CONTROLE EM TEMPO REAL DE ROBÔS MÓVEIS VIA *INTERNET*

Vinícius M. de Oliveira[*]
vinicius@ieee.org

Walter F. Lages[†]
w.fetter@ieee.org

[*]Fundação Universidade Federal do Rio Grande
Departamento de Matemática
Av. Itália, km 8
Rio Grande, RS 96200-900 BRASIL

[†]Universidade Federal do Rio Grande do Sul
Departamento de Engenharia Elétrica
Av. Osvaldo Aranha, 103
Porto Alegre, RS 90035-190 BRASIL

ABSTRACT

This article presents the implementation of a real time controller to mobile robots using an IP network to send data to the actuators and to receive data from the sensors. The elements are connected to an IP network and the real time control loops are configured by software. The time requirements of the control computation and robot simulation tasks are enforced by using the Real Time Application Interface (RTAI) Linux in LXRT hard real time mode. RTAI is also used for monitoring the time behavior of the network packets, because the real time control loops must run at an accurate rate to avoid problems in the system performance or stability. At last, some time measurements are depicted to analize the time behavior and performance of the control architecture.

KEYWORDS: real-time systems, Internet, mobile robot control

RESUMO

Este trabalho apresenta a implementação de um controlador em tempo real para robôs móveis baseado em rede IP para enviar dados aos atuadores e para receber dados dos sensores. Os elementos estão conectados em uma rede IP e as malhas de controle em tempo real são configuradas por *software*. Os requisitos de temporização das tarefas responsáveis pelo cômputo da lei de controle e pela simulação do robô são implementados por meio do *Real Time Application Interface* (RTAI) Linux, configurado no modo LXRT *hard real time*. O RTAI também é utilizado para monitorar o comportamento temporal dos pacotes da rede, pois o controle em tempo real deve executar em uma freqüência precisa, a fim de se evitar problemas na *performance* do sistema, e até mesmo na sua estabilidade. Apresenta-se o resultado obtido em duas simulações distintas, onde, na primeira, as duas tarefas estão executando num mesmo *host* e na segunda as tarefas ficam em *hosts* distintos. Por fim, algumas medidas de tempo são apresentadas para análise do comportamento temporal da arquitetura de controle adotada nesse trabalho.

PALAVRAS-CHAVE: tempo-real, *Internet*, controle de robôs móveis

1 INTRODUÇÃO

Ao longo dos últimos anos a *Internet* teve um crescimento muito grande, passando a fazer parte do dia-a-dia das pessoas, seja na realização de tarefas laboriosas ou mesmo na busca por conhecimento, até mesmo servindo como um meio de entretenimento para muitas pessoas. Paralelamente a esse grande avanço, o interesse de pesquisadores da área de controle também começou a envolver sistemas conectados em rede.

Existem diversas maneiras de se utilizar a *Internet* na área de controle de sistemas. Vários trabalhos apresentam a aplicação da *Internet* para teleoperação, por exemplo, utilizando uma página *web* para passar ao usuário as informações do sistema controlado e o usuário comanda, também por meio da página, o sistema remoto (Grange et al., 2000), controlando-o "manualmente" via *web*.

A característica de um sistema de controle em rede é apresentar uma ou mais malhas de controle fechadas por meio de um canal de comunicação serial. A utilização de redes de computadores em sistemas de controle, substituindo a tradicional ligação ponto-a-ponto, traz grandes vantagens:custos menores, maior confiabilidade, instalação e manutenção mais simples, redução no peso e na potência requerida. Sistemas de controle em rede podem ser aplicados em plantas industriais, aeronaves, etc.

A motivação deste trabalho é a utilização de redes convencionais de computadores, baseadas no protocolo IP, para implementar um sistema distribuído de controle. A simplicidade de instala-

ção e de manutenção são fatores importantes em se utilizar um sistema distribuído de controle baseado no protocolo IP.

Esta área é relativamente nova para a comunidade acadêmica de controle, tendo poucos artigos abordando o assunto, mas não é tão novo assim na indústria (Zhang et al., 2001). Sendo assim, existe ainda um grande espaço para desenvolvimento de pesquisa nesse tópico. A idéia consiste em utilizar uma rede de comunicação para transmissão de sinais de controle.

Tecnologias de redes convencionais de computadores têm sido adaptadas para tal aplicação, como *daisy-chained* RS-232, *multi-drop* RS-485, IEEE-488 e Ethernet e suas extensões (*wireless* Ethernet - IEEE-802.11). Além disso, redes especializadas foram desenvolvidas para aplicações industriais, como *CAN* (*CAN*, Último acesso em Abril/2003), *Foundation Fieldbus* (*Fieldbus*, Último acesso em Abril/2003) e *Profibus* (*Profibus*, Último acesso em Abril/2003).

Nesse trabalho apresenta-se o controle não-linear de um robô móvel com acionamento diferencial através da *Internet*, analisando-se os vários atrasos de tempo presentes em um sistema de controle sobre a *Internet* (Alt et al., 2002). Os atrasos e suas variações (*jitter*) ocorrem em um sistema de controle real e seus efeitos são analisados.

Os componentes do sistema estão conectados a uma rede IP e as malhas de controle em tempo real são configuradas por *software*. Os requisitos de tempo das malhas de controle, que deve rodar a uma freqüência de 100Hz, são garantidos pelo uso do RTAI no modo LXRT *hard real-time*.

Na seqüência serão apresentadas a arquitetura de controle via *Internet* utilizada nesse trabalho, o modo como o controlador é implementado, o robô móvel controlado e a lei de controle implementada. Ao fim tem-se os resultados obtidos com a utilização da arquitetura, mostrando medições de tempo para análise do comportamento da estrutura de controle. Na primeira simulação, tanto a tarefa Controlador quanto o *daemon* do robô estão executando na mesma máquina, o que praticamente elimina problemas de comunicação como atraso de datagramas, etc. Já na segunda simulação realizada, o *daemon* do robô executa numa máquina diferente da tarefa Controlador, a fim de se ter a influência da rede de comunicação. Com os resultados dessas simulações, pode-se verificar o impacto do atraso de comunicação introduzido pela rede IP na malha de controle.

2 ARQUITETURA DE CONTROLE VIA *INTERNET*

A figura 1 apresenta a arquitetura de controle utilizada nesse trabalho. A arquitetura consiste em se implementar o controle do robô em uma máquina conectada à *Internet*, e não diretamente ao robô, estando em qualquer ponto da rede. O robô (nesse caso a máquina que simula o robô) também deve estar conectada à *Internet*. Deste modo, o controlador envia ao robô as informações referentes aos valores de torque que devem ser fornecidos pelos motores e a cada período de amostragem o robô envia, à máquina que está computando a lei de controle, as informações obtidas pelos sensores do robô.

As redes de computadores para sistemas de controle diferem em muitos aspectos das redes de computadores tradicionais como na

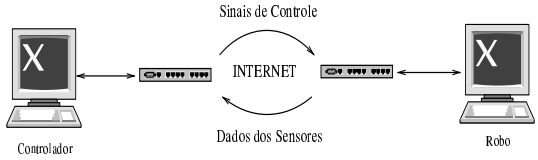

Figura 1: Arquitetura de Controle via *Internet*.

geração freqüente de pequenos pacotes que apresentam requisitos de tempo real. Um importante aspecto a ser considerado é que o objetivo em uma rede de controle não é transmitir dados digitais, mas sinais de controle e/ou sensores por meio digital. Sendo assim, métricas comuns em redes de computadores, como quantidade de dados e taxa de transferência, são menos importantes em uma rede para controle. As métricas adequadas para se avaliar a o desempenho de uma rede de controle estão relacionadas ao desempenho do controlador, como *overshoot*, tempo de subida, tempo de acomodação, etc.

Para sistemas de controle, as redes devem satisfazer a dois principais critérios: tempo de atraso limitado e garantia de transmissão.Em outras palavras, espera-se que uma mensagem seja transmitida com sucesso, dentro de um período de tempo limite (Lian et al., 2001). Vários protocolos têm sido adaptados (Ethernet–IEEE 802.3:CSMA/CD, Token Bus–IEEE 802.4), Token Ring–IEEE 802.5) e mesmo desenvolvidos (CAN, PROFIBUS, Foundation Fieldbus) para satisfazer tais restrições.

A rede lógica empregada nesse trabalho utiliza o protocolo não-determinístico Ethernet nas camadas física e de enlace, o protocolo IP na camada de rede e o protocolo UDP da camada de transporte. O protocolo IP tem por função transferir datagramas (blocos de dados) do *host* origem ao *host* destino, identificados pelos respectivos endereços IP. O protocolo se encarrega tanto da fragmentação da mensagem na origem quanto da reordenação e remontagem dos datagramas no destino. Cabe mencionar, ainda, que o protocolo IP é sem conexão e a comunicação não é confiável, não existindo mecanismo de controle de erros além de um *checksum* de cabeçalho (Soares et al., 1997).

Como protocolo de transporte optou-se por utilizar o *User Datagram Protocol* (UDP), dada a sua simplicidade. O fato deste protocolo não apresentar correção de erros nem recuperação de pacotes perdidos e/ou corrompidos durante a transmissão, no caso do controle em rede, poderia se tornar um problema, mas considerando que os sinais transmitidos (controle e sensores) são sinais contínuos, pretende-se a utilização de métodos para estimar os valores referentes aos pacotes perdidos. O uso de métodos de estimação mostra-se mais efetiva do que a substituição do protocolo UDP pelo TCP ou mesmo por algum método para recuperação de erros baseado em retransmissão, visto que, devido a restrições temporais, não é conveniente realizar-se retransmissões. A figura 2 mostra as camadas e os protocolos utilizados para o controle de robôs em tempo real.

3 CONTROLE EM REDE

A implementação dos programas que rodam sobre a camada UDP foi projetada com base na arquitetura cliente-servidor. O sistema que simula o robô é o servidor, enquanto o cliente executa funções de controle do robô. Os comandos enviados pelo controlador são ligar robô, desligar robô e aplicar uma determi-

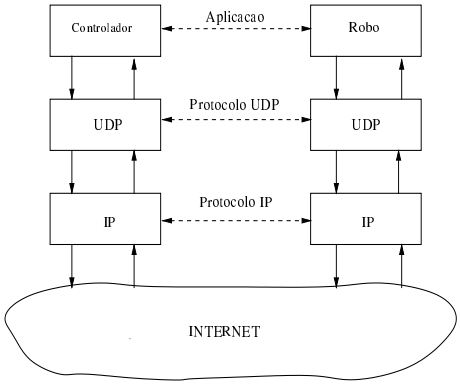

Figura 2: Camadas e protocolos da rede de controle.

nada tensão nos motores. A aplicação cliente, além de executar o controle do robô, serve de *interface* para o usuário. A figura 3 mostra a estrutura das tarefas **Controlador** e **Robô**.

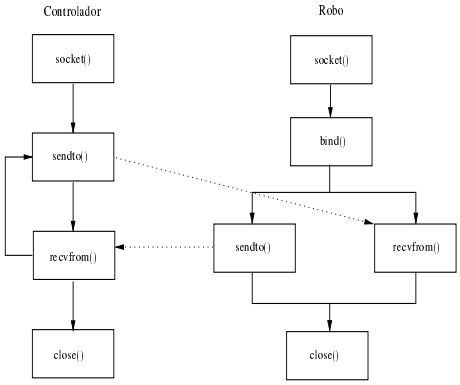

Figura 3: Estrutura do cliente e do servidor.

Em uma aplicação UDP típica, o cliente não estabelece conexão com o servidor, visto que esse protocolo é sem-conexão. Sendo assim, o controlador simplesmente envia dados ao robô através da função *sendto()*. De maneira semelhante, o robô executa a função *recvfrom()* para receber os dados do controlador.

3.1 *Daemon* Robô

O *daemon* Robô consiste em duas *threads*, sendo uma responsável por ficar à espera de dados do cliente e simular o robô, enquanto a outra é encarregada de obter os valores dos sensores e enviá-los ao cliente, conforme pode ser visto na figura 3.

Ao se inicializar o robô, há a criação e configuração de dois *sockets*, um para a comunicação entre o controlador e a *thread* que recebe as informações, e outro para a comunicação entre a *thread* que envia as informações e o controlador. Logo em seguida há a configuração do processo como sendo de tempo real. As *threads* são inicializadas e a responsável pelo recebimento das informações fica esperando o controlador enviá-las. Uma vez recebida a informação para ligar o robô, essa *thread* fica em um *loop*, recebendo os sinais do controlador, até que este envie a ordem de desligar o robô.

Já a *thread* responsável pelo envio das informações do robô para o controlador, após ser inicializada, fica bloqueada até que o robô seja ligado, pois só depois de acionado é que o robô poderá periodicamente enviar (com um período de 10ms) as informações ao controlador. Quando o controlador enviar ao robô a informação para desligar, essa *thread* também terminará.

3.2 A Tarefa Controlador

A arquitetura prevê que todas as operações de controle do robô seja executadas na tarefa **Controlador**. Esta tarefa basicamente implementa uma função que fica em *loop*, aguardando a chegada de datagramas na porta de comunicação.

Ao ser inicializado, o controlador cria um *socket*, definindo a porta local que será utilizada. Após, envia o comando para ligar o robô e, com base na posição inicial do robô, computa a lei de controle e a envia. Depois, coloca-se à espera da informação do robô, para novamente calcular a lei de controle, assim por diante, até o término da simulação.

Este trabalho empregou um controlador para um robô móvel não-holonômico, cujo modelo pode ser encarado como um modelo da dinâmica em cascata com o modelo da cinemática (conforme 4). Com base nesta característica, o controlador utiliza na malha de controle da cinemática uma transformação descontínua e na malha de controle da dinâmica faz uso de uma rede neural com aprendizado *online*. A lei de controle resultante será apresentada diretamente e sem as provas de estabilidade. Para maiores detalhes da técnica de controle utilizada veja (de Oliveira et al., 2001).

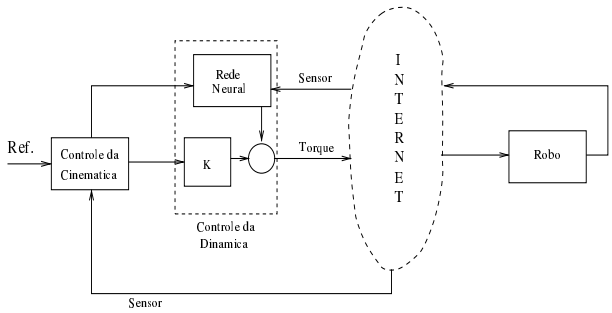

Figura 4: Estrutura de controle utilizada.

A lei de controle é dada pela equação $\tilde{\tau} = \hat{f} + \mathbf{K_4} e_c - \gamma$ onde $\mathbf{K_4}$ é uma matriz de ganhos, positiva definida, e $\hat{f}(x)$ é uma estimativa de termos da dinâmica do robô, dada pelo aprendizado da rede neural. O sinal γ está presente para garantir robustez ao controle.

4 SIMULAÇÕES

A fim de validar a arquitetura de controle utilizada nesse trabalho, realizou-se dois tipos de simulação, para uma melhor avaliação do uso desta arquitetura em sistemas reais.

A primeira simulação consistiu em executar tanto o processo referente ao controlador quanto o processo que simula o robô na mesma máquina. Já na segunda simulação a tarefa controlador e a tarefa robô foram executadas em máquinas diferentes.

Para avaliação do comportamento da arquitetura de controle em rede foram realizadas algumas medições de tempo, a saber:

1. diferença de tempo entre dois envios consecutivos de sinais

de controle para o *daemon* do robô;

2. diferença de tempo entre duas recepções consecutivas de informações dos sensores;

3. diferença de tempo entre a recepção das informações dos sensores e o envio dos sinais de controle, ou seja, o tempo necessário para se computar a lei de controle;

4. diferença de tempo entre dois envios consecutivos de informações de sensores para a tarefa Controlador;

5. diferença de tempo entre duas recepções consecutivas de sinais de controle;

6. diferença de tempo entre o envio das informações dos sensores e a recepção dos sinais de controle, ou seja, o tempo gasto para simulação do robô;

As medidas 1, 2 e 3 são realizadas na tarefa Controlador e as medidas 4, 5 e 6 são realizadas pelo *daemon* do robô.

4.1 Tarefas na mesma Máquina

As medidas de tempo realizadas com a execução das tarefas Controlador e Robô na mesma máquina são mostradas a seguir. Na figura 5 tem-se o tempo entre dois envios consecutivos de informações de sensores por parte do *daemon* do robô à tarefa Controlador. O tempo entre duas recepções consecutivas está apresentado na figura 6. Na figura 7 tem-se tempo entre a recepção dos sinais de controle e o envio das informações de sensores para a tarefa Controlador.

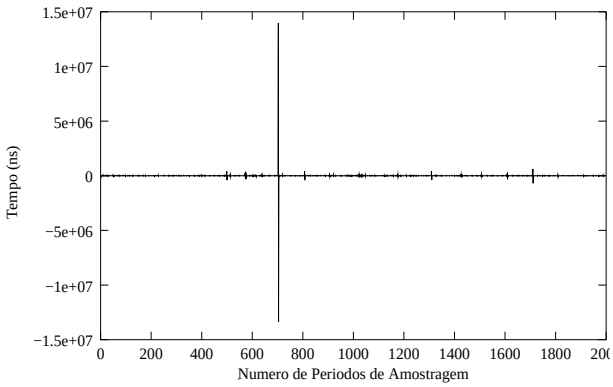

Figura 5: Tempo entre dois envios consecutivos de informações do robô para o controlador.

Na figura 8 tem-se o tempo entre dois envios consecutivos de sinais de controle por parte da tarefa Controlador ao *daemon* do robô. O tempo entre duas recepções consecutivas de por parte da tarefa Controlador está apresentado na figura 9. Na figura 10 tem-se tempo entre a recepção das informações dos sensores e o envio dos sinais de controle para o *daemon* do robô.

4.2 Tarefas em Máquinas Diferentes

As medidas de tempo realizadas com a execução das tarefas Controlador e Robô executando em máquinas distintas são mostradas a seguir. Na figura 11 tem-se o tempo entre dois envios consecutivos de informações de sensores por parte do *daemon* do

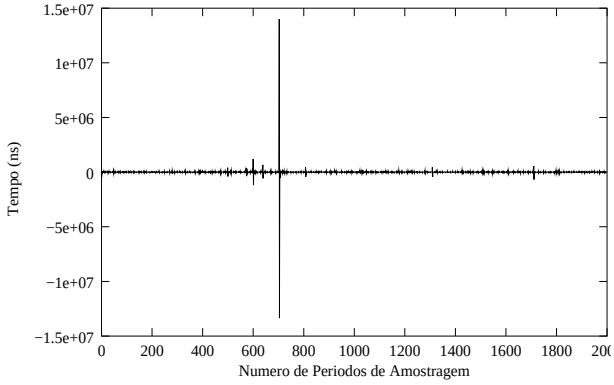

Figura 6: Tempo entre duas recepções de informações do controlador.

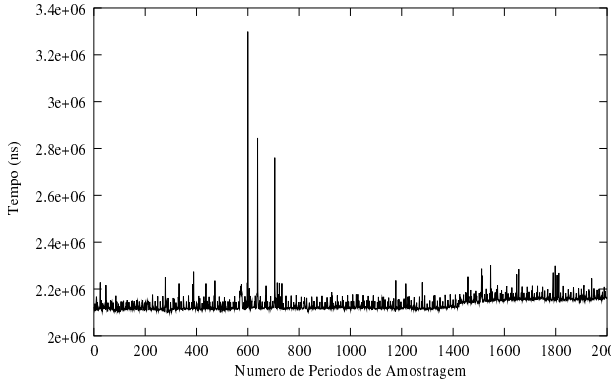

Figura 7: Tempo entre recepção do sinal de controle e o envio das informações dos sensores.

robô à tarefa Controlador. O tempo entre duas recepções consecutivas está apresentado na figura 12. Na figura 13 tem-se tempo entre a recepção dos sinais de controle e o envio das informações de sensores para a tarefa Controlador.

Na figura 14 tem-se o tempo entre dois envios consecutivos de sinais de controle por parte da tarefa Controlador ao *daemon* do robô. O tempo entre duas recepções consecutivas de por parte da tarefa Controlador está apresentado na figura 15. Na figura 16 tem-se tempo entre a recepção das informações dos sensores e o envio dos sinais de controle para o *daemon* do robô.

É importante salientar que as medidas de tempo mostradas nas figuras acima foram obtidas em situação real, utilizando a *Inter-*

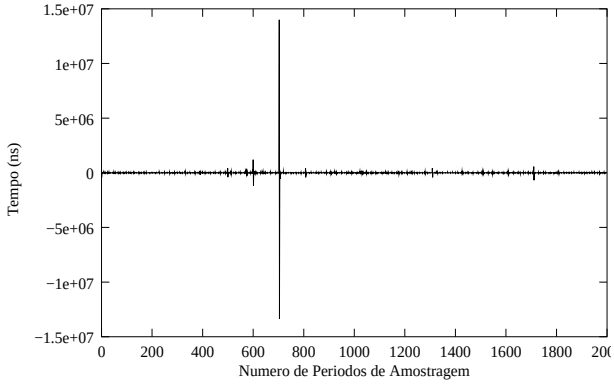

Figura 8: Tempo entre dois envios consecutivos de sinais de controle do controlador para o robô.

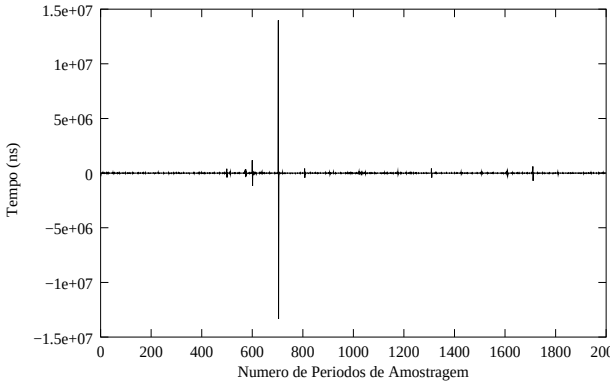

Figura 9: Tempo entre duas recepções de dados dos sensores.

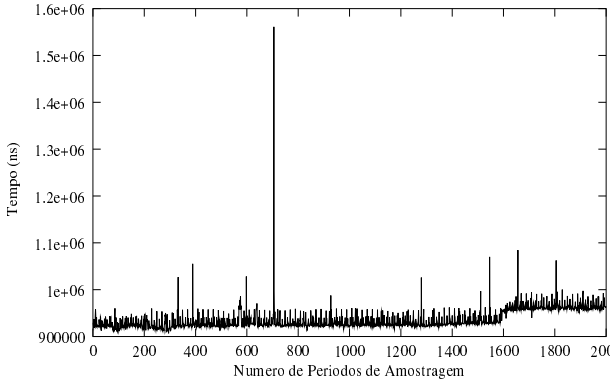

Figura 10: Tempo entre a recepção das informações dos sensores e o envio dos sinais de controle.

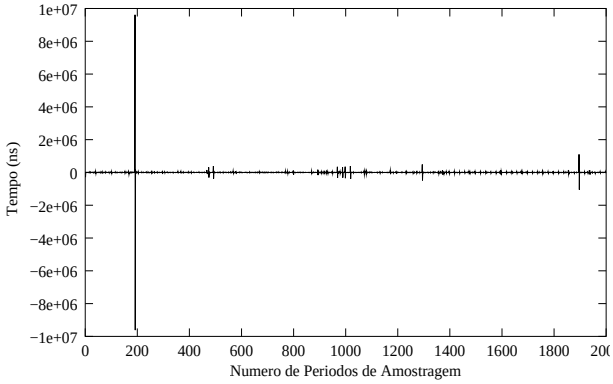

Figura 11: Tempo entre dois envios consecutivos de informações do robô para o controlador.

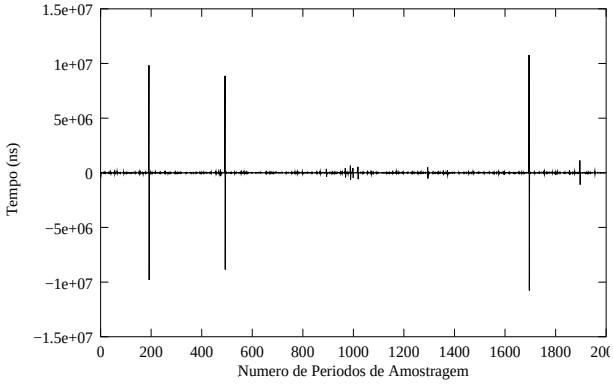

Figura 12: Tempo entre duas recepções consecutivas de informações do controlador.

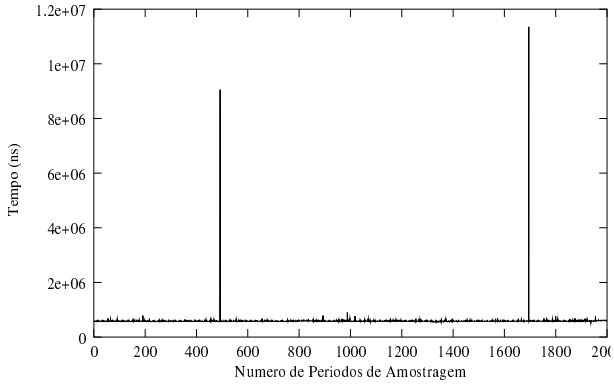

Figura 13: Tempo entre recepção do sinal de controle e o envio das informações dos sensores.

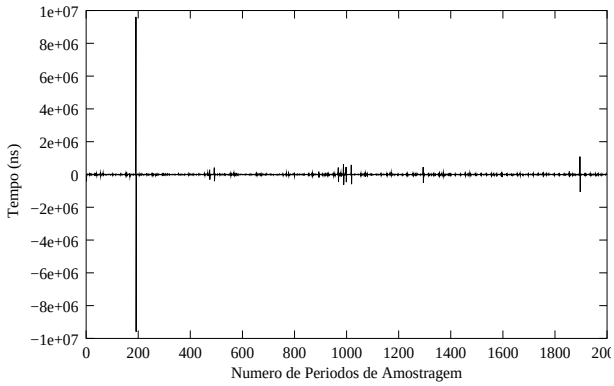

Figura 14: Tempo entre dois envios consecutivos de sinais de controle do controlador para o robô.

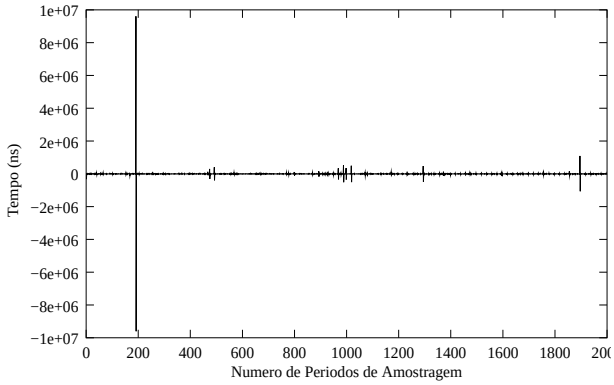

Figura 15: Tempo entre duas recepções de dados dos sensores.

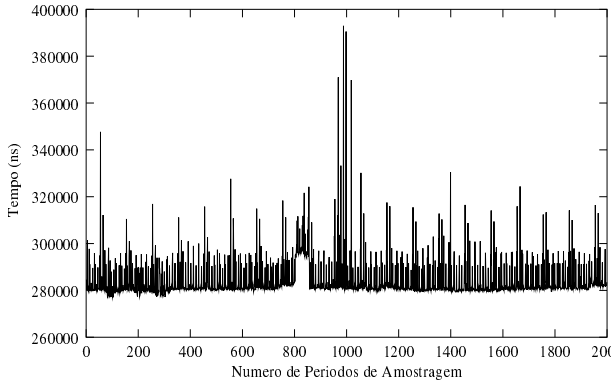

Figura 16: Tempo entre a recepção das informações dos sensores e o envio dos sinais de controle.

net como meio de ligação entre a máquina que executa o controle do robô e a máquina que simula do robô.

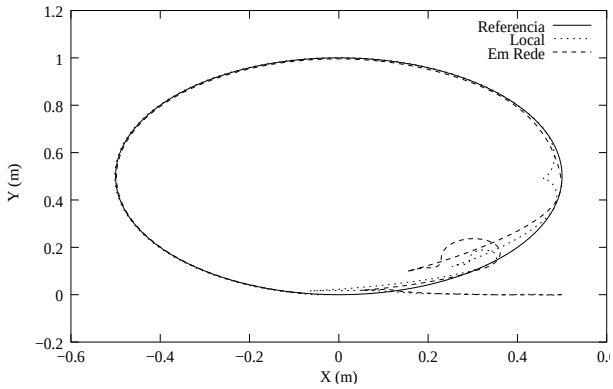

Figura 17: Movimento realizado pelo robô.

Na figura 17 apresenta-se a comparação do movimento realizado pelo robô em duas situações, sendo a trajetória de referência imposta ao robô representada pela linha cheia. A trajetória representada pela linha pontilhada foi obtida sem utilizar a arquitetura em rede e a linha cheia apresenta o movimento realizado pelo robô utilizando-se a arquitetura de controle apresentada nesse trabalho. Observa-se que as trajetórias realizadas são praticamente idênticas, o que indica um bom desempenho do controle via *Internet*.

5 CONCLUSÕES

A arquitetura de controle de robôs móveis baseada em um sistema de controle sobre rede IP, mais precisamente sobre a *Internet*, apresenta um comportamento aceitável, com base nas medições de tempo obtidas. Os requisitos de temporização das tarefas foram assegurados pelo uso do RTAI Linux no modo *hard real-time*.

As medições de tempo apresentadas nesse trabalho, mostram resultados satisfatórios, embora tenham sido obtidos em situações pouco práticas, isto é, com baixa carga na rede de comunicação. Como continuidade desse trabalho se pretende aplicar métodos para minimizar o impacto causado pelo atraso e/ou perda de datagramas durante a comunicação, além da utilização de técnicas, como filtro de Kalman, para estimação do conteúdo desses pacotes perdidos, uma vez que os sinais envolvidos são contínuos.

Para uma melhor avaliação da arquitetura de controle apresentada, é necessário se realizar uma análise com base em critérios de desempenho de sistemas de controle, como amortecimento, *overshoot*, etc, para se verificar a aplicação prática desse controle via *Internet*. Por fim, se pretende executar testes práticos, ou seja, em um robô real, para validar os resultados obtidos por simulação.

REFERÊNCIAS

Alt, G. H., da Silva Guerra, R. and Lages, W. F. (2002). An assessment of real-time robot control over IP network, *Proceedings of the Fourth Real-Time Linux Workshop*, University of Boston, Computer Science Department, Boston, USA, pp. 1–17.

CAN (Último acesso em Abril/2003).
*http://www.can.bosch.com

de Oliveira, V. M., de Pieri, E. R. and Lages, W. F. (2001). Controle em malha fechada de robôs móveis utilizando redes neurais e transformação descontínua, *Anais do V Simpósio Brasileiro de Automação Inteligente*.

Fieldbus (Último acesso em Abril/2003).
*http://www.fieldbus.org

Grange, S., Fong, T. and Baur, C. (2000). Effective vehicle teleoperation on the world wide web, *Proceedings of the International Conference on Robotics & Automation*, pp. 2007–2012.

Lian, F.-L., Moyne, J. R. and Tilbury, D. M. (2001). Performance evaluation of control networks: Ethernet, controlnet and devicenet, *IEEE Control Systems Magazine* **21**(1): 66–83.

Profibus (Último acesso em Abril/2003).
*http://www.profibus.com

Soares, L. F. G., Lemos, G. and Colcher, S. (1997). *Redes de Computadores: das LANs, MANs e WANs às Redes ATMs*, 2^a edn, Editora Campus.

Zhang, W., Branicky, M. S. and Phillips, S. M. (2001). Stability of networked control systems, *IEEE Control Systems Magazine* **21**(1): 84–99.