Robocontrol 2010
4º WORKSHOP DE ROBÓTICA APLICADA E AUTOMAÇÃO
unesp

DESENVOLVIMENTO DE UMA PROPOSTA DE LABORATÓRIO DE ROBÓTICA À DISTÂNCIA PARA ÁREA EDUCACIONAL

Antonio Valerio Netto [1], *Rogério Lembo Pereira* [2]

[1] Cientistas, São Carlos, Brasil, valerio@cientistas.com.br
[2] Cientistas, São Carlos, Brasil, contato@cientistas.com.br

Abstract: *This paper presents the results of the development of a robotics laboratory at the distance. The objective is to supply a platform of tests and experiments that it can be used by schools and universities that don't possess resources to acquire robotics kits, being limited to accomplish experiments using only simulators.*

Keywords: *Mobile robot, Internet, Control system*

Resumo: Este trabalho apresenta os resultados do desenvolvimento de um laboratório de robótica à distância (LRD). O objetivo é fornecer uma plataforma de testes e experimentos que possa ser utilizada por escolas e universidades que não possuem recursos para adquirir kits robóticos, ficando limitados a realizar experimentos somente com recursos de simuladores.

Palavras-chaves: Robôs móveis, Internet, Sistema de controle

1. INTRODUÇÃO

Este trabalho apresenta o desenvolvimento de um laboratório de robótica à distância (LRD). O objetivo é fornecer uma plataforma de testes e experimentos que possa ser utilizada por escolas e universidades que não possuem recursos para adquirir kits robóticos, ficando limitados a realizar experimentos somente com recursos de simuladores. Os experimentos com robôs reais são importantes, pois possibilitam a inclusão de variáveis difíceis de modelar ou de prever, como por exemplo, ruídos nos sensores, a inércia dos robôs, o desgaste nos motores, entre outras.

Este laboratório, no futuro, poderá permitir a realização de jogos de futebol de robôs *on-line*, para os quais os alunos poderão desenvolver estratégias de jogo envolvendo conceitos de computação e inteligência artificial e transmiti-los via Internet para serem executados dentro dos laboratórios robóticos. O LRD é baseado no acesso a Internet para o envio de comandos a um robô e para a recepção de vídeo em tempo real. O LRD é baseado em uma arquitetura cliente/servidor, que utilizará o protocolo HTTP (*Hipertext Transfer Protocol*) e um servidor WWW convencional (Apache) para disponibilizar uma interface multimídia que poderá ser acessada por um Cliente WWW (Internet Explorer, Netscape, Mozilla) por meio de aplicações WEB, além de um banco de dados relacional (MYSQL) para controle dos experimentos e cadastro de usuário. Adicionalmente, o laboratório suporta a transmissão de vídeos pela Internet, possibilitando que os usuários visualizem a execução de seus experimentos. A arquitetura básica do sistema é apresentada na Fig. 1.

2. INFRAESTRUTURA E TECNOLOGIA

A infra-estrutura que foi utilizada para o projeto disponibiliza um link para a Internet com 512 Kbps de transferência para *download* e *upload* de 128 Kbps, sendo inacessível à transmissão de vídeo pela Internet para acesso remoto. Existe uma política de uso e segurança que não autoriza a entrada e saída de vídeo digital. É importante salientar que o projeto foi realizado dentro da infra-estrutura de uma incubadora de empresas que disponibiliza apenas metade da banda larga para todas as pequenas empresas incubadas. Sendo assim, o LRD foi testado, inicialmente, na Intranet da empresa.

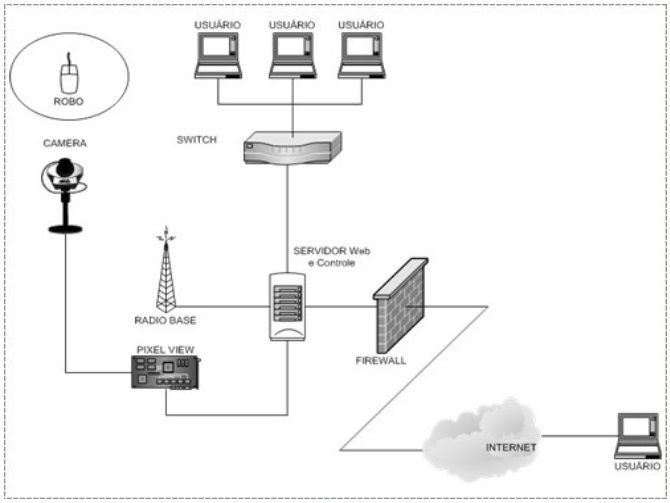

Fig. 1. Arquitetura do Laboratório de Robótica a Distância (LRD)

O LRD utiliza um servidor que executa vários tipos de serviços: controle de acesso dos usuários, transmissão de imagens da câmera do laboratório e rotinas para o acionamento do robô. O servidor possui um servidor WEB (Apache) [8] para o gerenciamento das conexões e para a execução do software de interface Web com os usuários.

As conexões dos usuários ao laboratório são realizadas em ambiente seguro utilizando o protocolo HTTPS [9]. Esse servidor utiliza os recursos das linguagens XHTML [10], CSS [11], Javascript [12], AJAX [13] e PHP [14] para disponibilizar o software de interface Web para os usuários. Esse servidor conecta por meio de uma rede *Ethernet* utilizando o protocolo TCP/IP [15].

Adicionalmente, esse servidor será capaz de executar rotinas escritas em linguagem C++ armazenadas no servidor contendo os comandos do robô. O servidor, também, é responsável pela aquisição das imagens transmitidas pela câmera CMOS s-303c do laboratório. As imagens

transmitidas pela câmera são captadas por uma placa de aquisição Pixelview [17] e transmitidas pela Intranet utilizando a tecnologia STREAMING [18]. O sistema de vídeo (câmera, placa de captura e software) deverá capturar, digitalizar e comprimir um sinal de vídeo NTSC ou PAL-M não modulado. O vídeo é armazenado em disco e transmitido via Intranet no padrão de compressão MPEG (*Moving Picture Experts Groups*) [20].

O sistema operacional utilizado foi à distribuição *Slackware* 10.0, *kernel* 2.6.13 do Linux. [7]. O *Slackware* é uma das mais antigas e conhecidas distribuições (sistema operacional e conjunto de aplicativos) do Linux. Criada em 1993 e mantida por Patrick Volkerding [21], o *Slackware* (ou simplesmente "Slack") tem como objetivo manter-se fiel aos padrões UNIX, rejeitando também ferramentas de configuração que escondam do usuário o real funcionamento do sistema. Além disso o *Slackware* é composto somente de aplicativos estáveis (e não de versões beta ou pré-releases). Simplicidade e estabilidade são duas características marcantes nesta distribuição, que é muito apreciada por usuários mais experientes. A tradução de *Slackware* é "Sistema preguiçoso" no sentido de que não aprecia ferramentas de configuração, assim sendo, as configurações do sistema são realizadas a partir da edição de documentos de texto, por isso sendo a preferida entre os usuários mais experientes.

Esse sistema possui seu próprio gerenciamento de pacotes, o *pkgtool* (*installpkg*, *upgradepkg*, *removekpg*, *explodepkg*, *makepkg*), sem gerenciamento de dependências (existem programas que adicionam esse gerenciamento, como o *slapt-get* e *swaret*). O formato dos pacotes ".tgz" é bastante simples, similar a um ".tar.gz" contendo apenas os arquivos a serem instalados em suas respectivas pastas em relação à raiz do sistema, além de um script com comandos complementares para a instalação. O *Slackware* é um Sistema Operacional Livre, ou seja, está disponível na Internet e todos têm acesso ao código-fonte, podendo então melhorá-lo ou adaptá-lo as suas próprias necessidades.

2.1. Transmissão de imagens pela Intranet/Internet

Com a possibilidade de transmissão de vídeos pela Internet; diversas aplicações começaram a surgir, como por exemplo, a vídeo-conferência, à assistência e/ou a realização de procedimentos médicos como a cirurgia à distância, etc. Basicamente, existem duas maneiras populares de se transmitir vídeo na Internet (Fig. 2) [24]:

O *download video,* pode-se fazer *download* para qualquer computador com acesso à *Internet* e guardar o vídeo no disco rígido. O *download* pode demorar alguns minutos ou algumas horas, dependendo do tamanho do ficheiro e da ligação à *Internet* (banda estreita ou banda larga). Uma vez concluída a transferência do vídeo pela *Internet*, é possível visualizar o vídeo em um computador que cumpra os requisitos mínimos de *hardware* e *software*, não sendo preciso para isso uma ligação à *Internet*.

O chamado *streaming video* é uma seqüência de imagens, que de forma comprimida são transmitidas pela Internet e mostradas no visor à chegada. Com o *streaming video*, o utilizador de Internet não precisa esperar pela conclusão do *download* do ficheiro para ver o vídeo. Isto acontece, porque o vídeo é enviado em um *stream* contínuo e é reproduzido à medida que chega ao computador. Dessa forma, é necessário selecionar (de modo automático ou pelo utilizador), se disponível, o *streaming video* mais adequado à velocidade da ligação da Internet do utilizador. À medida que o vídeo é recebido vai sendo criado um *buffer*. Quando há informação suficiente no *buffer*, o computador começa a reproduzir o vídeo. Um dos problemas do *streaming vídeo* é o congestionamento da Internet que pode provocar interrupções ou até paragem da reprodução. O utilizador necessita de um programa que descompacte e envie os dados do vídeo para a sua visualização. Este programa pode ser já parte integrante de um *browser* ou ser preciso realizar o seu *download* do site de um fabricante.

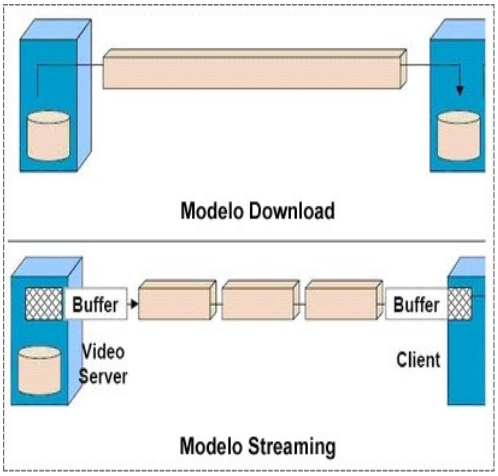

Fig. 2. Modelo *download* e modelo *streaming*

O modelo de transmissão empregado neste projeto é o *Streaming* [18]. Este modelo permite que a transmissão se torne mais leve e rápida para a execução de vídeo no computador cliente em tempo real. *Streaming,* que do inglês significa "fluxo contínuo", é uma forma de transmitir áudio e/ou vídeo pela Internet/Intranet, mas com uma particularidade muito especial: não é necessário baixar um arquivo inteiro para escutar o áudio ou assistir ao vídeo. Isso permite, por exemplo, que ocorra transmissão ao vivo de Rádio e TV por meio da Internet.

Adicionalmente, foi explorado o padrão de compressão MPEG para ser utilizado na transmissão da imagem. O software utilizado para a captura, compactação e transmissão de imagens foi o *FFmpeg*. Após o estudo desses componentes, a implementação do módulo foi iniciada pela captura de imagem da câmera CMOS, seguida da implementação da compressão e da transmissão do vídeo.

2.2. Plataforma robótica

O robô empregado neste trabalho mede 145 cm de altura 180 cm de diâmetro. Ele possui rodas omnidirecionais que possibilitam sua movimentação em qualquer direção, sendo cada uma acionada por um motor independente. Este robô também possui mecanismos opcionais de chute e drible para os experimentos com futebol de robôs.

O microcontrolador empregado no robô é o MSP430 da *Texas Instruments*. Esta família possui uma série de vantagens se comparada com outros microcontroladores

semelhantes. Dentre elas pode-se citar: baixo consumo; alto desempenho; número reduzido de instruções (menos de 30); facilidade de gravação e depuração (mediante um conector e interface especial chamado JTAG); grande quantidade de periféricos internos (como temporizadores e interfases seriais). O robô utilizado no projeto pode ser visto na Fig. 3.

O microcontrolador do robô se encarrega de controlar os *drivers* dos motores que acionam as rodas e demais mecanismos, como por exemplo, o drible e o chute. O microcontrolador também se encarrega de receber pela sua interface serial as mensagens recebidas do servidor para depois decodificá-las e comandar o mecanismo correspondente a efetuar a ação necessária. Outras atividades do microcontrolador incluem monitoramento de sinais provenientes de sensores de proximidade e outros sinais internos como temperatura e tensão da bateria.

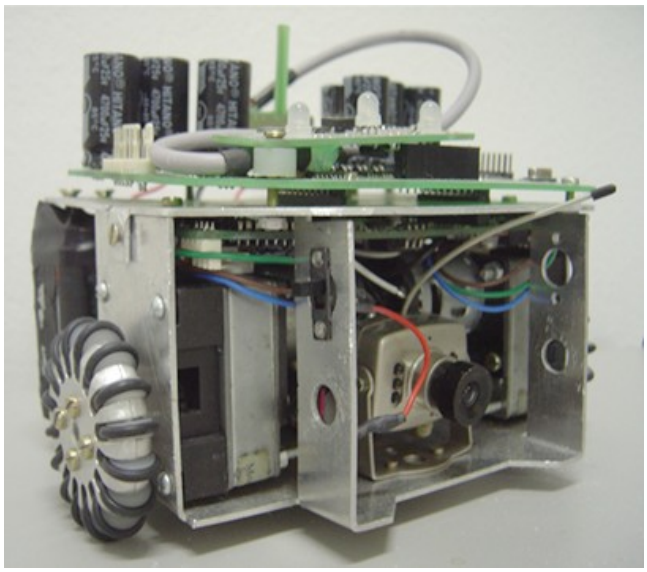

Fig. 3. Robô utilizado para os testes

3. CODIFICAÇÃO E COMPRESSÃO DE VÍDEO

Antes da transmissão do vídeo através da rede, o mesmo precisa ser digitalizado e comprimido. A necessidade por digitalização é óbvia: os computadores transmitem bits na rede, assim, toda informação precisa ser representada como uma seqüência de bits. A compressão é importante porque o vídeo não comprimido consome uma quantidade muito grande de armazenamento e largura de banda. Um vídeo é uma seqüência de imagens, normalmente exibidas a uma taxa constante, por exemplo a 24-30 imagens por segundo (fps – *frame per second*). Uma imagem digital não comprimida consiste em uma matriz de pontos, sendo cada ponto codificado em um número de bits que representam luminosidade e cor.

O padrão de compressão conhecido por MPEG é a técnica de compressão mais popular. Este inclui o MPEG 1 para vídeo com qualidade de CD-ROM (1.5 Mbps), MPEG 2 para vídeo com qualidade de DVD (3-6 Mbps), e MPEG 4 para compressão de vídeo orientada a objetos. O padrão de compressão H.261 que faz parte da arquitetura de protocolos H.323 também é muito popular na Internet. Existem ainda inúmeros padrões de compressão proprietários. O software utilizado para Codificação e Compressão de vídeo foi o *FFmpeg* [23] que já vem com um servidor de *streaming* para disponibilizar para os usuários. Ele também consegue capturar de uma fonte ao vivo de áudio/vídeo. O *FFmpeg* é uma solução completa para gravar, converter e transmitir áudio e vídeo. Este pacote inclui a biblioteca *libavcodec*, responsável pelos *codecs* de áudio/vídeo. O *FFmpeg* é desenvolvido sob *GNU/Linux*.

Com a técnica de *Streaming* o sinal de vídeo é transmitido ao cliente e sua apresentação inicia-se após uma momentânea espera para armazenamento dos dados em um *buffer* [22]. Nesta forma de transmitir vídeo não é preciso fazer o *download* prévio do arquivo, o computador do cliente vai recebendo as informações continuamente enquanto são mostrados ao usuário. Esta técnica reduz o tempo de início da exibição e também elimina a necessidade de armazenamento local do arquivo. Transmissões eficazes desses sinais de vídeo através de redes com baixa largura de banda requerem uma alta taxa de compressão de dados para garantir a qualidade visual da apresentação. A técnica de compressão mais comum atualmente é conhecida por MPEG. A Internet possui hoje uma grande variedade de aplicações multimídia das quais, três estão voltados à transmissão de vídeo em tempo real na rede (*Streaming* de vídeo armazenado, *Streaming* de vídeo ao vivo e *Streaming* de vídeo interativo em tempo real). O tipo de *Streaming* utilizado neste projeto foi o de vídeo ao vivo.

Este tipo de aplicação é similar à tradicional transmissão de rádio e televisão, conhecida como *broadcast*, na qual o cliente assume uma posição passiva e não controla quando o *stream* começa ou termina. A única diferença está no fato dessa transmissão ser realizada através da Internet. Tais aplicações permitem ao usuário receber um sinal de rádio ou televisão ao vivo que foi emitido de qualquer parte do mundo. Neste caso, como o *streaming* de vídeo não é armazenado em um servidor, o cliente não pode controlar a exibição da mídia. Neste tipo de transmissão podem existir muitos clientes recebendo o mesmo conteúdo simultaneamente com a sua distribuição ocorrendo de duas formas:

1. **Unicast**: é uma conexão ponto-a-ponto entre o cliente e o servidor, na qual cada cliente recebe seu próprio *stream* do servidor. Dessa forma, cada usuário conectado ao *stream* tem sua própria conexão e os dados vêm diretamente do servidor;
2. **Multicast:** ocorre quando o conteúdo é transmitido sobre uma rede na qual todos os clientes na rede compartilham o mesmo *stream*. Assim, preserva-se largura de banda, podendo ser extremamente útil para redes locais com baixa largura de banda.

O caminho do vídeo no *streaming*, desde a captura da imagem até sua visualização na tela do cliente, pode ser descrito a seguir. Inicialmente, um software codifica a imagem capturada em um formato apropriado para a transmissão. Então, um software de *streaming* abre um ou mais fluxos de transmissão para enviar o vídeo codificado. Finalmente, um software cliente recebe o fluxo de vídeo e o exibe no computador do usuário. Os softwares empregados neste caminho são brevemente descritos a seguir.

- **Encoder**: Esse é o software que roda no 'computador fonte', e serve para converter o vídeo da placa de captura para o formato na qual a mídia foi transmitida pela rede. Caso se queira transmitir para a Internet, é fortemente recomendado ter uma conexão com a internet de pelo menos 128kbps de *upload*.
- **Servidor**: Esse é o software que roda no 'servidor', e serve para distribuir o *streaming* de vídeo para vários clientes ao mesmo tempo. O ideal é que esse software rode numa máquina que tenha uma conexão com a Internet muito boa. O software *ffserver* pode ser executado no mesmo computador que está rodando o encoder *ffmpeg*.
- **Player**: Esse é o software que é executado no 'cliente', ou seja, no computador de quem irá assistir ao *streaming*.

4. METODOLOGIA E RESULTADOS

Para a instalação do servidor do LRD, foi utilizado o sistema operacional Linux, configurado com o *driver* da placa de aquisição de imagens Pixelview, o servidor web APACHE, o ambiente PHP e o banco de dados MYSQL. A distribuição Linux escolhida foi Slackware 10.0 com o Kernel 2.6.13. No início de sua instalação já é adicionado o Servidor web (Apache), PHP e MYSQL que foi utilizado no Laboratório. O Linux já possui um conjunto de *drivers* para o gerenciamento de placas de vídeo com o *chipset* da família Bt8xx. Desta forma, somente foi preciso conectar a placa PCI no computador para que o Linux a reconhecesse automaticamente e selecionasse o *driver bttv* apropriado para o seu funcionamento. Os comandos utilizados para verificar o *driver bttv* foram:

`#dmesg |grep bt:` para verificar se o driver está instalado.
`#modinfo -d bttv:` para verificar se o kernel possui suporte ao bttv.
`#bttv - v4l driver module for bt848/878 based cards:` para verificar se é necessário recompilar o kernel do linux.
`#lsmod |grep bt:` para verificar quais módulos estão instalados.

`# lspci:` para a verificação do driver instalado.

O *FFmpeg* é um software de conversão de arquivos de vídeo e áudio que funciona em linha de comando (sem interface gráfica). Algumas das funções do aplicativo são: converter arquivos de vários formatos, compactá-los se preciso e redimensionar a tela. Ele possui os seguintes componentes em sua solução:

- *ffmpeg* é um utilitário para a linha de comando que possui a função de converter um formato de vídeo para outro. Vale à pena lembrar que este programa suporta também a captura e codificação em tempo real de uma placa de TV ou webcam.
- *ffserver* é um programa servidor para transmitir de audio e vídeo via protocolo HTTP.
- *ffplay* é um *player* simples para o modo console que utiliza a biblioteca SDL com as bibliotecas *FFmpeg*.
- *libavcodec* é uma biblioteca que contém todos os codificadores e decodificadores de áudio/vídeo do pacote *FFmpeg* .
- o *libavformat* é uma biblioteca que contém analisadores e geradores de diversos formatos de áudio e vídeo.

Para a instalação desse software, foi necessário adquirir seu pacote e descompactá-lo em um diretório apropriado do Linux, o que pode ser efetuado pelos comandos abaixo:

```
$ tar -zxvf ffmpeg-0.4.9-pre1.tar.gz
$ cd ffmpeg-0.4.9-pre1/
```

Para a configuração do software, foi necessária a criação um arquivo de configuração para o *FFServer*. Isto é efetuado por meio do comando: `$ ./configure.`

Nesse arquivo, deve-se indicar qual o tipo de compressão será utilizado para a transmissão de vídeo. Durante a configuração do software, é criado um arquivo denominado *ffserver.conf,* que contém um *link http* para o servidor de transmissão. Este link contém um endereço e alguns parâmetros que são utilizados pelo servidor do *streaming* e deve ser acessada via browser pelo usuário para iniciar o processo de transmissão do vídeo. O conteúdo do link de acesso da transmissão é mostrado abaixo:

```
$ ffplay http://201.27.36.89:8090/test.rm -x 160 -y 120
```

O protótipo do software de interface WEB do LRD possui um visualizador media *player* que disponibilizará ao usuário o *streaming* transmitido pela câmera do laboratório. Adicionalmente, este software possui uma barra de controle de movimento do robô: frente, trás, rotação à direita e rotação à esquerda, incluindo opções para exibir *logs* das ações do robô. O desenvolvimento do protótipo de Interface WEB seguiu os processos de Engenharia de Software, garantindo a consistência e o cumprimento dos objetivos propostos no trabalho. O software foi implementado em linguagem XHTML, CSS, Javascript, AJAX e PHP, com chamadas a funções da linguagem C++. Na Fig. 4 é mostrado o protótipo de interface web do LRD.

Este protótipo foi desenvolvido com o objetivo de validar a tecnologia a ser empregada no LRD final e contém apenas as funcionalidades que representam maior risco tecnológico para o laboratório, que são a transmissão de vídeo e de comandos do robô pela Internet. Portanto, este protótipo não apresenta todas as funcionalidades que o laboratório possuirá.

Para a captura da imagem de vídeo do LRD pela Intranet/Internet, foi utilizada uma placa de captura da

Pixelview, modelo PlayTV MEPG2 [19] do Fabricante Prolink. Essa escolha deve-se ao chip Conexante Fusion 878A, que é um dispositivo multifuncional PCI baseado nos produtos BT878/879, que é compatível com a plataforma Linux. Para o acionamento da placa de captura utilizada, foi empregado o *driver bttv*. Este *driver* constitui uma interface entre a placa de aquisição de vídeo e a API Vídeo-4-Linux, que é utilizada para a aquisição da imagem capturada pela placa.

Fig. 4. Interface do sistema do LRD

A aquisição de imagens do LRD é realizada por meio da API denominada Vídeo-4-Linux [25]. Essa API constitui uma interface com placas de captura de vídeo, sintonia de sinal de TV, rádio e teletexto. Ela surgiu por volta de 1998 (Alan Cox), sendo incorporada ao Kernel 2.1.x do Linux. Essa API é utilizada pelo software de captura de imagens e transmissão de vídeo empregado neste projeto: o *Ffmpeg*.

4.1. Controle do robô

O controle do robô é realizado mediante a execução do comando *"mov"* (movimento) na mesma máquina do servidor web, que possui um modulo rádio-base conectado em uma das suas portas USB (USB 1.1 – 2.0). O comando *"mov"* foi implementado em C++ e aceita três parâmetros básicos: *forward*, *backward*, *right* e *left* que permitem a realização da movimentação para frente, para trás, para a direita (rotação) e para esquerda (rotação) respectivamente. O movimento indicado pelo parâmetro é realizado pelo robô durante um breve intervalo de tempo predeterminado.

A execução do comando *"mov"* provoca o envio de uma mensagem pela porta USB para o dispositivo de comunicação por rádio (módulo rádio-base), o qual se encarrega transmití-lo ao robô. O comando *"mov"* precisa ser executado com privilégios de *root* para poder interagir com a porta USB.

O protótipo desenvolvido utiliza apenas um robô, mas existe a possibilidade de comunicação com mais de um robô. Por isso, cada robô tem associado um endereço interno único que pode ser utilizado durante a transmissão de dados. Isto possibilita o encaminhamento de mensagens pelo sistema aos diferentes robôs de forma independente. O sistema pode enviar mensagens ao robô a qualquer momento, mas o robô nunca pode tomar a iniciativa em uma comunicação, podendo enviar informação ao sistema apenas quando solicitado. A funcionalidade de comunicação do PC com o robô é fornecida pela biblioteca *"libcomunica.a"* (para C++). Os comandos do robô são executados por meio de um *script* PHP existente na interface do LRD. Este *script* recebe a indicação do botão pressionado (*direcao*), e executa uma chamada do movimento *"mov"* com o parâmetro apropriado. Este *script* de comandos é mostrado na Fig. 5.

```
1  <?php header('Content-type: application/xml; charset=UTF-8;');
2
3
4  $direcao = $_POST["direcao"];
5
6  if ($direcao == "frente") {
7
8     echo "FRENTE";
9     exec('sudo /var/www/htdocs/LRD/mov forward');
10
11 } else if ($direcao == "atras") {
12
13    echo "ATRAS";
14    exec('sudo /var/www/htdocs/LRD/mov backward');
15
16 } else if ($direcao == "direita") {
17
18    echo "DIREITA";
19    exec('sudo /var/www/htdocs/LRD/mov right');
20
21 } else if ($direcao == "esquerda") {
22
23    echo "ESQUERDA";
24    exec('sudo /var/www/htdocs/LRD/mov left');
25 }
```

Fig. 5. *Script* de chamada dos comandos do robô

5. CONSIDERAÇÕES FINAIS

O desenvolvimento do protótipo do LRD foi um importante passo no sentido de se desenvolver um projeto futuro de um Laboratório de Robótica a Distância. Os testes de acionamento e movimentação do robô, bem como a transmissão de vídeo, foram concluídos com êxito. No trabalho proposto, a transmissão da imagem foi realizada dentro a Intranet da infra-estrutura de rede da incubadora de empresas, devido à estrutura e a política de segurança da mesma, não permitindo a transmissão do vídeo para acesso remoto. Entretanto, a tecnologia de transmissão para Internet é a mesma empregada para Intranet e mostrou-se satisfatória. A transmissão de comandos para o robô foi realizada por meio de chamadas remotas de biblioteca localizada no servidor do LRD. Essa arquitetura de funcionamento evita o tráfego de grande quantidade de dados pela Intranet/Internet e possibilita a escala do LRD para múltiplos usuários.

O protótipo desenvolvido mostrou a viabilidade de uso das tecnologias de transmissão de imagens e comandos via Intranet/Internet, reduzindo o risco tecnológico para um projeto de grande escala. Além disso, a arquitetura do LRD e os módulos utilizados no desenvolvimento deste protótipo permitem a extensão do laboratório para o uso com múltiplos robôs.

AGRADECIMENTOS

Os autores agradecem o apoio financeiro da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) por meio do programa de Inovação Tecnológico em Pequenas Empresas (Processo no. 05/55125-6).

REFERENCIAS BIBLIOGRÁFICAS

[1] Transmissão de Vídeo pela Internet e Web TVs, http://docs.indymedia.org/view/Sysadmin/WebTVPt#DarwinStreamingServer [Acesso em Jun/2009]

[2] Instalação do software XAWTV, http://linux.bytesex.org/xawtv [Acesso em Mai/2009]

[3] Instalação do software TVTIME, http://tvtime.sourceforge.net/downloads.php [Acesso em Jun/2009]

[4] Instalação da Placa de Captura, http://www.vivaolinux.com.br/artigos/ verArtigo.php?codigo=3940&pagina=3 [Acesso em Jun/2009]

[5] Instalação do driver BTTV, http://linux.bytesex.org/v4l2/bttv.html [Acesso em Jun/2009]

[6] Perez Ávila, R. (2004). Streaming: Crie sua Própria Rádio Web e TV Digital, Ed. Visual Books, Santa Catarina.

[7] http://pt.wikipedia.org/wiki/Slackware_Linux [Acesso em Mar/2009]

[8] http://httpd.apache.org/ [Acesso em Mar/2009]

[9] Descrição do Protocolo HTTPS: http://pt.wikipedia.org/wiki/HTTPS [Acesso em Mar/2009]

[10] XHTML: http://pt.wikipedia.org/wiki/XHTML [Acesso em Jun/2009]

[11] CSS: http://www.maujor.com/ [Acesso em Jun/2009]

[12] JAVASCRIPT: http://pt.wikipedia.org/wiki/JavaScript [Acesso em Jun/2009]

[13] AJAX: http://pt.wikipedia.org/wiki/AJAX_(Web) [Acesso em Jun/2009]

[14] PHP: http://www.php.net/ [Acesso em Jun/2009]

[15] TCP/IP: http://pt.wikipedia.org/wiki/TCP/IP [Acesso em Mar/2009]

[16] Manual On-line do Ffmpeg, contendo comandos, http://www.estudiolivre.org/tiki-index.phppage=manual+do+FFMPEG [Acesso em Jun/2009]

[17] Placa de Vídeo PixelView. http://www.pixelview.com.br/ [Acesso em Ago/2009]

[18] Tecnologia Streaming: http://pt.wikipedia.org/wiki/Streaming_Media [Acesso em Ago/2007]

[19] Fabricante da Placa de Aquisição: http://www.pixelview.com.br/play_tv_mpeg2.asp [Acesso em Jun/2007]

[20] Formato MPEG: http://penta3.ufrgs.br/videoconferencia/manual/mpeg.htm [Acesso em Jun/2007]

[21] http://pt.wikipedia.org/wiki/Patrick_Volkerding [Acesso em Ago/2007]

[22] http://pt.wikipedia.org/wiki/Buffer [Acesso em Ago/2007]

[23] FFMpeg. http://ffmpeg.mplayerhq.hu/ [Acesso em Jun/2007]

[24] Transmissão de Vídeo pela Internet. http://www.img.lx.it.pt/~fp/cav/ano2005_2006/Trabalho_2/ home.htm [Acesso em Jun/2007]

[25] Vídeo for Linux. http://linuxtv.org/v4lwiki/index.php/Main_Page [Acesso em Jun/2007]