USO DE IMAGENS PARA O POSICIONAMENTO DE MANIPULADORES ROBÓTICOS

Marcio P. Costa[1]

[1]Faculdade de Tecnologia SENAI-CIMATEC E-mail: marciopc@gmail.com

USING IMAGES TO POSITION ROBOTIC MANIPULATOR

Resumo: *Identificação visual do mundo ao nosso redor, transformada em informação utilizável por manipuladores robóticos e dispositivos autônomos, facilita e reduz os custos de construção desses dispositivos, substituindo o uso de sensores específicos e complexos por câmeras de uso geral. Este artigo apresenta um estudo do uso de uma câmera de vídeo genérica para visualização de peças dispostas sobre uma mesa em ambiente controlado. Essas informações são passadas para um manipulador robótico de forma a permitir que o mesmo se movimente até a peça e consiga pegá-la.*

Palavras-Chaves: *Visão computacional; Movimento por Imagem; Identificação de contornos; Comunicação com manipuladores robóticos;*

Abstract: *Visual identification of the world around us, transformed into information usable by robotic manipulators and autonomous devices, facilitates and reduces the construction costs of these devices, replacing the use of specific and complex sensors by general-purpose video cameras. This paper presents a study of the use of a generic video camera to view objects arranged on a table in a controlled environment. This information is passed to a robotic manipulator so that it can move toward it and can pick it up*.

Keywords: *Computer vision; Movement by Image; Contour identification; Communication with robotic manipulators;*

1. INTRODUÇÃO

A automação industrial e a robótica mundial surgiram utilizando conceitos de sensores e atuadores que permitiam obter informações sobre o ambiente a ser automatizado e atuar de acordo com o desejado. Sensores de nível, temperatura, pressão, contato, entre outros, fornecem informações que são processadas e transformadas em sinal para atuadores que permitem controlar essas e outras variáveis relacionadas no ambiente automatizado.

Algumas variáveis são de difícil medição, ou complexas demais para se obter o nível de informação suficiente para que possa ser utilizada para o controle do processo. Como seria um sensor para identificar onde estão vários objetos dispostos sobre uma mesa, de forma a permitir que um manipulador robótico possa movimentar-se até eles, girar na posição correta para cada um, e pegá-los?

Segundo a Data Science Academy[1], visão computacional é uma disciplina que estuda como reconstruir, interromper e compreender uma cena 3d a partir de suas imagens 2d em termos das propriedades da estrutura presente na cena. A identificação dessas propriedades se dá através do uso de ferramentas de tratamento de imagens. A evolução da capacidade computacional dos últimos anos tem permitido obter-se cada vez mais detalhes e informações das imagens tratadas, a velocidades cada vez maiores. O que não era possível fazer com sensores complexos e específicos, hoje é possível fazer com câmeras de vídeo genéricas e baratas, aliadas a algum poder computacional para lidar com as imagens obtidas.

2. FUNDAMENTAÇÃO TEÓRICA

Mecatrônica é uma área do estudo científico que exige a agregação de conhecimentos em diversas disciplinas. Segundo Adamowski e Furukawa[5], “Mecatrônica pode ser definida como integração da Mecânica, Eletrônica e Computação de forma concorrente”. Na verdade, projetos de robótica podem exigir conhecimentos ainda mais específicos, como química, soldagem, tornearia ou outras habilidades que se deseje implementar no manipulador robótico. Nesse estudo em específico, foram utilizados conhecimentos da área de redes de computadores, desenvolvimento em linguagem JAVA, manipulação robótica utilizando linguagem RAPID®, e noções de visão computacional.

2.1. Redes de computadores

Apesar de, segundo Kurose e Ross[2], o termo redes de computadores estar começando a soar um tanto desatualizado, devido à diversidade de componentes que na verdade estão sendo interligados, ele refere-se à interconexão entre duas ou mais máquinas no intuito de trocar informações e compartilhar recursos. Essa comunicação ocorre através da utilização de

protocolos de rede, que definem regras para a troca de mensagens. Segundo Tanenbaum[7], “Basicamente, um protocolo é um acordo entre as partes que se comunicam, estabelecendo como se dará a comunicação”. Existem diversos protocolos de redes que são utilizados atualmente, porém, desde o surgimento e explosão da Internet, o protocolo TCP/IP vem se tornando o padrão mundial de fato, e está a cada dia penetrando mais nos ambientes de redes industriais. Protocolos industriais como Ethernet/IP, Profinet, Controlnet, Devicenet são exemplos de pilhas de protocolo industrial que utilizam o TCP/IP na sua base.

A comunicação entre computadores via TCP/IP ocorre através de pares de endereços TCP(Transmission Control Protocol) e IP(Internet Protocol). O Endereço IP permite que o dado trafegue entre as máquinas, inclusive por trajetos intercontinentais. O Endereço TCP, chamado de Porta, permite que o dado seja direcionado para a aplicação correta no servidor de destino. O processo que ocorre é o seguinte:

1 – Máquina 1 executa uma aplicação A que ‘escuta’ uma determinada Porta por conexões

2 – Máquina 2, executa uma aplicação cliente que conecta nessa Porta.

Essa comunicação ocorre através de um elemento da programação Cliente-Servidor no protocolo TCP/IP que é chamado de Socket. Voltando ao exemplo anterior, a Máquina 1 abriu um Socket Server, que fica aguardando a conexão. A Máquina 2 abriu um Socket Client direcionado para a Máquina 1, na Porta da aplicação A. Assim, os sockets podem trocar mensagens, com os devidos tratamentos pelas aplicações cliente e servidora.

2.2. Manipulação robótica – RAPID

RAPID é uma linguagem de programação de alto nível usada para controlar Robôs Industriais da ABB. É uma linguagem nativa de diversos controladores ABB e possui estruturas de controle de código, loops, condicionais, operadores lógicos e aritméticos, chamadas de sub-rotinas, error handlers e etc, além das funções específicas de manipulação robótica em si, como comandos de movimento, de obtenção de posicionamento e sinalização de entrada e saída do controlador.

Como foi especificamente desenhada para manipulação de robôs, possui inúmeros recursos desenvolvidos para essa finalidade. “As características básicas do movimento de robôs são determinadas pelos dados especificados em cada instrução de posicionamento”[4].

Este estudo utilizou a linguagem RAPID para movimentar o manipulador robótico da ABB, em ambiente simulado RobotStudio 6.02. Nesse ambiente foi criado um programa que utiliza um Socket Client para conectar em uma aplicação remota que irá fornecer informação de localização para onde o manipulador deverá ser movido.

2.3. Linguagem Java – Framework OpenCV

A linguagem de programação Java, bem como seus diversos ambientes de desenvolvimento (IDEs) são amplamente conhecidos e utilizados, não somente em ambientes acadêmicos, mas também no comercio, na indústria e em todas as áreas para onde se pode desenvolver sistemas. Também é uma linguagem de programação de alto nível, porém se destaca de algumas outras por ser interpretada. Ou seja, os sistemas feitos em Java não são compilados para o código de máquina, individualizado para cada modelo específico de hardware e sistema operacional. No caso dos programas em Java, eles precisam de uma Máquina Virtual Java, que interpreta o programa desenvolvido.

Esse tipo de linguagem de desenvolvimento não é tão incomum, mas não era bem visto em outros tempos, pois o processo de interpretar o código para executá-lo torna os programas muito mais lentos que os compilados. Porém, independentemente dessa perda de performance, o Java se tornou um padrão de desenvolvimento mundial já a algum tempo. A SUN, e posteriormente a ORACLE, foram desenvolvendo máquinas virtuais Java (JVMs, JREs e JDKs) para os mais diversos ambientes e sistemas, desde máquinas gigantes com sistemas operacionais complexos, como o VMS, ou os diversos tipos de UNIX, como também para celulares e dispositivos embarcados. A maior parte dos dispositivos embarcados da atual internet das coisas, utiliza máquinas virtuais Java para rodar seus códigos.

Neste estudo em específico, foi utilizado o Java para realizar algumas atividades muito importantes no contexto. O programa Java é responsável por abrir o Socket Server e ficar aguardando conexão, enquanto captura imagens provenientes de uma câmera de uso geral, conectada ao computador, além de permitir interação com o usuário. Ou seja, o sistema Java desenvolvido para esse estudo também se utiliza do conceito de aplicativo multi-thread, ou seja, ele divide o processamento em alguns threads, para permitir que as atividades possam ser executadas simultaneamente.

Para captura e manipulação das imagens, foi utilizado o framework OpenCV 3.1 [3], que oferece muitas possibilidades no tratamento de imagens. Foram utilizadas funções básicas de desenho de círculos e elipses, e algumas funções mais complexas de detecção de contornos, atenuação de bordas entre outras.

2.4. Visão computacional

A visão computacional é uma das áreas da informática que tem tido uma evolução muito significativa nos últimos anos. É uma área de conhecimento que utiliza muito poder computacional para obter resultados importantes, e o aumento da capacidade de processamento dos computadores atuais tem permitido execução, em tempo real, de funções complexas, que antes levariam muito tempo para serem executadas.

É mais uma área da computação que pretende aumentar a percepção e compreensão de máquinas e equipamentos a respeito do mundo à sua volta. À medida que os estudos e teorias nessa área vão progredindo, são criados modelos cada vez mais completos dos objetos reais, que permitem melhores tomadas de decisão, por parte das máquinas, em situações e cenas reais.

O desafio imposto a este estudo, na área de visão computacional, é um dos problemas mais complexos nessa área, que é a detecção de bordas e identificação de objetos. Segundo Rios [6], “Um dos problemas mais importantes no reconhecimento de objetos em imagens é a detecção de bordas. Basicamente, consiste em detectar regiões da imagem nas quais ocorre uma mudança abrupta de brilho na imagem, que, geralmente, representam uma mudança das características do que está sendo visto”.

No estudo em questão, os aspectos utilizados foram bastante facilitados, uma vez que foi montado um ambiente controlado, sem interferências externas, e as peças a serem identificadas também eram peças de geometria simples, o que facilitou a identificação de seus contornos e posicionamentos. Para transferência de informação, houve necessidade de transformar os dados obtidos a partir do ponto de visão da câmera para o ponto de acesso do manipulador robótico. Foram levados em consideração as condições de escala, e a rotação dos eixos. Essas duas transformações foram dadas como suficientes para o nível de precisão do movimento desejado, deixando interferências de perspectiva fora do estudo.

3. METODOLOGIA

Ferramentas atuais de virtualização e simulação facilitam bastante a realização dos estudos e experimentos em diversas áreas da ciência e da indústria. Nesse estudo foram criadas duas máquinas virtuais para representarem respectivamente o manipulador robótico e o visualizador/tratador da imagem obtida pela câmera.

Na Máquina Virtual 01 (VM01), foi instalado o sistema operacional Microsoft Windows 7 32 bits, com a suíte ABB RobotStudio 6.03.02, onde foi escolhido o modelo IRB-260, que tem uma boa amplitude de movimentos (1,50m) e permitiria melhor visualização dos deslocamentos entre os diversos pontos do experimento. Foi criada uma Solução com Estação e Controladora, incluindo a opção de PC Communication, para funcionamento da comunicação via rede.

Na Máquina Virtual 02 (VM02), também foi instalado o sistema operacional Microsoft Windows 7 32 bits, a suíte de desenvolvimento eclipse IDE for Java Developers, release Neon, e o framework de manipulação de imagens OpenCV 3.1. Nessa máquina foi instalada a webcam que será utilizada para identificação do posicionamento dos objetos.

O princípio aplicado para a comunicação foi que, uma vez que precisasse alcançar uma nova peça, o manipulador robótico (VM01) solicitaria à máquina com a câmera (VM02) que indicasse as informações da mesma, e obteria suas

coordenadas e o ângulo em que se encontra para posicionamento da ferramenta. O primeiro passo executado no processo é a criação de um Socket Server na VM02, que tratará toda a comunicação entre as máquinas. Em seguida, já com a janela da aplicação aberta, o programa fica aguardando a conexão da VM01.

A VM01 por sua vez abre um Socket Client e solicita informações do próximo objeto disponível. Porém, o primeiro contato é utilizado pela VM02 para solicitar informações sobre os limites reais dos extremos da sua imagem, em valores conhecidos pelo manipulador. Com essa informação, é possível estabelecer uma escala para calcular o valor de cada eixo que deve ser enviado para o manipulador (Figura 1), levando em consideração a resolução da câmera e as coordenadas limites definidas pelo manipulador.

Uma vez sendo possível transladar essas coordenadas, o algoritmo da VM02 segue o processo, com a detecção dos contornos existentes na imagem, para identificação de suas coordenadas. Essas coordenadas identificadas e transladadas ficam então disponíveis para a VM01 na sua próxima solicitação.

Para que as máquinas se entendam, foi necessário criar uma lista de palavras que seriam trocadas e compreendidas de lado a lado. O protocolo criado está descrito nas tabelas 1 e 2.

O algoritmo trata a imagem obtida com algumas transformações que permitem melhor detecção dos objetos. A primeira transformação é a remoção das cores da imagem, transformando-a em escala de cinza, e reduzindo os possíveis impactos delas na detecção de contornos. Em seguida é aplicado um filtro para atenuar bordas, com o intuito de evitar detecção de falsos contornos na imagem. Após transformada, é a função *findContours*(Figura 2), que percorre a imagem e gera um array de contornos identificados.

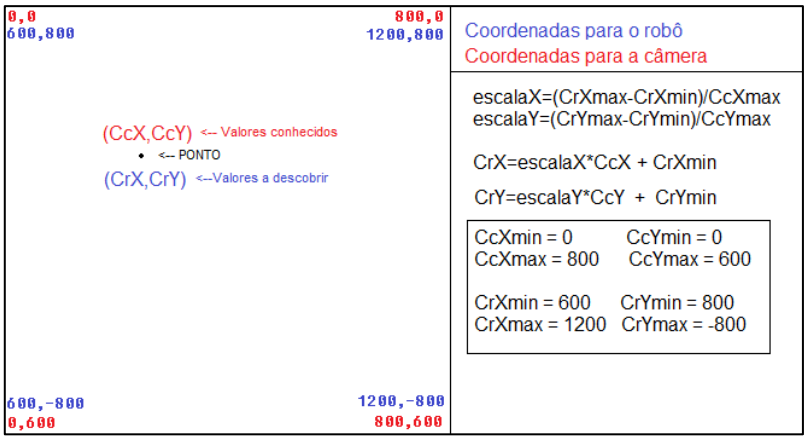

Figura 1. Escala e translação dos pontos.

Tabela 1. Vocábulos para comunicação

VM01	VM02
“PROX”	“LIMITS”
“FECHA”	“FECHA”
“[0-9].*; [0-9].*; [0-9].*; [0-9].*;”	“-1”
	“[0-9].*; [0-9].*; [0-9].*; [0-9].*;”

Tabela 2. Cenários de troca de mensagens

Primeiro contato	VM01(“PROX”) -> VM02(LIMITS) -> VM01(“[0-9].*; [0-9].*; [0-9].*; [0-9].*;”)
Próximos contatos	VM01(“PROX”) -> VM02(“[0-9].*; [0-9].*; [0-9].*; [0-9].*;”)
Acabaram as peças	VM01(“PROX”) -> VM02(“-1”)
Finalizado pela VM02	VM01(“PROX”) -> VM02(“FECHA”)
Finalizado pela VM01	VM01(“FECHA”) -> VM02(“FECHA”)

```
Imgproc.threshold(frame_gray, threshold_output, thresh, 255, Imgproc.THRESH_BINARY );
Imgproc.findContours(threshold_output, contours, hierarchy, Imgproc.RETR_TREE, Imgproc.CHAIN_APPROX_SIMPLE, new Point(0, 0) );
```

Figura 2. Atenuação e detecção de bordas

Para uma melhor visualização, o programa circula com elipses os objetos identificados, colocando um pequeno círculo no centro apenas do objeto que será informado ao robô na sua próxima solicitação. Quando a VM01 recebe esses valores, ela os utiliza para mover o robô, utilizando as coordenadas x,y,z passadas.

4. RESULTADOS E DISCUSSÃO

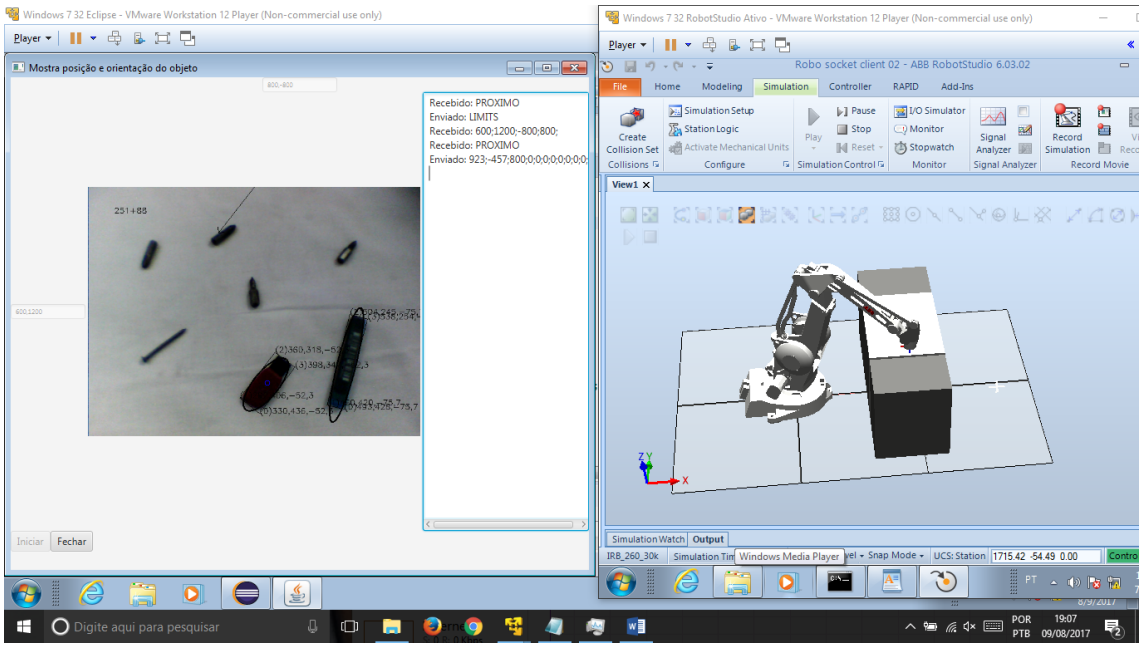

Figura 3. Tela do experimento em andamento.

Apesar do nível de dificuldade para implementação do projeto, do volume de conhecimentos necessários e da multidisciplinaridade envolvida, este estudo consegue obter bons resultados para o ambiente controlado proposto. É certo que o estudo realizado se limitou a apresentar a viabilidade da solução, desenvolvendo uma interface simples, um protocolo extremamente básico e complexidade de movimentos restritos ao 2D. Mas essa simplicidade foi suficiente para que fosse possível perceber que a solução é viável e tem baixo

custo. Todo o processo foi realizado com máquinas virtuais, câmera de uso geral com baixa qualidade de imagem e simulador robótico.

Percebe-se pelos resultados apresentados que tanto os contornos identificados, como a disposição das peças, além da orientação da peça 'da vez', indicada na seta superior (Figura 3), são informações suficientes para que o manipulador robótico obtenha sucesso na tarefa de pegar o objeto.

5. ESPECTATIVAS FUTURAS

Existe possibilidade de se implementar a mesma solução em ambientes mais complexos, com câmeras de melhor qualidade, interface que permita uma maior interatividade com o sistema, tanto do lado da câmera quanto do lado do robô, protocolo mais elaborado, permitindo melhor troca de informações e em maior variedade de tipos trafegados. Esses avanços certamente levariam à criação de uma solução real viável para uso na indústria em diversas situações onde as imagens poderiam auxiliar robôs a tomarem suas decisões, alcançarem suas peças, identificarem problemas e melhorarem processos.

Segue como sugestão para um estudo posterior, a implementação física do ambiente, em situação real, com utilização de manipuladores e ferramentas reais, para identificação de necessidades específicas e aprimoramento da capacidade da solução, inclusive permitindo por exemplo que a informação possa ser enviada a mais de um manipulador que a solicite.

6. REFERÊNCIAS

[1] Equipe Data Science Academy – Disponível em <http://datascienceacademy.com.br/blog/o-que-e-visao-computacional/>.

[2] Kurose, James F., Ross, Keith W. - Redes de computadores e a Internet: uma abordagem top-down, vol. 5, cap. 1, pag. 3

[3] Open Source Computer Vision Library - Disponível em <http://opencv.org/>

[4] RAPID reference manual – RAPID overview online – ABB Flexible Automation - Disponível em <http://rab.ict.pwr.wroc.pl/irb1400/overviewrev1.pdf>

[5] Adamowski, Julio C. et al., Furukawa, Celso M. et al - Mecatrônica: Uma Abordagem Voltada à Automação Industrial - disponível em

[6] Rios, Luiz R. S., Visão computacional - DCC UFBA - Disponível em <http://homes.dcc.ufba.br/~luizromario/Apresenta%C3%A7%C3%A3o%20de%20IA/Artigo%20(final).pdf>

[7] Tanenbaum, Andrew S., J. Wetherall, David - Redes de Computadores, Vol. 4, cap. 1, pag. 37 - Pearson Education