UNIVERSIDADE POSITIVO
NÚCLEO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
ENGENHARIA DA COMPUTAÇÃO

AMBIENTE EXPERIMENTAL PARA CIRURGIA À DISTANCIA

Sebastián Nahuel Fantini
Wagner Mariotto Bonfiglio

Monografia apresentada à disciplina de Trabalho de Conclusão de Curso como requisito parcial à conclusão do Curso de Engenharia da Computação, orientada pelo Prof. Alessando Brawerman.

UP/NCET
Curitiba
2008

TERMO DE APROVAÇÃO

Sebastián Nahuel Fantini
Wagner Mariotto Bonfiglio

Ambiente experimental para cirurgia à distância

Monografia aprovada como requisito parcial à conclusão do curso de Engenharia da Computação da Universidade Positivo, pela seguinte banca examinadora:

Prof. Alessando Brawerman (Orientador)

Prof. Edson Pedro Ferlin

Prof. Mauricio Perretto

Curitiba, 8 de Dezembro de 2008.

AGRADECIMENTOS

Agradecemos aos nossos familiares, professores e amigos que nos ajudaram direta ou indiretamente na realização deste trabalho.

RESUMO

Este projeto consiste no desenvolvimento de um ambiente experimental propício à realização de cirurgias à distância. Parte deste estudo incide na conexão de um *joystick* a um computador conectado à Internet e, com isto, o médico será capaz de operar um paciente a quilômetros de distância utilizando um braço robótico. Para tanto é necessário que o paciente encontre-se em um ambiente que disponha de um computador também conectado à Internet e um robô disposto de maneira a possibilitar o procedimento desejado. Além disto, devem ser dispostas câmeras que transmitem as imagens da sala cirúrgica e do paciente, via Internet, para o médico no local remoto.

A aquisição dos movimentos necessários para que o especialista realize as tarefas por meio de um robô é feita, no computador do especialista, por intermédio de um *software* que recebe comandos vindos de um *joystick*. Esses comandos são enviados para o computador que está próximo ao paciente, utilizando-se uma VPN (*Virtual Private Network*). Nessa fase, o computador interpreta a informação recebida e envia os comandos para a interface que os traduz de uma forma que o braço robô entenda e realize os movimentos aplicados pelo médico.

Este projeto mostra a possibilidade de se ter um ambiente cirúrgico à distância visando apresentar o cenário que possa ser aplicado posteriormente em um desenvolvimento mais complexo. Isso se deve ao fato de reconhecer que aspectos que envolvam uma cirurgia, tais como ambiente limpo, precisão de movimentos, velocidade de transmissão de dados e possíveis complicações no decorrer do procedimento cirúrgico devem ter cuidados especiais e auxílio de profissionais de outras áreas.

Palavras chave:

Cirurgia à distância, VPN, *joystick*, robótica.

Beginning of an Environment for Telesurgery

ABSTRACT

This project is initially to develop an experimental environment propitious to do telesurgeries. Parto f this study involves the connection of a joystick to a computer connected to the Internet, and thus, the doctor will be able to operate a patient remotly using a robotic arm. This requires the patient to be situated in an environment that has a computer, also connected to the Internet, and a robot arranged in a way to enable the desired procedure. Besides, the webcams should be displayed to transmit surgery's room and patient images, over the Internet, to the medic in remote place.

The acquisition of movements necessary for the specialist perform the tasks using a robot is made in the medic computer, via a software that receives commands from a joystick. These commands are sent to the computer that is close to the patient, using a VPN (Virtual Private Network). At that stage, the computer interprets the information it receives and sends commands to the interface that translates in a way that the robot arm understand and perform the movements applied by the physician.

This project shows the possibility of having a surgical environment for distance present the scenario that targeting can be implemented later in a development more complex. This is due to the fact recognize that aspects involving a surgery, such as clean environment, precision of movement, speed of data transmission and possible complications during the surgery must take special care and assistance of professionals from other areas.

Key words:

Telesurgery, VPN, joystick, robotic

SUMÁRIO

LISTA DE FIGURAS

LISTA DE SIGLAS

NCET - Núcleo de Ciências Exatas e Tecnológicas

UP – Universidade Positivo

VPN – *Virtual Private Network*

USB – *Universal Serial Bus*

API – *Application Program Interface*

PC – *Personal Computer*

ROM – *Read Only Memory*

RAM – *Random Access Memory*

CPU – *Central Process Unit*

TCP - *Transmission Control Protocol*

IP – *Internet Protocol*

LED – *Light Emitting Diode*

SDK – *Software Development Kit*

PIC – *Periferal Interface Controler*

DIP – *Dual In-Line Package*

EEPROM - *Electrically-Erasable Programmable Read-Only Memory*

PWM – *Pulse Width Modulation*

VNC – *Virtual Network Computing*

LISTA DE SÍMBOLOS

mA – mili Amperes

MIPS – milhões de instruções por segundo

V - volts

ms - milisegundos

CAPÍTULO 1 - INTRODUÇÃO

Nos últimos anos a quantidade de cirurgias realizadas no mundo vem crescendo em grande escala. Segundo Lui (2004), haverá um aumento de 14,7% no número de procedimentos cirúrgicos entre os anos de 2000 e 2010 no mundo e de 31,5% até 2020. Um grande problema que a sociedade enfrenta em relação a isso é que a infra-estrutura e a quantidade de especialistas por local não acompanham proporcionalmente este crescimento. Muitas vezes pacientes precisam fazer pequenos procedimentos cirúrgicos e não encontram seus médicos regulares no local. Outras vezes estes mesmos pacientes preferem enfrentar uma viagem e serem tratados no exterior. Quando o problema é muito grave e o paciente já não tem condições físicas, a viagem deixa de ser uma alternativa viável.

Este projeto tem por objetivo a criação de um ambiente experimental para realização de cirurgia à distância, evitando dessa forma a necessidade de viagem ou até mesmo o deslocamento do médico. O trabalho consiste no desenvolvimento de um conjunto de *hardware* e *software* capaz de simular o ambiente cirúrgico. Neste projeto, de cunho científico e experimental, o objetivo é fornecer um braço robótico que é controlado por um médico à distância por meio de uma rede privada e um *joystick*. A visualização do ambiente cirúrgico pelo médico será possível através de uma *WebCam* que transmite imagens do ambiente cirúrgico e do paciente via VNC (*Virtual Network Computing*).

A principal motivação para realizar este projeto é a grande evolução que certamente haverá nesta área. A cirurgia à distância pode ser de grande utilidade para a sociedade, resolvendo grande parte dos problemas citados anteriormente. Este projeto visa abrir as portas para que muitos outros trabalhos sejam feitos em cima deste, aperfeiçoando-o e diminuindo a necessidade da presença do médico no ambiente cirúrgico.

O restante desta monografia está dividido da seguinte maneira: o capítulo 2 apresenta conceitos básicos para formar uma fundamentação teórica e propiciar um melhor entendimento ao leitor. O capítulo 3 mostra alguns trabalhos realizados em áreas relacionadas a este projeto para que seja possível uma análise dos estudos já feitos nas áreas de interesse. No capítulo 4 é apresentada a especificação do projeto, explicando o desenvolvimento do *software* e do *hardware* e como cada bloco do projeto se integra para o resultado final. O detalhamento de cada etapa do desenvolvimento e a implementação do projeto é parte do capítulo 5, deixando os resultados obtidos para o capítulo 6. Ainda fazem parte desta monografia, apêndices, manual técnico e do usuário, artigo e a tabela de custos.

CAPÍTULO 2 – FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são explicados os conceitos teóricos, utilizados no projeto desenvolvido, para possibilitar um maior entendimento dos assuntos tratados.

2.1- Virtual Private Network (VPN)

Em uma comunicação normal entre dois computadores, via Internet, os dados trocados entre eles estão vulneráveis a interceptação e modificação da informação, dentre outros casos que podem prejudicar a correta troca de informações (GUIMARÃES, 2006). Quando se têm um sistema crítico, ou seja, que necessita da segurança dos dados (não permitindo que o dado seja lido, modificado nem perdido), é preciso procurar uma solução que garanta a entrega segura dos dados ao destino.

Um método para que tudo aconteça sem problemas é ligar os dois computadores diretamente, sem passar pela Internet. Para realizar este tipo de ligação entre computadores muito distantes a utilização deste método fica inviável.

Outra solução, mais indicada para o caso supracitado, é a VPN (*Virtual Private Network*), método que utiliza o princípio de tunelamento que isola o caminho a ser percorrido pela informação na rede pública (como por exemplo, a Internet). Resumindo, a VPN acaba criando uma rede privada dentro de uma rede pública.

Este método oferece uma série de benefícios, como por exemplo a garantia que os dados que trafeguem entre as partes envolvidas sejam totalmente privados, de forma que caso sejam capturados não possam ser entendidos. Provê também autenticação, permitindo que somente os usuários autorizados a fazer parte da VPN realizem a troca de informações, além de integridade dos dados, garantindo que o destino descubra alterações nos dados recebidos, caso haja alguma mudança.

Neste trabalho foi implementada uma rede virtual privada (VPN) interligando os computadores do médico e do robô, já que envolve informações críticas que precisam de proteção e não podem simplesmente ser transmitidas em uma rede pública insegura.

Na VPN existe ainda o processo de tunelamento, que pode ser entendido como o encapsulamento de um protocolo dentro de outro (SILVA, 2002). Primeiramente, o pacote a ser transmitido é criptografado, ficando ilegível caso seja interceptado no meio da transmissão. O pacote, já criptografado, é encapsulado com um cabeçalho adicional que contém informações de roteamento os quais permitem a transferência do pacote ao longo da rede. Por fim o pacote é transportado através da Internet até chegar ao destino onde é desencapsulado e descriptografado,

voltando à sua forma original. O caminho lógico a ser percorrido pelo pacote na rede é chamado de túnel. Na Figura 1 observa-se o processo de tunelamento, em que um pacote recebe o cabeçalho adicional e passa pelo túnel, já devidamente criptografado como foi explicado acima, até chegar ao destino que reconhece o cabeçalho adicional e utiliza o pacote em sua forma natural.

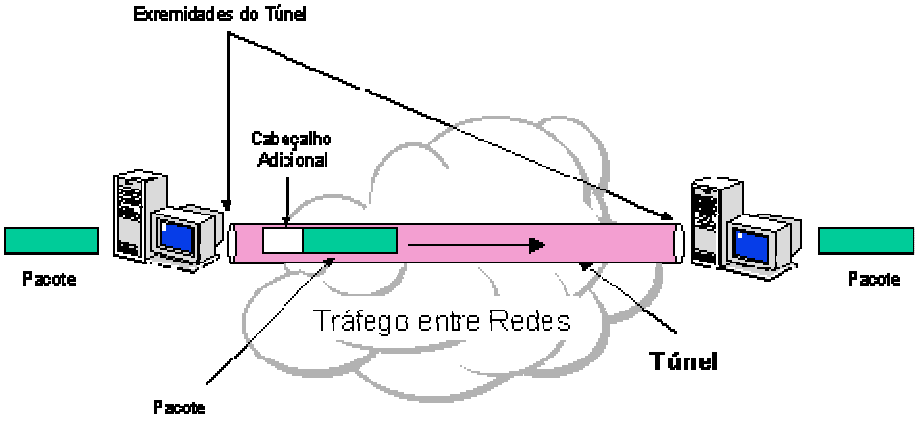

Figura 1. Método de tunelamento
FONTE: LIOU (1998).

2.2- Robótica

A robótica vem evoluindo de forma muito acelerada nos últimos anos, sendo visíveis as melhorias e novas tecnologias que surgem a cada dia nesta área. Desde que Isaac Asimov escreveu o livro "Eu, robô" (ASIMOV, 2004), publicado pela primeira vez em 1950, houve uma grande evolução nas pesquisas sobre o assunto. Robôs com uma inteligência quase natural, muito visto no livro de Asimov, não despontam como um caminho natural a ser seguido, porém, cada vez mais, os seres guiados por engrenagens vêm tomando o lugar do homem em diversas tarefas.

Hoje é possível ver um robô substituindo diversos homens na linha de produção de uma fábrica, assim como realizando tarefas impossíveis de serem realizadas por seres humanos em ambientes extraterrestres. Também é possível observar homens e robôs trabalhando em conjunto em atividades que necessitam de sensibilidade humana associada à precisão robótica.

Um braço mecânico pode ter diversas configurações, variando o número e tipo das juntas, as condições suportadas (calor, umidade, etc.), o peso suportado e outras variáveis definidas de acordo com o projeto (GORDON, 2006).

Robôs inteligentes são aqueles que possuem inteligência embutida e são capazes de interagir com o meio utilizando sensores para tomar decisões em tempo real.

Robôs com controle por computador não têm a capacidade de interagir com o ambiente e são manipulados por sistemas multifuncionais controlados por computador.

Os robôs de aprendizagem são limitados a simplesmente repetir uma seqüência de movimentos indicados pela intervenção de um operador ou memorizados.

Finalmente, existem os robôs manipuladores, os quais são sistemas mecânicos multifuncionais que possuem um sistema de controle sensível e tem o movimento de seus membros manualmente quando o operador controla diretamente os movimentos.

2.3- Microcontrolador PIC18F452

Um mircrocontrolador reúne no mesmo chip diversos elementos de um microcomputador: microprocessador, memórias RAM e ROM, temporizadores, contadores, canal de comunicação serial e portas de entrada e saída. Os microcontroladores PIC (*Periferal Interface Controler*) fabricados pela Microchip, são muito utilizados quando é necessário um sistema embarcado no projeto (SILVA, 2006).

O microcontrolador escolhido no projeto foi o PIC18F452 já que este possui uma simplicidade de aplicação, memórias RAM e ROM suficientemente dimensionadas para armazenar o código do projeto além de ser o modelo de uso geral da família 18F de microcontroladores PIC.

Esta família de PIC tem como vantagem o fato de possuir mais instruções em código de máquina (75 contra 35 da séria 16F) que é otimizada para ser usada com compiladores de linguagem C. Além disso, possui uma velocidade de processamento alta, na maioria até 10 MIPS (a 40MHz de *clock*) com alimentação entre 2 e 5,5V. O seu encapsulamento DIP consiste em 40 pinos (SOUZA, 2007).

Este modelo de microcontrolador possui 32K bytes de memória de programa flash, 1536 bytes de memória RAM e 256 bytes de memória EEPROM.

O PIC18F452 é usado no projeto para fazer a comunicação serial e controlar os servo-motores. A programação do Firmware foi feita na linguagem C.

2.4- Servo Motor

O servo motor é um dispositivo eletromecânico que, a partir de um sinal elétrico na sua entrada, pode ter seu eixo posicionado em uma determinada posição angular. Por ser pequeno e compacto, além de permitir um posicionamento preciso de seu eixo, o servo-motor é largamente utilizado em robótica e modelismo (SANTOS, 2007).

O funcionamento básico do servo-motor utiliza um circuito de controle e um potenciômetro ligado ao eixo de saída. Utilizando este potenciômetro, o circuito de controle pode monitorar o ângulo do eixo do servo-motor. Se o ângulo no qual o eixo se encontra for o ângulo desejado, o motor pára. Caso contrário, o motor é ativado e começa a girar até que a posição desejada seja obtida. O ângulo de giro do eixo varia de 0 a 180 graus, sendo estes limites definidos através de um limitador nas engrenagens do servo-motor.

Um servo-motor é constituído de diversas partes explicadas a seguir:

- Circuito de controle: é responsável por receber os sinais e energia do receptor, verificar o ângulo atual do eixo do servo-motor e controlar o motor tendo como referência a posição do potenciômetro e o sinal receptor.
- Potenciômetro: ligado ao eixo de saída do servo-motor é usado para monitorar a posição do mesmo.
- Motor: é o encarregado de movimentar as engrenagens e o eixo principal do servo.
- Engrenagens: graças a elas transfere-se mais força ao eixo principal de saída. Elas movimentam o potenciômetro junto com o eixo.
- Caixa do servo.

Para definir a posição em que cada servo-motor deve permanecer, deve-se definir a largura dos pulsos mandados através de uma técnica conhecida como PWM. Este sinal pode ter 0 Volts ou 5Volts e possui um período igual a 20ms. Quando é detectado um sinal de subida (isto é, de 0V para 5V), durante 1ms até 2ms, o servo-motor deve verificar a posição atual e se dirigir para a posição desejada dependendo da duração deste pulso.

Como é demonstrado na Figura 2, com um pulso de 1ms o servo-motor deve ir para a posição de 0 graus. Caso o pulso seja 2ms vai pra 180 graus, podendo ser usado qualquer valor intermediário entre 1ms e 2ms para posições proporcionalmente intermediárias.

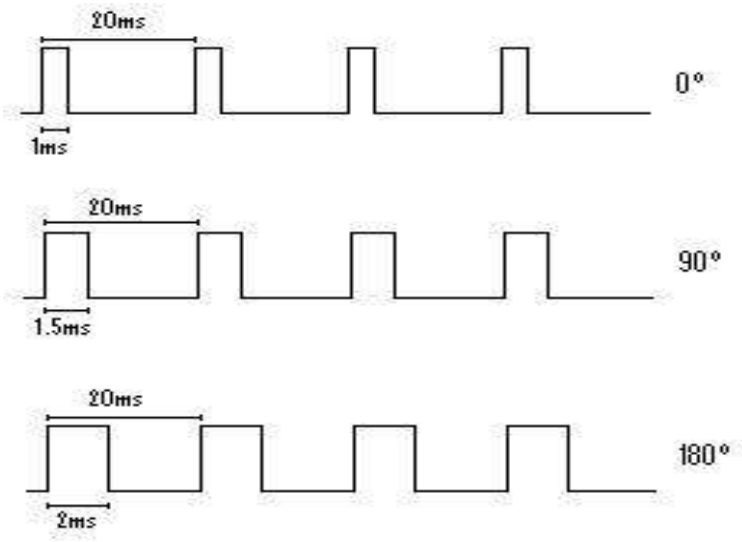

Figura 2. Pulso PWM

O modelo de servo-motor utilizado no projeto para a construção do robô foi o HS-422 da Hitech observado na Figura 3. Neste projeto foram utilizados seis servo-motores dispostos de maneira a simular um braço robótico.

Figura 3. Servo Motor HS-422

2.5- VNC

VNC (*Virtual Network Computing*) é um protocolo que permite visualizar e interagir remotamente com a área de trabalho de qualquer computador do mundo. Está disponível praticamente para todas as plataformas, tendo versões do VNC para Linux, Windows, MacOS, Solaris, BeOS, Amiga e até mesmo para *palmtops* com o Windows Mobile ou com o Palm OS.

Pode-se usá-lo para acessar remotamente computadores conectados na mesma rede local ou até mesmo na Internet.

O VNC se divide em dois módulos: o módulo servidor e o módulo cliente. O módulo servidor deve ser instalado no computador que ficará acessível para compartilhar sua área de trabalho. Esta área poderá ser acessada usando o modulo cliente desde qualquer outro computador. Um fato interessante que diz respeito à flexibilidade é que os módulos são intercompátiveis, ou seja, é possível acessar um computador rodando Linux a partir de outro que roda Windows, por exemplo.

O programa mostra na tela do cliente exatamente o mesmo que se encontra no servidor que está sendo acessado, dando a impressão que está-se "sentados na frente do outro computador". Este mecanismo é muito utilizado pelas equipes de suporte para poderem observar exatamente a tela do usuário que esta tendo problemas.

Para que o médico visualize o ambiente cirúrgico, uma VNC é criada entre os computadores para que a imagem da *WebCam* que aparece na tela do computador servidor seja visualizada no computador cliente.

2.6. Cliente/Servidor

A arquitetura Cliente/Servidor é um modelo computacional em que o processamento da informação é dividido entre um computador cliente e um computador servidor (GUIMARÃES, 2006). Ambas as partes devem estar conectadas entre si mediante uma rede. O cliente geralmente envia requisições para o servidor e este processa a informação, devolvendo alguma resposta de acordo com as informações mandadas pelo cliente.

No caso deste projeto, o cliente envia os comandos captados do *joystick* e o servidor analisa estas informações para enviar à interface o comando desejado pelo especialista.

CAPÍTULO 3 – TRABALHOS RELACIONADOS

Com o objetivo de se aprofundar no assunto do projeto e visar um maior entendimento dos temas, foram consultados diversos trabalhos relacionados ao assunto em questão. Segue abaixo, a descrição dos trabalhos mais significativos:

O projeto Integration of the Robotic Arm Control System (WU et al, 1994) consiste na substituição da interface VAL controller, do robô PUMA 562, por uma interface própria, a Robotic Arm Control System (RACS). Esta nova interface tem como objetivo realizar testes avançados e implementar novos algoritmos para controlar o robô com o propósito de obter movimentos mais precisos. A RACS pode ser utilizada em conjunto com a interface VAL controller, assim como de forma independente. Este projeto é uma boa opção para se obter melhorias em relação às interfaces padrões dos robôs, que muitas vezes podem apresentar algumas limitações. No caso do projeto de cirurgia à distância, a melhoria desejada não seria obter somente avanços em relação à precisão, mas também em funcionalidade, permitindo a manipulação do robô via Internet.

O projeto Joystick para controle do braço mecânico ED-7220C (SEVERINO, 2005) tem como objetivo a elaboração de um joystick e uma interface para controle do braço-robô ED-7220C da Ed-Laboratories. Foi feita também uma interface de comunicação, a qual capta os dados do joystick e os envia a um PC. Este contém um software que analisa os dados e posteriormente os envia à interface do braço-robô. Neste trabalho de controle do ED-7220C, o hardware desenvolvido faz o tratamento dos comandos vindos diretamente do joystick. Entretanto, no trabalho descrito nesta monografia, o hardware recebe os comandos do PC e os envia aos servos constituintes do robô. Além disso, existe a implementação de uma rede VPN, possibilitando o controle do robô à distância.

O projeto Web Controled Robotic Arm (MARQUES et al, 2005) visa a movimentação de um braço-robô através da Internet usando um web browser comum. O usuário faz a entrada dos comandos através do mouse e do teclado em uma página web, que também oferece a visualização do robô por meio de uma webcam. É utilizada a placa SBC Rabit 2000 TCP/IP, a qual pode se conectar diretamente à Internet eliminando a necessidade de um PC do lado do robô. Tendo em vista que o objetivo deste projeto é o ensino da robótica com a possibilidade de ser inserido num sistema global de e-learning, a precisão dos movimentos não é algo fundamental e por isso optou-se pelo controle pelo teclado por meio de uma página web.

Por sua vez, o projeto Robotic Telementoring/Telesurgical System and Randomized Evaluation Study (PATRICIU et al, 2005) consiste em um sistema de ensino à distância de

técnicas cirúrgicas, usando comunicação de áudio e vídeo, e controle de um robô remotamente responsável por fazer a cirurgia propriamente dita. O sistema pode ser usado seguramente em incisões percutâneas ao rim. Em um ambiente distante da sala de operação, o especialista comanda os movimentos do PAKY-RCM Robot. Existe uma equipe de médicos qualificados na sala cirúrgica os quais assumem o controle do robô em caso de falha de conexão. O sistema foi testado e os resultados foram aprovados, colocando o robô no London Guy's Hospital, em Londres, e controlando-o a partir da Universidade de Johns Hopkings, em Baltimore. Esse trabalho tem uma grande relação ao projeto proposto nesta monografia já que também busca a realização de cirurgias a grandes distâncias.

No projeto Robot Manipulator Control Under UNIX RCCL: A Robot Control “C” Library é apresentado um sistema de controle genérico para robôs, escrito na linguagem C e que roda em sistemas UNIX. Este trabalho poderia ser complementar ao projeto desta monografia, uma vez que a intenção é criar um software genérico para controle de robôs.

CAPÍTULO 4 – ESPECIFICAÇÃO DO PROJETO

O projeto desenvolvido é constituido de diversas partes incluindo *hardware* e *software*. Dois computadores são conectados pela Internet por meio de uma rede VPN. Ao primeiro computador é conectado um *joystick* do videogame Xbox 360 o qual será utilizado pelo usuário (médico). O segundo computador, que pode estar a milhares de quilômetros de distância, se encontra conectado a uma interface computador-robô, desenvolvida para interpretar os dados vindos deste PC e mandar as informações para o braço robô para que este efetue os movimentos. Na Figura 4 pode-se observar uma representação gráfica do projeto.

O funcionamento básico do sistema começa com o médico utilizando o *joystick* para indicar quais são os movimentos a serem realizados pelo robô. Estes dados são capturados no PC do médico (onde o *joystick* está conectado) e são enviados utilizando a rede privada para o outro PC. Uma vez recebidos os dados, o *software* identifica qual é a ação desejada pelo médico e transmite essa informaçao à interface. A interface é encarregada de traduzir a informação para uma linguagem que o robô entenda. Uma vez traduzido, os comandos adequados são enviados para o robô poder executar o movimento requerido.

O médico será capaz de observar todos os movimentos do robô por meio de uma webcam instalada no computador do robô que transmitirá em tempo real as imagens do robô usando VNC.

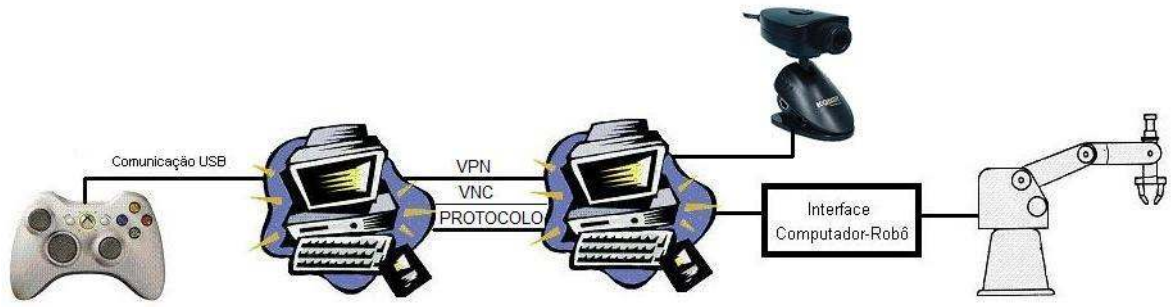

Figura 4. Representação gráfica do projeto

4.1- Especificação do *Software*

O *software* do projeto é composto por dois programas. Um deles é executado no computador do médico. O outro no PC onde o robô se encontra. Conectados por meio de uma rede privada, eles trocam informações entre si para realizar a operação à distância.

4.1.1- *Software* cliente

O *software* cliente é executado no computador onde o médico se encontra e é responsável pela aquisição dos comandos do *joystick e* envio destes para o computador remoto, onde o robô está localizado e o *software* servidor está rodando.

Na Figura 5 é apresentado o fluxograma deste programa. Após ter a conexão corretamente configurada, o programa fica lendo os dados do controle. Quando um novo comando é detectado faz a verificação. Se este for um comando de movimentação ou mudança de motor ativo, ele envia a informação para o *software* servidor. Caso seja requisitado para sair do programa, o *software* é desconectado e acaba sua execução.

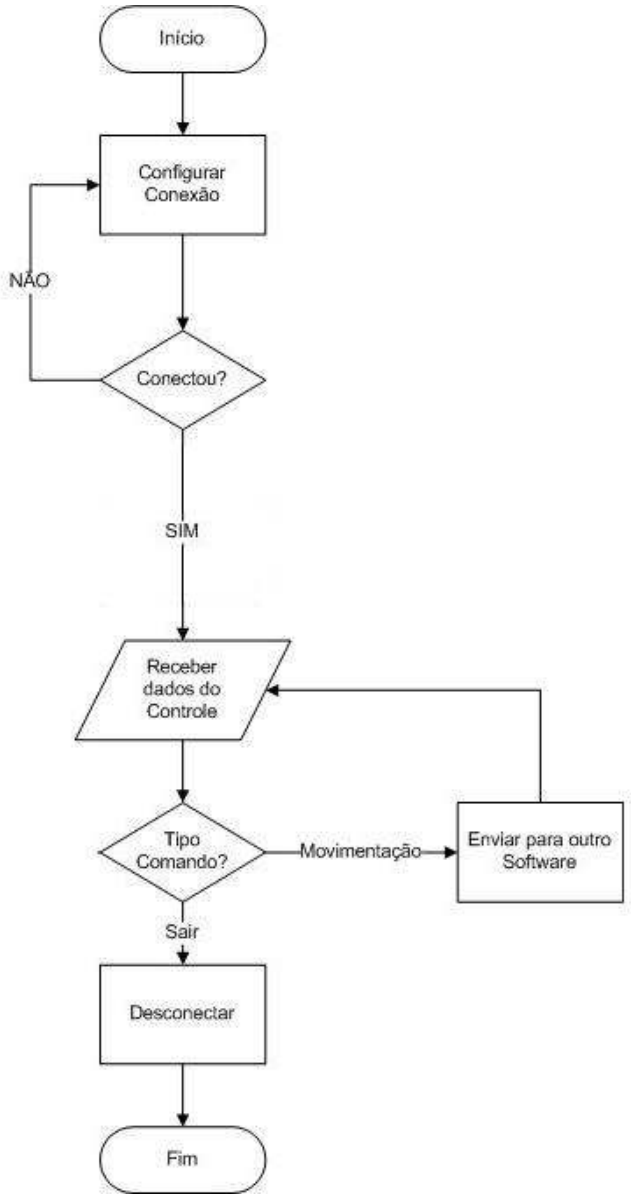

Figura 5. Fluxograma principal do *software* cliente

Para a aquisição dos comandos do *joystick* foi utilizado o Microsoft DirectX, que é uma coleção de APIs (interface de programação de aplicativos) usadas na programação de jogos e outras aplicações multimídia de alto desempenho para o sistema operacional Microsoft Windows. Uma API é um conjunto de rotinas e padrões definidos por um *software* para ser usada em um programa. Este não precisa se ocupar de pequenos detalhes da implementação já que apenas usará as funcionalidades da API. O DirectX,, lançado em 1995 pela Microsoft, possui diversos componentes que abrangem as tarefas que o programador pode precisar. Uma delas é a necessidade de manipular dispositivos de controle tais como, teclados, mouses e *joysticks*. O componente DirectInput é encarregado de lidar com isso. O XInput, que faz parte do DirectInput, é uma API que permite as aplicações Windows receber comandos vindos do controle do Xbox360. É suportado pelo Windows XP (Service Pack 1) ou superior e permite que sejam conectados até quatro controles no PC. Utilizando funções definidas é possível obter uma entrada no programa vinda do controle do Xbox 360. No projeto desenvolvido é utilizado o XInput e suas funções para indicar ao robô quais movimentos realizar de acordo com o desejo do médico.

4.1.2- *Software* servidor

O *software* servidor não terá interação diretamente com o usuário depois que a conexão for estabelecida. A princípio basta informar o IP do cliente que irá conectar ao programa. Ele terá como função receber os comandos do *software* do médico e enviá-los à interface própria do braço.

O seu fluxograma pode ser observado na Figura 6. Uma vez que a conexão seja feita, o *software* deve ler os comandos recebidos pela VPN e interpretar qual atividade está sendo requerida pelo médico. Após identificar a ação desejada, ele envia as informações para a interface computador-robô. Caso receba uma solicitação para sair, o programa é encerrado.

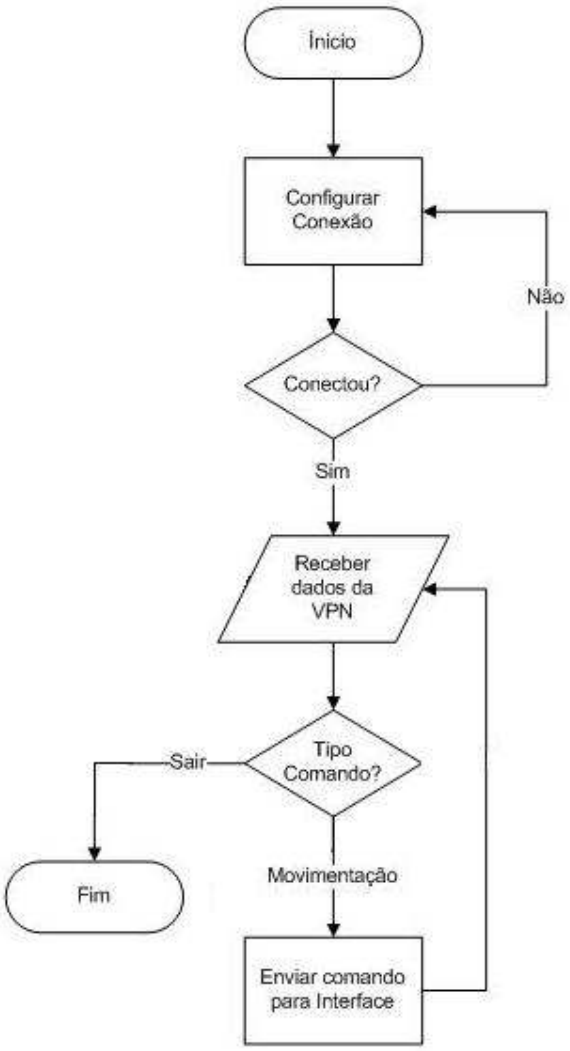

Figura 6. Fluxograma principal do *software* do braço robô

4.2- Especificação do *Hardware*

O projeto é composto pelos seguintes blocos de *hardware*: *joystick*, dois computadores interligados via VPN, uma câmera web, interface computador-robô e braço robô.

4.2.1- Bloco do *Joystick* Xbox 360 Controller

Joystick é um periférico utilizado freqüentemente em jogos eletrônicos, para o controle de máquinas como guindastes, colheitadeiras, elevadores dentre outros. Atualmente, existem diversos modelos, cada um com suas características e vantagens para fins específicos.

O *Joystick* do videogame Xbox 360, mostrado na Figura 7 possui conexão USB (*Universal Serial Bus*) e é composto por dois direcionais analógicos, um direcional digital, oito botões digitais e duas alavancas analógicas. Todas essas características permitem uma grande precisão nos comandos em diversos eixos e uma grande variedade de ações pré-definidas. Outra característica importante do *joystick* é a fácil configuração com a plataforma Windows, sendo detectado automaticamente por este.

O botão A (Verde) seleciona o motor conectado na saída número 1, o botão B (Vermelho) seleciona o motor 2, o botão X (Azul) seleciona o motor 3, o botão Y (Amarelo) seleciona o motor 4, o botão LB (parte de cima do controle, lado esquerdo) seleciona o motor 5 e o botão RB (parte de cima do controle, lado direito) seleciona o motor 6. O direcional analógico esquerdo, visto na figura como o botão na extrema esquerda, é utilizado para movimentar o motor selecionado para esquerda ou direita.

Figura 7. *Xbox360 Controller*

Utilizando o *joystick*, o médico realiza os comandos os quais o robô deve realizar. O Xbox 360 Controller é conectado ao computador usando a porta USB e um *software* adquire os comandos para serem tratados.

A deliberação do não desenvolvimento de um *joystick* próprio durante a realização deste projeto foi considerada a melhor opção devido às qualidades proporcionadas pelo *joystick Xbox 360 Controller*.

4.2.2- Bloco da Interface

A interface de *hardware* desenvolvida neste projeto conecta-se com o computador pela porta serial. Ela recebe os dados vindos deste utilizando um protocolo de comunicação PC-

interface, interpreta a informação recebida e envia o sinal correspondente ao robô para que este execute a informação.

A interpretação dos dados recebidos é feita pelo *firmware* contido no microcontrolador PIC18F452 da Microchip onde será identificado qual motor do braço precisa ser ativado e em que direção este precisa se movimentar.

4.2.3- Bloco do braço robô

O robô utilizado é o ROBIX da Advanced Design Inc, disponível para a compra no site www.robix.com (ROBIX, 2008). Este robô é montado utilizando um kit constituído de seis servo-motores e diversas juntas de tamanhos variados, além de cabos, manuais, *softwares* e ferramentas necessárias para a montagem do robô, tornando possível diversas configurações de montagem.

O ROBIX já vem acompanhado de uma interface para controle dos servos. Esta interface, chamada de RCS-6, pode controlar os servo-motores e possui duas saídas que fornecem 150mA para controle de outros eventuais componentes que o projeto possa necessitar, como LEDs, sensores, etc. No projeto aqui descrito não foi usada esta interface e foi desenvolvida uma interface própria, como descrito anteriormente, para fazer a comunicação entre o computador e o robô. A Figura 8 representa o kit inteiro que acompanha o ROBIX. Como pode ser visto na Figura 8, no lado direito, o kit é composto por 6 servo-motores, além de várias juntas metálicas para apoiar os motores além de algumas ferramentas para montar o robô.

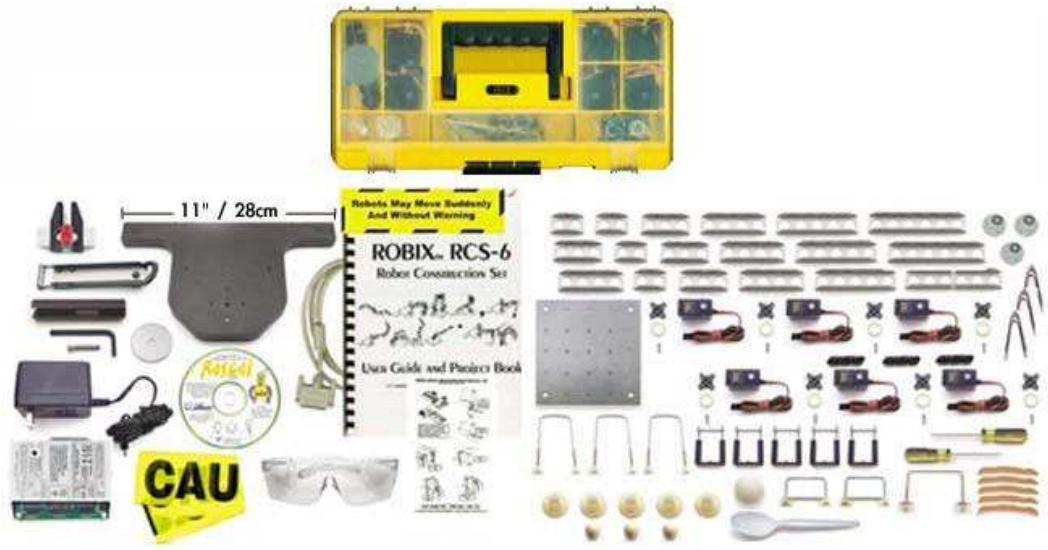

Figura 8. Kit Robix
FONTE: ROBIX (2008)

4.2.4- Bloco da câmera web

Uma câmera web é instalada no computador servidor. Ela é colocada de forma a captar as imagens do braço robô para que o médico seja capaz de observar todos os movimentos do robô através do VNC que é implementado entre os computadores.

No projeto de cirurgia a distância, as imagens da câmera web são exibidas no computador servidor. Utilizando o VNC, o médico situado no computador cliente se conectará remotamente ao PC do robô e terá a possibilidade de visualizar as mesmas imagens e observar os movimentos do robô em tempo real.

CAPÍTULO 5 – DESENVOLVIMENTO E IMPLEMENTAÇÃO

O desenvolvimento do projeto envolve diversos aspectos, envolvendo tanto *software* como *hardware*. A seguir, os passos seguidos durante esse desenvolvimento são explicados mais detalhadamente.

5.1- Aquisição dos comandos do *Joystick*

Ambos os *software*, tanto o cliente como o servidor, foram desenvolvidos na linguagem C++ utilizando a plataforma Borland C++ Builder 6.0. A escolha desta plataforma se deve ao fato de possuir diversos componentes que facilitam a programação de um ambiente visual e com conexão cliente/servidor.

Os valores para cada botão, alavanca e *triggers* do *joystick* são representados com números inteiros. No caso dos direcionais analógicos, os valores são representados pelos eixos X e Y e variam de -32768 até 32767 cada eixo. O valor de X do direcional analógico representa a posição no eixo horizontal enquanto o valor de Y diz respeito ao eixo vertical. Os *triggers*, direito e esquerdo têm seus valores variando entre 0 e 255. Por sua vez, os demais botões, tanto o direcional digital como os oito botões digitais, têm seus valores representados em apenas uma variável. Por este motivo, é necessário fazer uma máscara de bits para identificar quais botões estão sendo pressionados. Os valores para cada botão estão representados na Figura 9. Fazendo a operação lógica E (AND) é possível saber se determinado botão está ou não apertado.

```
#define XINPUT_GAMEPAD_DPAD_UP          0x00000001
#define XINPUT_GAMEPAD_DPAD_DOWN        0x00000002
#define XINPUT_GAMEPAD_DPAD_LEFT        0x00000004
#define XINPUT_GAMEPAD_DPAD_RIGHT       0x00000008
#define XINPUT_GAMEPAD_START            0x00000010
#define XINPUT_GAMEPAD_BACK             0x00000020
#define XINPUT_GAMEPAD_LEFT_THUMB       0x00000040
#define XINPUT_GAMEPAD_RIGHT_THUMB      0x00000080
#define XINPUT_GAMEPAD_LEFT_SHOULDER    0x0100
#define XINPUT_GAMEPAD_RIGHT_SHOULDER   0x0200
#define XINPUT_GAMEPAD_A                0x1000
#define XINPUT_GAMEPAD_B                0x2000
#define XINPUT_GAMEPAD_X                0x4000
#define XINPUT_GAMEPAD_Y                0x8000
```

Figura 9. Máscara de bits dos botões digitais do *Xbox360 Controller*
FONTE: XINPUT (2008)

Para que esta captura seja feita, é utilizada a função *XinputGetState*, presente no Xinput que é um componente do DirectX SDK. Esta função tem o seguinte protótipo:

```
DWORD XInputGetState(
  DWORD dwUserIndex,
  XINPUT_STATE* pState
);
```

O dwUserIndex é o número do controle que a leitura deve ser feita, indo de 0 a 3 (controle 1 ao 4) e pState é o ponteiro que receberá o estado atual de seus botões. Este ponteiro aponta para uma struct a qual possui variáveis referentes a cada item do controle (botões, direcionais, alavancas). O seu protótipo pode ser visto abaixo:

```
typedef struct _XINPUT_GAMEPAD
{
   WORD                wButtons;
   BYTE                bLeftTrigger;
   BYTE                bRightTrigger;
   SHORT               sThumbLX;
   SHORT               sThumbLY;
   SHORT               sThumbRX;
   SHORT               sThumbRY;
} XINPUT_GAMEPAD, *PXINPUT_GAMEPAD;
```

No projeto aqui descrito são usados os botões A, B, X, Y, LB e RB do *joystick* para representar os seis diferentes motores que constituem o robô. Quando o médico aperta um desses botões, é feita a captura dessa informação para setar no *firmware* do microcontrolador qual motor deve ser ativado, depdendendo do botão apertado. O motor ativo pode se movimentar tanto para a esquerda quanto para a direita. Para controlar a direção do movimento foi usado o direcional analógico esquerdo do Xbox 360 Controller. Dependendo o valor X (horizontal) deste direcional é identificado se o médico quer mexer o servo-motor para a esquerda ou direita.

O robô foi montado usando uma configuração que simula um braço mecânico. Os servo-motores foram conectados entre eles, em sua maioria, de forma horizontal, mas com alguns de forma vertical com o propósito de ter mais articulações possiveis no robô. Por esse motivo, para alguns motores a direção do movimento, esquerda ou direita, pode ser entendido como “acima” e “abaixo” para visualização do operador do braço.

5.2– Implementação da VPN

Para configurar a rede VPN entre os dois computadores é utilizado o *software* OpenVPN (OpenVPN, 2008).

OpenVPN é uma solução open source completa que possui uma grande variedade de configurações, incluindo acesso remoto, criação de uma rede privada entre dois computadores, segurança Wi-Fi, etc.

Para criação da rede VPN utilizando o OpenVPN, é necessário instalar o *software* nos dois computadores, criar uma chave de segurança (podendo ser gerada pelo programa), além de realizar outras configurações e colocá-las em um arquivo de configuração com a extensão .ovpn em um diretório específico.

Uma vez criada a rede privada, um novo IP é definido para cada um dos computadores como pode ser visto na Figura 10, sendo que estes novos IPs devem ser usados para que a troca de dados entre os dois PCs seja feita utilizando o tunelamento da VPN.

Na Figura 11 pode-se observar as configurações que são feitas pelo OpenVPN enquanto está se criando a rede privada virtual.

Figura 10. Definição de IP pelo OpenVPN

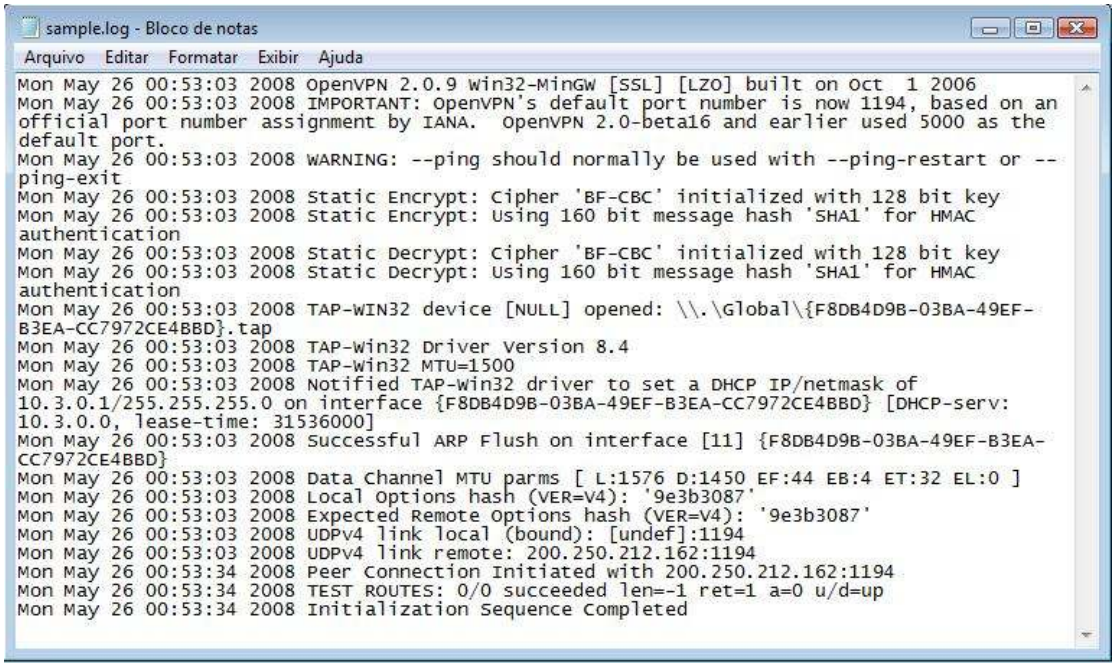

Figura 11. Configurações feitas pelo OpenVPN no processo de tunelamento

5.3– Enviando os Dados do Controle via VPN

O *software* cliente, desenvolvido na linguagem C++, realiza o envio dos dados capturados pelo *joystick*. Este programa é executado no computador cliente e requer a entrada do IP para conectar-se com o outro PC. Neste caso deve-se utilizar o IP designado pelo OpenVPN. Com isto, todos os dados que transitam pela rede utilizam os protocolos de criptografia referentes a VPN.

O programa faz a verificação para saber qual botão está sendo pressionado, assim é possível descobrir a ação desejada do médico. Logo após, esta informação é enviada para o computador servidor. A seguir é demonstrado um trecho do código que faz esta verificação e envia a mensagem:

```
msg_enviada = "";
if(state.Gamepad.wButtons == 4096)          msg_enviada = "ativaA";
else if(state.Gamepad.wButtons == 8192)     msg_enviada = "ativaB";
else if(state.Gamepad.wButtons == 16384)    msg_enviada = "ativaC";
else if(state.Gamepad.wButtons == 32768)    msg_enviada = "ativaD";
else if(state.Gamepad.wButtons == 256)      msg_enviada = "ativaE";
else if(state.Gamepad.wButtons == 512)      msg_enviada = "ativaF";
```

```
if(msg_enviada != "")
        ClientSocket1->Socket->SendText(msg_enviada);

msg_enviada = "";
if(state.Gamepad.sThumbLX > 10000)       msg_enviada = "mexeDireita";
else if(state.Gamepad.sThumbLX < -10000)          msg_enviada = "mexeEsquerda";

if(msg_enviada != "")
        ClientSocket1->Socket->SendText(msg_enviada);
```

A variável “msg_enviada” do tipo String é a variável que será mandada para o servidor e indica a atividade que este deve passar à interface do robô. A variável wButtons indica qual botão está sendo pressionado dependendo seu valor. Quando o valor do sThumbLX é verificado para detectar desejo de movimento a esquerda ou direita, a ação é detectada somente se o valor desta variável for maior a 10000 (para direita) ou menor a -10000 (para esquerda). Isto se deve ao fato de que quando os direcionais analógicos não são pressionados, a variável sThumbLX não tem seu valor exatamente igual a zero. Para que seja detectada a ação de movimentar a esquerda ou direita, o médico deve movimentar a avalanca de forma que seu valor não esteja entre os limites 10000 e -10000.

5.4– Comunicando o Computador com a Interface

O programa que executa no computador servidor, recebe uma variável do tipo *String* chamada QUADRO que indica a ação que o robô devera realizar. Este *software* lê a variável recebida e conforme o valor envia à interface um dado do tipo char. Este dado pode ter valores de ‘a’ até ‘h’. A comunicação entre o PC e a interface é feita usando a porta serial do computador. A seguir pode-se observar um trecho do código deste programa onde a variável recebida é lida e, de acordo seu valor, é enviado à interface o caractere correspondente.

```
QUADRO = Socket->ReceiveText();

if(QUADRO == "ativaA")
        porta->EnviaDado("a",1);
if(QUADRO == "ativaB")
        porta->EnviaDado("b",1);
if(QUADRO == "ativaC")
        porta->EnviaDado("c",1);
if(QUADRO == "ativaD")
        porta->EnviaDado("d",1);
if(QUADRO == "ativaE")
        porta->EnviaDado("e",1);
if(QUADRO == "ativaF")
        porta->EnviaDado("f",1);
if(QUADRO == "mexeEsquerda")
        porta->EnviaDado("g",1);
if(QUADRO == "mexeDireita")
        porta->EnviaDado("h",1);
```

5.5 – Desenvolvimento da Interface

A interface desenvolvida para comunicar o computador com o robô, ilustrada na Figura 12, tem como principal componente o microcontrolador PIC 18F452. Nele está contido o *firmware* o qual interpreta o sinal de entrada vindo do PC e envia um sinal de saída aos servo motores.

A alimentação da interface é feita usando uma fonte de 5V. Outro componente importante é o MAX-232, o qual é responsável por fazer a comunicação serial e permitir ao microcontrolador receber os dados do PC por meio de um cabo serial usando o conector DB-09.

Quem da partida ao microcontrolador é o relógio (clock) e é obtido a partir do componente eletrônico chamado oscilador. Nos pinos 13 (OSC1) e 14 (OSC2) do PIC 18F452 é conectado um oscilador de 20MHz para ter uma sincronia de todos os eventos do circuito digital.

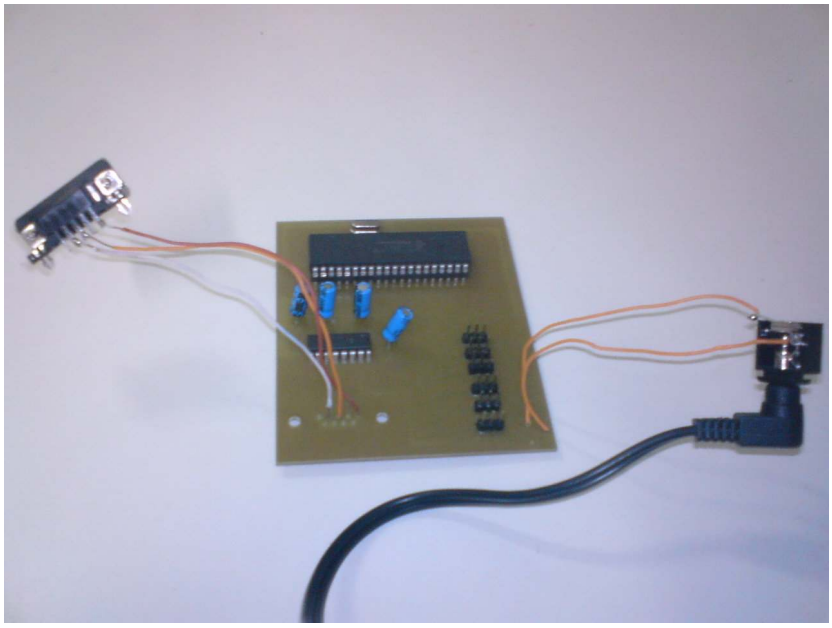

Figura 12. Interface computador-robô

.

A Figura 13 apresenta uma ilustração do diagrama esquemático da interface. Os seis conectores, de três pinos cada um, (representados no esquemático com a sigla X2 até X7) são usados para conectar os servos ao circuito, sendo que o primeiro pino é o sinal de controle do servo, conectado a um dos pinos de saída do microcontrolador. Os outros dois pinos são conectados no 5V e no terra.

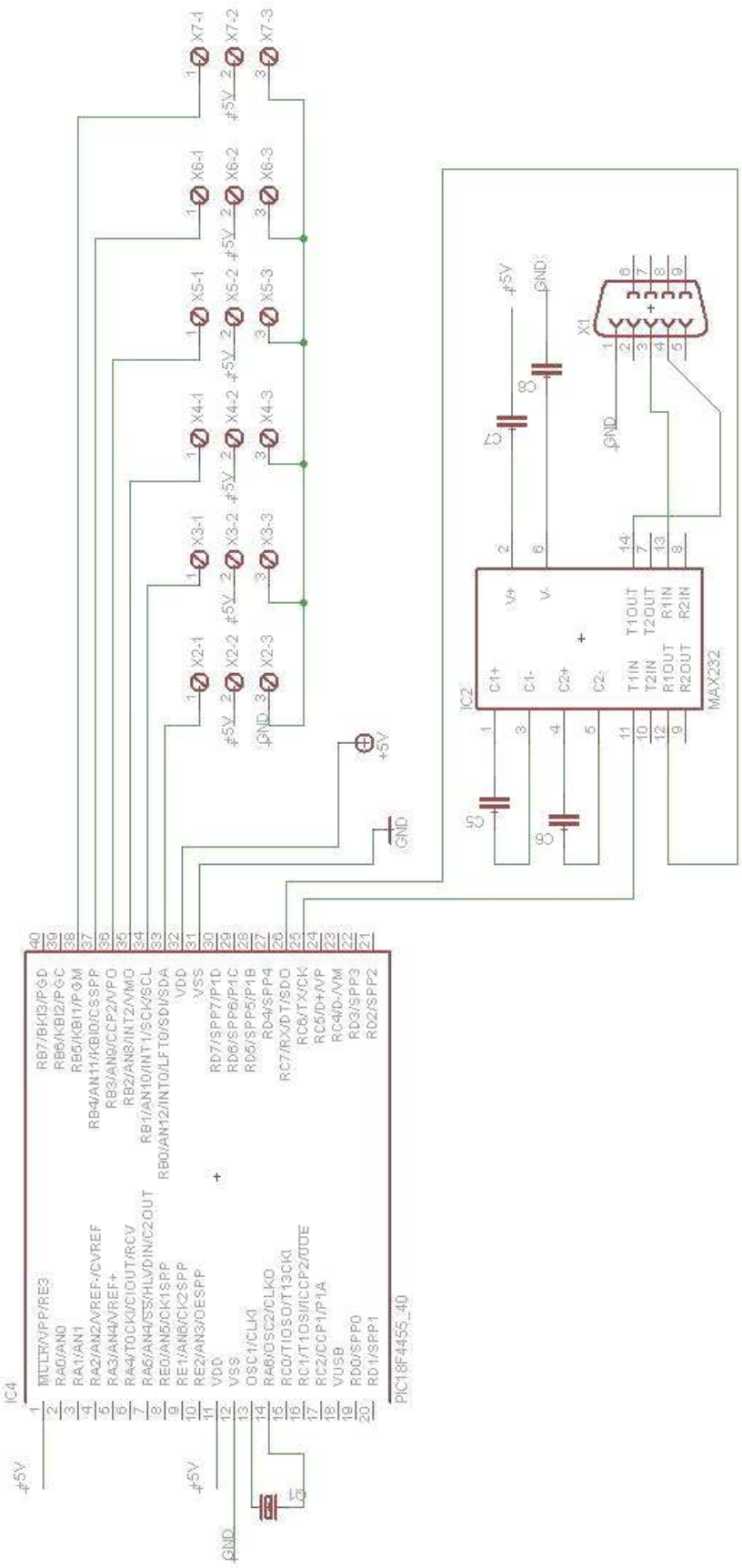

Figura 13. Diagrama esquemático da interface.

O *firmware* contido no microcontrolador foi desenvolvido na linguagem C usando o programa PCW para compilar o código e gerar o arquivo com extensão '.hex'. Este arquivo é gravado diretamente no microcontrolador através do programa IC-Prog.

O programa fica lendo continuamente a entrada do PIC e quando este recebe um novo dado procedente do computador é feita a interpretação do mesmo. O dado recebido pode indicar duas possíveis ações: mexer o motor (para a esquerda ou direita) ou mudar o motor ativo. Este dado é do tipo char e tem como possíveis valores as letras de 'a' ate 'h' sendo que um valor 'a' representa a ativação do motor 1, o valor 'b' representa a ativação do motor 2 e assim por diante. As letras 'g' e 'h' representam a necessidade de movimentar o motor à esquerda e direita respectivamente.

O programa contém uma variável inteira chamada 'motorAtivo' a qual diz respeito ao motor que o médico quer movimentar naquele momento, podendo ter valor variado de 1 a 6, representando cada um dos motores.

As portas de saída RB do microcontrolador são conectadas aos sinais de controle dos seis servo-motores. Quando o programa receber um dado indicando a mudança do motor ativo, a variável 'motorAtivo' é atualizada. Se o dado recebido do computador indicar movimento a esquerda ou direita, o sinal de saída (5V) é enviado à porta de saída indicada pela variável 'motorAtivo'.

Os servo-motores podem se mexer de 0 a 180 graus dependendo a duração do sinal enviado a eles. Cada vez que for pedido um movimento a esquerda ou direita, o servo-motor se mexe 18 graus em relação à posição anterior.

O *firmware* é responsável por controlar se o motor já se encontra em algum dos seus dois limites. Por exemplo, se um servo-motor estiver na posição 180 graus e o médico enviar um comando para se mexer à direita o sinal de saída não será enviado ao servo já que será impossível realizar o movimento em questão.

5.6– Implementando o VNC entre os Computadores

Existem diversos programas que oferecem VNC. As duas versões mais utilizadas são o TightVNC e o Real VNC. Pelo fato de possuir mais recursos, como por exemplo, funções de chat e transferência de arquivos, foi escolhido o programa Real VNC (RealVNC, 2008).

Após a instalação do Real VNC, nos dois computadores, o módulo servidor é executado no PC do robô. Varias configurações podem ser feitas tais como especificar em qual porta os clientes irão se conectar assim como a possibilidade de definir um tempo máximo de conexão,

desconectando automaticamente os clientes após este período. Pode-se configurar também se é permitido que os clientes possam modificar a área de trabalho do servidor ou somente usá-la no estilo leitura. Neste projeto está opção de modificação da área de trabalho não é escolhida já que o módulo ciente se conectará remotamente ao servidor somente para visualizar as imagens procedentes da câmera *web*.

Para evitar que qualquer cliente possa se conectar no servidor, o Real VNC oferece uma autenticação por senha exigida aos clientes que queiram conectar. No computador cliente é executado o módulo cliente do Real VNC, já que este irá se conectar remotamente ao servidor para visualizar a transmissão das imagens da webcam. Pode-se executar o modo de escuta (listening mode) ou o modo visualizador (viewer mode). Executando o listening mode é permitido que os servidores adicionem o computador como seus cliente. Para isso, o servidor deverá adicionar o cliente informando o IP do computador. Caso seja escolhido o *viewer mode*, pode-se escolher o servidor ao qual é desejado conectar digitando o IP do mesmo e a senha de autenticação se for requerida pelo servidor.

No PC do médico, é executado o modo viewer do cliente e informado qual é o IP do servidor em que o robô se encontra localizado. É importante notar que nesta fase usa-se o IP designado pelo VPN, já configurado, para o servidor. Na Figura 14 é vista uma ilustração disso.

Figura 14. Conectando remotamente ao servidor usando VNC

CAPÍTULO 6 – VALIDAÇÃO E RESULTADOS

O resultado deste trabalho é um sistema capaz de controlar um robô a distância. No decorrer do projeto foram realizados diversos testes para se conseguir resultados satifastórios no que diz respeito à conexão, construção do robô, captura de comandos de um *joystick*, transmissão de dados via *sockets* e envio de um sinal ao servo motores por meio de um microcontorlador.

Foi observado que o timer do programa que faz a leitura do *joystick* deve ser de no mínimo 300ms sendo este o tempo suficiente para executar o trecho do código referente à analise dos dados e o envio pela VPN. Foi testado um timer de 100ms o qual não teve o sucesso esperado fazendo o programa se perder no caminho e não poder responder a todas as requisições feitas pelo médico através do *joystick*. Isso ocorre devido à velocidade do PIC18F452 que não se mostrou suficiente para interpretar os comandos e enviar a instrução para o servo motor. Em uma cirurgia real este tempo é inaceitável e este fato deve ser corrigido com a troca do microcontrolador por algum modelo mais eficiente.

A fonte utilizada para alimentar a interface computador-robô oferece como saída 5V e 2,5A de corrente. Este valor foi suficiente considerando que cada servo motor somente requer 150mA.

Os servo-motores HS-422 utilizados para a construção do robô supriram a necessidade do projeto que é mostrar movimentos no robô. Porém se deseja-se ter maior precisão nos movimentos deve-se optar por outros motores que possibilitem um movimento mais preciso dos mesmos.

O braço robótico foi construído de forma a poder ter a maior quantidade de articulações possíveis no nosso robô. Diversas configurações são possíveis de obter, mas foi decidido por uma que simule um braço mecânico. Um dos motores foi utilizado para movimentar uma garra podendo abrir e fechar a mesma. Na Figura 15, tem-se uma imagem do robô construído.

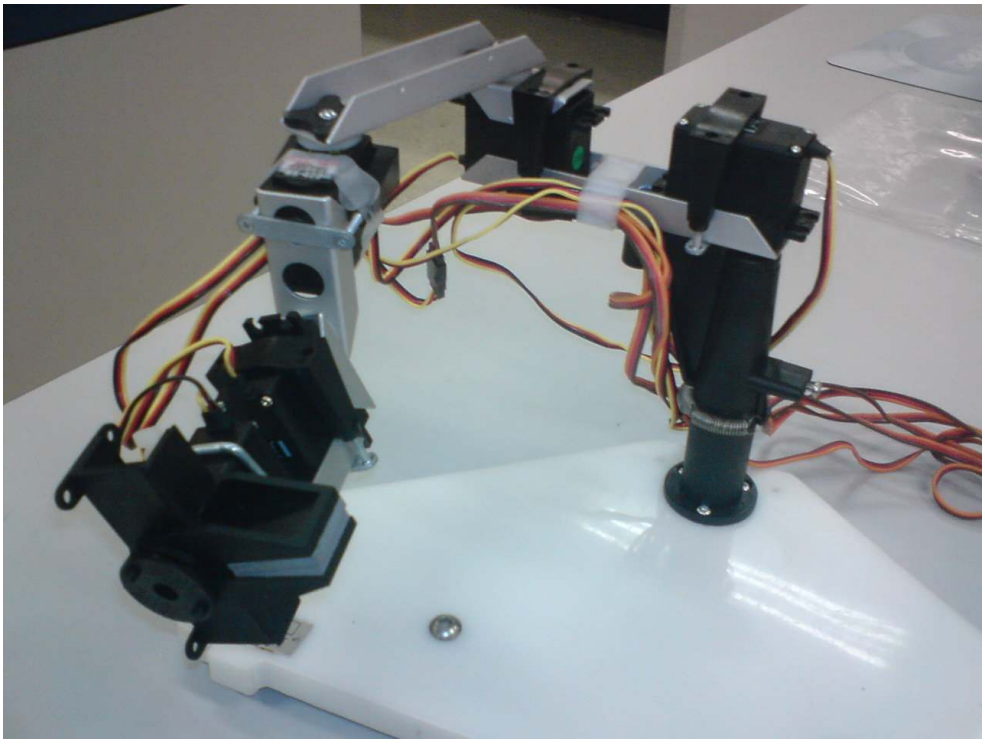

Figura 15. Braço robô.

Na Figura 16 pode-se observar o *software* do médico fazendo uma requisição para ativar o motor B do robô. O programa lê a variável referente aos botões e detecta que o valor 8192 corresponde ao botão B sendo pressionado pelo usuário.

Figura 16. Médico pressionando o botão B.

Após identificar que o botão B está sendo pressionado, a mensagem é enviada para o *software* do robô indicando que precisa ativar o motor B, como mostrado na Figura 17.

Figura 17. Programa recebendo mensagem para ativar o motor B.

A continuação o médico deseja mexer o motor ativo (neste caso o motor B), por isso movimenta o direcional analógico para a direita. O programa detecta que o valor da variável deste direcional é maior a 10000, entendendo que uma requisição para mexer à direita deve ser enviada. Na Figura 18, vê-se que o valor do LEFT-X (que representa a alavanca da esquerda) tem valor igual a 32767. Isto significa que o médico está movendo ela o máximo possível para a direita, já que o máximo valor que ela pode ter é 32767.

Figura 18. Médico mexendo a alavanca para a direita

Uma vez que este movimento é detectado pelo timer, a nova mensagem é enviada para o programa do robô. Na Figura 19 observa-se este processo. É importante salientar que neste caso o médico manteve a alavanca à direita por mais de 300ms, por isso foi enviada mais de uma mensagem do tipo "mexeDireita". Neste caso o robô continua se movimentando à direita até que o médico solte o direcional analógico ou o servo motor chegue no seu limite máximo de posição possível.

Figura 19. Programa recebendo mensagem para mexer o motor à esquerda.

CAPÍTULO 7 – CONCLUSÃO

Este trabalho procurou simular a possibilidade de ter um ambiente propicio à realização de cirurgias à distância. O objetivo principal do projeto que era o controle de um braço robô remotamente desde outro PC foi atingido tendo resultados satisfatórios.

O microcontrolador utilizado (PIC18F452) mostrou ser apto para suprir as necessidades do projeto tendo importância na comunicação entre o computador com o robô fazendo parte da interface desenvolvida.

Este projeto nos permite a realização de melhorias nele como, por exemplo, obter movimentos mais precisos nos motores. Outro item passível de melhoras é a interface computador-robô. Um projeto que desenvolva uma interface de rede que possa se conectar à Internet possibilitaria a eliminação do segundo PC. Outra melhora na interface é a troca do microcontrolador por algum mais eficiente, visando diminuir o *delay* de 300ms, tempo inaceitável para uma cirurgia real. O computador do médico poderia ser conectado via VPN diretamente com a interface de rede desenvolvida não sendo mais necessária a presença de um computador no local do robô.

Outra implementação possível em cima do projeto é realizar a captura das imagens da webcam no próprio *software* do PC do robô e enviar elas diretamente para o *software* do médico. Desta forma não seria necessário utilizar o VNC para visualizar a área de trabalho do PC do robô.

Para poder chegar a ter um sistema capaz de fazer a cirurgia à distância, é evidente que vários fatores devem ser alcançados. Um robô com motores de alta precisão e um processador de alta velocidade seriam indispensáveis para o procedimento.

A idéia original do projeto diz respeito à realização de cirurgias à distância, porém a possibilidade de controlar um robô remotamente de qualquer parte do mundo, nos traz muitas outras aplicações como, por exemplo, poder desativar uma bomba estando longe dela utilizando de um robô.

CAPÍTULO 8 - REFERÊNCIAS BIBLIOGRÁFICAS

ASIMOV, I.; **Eu robô.** Editora Ediouro. São Paulo. 2004

Cirurgia disponível em <http://pt.wikipedia.org/wiki/Cirurgia>. Acessado em Março de 2008.

Classificação Geral dos Robôs em <http://www.din.uem.br/ia/robotica/classif.htm>. Acessado em Outubro de 2008.

DirectX disponível em <http://pt.wikipedia.org/wiki/DirectX>. Acessado em Maio de 2008.

GUIMARÃES, A. G.; et al – **Segurança em redes privadas virtuais – VPNs.** Editora Brasport. São Paulo, 2006

HS-422 disponível em <http://www.hitecrcd.com/servos/show?name=HS-422>. Acessado em Outubro de 2008

LUI, J. H.; et al - **The Increasing Workload of General Surgery** – Disponível em <http://cat.inist.fr/?aModele=afficheN&cpsidt=15684879>. Acessado em Março de 2008.

LIOU, K. C.; - Rede Privada Virtual – VPN – Disponível em <http://www.rnp.br/newsgen/9811/vpn.html>. Acessado em Março de 2008.

MARQUES, A. L.; et al - **Web Controlled Robotic Arm** – Disponível em <http://www.aedie.org/9CHLIE-paper-send/381-MARQUES.pdf>. Acessado em Março de 2008.

McCOMB, G.; PREDKO, KYKE – **Robot Builder's Bonanza.** Mc Graw Hill. 2006

MICROCHIP Technolog Incorporated. PIC18F452 Datasheet disponível em < http://ww1.microchip.com/downloads/en/devicedoc/39564c.pdf> Acessado em Outubro de 2008.

OPEN VPN disponivel em <http://www.openvpn.net>. Accessado em Maio de 2008.

PATRICIU, A.; et al - Robotic Telementoring/Telesurgical System and Randomized Evaluation Study – **Engieneering in Medicine and Biology Society**. Shanghai. pg 2167-2170. 2005.

PEREIRA, F; **Microcontroladores PIC Programação Em C**. Editora Érica. São Paulo. 2003

RealVNC disponivel em <http://www.realvnc.com>. Acessado em Outubro de 2008.

RealVNC disponível em <www.realvnc.com>. Acessado em Março de 2008.

Rede Privada Virtual – VPN disponível em <http://www.rnp.br/newsgen/9811/vpn.html>. Acessado em Março de 2008.

ROBIX disponível em <http://www.robix.com/contents.html>. Acessado em Maio de 2008

SANTOS, A. - 2007 - **Servomotores**. Disponível em <http://www.sumoderobos.org/artigos/servomotores.pdf>. Acessado em Outubro de 2008.

SEVERINO, E. C. ***Joystick* para controle do braço mecânico ED-7220C**. Curitiba, 2005. Monografia (Graduação). Unicenp

SILVA, L. S. Da – **Virtual Private Network (VPN).** Editora Novatec. São Paulo, 2002

SILVA, R. A. – **Programando microcontroladores PIC – Linguagem C.** Ensino Profissional. 2006

SOUZA, V. A. – **Projetando com os microcontroladores da família PIC18: Uma nova percepção.** Ensino Profissional. 2007.

SOUZA, D. J; **Desbravando o PIC.** Editora Érica. São Paulo. 2001

VNC tutoriais disponível em <http://www.guiadohardware.net/tutoriais/vnc/>. Acessado em Outubro de 2008.

VPN disponível em <http://pt.wikipedia.org/wiki/VPN>. Acessado em Março de 2008.