Interfaceamento amigável para robôs manipuladores

Aluno: Rafael Ramos de Matos
Orientador: João Yoshiyuki Ishihara

Universidade de Brasília

Palavras-chave: iniciação científica, robótica, manipuladores, ROS.

***Resumo*—Este trabalho aborda o estudo e implementação de uma interface de comunicação para o manipulador UR3 da Universal Robots utilizando ROS (Robot Operating System). Essa interface permite a coleta de dados do estado das variáveis de junta (posição, velocidade, força aplicada) do robô e o controle do mesmo através de um cabo Ethernet.**

A construção dessa interface se faz necessária devido a arquitetura de fábrica do robô UR3 ser de difícil acesso para coleta e envio dados. Partindo dessa premissa, foi implementado uma interface de comunicação para o robô manipulador UR3 da *Universal Robots* com intuito de fornecer diversas informações para o seus usuários.

Com essas informações, pode-se fazer o monitoramento do estado do robô, seu controle (tanto cinemático, quanto dinâmico) e fornecer essas informações para outras máquinas.

1. Introdução

Com o contínuo crescimento da automação fabril, segundo [4], a utilização de robôs em linhas de produção industriais se tornou não só comum, mas necessária para a produção de bens de consumo nos dias de hoje. Assim, há um crescente interesse em fazer a intercomunicação entre as máquinas do ambiente de produção para melhor gestão da linha de produção e melhor interação humano-robô.

De acordo com [7], os manipuladores robóticos são amplamente utilizados em fábricas com as mais diversas atividades como pintura, soldagem, transporte e montagem. No entanto, para que seja possível a realização de tais atividades, com precisão e segurança, é preciso que haja uma boa interface de comunicação que permita o controle do robô manipulador, a integração com outros robôs e a observação de seus estados para possível identificação de falhas no sistema como um todo.

O objetivo desse projeto de pesquisa é desenvolver uma interface de comunicação que permita o usuário ter acesso às variáveis de juntas e enviar comandos de controle para o robô manipulador UR3 (Figura 1). Essa interface está sendo proposta devido a arquitetura do robô UR3 ser fechada para coleta e envio de dados por um computador externo, o que impossibilita a identificação do sistema dinâmico do UR3 e a implementação de novos controladores para o robô.

Partindo dessa premissa, foi desenvolvida uma interface de baixo nível que permite a realização de envio e coleta de dados para o robô UR3. A solução escolhida é baseada em um *framework* chamado *ROS - Robot Operating System* usando a linguagem de programação C++ [3]. *ROS* é uma coleção de *frameworks* de software para desenvolvimento de aplicações em robótica que fornece a funcionalidade de um sistema operacional em um *cluster* de computadores heterogêneo. Mais detalhes sobre ROS pode ser encontrado em [9].

2. UR3

O UR3 (Figura 1), alvo principal desse projeto de pesquisa, é um robô colaborativo de mesa que possui carga útil de 3 kg e configuração esférica. Seu tamanho reduzido o torna adequado para situações de espaço de trabalho limitado. Com seu giro infinito na junta final, diversas atividades podem ser realizadas com garras fixadas no conector da ferramenta do robô, o que o torna muito hábil.

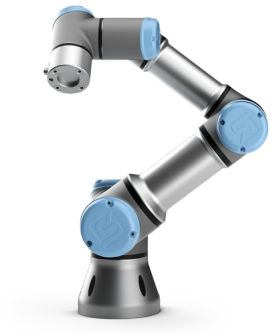

Figura 1. UR3

2.1. Aplicações

O braço robótico manipulador UR3 possuis diversas aplicações, entre elas estão:

- Trabalho de laboratório
- Tarefas de montagem
- Polimento
- Soldagem
- Colagem
- Aparafusar
- Pintura
- Pegar e Soltar

2.2. Especificação

A Tabela 1, extraída de [5], mostra as especificações gerais do manipulador robótico UR3. Geralmente, essas especificações são utilizadas para definir qual tipo de tarefa será realizada pelo robô e quais são os pré-requisitos para o seu bom funcionamento.

9

Tabela 1. ESPECIFICAÇÕES GERAIS DO UR3

Configuração do manipulador	Esférica
Masa	11,2 kg
Carga útil	3 kg
Alcance	500 mm
Pegada	Ø 128 mm
Graus de liberdade	6 (juntas rotativas)
Intervalos da articulação do pulso	+/- infinito
Intervalos das demais articulações	+/- 360°
Aceleração da articulação do pulso	360 graus / s
Aceleração das demais articulações	180 graus / s

2.3. Espaço de Trabalho

O espaço de trabalho de um manipulador, é o volume de espaço que o efetor final do manipulador pode alcançar. O tamanho e a forma da área de trabalho dependem da geometria coordenada do braço do robô e também do número de graus de liberdade. O UR3 possui 6 graus de liberdade e, por ser um robô articulado, tem uma configuração esférica.

A Figura 2, extraída de [5] página 35, mostra o espaço de trabalho do robô UR3, onde o raio da esfera mede 450 mm e o diâmetro do cilindro mede 200 mm. Assim, o espaço de trabalho total do robô manipulador UR3 é o volume da esfera menos o volume do cilindro.

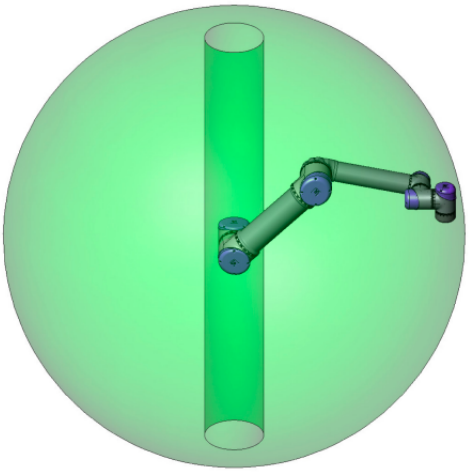

Figura 2. Espaço de trabalho do robô manipulador UR3

2.4. Sistema de coordenadas

A Figura 3, extraída de [5] página 165, mostra o sistema de coordenadas da base e do efetuador terminal no robô manipulador UR3. Notando que na literatura há várias possibilidades diferentes de orientação para as coordenadas nas juntas, usando o sistema de coordenada da base, foram levantados experimentalmente e marcados no próprio robô a indicação do sistema de coordenada de cada junta do robô tomando como referência o algoritmo de DENAVIT-HARTENBERG de [8].

A indicação do sistema de coordenada de cada junta do robô UR3 está indicada na ilustração da Figura 4. Tal ilustração foi feita usando o *software* **CoppeliaSim** [6].

2.5. Diagrama de comunicação

O diagrama de comunicação na Figura 5, mostra o fluxo de dados que transita entre um computador com sistema operacional linux (Linux PC) e o robô manipulador UR3. Nesse diagrama, vemos, também, a presença de um intermediário que faz a conexão entre Linux PC e o UR3 via cabo Ethernet, chamado *Control Box*, que nada mais é a

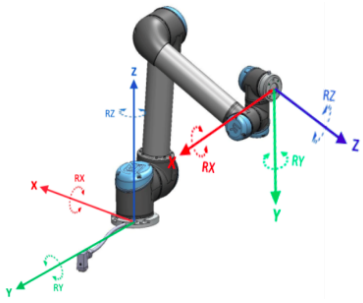

Figura 3. Ilustração da orientação da base do robô UR3

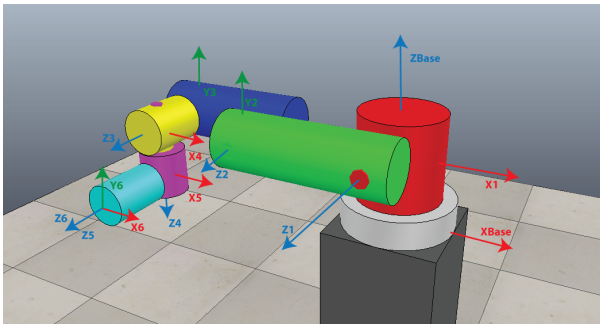

Figura 4. Ilustração da orientação de cada articulação do robô UR3

parte que compete ao *software* do robô UR3 implementada pela fabricante.

A interface de comunicação implementada nesse projeto de pesquisa, está presente no Linux PC e faz todo gerenciamento de dados durante o funcionamento do sistema como um todo. Além disso, a interface mostra, em tempo real, as variáveis de junta do robô UR3 em gráficos dinâmicos.

Para mais detalhe da descrição do sistema contido na Figura 5, acesse https://cooprobo.readthedocs.io/en/latest/manipulators/ur3.html.

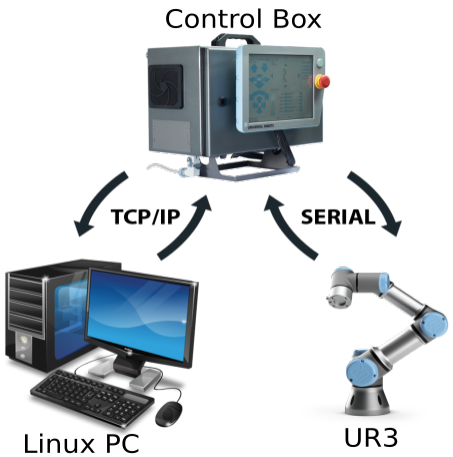

Figura 5. Diagrama de comunicação do robô

3. Metodologia

A metodologia se deu, primeiramente, de estudo para adaptação dos robôs disponíveis para trabalho cooperativo e compreensão de técnicas básicas de melhoria do ambiente de funcionamento dos robôs presente no Laboratório de Automação e Robótica (LARA). Buscando, assim, as melhores estratégias para dotar os robôs de funcionalidades como localização e mapeamento do ambiente, detecção de pessoas, manipulação de objetos e cooperação segura com outros agentes.

Em seguida, entrou-se na etapa de investigação e desenvolvimento. A partir da definição do estado da arte de [2], foram determinadas as melhores estratégias de estudo e implementação de técnicas de comunicação entre robôs. Inicialmente a comunicação foi feita entre o robô UR3 e o computador do usuário e posteriormente, foram incorporados os robôs móveis.

Na etapa final, como é mencionado no plano de trabalho, seria feito o desenvolvimento e aplicação de outras técnicas em estimação e controle de sistemas multi-agentes, manipulação cooperativa e interação homem-robô. Porém, devido a pandemia da COVID-19, se tornou inviável a presença de estudantes e pesquisadores no LARA, assim, o andamento planejado no plano de trabalho original foi comprometido.

3.1. Ferramentas utilizadas

- Framework ROS
- Computador com Linux
- Robô manipulador UR3
- C++
- Cabo Ethernet
- Roteador de Internet

3.2. Testes realizados

No decorrer do desenvolvimento da interface de comunicação, foram feitos testes para averiguar a consistência dos pacotes de dados que estavam sendo transmitidos entre o computador e a *Control Box*. Para fazer essa averiguação, coletou-se algumas amostras de dados referente às variáveis de juntas do robô UR3, afim de, visualmente, verificar se tinha alguma mudança abrupta nos sinas coletados pela interface.

Durante essa fase do projeto, verificou-se que estava ocorrendo perda de pacotes de dados pela interface de comunicação. Esse problema estava sendo causado pela falta de sincronia entre o Linux PC e a *Control Box* durante a execução do ROS no Linux PC. Sabendo desse detalhe, a falta de sincronia, ajustou-se a frequência de execução do ROS para uma frequência similar a frequência de execução do código do robô na *Control Box*, resolvendo esse problema facilmente.

4. Resultados

A implementação da interface de comunicação feita em C++ [3], aliada as funcionalidades do *framework* ROS, se desdobrou em um pacote que pode ser usado por qualquer robô manipulador da série UR3. Além disso a forma final do projeto, tem a capacidade de se comunicar com qualquer máquina que contenha o ROS instalado em seu sistema operacional. Além disso, a interface implementada neste projeto, possui uma extensão para coletar dados e controlar uma a garra robótica (Fgura 6) RG2 da OnRobot.

Figura 6. Garra robótica RG2 da OnRobot

Mais detalhe da implementação do código da interface de comunicação pode ser encontrado no *site* https://github.com/lara-unb/UR3_interface.

4.1. Interface

A interface de comunicação é via **Terminal** do Linux (vide Figura 8) e comandos de comunicação entre o robô e o terminal foram implementados em ROS. Assim, para fazer o carregamento da interface de comunicação e fazer seu uso, é necessário inserir comando típicos do ROS no **Terminal** do Linux. A figura 8, também mostra, um exemplo de como fazer o carregamento da interface de comunicação usando o comando *rosrun ur3 interface*, esse comando carrega todos os módulos necessários para que a comunicação entre o Linux PC e o robô UR3 ocorra da forma correta.

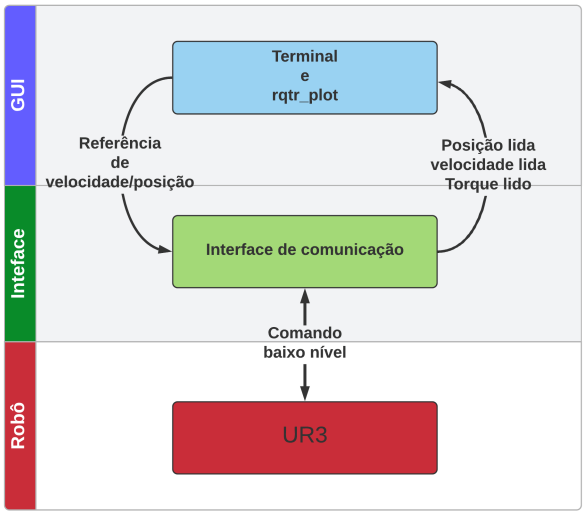

Figura 7. Diagrama de fluxo de dados usando a interface

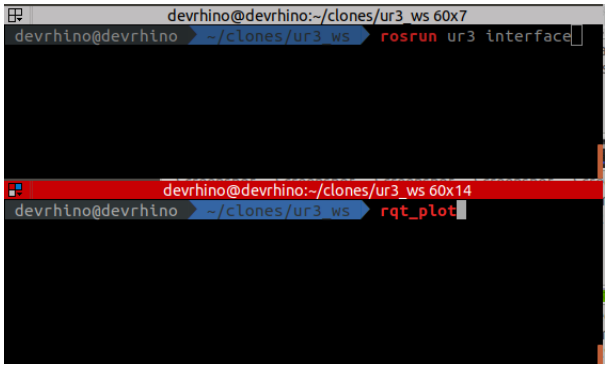

Figura 8. Carregamento da interface usando o Terminal do Linux

Uma forma de visualizar os dados para que possa ser feita uma checagem visual dos estados das juntas do robô, é fazer o uso do *plugin* **rqt_plot** [1]. Com esse *plugin*, que pode ser carregado usando o **Terminal** do Linux (como a interface), é possível fazer uso de uma GUI (*Graphical User Interface*) que permite visualizar valores numéricos em um gráfico 2D.

Na Figura 9, é mostrado um exemplo de como fica a visualização de dados pela a interface em tempo real com o *plugin* **rqt_plot**. Os dados mostrados na figura são, respectivamente, a referência de posição (em azul) e a posição lida (em vermelho) em função do tempo. Como característica, o **rqt_plot** é uma ferramenta de visualização de dados bem robusta, visto que ela conta com mecanismos de inserção de novos dados, zoom, filtragem, e mecanismo de salvar os dados de experimento. Além disso, mesmo que a interface de comunicação tenha uma implementação em baixo nível (ROS), isso não se torna um problema quando fazemos o uso de ferramentas presentes no ROS como o **rqt_plot**.

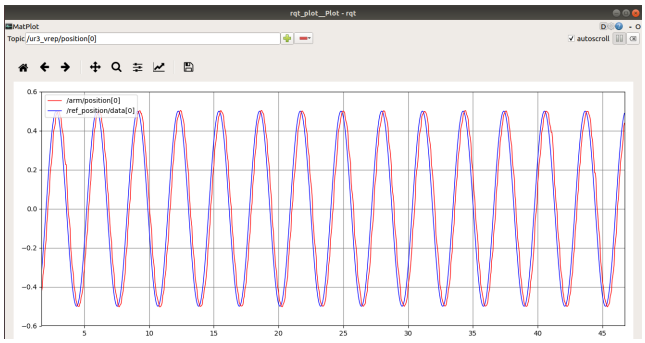

Figura 9. Carregamento da interface usando o Terminal do Linux

4.2. Dados coletáveis para análise

Os dados foram coletados usando o ROS através do pacote **rosbag**. Esse pacote pode armazenar os dados que estão sendo coletados pelo ROS em um metadado para ser usado posteriormente. Os dados mais relevantes que a interface de comunicação extrai do robô UR3 são: posição, velocidade e torque das juntas.

As Figuras 10, 11 e 12 representam, para cada junta, respectivamente, posição, velocidade e torque do robô UR3. Nas Figuras 10 e 11, podemos perceber que, nos sinais de posição e velocidade, a presença de ruído não se faz presente de maneira significativa.

Os sinais de Torques, presente na figura 12, apresentam uma parcela de ruído significativo. Esse ruído é devido ao sinal de torque ser extraído da corrente aplicada nos motores das juntas do robô e como a dinâmica de um sinal elétrico é bem mais rápido que a dinâmica de um sinal mecânico, podemos ver a presença de ruido no sinal de torque.

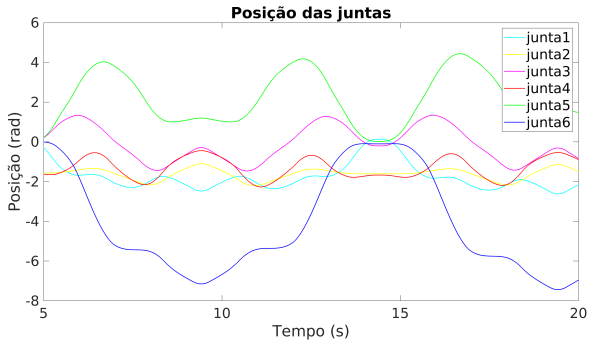

Figura 10. Posições das juntas do robô UR3 fornecidas na interface e plotadas usando Matlab

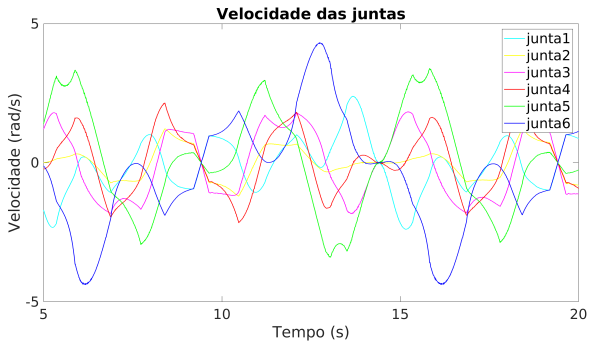

Figura 11. Velocidades das juntas do robô UR3 fornecidas na interface e plotadas usando Matlab

Os dados coletáveis pela interface desenvolvida viabilizam análise, projeto e implementação de controladores que devem ser feitos no futuro por outros alunos.

5. Conclusão

Ao fim do projeto de pesquisa, foi concluído, com sucesso, a implementação de uma interface para o robô manipulador UR3. Essa interface de comunicação permite, através de um cabo Ethernet, a comunicação entre diversas máquinas/robôs presentes no mesmo ambiente e que tem a possibilidade de ser usada em locais de linhas de montagem de fábricas manufatureiras. A extensão implementada para

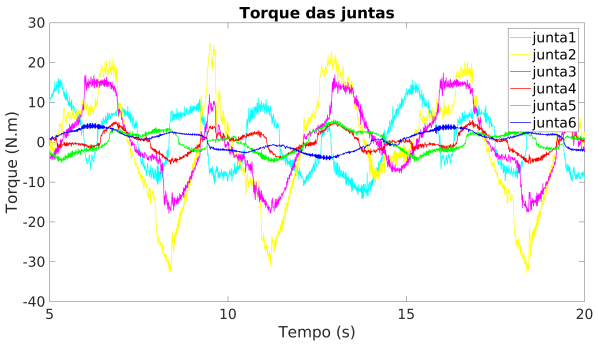

Figura 12. Torques das juntas do robô UR3 fornecidas na interface e plotadas usando Matlab

fornecer suporte a garra robótica RG2 também foi concluída com êxito, permitindo, assim, o braço robótico executar tarefas como pegar e largar objetos pequenos durante a execução de uma tarefa.

Devido as limitações causadas pela pandemia da COVID-19, não foi possível montar e testar a proposta de desenvolver técnicas em estimação e controle de sistemas multi-agentes, manipulação cooperativa e interação homem-robô. A pesquisa será continuada e os próximos passos serão destinados a descrição e identificação do sistema do robô UR3. Esses passos consistem em determinar os parâmetros de DENAVIT-HARTENBERG com base em [8], implementar controladores cinemáticos e dinâmicos, identificar o sistema dinâmicos do robô manipulador UR3 por métodos lineares e realizar a integração em uma plataforma de sistemas multi-agentes.

Referências

[1] Dorian Scholz, Dirk Thomas. rqt_ plot.

[2] Felix Messmer, Kelsey Hawkins, Shaun Edwards, Stuart Glaser, Wim Meeussen. Robotic operating system.

[3] ISO. *ISO/IEC 14882:2011 Information technology — Programming languages — C++*. International Organization for Standardization, Geneva, Switzerland, February 2012.

[4] Aline S. Aires Heráclito L. J. Pontes Dmontier P. A. Junior Marcos R. Albertin, Maria Luiza B. E. Elienesio. Principais inovações tecnológicas da indústria 4.0 e suas aplicações e implicações na manufatura. In *XXIV SIMPEB*, pages 3–5, Bauru, SP, Brasil, 2017.

[5] UNIVERSAL ROBOTS. *User Manual UR3/CB3*. UNIVERSAL ROBOTS.

[6] E. Rohmer, S. P. N. Singh, and M. Freese. Coppeliasim (formerly v-rep): a versatile and scalable robot simulation framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*, 2013.

[7] João Maurício Rosário. *Robótica Industrial I: Modelagem, Utilização e Programação*, volume 1. Baraúna, 2010.

[8] M.W. Spong, S. Hutchinson, and M. Vidyasagar. *Robot Modeling and Control*. 2005.

[9] Stanford Artificial Intelligence Laboratory et al. Robotic operating system.