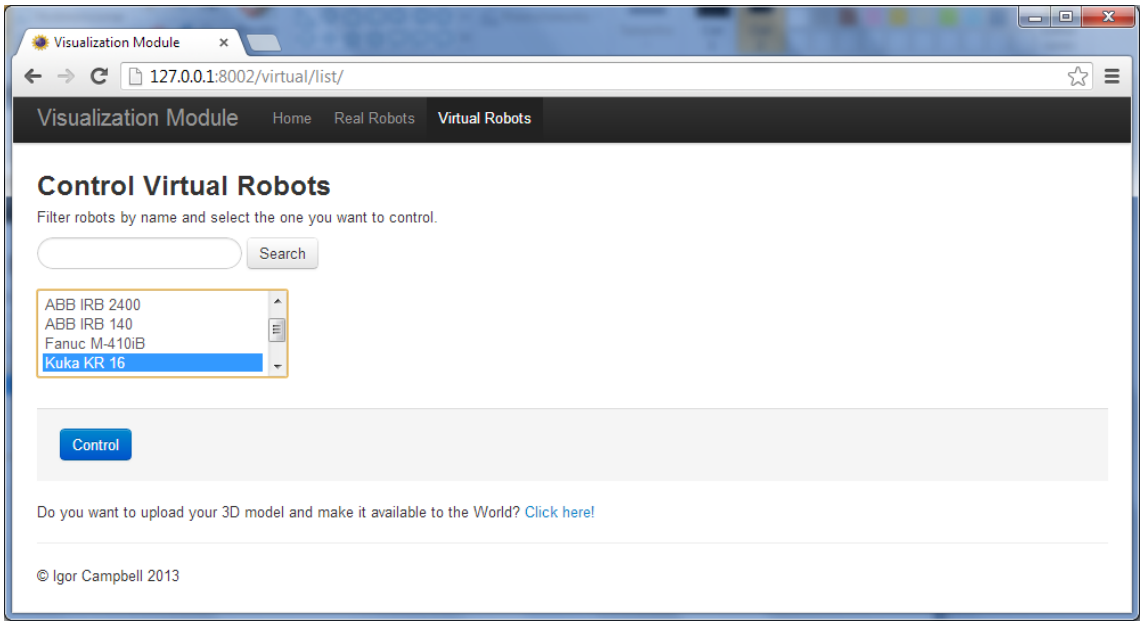


Figura 3.10: Lista de robôs virtuais do módulo de visualização

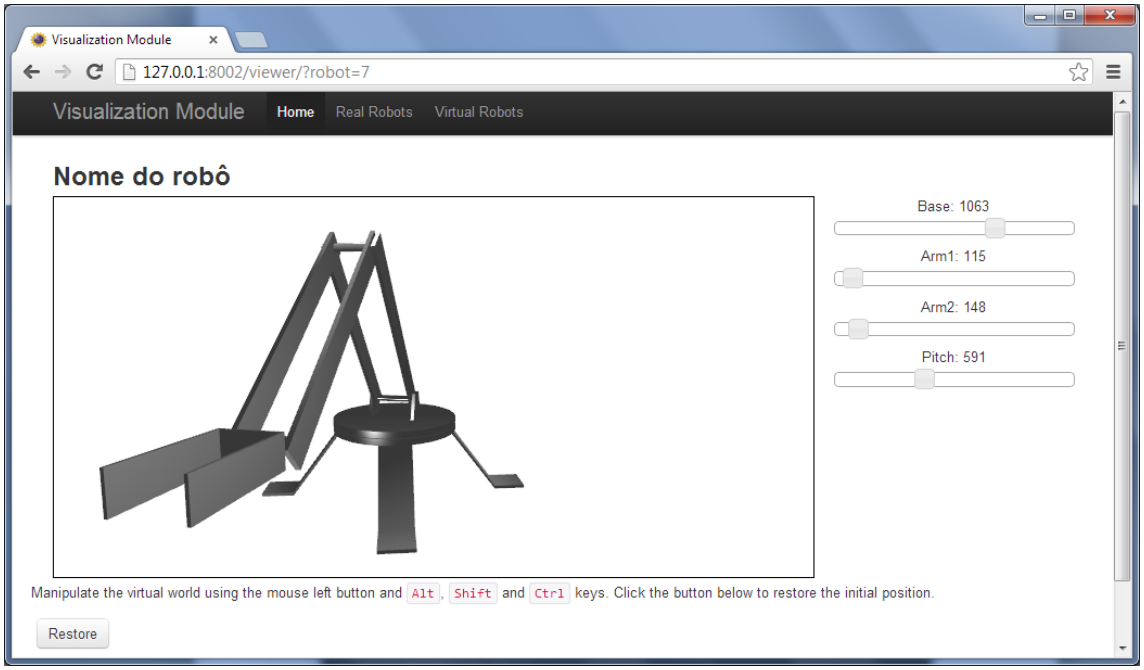


Figura 3.11: Controle de um robô virtual no módulo de visualização

Figura 3.12: Conversão de COLLADA para formato interno - parte 1

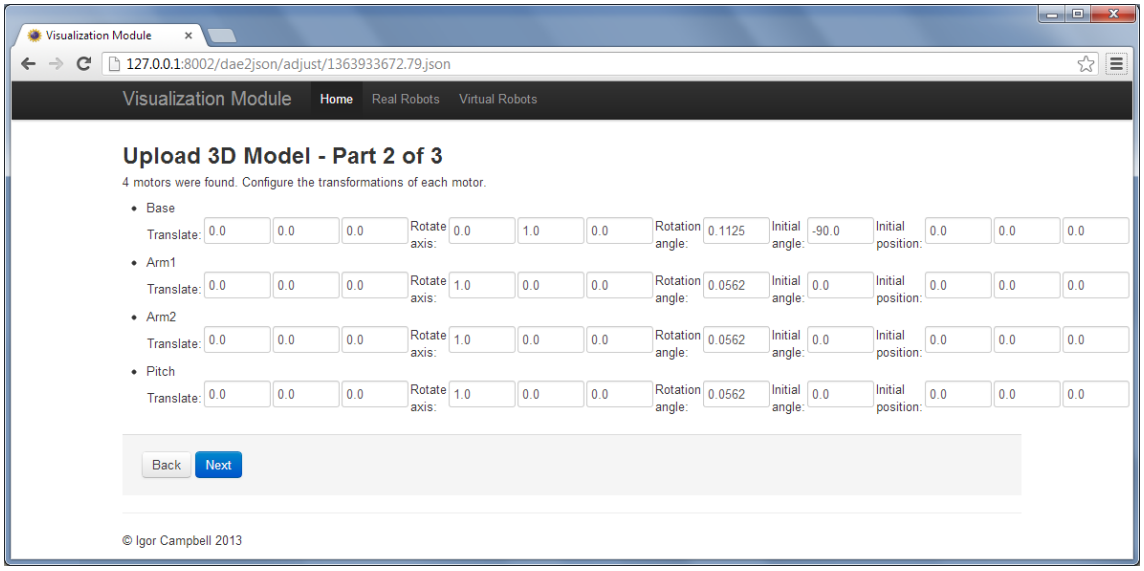


Figura 3.13: Conversão de COLLADA para formato interno - parte 2

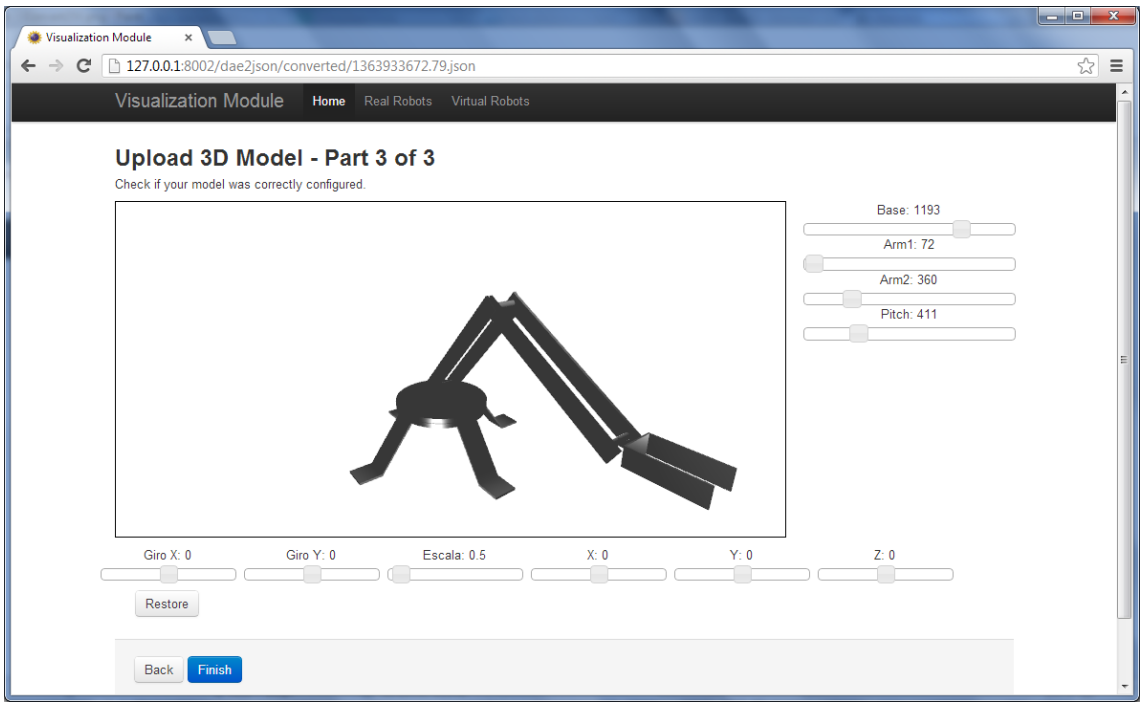


Figura 3.14: Conversão de COLLADA para formato interno - parte 3

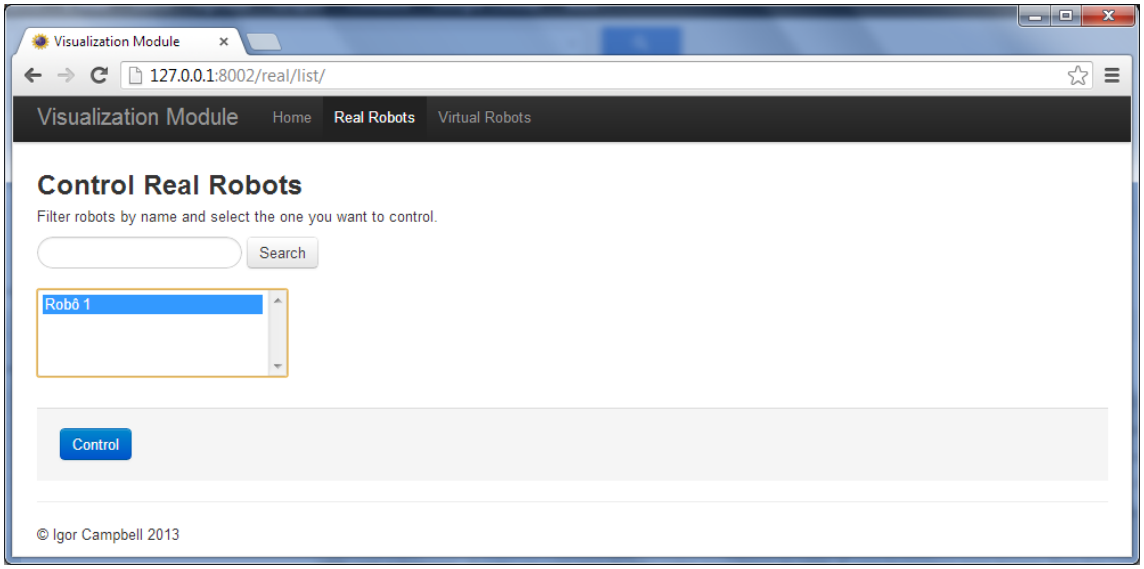


Figura 3.15: Lista de robôs reais do módulo de visualização

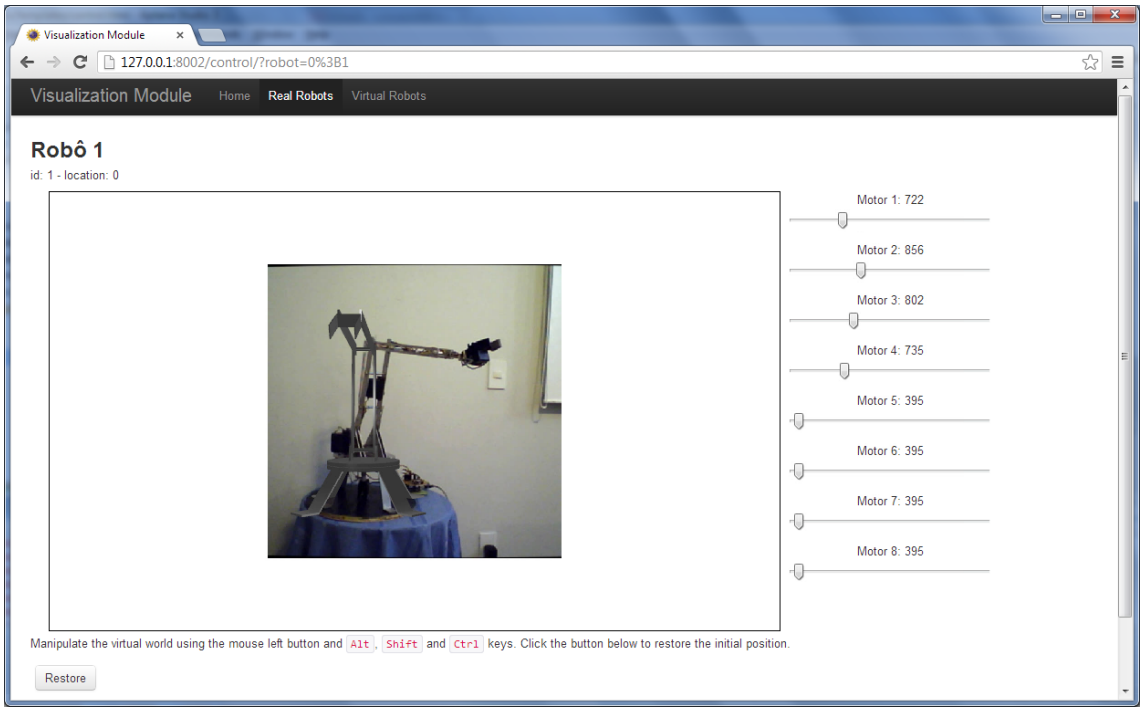


Figura 3.16: Controle de um robô real no módulo de visualização - robô em movimento

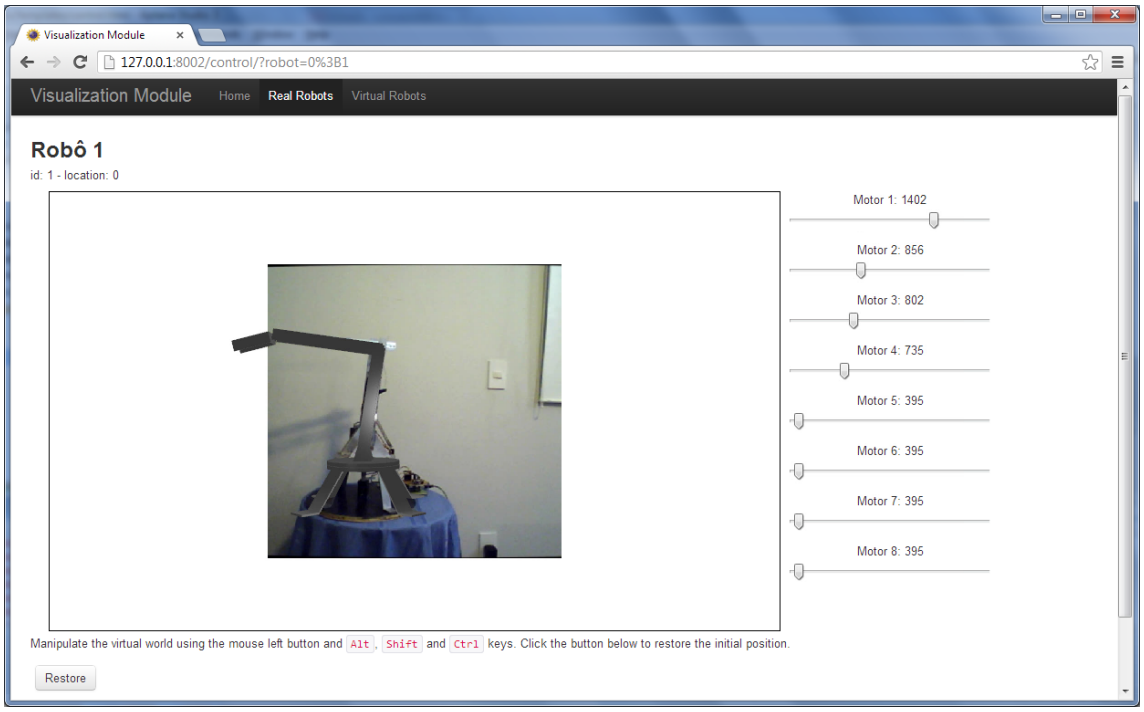


Figura 3.17: Controle de um robô real no módulo de visualização - robô parado

3.5 Tecnologias Utilizadas

Os três subsistemas são sistemas Web (Capítulo 4) e todas as suas interfaces podem ser acessadas através de um navegador de Internet. Os módulos foram desenvolvidos na linguagem de programação Python utilizando a biblioteca Django. A interface de usuários dos módulos foi desenvolvida utilizando os componentes de interface de usuário do Twitter Bootstrap. As páginas Web utilizaram a biblioteca jQuery para criar interfaces de usuário mais ricas e interativas.

3.5.1 Django

O Django é uma biblioteca da linguagem de programação Python para o desenvolvimento de sistemas Web.

3.5.2 Twitter Bootstrap

O Twitter Bootstrap é um conjunto de componentes de interface de usuário que permite a montagem de páginas Web com características visuais utilizadas em muitos sites populares. A utilização destes componentes diminui os esforços gastos com a complexidade dos aspectos visuais das páginas Web, podendo se preocupar mais com as funcionalidades do sistema.

3.5.3 jQuery

jQuery é uma biblioteca para o JavaScript que provê facilidades para a manipulação dos elementos HTML da página e para o tratamento de eventos. Esta biblioteca também facilita o envio de requisições aos servidores utilizando a tecnologia AJAX, de forma que a página Web não precise ser totalmente recarregada a cada requisição ao servidor.

Capítulo 4

Tecnologias Web

4.1 Sistemas Web

Os três sistemas utilizados no projeto são sistemas Web. Estes sistemas são baseados na arquitetura dividida em servidor e clientes, aonde os clientes acessam o sistema através da Internet utilizando um navegador, permitindo o fácil acesso e sem grandes requisitos de sistema.

Os servidores Web enviam o conteúdo para os navegadores dos clientes que processam este conteúdo exibindo páginas Web para o usuário. O conteúdo enviado pelo servidor Web pode ser de diversos tipos, mas os principais são códigos HTML, códigos CSS, códigos JavaScript e arquivos de mídia comuns.

Os códigos HTML (Hypertext Markup Language) são interpretados por qualquer navegador e são responsáveis por montar a estrutura dos elementos da página Web na tela do navegador.

Os códigos CSS (Cascading Style Sheet) tiveram sua primeira especificação finalizada somente em 1996, portanto, alguns navegadores mais antigos não são capazes de interpretá-los. A última especificação do CSS (CSS 3) ainda possui alguns recursos em aberto sendo que muitos dos recursos já estabelecidos ainda não são totalmente suportados por alguns dos navegadores modernos. Os códigos CSS são utilizados para alterar a formatação e o aspecto visual das páginas Web, configurando as propriedades visuais dos elementos do código HTML.

Os códigos JavaScript acrescentam novas funcionalidades à página Web, identificando a interação do usuário com a página e atribuindo um comportamento mais

dinâmico e interessante.

As páginas Web são utilizadas nos três sistemas deste projeto. O módulo de controle utiliza páginas Web somente para o acesso do administrador, para funcionalidades como a conexão dos robôs, a visualização e configuração das câmeras e a associação entre câmeras e robôs. O MA utiliza as páginas Web para que os usuários visualizem e controlem os robôs. O MV utiliza as páginas Web de forma bem semelhante ao módulo de acesso, porém acrescenta à visualização dos robôs uma simulação dos movimentos, utilizando técnicas de renderização através da biblioteca gráfica WebGL.

A utilização de páginas Web para a apresentação de conteúdos para os usuários tem se tornado cada vez mais eficiente, porém as páginas Web não são eficientes para a comunicação entre máquinas, já que elas são desenvolvidas com foco na facilidade de visualização das informações por um ser humano. Isto torna confusa a leitura de uma página por uma máquina, pois é preciso identificar as informações relevantes da página entre diversos códigos HTML, CSS, JavaScript e muitas vezes em informações visuais contidas em imagens.

Neste projeto, os módulos de controle precisam se comunicar com o módulo de acesso, que por sua vez se comunica com o módulo de visualização. Para que esta comunicação seja eficaz, foi utilizada uma tecnologia chamada serviço Web.

Serviço Web é uma tecnologia desenvolvida para a comunicação eficaz entre máquinas utilizando como base os sistemas Web. Nos serviços Web a mesma arquitetura cliente-servidor é utilizada, mas neste caso o cliente e o servidor são máquinas. Os servidores enviam as informações para os clientes em um formato pré-definido, evitando erros na interpretação dos dados. Existem diversos tipos de serviços Web, sendo que os dois mais utilizados são SOAP (Simple Object Access Protocol) e REST (Representational State Transfer).

O SOAP é um tecnologia baseada no XML (Exchange Markup Language) para a troca de informações através do protocolo HTTP. O SOAP cria uma estrutura de dados para a transmissão das informações, incluindo nesta estrutura de dados um conjunto de informações que identifica a origem e a função de cada mensagem enviada.

O REST utiliza os próprios recursos do HTTP para incluir as informações

referentes à mensagem enviada. Por esse motivo, sua utilização é mais simples do que a do SOAP, porém há diversas discussões [8, 16, 17, 9, 6] sobre as comunicações que empregam REST serem menos seguras e robustas. Pela facilidade de desenvolvimento e de utilização, os sistemas REST têm apresentado um crescimento considerável nos últimos anos, mas a tecnologia SOAP ainda domina os sistemas de grande porte.

Os serviços Web neste projeto empregam a tecnologia REST apresentando grande parte das informações no formato JSON (JavaScript Simple Object Notation).

O JSON é uma notação de dados que define o formato de representação para listas, dicionários e tipos simples de dados (valores numéricos e textuais) para a utilização em comunicações. Este formato apresenta diversas vantagens em relação ao formato XML, tais como a facilidade de interpretar as informações e as mensagens serem mais compactas.

Na seção seguinte serão detalhados os serviços Web de cada um dos subsistemas utilizados neste projeto, apresentando os recursos que cada um deles oferece.

4.1.1 Módulo de Controle

O módulo de controle apresenta os recursos que serão acessados pelo módulo de acesso.

Lista de robôs A lista de robôs conectados pode ser obtida fazendo uma requisição do tipo GET no recurso de endereço /robots/. A resposta da requisição será uma mensagem no formato JSON apresentando uma lista com o nome (chave "name") e o identificador (chave "id") de cada robô, como mostrado no exemplo:

```
1 [
2   {"name":"Robô 1", "id":1},
3   {"name":"Robô 2", "id":2}
4 ]
```

Detalhes de um robô Algumas informações mais detalhadas de um robô podem ser obtidas fazendo uma requisição do tipo GET para o recurso de endereço

/robot/<robot_id>/, onde <robot_id> representa o código de identificação do robô que se deseja obter os detalhes. A resposta da requisição será uma mensagem no formato JSON contendo o nome do robô (chave “name”), os identificador (chave “id”), a lista de motores, contendo os valores mínimos, máximos e a posição atual de cada motor (chave “motors”), o tipo de robô (chave ”type“) e o identificador da câmera associada a este robô, se houver alguma (chave “cam”). Um exemplo de mensagem de resposta da requisição é apresentado abaixo:

```
{
  "name":"Robô 1",
  "id":1,
  "motors":[[0,2000,500],[100,1000,200]],
  "type":"articulated",
  "cam":1
}
```

Mover robô Para movimentar um dos motores do robô, é feita uma requisição do tipo POST para o endereço /robot/<robot_id>/, onde <robot_id> é o identificador do robô que deve se movimentar. A requisição deve possuir os argumentos “motor” e “position” e ambos devem ser valores numéricos. O argumento “motor” indica o índice do motor que será movido e o argumento “position” determina a posição para a qual o motor se movimentará. A resposta para a requisição será vazia se não houver erros. No caso de ocorrência de algum erro, a resposta será uma mensagem HTTP de código 400 (HTTP Bad Request), que indica que a requisição feita foi inválida, enviando também um texto de descrição do erro ocorrido.

Obter imagem da câmera A imagem da câmera pode ser obtida fazendo uma requisição do tipo GET no endereço /cam/<cam_id>/, onde <cam_id> é o identificador da câmera desejada. A resposta da requisição será uma imagem codificada no formato JPEG. A imagem obtida será a imagem da câmera no momento da requisição.

Obter matrizes da câmera As matrizes de calibração e posicionamento da câmera podem ser obtidas fazendo uma requisição do tipo GET no endereço /cam/<cam_id>,

onde <cam_id> é o identificador da câmera desejada. O endereço utilizado é o mesmo para obter a imagem da câmera, mas para obter as matrizes da câmera, o argumento "matrix" deve ser passado. A resposta da requisição será uma mensagem no formato JSON contendo a matriz de projeção (chave "projection") e a matriz de modelo-visualização da câmera (chave "modelview"), como no exemplo:

```
{
  "projection":
    [1.0,0.0,0.0,0.0,
     0.0,1.0,0.0,0.0,
     0.0,0.0,1.0,0.0,
     0.0,0.0,0.0,1.0],
 "modelview":
    [1.0,0.0,0.0,0.0,
     0.0,1.0,0.0,0.0,
     0.0,0.0,1.0,0.0,
     0.0,0.0,0.0,1.0]
}
```

4.1.2 Módulo de Acesso

O módulo de acesso apresenta recursos que serão acessados pelo módulo de visualização e poderão ser acessados por qualquer sistema que possa ser desenvolvido baseado no módulo de acesso.

Lista de robôs A lista de robôs do módulo de acesso é o conjunto das lista de robôs de todos os módulos de controle conectados. A lista pode ser obtida fazendo uma requisição do tipo GET no endereço /robots/. A resposta da requisição será uma mensagem no formato JSON e seguirá o modelo:

```
[
  {"name":"Robô 1", "location":1, "id":1},
  {"name":"Robô X", "location":2, "id":5}
]
```

Onde "name" é o nome do modelo do robô, "location" é a identificação do módulo de controle ao qual o robô está conectado e "id" é o identificador do

robô em seu módulo de controle. A obtenção desta lista é feita através da requisição do tipo GET para o endereço /robots/ de cada um dos módulos de controle conectados.

Detalhes de um robô Este recurso é obtido através da requisição do tipo GET no endereço /robot/<location_id>/<robot_id>/, onde <location_id> representa o identificador do módulo de controle e <robot_id> é o identificador do robô. Esta requisição será encaminhada para o endereço /robot/<robot_id>/ do módulo de controle desejado.

Mover robô Este recurso é obtido através da requisição do tipo POST no endereço /robot/<location_id>/<robot_id>/, onde <location_id> representa o identificador do módulo de controle e <robot_id> é o identificador do robô. A requisição será encaminhada para o endereço /robot/<robot_id>/ do módulo de controle identificado, para que este possa movimentar o robô desejado.

Obter imagem da câmera Este recurso é obtido através de uma requisição do tipo GET no endereço /cam/<location_id>/<robot_id>/, onde <location_id> é o identificador do módulo de controle e <robot_id> é o identificador do robô cuja câmera associada deseja-se obter a imagem. A requisição será encaminhada para o endereço /cam/<cam_id>/ do módulo de controle identificado, onde <cam_id> é o identificador da câmera associada ao robô identificado por <robot_id>.

Obter matrizes da câmera Este recurso é obtido da mesma forma que o recurso de obter imagens da câmera, porém neste recurso deve ser passado o parâmetro "matrix". A requisição será encaminhada para /cam/<cam_id>/ do módulo de controle desejado, onde <cam_id> é o identificador da câmera associada ao robô desejado.

4.1.3 Módulo de Visualização

O módulo de visualização apresenta os mesmos recursos do módulo de acesso. As requisições feitas para obter estes recursos serão redirecionadas para o módulo de acesso. Além dos recursos do módulo de acesso, o módulo de visualização adiciona

2 novos recursos:

Lista de modelos virtuais A lista de robôs virtuais pode ser obtida através da requisição do tipo GET no endereço /models/. Se o argumento "name" for passado, os resultados serão filtrados, sendo que o valor contido em "name" será comparado com o nome de modelo de cada robô virtual, apresentando na lista somente os modelos cujo nome se assemelha a "name". A resposta da requisição será uma mensagem no formato JSON seguindo o modelo:

```
1 [
2   {"name":"Robô 1", "id":1},
3   {"name":"Robô X", "id":2}
4 ]
```

Onde "name" é o nome do modelo de robô virtual e "id" é o identificador do robô virtual.

Obter modelo virtual O modelo virtual pode ser obtido através da requisição do tipo GET no endereço /model/model_id/, onde <model_id> é o identificador do modelo virtual. A resposta da requisição é um arquivo no formato JSON com as informações sobre o modelo virtual. A estrutura deste arquivo será apresentada na seção 5.2.1.

4.2 Descrição dos Serviços Web

Os serviços Web baseados na tecnologia SOAP utilizam uma estrutura para descrição detalhada dos recursos disponibilizados. Esta estrutura é o WSDL (Web Service Description Language). O desenvolvimento de serviços Web SOAP conta com diversas ferramentas para a geração automática de WSDL, além de ferramentas para a geração de uma estrutura básica de clientes para o serviço desenvolvido, baseado no WSDL, agilizando bastante o processo de desenvolvimento de sistemas deste tipo.

O WSDL em sua versão 1.0 não apresenta uma boa forma de descrever os serviços Web baseados na tecnologia REST, porém sua versão 2.0 já apresenta recursos que possibilitam descrever corretamente os serviços REST. Antes da criação do WSDL 2.0, foi desenvolvida uma estrutura de descrição para os serviços REST, o

WADL (Web Application Description Language), porém o W3C (World Wide Web Consortium), organização internacional de padronização de tecnologias para a Web, não tem planos para o reconhecimento do WADL como um padrão para a Web, somente o WSDL.

Apesar do WADL e do WSDL 2.0 possibilitarem a descrição dos serviços REST, a utilização destas estruturas não é tão comum quanto o WSDL para o SOAP, pois não há boas ferramentas para gerá-los automaticamente, como há para o SOAP.

Neste projeto foram desenvolvidas duas ferramentas para facilitar o desenvolvimento de serviços e clientes REST. A primeira ferramenta possibilita a geração automática de estruturas WADL para descrever os serviços REST. A segunda ferramenta permite a geração da estrutura básica de clientes REST, baseadas no WADL. A ferramenta de geração de clientes REST gera uma estrutura funcional, mas que ainda requer ajustes manuais para que o cliente funcione em sua forma ideal, entretanto a utilização destas duas ferramentas já agiliza consideravelmente o desenvolvimento de serviços e clientes REST.

Capítulo 5

Visualização dos Robôs

5.1 Imagens das Câmeras

Os sistemas de acesso e de visualização permitem acessar as imagens das câmeras associadas aos robôs. Através de páginas Web, os usuários podem visualizar as imagens do robô escolhido em tempo real. Ao acessar a página de visualização da câmera de um robô, a página fará uma requisição para o sistema de controle no qual a câmera está conectada. Esta requisição fará o sistema de controle captar a imagem da câmera, transmitir para o sistema de acesso ou de visualização que retransmitirá para a página para que a imagem seja exibida para o usuário.

5.1.1 Captação

A captação das imagens é realizada através de uma Webcam conectada ao sistema de controle. As câmeras devem ser corretamente instaladas no servidor de controle de acordo com as instruções do fabricante da câmera. O software do sistema de controle é capaz de detectar as câmeras instaladas, porém não é possível obter o modelo de cada uma delas, cabendo ao administrador selecionar o modelo de câmera em questão. O software de controle utiliza a biblioteca PyOpenCV para a manipulação das câmeras. Ele capta a imagem da câmera desejada, aplica a calibração da câmera, removendo as distorções que ela possui, e codifica a imagem no formato JPEG, reduzindo o tamanho da imagem, o que facilita a transmissão.

5.1.2 Transmissão

A requisição da imagem da câmera é realizada através do serviço Web do sistema de controle pelo sistema de acesso. O sistema de controle captará a imagem e transmitirá para o sistema de acesso no formato JPEG utilizando o protocolo de comunicação HTTP. Ao receber a imagem, o sistema de acesso retransmitirá a imagem para a página Web no navegador do usuário, utilizando a mesma codificação e protocolo.

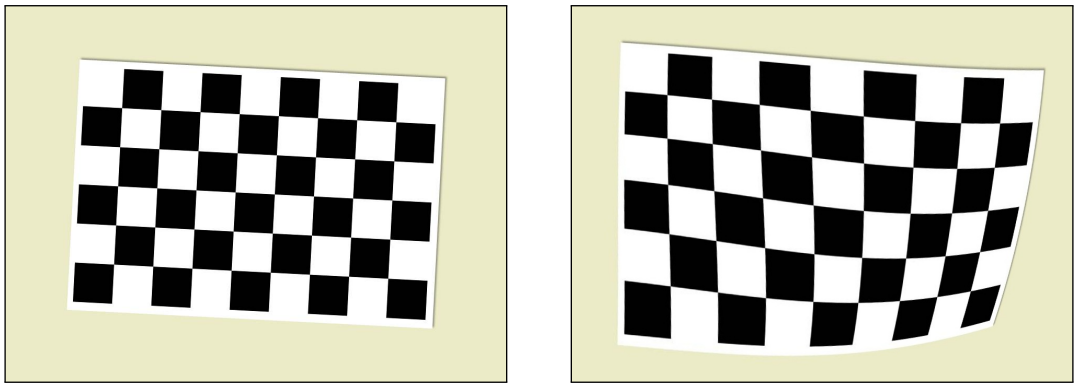
5.1.3 Exibição

A imagem recebida na página Web no navegador do usuário será exibida utilizando o elemento HTML img. A página Web para a visualização das imagens do robô utiliza um código javascript para alterar, em intervalos regulares de tempo, o parâmetro src do elemento img do HTML, com isso todo o processo de captação, transmissão e exibição das imagens do robô é repetido, atualizando frequentemente as imagens que são exibidas ao usuário.

5.1.4 Calibração das câmeras

No processo de captação das imagens das câmeras, a calibração das câmeras é aplicada nas imagens. Esta calibração deve ser realizada pelo administrador do sistema de controle. Toda câmera pode apresentar pequenas irregularidades no formato de suas lentes ou no posicionamento de seus sensores. Estas irregularidades provocam pequenas distorções na imagem captada. A figura 5.1a apresenta uma imagem sem distorções e a figura 5.1b apresenta a mesma imagem com uma distorção de câmera acentuada.

As distorções podem atrapalhar na visualização correta do robô, por isso elas devem ser eliminadas realizando a calibração da câmera. Para tanto é utilizado um elemento visual de referência e diversas imagens da câmera são processadas utilizando este elemento de referência. Com isso é possível identificar as regiões da imagem da câmera em que este elemento de referência não apresenta seu formato real, o que indica uma distorção na imagem, que pode ser medida verificando a diferença entre o formato captado pela câmera e o formato real do elemento de



(a) Imagem sem distorção (b) Imagem com distorção

Figura 5.1: Efeito das irregularidades das lentes das câmeras

referência. Após realizar as medidas de distorção da câmera, é possível aplicar nas imagens uma remoção de distorção, adicionando uma transformação oposta à distorção da câmera, anulando as irregularidades.

Para a calibração das câmeras o software de controle utiliza as funções findChessboardCorners, CalibrateCamera2 e undistort da biblioteca PyOpenCV. A função findChessboardCorners utiliza uma imagem de referência semelhante a um tabuleiro de xadrez (quadriculado de branco e preto) e identifica as quinas dos quadrados. Como as dimensões reais do tabuleiro são conhecidas, é possível inferir a distorção imposta pela lente. A função findChessboardCorners deve ser executada diversas vezes com o tabuleiro de xadrez posicionado em diferentes partes da câmera e todos os pontos encontrados são armazenados. Os pontos armazenados são processados pela função CalibrateCamera2 para que as distorções da câmera sejam calculadas e armazenadas na matriz de calibração. A função undistort utiliza a matriz de calibração da câmera para anular as distorções na imagem. Esta função é executada toda vez que a imagem da câmera é captada, apresentando sempre ao usuário a imagem já corrigida.

5.2 Robôs Virtuais

O sistema de visualização apresenta um recurso de visualizar um modelo virtual do robô. Este modelo virtual é uma representação tridimensional do robô virtual, apresentando os mesmos movimentos que o robô real deve apresentar. Para

a apresentação deste modelo virtual é utilizada a tecnologia WebGL (Web Graphics Library) que não requer o uso de plugins. O WebGL é uma biblioteca gráfica JavaScript baseada no padrão OpenGL ES 2.0 e permite o uso de GPUs (Graphics Processing Units) para acelerar a geração das imagens. O WebGL utiliza o elemento canvas do HTML5 para apresentar seu conteúdo. A tecnologia ainda é experimental e alguns navegadores atuais ainda não a suportam. O processo de geração das imagens se assemelha a montagem de uma cena, onde devem ser posicionados os objetos que aparecerão na cena, as luzes que iluminarão estes objetos e a câmera que captura a cena.

Os objetos da cena são representados pelas faces e pelos vértices contidos em cada face. A câmera possui algumas propriedades que devem ser definidas, tais como sua posição na cena, o ângulo de abertura de sua lente, a direção apontada, entre outras. As propriedades das luzes podem ser o tipo de luz (direcional, ambiente, posicional, etc), a posição/direção e as suas cores, dependendo do método de cálculo utilizado para a iluminação. Os objetos podem apresentar diversas propriedades visuais, dependendo do método de cálculo de iluminação utilizado e do uso de recursos como texturas e transparência. As propriedades utilizadas neste projeto são as informações de textura que devem ser aplicadas a cada face e as propriedades para o cálculo de iluminação.

O método de cálculo de iluminação utilizado neste projeto é o modelo de reflexão de Phong, que utiliza as cores ambiente, especular e difusa das luzes e dos objetos, além das informações dos vetores normais de cada vértice e a posição ou direção das fontes de luz.

O método Phong calcula a cor de cada ponto da imagem final através da projeção do raio luminoso vindo da fonte de luz, refletindo no objeto e chegando à câmera. A figura 5.2 representa os elementos utilizados no cálculo.

5.2.1 Arquivo de modelos virtuais

A visualização dos modelos virtuais dos robôs utiliza o WebGL para gerar as imagens tridimensionais que representam os robôs. Estes objetos são descritos no arquivo de modelos virtuais.

O arquivo de modelos virtuais possui sua estrutura no formato JSON, pos-

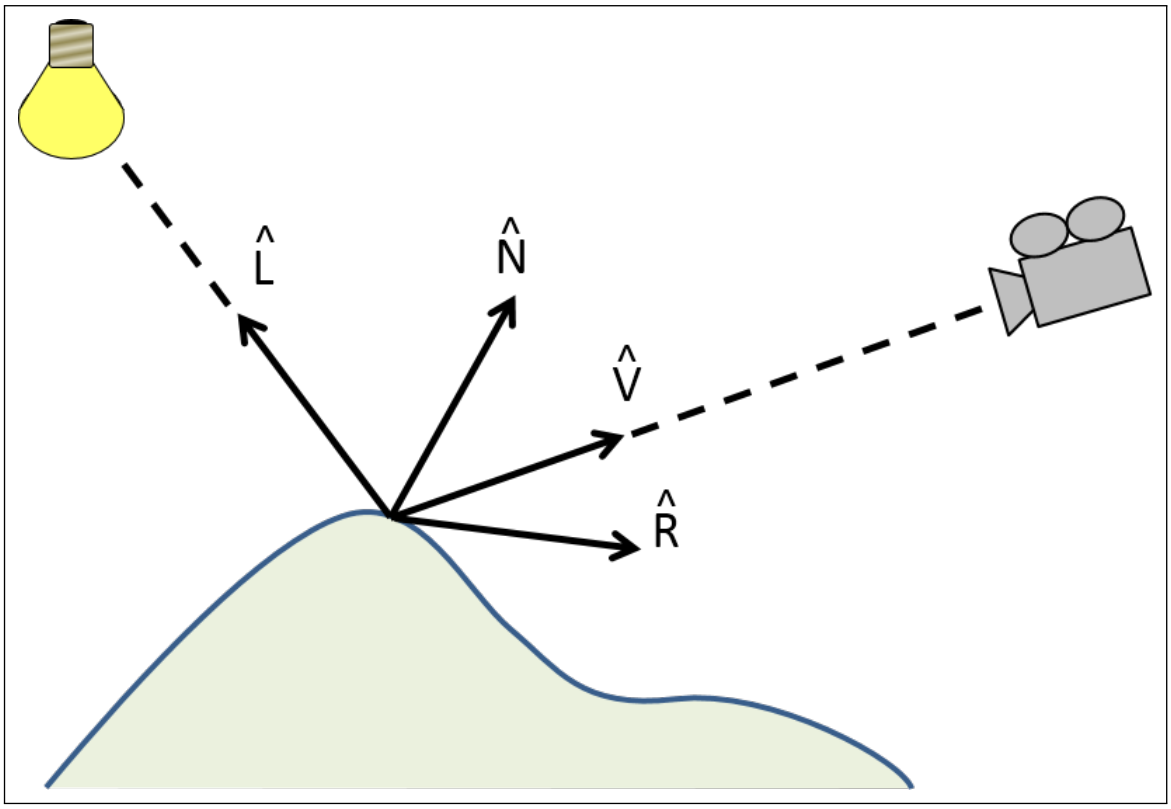


Figura 5.2: Elementos utilizados no cálculo de iluminação

suindo uma lista de materiais e objetos da cena. A lista de materiais possui as informações dos materiais referenciados pelos objetos. Os materiais representam os aspectos visuais dos objetos na cena, possuindo os coeficientes de reflexão ambiente, difuso e especular e o coeficiente de brilho que serão utilizados no cálculo de iluminação pelo WebGL. A estrutura de representação dos materiais no arquivo de modelos virtuais segue o modelo abaixo:

```
{
  "materialId":"material1",
  "ambient":[1.0,1.0,1.0,1.0],
  "diffuse":[1.0,1.0,1.0,1.0],
  "specular":[1.0,1.0,1.0,1.0],
  "shininess":128.0
}
```

Neste modelo, “materialId” é o identificador do material, para que os objetos o referenciem, “ambient”, “difuse” e “specular” são os coeficientes de reflexão ambiente, difusa e especular, respectivamente e “shininess” é o coeficiente de brilho. Os coeficientes de reflexão são definidos por uma lista com 4 elementos dividindo o

coeficiente em suas parcelas do sistema RGBA.

A lista de objetos do arquivo de modelos virtuais possui as informações dos vértices e faces dos objetos assim como informações dos vetores normais, referências para os materiais utilizados por cada objeto e uma lista de transformações referentes às movimentações que o robô deve realizar. Cada objeto do arquivo de modelos virtuais segue uma estrutura semelhante ao modelo abaixo:

```
{
  "vertex":[0.0,0.0,0.0, 0.0,1.0,0.0, 1.0,0.0,0.0, 1.0,1.0,0.0],
  "indice":[0,1,2, 1,3,2],
  "material":"material1",
  "normal":[0.0,0.0,-1.0, 0.0,1.0,0.0, 0.0,-1.0,0.0, 0.0,0.0,1.0],
  "transformations":[
    {
      "index":0,
      "translate":[0.0,0.0,0.0],
      "rotate":[0.1,0.0,1.0,0.0],
      "pivot":[0.0,0.0,0.0]
    }
  ]
}
```

No modelo acima, “vertex” é a lista de vértices que compõem o objeto, “indice” é a lista que indica quais vértices compõem cada face, “material” é a referência para um dos materiais contidos no arquivo, “normal” é a lista de vetores normais para cada vértice do objeto e “transformations” é a lista de transformações que o objeto pode sofrer.

A lista de vértices do objeto é organizada de forma que cada conjunto de três valores da lista represente as coordenadas x,y,z de um vértice.

A lista dos vértices que compõem as faces é organizada de forma que cada conjunto de três valores da lista represente uma face, portanto, somente faces triangulares são representadas neste modelo. Cada valor da lista representa o índice de um vértice. No exemplo acima, os índices 0, 1 e 2 indicam os vértices [0.0,0.0,0.0], [0.0,1.0,0.0] e [1.0,0.0,0.0] respectivamente, formando uma face do objeto.

A lista de vetores normais do objeto funciona de forma semelhante a lista de vértices, ou seja, cada vetor é um conjunto de três valores da lista. Cada vetor da

lista representa o vetor normal do vértice na mesma posição da lista de vértices. No exemplo acima, o vértice [0.0,0.0,0.0] possui o vetor [0.0,0.0,-1.0] como vetor normal e o vértice [0.0,1.0,0.0] possui o vetor [0.0,1.0,0.0] como vetor normal.

A lista de transformações é utilizada para que o modelo do robô virtual se movimente quando o robô real se movimentar. Para isso, as transformações possuem o elemento "index" que representa o índice do motor do robô real e os elementos "translate" e "rotate" que representam as transformações de translação e rotação que devem ser feitas quando o motor identificado se mover. O elemento "pivot" representa o ponto em torno do qual as transformações serão realizadas. Para exemplificar o funcionamento dos elementos da lista de transformações utilizaremos o modelo acima. Caso o robô real movimente seu motor de índice 0 para a posição 20, o modelo virtual deverá realizar uma rotação em torno do ponto [0.0,0.0,0.0] no eixo de rotação [0.0,1.0,0.0] e a rotação será de 2 graus, pois o valor de rotação 0.1 deve ser multiplicado pela posição do motor (20). Com estas transformações, o robô virtual deve apresentar movimentos semelhantes ao robô real, se o modelo virtual for corretamente configurado.

Capítulo 6

Testes

6.1 Compatibilidade com Navegadores

A possibilidade de acessar o sistema sem restrições de dispositivos ou de navegadores de Internet é fortemente desejável para a maioria dos projetos Web, entretanto muitos recursos ainda não estão disponíveis em alguns navegadores. Este projeto utiliza elementos da quinta versão do HTML, que ainda está em processo de especificação e a interface WebGL, ainda em fase experimental, portanto, é esperado que a compatibilidade deste projeto com os navegadores atuais seja limitada.

O teste de compatibilidade deste projeto foi realizado com os navegadores mais utilizados mundialmente. Na data em que os testes foram realizados (17/03/2013), a proporção de uso dos navegadores era de acordo com a tabela 6.1 e a proporção de uso de navegadores para dispositivos móveis era de acordo com a tabela 6.2 [18, 19].

Navegador	**Proporção de uso no mundo**
Chrome 25.0	32,71
Internet Explorer 9.0	16,69
Firefox 19.0	14,83
Internet Explorer 8.0	10,58
Safari 6.0	2,25
Safari 5.1	1,68
Internet Explorer 10.0	1,57
Chrome 24.0	1,48
Firefox 18.0	1,0
Opera 12.1	0,86
Internet Explorer 7.0	0,66
Safari 5.0	0,58
Firefox 16.0	0,57
Firefox 12.0	0,55
Chrome 21.0	0,51
Firefox 3.6	0,51
Chrome 23.0	0,5

Tabela 6.1: Proporção de uso dos navegadores no mundo [1]

[1]Navegadores para dispositvos móveis ou com menos de 0,5% de uso foram omitidos.

Navegador	Proporção de uso no mundo
Android	30,88
iPhone	24,37
Opera	15,55
UC Browser	8,29
Nokia	6,96
BlackBerry	2,97
iPod Touch	2,63
NetFront	2,51
Chrome	1,91
IEMobile	1,15
Dolphin	0,65

Tabela 6.2: Proporção de uso dos navegadores para dispositivos móveis no mundo [2]

Os navegadores Chrome 21.0, Chrome 23.0, Chrome 24.0, Chrome 25.0, Firefox 3.6, Firefox 12.0, Firefox 16.0, Firefox 18.0, Firefox 19.0 e Opera 12.1 foram executados nos sistemas operacionais Windows 7 Professional 64 bits e Ubuntu Linux Desktop 12.04 64 bits. Os navegadores Internet Explorer 7.0 e Internet Explorer 8.0 foram executados no sistema operacional Windows Xp 32 bits. Os navegadores Internet Explorer 9.0, Internet Explorer 10.0, Safari 5.0, Safari 5.1 e Safari 6.0 foram executados no sistema operacional Windows 7 Professional 64 bits.

Os testes dos navegadores de dispositivos móveis ficaram restritos aos dispositivos disponíveis, iPhone 4S e Samsung Galaxy SIII. Os testes dos navegadores Nokia, BlackBerry, iPod Touch (Safari para iPod Touch) e IEMobile não foram realizados porque não havia dispositivos compatíveis com estes navegadores disponíveis. Os testes dos navegadores Android, Opera, UCBrowser, NetFront, Chrome e Dolphin foram realizados no dispositivo Samsung Galaxy SIII com o sistema operacional Android 4.1.2 e os teste do navegador iPhone (Safari para iPhone) foi realizado no dispositivo iPhone 4S com o sistema operacional iOS 6.

Foi realizado um primeiro grupo de testes nos navegadores listados e foi

[2]Navegadores com menos de 0,5% de uso foram omitidos.

identificado que grande parte deles não é compatível com o elemento range do html5, desenhando uma caixa de entrada de texto onde deveria ficar uma barra de rolagem para selecionar as posições dos robôs. Este problema de compatibilidade ocorreu em navegadores modernos como o Firefox 19.0, o terceiro mais utilizado no mundo. Esse problema afetou os 3 subsistemas de forma significativa em relação a sua usabilidade, mas existem algumas bibliotecas que criam barras de rolagem que funcionam muito bem, como o elemento slider da biblioteca jQueryui, que foi utilizado neste projeto para resolver este problema de compatibilidade.

Um segundo grupo de testes foi realizado e os resultados estão nas tabelas 6.3 e 6.4.

Navegador	**Compatibilidade**		
	MC	**MA**	**MV**
Chrome 25.0	Sim	Sim	Sim
Internet Explorer 9.0	Sim	Sim	Exige instalação de extensão
Firefox 19.0	Sim	Sim	Sim
Internet Explorer 8.0	Sim	Sim	Exige instalação de extensão
Safari 6.0	Sim	Sim	Exige configuração adicional
Safari 5.1	Sim	Sim	Exige configuração adicional
Internet Explorer 10.0	Sim	Sim	Exige instalação de extensão
Chrome 24.0	Sim	Sim	Sim
Firefox 18.0	Sim	Sim	Sim
Opera 12.1	Sim	Sim	Sim
Internet Explorer 7.0	Sim	Sim	Exige instalação de extensão
Safari 5.0	Sim	Sim	Não
Firefox 16.0	Sim	Sim	Sim
Firefox 12.0	Sim	Sim	Sim
Chrome 21.0	Sim	Sim	Sim
Firefox 3.6	Sim	Sim	Não
Chrome 23.0	Sim	Sim	Sim

Tabela 6.3: Compatibilidade dos navegadores com os subsistemas

Os resultados acima mostraram que os subsistemas MC e MA tiveram com-

Navegador	**Compatibilidade**		
	MC	**MA**	**MV**
Android	Sim	Sim	Sim
iPhone	Sim	Sim	Sim
Opera	Sim	Sim	Sim
UC Browser	Sim	Sim	Não
NetFront	Sim	Sim	Não
Chrome	Sim	Sim	Não
Dolphin	Sim	Sim	Não

Tabela 6.4: Compatibilidade dos navegadores de dispositivos móveis com os subsistemas

patibilidade de 100% com os navegadores testados. Já o MV foi compatível com uma variedade menor de navegadores devido ao uso da tecnologia WebGL, mas com a estabilização desta tecnologia é esperado que a compatibilidade com os dispositivos aumente consideravelmente.

Capítulo 7

Conclusão

O principal objetivo deste trabalho foi o desenvolvimento de um sistema Web para controle e simulação de braços robóticos. O projeto foi dividido em três subsistemas, sendo o primeiro (MC) responsável pelo controle direto dos sistemas robóticos e das câmeras, o segundo (MA) responsável pela centralização do acesso a todos os MCs e o terceiro (MV) responsável pela visualização tridimensional dos sistemas robóticos. Com a integração destes três subsistemas o usuário pode visualizar e controlar os robôs de qualquer parte do mundo através da Internet, uma vez que o projeto foi desenvolvido utilizando sistemas Web.

Constatou-se que este projeto constitui também uma base para o controle de sistemas robóticos através da Internet para que novas ideias sejam desenvolvidas a partir desta base, contendo diversas funcionalidades e possibilidades de interações com outros sistemas. Os testes realizados evidenciam a extensa compatibilidade desse projeto, comprovando que o sistema pode ser acessado por inúmeros dispositivos, sistemas operacionais e navegadores, e a perspectiva é de aumentar consideravelmente esta variedade com a estabilização das tecnologias utilizadas, a exemplo do MV, que apresentou compatibilidade abaixo do esperado.

7.1 Trabalhos Futuros

O sistema, por utilizar serviços WEB, pode ser facilmente estendido, possibilitando o desenvolvimento de novas funcionalidades sem mudar a estrutura já existente. Algumas funcionalidades são listadas abaixo demonstrando a aplicabili-

dade do projeto.

Controle por cinemática inversa O sistema apresenta o controle dos robôs através da cinemática direta (controla-se o movimento de cada um dos motores), o que pode dificultar a operação em alguns casos. O controle dos robôs através da cinemática inversa (controla-se a posição cartesiana da extremidade do robô) poderia facilitar a operação em muitos casos. [3, 1]

Prevenção de colisões Através do processamento das imagens das câmeras e do robô virtual pode ser desenvolvida uma extensão para detectar e evitar colisões de partes do robô com outros objetos da área de alcance do robô.

Referências

[1] T Asfour and R Dillmann. Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem. In *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*, volume 2, pages 1407–1412. IEEE, 2003.

[2] E. Britannica. Robot. In *E. Britannica*. Encyclopaedia Britannica Print Set Suite, 2007.

[3] Samuel R Buss. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. *University of California, San Diego, Typeset manuscript, available from* `http://math.ucsd.edu/~sbuss/ResearchWeb`, 2004.

[4] J.J. Craig. *Introduction to robotics: mechanics and control*. Prentice Hall, 2004.

[5] Django. Django web framework. `http://www.djangoproject.com/`.

[6] J. Flanders. More on rest. *Online:* `http://msdn.microsoft.com/en-us/magazine/dd942839.aspx`, 2009.

[7] J. Forshaw. Webgl-a new dimension for browser exploitation. *Online:* `http://www.contextis.com/resources/blog/webgl`, 2011.

[8] S. Francia. Soap vs. rest. *Online:* `http://spf13.com/post/soap-vs-rest/`, 2010.

[9] Strikeiron Inc. Using rest and soap. *Online:* `http://www.strikeiron.com/rest-and-soap/`.

[10] A.V. Libin and E.V. Libin. Person-robot interactions from the robopsychologists' point of view: the robotic psychology and robotherapy approach. *Proceedings of the IEEE*, 92(11):1789–1803, 2004.

[11] S. Lichiardopol. A survey on teleoperation. *Technische Universiteit Eindhoven*, 2007.

[12] C. Marrin. Webgl specification. *Khronos WebGL Working Group*, 2011.

[13] B.T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, 18(6):311–317, 1975.

[14] PRIBERAM. robô. In *Dicionário Priberam da Língua Portuguesa*. PRIBERAM, dez 2012.

[15] U.S.B. Revision. 2.0 specification, 2000.

[16] M. Rozlog. Rest and soap: When should i use each (or both)? *Online: http://www.infoq.com/articles/rest-soap-when-to-use-each*, 2010.

[17] T. Singh. Rest vs. soap – the right web-service. *Online: http://geeknizer.com/rest-vs-soap-using-http-choosing-the-right-webservice-protocol/*, 2009.

[18] StatCounter Global Stats. Browser version. *Online: http://gs.statcounter.com/#browser_version-ww-monthly-201303-201303-bar*, Março 2013.

[19] StatCounter Global Stats. Mobile browser. *Online: http://gs.statcounter.com/#mobile_browser-ww-monthly-201303-201303-bar*, Março 2013.