UMA ARQUITETURA PARA TELEOPERAÇÃO INTEGRANDO INTERFACE NATURAL, REALIMENTAÇÃO DE FORÇA E SERVOVISÃO

Gabriel C. da Motta-Ribeiro*, Antonio C. Leite*, Pål J. From†, Fernando Lizarralde*, Liu Hsu*

*Programa de Engenharia Elétrica/COPPE
Universidade Federal do Rio de Janeiro - UFRJ
Rio de Janeiro, RJ, Brasil

†Department of Mathematical Sciences and Technology
Norwegian University of Life Sciences
Ås, Norway

Emails: gabrielcasulari@poli.ufrj.br, toni@coep.ufrj.br, pal.johan.from@umb.no, {fernando, liu}@coep.ufrj.br

Abstract— The teleoperation of robotic devices allows that the operators perform several tasks with safety in remote areas and hazardous environments. However, interfaces and architectures proposed for these systems are limited to operating point-to-point or have complex and application associated programming structures, making difficult extent and rapid prototyping. This paper presents a new teleoperation architecture performed using Matlab and Robot Raconteur. These tools allow one to change input and robotics devices, as well as the control strategies. Currently in the developed application are considered natural interface, haptic and visual servo control for a robotic manipulator. The concept was tested in pick-and-place tasks, remotely executed by using a PUMA 560 manipulator, a CCD camera, a Phantom Omni haptic device and a Kinect.

Keywords— Teleoperation, Force Feedback, Remote Control, Visual Servoing

Resumo— A teleoperação de dispositivos robóticos permite que operadores executem diversas tarefas com segurança, em áreas remotas e ambientes perigosos. Entretanto, as interfaces e arquiteturas propostas para estes sistemas são limitadas a operação ponto a ponto ou possuem estruturas de programação complexas e associadas a aplicação, dificultando extensão e prototipagem rápidas. Neste trabalho é apresentada uma nova arquitetura de teleoperação implementada através de Matlab e Robot Raconteur. Essas ferramentas permitem a alteração de dispositivos de entrada e sistemas robóticos, bem como das estratégias de controle. Atualmente, na aplicação desenvolvida são consideradas interface natural, háptica e servovisão para um manipulador robótico. O conceito foi testado em tarefas de *pick-and-place* remotamente executadas usando um manipulador PUMA 560, uma câmera CCD, um dispositivo háptico Phantom Omni e Kinect.

Palavras-chave— Teleoperação, Realimentação de Força, Controle Remoto, Servovisão

1 Introdução

A utilização de robôs operados remotamente para realizar tarefas em ambientes hostis ou locais de difícil acesso tem despertado o interesse da comunidade científica nos últimos anos (Bellingham and Rajan, 2007; Trevelyan et al., 2008). Neste contexto, a teleoperação uni ou bilateral (Niemeyer et al., 2008) apresenta-se como uma solução viável para a execução de tarefas de inspeção e manutenção em diversas aplicações, por exemplo, nuclear (Iborra et al., 2003), espacial (Preusche et al., 2006) e *offshore* (Skourup and Pretlove, 2009), a fim de reduzir os custos da operação e melhorar as condições de saúde, segurança e meio ambiente. A instabilidade em operações bilaterais, gerada pelo atraso nos meios de comunicação, foi estudada em termos da estrutura de controle (Hokayem and Spong, 2006). Entretanto os trabalhos referentes a estas técnicas, geralmente, não especificam a arquitetura utilizada como plataforma experimental.

No âmbito acadêmico, a possibilidade de utilizar equipamentos robóticos localizados fora do espaço físico de um determinado laboratório ou instituição permite o compartilhamento de recursos de *hardware* entre diferentes centros de pesquisa. Nesta aplicação é importante utilizar uma arquitetura flexível que permita que a estrutura seja reconfigurada ou expandida, alterando-se estratégias de controle e adicionando novos equipamentos de forma simples e rápida.

As arquiteturas e interfaces de teleoperação propostas na literatura são limitadas a operações ponto a ponto (Taylor et al., 1999; Bambang, 2007) ou trajetórias planejadas antes da execução (Safaric et al., 2003) sem uma forma de realimentação visual durante o movimento, operações bilaterais com os sistemas mestre e escravo no mesmo ambiente (Oboe and Fiorini, 1998; Munir and Book, 2003; Goto et al., 2010) ou dependentes de um padrão de comunicação em rede pouco difundido (Anderson, 1995). O OROCOS (Soetens and Bruyninckx, 2005) e ROS (Quigley et al., 2009) estão se difundido como plataformas de desenvolvimento de aplicações robóticas. Entretanto a primeira não se aplica a este trabalho por ser desenvolvida para controle em tempo real e a segunda pode apresentar mal funcionamento devido

a temporização de diferentes fontes de mensagens, o que é crítico a grandes distâncias.

O objetivo deste trabalho é apresentar uma nova arquitetura de controle realizada através de Matlab (The MathWorks, Inc.) e Robot Raconteur (RR) (Wason and Wen, 2011) que integra em uma única interface gráfica (GUI) diferentes modalidades de controle. A realimentação pode ser visual através de uma câmera com *stream online* e de força com dispositivos hápticos, permitindo operações com controle uni ou bilateral. As ferramentas utilizadas permitem a prototipagem rápida com alteração de dispositivos de entrada, dispositivos robóticos e estratégias de controle.

Experimentos realizados com um robô PUMA 560 do Rensselaer Polytechnic Institute (Troy, NY, EUA) comandado do Laboratório de Controle da COPPE/UFRJ, ilustram a viabilidade da estrutura de teleoperação apresentada.

2 Arquitetura Proposta

A arquitetura de teleoperação proposta permite a realização das seguintes ações, individuais ou combinadas:

1. Movimentação mestre/escravo e realimentação visual;
2. Controle de posição via servovisão;
3. Alteração do estado da garra;
4. Realimentação de força,

a fim de garantir a realização bem sucedida de uma tarefa teleoperada (e.g., *pick-and-place*). A primeira operação pode ser realizada com o Phantom Omni (Sensable) ou o Kinect (Microsoft Corp.).

As diferentes interfaces de comando do robô localizadas no ambiente remoto são integradas através do Matlab e as posições no espaço das juntas, ou erro no espaço da imagem, são enviadas através da internet para o ambiente local onde estão as malhas de controle do manipulador robótico. Os fluxos de informação desta arquitetura são apresentados na Figura 1, sendo a infraestrutura local e remota detalhadas nas seções seguintes.

2.1 Comunicação

Como mostrado na Figura 1 existem duas vias de comunicação entre os ambientes remoto e local, uma para operação do manipulador e outra para transmissão de vídeo. A primeira é realizada através de comunicação UDP servidor/cliente implementada em Matlab através do pacote *pnet*. Para estabelecer a conexão o cliente envia um comando para o servidor, conhecendo previamente o seu endereço IP, este então reconhece o endereço do cliente e inicia o envio de comandos de acordo com o modo de operação estabelecido. O controle de acesso ao manipulador é realizado pelo ambiente local, que define a conexão com o servidor UDP remoto (ver Figura 2). Além disso, como o sistema de controle está em uma rede virtual local (ver seção 2.2) ao se requisitar a conexão o roteador é capaz de resolver as questões de roteamento interno de portas. As possíveis barreiras de *firewall* estabelecidas pela rede do ambiente local podem ser ultrapassadas através da utilização de uma VPN (em inglês, *virtual private network*).

A segunda via de comunicação utiliza protocolo TCP para o envio das imagens do ambiente local como realimentação para o operador remoto. Neste caso o servidor é local, pois o recurso acessado pode ser utilizado de forma segura por múltiplos clientes, no ambiente remoto de operação ou não. O gerenciamento de conexões e a transmissão da imagem são realizados através da arquitetura RR (Wason and Wen, 2011).

Robot Raconteur

O RR é uma arquitetura e biblioteca de comunicação para aplicações de robótica e automação. É baseada em nós, tipicamente processos em uma máquina, que se conectam através de canais em um modelo servidor/cliente. A conexão é realizada através de uma requisição do nó cliente que conhece previamente o endereço IP do nó servidor, este então expõem suas funcionalidades por meio de um texto com as definições de suas estruturas e objetos em um formato independente de linguagem. Desta forma o cliente não necessita conhecer previamente as características do servidor, podendo o mesmo ser alterado de forma independente (Wason and Wen, 2011).

Os canais de transporte de mensagem implementados permitem a comunicação através de TCP/IP para uma rede local ou internet, de uma *pipe* para processos internos em uma máquina Windows e da porta serial para processadores embarcados de baixo consumo. O RR considera sistemas em tempo real ou baseados em eventos e permite programação em diferentes linguagens. Atualmente são permitidos nós desenvolvidos em linguagens C#, C++ e Matlab ou que utilizem a plataforma Arduino (Wason and Wen, 2011). A possibilidade de integração com o Matlab e o reconhecimento de funcionalidades no ato da conexão, tornam o RR mais adequado a prototipagem rápida que ferramentas mais gerais como o ROS (Quigley et al., 2009).

2.2 Infraestrutura Local

O ambiente de operação local contém um manipulador robótico com um sensor de força acoplado ao efetuador, uma malha de controle para os atuadores das juntas e uma câmera conforme será descrito a seguir.

ISBN: 978-85-8001-069-5

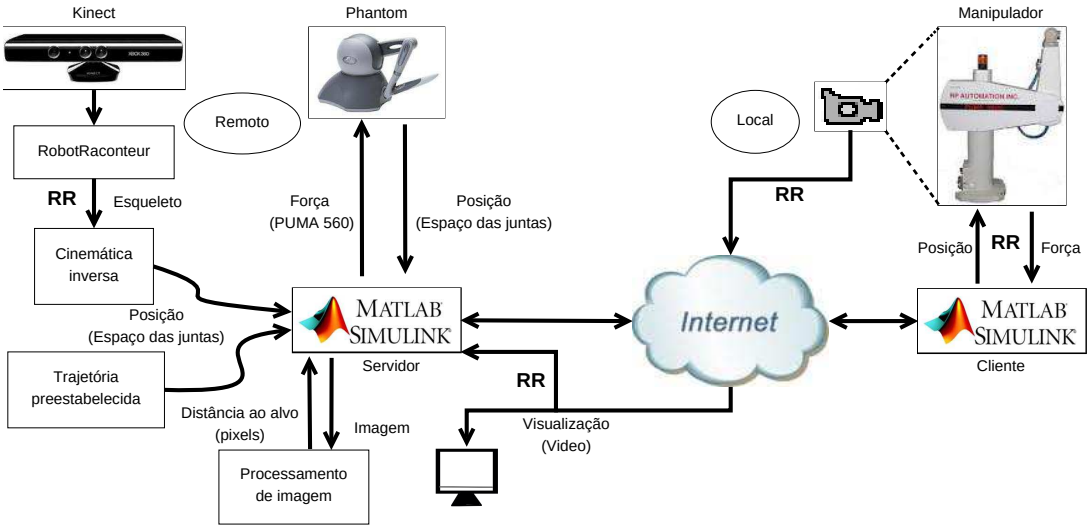

Figura 1: Diagrama de interfaces entre as diferentes modalidades de entrada (ambiente remoto) e o sistema de controle (ambiente local). **RR** indica a comunicação através do Robot Raconteur.

Sistema de controle

A malha de controle do sistema robótico é implementada de forma digital em um computador utilizando um *kernel* em tempo real com código gerado através da ferramenta *XPC Target* para Matlab/Simulink. Os comandos são enviados através de outros computadores rodando Matlab dentro de uma rede interna do laboratório, com a integração realizada através do RR.

Câmera

Na arquitetura proposta a câmera presente no ambiente local fornece a realimentação visual para o operador e a entrada para o controle por servovisão, sendo a informação capitada enviada através de um servidor desenvolvido com o RR. No caso da realimentação o vídeo é observado de forma contínua em um aplicativo cliente desenvolvido com a mesma biblioteca, enquanto que na servovisão imagens são amostradas diretamente no Matlab utilizando um nó nativo do RR.

2.3 Infraestrutura Remota

No ambiente remoto estão presentes os dispositivos mestre, Phantom e Kinect, integrados por uma GUI, com a qual também é implementada a servovisão.

Kinect

O Kinect é um dispositivo composto por uma câmera RGB e uma câmera e emissor infravermelho. Este gera uma nuvem de pontos cuja deformação permite ao dispositivo fornecer informações de profundidade junto a imagem do ambiente, sendo classificado como um dispositivo RGBD (em inglês, *red, green, blue and depth*). O Kinect foi desenvolvido para fornecer interface natural a jogos eletrônicos, a partir do rastreamento dos movimentos do usuário. O rastreamento é realizado através de 20 pontos compreendendo: as articulações dos braços e pernas, mãos, pés, cabeça, centro da linha dos ombros, centro de massa do corpo (próximo ao umbigo) e um ponto intermediário entre os dois últimos. Para estender esta capacidade a plataforma Windows foram utilizados o *driver* e kit de desenvolvimento oficiais da Microsoft, sendo a interface com Matlab realizada por um servidor TCP utilizando o RR.

O servidor fornece, quando requisitado, um vetor de 80 elementos com a posição e a qualidade da identificação de cada um dos 20 pontos,

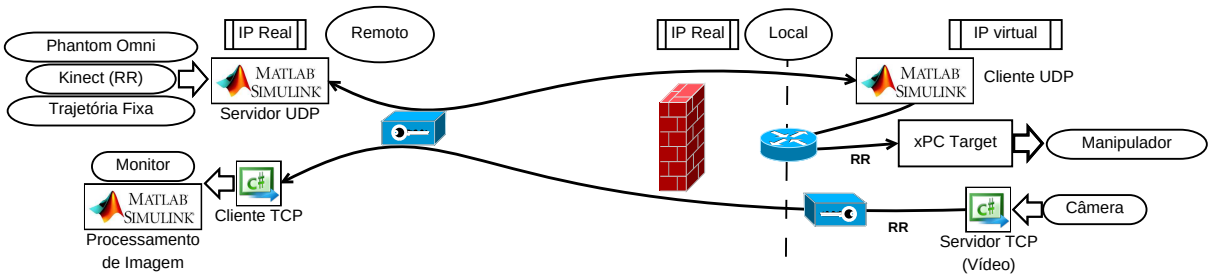

Figura 2: Diagrama representativo das vias de comunicação para a arquitetura de teleoperação proposta. **RR** indica a comunicação através do Robot Raconteur.

ISBN: 978-85-8001-069-5

para até dois esqueletos. Para a aplicação atual somente três pontos, mostrados na Figura 3, referentes ao braço direito são utilizados.

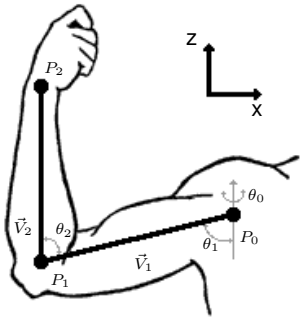

Figura 3: Pontos (P_0, P_1 e P_2) do esqueleto fornecido pelo Kinect e utilizados para controle do manipulador.

Para movimentar o manipulador são utilizados os ângulos do ombro (θ_0 e θ_1) e do cotovelo (θ_2), onde o braço é modelado por meio dos vetores:

$$\begin{aligned} \vec{V}_1 &= R_z(\theta_0)R_y(\theta_1)(-\vec{y}) \\ \vec{V}_2 &= R_z(\theta_0)R_y(\theta_1)R_y(\theta_2)(-\vec{z}) \end{aligned} \quad (1)$$

onde $R_z(\theta_0)$ é uma rotação elementar de θ_0 em torno do eixo $\vec{z}$ e $R_y(\theta_i)$ é uma rotação elementar de θ_i em torno do eixo $\vec{y}$ para $i = 1, 2$. Note que, os vetores $\vec{V}_1$ e $\vec{V}_2$, definidos na Figura 3, podem ser obtidos por cinemática inversa através dos subproblemas 2 e 1 de Paden-Kahan (Murray et al., 1994). Este modelo considera a posição zero do manipulador com o elo 1 alinhado ao eixo $-\vec{y}$ e o elo 2 ao eixo $\vec{z}$, além de um movimento inverso para os ângulos θ_1 e θ_2. Esta inversão é justificada para se obter melhor ergonomia em operações com o cotovelo do robô para o alto, por exemplo coletando objetos sobre uma mesa.

Phantom Omni

O Phantom é um mecanismo robótico com 6 juntas de revolução, 3 atuadas e 3 não atuadas. Possui uma biblioteca de desenvolvimento em C/C++ com a qual é possível programar aplicações gráficas que o utilizam como dispositivo de entrada de 6 graus de mobilidade com realimentação em 3 graus, ou controlar diretamente o *hardware*.

Para integrar o Phantom no sistema utilizou-se a capacidade do Matlab de reproduzir funções escritas em linguagem C pré-compiladas em arquivos MEX. A função desenvolvida inicializa o laço de controle do dispositivo e permite a leitura das posições das juntas e a escrita de um vetor de forças de forma assíncrona. Para ajustar a leitura dos ângulos à cinemática de diferentes manipuladores pode-se realizar rotações e translações independentes para cada junta.

Servovisão

Servovisão consiste em controlar a posição de um manipulador (ou de um robô móvel) a partir da realimentação visual obtida por um ou mais dispositivos de captura de imagem (Hutchinson et al., 1996). Neste trabalho propõe-se uma implementação transparente ao operador, o qual somente indica a posição inicial de um marcador colorido (vermelho, verde ou azul) no punho do manipulador e a posição final desejada (alvo). A posição inicial é necessária devido a execução do processamento de imagem em uma sub-janela, considerando-se um movimento lento em relação a amostragem das imagens. Este processamento é realizados em cinco etapas:

1. subtração da imagem em tons de cinza da camada da cor do marcador;
2. filtragem através de filtro da mediana;
3. transformação de escala de cinza para preto e branco;
4. remoção de objetos pequenos;
5. cálculo do centroide do objeto identificado.

Uma vez obtido o centroide do marcador no punho o erro de imagem (em pixels), obtido a partir da diferença entre a posição desejada e a posição do centroide, é transmitido para o ambiente local. Todo processamento é realizado em Matlab fazendo o uso do pacote *Image Processing* através das linhas de código apresentadas no Código 1.

Código 1: Processamento da imagem.

```
function [Xc,Yc,falha]=getcentroid(data,Xci,
    Yci,L)

Xci=round(Xci); Yci=round(Yci);
subwin_x=max(Xci-L,1):min(Xci+L,480);
subwin_y=max(Yci-L,1):min(Yci+L,640);
diff_im=imsubtract(data(subwin_x,subwin_y,1),
    rgb2gray(data(subwin_x,subwin_y,:)));

diff_im=medfilt2(diff_im, [4 4]);

diff_im=im2bw(diff_im,0.20);

diff_im=bwareaopen(diff_im,30);

stats=regionprops(diff_im, 'Centroid');

for object=1:length(stats)
    bc=stats(object).Centroid;
    bc=bc+[max(Yci-L,1)-1 max(Xci-L,1)-1];
end
if exist('bc','var')
    Xc=round(bc(1)); Yc=round(bc(2));
    falha=0;
else
    falha=1;
    Xc=Yci; Yc=Xci;
end
```

Interface Gráfica

A operação no ambiente remoto é centralizada em uma GUI desenvolvida em Matlab permitindo escolher a modalidade de controle, configurar o servidor e controlar o estado da garra no efetuador,

ISBN: 978-85-8001-069-5

além de fornecer informações para auxiliar o operador. A Figura 4 apresenta a GUI, onde na parte superior é realizada a seleção do controle, abaixo configuram-se as portas do cliente e do servidor, a posição inicial das juntas do manipulador e o período de amostragem. A opção de reiniciar sempre retorna a posição inical escolhida cada vez que a operação é parada e reiniciada, caso contrário isto ocorre apenas se o servidor for desligado.

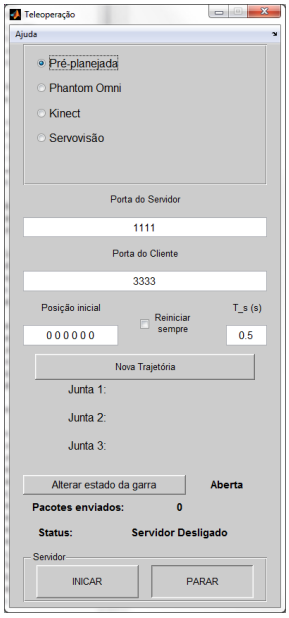

Figura 4: Interface gráfica na configuração para operação com trajetória preestabelecida.

Na região inferior da interface estão o controle da garra, uma indicação do estado da mesma, informações sobre as mensagens enviadas, o estado do servidor (desligado, esperando conexão ou conectado) e os botões para ligar e desligar o mesmo. A região central apresenta configurações e informações referentes a modalidade de controle selecionada, sendo alterada de acordo com cada uma. Na Figura 4 é apresentada a interface para o envio de uma trajetória preestabelecida e, neste caso, existe um botão para alterar a trajetória de cada junta e a cada instante é apresentado ao operador o valor do ângulo enviado. Para a operação com Phantom o painel mostra somente os ângulos de cada junta e para a servovisão o painel apresenta as posição do alvo e do marcador no punho do manipulador além do erro de imagem em pixels, como representado na Figura 5.

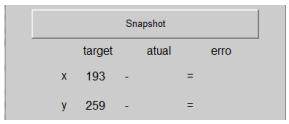

Figura 5: Campo de comandos e informações referentes a servovisão.

Para a servovisão também é apresentada uma figura com uma foto do ambiente local adquirida a partir da requisição do operador pelo botão *snapshot* ou ao se iniciar o procedimento. É através desta figura que a posição inicial do marcador no punho e o alvo são informados. As coordenadas do alvo podem ser alteradas a qualquer instante de tempo bastando selecionar um novo local da imagem, enquanto o marcador deve ser relocalizado apenas quando o processamento da imagem não o faz automaticamente.

Na operação através do Kinect nenhuma informação é mostrada no painel principal e o operador é realimentado por duas figuras. A primeira apresenta uma reconstrução do braço através dos pontos de esqueleto recebidos do dispositivo. A segunda varia de acordo com o momento de operação: ao ser iniciada apresenta a diferença absoluta em relação a última posição (ou a posição inicial informada na interface), após esta diferença atingir o limite de 10^o em todas as juntas a figura é trocada para o ângulo e os limites, superior e inferior, de cada uma. Exemplos destas figuras são apresentados na Figura 6, da esquerda para direita.

3 Testes Experimentais

Para o teste experimental da arquitetura proposta o ambiente de operação local utilizado pertence ao Rensselaer Polytechnic Institute (Troy, NY, EUA). O ambiente experimental é composto por dois manipuladores robóticos PUMA 560 montados sobre um sistema de trilho contendo uma junta prismática e duas de revolução (Wason and Wen, 2011). Entretanto, nos ensaios experimentais será utilizado apenas um dos manipuladores com a base fixa, uma câmera CCD e um sensor de força JR3 (JR3 Inc.). Para avaliar a viabilidade da estrutura estabelecida duas tarefas teste foram desenvolvidas, uma abrangendo os três primeiros itens enumerados na Seção 2 e a outra para a realimentação de força.

A primeira tarefa teste consiste em segurar um objeto alvo sobre um suporte em uma mesa e deixa-lo dentro de uma cesta (*pick-and-place*). O manipulador é inicializado em uma posição esticada para frente, com vetor de ângulos das juntas dado por $\theta = [0\ 0\ \frac{\pi}{2}\ 0\ \frac{\pi}{3}\ 0]^T\ rad$. Em seguida ele é alinhado ao objeto alvo em um posicionamento lateral com $\theta_1 = -\frac{\pi}{2}\ rad$ de forma a obter o marcador do punho no quadro da câmera. Então, o manipulador é levado até o objeto e, após agarrar o mesmo, o caminho inverso é realizado. Os grandes deslocamentos são realizados através do Phantom ou Kinect, enquanto o posicionamento fino utiliza a servovisão. As etapas estão exemplificadas na Figura 7 e os números representam as ações enumeradas na Seção 2.

Durante o experimento a etapa de servovisão foi realizada através de um algoritmo iterativo do tipo *look-and-move* (Hutchinson et al., 1996). O

ISBN: 978-85-8001-069-5

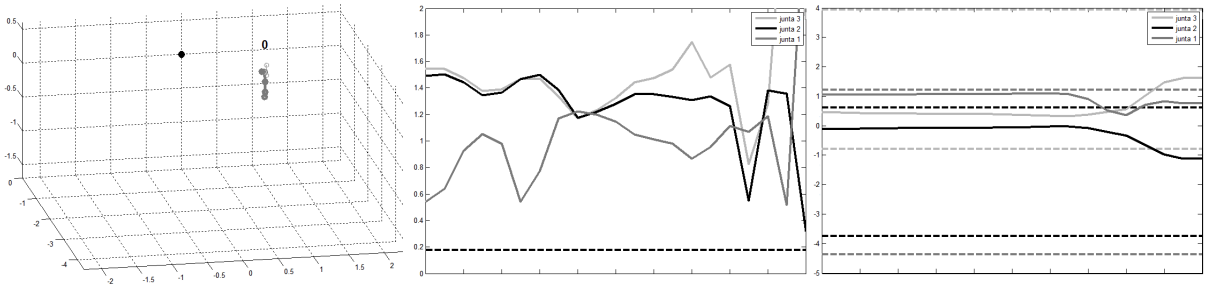

Figura 6: Exemplos de figuras para auxílio do operador durante o uso do Kinect. Painel esquerdo - reconstrução do braço através dos pontos fornecidos pelo dispositivo; painel central - diferença entre ângulos das juntas no Kinect e no manipulador antes de iniciar a operação; painel direito - ângulos enviados ao manipulador (linhas sólidas) e limites (linhas tracejadas) para cada junta. Nos dois últimos os eixos são radianos e amostras.

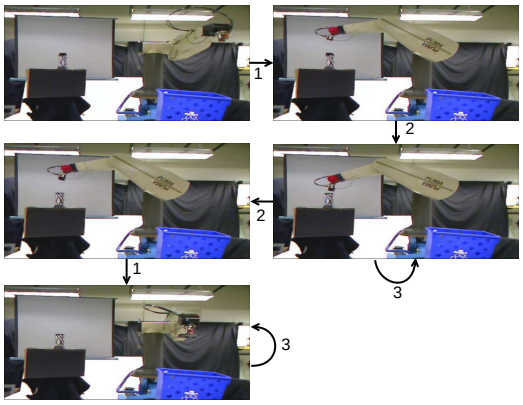

Figura 7: Exemplo do experimento com as diferentes modalidades de controle.

robô é controlado no espaço das juntas, em uma malha de posição com a referência (q_d) dada por:

$$q_d(k) = q(k-1) + K J_p^{-1}(q(k-1))\epsilon(k), \qquad (2)$$

sendo J_p o Jacobiano de posição do manipulador, K uma matriz de ganhos positivos, q a posição atual das juntas e ϵ o erro de imagem. Neste caso se o plano de trabalho e o plano da câmera são paralelos com uma rotação ϕ, $|\phi| < \frac{\pi}{2}$, é possível garantir que o erro na imagem converge para zero (Zachi et al., 2006).

A tarefa referente a realimentação de força é dividida em duas etapas: regulação e rastreamento. Em ambas um operador local atua diretamente no sensor de força instalado no efetuador do manipulador e as leituras deste são enviadas ao ambiente remoto servindo como entrada de força para o Phantom, cujos ângulos das juntas são enviados para manipulador local. Na primeira etapa o operador remoto deve manter a posição do dispositivo fixa, enquanto na segunda o dispositivo pode se movimentar, resultando em um movimento do manipulador.

4 Observações dos Experimentos

A seguir são apresentados resultados para avaliação da eficácia da estrutura proposta. Primeiramente, destaca-se que a estrutura utilizada permitiu a realização da tarefa de *pick and place*[1], porém devido ao grande atraso (próximo a $1\,s$), a baixa taxa de amostragem ($0,5\,s$) e a operação unilateral, somente com realimentação visual, os movimentos devem ser realizados em baixa velocidade.

Na Figura 8 são apresentados os ângulos enviados para o ambiente local durante a movimentação mestre/escravo, tanto com Phantom quanto com o Kinect. Observa-se que inicialmente os valores são constantes durante o período em que o operador ajusta o controlador a posição atual do manipulador, de forma a não ocorrem saltos na posição das juntas. Após este ajuste a junta 1 é levada a posição $-\frac{\pi}{2}\,rad$ durante a etapa inicial e retorna próximo a zero na etapa final, assim como definido na tarefa. Ressalta-se que a operação com o Kinect se mostrou mais difícil, com maiores oscilações e tempo de execução, pois o PUMA 560 não possuí todos os graus de mobilidade do braço humano tornando o movimento pouco natural, com o operador virando todo o corpo e dificultando a visualização da tela de operação. Além disso, apesar das inversões consideradas na cinemática inversa tornarem mais ergonômica a posição de cotovelo do manipulador para cima, elas tornam também os movimentos do braço menos intuitivos.

Durante a servovisão, devido ao procedimento de obtenção da imagem pelo nó Matlab do RR, o intervalo de tempo entre o envio de informações para o ambiente local aumenta, podendo aproximar-se de $3\,s$. Com isso o deslocamento do manipulador deve ser realizado com velocidade ainda menor de forma a manter o marcador dentro da sub-janela de processamento da imagem. Este atraso na atualização do erro, porém, não inviabiliza o controle como pode ser observado na Figura 9. A aproximação é realizada de forma gradual, enquanto a execução do trajeto de volta,

[1]Um vídeo com a tarefa de *pick and place* está em `http://www.youtube.com/watch?v=J5vRgD_kieY`. O uso de dois dispositivos Phantom controlando um manipulador de dois braços está em `http://www.youtube.com/watch?v=B1hhOHAGjnk`.

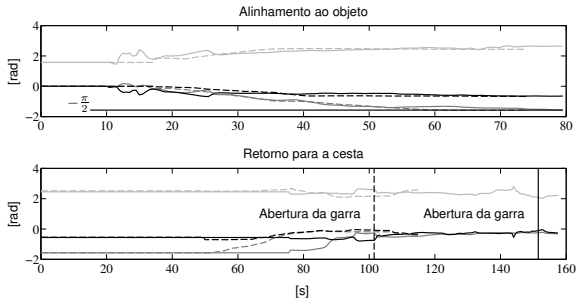

Figura 8: Trajetórias da juntas 1 (cinza escuro), 2 (preto) e 3 (cinza claro) durante a movimentação mestre escravo com o Kinect (contínuo) e Phantom (tracejado).

após o fechamento da garra, ocorre em uma etapa única. Uma alternativa para aumentar a velocidade do controlador é realizar o processamento local, com o operador remoto capturando a imagem somente para indicar o marcador e o alvo e sendo realimentado pela informação de erro de imagem.

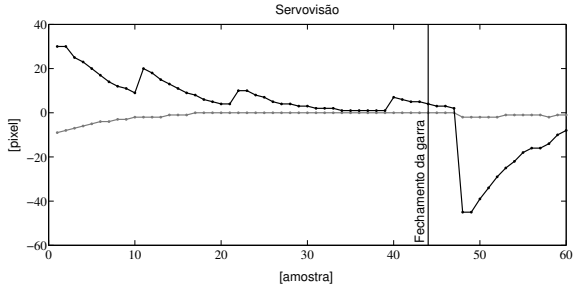

Figura 9: Erro em pixels nos eixos x (cinza) e y (preto) durante a servovisão.

Como a estratégia de servovisão adotada considera somente o movimento planar, ao fim da etapa inicial de movimentação mestre/escravo o manipulador deve estar alinhado ao objeto. A tarefa, porém, é dificultada pela falta da informação de profundidade na realimentação para o operador remoto. A profundidade da imagem pode ser acrescentada através de uma visão não-frontal a partir de outras câmeras (Bambang, 2007), de duas câmeras realizando estereoscopia (Hutchinson et al., 1996) ou utilizando a medida da área do marcador no punho (Zachi et al., 2006), entre outros.

O teste de realimentação de força mostrou que é possível reproduzir no Phantom as forças medidas no efetuador do manipulador. Além, observa-se que devido a pequena amplitude do movimento na etapa de regulação a operação é estável mesmo com o atraso na comunicação, o mesmo não ocorrendo durante o rastreamento. A instabilidade na teleoperação bilateral, com realimentação de força, devido a presença de atrasos pode ser eliminada utilizando uma das técnicas apresentadas por Hokayem and Spong (2006).

5 Comparação com Outras Arquiteturas

Taylor et al. (1999) e Bambang (2007) apresentam interfaces para operação de manipuladores através da internet limitadas a operação ponto a ponto, onde o operador fornece uma posição e deve esperar o fim do movimento para indicar um novo alvo. As posições podem ser definidas no espaço das juntas por meio de um formulário (Bambang, 2007) ou no espaço cartesiano também através de um formulário, clicando em uma imagem ou em um modelo virtual (Taylor et al., 1999). Já na interface de Safaric et al. (2003) a tarefa pode ser completamente planejada em um modelo virtual do manipulador, sendo posteriormente enviada e executada no robô real. Estas funções são realizadas através das opções de trajetória preestabelecida e servovisão, limitada a um plano por utilizar somente uma câmera. Em todos os trabalhos a interface gráfica é desenvolvida para navegadores utilizando CGI, HTML e Java.

A SMART, arquitetura modular de controle apresentada por Anderson (1995), é implementada em uma rede VME (*Virtual Module Europe*) de sensores, atuadores e computadores utilizando sistema operacional VxWorks. A interface de comando está em uma máquina que se conecta a rede através do padrão Ethernet. É possível criar malhas para teleoperação autônoma ou bilateral através da interligação de blocos pré-programados. Após conectar todos os blocos são gerados códigos em linguagem C, que podem ser modificados para atender especificações da tarefa e então compilados e enviados para os respectivos computadores ou dispositivos na rede.

Uma arquitetura que combina operação bilateral e servovisão é apresentada por Goto et al. (2010). O mestre e o escravo são controlados por dois computadores rodando Linux de tempo real. A comunicação entre o ambiente remoto e o local, das posições dos manipuladores, é realizada por protocolo TCP, com previsão para implementação em internet, mas testado somente em rede local. Como proposto aqui, a servovisão é responsável pelo posicionamento fino, porém é utilizada uma câmera no punho que é alinhada ao alvo de forma automática após este entrar no quadro da imagem. O processamento é local em um terceiro computador se comunicando via TCP com o controlador do manipulador escravo.

Realizando somente o controle bilateral Oboe and Fiorini (1998) utilizam um sistema em tempo real criado pelos autores que se comunica com a internet através de uma *socket*. Munir and Book (2003) utilizam um processo em tempo real no Windows NT com o auxílio do Hyperkernel, um programa que cria um *kernel* paralelo ao do sistema operacional, realizando a comunicação com a internet através do NT que acessa uma memória compartilhada com o Hyperkernel. Em ambos

ISBN: 978-85-8001-069-5

um computador remoto retransmite o sinal de controle do manipulador mestre, através da internet, para uma simulação do mesmo utilizada como escravo, que realimenta o sistema de controle diretamente. Hirche et al. (2004) controlam um manipulador de 7 graus de mobilidade em um espaço operacional de 2 graus e Stanczyk and Buss (2004) estendem o controle através de um manipulador mestre com 6 graus de mobilidade. Os trabalhos também realizam o controle dos dispositivos robóticos utilizando a combinação Matlab/Simulink *XPC Target*, porém não especificam como é realizada a comunicação UDP entre as máquinas e consideram a realimentação visual um problema já resolvido.

Por outro lado, a nova arquitetura proposta neste trabalho realiza somente controle unilateral, porém já apresenta uma via para a realimentação de força possibilitando a implementação do caso bilateral. É válido ressaltar que as diferentes interfaces propostas anteriormente não apresentam visualização *online* do ambiente de operação. As imagens são mostradas no início e fim do movimento (Taylor et al., 1999), um vídeo da tarefa é mostrado após a sua execução (Safaric et al., 2003), apenas um modelo virtual é apresentado ao operador (Bambang, 2007) ou o mestre e o escravo estão no mesmo ambiente (Oboe and Fiorini, 1998; Munir and Book, 2003; Goto et al., 2010).

6 Conclusões

A estrutura para teleoperação proposta neste trabalho se mostrou capaz de realizar operações unilaterais com movimentação mestre/escravo e servovisão utilizando realimentação visual. Ela permite ainda o envio de forças medidas no manipulador local que são reproduzidas através do Phantom para o operador remoto. Todas as etapas são realizadas com o Matlab ou aplicativos executáveis, gratuitos e de código aberto (RR), permitindo a prototipagem rápida e a extensão da arquitetura em uma linguagem de alto nível.

Agradecimentos

Este trabalho foi financiado pelo CENPES/PETROBRAS no convênio COPPETEC intitulado "Sistemas de Produção *Offshore* com Alto Grau de Automação", e pelo CNPq, FAPERJ e CAPES. Os autores agradecem ao Prof. John Wen e ao Dr. John Wason pelo apoio para realizar os testes experimentais no CATS/RPI.

Referências

Anderson, R. (1995). Smart: a modular control architecture for telerobotics, *IEEE Robot. Autom. Mag.* **2**(3): 10 –18.

Bambang, R. (2007). Development of architectures for internet telerobotics systems, *J. Bionic Eng.* **4**(4): 291–297.

Bellingham, J. G. and Rajan, K. (2007). Robotics in remote and hostile environments, *AAAS Science Magazine* **318**(5853): 1098–1102.

Goto, S., Naka, T., Matsuda, Y. and Egashira, N. (2010). Teleoperation system of robot arms combined with remote control and visual servo control, *Proc. of SICE Annual Conf.*, pp. 1975 –1981.

Hirche, S., Stanczyk, B. and Buss, M. (2004). Haptic tele-assembly over the internet, *Proc. of EuroHaptics Conf.*, pp. 417–421.

Hokayem, P. and Spong, M. (2006). Bilateral teleoperation: An historical survey, *Automatica* **42**(12): 2035–2057.

Hutchinson, S., Hager, G. and Corke, P. (1996). A tutorial on visual servo control, *IEEE Trans. Robot. Autom.* **12**(5): 651 –670.

Iborra, A., Pastor, J., Alvarez, B., Fernandez, C. and Merono, J. (2003). Robots in radioactive environments, *IEEE Robot. Autom. Mag.* **10**(4): 12–22.

Munir, S. and Book, W. J. (2003). Control techniques and programming issues for time delayed internet based teleoperation, *J. DYN. SYST.* **125**(2): 205–214.

Murray, R. M., Li, Z. and Sastry, S. S. (1994). *A mathematical introduction to robotic manipulation*, Boca raton, FL, USA.

Niemeyer, G., Preusche, C. and Hirzinger, G. (2008). Telerobotics, *in* B. Siciliano and O. Khatib (eds), *Springer Handbook of Robotics*, Springer, pp. 741–757.

Oboe, R. and Fiorini, P. (1998). A design and control environment for internet-based telerobotics, *Int. J. Robot. Res.* **17**(4): 433–449.

Preusche, C., Reintsema, D., Landzettel, K. and Hirzinger, G. (2006). Robotics component verification on iss rokviss - preliminary results for telepresence, *Proc. of IROS*, pp. 4595–4601.

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

Safaric, R., Hedrih, I., Klobucar, R. and Sorgo, B. (2003). Remote controlled robot arm, *Proc. of ICIT*, Vol. 2, pp. 1202–1207.

Skourup, C. and Pretlove, J. (2009). The robotized field operator - greater safety and productivity by design, *Productivity Solutions - ABB Process Automation* pp. 68–73.

Soetens, P. and Bruyninckx, H. (2005). Realtime hybrid task-based control for robots and machine tools, *Proc. of ICRA*, pp. 260–265.

Stanczyk, B. and Buss, M. (2004). Development of a telerobotic system for exploration of hazardous environments, *Proc. of IROS*, Vol. 3, pp. 2532–2537.

Taylor, K., Dalton, B. and Trevelyan, J. (1999). Web-based telerobotics, *Robotica* **17**(1): 49–57.

Trevelyan, J., Kang, S.-C. and Hamel, W. R. (2008). Robotics in hazardous applications, *in* B. Siciliano and O. Khatib (eds), *Springer Handbook of Robotics*, Springer, pp. 1101–1126.

Wason, J. and Wen, J. (2011). Robot raconteur: A communication architecture and library for robotic and automation systems, *Proc. of CASE*, pp. 761–766.

Zachi, A. R. L., Hsu, L., Lizarralde, F. and Leite, A. C. (2006). Adaptive control of nonlinear visual servoing systems for 3D cartesian tracking, *Automation & Control Brazilian Mag.* **17**(4): 381–390.