Departamento de Engenharia Mecânica
Faculdade de Ciências e Tecnologia da Universidade de Coimbra

INTEGRAÇÃO DE SISTEMAS DE VISÃO EM CÉLULAS ROBÓTICAS

Nuno Alberto Marques Mendes

DISSERTAÇÃO PARA OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA MECÂNICA

Fevereiro de 2009

Departamento de Engenharia Mecânica
Faculdade de Ciências e Tecnologia da Universidade de Coimbra

INTEGRAÇÃO DE SISTEMAS DE VISÃO EM CÉLULAS ROBÓTICAS

Nuno Alberto Marques Mendes

Dissertação para obtenção do Grau de Mestre em

Engenharia Mecânica

Júri:

Presidente: Professora Doutora Marta Oliveira
Orientador: Professor Doutor J. Norberto Pires
Vogal: Professor Doutor Altino Loureiro

Fevereiro de 2009

AGRADECIMENTOS

Os meus agradecimentos vão para todos os que tornaram este trabalho possível, e que de uma forma ou de outra contribuíram para o resultado que podemos encontrar nestas páginas. É com enorme sentimento de gratidão que destaco:

- O Professor Doutor J. Norberto Pires, orientador deste trabalho, pelas facilidades concedidas na utilização do laboratório de controlo e robótica e do apoio prestado ao longo de todo este trabalho.
- O Engenheiro Pedro Neto por toda a disponibilidade e paciência demonstrada, e todo o conhecimento que me transmitiu.
- Os Engenheiros Germano Veiga, Ricardo Araújo e o Professor Doutor Francisco Caramelo que em momentos de pouca clarividência me transmitiram dicas cruciais.
- Finalmente, aos meus familiares e amigos por toda a motivação, apoio, paciência, compreensão e disponibilidade.

A mente que se abre a uma nova ideia
jamais voltará ao seu tamanho original…
(Albert Einstein)

RESUMO

A presente dissertação aborda o controlo de robôs[1] industriais baseado em sistemas de visão. O principal objectivo é o desenvolvimento e implantação de um Sistema de Visão que funcione como elemento sensor ao controlo de um robô industrial. O Sistema de Visão detecta características de objectos, nos quais são realizadas operações de manipulação "*pick-and-place*" e reprodução de contornos.

Sobre o tema é apresentado o estado da arte e ainda as ferramentas de Visão por Computador necessárias à sua implantação.

O trabalho desenvolvido é um trabalho que abrange diversas áreas. Algumas das quais se passam a enumerar: distorção de imagem, processamento de imagem, precisão de robôs industriais, geração automática de código para robôs, entre outras. Qualquer uma destas áreas é consideravelmente vasta e como tal têm sido, e podem ser, alvo de estudos e dissertações aprofundadas. Com o intuito de não tornar a presente dissertação demasiado extensa, os diversos assuntos abordados serão tratados de uma forma tão resumida quanto possível incidindo no essencial sobre os aspectos relevantes para o presente trabalho. Também a aplicação desenvolvida é considerada bastante extensa, contendo inúmeras linhas de código, pelo que nesta dissertação os aspectos de programação serão apresentados muito superficialmente, sendo dada principal relevância à abordagem adoptada para solucionar as diversas dificuldades ou objectivos que se pretendiam colmatar. No entanto, será abordado extensivamente a interacção do utilizador com o *software* desenvolvido.

Serão também apresentados alguns dos resultados obtidos em testes executados durante o desenvolvimento do sistema.

Será por fim apresentado uma análise dos resultados obtidos, bem como de sugestões da aplicabilidade do Sistema de Visão em ambiente industrial e de trabalhos futuros a realizar nesta área.

[1] Ao longo do presente texto é usado por diversas vezes a palavra robô. No contexto deste trabalho pretende-se com esta palavra fazer referência a robôs manipuladores, sendo qualquer outro caso devidamente mencionado.

A implantação deste Sistema de Visão em testes realizados, revelou um excelente desempenho quando os objectos se encontram no centro da imagem. No entanto, quando estes se afastam do centro da imagem e possuem altura considerável (não podendo ser considerados como objectos bidimensionais) os resultados pioram ligeiramente todavia não colocam em causa a realização das operações.

Palavras-chave: Visão por Computador, Representação de Contornos, Manipulação de Objectos, Robôs Manipuladores, Robótica Industrial.

ABSTRACT

This thesis addresses the control of industrial robots[2] based on Vision Systems. The main objective is the development and deployment of a system that works as a Vision sensing element to the control of an industrial robot. The Vision System detects features of objects, in which operations are conducted (manipulation "pick-and-place" and reproduction of contours). The current state of the art and some Computer-Vision tools necessary for the deployment of the system are presented. Several areas are covered, including image distortion, image processing, accuracy of industrial robots, generation automatic code for robots, among others. Any of these areas are considerably larger, so that have been subject of depth studies and dissertations.

In order to not make this essay too long, the various issues raised will be treated in a way as short as possible, addressing relevant information about the developed work. The developed application is considered quite large, containing many lines of code, so the programming aspects will be presented very superficially, given relevance to the approach used to solve the problems and reach the targets. However, the interaction between the user and the developed software will be extensively discussed.

To attest the viability of the developed robotic cell, several tests are made. Finally, the results obtained in tests are analyzed, possible industrial applications analyzed, and future discussed.

The implemented system presented good results, even in the presence of non-controlled sources of light and shadows. Nevertheless, the method to calibrate the camera should be improved.

Keywords: Computer Vision, Reproduction of Contours, Pick-and-place, Manipulators Robots, Industrial Robotic.

[2] The word robot is many times referred in the thesis. In the context of this work, this word refers to robotic manipulators. Any other case will be duly mentioned.

ÍNDICE

LISTA DE TABELAS, FIGURAS E ABREVIATURAS

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE ABERVIATURAS

Capítulo 3

d Distância entre o pixel a analisar e outro pixel da sua vizinhança;

g Valor de brilho quantificado para um pixel;

n Número de pontos em píxeis pertencentes ao objecto;

t Valor do brilho que estabelece o que é objecto na imagem e o que não é;

x Coordenada do pixel na vizinhança do pixel a analisar, na direcção horizontal;

x_c Coordenada do pixel a analisar na direcção horizontal;

X_g Coordenada na direcção horizontal do centro de massa do objecto presente na imagem em pixéis;

X_i Coordenada na direcção horizontal do pixel *i* do objecto presente na imagem em pixéis;

y Coordenada do pixel na vizinhança do pixel a analisar, na direcção vertical;

y_c Coordenada do pixel a analisar na direcção vertical;

Y_g Coordenada na direcção vertical do centro de massa do objecto presente na imagem em pixéis;

Y_i Coordenada na direcção vertical do pixel *i* do objecto presente na imagem em pixéis;

σ Factor de suavização.

Capítulo 4

$C_{q,x}$ Constante de conversão do quadrante *q* segundo a direcção horizontal *x*;

$C_{q,y}$ Constante de conversão do quadrante *q* segundo a direcção vertical *y*;

$K_{q,x}^{[mm]}$ Média dos pontos recolhidos no quadrante *q* e na direcção horizontal, segundo o referencial cena em milímetros;

$K_{q,x}^{[pix]}$ Média dos pontos recolhidos no quadrante *q* e na direcção horizontal, segundo o referencial imagem em píxeis;

$K_{q,y}^{[mm]}$ Média dos pontos recolhidos no quadrante *q* e na direcção vertical, segundo o referencial cena em milímetros;

$K_{q,y}^{[pix]}$ Média dos pontos recolhidos no quadrante *q* e na direcção vertical, segundo o referencial imagem em píxeis;

$x_i^{[mm]}$ Coordenada do ponto *i* segundo a direcção horizontal, no referencial cena em milímetros;

$x_j^{[pix]}$ Coordenada do ponto *i* segundo a direcção horizontal, no referencial imagem em píxeis;

$y_i^{[mm]}$ Coordenada do ponto *i* segundo a direcção vertical, no referencial cena em milímetros;

$y_j^{[pix]}$ Coordenada do ponto *i* segundo a direcção vertical, no referencial imagem em píxeis;

cc Ponto principal;

fc Distância focal;

kc Coeficiente de distorção;

n Número de pontos recolhidos;

Rx Orientação do robô para um ponto na direcção Rx;

Ry Orientação do robô para um ponto na direcção Ry;

Rz Orientação do robô para um ponto na direcção Rz;

x coordenada de um ponto na imagem no referencial imagem em pixéis segundo a direcção horizontal;

X Coordenada de um ponto no referencial do robô na direcção x;

x_c coordenada de um ponto da imagem no referencial cena em milímetros segundo a direcção horizontal;

y coordenada de um ponto na imagem no referencial imagem em píxeis segundo a direcção vertical;

Y Coordenada de um ponto no referencial do robô na direcção y;

y_c coordenada de um ponto da imagem no referencial cena em milímetros segundo a direcção vertical;

Z Coordenada de um ponto no referencial do robô na direcção z;

α Coeficiente de distorção entre eixos;

Capítulo 5

d Distância entre o ponto de referência e um ponto na sua periferia;

$x_{estab.}$ Coordenada de um ponto no referencial imagem, em píxeis, segundo a direcção horizontal, que foi estabelecido como pertencente aos pontos da matriz Ordenação de Pontos;

x_i Coordenada de um ponto no referencial imagem, em píxeis, segundo a direcção horizontal, pertencente aos pontos da matriz Obtenção de Pontos;

$y_{estab.}$ Coordenada de um ponto no referencial imagem, em píxeis, segundo a direcção vertical, que foi estabelecido como pertencente aos pontos da matriz Ordenação de Pontos;

y_i Coordenada de um ponto no referencial imagem, em píxeis, segundo a direcção vertical, pertencente aos pontos da matriz Obtenção de Pontos;

CAPÍTULO UM

1 Introdução

Hoje em dia, a automação é indispensável no sector industrial. Os benefícios de uma máquina fazer o trabalho, para o qual seriam necessárias várias pessoas para o realizar, mais rapidamente, com menos erros e com uma maior disponibilidade traduz-se, muitas vezes, num bom investimento a longo prazo para uma empresa (ganhos de produtividade e qualidade). No entanto, atendendo ao elevado investimento inicial, dever-se-á realizar um estudo económico acerca deste, a fim de verificar a viabilidade do investimento.

Actualmente assiste-se a um mercado global onde a concorrência vem de "todos os cantos" do planeta, sendo um dos principais objectivos das empresas a diminuição do ciclo de produção dum produto tendo em vista a redução do seu preço final, mantendo sempre níveis de qualidade elevados que devem ser melhorados de dia para dia. Por outro lado, o *ciclo de vida* dum produto é muito baixo, o que leva a que um produto que se produz hoje em grandes quantidades amanhã já não seja produzido. Este facto faz com que hoje em dia a *automação flexível* se tenha vindo a sobrepor à *automação rígida* (máquinas dedicadas a fazer uma determinada operação num determinado produto). De realçar a *robótica de manipulação* que é, em muitos casos, um elemento de extrema importância na *automação flexível*.

O objectivo deste trabalho é o desenvolvimento e concretização de um sistema robótico, onde se utiliza um Sistema de Visão como elemento sensor ao controlo de um robô industrial. Pretende-se que o Sistema de Visão detecte objectos (áreas, contornos, cores e posições) para a realização de dois tipos de operações distintas. Manipulação de objectos "*pick-and-place*" ou reprodução de contornos dos mesmos.

1.1 Organização da Tese

Este trabalho está dividido em sete capítulos, os quais se passam a explicitar:

Para além do já exposto, o capítulo um apresenta ainda a história da robótica, os tipos de robôs industriais existentes e os seus principais parâmetros.

O capítulo dois apresenta um levantamento sobre o estado da arte acerca de aplicações onde se utilizam sistemas de visão no "controlo" de robôs, assim como os vários tipos de sistemas de visão existentes e as diversas configurações dos sistemas de visão em células robóticas.

O capítulo três apresenta o aparato experimental a realizar e os acessórios de visão utilizados para o fim proposto. Demonstra também os processos de extracção de informação de uma imagem e métodos matemáticos utilizados na elaboração de *software* de aquisição e processamento de imagem.

O capítulo quatro aborda a calibração de imagem tendo em vista a redução de distorção da imagem e a imprecisão do robô.

O capítulo cinco apresenta as ferramentas utilizadas para comunicar com os dispositivos utilizados, demonstra a necessidade de reduzir o número de pontos resultantes da extracção de informação da imagem e o método utilizado. Este capítulo também analisa a cinemática do ponto de vista das trajectórias a realizar pelo robô.

O capítulo seis apresenta o *software* desenvolvido e explica, de forma breve, como o utilizador o deve utilizar.

Finalmente, o capítulo sete resume contribuições propostas na tese e apresenta reflexões. Apresenta ainda possíveis aplicações na indústria do sistema desenvolvido e possíveis optimizações a realizar no futuro.

1.2 História da robótica industrial

George Devol recebeu as primeiras patentes sobre robótica em 1954. A primeira companhia a produzir um robô industrial foi a Unimation, fundada por George Devol e Joseph F. Engelberger em 1956, sendo baseada nas patentes originais de Devol. Os robôs da Unimation também eram chamados de "máquinas de transferência programadas", visto que a sua principal função era a transferência de objectos de um ponto para outro, a figura 1.1 ilustra este robô. Eles utilizavam actuadores hidráulicos e eram programados com "conjuntos de coordenadas", podendo-se considerar como exemplo um robô em que os ângulos de todas as juntas são armazenados durante uma fase de aprendizagem e, então repetidos durante a operação normal.

Por muito tempo o único concorrente da Unimation foi a Cincinnati Milacron. No entanto durante os anos 70 esta situação alterou-se radicalmente, quando um grande número de conglomerados japoneses começou a produzir robôs industriais similares. A Unimation tinha obtido patentes nos Estados Unidos, porém não as obteve no Japão, que se recusou a seguir as leis de patentes internacionais, de modo que os projectos foram copiados.

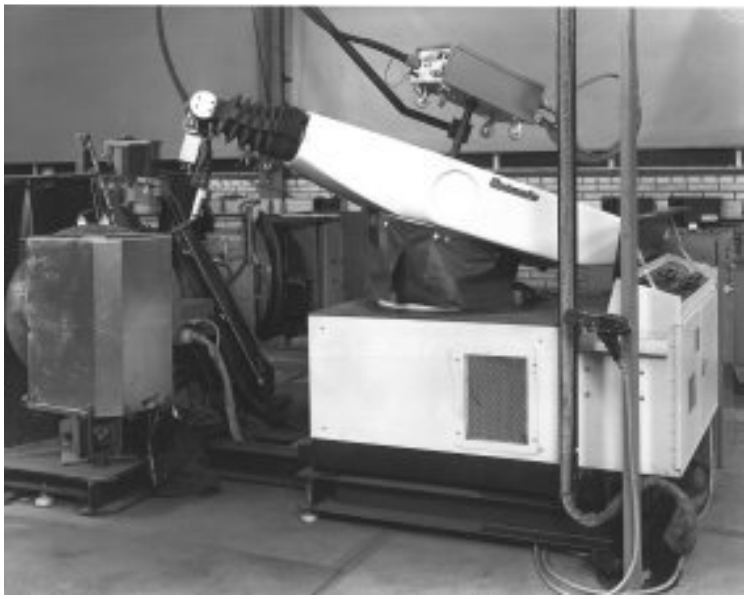


Figura 1.1: Robô UNIMATION

Em 1969 Victor Scheinman inventou o braço de Stanford na Universidade de Stanford, um robô articulado de 6 eixos, totalmente eléctrico, projectado de modo a permitir uma solução utilizando a anatomia de um braço. Isto permitiu que o robô fosse capaz de seguir com um elevado grau de precisão trajectórias arbitrárias no espaço, aumentando as possibilidades de utilização de robôs em aplicações mais sofisticadas tais como montagem de componentes e soldadura. Scheinman em seguida projectou um segundo braço para o MIT AI Lab, chamado de "braço do MIT". Sheinman vendeu os seus projectos para a Unimation, a qual os desenvolveu com o auxílio da General Motors e posteriormente o comercializou como a Máquina Programável Universal para Montagem (PUMA), a figura 1.2 ilustra este tipo de robô.

Em 1973, a KUKA construiu o seu primeiro robô industrial, conhecido com FAMULUS, sendo este o primeiro robô industrial articulado a possuir seis eixos controlados electronicamente.

O interesse na robótica industrial aumentou no final dos anos 70 e muitas companhias entraram no campo, incluindo grandes empresas como a General Eletric e a General Motors (que formaram o empreendimento FANUC Robotics juntamente com a FANUC do Japão).

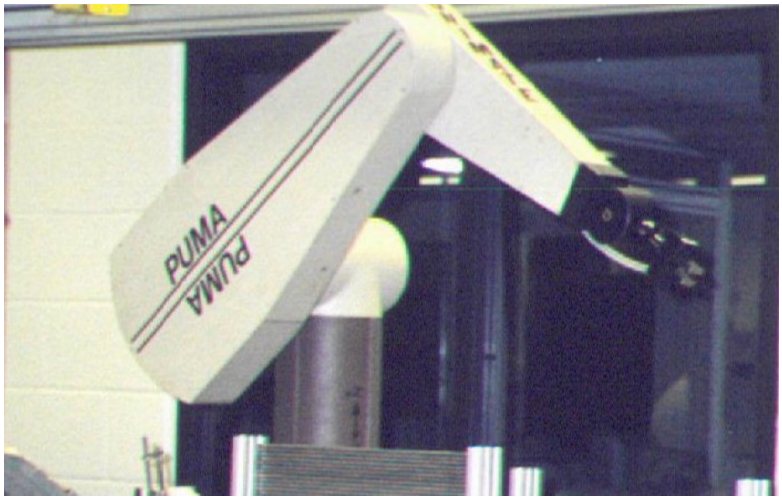


Figura 1.2: Robô PUMA

No momento mais intenso do crescimento da robótica em 1984, a Unimation foi comprada pela Westinghouse, que mais tarde, em 1988, acabou por a vender a Stäubli Faverges SCA, de França. A Stäubli ainda fabrica robôs articulados para a indústria em geral, inclusive em 2004 comprou a divisão de robótica da Bosch.

Eventualmente a visão limitada da indústria americana foi substituída pelos recursos financeiros e, grande parte do mercado interno usufruído pelas indústrias japonesas. Apenas um pequeno número de companhias não-japonesas foram capazes de se manter nesta área, incluindo a Adept Technology, a Stäubli-Unimation, a companhia Sueca-Suiça ABB (ASEA Brown-Boveri), a companhia COMAU (pertencente ao Grupo Fiat), a construtora Austríaca igm Robotersysteme AG e a companhia Alemã KUKA Robotics.

1.3 Tipos de robôs industriais

As configurações de robôs utilizadas mais comummente na automação industrial incluem os robôs articulados (o tipo mais comum), os robôs SCARA, e os robôs cartesianos (também conhecidos como robôs x-y-z) [1, 2], ver figura 1.3. "*Mas o que é isto de robô industrial?*" De acordo com a *Robotic Industries Association*, robô industrial é um "*manipulador*

multifuncional, reprogramável, projectado para movimentar materiais, ferramentas ou peças, através de movimentos programados". Uma definição mais técnica é apresentada pela norma ISO 10218, como sendo "*uma máquina manipuladora, com vários graus de liberdade, controlada automaticamente, reprogramável, multi-funcional, com base fixa ou móvel para utilização em aplicação industrial*".

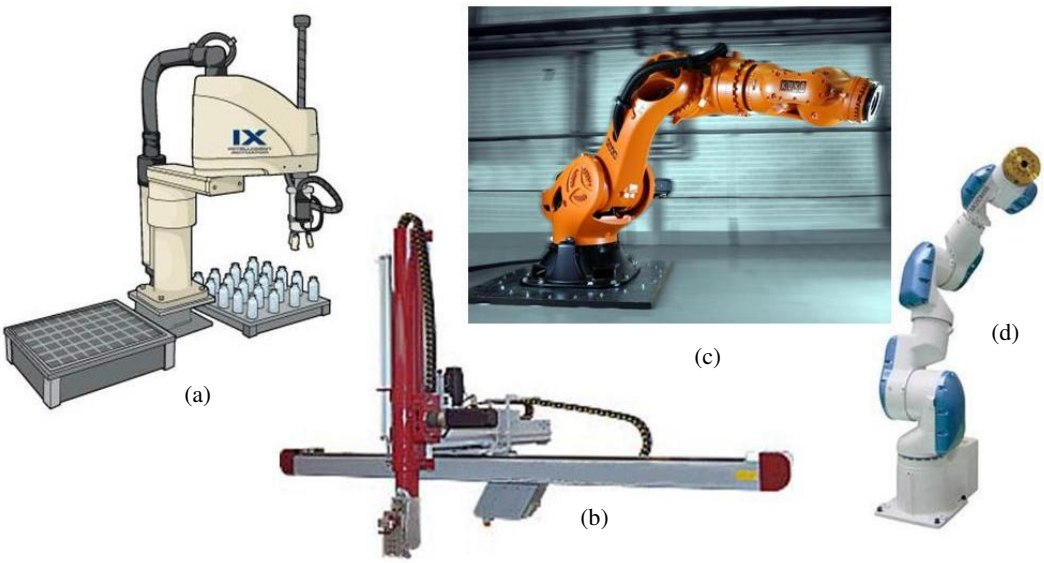


Figura 1.3: (a) Robô tipo SCARA; (b) Robô Cartesiano; (c) Robô Articulado de 6 eixos; (d) Robô Articulado de 7 eixos.

No contexto da robótica, a maior parte dos robôs industriais seria categorizada como braços robóticos (inerente no uso da palavra "manipulador" mencionada na definição da ISO). Os robôs industriais possuem diferentes níveis de autonomia. Alguns são programados para realizarem acções repetidamente sem nenhuma variação, com um nível elevado de precisão. Estas acções são determinadas por rotinas pré-programadas que especificam a direcção, aceleração, velocidade e distância de uma série de movimentos coordenados. Outros são mais flexíveis em relação à orientação do objecto em que trabalham ou com o trabalho que realizam sobre o objecto, o qual pode eventualmente ser identificado pelo robô.

Geralmente os robôs são programados para operar em ambientes bem conhecidos. No entanto, hoje em dia procura-se que eles sejam cada vez mais "inteligentes", i.e. sejam capazes de tomar alguns tipos de decisões.

Com vista a colocar este aspecto em prática, actualmente é frequente os robôs utilizarem vários tipos de sensores que ajudem no reconhecimento e interpretação do ambiente de trabalho em que se inserem, tais como sensores de força para reconhecer o contacto com o

ambiente de trabalho, sensores de ultra-sons e laser para medir distâncias e câmaras para visualizar o ambiente de trabalho na sua globalidade e a partir da imagem extrair a mais diversa informação. A inteligência artificial, e as suas variações, possuem uma importância crescente nos robôs industriais modernos.

Figura 1.4: Robô industrial MOTOMAN HP6.

Um conceito importante é o volume de trabalho do robô, ou seja, o conjunto de todos os pontos que podem ser alcançados pela extremidade do robô, durante a sua movimentação. Assim, os elementos que serão manipulados e/ou processados pelo robô devem ser arranjados para ficarem dentro desse volume de trabalho.

No presente trabalho utiliza-se o robô MOTOMAN HP6 com o controlador NX100, o qual se ilustra na figura 1.4, na tabela 1.1 apresenta-se as características deste robô.

O volume de trabalho do robô MOTOMAN HP6 é apresenta na figura 1.3.

Tabela 1.1: Especificações do Robô

MOTOMAN HP6	
Capacidade de carga	6 [Kg]
Eixos controlados	6
Repetibilidade	+/- 0.08 [mm]
Massa	130 [Kg]
Controlador	NX100

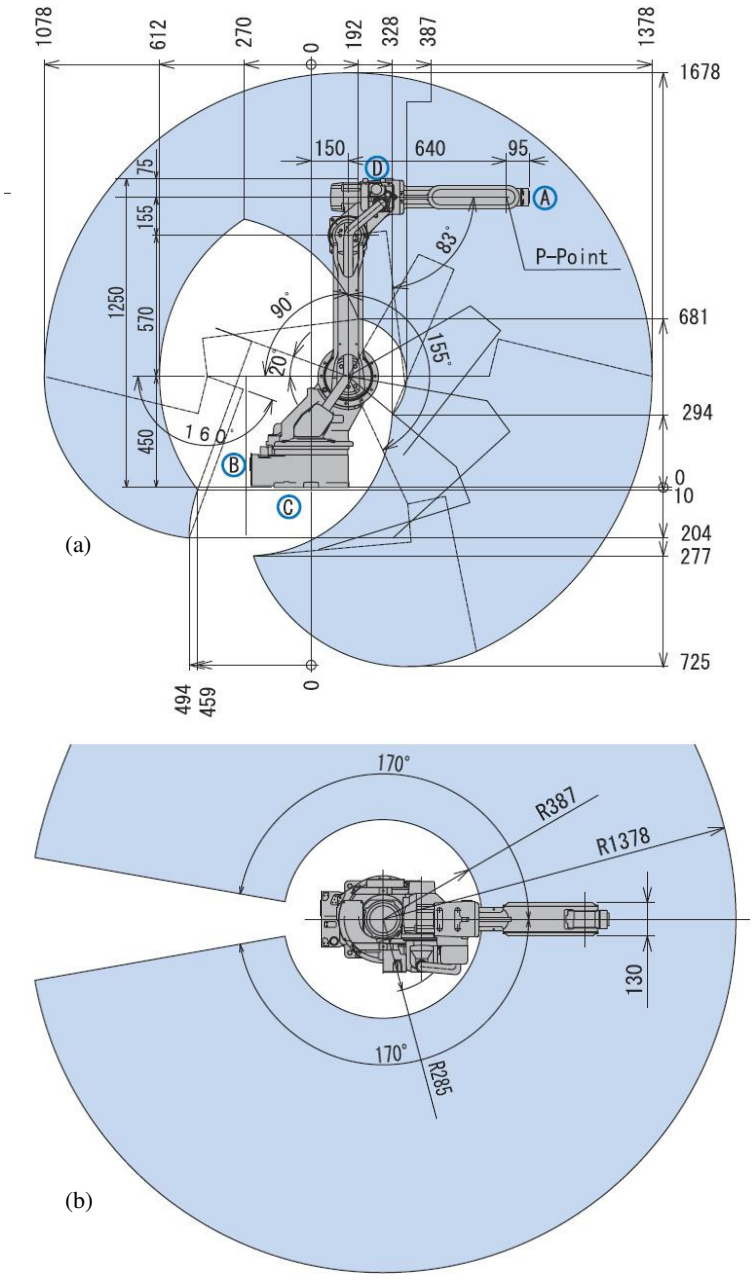


Figura 1.5: Volume de trabalho do robô MOTOMAN HP6: (a) No plano XZ; (b) No plano XY.

1.4 Descrição técnica e Definição de parâmetros de um Robô

- ✓ Número de eixos – número de graus de liberdade de um robô. Dois eixos são necessários para se alcançar qualquer ponto no plano, três eixos são necessários para se alcançar qualquer ponto no espaço. Para controlar completamente a orientação do extremo de um braço rotativo, outros três eixos são necessários.

- ✓ Cinemática – disposição dos membros e juntas de um robô, os quais determinam os possíveis movimentos do mesmo. As categorias cinemáticas dos robôs incluem articulados, cartesianos, paralelos e SCARA.
- ✓ Volume de trabalho – região do espaço que um robô pode alcançar.
- ✓ Capacidade de carga – quantidade de peso que um robô pode levantar.
- ✓ Velocidade – rapidez com que um robô pode posicionar o extremo do seu braço.
- ✓ Precisão – o quão próximo da posição desejada o robô pode alcançar. A precisão pode variar com a velocidade e a posição no ambiente de trabalho. Ela pode ser aumentada através da calibração.
- ✓ Repetibilidade – é a medida de quão perto se pode posicionar um mecanismo de um ponto previamente armazenado.
- ✓ Controle dos movimentos – para algumas aplicações, tais como montagens repetitivas, o robô precisa apenas de executar repetidamente um número limitado de posições pré-programadas. Para aplicações mais sofisticadas, tais como a soldadura, o movimento deve ser continuamente controlado para que se siga um caminho no espaço, com a velocidade e orientação controladas.
- ✓ Fonte de energia – alguns robôs utilizam motores eléctricos, enquanto outros utilizam actuadores hidráulicos. O primeiro é mais rápido, enquanto o segundo é mais forte.
- ✓ Acoplamento – alguns robôs conectam os motores eléctricos às juntas através de caixas de redução, outro conectam os motores directamente às juntas (acoplamento directo).

Figura 1.6: Robôs industriais a realizar operações de montagem num chassi de um veículo.

CAPÍTULO DOIS

2 Estado da Arte

Nos últimos tempos tem-se vindo a assistir, no meio industrial, a uma crescente aposta nos sistemas de visão com a finalidade de contribuir para o controlo de robôs manipuladores, denominado na literatura como "*visual servoing*", contolo visual de robôs. Tal facto deve-se a este tipo de sensor permitir retirar muita informação sobre o ambiente circundante ao robô e de possuir uma elevada precisão, quando devidamente calibrado. Um outro aspecto que em muito contribuiu para a crescente ascensão deste tipo de sensor foi o aumento da capacidade de cálculo dos computadores. O processamento de imagem carece de algoritmos que são bastante complexos em quantidade de operações de cálculo, o que se traduz em tempo de cálculo. Veja-se que no final da década de setenta, data do primeiro sistema de controlo visual, o processamento de imagem era na ordem das dezenas de segundo em comparação com os sistemas actuais, onde se pode encontrar sistemas de processamento de apenas 1 [ms], pelo que se pode dizer que os sistemas actuais são cada vez mais em "tempo-real".

Há quem chegue a afirmar que a utilização de visão tornará os robôs autónomos [3], "*A utilização de câmaras poderá, num futuro que esperamos próximo, tornar os robôs completamente autónomos para operar em meios desconhecidos ou dificilmente modeláveis.*".

Nas actuais aplicações de robôs na indústria, estes movimentam-se em ambientes estruturados, i.e. em ambientes de trabalho especialmente projectados para a realização das várias tarefas para que são programados, tarefas específicas. Porém, o mundo em que vivemos e em que o robô se encontra inserido tem características dinâmicas, quer através do seu próprio movimento, quer através do movimento de objectos no ambiente de trabalho do robô, ou ainda de outros agentes aí existentes a realizar as suas tarefas [4], i.e. ambientes não estruturados. É considerando este novo facto que é imperioso, a fim de o robô poder interagir de uma forma harmoniosa com o meio que o rodeia, muni-lo com sensores capazes de

adquirir mais informação que os clássicos *encoders* e *tacómetros*, colocados nas juntas dos robôs. Um dos sensores que melhor reúne as características necessárias ao adequado desempenho em ambientes não-estruturados é a visão. Tal escolha é justificada pelos muitos sistemas biológicos que a utilizam para reunir informação sobre o ambiente que os rodeia, pelo incremento das suas características e crescente disponibilidade no mercado a cada vez mais baixo custo [4], podendo ainda ser facilmente acoplados a computadores. Assim, através do sensor de visão, **câmara**, é possível extrair do ambiente de trabalho a informação visual necessária à realização de tarefas por parte de robôs.

Em [5] e [6] são apresentados dois exemplos de aplicação dos sistemas de visão no controlo parcial de robôs. No primeiro são investigadas estratégias para o controle de robôs manipuladores, usando realimentação por visão, na realização da tarefa de aproximação e manipulação de um objecto. No segundo é utilizado um sistema de visão para controlar um robô na industria alimentar, mais propriamente para fazer a triagem e manuseamento de empadas de carne.

Os sistemas de visão podem também ser utilizados na detecção e interpretação de gestos manuais, permitindo a interacção entre o utilizador e o robô [7].

2.1 Tipos de Sistemas de Visão

Os sistemas de visão podem ser subdivididos em dois tipos de abordagens, nomeadamente sistemas de visão activa e sistemas de visão passiva.

2.1.1 Sistemas de Visão Activa

Sistemas de visão activa referem-se ao caso onde é utilizado uma câmara e um projector. O projector é utilizado para projectar uma luz padrão no objecto enquanto a câmara adquire uma imagem do objecto com a projecção padrão. Um exemplo onde este sistema de visão é utilizado encontra-se descrito em [8], a figura 2.1 pretende ilustrar um esquema deste tipo de sistema de visão. Este tipo de sistema é amplamente utilizado no reconhecimento de objectos 3D, possibilitando obter informação sobre a superfície 3D de um objecto conhecendo o padrão de luz projectada sobre a superfície e analisando a forma da luz padrão na imagem adquirida. Um aspecto negativo desta abordagem é a necessidade de hardware adicional, o

que não é desejável. Em muitas aplicações reais, é importante reduzir os custos de produção e inspecção. Pelo que a introdução de hardware adicional aumentará o custo.

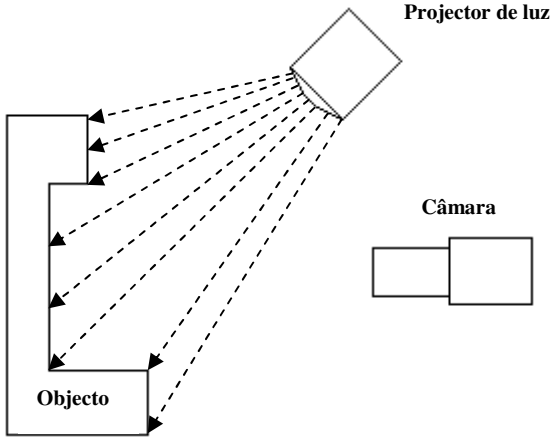


Figura 2.1: Esquema representativo de um Sistema de Visão Activa

Os sistemas de visão laser são tipicamente um exemplo de sistemas de visão activa. Uma técnica muito utilizada nestes sistemas de visão laser é a projecção de uma linha laser sobre um objecto que, vista numa câmara (normalmente CCD "*charge coupled device*" ou CMOS "*complementary metal oxide semiconductor*"), apresenta uma alteração da sua forma e posição, que estão directamente relacionadas com a forma do objecto segundo essa linha. A informação volumétrica de um objecto pode então ser calculada. Este é um dos métodos mais importantes na obtenção de imagens tridimensionais (triangulação por laser), pela sua característica simples, de não contacto, de alta precisão e com rápida velocidade de medição.

Em [9] é apresentado exaustivamente um sistema comercial deste tipo. Designado na literatura por "*Servo-Robot*" este sistema é aplicado a operações de soldadura, permitindo controlar o robô de uma forma semi-automática. Dado a sua capacidade de extracção de informação/distâncias da cena observada pela câmara (câmara/tocha de soldadura/partes do objecto a soldar) e consoante os parâmetros fornecidos ao sistema, ele permite efectuar operações de aproximação ao início da soldadura, conduzir o robô durante a soldadura com base na preparação de junta, e inspeccionar a soldadura realizada com base na sua geometria.

Um outro exemplo de sistemas de visão activa é apresentado em [6], este trabalho consiste no desenvolvimento e construção de um sistema que retira informação tridimensional de objectos através de técnicas de triangulação de um laser e uma câmara. Informação essa utilizada no controlo de um robô manipulador para efectuar operações de manuseamento de objectos.

2.1.2 Sistemas de Visão Passiva

Sistemas de visão passiva referem-se ao caso onde uma ou mais câmaras são utilizadas para adquirir imagem de um objecto. A figura 2.2 pretende ilustrar um esquema deste tipo de sistema de visão. Este tipo de sistema difere dos sistemas de visão activa na possibilidade de obter todas as imagens necessárias do objecto utilizando apenas uma câmara [10]. Com este tipo de abordagem não se obtém a mesma quantidade de informação que se consegue obter com os sistemas de visão activa, pelo que a quantidade de hardware utilizada faz-se notar. A fim de atenuar essa diferença normalmente são desenvolvidos algoritmos bastante complexos.

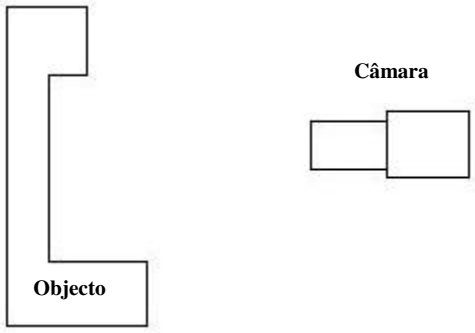


Figura 2.2: Esquema representativo de um Sistema de Visão Passivo

A utilização de sistemas de visão para controlar robôs manipuladores que operam em ambientes industriais, desconhecidos, ou dificilmente modeláveis requer uma elevada precisão, fiabilidade e facilidade de colocação em funcionamento.

A obtenção de elevada precisão nos sistemas de visão tem associada a qualidade da informação referente ao objecto no ambiente que o rodeia. Assim pode ser necessário utilizar, em vez de visão monocular, visão estéreo ou ainda aumentar o número de câmaras a observar o objecto. A utilização de visão em conjunto com outro tipo de sensores, como os já referidos ultra-sons, laser ou sensores de força, permite aumentar o conhecimento do meio em que o robô se movimenta e também aumentar a precisão nas medidas necessárias ao seu controlo.

A fiabilidade de um sistema de controlo visual está dependente da fiabilidade dos equipamentos mecânicos e eléctricos que o constituem, assim como a robustez na variação dos parâmetros de calibração e a convergência a partir de uma posição inicial.

A facilidade de colocação em funcionamento de um sistema de controlo visual está dependente da sua fase de calibração, i.e. esta fase deverá ser simplificada o mais possível e no caso ideal eliminada.

A extracção da informação a partir da imagem, com vista a caracterizar o espaço de trabalho do robô manipulador, pode ser feita de duas formas:

- Utilizando informação bidimensional expressa directamente nas coordenadas do plano da imagem, controlo visual 2D [11].
- Utilizando informação tridimensional em que modelos da câmara e objecto são utilizados para determinar a pose do objecto relativamente aos referenciais da câmara, do robô, ou do mundo, controlo visual 3D [12].
- Na literatura encontra-se uma terceira forma de caracterização, denominada por controlo visual híbrido [13]. Esta última não é mais que a utilização em simultâneo dos dois tipos de caracterização mencionada em cima, de forma a melhorar o desempenho global do sistema. A fim de controlar o robô manipulador com base nas características do objecto na imagem, é necessário estabelecer a relação entre estas e as coordenadas da câmara, relativamente ao referencial do mundo ou do elemento terminal do robô manipulador.

2.2 Configuração física dum sistema de visão

Quando se utiliza um sistema de visão para o controlo parcial de um robô manipulador, deve-se ter a noção que as características recolhidas a partir da imagem ou imagens dependem de vários factores. São eles o número de câmaras utilizadas, da sua configuração relativamente ao robô, da sua calibração e ainda do conhecimento prévio que se poderá obter do ambiente de trabalho onde se encontra o objecto.

A dependência existente em relação ao número de câmaras do sistema, permite desde logo definir **visão monocular** (uma câmara) e **visão estéreo** (duas câmaras ligadas rigidamente) e ainda **sistemas de câmaras redundantes**.

Em relação à configuração da(s) câmara(s) relativamente ao robô, existem duas opções possíveis. Uma, em que a(s) câmara(s) encontra(m)-se ligadas rigidamente ao elemento terminal do robô e a "olhar" para o objecto, *eye-in-hand*. E outra, em que a(s) câmaras(s) se encontra(m) no espaço a "olhar" para o elemento terminal do robô e para o objecto simultaneamente, *eye-to-hand*.

Estes dois tipos de configuração física dos sistemas de visão serão desenvolvidos nos próximos subcapítulos.

2.2.1 Visão Monocular

Um sistema de visão monocular utiliza somente uma câmara, que pode ser colocada no elemento terminal do robô a olhar o objecto ou ainda no espaço a olhar simultaneamente para o elemento terminal do robô e para o objecto. A principal vantagem deste tipo de sistema é o de minimizar o tempo de processamento necessário para extrair as informações visuais, sendo outra vantagem o seu custo. No entanto, tem a desvantagem de não ser possível determinar a profundidade, distância entre a câmara e o objecto de uma forma exacta, sendo necessária a sua estimação.

2.2.1.1 Câmara a olhar o objecto

O sistema de visão monocular em que a câmara se encontra colocada no elemento terminal do robô, *eye-in-hand*, e a olhar para o objecto, ver figura 2.3, tem a particularidade de ser o mais comum nas aplicações de controlo visual de robôs manipuladores.

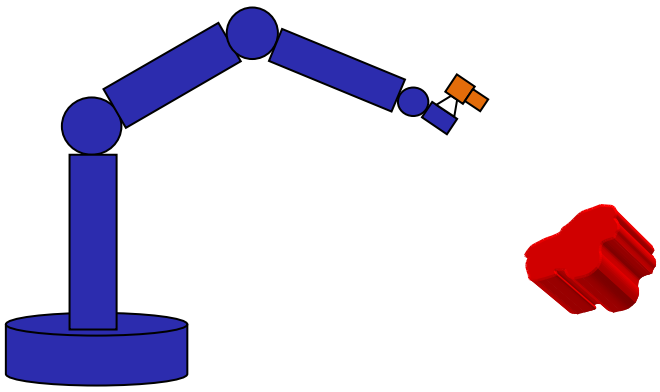


Figura 2.3: Visão Monocolar com a câmara no elemento terminal e a olhar para o objecto, *eye-in-hand*.

Para este tipo de configuração as aplicações típicas são as de seguir um determinado objecto ou de mover o robô entre duas posições pré-definidas na imagem, usualmente na mesma imagem. Como exemplo de posições definidas em imagens diferentes, temos o recente trabalho de Remazeilles [14], baseado numa base de dados de imagens (memória virtual). O

trabalho proposto em [10], ilustra um sistema monocular *eye-in-hand* para reconhecimento de objectos 2D e 3D. Neste estudo são adquiridas imagens do objecto e estas são comparadas com outras imagens armazenadas na base de dados.

2.2.1.2 Câmara a olhar o robô manipulador e o objecto

O tipo de sistema em que uma câmara se encontra a olhar o robô manipulador e o objecto, ver figura 2.4, *eye-to-hand*, foram os primeiros a surgir nos trabalhos sobre controlo visual de robôs manipuladores, [15]. Estes tipos de sistemas requerem calibração pois utilizam como variáveis a controlar a pose do objecto relativamente à câmara ou ao elemento terminal do robô. Neste tipo de sistemas é ainda necessário determinar a posição do elemento terminal do robô relativamente à câmara, o mesmo é dizer as coordenadas da câmara no referencial do mundo.

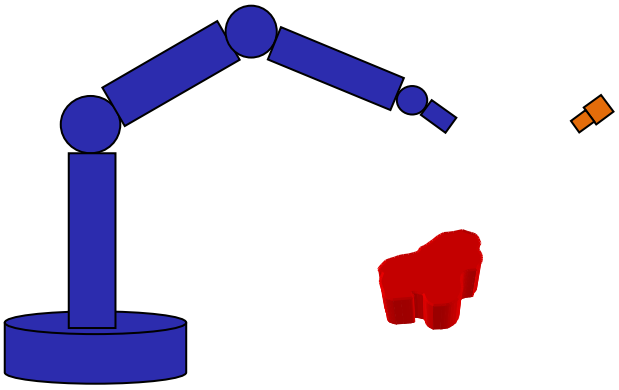


Figura 2.4: Visão Monocular com câmara a olhar para o objecto e para o robô manipulador, *eye-to-hand*.

2.2.2 Visão Estéreo

Um sistema de visão estéreo é constituído por duas câmaras rigidamente ligadas com a finalidade de obter duas imagens do mesmo objecto. Este sistema de visão é bastante útil quando se pretende obter informação tridimensional, por exemplo de um determinado objecto dentro do campo de visão do par estéreo. Relativamente ao sistema de visão monocular, a utilização de visão estéreo facilita a obtenção da profundidade embora penalizando o tempo de processamento das imagens e o custo do sistema. Em seguida são apresentadas as configurações mais usuais deste sistema de visão, em tudo idênticas à visão monocular, i.e.

com as câmaras a olhar para o objecto ou a olhar para o elemento terminal do robô e para o objecto simultaneamente.

2.2.2.1 Câmaras a olhar o objecto

A utilização de um par estéreo rigidamente ligado ao elemento terminal, ver figura 2.5, não é muito usual, devido ao facto de a este tipo de sistema estar associado um maior volume (relativamente à visão monocular), o que dificultará outro tipo de aplicações, tais como manipulação de objectos. Contudo, a miniaturização cada vez maior dos sistemas de visão tem vindo a esbater este obstáculo. A utilização deste tipo de sistema induz um outro problema que tem a ver com a perda de precisão dos algoritmos de reconstrução tridimensional, pois a distância entre os eixos ópticos das duas câmaras (*baseline*) [16], deste tipo de sistema devera ser pequena. Uma aplicação deste tipo é detalhado em [17].

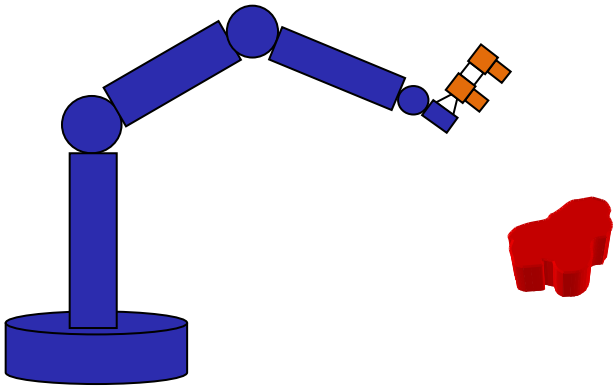


Figura 2.5: Visão Estéreo com as câmaras colocadas no elemento terminal e a olhar para o objecto.

2.2.2.2 Câmaras a olhar o robô manipulador e o objecto

Quando as câmaras do par estéreo são colocadas num local pré-definido no espaço, ver figura 2.6, *eye-to-hand*, os sistemas de visão estéreo são mais utilizados no controlo de robôs manipuladores. Tal acontecimento deve-se ao facto de, neste caso não existirem restrições à *baseline* do par estéreo que quando colocado no elemento terminal a condicionava. Consegue-se assim o comprimento necessário na *baseline* do par estéreo para que os resultados da reconstrução 3D sejam precisos.

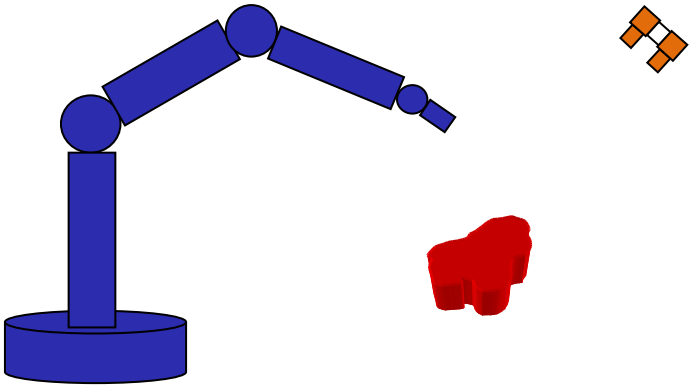


Figura 2.6: Visão Estéreo com as câmaras a olhar para o objecto e para o robô manipulador.

2.2.3 Sistemas com Câmaras Redundantes

Quando são utilizadas mais que duas câmaras, os sistemas definem-se como redundantes, pois a partir de um par estéreo (duas câmaras) é possível reconstruir a informação tridimensional de um objecto que se encontra no seu campo de visão. A utilização deste tipo de sistemas permite obter informação adicional sobre o que se está a visualizar [4]. No entanto, o *matching*[3] [4] entre as várias imagens da cena que se está a visualizar é um procedimento complexo, pesado computacionalmente e com custos muito elevados. Como consequência dos factos apresentados, a utilização destes sistemas no controlo visual de robôs é bastante rara. Note-se que neste tipo de sistemas também se incluem a combinação de visão estéreo com visão monocular, como é o caso por exemplo de um robô com uma câmara no elemento terminal e um par estéreo a visualizar o robô e o objecto.

2.3 Dificuldades dos Sistemas de Visão em Operar

Muitas vezes os robôs operam em ambientes severos, os quais não oferecem as condições mínimas para que o sistema de visão possa operar com a eficiência desejada. Esta situação pode ocorrer em ambientes com sobreposição de sombra dos objectos, em condições de iluminação deficiente ou com iluminação excessiva. De modo a contornar este obstáculo, a célula de produção deve ser projectada tendo em conta este factor. A iluminação da área de

3 matching – relação existente entre diferentes imagens da mesma cena.

trabalho do robô deve ser controlada de modo a que a imagem adquirida possa ser processada eficientemente.

Uma solução para eliminar este problema, muito utilizada na indústria, é evitar parcialmente ou totalmente a luz solar e luz proveniente de iluminação na área de trabalho do robô e, ilumina-la com luz artificial que facilmente é controlada.

Hoje em dia o sucesso das máquinas de visão no campo da robótica está bem demonstrado. Esta é uma área de estudo muito extensa que todos os dias apresenta novas soluções e ainda muitos mais desenvolvimentos são esperadas (por exemplo em modelação 3D, bin-picking, etc.).

CAPÍTULO TRÊS

3 Aquisição e Tratamento de Imagem

A realização da presente dissertação visa o desenvolvimento de uma aplicação que utilize visão para o controlo de uma célula robótica. Para esse efeito, foi utilizado um sistema de *visão passivo monocular* com a configuração *eye-in-hand*. Utilizou-se este sistema uma vez que a aplicação pretendida não necessita da aquisição de informação tridimensional. A informação acerca da terceira dimensão é estabelecida com base em características dos objectos presentes na cena, na sua forma planar. Outra razão para esta escolha prende-se com o objectivo de dotar o sistema com uma maior flexibilidade, i.e. estando a câmara fixa à extremidade móvel do robô é possível utilizar várias áreas de trabalho.

3.1 Bibliotecas de Visão

O processamento de imagem ao longo dos últimos tempos tem vindo a ser muito utilizado, pois a sua aplicação está presente nas mais diversas áreas. Seja a nível industrial bem como a nível doméstico. Dada a sua elevada utilização surgiu a ideia de criar bibliotecas de programação. Subprogramas facilmente incorporáveis em programas a desenvolver, esses subprogramas deviriam ser genéricos e muito flexíveis, para que os programadores das mais diversificadas áreas que utilizem processamento de imagem pudessem usufruir deles, despendendo muito menos tempo na elaboração dos seus projectos. Uma vez, os algoritmos já estarem desenvolvidos e implementados, de uma forma correcta e eficiente.

Este tipo de ferramenta apenas pretende ser parte integrante do *software* de programação (Linguagem de Programação), estando assim dependente deste. Existem várias bibliotecas disponíveis na internet (normalmente grátis) que podem ser utilizadas, destacam-se *OpenCV* que é utilizado na linguagem de programação C/C++, *SharperCV* utilizado em C#, *Aforge.NET* utilizado em C#, etc. Nesta dissertação é desenvolvido durante o aparato

experimental uma aplicação que utiliza a biblioteca *SharperCV*. Foi escolhida esta biblioteca uma vez se estar a utilizar a linguagem de programação C#, esta biblioteca oferecer as funcionalidades desejadas para o trabalho pretendido e ser de fácil interpretação.

3.2 Característica da Câmara e da Lente

A **câmara** utilizada na realização desta dissertação foi uma *uEye UI-1410-C*, na figura 3.1 é ilustrada esta câmara e as suas características são apresentadas em baixo. Foi escolhida esta câmara uma vez ser utilizada na indústria, cumprir os requisitos necessários e existir no Laboratório de Robótica do Departamento de Engenharia Mecânica da Faculdade de Ciências e Tecnologia da Universidade de Coimbra, onde este trabalho foi realizado.

Figura 3.1: Câmara uEye UI-1410-C.

Características da câmara uEye UI-1410-C:

- ✓ Sensor CMOS 1/3”
- ✓ Resolução 640 (H) × 480 (V) píxeis
- ✓ Tamanho dos píxeis 7.5 (H) × 7.5 (V) um
- ✓ Formato VGA a cores
- ✓ Velocidade 13 img/seg a máxima resolução
- ✓ Aquisição selecionavel, que permite aumentar a velocidade
- ✓ Conexão Trigger externo
- ✓ Obturador rotativo
- ✓ Parâmetros seleccionáveis através de conexão digital

- ✓ Adaptador C
- ✓ Conexão digital USB 2.0
- ✓ Tamanho 34 × 32 × 27.4 mm
- ✓ Peso 62 g

A **Lente** utilizada na realização desta dissertação é uma lente de ampliação de 8×. Utilizou-se esta lente uma vez ser necessário uma "grande" ampliação, dado que o robô não permitira colocar a câmara a uma grande distância da cena a adquirir. Esta lente revelou-se uma boa solução constituindo uma boa relação entre a distância câmara/cena e a área focada pela câmara.

Um factor que se verifica nas lentes é, tendo em conta a relação preço/qualidade, quando se aumenta a ampliação da lente a distorção também aumenta. Como se apresenta no capítulo quatro a distorção é um factor que afecta gravemente um sistema de visão, tendo em conta estes dois factores nesta fase foi posto em causa a escolha da lente, uma vez que surgiu a hipótese de utilizar uma lente de menor ampliação e adquirir várias imagens construindo a cena a partir dessas imagens. No entanto, essa hipótese foi afastada dado que seria necessário movimentar o robô para várias posições de modo a adquirir as imagens, o que aumentaria consideravelmente o tempo de ciclo.

3.3 Extracção de Informação de uma Cena

Quando se fala em sistemas de visão, estes devem ter associado *software* capaz de extrair a informação presente na(s) cena(s) observada(s) pela(s) câmara(s), vulgarmente designado na literatura por Processamento de Imagem. Segundo *Albuquerque* [18] "*Processar uma imagem consiste em transformá-la sucessivamente com o objectivo de extrair mais facilmente a informação nela contida*". Na realidade, Processamento de Imagem é apenas um dos passos que é necessário percorrer para poder extrair a informação referida em cima. De uma forma generalista, pode dividir este processo em três passos, são eles:

- ➢ Processamento de imagem: Imagem à entrada → Imagem à saída
- ➢ Análise de imagem: Imagem à entrada → Medidas à Saída
- ➢ Interpretação de imagem: Imagem à entrada → Descrição de Alto Nível à Saída

Tabela 3.1: Etapas do Processo de Extracção de Informação de uma Cena

	ETAPA		DESCRIÇÃO
1	Processamento de Imagem	Aquisição de imagem	Amostragem, armazenamento e compactação
2		Melhoramento de imagem ("*image enhancement*")	Pré-tratamento digital da imagem
3		Segmentação da informação	Extracção dos objectos do "fundo" da imagem
4	Análise de Imagem ou Parametrização		Determinação de parâmetros dos objectos presentes na imagem
5	Interpretação de Imagem		Classificação do(s) objecto(s)

De uma forma mais detalhada passa-se agora a descrever o Processo de Extracção de Informação de uma Cena (apresentado na tabela 3.1). Neste processo, o primeiro passo a efectuar é a aquisição de imagem [16], que consiste em uma câmara captar uma imagem real de uma cena. A câmara armazena a informação (imagem capturada) temporariamente sob a forma de sinal analógico e, de seguida converte-a na sua forma digital. A imagem digital é discretizada espacialmente (ou seja em *x* e *y*) originando uma matriz de pontos, *píxeis*, figura 3.2. Posto isto a informação digital é enviada a um computador.

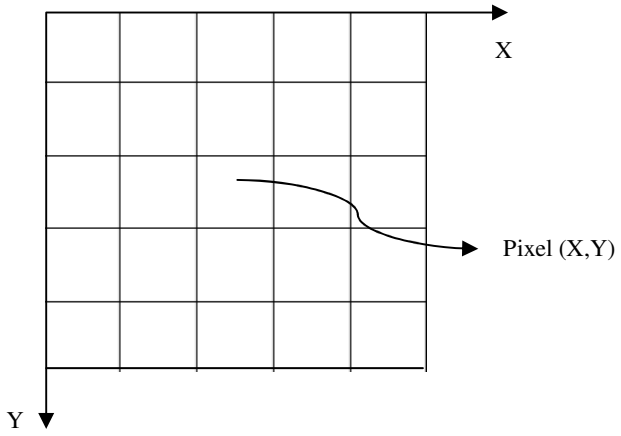


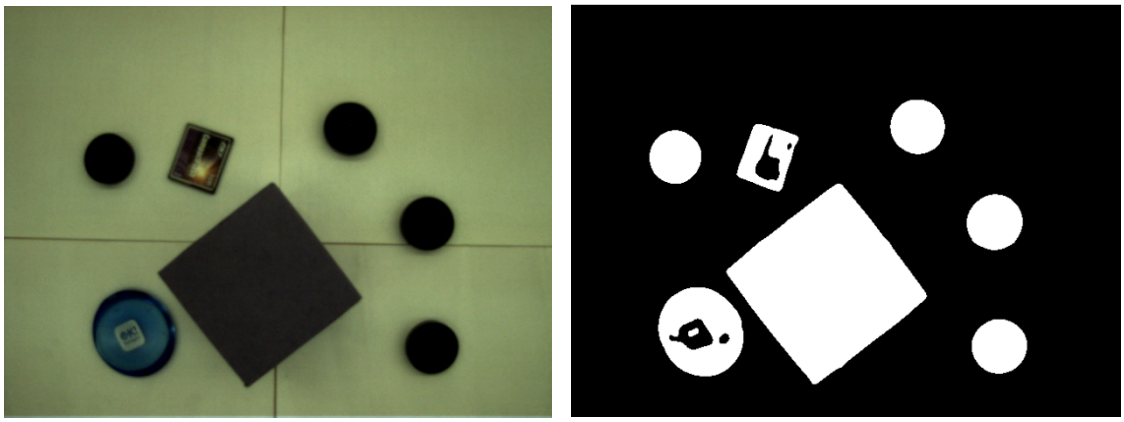
Figura 3.2: Grelha de píxeis.

O passo seguinte é o melhoramento da imagem "image enhancement", nesta etapa é realizada um pré-tratamento da imagem, i.e. a imagem é corrigida de forma a ser conseguida uma identificação mais fácil dos objectos na imagem. Para isso a imagem é filtrada, ou seja os

píxeis são processados e são aplicados algoritmos que utilizam na sua base matemática a Convuloção, Análise de Fourier e Análise Estatística [18]. As operações realizadas nesta etapa são a redução de ruído e o aumento do contraste, entre outras.

Numa imagem podem co-existir vários objectos, sendo que só parte deles são geralmente de interesse para a tarefa a realizar pelo robô, logo é necessário localizar estes últimos na imagem.

O passo seguinte a realizar é a segmentação da informação, nesta fase a imagem é discretizada em luminância (níveis de cinza). Esta transformação em níveis de cinza visa colocar em evidência as regiões de interesse e reduzir a quantidade de informação para um processamento de imagem mais rápido. As regiões de interesse serão posteriormente analisadas por algoritmos especializados em busca de informações ditas de "alto nível".



(a) (b)

Figura 3.3: (a) Imagem de uma cena real; (b) Imagem Binarizada.

Como se demonstra nas figuras 3.3 (a) e (b), a segunda representa a imagem segmentada da primeira. Esta segmentação culminou em apenas dois tipos de regiões, as regiões pertencentes ao relevo da imagem "*foreground*" (região a branco) e a região do fundo da imagem "*background*" (região a preto). Esta imagem, com dois níveis de cinza, é conhecida como imagem Binária. Devido a grandes facilidades na manipulação deste tipo de imagem, principalmente porque se reduz significativamente a quantidade de dados, elas são frequentemente utilizadas no processo de tratamento da informação.

Após o objecto estar localizado na imagem é necessário parametriza-lo, i.e. extrair e calcular as características do objecto na imagem, sobre as quais se procederá à fase de interpretação da imagem. Estas características extraídas da imagem podem ser pontos, curvas ou estruturas particulares de níveis de cinzento, entre outras explicitadas em [4, 16, 19, 20,

21]. As características do objecto na imagem são processadas e calculados parâmetros do objecto na imagem, tais como: área, perímetro, forma, descrição estrutural, topologia, etc. De referir que as características do objecto na imagem são a base do controlo visual baseado na imagem. Na figura 3.4 encontram-se esquematizadas todas as operações conducentes à extracção de informação de uma imagem ou sequência de imagens e, serve de base ao aparato experimental presente nesta dissertação.

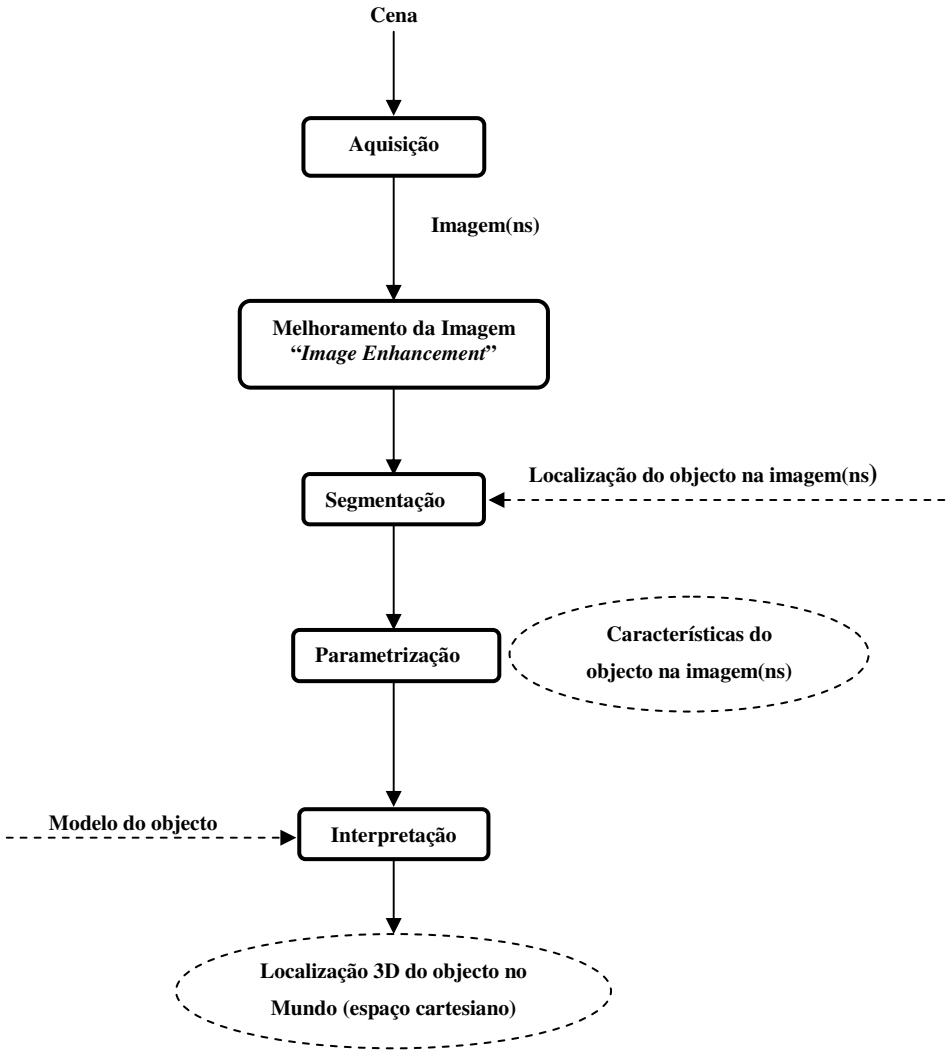


Figura 3.4: Resumo dos procedimentos de Visão por Computador.

3.4 Software de Visão Desenvolvido

O software desenvolvido para esta aplicação utiliza a linguagem Microsoft *Visual Studio 2005 – C#* e o sistema operativo utilizado é o *Windows Vista*™.

Este software consiste no processamento de imagem que se inicia com a aquisição de uma imagem, utilizando uma câmara *uEye UI-1410-C*, de seguida a imagem é analisada e interpretada utilizando funções da biblioteca *SharperCV*. O objectivo é extrair da imagem adquirida a informação dita de "alto nível", cor, posição (x , y), e área de contorno do objecto ou objectos presentes na imagem. A informação extraída da imagem é utilizada para controlar um robô. Neste subcapítulo será apresentado o software desenvolvido "*IndustrialRobotCV*".

3.4.1 Aquisição de Pontos

Como referido no subcapítulo 3.3 o processamento de imagem divide-se em três etapas: aquisição, melhoramento da imagem e segmentação. Para fazer a aquisição de imagem é usada uma biblioteca disponibilizada pelo fabricante da câmara *uEye*, designada de *uEyeCamlib*. São captadas imagens de dimensão (640 , 480).

O código principal passa-se a ilustrar:

```
int camera_type = axuEyeCam1.GetCameraType();
axuEyeCam1.InitCamera(1);
axuEyeCam1.SetImageSize(640 , 480);
axuEyeCam1.SetColorMode(1); // bit depth = 24
```

(a) (b)

Figura 3.5: a) Imagem Normal; b) Imagem Suavizada.

A imagem é armazenada e passa-se a realizar o seu melhoramento através da função ***cvSmooth***, presente na biblioteca *SharperCV*. Esta função, como se pode logo a partida imaginar, a partir da tradução da palavra inglesa "Smooth", tornará a imagem mais suave. Este processo normalmente desfoca a imagem, como se pode ver na figura 3.5, no entanto as

diferentes regiões na imagem são evidenciadas. Na prática esta função aplica o modelo matemático de um *filtro gaussiano* à imagem, i.e. faz uma média ponderada pela equação gaussiana, equação 3.1.

O passo seguinte é a aplicação da função ***cvThreshold***, esta função reduz o ruído presente na imagem e transforma-a na sua escala de cinza ou seja segmenta a imagem. Neste caso irá segmentar a imagem em apenas dois níveis de cinza, por outras palavras binariza a imagem.

$$g(x,y) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{d^2}{2\sigma^2}} \qquad 3.1$$

$$d = \sqrt{(x - x_c)^2 + (y - y_c)^2} \qquad 3.2$$

Na prática é quantificado o valor do brilho numa região em torno de cada pixel (função *g*, ver equação 3.3 e 3.4), estando pré-estabelecido um valor para o brilho (t) no qual valores superiores a este são considerados como façam parte do objecto, aparecendo na imagem binarizada a branco e, valores inferiores são considerados como façam parte do fundo da imagem, aparecendo na imagem binarizada a preto. Este valor está pré-estabelecido, no entanto o utilizador quando inicia o programa deve optimiza-lo, uma vez ele estar dependente da luminosidade incidente nos objectos (normalmente varia ao longo do dia) e da propriedade de reflexão dos materiais que constituem os objectos.

$$T_{global}(g) \begin{cases} 0\,, se\ g < t \ \rightarrow (preto) & 3.3 \\ 1\,, se\ g \geq t \ \rightarrow (branco) & 3.4 \end{cases}$$

Após a binarização da imagem é utilizada a função ***convex hull***, esta é uma função matemática que neste caso é aplicada a um espaço bidimensional (imagem). Recorrendo a ela pretende-se estabelecer a fronteira externa de cada objecto na imagem. Esta função permite gerar um polígono (geralmente irregular) que engloba todos os pontos pertencentes à região considerada como fazendo parte de um objecto, por outras palavras o polígono gerado é a área convexa mínima que engloba todos os pontos pertencentes ao respectivo objecto. Um exemplo deste tipo é ilustrado na figura 3.6. No entanto, a convexidade é analisada apenas numa pequena área em torno do ponto a analisar, surgindo assim objectos com partes concavas.

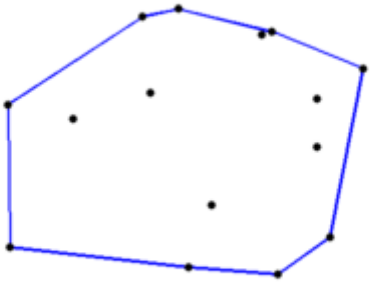


Figura 3.6: Aplicação da função *Convex Hull*.

Após os contornos externos estarem encontrados, eles são desenhados na imagem. O que de facto o *software* faz é desenhar os pontos do contorno, permitindo ao utilizador optimizar o valor do brilho (t) até que os objectos estejam bem definidos na imagem. No programa em concreto tanto a imagem com os objectos definidos como a escolha do valor para o brilho (t) aparecem numa janela com o nome "*contours*". Na figura 3.7 apresenta-se esta janela, na parte superior de imagem encontra-se uma barra de deslocamento que permite optimizar o valor do brilho (t) e no centro da imagem encontram-se definidos a rosa os contornos de dois objectos.

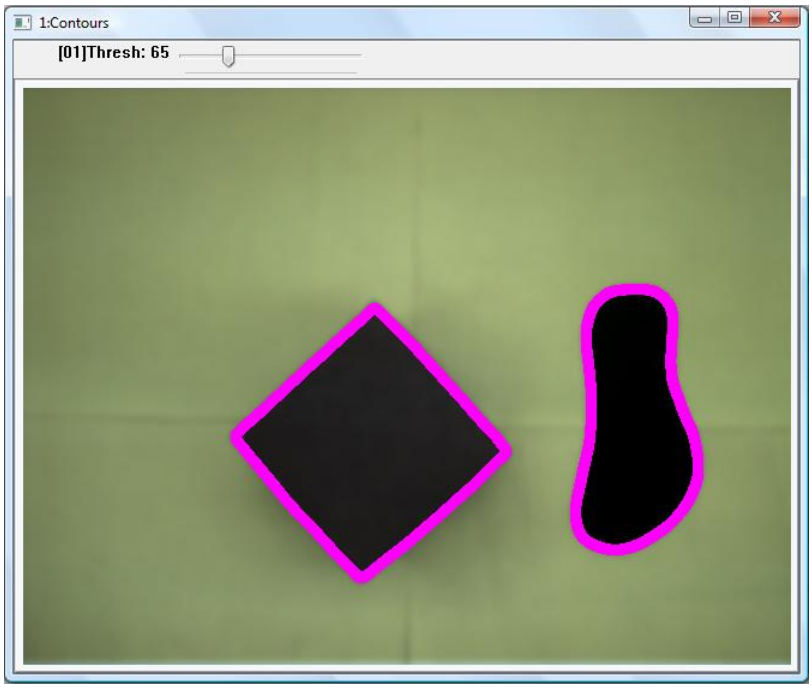


Figura 3.7: Janela *Contours* do programa *IndustrialRobotCV*.

Este método apresenta algumas falhas, uma vez só considerar contornos externos, o objecto detectado por vezes apenas é uma aproximação do objecto real. As falhas mais graves detectadas são quando os objectos contêm furos, uma vez estes serem desprezados e tudo é considerado como fazendo parte do objecto. Situação semelhante encontra-se quando na imagem figura um objecto a rodear outro. Neste caso o *software* considera os dois objectos como um só, o objecto exterior, mesmo no caso de existir espaço entre os objectos.

A função descrita anteriormente, no *software* desenvolvido toma o nome de *Vertexcountors*, esta função calcula como descrito anteriormente e "escreve" numa variável, de seu nome *cs[i]*: o número de objectos, as coordenadas de cada ponto pertencente a cada objecto e a área de cada objecto.

O centro de gravidade de cada objecto (X_g , Y_g) é determinado com base na posição dos pontos do contorno de cada objecto (X_i , Y_i), calculando a média desses pontos, equação 3.5.

$$\left(X_g, Y_g\right) = \frac{\sum_i^n (X_i, Y_i)}{n} \qquad 3.5$$

Para extrair a cor do objecto é analisado o pixel do centro de gravidade e de mais alguns que o rodeiam, transformando a cor representada na imagem numa cor RGB. A cor real por vezes é difícil de determinar uma vez depender da luminosidade, sombras e capacidade reflectora dos materiais que constituem os objectos.

CAPÍTULO QUATRO

4 Calibração da Câmara e da Imagem

Com o objectivo de calibrar a câmara, começou-se por encontrar uma posição e orientação para o robô, onde a câmara conseguisse focar toda a área pretendida e o plano da imagem na câmara fosse o mais paralelo possível à cena real que se pretendia adquirir. Uma vez a análise desse paralelismo apenas ser realizada através de uma análise macroscópica a "olho nu", está desde logo sujeita a erros. Na prática a orientação escolhida mais não foi aquela que colocava a extremidade livre do robô paralela a sua própria base.

Uma grande questão que surgiu ao longo da elaboração deste trabalho foi o facto da lente utilizada introduzir uma grande distorção na imagem (ver figura 4.1). Uma vez a lente ser de grande ampliação e a sua geometria ser de baixo rigor dimensional, afectando gravemente a qualidade final da imagem. Duas soluções logo à partida surgiram, adquirir uma nova lente de "boa qualidade" ou resolver o problema utilizando *software*. Acabou-se por optar pela segunda solução, uma vez que uma lente de "boa qualidade" é extremamente cara.

Recorrendo a software e com vista a solucionar este problema, dois métodos foram estudados, implantados e testados, culminando na adopção de um deles.

4.1 Calibração de Imagem (Primeiro Método)

Neste primeiro método começou-se por calibrar a câmara em relação ao canto superior esquerdo da imagem, através de um padrão xadrez. Escolheu-se este ponto, uma vez ser de fácil localização na imagem e de se poder criar facilmente uma relação entre o referencial cena e o referencial do robô, o que facilita a calibração do robô. Desde logo verificou-se uma grande distorção na imagem, denominado na literatura por *efeito olho de peixe*, como se pode visualizar na figura 4.1. De modo que a referência a partir do canto superior esquerdo foi de um ajuste extremamente difícil, uma vez não se conseguir colocar com muito rigor o

referencial da imagem adquiria com as mesmas direcções do referencial cena. Na figura 4.1, as linhas representadas são na verdade linhas equidistantes de trajectória rectilínea, facilmente se observa que a distorção existente é superior nos cantos da imagem e vai diminuindo até ao seu centro. Um outro aspecto importante, que é de realçar, é a inexistência de ortogonalidade entre as duas direcções, horizontal e vertical (x ; y), tendo como origem o ponto representado pelo canto superior esquerdo da imagem. De modo que para utilizar esse ponto como origem do referencial, ele foi definido na cena e na imagem e tentou-se alinha-lo na direcção vertical (y) com o ponto representado pelo canto inferior esquerdo, que estivesse na mesma direcção vertical na cena. Uma vez esse alinhamento ser feito macroscopicamente logo nesta fase existirá um erro.

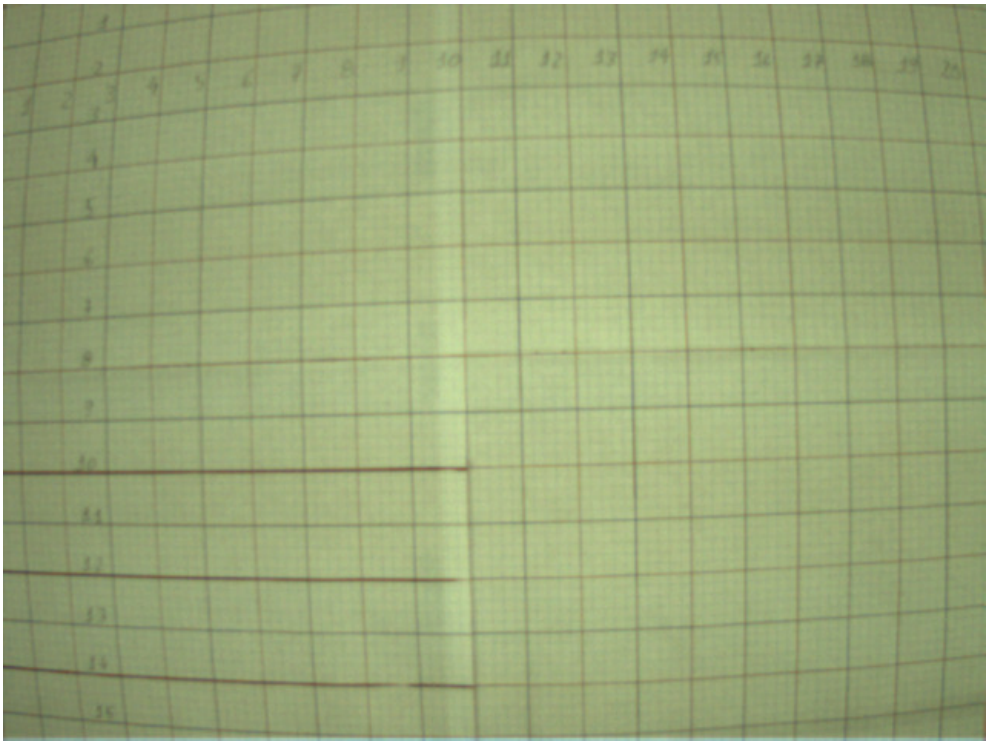


Figura 4.1: Padrão Xadrez utilizado na primeira calibração.

O passo seguinte foi a implantação do método que visa quantificar a distorção e estabelecer a relação entre píxeis e milímetros. O padrão xadrez foi construído fazendo linhas equidistantes, que distam 25 [mm] entre si. Uma vez o canto superior esquerdo ser considerado como origem do referencial cena, todas as coordenadas dos pontos que figuram na imagem são facilmente recolhidas. Considerando as linhas horizontais como direcção x e sentido da esquerda para a direita e, as linhas verticais como direcção y e sentido de cima para

baixo, foi realizada com base na imagem, a recolha de coordenadas em milímetros e também as correspondentes em píxeis.

A imagem foi discretizada em intervalos na unidade [pixel] nas duas direcções e a informação recolhida agrupou-se nos diversos intervalos, como se ilustra na tabela 4.1. A discretização da imagem foi feita em 20 intervalos segundo a direcção horizontal (x), para calibrar o eixo dos yy's, e em 15 intervalos segundo a direcção vertical (y), para calibrar o eixo dos xx's.

Tabela 4.1: Discretização da imagem [pixel].

Intervalos na direcção x:	
1	[0 ; 30]
2	]30 ; 60]
3	]60 ; 90]
4	]90 ; 120]
5	]120 ; 150]
6	]150 ; 185]
7	]185 ; 215]
8	]215 ; 250]
9	]250 ; 285]
10	]285 ; 320]
11	]320 ; 355]
12	]355 ; 390]
13	]390 ; 425]
14	]425 ; 460]
15	]460 ; 495]
16	]495 ; 525]
17	]525 ; 560]
18	]560 ; 590]
19	]590 ; 615]
20	]615 ; 640]

Intervalos na direcção y:	
1	[0 ; 20]
2	]20 ; 50]
3	]50 ; 80]
4	]80 ; 110]
5	]110 ; 145]
6	]145 ; 180]
7	]180 ; 215]
8	]215 ; 250]
9	]250 ; 280]
10	]280 ; 315]
11	]315 ; 350]
12	]350 ; 385]
13	]385 ; 420]
14	]420 ; 450]
15	]450 ; 480]

Os intervalos em que a imagem foi discretizada como se pode ver não têm todos a mesma amplitude, este facto deve-se à dispersão dos dados recolhidos. Para estabelecer os limites dos intervalos fez-se uma média com os três pontos pertencentes às extremidades dos intervalos.

Posto isto passou-se a calcular uma equação matemática para cada intervalo que representasse as relações pretendidas, distorção e relação píxeis milímetros.

4.1.1 Equações Matemáticas

Vários são os tipos de funções que podem representar um conjunto de pontos, como por exemplo função exponencial, logarítmica, potencial, linear, polinomiais de várias ordens, entre outras. No entanto, há sempre uma que ajusta melhor cada conjunto de pontos. De modo a encontrar o tipo de regressão que melhor ajusta esta situação foram calculados vários tipos de regressão e o respectivo erro entre a equação calculada e a situação real.

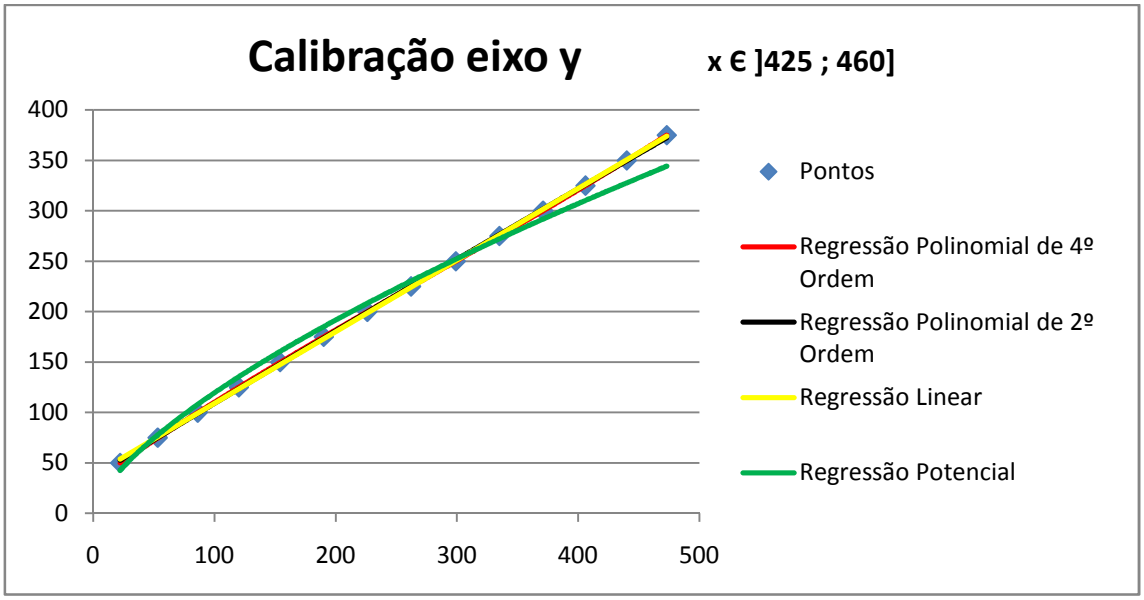


Gráfico 4.1: Calibração do eixo y no intervalo x ∈]425 ; 460].

Foram calculados vários tipos de regressão para vários conjuntos de pontos, no entanto neste documento apenas será ilustrado para um conjunto de pontos. No gráfico 4.1 está representado a calibração do eixo y num dos intervalos, como se pode ver, a Regressão Potencial faz um péssimo ajustamento relativamente aos outros tipos de regressão representados.

Para este caso onde se efectuou a calibração do eixo y, no intervalo onde x ∈]425 ; 460], foram calculadas quatro tipos de funções de ajustamento e calculou-se o erro médio que pode ser visualizado na tabela 4.2.

Tabela 4.2: Erros nas regressões calculadas.

Tipo de Regressão:	Equação:	Coeficiente de Correlação (r^2)	Erro médio:
Regressão Potencial:	$y_c = 4{,}3241y^{0{,}7116}$	0,987	8,409132259
Regressão Linear:	$y_c = 0{,}7096y + 38{,}283$	0,9997	1,391714286
Regressão Polinomial de 2ª Ordem:	$y_c = -5E\text{-}05y^2 + 0{,}7322y + 36{,}417$	0,9998	1,232589286
Regressão Polinomial de 4ª Ordem:	$y_c = 8E\text{-}11y^4 + 6E\text{-}07y^3 - 0{,}0005y^2 + 0{,}8309y + 32{,}094$	1	2,148795622

Da análise da tabela 4.2 facilmente se conclui que o melhor ajustamento é realizado pela regressão polinomial de segunda ordem, uma vez o coeficiente de correlação ser muito próximo de um e o erro médio ser inferior a qualquer outro tipo de ajustamento analisado. Tal facto já era esperado uma vez a distorção introduzida, em cada intervalo, aparentar a descrição de uma forma circular.

Análise análoga foi efectuada para os outros intervalos e também na direcção vertical (y), sendo os resultados obtidos em tudo semelhantes ao já referido em cima. Passa-se a ilustrar as equações calculadas resultantes da análise efectuada nas tabelas 4.3 e 4.4.

Tabela 4.3: Equações de calibração na direcção vertical.

Intervalos na direcção x [pixel]:		Equação:	Erro médio [mm]:
1	[0 ; 30]	$y_c = -7E\text{-}05x^2 + 0{,}8201x + 7{,}5919$	0,979685333
2	]30 ; 60]	$y_c = -6E\text{-}05x^2 + 0{,}8022x + 11{,}952$	0,946632
3	]60 ; 90]	$y_c = -7E\text{-}05x^2 + 0{,}7966x + 14{,}992$	1,096307333
4	]90 ; 120]	$y_c = -6E\text{-}05x^2 + 0{,}7817x + 18{,}407$	1,35428
5	]120 ; 150]	$y_c = -7E\text{-}05y^2 + 0{,}775y + 22{,}002$	1,205010667
6	]150 ; 185]	$y_c = -8E\text{-}05y^2 + 0{,}7704y + 24{,}398$	1,254714667
7	]185 ; 215]	$y_c = -7E\text{-}05y^2 + 0{,}7582y + 27{,}42$	1,414704
8	]215 ; 250]	$y_c = -5E\text{-}05y^2 + 0{,}7378y + 31{,}501$	1,153732143
9	]250 ; 285]	$y_c = -5E\text{-}05y^2 + 0{,}7333y + 33{,}35$	1,112346429
10	]285 ; 320]	$y_c = -8E\text{-}05y^2 + 0{,}7421y + 34{,}386$	0,933301538
11	]320 ; 355]	$y_c = -5E\text{-}05y^2 + 0{,}7285y + 36{,}031$	0,976907143
12	]355 ; 390]	$y_c = -5E\text{-}05y^2 + 0{,}7287y + 36{,}808$	1,043807143
13	]390 ; 425]	$y_c = -4E\text{-}05y^2 + 0{,}7245y + 36{,}986$	1,038884286
14	]425 ; 460]	$y_c = -5E\text{-}05y^2 + 0{,}7322y + 36{,}417$	1,232589286
15	]460 ; 495]	$y_c = -4E\text{-}05y^2 + 0{,}7323y + 36{,}306$	1,072897143

16	]495 ; 525]	$y_c = -4E\text{-}05y^2 + 0{,}739y + 35{,}468$	1,202737143
17	]525 ; 560]	$y_c = -4E\text{-}05y^2 + 0{,}747y + 34{,}203$	1,05048
18	]560 ; 590]	$y_c = -5E\text{-}05y^2 + 0{,}762y + 32{,}007$	0,998860714
19	]590 ; 615]	$y_c = -4E\text{-}05y^2 + 0{,}7662y + 30{,}707$	1,048074286
20	]615 ; 640]	$y_c = -4E\text{-}05y^2 + 0{,}7662y + 30{,}707$	2,474211429

No intervalo 20, uma vez a recolha de pontos apenas ter permitido adquirir sete pontos, verificou-se que estes não eram suficientes. Tentou-se corrigir a distorção e a conversão de píxeis em milímetros, com a equação obtida do intervalo 19 o que se verificou ser mais viável que as equações calculadas com os sete pontos pertencentes ao intervalo.

Tabela 4.4: Equações de calibração na direcção horizontal.

Intervalos na direcção y [pixel]:		Equação:	Erro médio [mm]:
1	[0 ; 20]	$x_c = 1{,}4647x^{0{,}9064}$	0,929187484
2	]20 ; 50]	$x_c = -6E\text{-}05x^2 + 0{,}8025x + 15{,}594$	2,649186
3	]50 ; 80]	$x_c = -6E\text{-}05x^2 + 0{,}7916x + 18{,}112$	2,744973
4	]80 ; 110]	$x_c = -6E\text{-}05x^2 + 0{,}7865x + 20{,}04$	2,965642
5	]110 ; 145]	$x_c = -5E\text{-}05x^2 + 0{,}7759x + 22{,}223$	3,1505875
6	]145 ; 180]	$x_c = -5E\text{-}05x^2 + 0{,}7706x + 23{,}294$	3,1000925
7	]180 ; 215]	$x_c = -8E\text{-}05x^2 + 0{,}7827x + 22{,}817$	2,847954737
8	]215 ; 250]	$x_c = -9E\text{-}05x^2 + 0{,}7785x + 23{,}695$	2,661156842
9	]250 ; 280]	$x_c = -9E\text{-}05x^2 + 0{,}7803x + 22{,}571$	2,550101053
10	]280 ; 315]	$x_c = -8E\text{-}05x^2 + 0{,}7781x + 22{,}254$	2,881631579
11	]315 ; 350]	$x_c = -8E\text{-}05x^2 + 0{,}7793x + 21{,}246$	2,852350526
12	]350 ; 385]	$x_c = -8E\text{-}05x^2 + 0{,}7806x + 19{,}71$	2,723174737
13	]385 ; 420]	$x_c = -8E\text{-}05x^2 + 0{,}7834x + 17{,}744$	2,508675789
14	]420 ; 450]	$x_c = -8E\text{-}05x^2 + 0{,}7893x + 15{,}015$	2,380465263
15	]450 ; 480]	$x_c = -5E\text{-}05x^2 + 0{,}7824x + 12{,}935$	2,976828947

Na calibração do eixo horizontal x, no intervalo 1, y $\in$ [0 ; 20], devido há existência de poucos pontos a regressão potencial revelou-se uma melhor solução.

Procedeu-se ao desenvolvimento de *software*, em Microsoft Visual C# "*Camera_Calibration.cs*", no qual foram implantadas as equações calculadas, este subprograma recebe coordenadas em pixéis do referencial imagem (imagem com distorção) e retornas em coordenadas do referencial cena em milímetros. Este subprograma foi incorporado no *software IndustrialRobotCV*, o que permitiu efectuar testes e verificar o erro cometido segundo esta abordagem.

4.2 Calibração de Imagem (Segundo Método)

O segundo método consiste na incorporação de uma ferramenta já existente e disponível na internet [22]. Esta ferramenta foi alvo de teste em muitas aplicações e de um artigo científico [23], o que desde logo lhe conferiu expectativas harmoniosas.

Como no método anterior, inicialmente procurou-se encontrar uma posição para o robô adquirir imagem. Esta posição é ligeiramente diferente da posição do método anterior, uma vez que neste caso o ponto de origem do referencial imagem é o centro da imagem. Como é óbvio poder-se-ia utilizar o mesmo ponto para o robô, no entanto quando se procedesse há calibração do robô a matriz rotação seria mais complexa do que um simples factor de escala. Uma outra razão pela qual se decidiu procurar outra posição para o robô foi para facilitar a identificação do centro da imagem na cena. Para realizar essa identificação utilizou-se uma folha xadrez, que se ilustra na figura 4.2 e, desenvolveu-se um pequeno software que permitisse identificar o centro na imagem e as direcções que o referencial imagem deve seguir.

Para este caso escolheu-se como ponto de referência (origem do referencial) o centro da imagem uma vez este ser dos pontos onde existe menos distorção. E como se descreverá mais adiante neste documento, a ferramenta utilizada reconstrói a imagem a partir do seu centro, pelo que é importante colocar o centro do referencial num ponto que não irá ter alteração na distribuição da imagem.

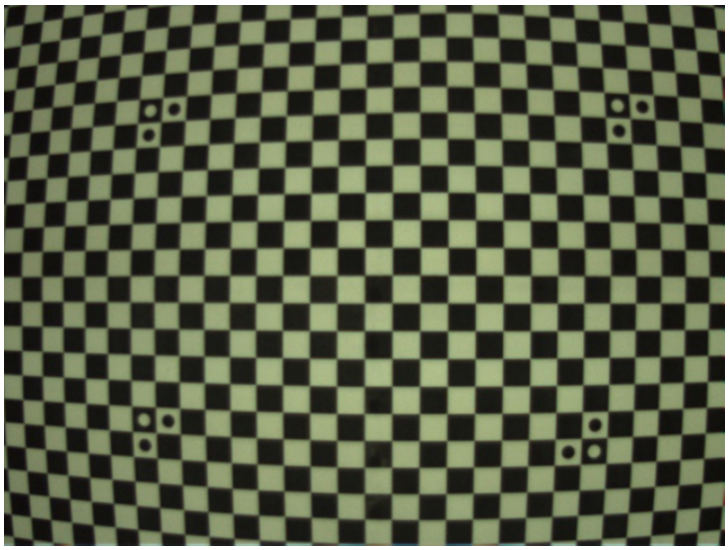


Figura 4.2: Padrão xadrez utilizado no segundo método de calibração.

Para definir o ponto de aquisição de imagem, inicialmente o padrão xadrez foi orientado com a orientação do robô, fazendo coincidir as direcções dos referenciais imagem, cena e robô. De seguida, fixou-se o padrão xadrez à mesa com a orientação estabelecida. Posto isto passou-se a procurar uma distância entre a mesa e a câmara, em que a imagem captada foca-se toda a área pretendia. Encontrada essa distância utilizou-se o *software* ilustrado na figura 4.3, para fazer coincidir os referenciais já mencionados, para isso o cruzamento das linhas a vermelho definem o centro da imagem, logo força-se que o ponto definido por esse cruzamento projecte o ponto de origem do referencial cena. Para encontrar a orientação correcta fez-se coincidir as linhas a vermelho com os lados dos quadrados que estão representados no padrão, como se apresenta na figura 4.3.

A posição e a orientação encontrada para esta situação foram:

X = - 68,901; Rx = 180,00;

Y = - 623,780; Ry = 0,00;

Z = - 83,036; Rz = - 23,66.

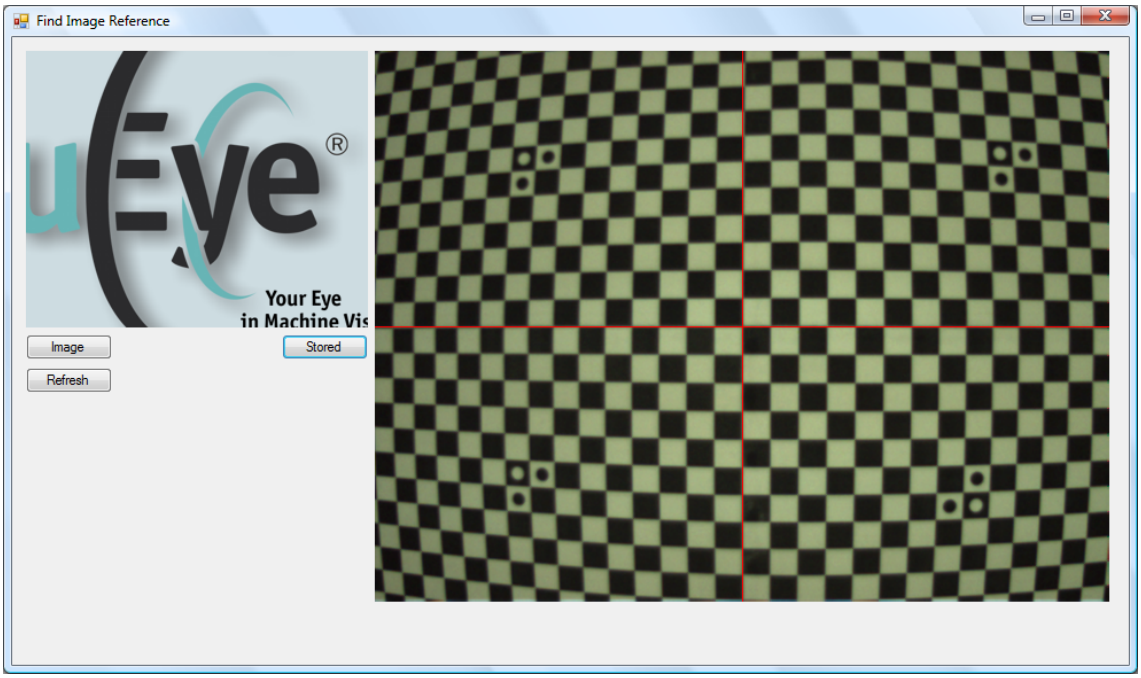


Figura 4.3: Software para encontrar e orientar o referencial imagem.

Como já foi referido a imagem recolhida tem o *efeito olho de peixe* bastante acentuado, a ferramenta utilizada neste método contempla reduzir esse efeito. A ferramenta denominada *Camera Calibration Toolbox for Matlab* tem como princípio, a distorção distribui-se uniformemente na imagem de uma forma radial, como se ilustra na figura 4.4. Na mesma

figura como se pode ver há uma referência a uma distorção tangencial, na realidade a distorção é radial e tangencial, no entanto a distorção tangencial é praticamente desprezável face a distorção radial.

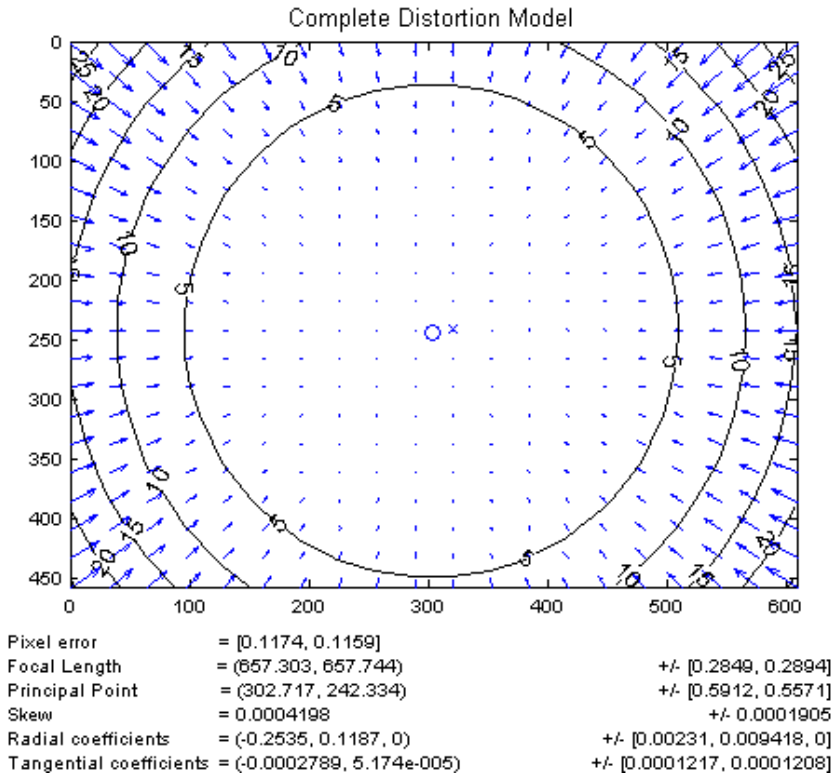


Figura 4.4: Modelo de distorção (radial e tangencial) da ferramenta *Camera Calibration Toolbox for Matlab*.

4.2.1 Determinação dos parâmetros de calibração

Para operar com a ferramenta (*Camera Calibration Toolbox for Matlab*) durante esta fase de determinação dos parâmetros de calibração utiliza-se o software na sua forma "original", i.e. operando directamente com o software *MatLab*.

Inicialmente, com o robô colocado na posição da captação de imagem, adquire-se uma imagem da cena. Estando apenas presente na cena o padrão xadrez, uma imagem típica é ilustrada na figura 4.2. Esta fase é realizada pelo software *IndustrialRobotCV*.

De seguida passa-se a utilizar a ferramenta *MatLab* (designada vulgarmente por *toolbox*), mais precisamente o subprograma com o nome *calib_gui.m*. Durante a utilização desta funcionalidade da *toolbox*, será necessário carregar a imagem no programa, definir os eixos coordenados na imagem (apenas válidos para esta abordagem), definir a área da imagem a analisar e fornecer alguns parâmetros. Os parâmetros a fornecer são o número de pontos a

detectar na imagem, as dimensões de um rectângulo que define uma pequena área próxima de cada ponto a ser detectado e a estimação do factor de distorção. A pequena área definida pelo rectângulo visa ser analisada cuidadosamente e dentro dela encontrado o ponto que se pretende saber as suas coordenadas. O factor de distorção é dado e optimizado manualmente pelo utilizador. Na figura 4.5 ilustra-se uma imagem a partir da qual foram extraídos os parâmetros de calibração.

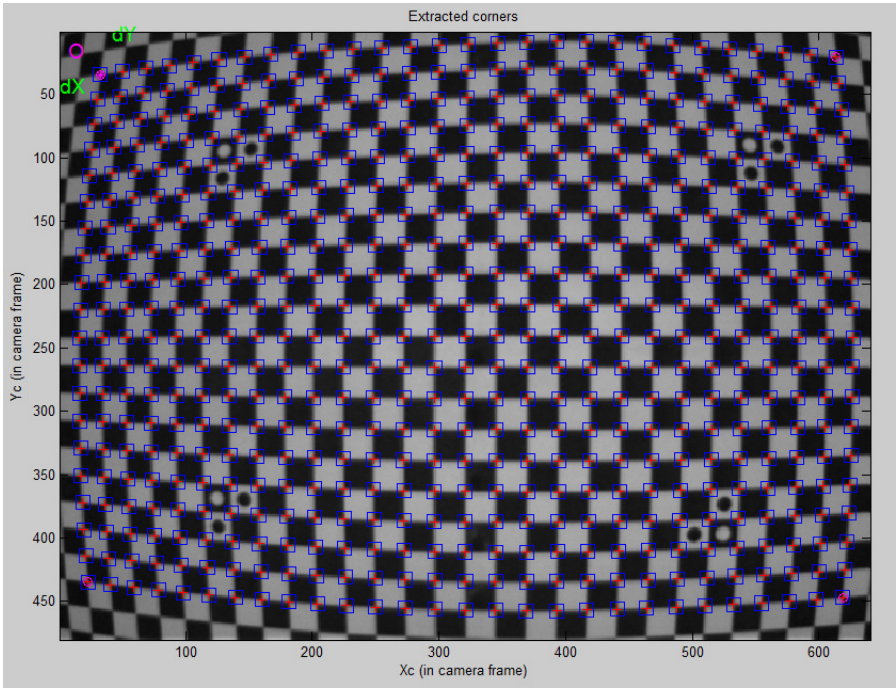


Figura 4.5: Imagem utilizada para estimar os parâmetros de calibração.

Com todos estes dados a imagem é processada e são adquiridos pontos presentes na imagem, esses pontos são estabelecidos pela intersecção dos cantos dos quadrados no papel xadrez. As coordenadas desses pontos são processados e são calculados alguns parâmetros. Os parâmetros de interesse para a situação em causa são matrizes que se designam por distância focal (fc), ponto principal (cc), ângulo entre os eixos coordenados na imagem (alpha_c) e distorção (kc). Todos os parâmetros calculados são armazenados num ficheiro com o nome *Calib_Results.m*.

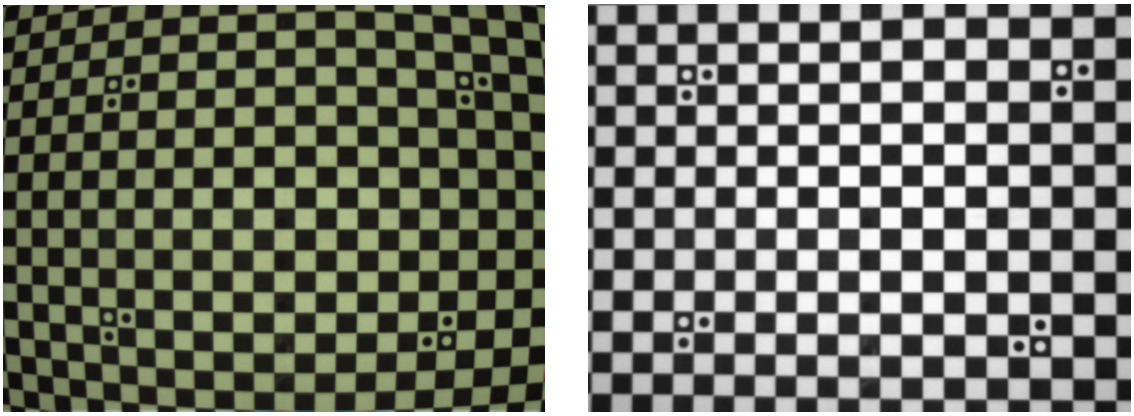
Uma explicação exaustiva acerca do modo de operar com a *toolbox* "*Camera Calibration Toolbox for Matlab*" encontra-se descrita em [22] e o método matemático encontra-se descrito extensivamente em [24].

Para esta situação resultou os seguintes dados:

```
%-- Focal length:
fc = [ 1426.107141691488800 ; 1429.227085380307500 ];

%-- Principal point:
cc = [ 319.500000000000000 ; 239.500000000000000 ];

%-- Skew coefficient:
alpha_c = 0.000000000000000;

%-- Distortion coefficients:
kc = [ -1.655003412331441 ; 3.786115523168508 ; 0.021876887169198 ;
0.018670727080197 ; 0.000000000000000 ];
```

4.2.2 Reconstrução da imagem sem distorção

Numa segunda fase, e esta será sempre efectuada desde que se pretenda adquirir informação da imagem, passa-se a "correr" o subprograma da *toolbox* com o nome de *undistort_image.m*. Esta função o que faz é reconstruir a imagem, atenuando-lhe significativamente o *efeito olho de peixe*, na figura 4.6 a) demonstra-se uma imagem adquirida com o *efeito olho de peixe*, na figura 4.6 b) demonstra-se a imagem reconstruída da primeira aplicando, o método de redução do *efeito olho de peixe*.



(a) (b)

Figura 4.6: (a) Imagem adquirida com *efeito olho de peixe*; (b) Imagem reconstruída segundo o método de redução do *efeito olho de peixe*.

Este método utiliza os parâmetros anteriormente calculados, guardados no ficheiro *Calib_Results.m*, e reconstrói a imagem pixel por pixel.

Uma vez este método ser "chamado" muitas vezes pelo *software IndustrialRobotCV*, ou seja sempre que se adquire imagem é necessário que ela seja submetida a este processo para ser reconstruída. É de todo o interesse automatizar o método e incorpora-lo no software

IndustrialRobotCV. Para alcançar este objectivo a *toolbox* foi modificada e incorporou-se no software principal um objecto COM do *MatLab* que permite comunicar com ele de uma forma automática, sem que o utilizador do *software* tenha de interagir com o *MatLab*.

De uma forma genérica e resumida apresenta-se na figura 4.7 um esquema com os passos seguidos pelo software, sendo que apenas os dois primeiros passos são abordados no presente capítulo (Calibração de Imagem) e o terceiro passo é abordado no terceiro capítulo (Extracção de Informação de uma Cena).

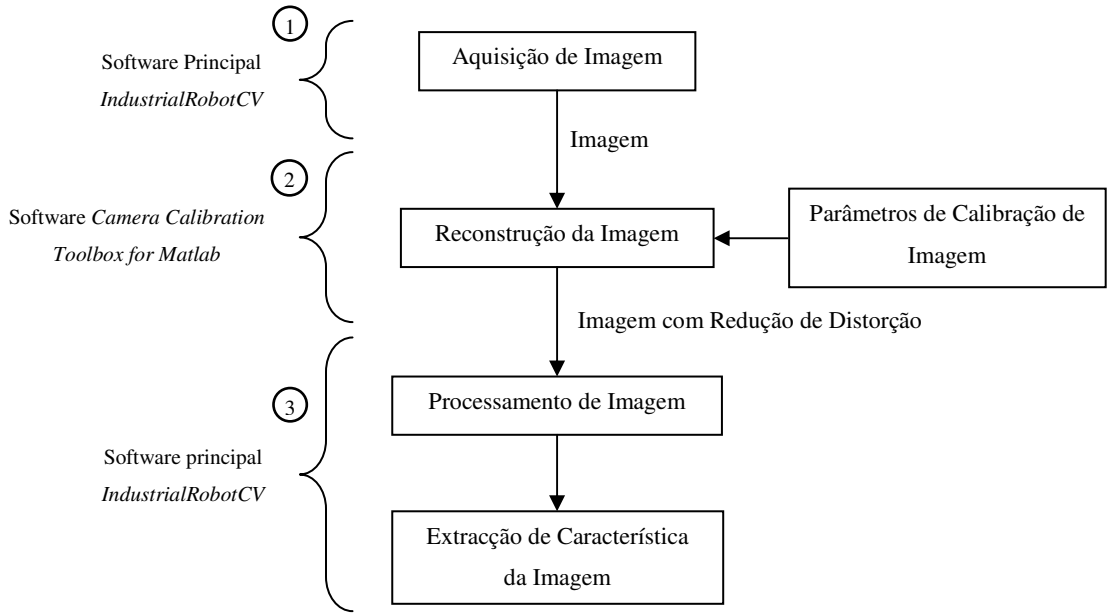


Figura 4.7: Esquema representativo da Calibração de Imagem.

4.2.3 Conversão de Coordenadas

Quando se cria uma nova imagem sem distorção, está-se a criar juntamente um novo referencial imagem, em que a unidade é o pixel. Os pontos presentes na imagem em pixéis necessitam de ser convertidos no referencial cena, expresso em milímetros. Para efectuar essa conversão é estimado o valor de um pixel em milímetros (*constante de conversão*). No entanto, devido à existência de alguma distorção na nova imagem e verificar-se que essa distorção varia de quadrante para quadrante, estima-se uma *constante de conversão* para cada quadrante e para cada direcção do referencial imagem, ou seja, são calculadas duas *constantes de conversão* para cada quadrante, uma para a direcção x ($C_{q,x}$) e outra para a direcção y ($C_{q,y}$).

O cálculo das *constantes de conversão* requer a aquisição das coordenadas, no referencial imagem, de um conjunto de pontos em cada quadrante ($x^{[pix]}$; $y^{[pix]}$). Esses pontos devem estar distribuídos por todo, respectivo, quadrante (q). As coordenadas dos respectivos pontos no referencial cena devem igualmente ser recolhidas ($x^{[mm]}$; $y^{[mm]}$). Com esses dados são calculadas as médias dos pontos recolhidos por quadrante e por direcção, segundo o referencial imagem (em píxeis, equações 4.2 e 4.4) ($K_{q,x}^{[pix]}$ e $K_{q,y}^{[pix]}$) e segundo o referencial cena (em milímetros, equações 4.1 e 4.3) ($K_{q,x}^{[mm]}$ e $K_{q,y}^{[mm]}$). Dividindo o valor de cada constante representante da média de pontos por quadrante e por direcção no referencial cena, pelo valor da respectiva constante da média de pontos por quadrante e por direcção no referencial imagem obtêm-se as oitos *constantes de conversão* (equações 4.5 e 4.6).

$$K_{q,x}^{[mm]} = \frac{\sum_{i=1}^{n} x_i^{[mm]}}{n} \qquad 4.1$$

$$K_{q,x}^{[pix]} = \frac{\sum_{j=1}^{n} x_j^{[pix]}}{n} \qquad 4.2$$

$$K_{q,y}^{[mm]} = \frac{\sum_{i=1}^{n} y_i^{[mm]}}{n} \qquad 4.3$$

$$K_{q,y}^{[pix]} = \frac{\sum_{j=1}^{n} y_j^{[pix]}}{n} \qquad 4.4$$

$$C_{q,x} = \frac{K_{q,x}^{[mm]}}{K_{q,x}^{[mm]}} \qquad 4.5$$

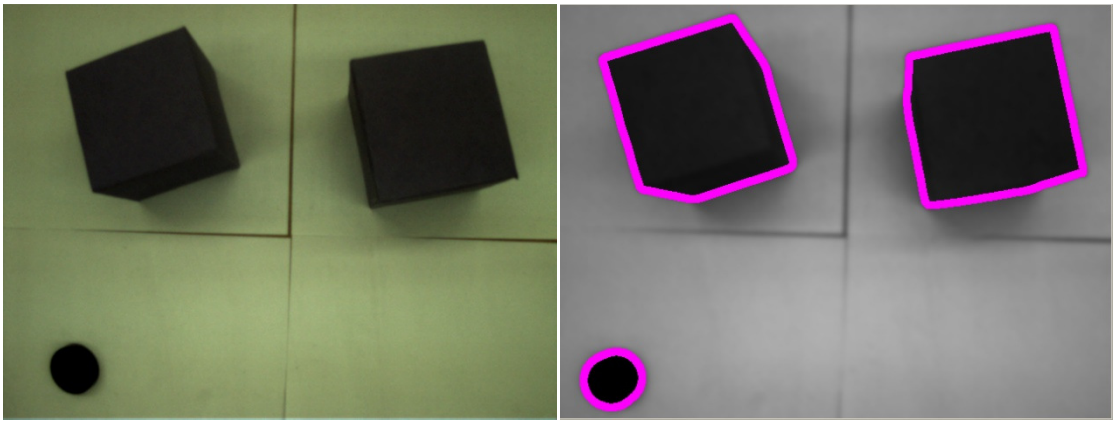
$$C_{q,y} = \frac{K_{q,y}^{[mm]}}{K_{q,y}^{[mm]}} \qquad 4.6$$

Para estimar o valor das *constantes de conversão* foram adquiridos vinte pontos (n) em cada quadrante, distribuídos aleatoriamente por cada um deles.

Para efectuar a conversão multiplica-se simplesmente o valor de cada coordenada, de cada ponto pretendido, pela respectiva *constante de conversão*.

4.3 Comparação dos Métodos de Calibração de Imagem

Nos testes realizados segundo o primeiro método de calibração de imagem verificou-se que o erro cometido variava bastante de ponto para ponto. No entanto não foi encontrada qualquer relação entre os erros cometidos de ponto para ponto, uma vez que eles provêm de fontes bastante distintas. Entre outras evidenciam-se a dificuldade de colocar a câmara numa posição onde a CCD está rigorosamente paralela ao plano de aquisição de imagem, o *efeito olho de peixe* introduzido pela lente, que não é corrigido totalmente como referido anteriormente. Outro factor é a altura dos objectos. Quando estes aparecem na imagem afastados do seu centro leva a que eles não sejam bem representados na imagem, i.e. não aparecem rebatidos na sua vista em planta, as suas faces laterais também aparecem representadas, esta situação ilustra-se na figura 4.8. O que leva o *software* a considerar o contorno maior do que realmente ele é, por sua vez, a estimação do centro de massa do objecto é calculada com base em alguns pontos errados o que leva a conter erros. Estes erros serão mais pronunciados, quanto mais alto for o objecto e quanto mais afastado ele estiver do centro da imagem. A sombra do objecto por vezes também é considerada como fazendo parte dele, ocorrendo um erro semelhante ao apresentado anteriormente.



(a) (b)

Figura 4.8: a) Imagem captada pela câmara; b) Imagem a definir os contornos dos objectos (contornos a rosa).

É ainda de referir, um factor de grande importância que tem vindo a ser alvo de estudo por parte de alguns investigadores, designado na literatura por "*absolute accuracy*" [25]. Este factor, de uma forma breve, diz respeito à posição final atingida pelo robô quando é solicitado o seu movimento, essa posição nem sempre é atingida com a precisão desejada. Sendo este erro em muitas aplicações bastante significativo.

Este método permitiu erros inferiores a nove milímetros o que não sendo considerado um mau resultado, para a aplicação em questão é excessivo. Para além disso este método não é muito versátil, veja-se caso se pretenda altera-se a distância entre a câmara e o plano de trabalho será necessário recalcular todas as equações e implanta-las novamente no subprograma de calibração, outra solução seria construir software que fizesse todo isto autonomamente o que seria bastante complexo.

Em relação ao segundo método, este apresentou um erro máximo bastante inferior ao método anterior, de apenas dois milímetros. O que se considera para a aplicação em questão uma solução aceitável. O processo de calibração de imagem embora envolva um método bastante mais complexo que o primeiro, esse facto não se traduz num aumento excessivo do tempo de cálculo. Este método é bastante mais versátil que o anterior, analise-se a mesma questão proposta anteriormente, em que se altera a distância entre a câmara e o plano de trabalho neste caso basta recalcular os parâmetros de calibração o que tomará apenas alguns minutos, ou caso se pretenda evoluir o *software*, facilmente se incorporam vários ficheiros com os parâmetros de calibração dispondo assim de vários planos de trabalho. Uma particularidade deste método, que qualquer Engenheiro valoriza, é o facto de ele seguir uma abordagem com significado físico e não apenas matemático, como é o caso do primeiro método.

De modo que se considera o segundo método uma solução melhor que o primeiro, sendo este o método utilizado na aplicação desenvolvida.

De referir ainda, que neste segundo método foi implantado uma metodologia de redução da imprecisão do robô (*absolute accuracy*). Uma vez se ter verificado que essa imprecisão aumentava com o aumento da distância a um ponto de referência, o presente método corrige essa imprecisão linearmente, i.e. dividindo a cena, no referencial cena por quadrantes e registando em que pontos são cometidos erros de um milímetro. O método incrementa entre cada dois pontos recolhidos um milímetro de forma linear, às coordenadas do ponto pretendido. Para melhor entender esta correcção ilustra-se no gráfico 4.2 os valores a incrementar no terceiro quadrante segundo a direcção horizontal x.

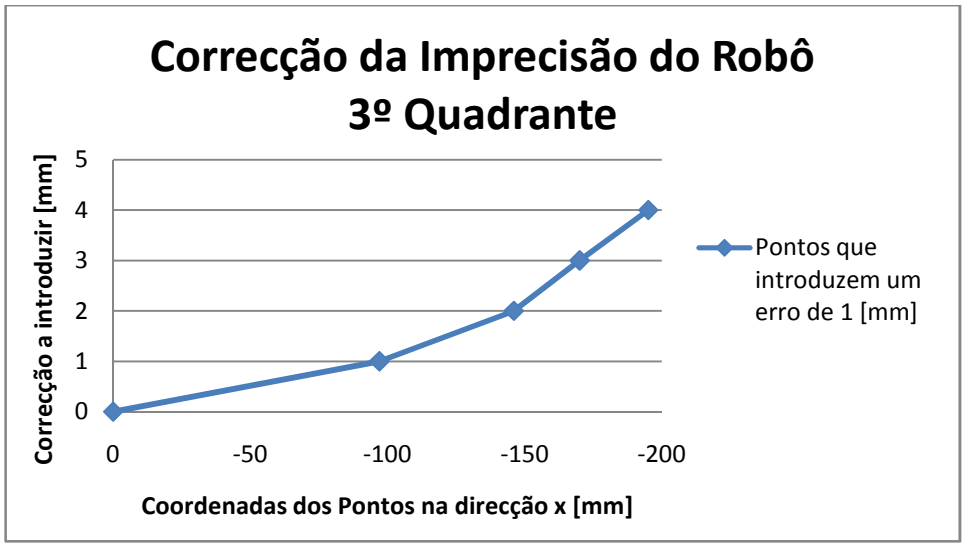


Gráfico 4.2: Correcção da imprecisão do robô segundo a direcção x no 3º quadrante.

CAPÍTULO CINCO

5 Robô Industrial

A aplicação desenvolvida como já foi descrito anteriormente tem dois modos de operação. Um visa efectuar operações de manipulação "*pick-and-place*" em objectos e o outro visa reproduzir contornos de objectos.

Para levar a cabo esta aplicação o material utilizado foi: um robô industrial, uma câmara CMOS, uma lente, uma válvula de vácuo, uma ventosa (utilizada para agarrar os objectos) e uma ferramenta composta por um lápis com um sistema de mola, para conferir alguma flexibilidade ao sistema a desenhar (utilizado para reproduzir contornos). Apresenta-se na figura 5.1 a célula robótica a operar no modo de reprodução de contornos.

Figura 5.1: Célula robótica (modo representação de contornos).

5.1 Comunicação

O sistema desenvolvido é composto por um *software* que gere todo o sistema "*IndustrialRobotCV*", ou seja, comunica com a câmara e com o robô partilhando informação. Esse *software* apresenta uma interface bastante amigável para o utilizador. A comunicação com o robô é feita recorrendo a uma *Data Link Library* (DLL) desenvolvida no Laboratório de Robótica Industrial do Departamento de Engenharia Mecânica da Universidade de Coimbra e que tem como base o protocolo de comunicação TCP/IP (Ethernet), permitindo controlar o robô remotamente.

A comunicação com a câmara também é feita recorrendo a uma DLL que o fabricante disponibiliza, mas neste caso a transferência de dados é feita através de um cabo USB, na figura 5.2 ilustra-se os modos de comunicação entre o *hardware* utilizado.

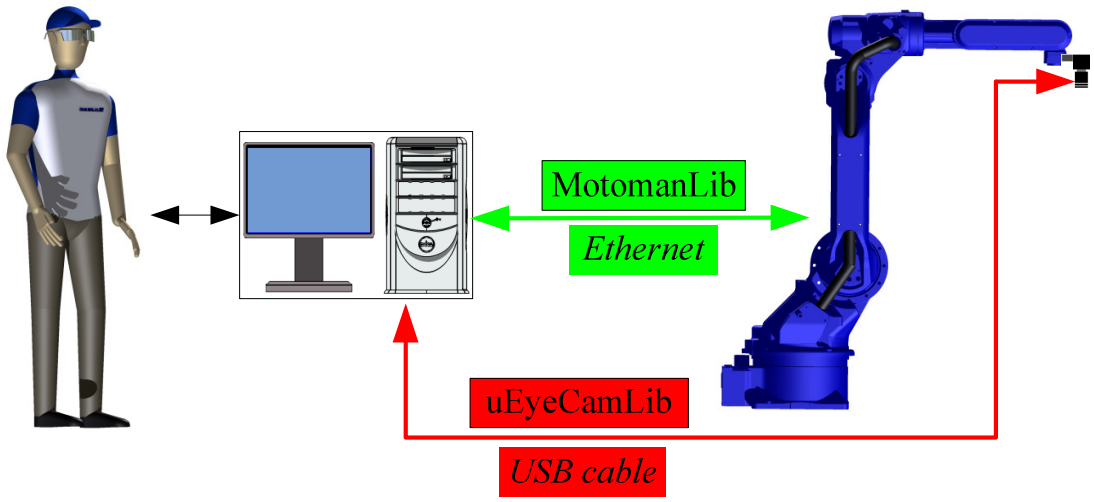


Figura 5.2: Comunicação utilizada na aplicação.

A aplicação desenvolvida recorre a vários objectos e softwares para atingir um fim. Utiliza a biblioteca *sharpercv* para fazer o "tratamento" de imagem, a *uEyecamlib* para extrair imagem da câmara, a *MotomanLib* para comunicar com o robô e ainda, utiliza um objecto COM do *MatLab* que permite comunicar com a aplicação *MatLab*, ou seja iniciar o *MatLab* e correr comandos, que irão ser úteis para a calibração de imagem. Um esquema da utilização dessas ferramentas pelo software principal ilustra-se na figura 5.3.

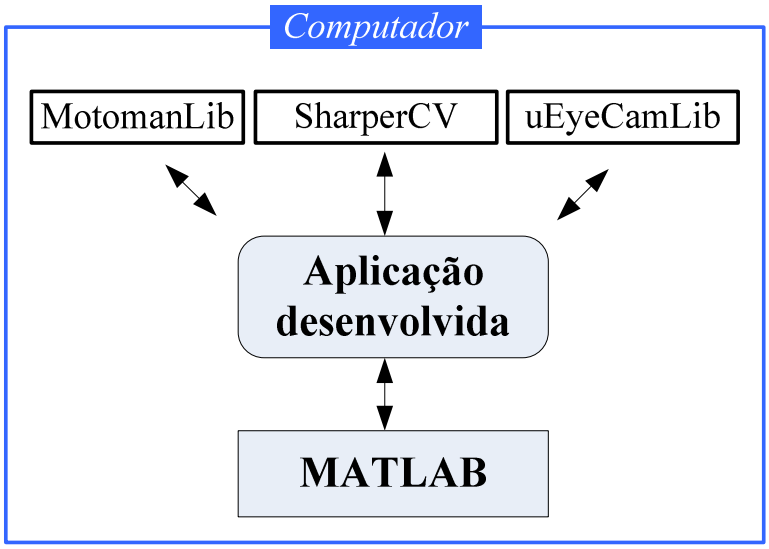


Figura 5.3: Esquema representativo dos objectos e subprogramas utilizados nesta aplicação.

5.2 Envio de Pontos

No modo de detecção de trajectórias, o software desenvolvido gera um ficheiro JBI com o código do robô. Neste caso, força-se o robô a movimentar-se ao longo dos pontos predefinidos. Passa-se a apresentar um exemplo típico de um ficheiro JBI a enviar ao robô:

```
/JOB
//NAME MOVI
//POS
///NPOS 0,0,0,1,0,0
///TOOL 0
///POSTYPE ROBOT
///RECTAN
///RCONF 0,0,0,0,0,0,0,0
P0001= 400,0,450,0,0,0
…
//INST
///DATE 2008/12/16 11:26
///ATTR SC,RW
///GROUP1 RB1
NOP
MOVJ P001 VJ=5.00
…
END
```

No modo de apanhar peças "*pick-and-place*", são enviadas remotamente para o robô as coordenadas que ele necessita para se movimentar e a velocidade com que se pretende que ele efectue esse movimento. O robô procede de imediato à realização desse movimento.

Demonstra-se de seguida um exemplo do código utilizado para movimentar o robô:

```
robot.Movj_.X = 213;
robot.Movj_.Y = -700;
robot.Movj_.Z = -400;
robot.Movj_.RX = 180;
robot.Movj_.RY = 0;
robot.Movj_.RZ = 110;
robot.Movj_.Speed = 5; ;
robot.OnMovj(robot.Movj_);
```

Para controlar o vácuo, utiliza-se uma válvula de vácuo que é controlada pelo Controlador do Robô, para controlar a válvula a partir do computador é necessário enviar a ordem para o Controlador e este por sua vez, enviar o sinal para ligar a válvula de vácuo. Para promover a criação do sinal no controlador é lhe enviado a partir do *software* "*IndustrialRobotCV*" uma variável que é memorizadas na memória do Controlador. Ao mesmo tempo o Controlador está a "correr" um programa, neste estão a ser verificadas as posições de memória em que o Controlador guarda os dados enviados pelo *software* de controlo, quando nessas posições de memória figura um determinado valor pré-estabelecido o controlador emite o sinal e a válvula liga o vácuo. Para desligar o vácuo é em tudo semelhante ao anterior apenas difere no valor que a variável enviada ao Controlador toma.

Apresenta-se de seguida um excerto do código que está a correr no Controlador do robô:

```
JUMP *LABEL31 IF I001<>19821
DOUT OT#(1) OFF
DOUT OT#(2) OFF
SET I001 0
*LABEL31
JUMP *LABEL4 IF I001<>1981
DOUT OT#(2) ON
SET I001 0
*LABEL4
```

Apresenta-se também o código do *software IndustrialRobotCV* que envia a variável ao Controlador do robô:

```
write_stat = robot.WriteInt("1", 1981);
```

5.3 Ordenação e Redução de Pontos

Para reproduzir contornos é necessário ordenar os pontos provenientes da Extracção de Informação da Imagem. Este facto deve-se há necessidade de enviar os pontos ao robô segundo a ordem que se pretende que este os percorra. Uma vez eles estarem armazenados de uma forma desordenada na variável resultante da Extracção de Informação da Imagem, *cs[]*, o *software* carece deste procedimento.

Para ordenar os pontos foi desenvolvido um algoritmo que se passa a descrever. Os pontos pertencentes a cada objecto são ordenados consoante o operador escolhe o objecto, sendo cada objecto ordenado individualmente. Para cada objecto o primeiro ponto da matriz de pontos proveniente do passo de Obtenção dos Pontos (x_i ; y_i) é estabelecido como primeiro ponto na matriz de pontos do passo Ordenação de Pontos ($x_{estab.}$; $y_{estab.}$). O segundo ponto é calculado pela distância mínima ao primeiro. A distância mínima é calculada pela equação 3.6. Os pontos seguintes são calculados como o segundo ponto, ou seja pelo calculo da distância mínima ao último ponto considerado.

$$d = \sqrt{(x_{estab.} - x_i)^2 + (y_{estab.} - y_i)^2} \qquad 3.6$$

Deve ser feita uma redução dos pontos pertencentes ao contorno do objecto. Esta redução de pontos deve-se ao facto da aquisição de imagem resultarem demasiados pontos. Como se verá mais adiante, se o número de pontos enviados ao robô for muito elevado e caso isso aconteça eles estariam muito próximos uns dos outros, o tempo de ciclo aumentaria substancialmente, bem como algumas das funções que o robô disponibiliza para definir trajectórias deixariam de fazer sentido.

A fim de efectuar a redução de pontos é estabelecido uma distância mínima entre pontos, sendo apenas considerados aqueles que pertencem à extremidade dessa distância. Aqueles que pertencem a distâncias inferior são desprezados. Uma vez encontrado o ponto com a distância igual ou ligeiramente superior à distância mínima estabelecida, ele é copiado para a matriz de pontos a enviar ao robô. De referir que o primeiro ponto também é copiado para esta matriz. De seguida o processo é repetido, mas neste caso o cálculo é efectuado para os pontos seguintes em relação ao último calculado, que cumpre a distância mínima. O processo de cálculo é repetido um número suficiente de vezes até que todos os pontos sejam analisados e sejam encontrados aqueles que de facto cumprem a distância mínima.

Para implantar este método estipulou-se que cada objecto seria sempre representado por um número de pontos nunca superior a 120 e as distâncias poderiam variar de objecto para objecto. Ou seja é estipulado uma distância mínima de referência entre pontos de 5 píxeis e encontrada uma nova matriz de pontos, posto isto é verificado a condição de 120 pontos. Caso a condição não seja satisfeita é incrementado mais um milímetro à distância de mínima de referência e recalculada uma nova matriz de pontos, o processo é repetido sucessivamente até a condição ser satisfeita.

5.4 Tipos de Movimentos do Robô

Para movimentar o robô, a sua programação disponibiliza quatro formas diferentes, Movimentos Lineares (MOVL), Movimentos Circulares (MOVC), Movimentos *Spline* (MOVS) e Movimentos de Junta (MOVJ).

Os **Movimentos Lineares** correspondem a trajectórias rectas, que unem o ponto em que o robô se encontra e o ponto seguinte a atingir. Sendo o "caminho" mais curto[4] a percorrer pelo robô. Na figura 5.4 é ilustrado um movimento deste tipo no deslocamento do ponto P3 para o ponto P4.

Os **Movimentos Circulares** são movimentos em arco, em que a extremidade livre da ferramenta do robô se desloca segundo uma trajectória que tem a forma de um arco bem definido. Neste tipo de movimento são necessários três pontos no mínimo para ser possível aplicar o método matemático de interpolação circular, na aplicação do método e com base nos pontos fornecidos é estabelecido se o arco é côncavo ou convexo.

Um inconveniente desta função é o facto de no caso dos três pontos estarem alinhados o método de interpolação circular não pode ser aplicado e ocorre um erro, pelo que quando se utiliza esta função tem que se garantir que os três pontos correspondentes a esta instrução não estão segundo a mesma recta. Outra imposição desta função é a distância entre os dois primeiros pontos, esta deve ser a mesma que a distância entre os dois últimos. Na figura 5.3 nos pontos P1, P2 e P3 são utilizados movimentos circulares, como se pode verificar os movimentos que descrevem arcos são apenas os dois últimos, sendo o primeiro necessário ao cálculo no entanto, realiza um movimento rectilíneo. Isto verifica-se caso o robô esteja e iniciar o seu movimento ou seja precedido de um movimento linear ou de junta. Caso existam mais de três deslocamentos adjacentes onde se utilizem movimentos circulares, esses movimentos descreverão todos semi-arcos.

[4] "Caminho mais curto" neste caso representa a distância mínima percorrida pela extremidade livre da ferramenta e não a rotação das juntas do robô.

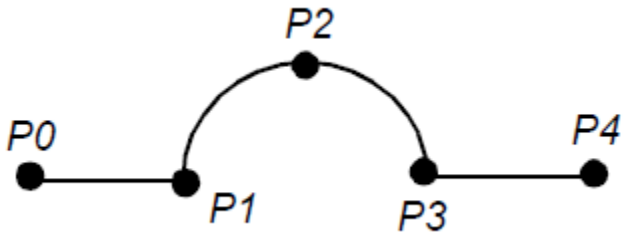


Figura 5.4: Movimento Circular nos pontos P1–P2–P3 e Movimento Linear para P4.

Movimentos *Spline* são movimentos nos quais é aplicado a interpolação polinomial de segundo grau para calcular a sua trajectória. Esta descreve uma parábola e como no movimento circular, neste caso também são necessários três pontos no mínimo para aplicar este método. Neste caso não há o problema dos pontos estarem alinhados uma vez que caso isto ocorra o movimento será rectilíneo. No entanto, a limitação da distância entre pontos neste caso também é igualmente válida. Na figura 5.5 são utilizados movimentos *Spline* para movimentar o robô para os pontos P1, P2,P3, P4 e P5.

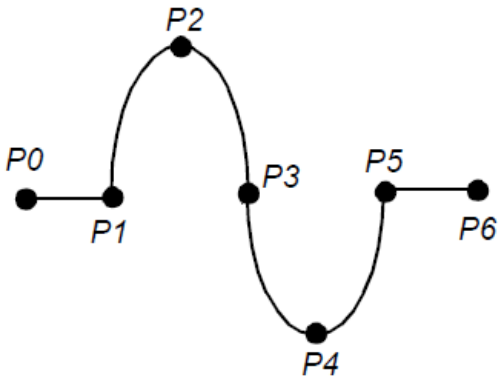


Figura 5.5: Movimento *Spline* nos pontos P1–P2–P3–P4–P5.

Movimentos de Junta são os movimentos que minimizam a rotação das juntas do robô. Neste movimento muitas vezes o utilizador não tem a noção de qual será a trajectória da extremidade livre da ferramenta do robô, de modo que este tipo de movimento é utilizado essencialmente em movimentações de aproximação.

5.5 Estudo do Tipo de Movimento a Utilizar

Nos movimentos de aproximação (movimento inicial) e afastamento (movimento final) são utilizados Movimentos de Junta uma vez serem os movimentos mais rápidos e mais fáceis de realizar pelo robô. Pois estes não são afectados pelos limites de rotação das juntas do robô no espaço de trabalho deste, dado que o robô tem em conta o ângulo de rotação que é possível realizar em cada junta.

Nos movimentos onde se pretende percorrer o contorno da peça pode-se utilizar Movimentos Lineares, Movimentos Circulares e Movimentos *Spline*.

Na figura 5.6 apresenta-se o deslocamento do robô percorrendo vários pontos utilizando exclusivamente Movimentos Lineares. Estes pontos foram escolhidos tendo em conta as geometrias das peças que se poderão encontrar na prática. Como se pode ver e como já foi referido em cima os pontos são percorridos segundo segmentos de recta. Uma vez os pontos definirem um contorno bastante arredondado faz se notar as quinas vivas devido à trajectória ser pouco flexível. Esta seria uma boa solução caso os pontos definissem um contorno mais linear.

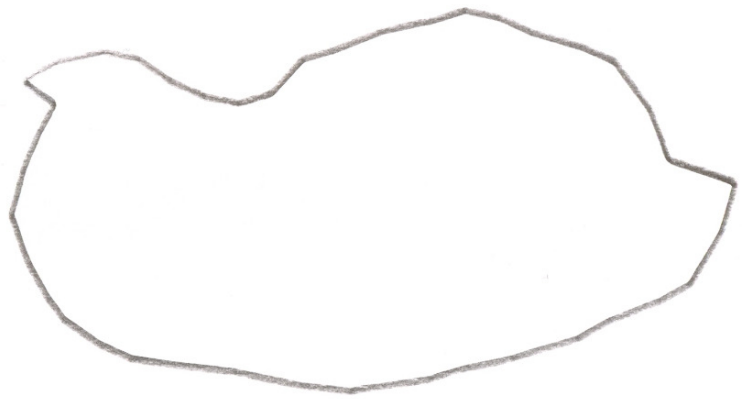


Figura 5.6: Contorno efectuado com Movimentos Lineares.

Na figura 5.7 apresenta-se o contorno utilizando os mesmos pontos que no exemplo anterior mas neste caso utilizando Movimentos Circulares. Como se pode ver o contorno é percorrido de uma forma bastante mais suave em relação ao anterior, eliminando as quinas vivas na peça.

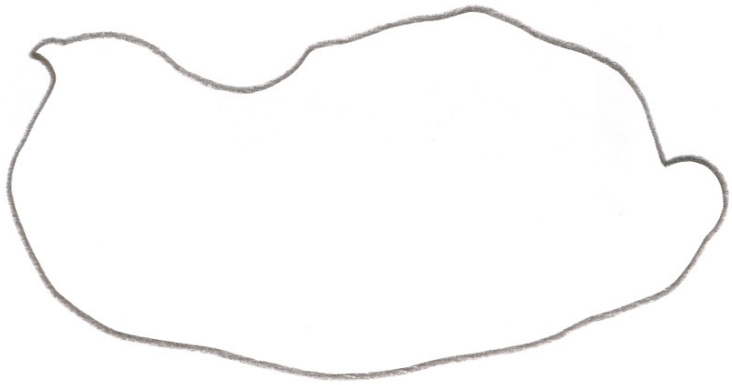


Figura 5.7: Contorno efectuado com Movimentos Circulares.

Na figura 5.8 apresenta-se o contorno utilizando os mesmos pontos que nos exemplos anteriores mas neste caso utilizando Movimentos Spline. Como se pode ver o contorno é percorrido de uma forma ainda mais suave que no exemplo anterior, a eliminação das quinas vivas neste caso ainda é mais notória.

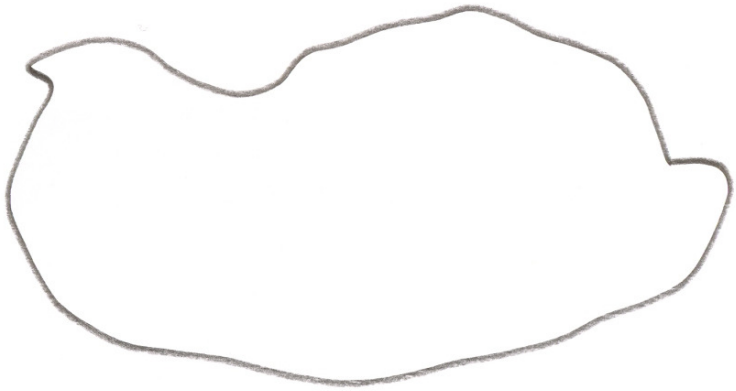


Figura 5.8: Contorno efectuado com Movimentos *Spline*.

Uma função que o robô disponibiliza e não deve ser desprezada é a utilização de Movimentos Lineares Suavizados. Esta função permite percorrer os pontos exactamente da mesma forma que nos Movimentos Lineares, no entanto quando se está na presença de uma quina viva por vezes o robô não chega a atingir o ponto descrevendo uma trajectória ligeiramente circular antes de o atingir. O robô utilizado disponibiliza quatro níveis de curvatura (PL) como se pode ver na figura 5.9.

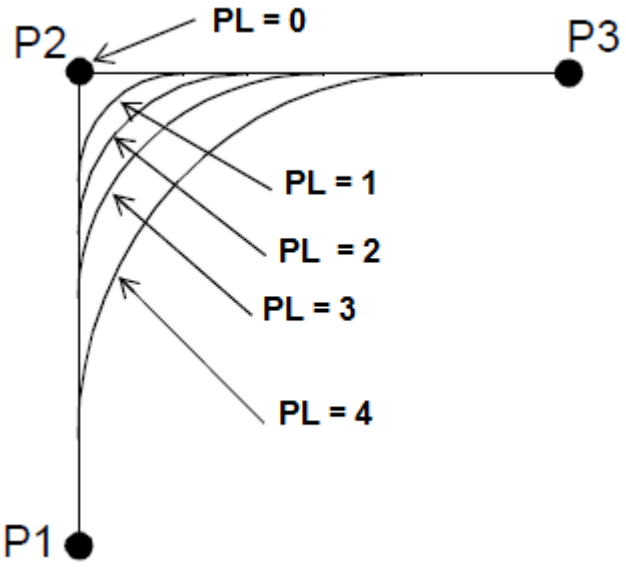


Figura 5.9: Níveis de Suavização do Movimento Linear.

Na figura 5.10 é apresentado uma imagem com três dos níveis de suavização permitidos no Movimento Linear. Como se pode verificar há uma certa melhoria na suavidade do movimento quando se utiliza suavização.

Devido aos pontos estarem demasiado próximos não se nota grande diferença entre os vários níveis de suavização. No entanto no âmbito deste trabalho os pontos a percorrer estarão sempre a distâncias relativamente próximas.

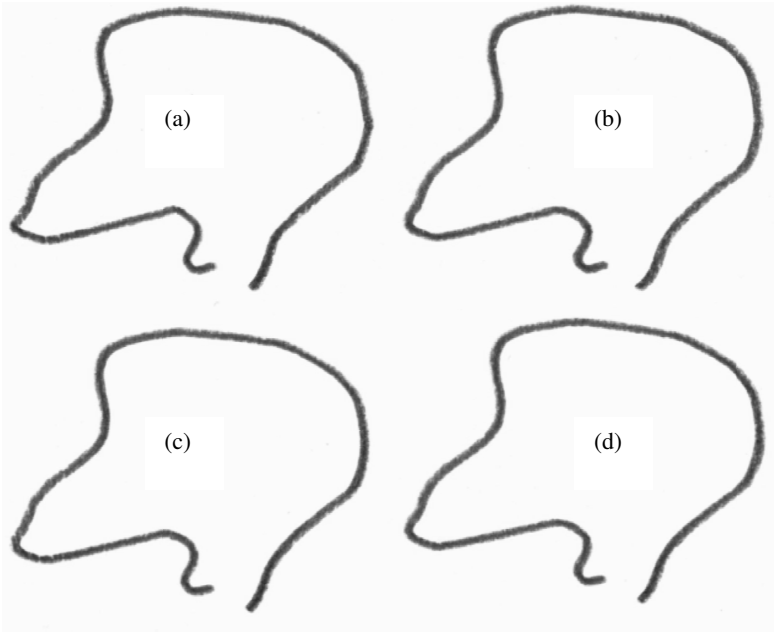


Figura 5.10: Contornos efectuados com Movimentos Lineares Suavizados (a) PL = 0; (b) PL = 1; (c) PL = 2; (d) PL = 3.

Apresentados os tipos de movimentos deve-se agora escolher o mais apropriado ao caso em questão. O Movimento Linear já está visível que não seria uma boa solução. O Movimento Circular e o Movimento *Spline* seriam boas soluções no entanto, dado as suas limitações não são utilizados. De referir, que estes dois movimentos poderiam ter sido utilizados, bastando para isso, por exemplo no caso do Movimento Circular garantir que cada conjunto de três pontos adjacentes não estariam alinhados segundo a mesma recta. Este problema seria facilmente resolvido fazendo a verificação (utilizando por exemplo a formulação do produto interno aplicado aos vectores $\overrightarrow{P_{i-1}P_i}$ e $\overrightarrow{P_{i+1}P_i}$ e analisando o ângulo gerado entre os dois vectores) e caso necessário gerar pontos ligeiramente desalinhados. O Movimento Linear Suavizado proporciona contornos suficientemente suaves e sem necessidade de efectuar cálculos para efectuar verificações ou gerar novos pontos.

5.6 Calibração do Robô

Para enviar as coordenadas dos pontos ao robô é necessário transformá-las em coordenadas conhecidas pelo robô, ou seja transformar as coordenadas no *referencial cena* num referencial conhecido pelo robô. O robô disponibiliza vários tipos de referenciais, no entanto para esta situação apenas serão utilizados dois tipos. Um dos referenciais utilizados é chamado ***referencial robot***, posicionado no centro da primeira junta do robô, ele está definido por defeito não podendo ser alterado. Este referencial é utilizado quando se envia ao robô "ordens" de deslocamento de forma remota. A calibração deste referencial é bastante simples, uma vez que o *referencial cena* tem as mesmas direcções que o *referencial robot* diferindo apenas na posição no espaço. Assim para realizar esta calibração basta incrementar aos pontos que resultam da calibração de imagem, as coordenadas do ponto de origem do *referencial cena*, expressas no *referencial robot*.

O outro tipo de referencial utilizado é o ***referencial user***, este é definido pelo utilizador não só em posição mas também em termos de orientação. Este referencial fez-se coincidir com o *referencial cena* e as coordenadas provenientes da calibração de imagem são directamente utilizadas no controlo do robô. Nesta aplicação o *referencial user* é utilizado como referencial nos *ficheiros JBI*, minimizando assim o número de cálculos necessários neste tipo de calibração.

CAPÍTULO SEIS

6 Funcionamento do Software Desenvolvido

Como já referido, o *software* desenvolvido tem dois tipos de funcionalidades, manipulação de objectos "*pick-and-place*" e reprodução de contornos "*Contour*". A primeira funcionalidade tem dois modos de operação, Manual e Automático. No modo Manual o operador escolhe um objecto presente na imagem, o robô apanha-o e coloca-o na respectiva posição de descarga. No modo Automático o robô apanha todos os objectos presentes na área de trabalho, um de cada vez. Os objectos são identificados através da sua área, contorno, cor e posição. A figura 6.1 apresenta o interface do *software* desenvolvido.

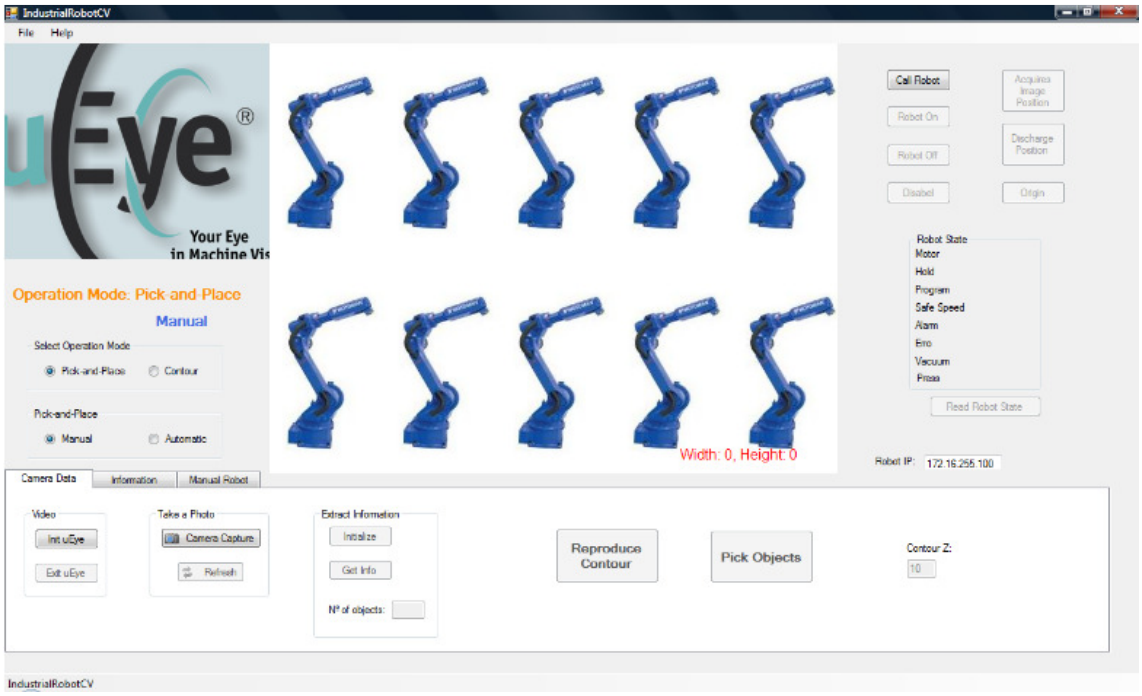


Figura 6.1: Interface do "*IndustrialRobotCV*".

Para escolher o tipo de operação basta clicar no nome da operação pretendida, caso se seleccione a operação de manipulação deve se escolher o modo de operação que é em tudo semelhante à selecção do tipo de operação, uma imagem destas funções ilustra-se na figura 6.2. Caso o operador não seleccione nenhuma das operações, o software assume por defeito operações de manipulação no modo manual. Consoante é escolhido o tipo e modo de operação o software apresenta uma mensagem informativa.

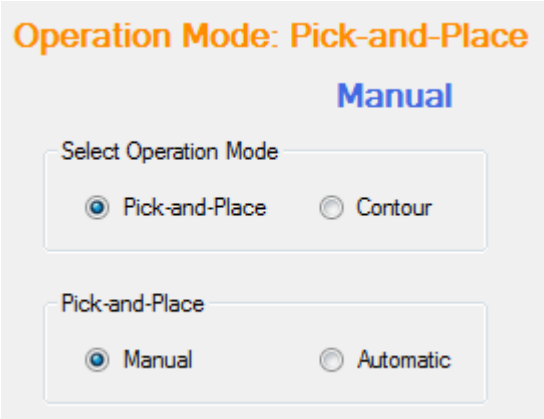


Figura 6.2: Escolha do Tipo e do Modo de Operação.

Para operar com este *software*, a primeira coisa que deve ser feita é seleccionar o tipo e o modo de operação, já apresentados. De seguida deve ser adquirida a imagem, para isso clica-se no botão *Camera Capture* e posteriormente no botão *Refresh*, apenas na primeira vez que se utiliza o *software* é necessário clicar no primeiro botão referido nas seguintes utilizações basta clicar no segundo botão referido. Na figura 6.3 são apresentados estes dois botões.

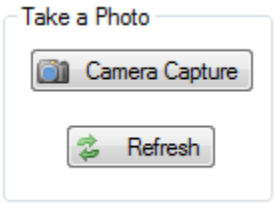


Figura 6.3: Botões de aquisição de imagem.

O passo seguinte é o processamento de imagem, e neste caso deve-se clicar no botão *Initialize*. O *software* reduz o *efeito-olho-de-peixe* presente na imagem (apresentado no capítulo quatro) e, o utilizador deve optimizar o contorno da imagem (janela Contours) como

apresentado no subcapítulo 3.4.1. Segue-se a aquisição de informação a partir da imagem e para isso basta clicar no botão *Get Info*. O *software* passa a informar o utilizador verbalmente acerca do número de objectos presente na imagem e também apresenta uma mensagem acerca do mesmo. Apresenta-se estas funções na figura 6.4.

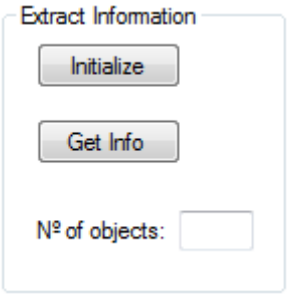


Figura 6.4: Extracção de informação da imagem.

Deve ser estabelecida a comunicação entre o computador utilizado e o robô, para isso clica-se no botão *Call Robot* e devem-se ligar os motores do robô, clicar no botão *Robot On*.

Consoante o tipo de operação que se pretende realizar nesta fase a utilização do *software* será distinta:

Para operar no modo de manipulação de objectos manualmente, este modo deve estar escolhido como já descrito anteriormente e, deve-se clicar em cima do objecto presente na imagem o mais próximo do seu centro de massa quanto possível. O robô procederá a apanhar o objecto e a coloca-lo na posição de descarga, além de actualizar o número e as características dos objectos presentes na área de trabalho.

Para operar no modo automático basta clicar no botão *Pick Objects* e até que existirem objectos na área de trabalho o robô apanhará todos os objectos, um por um.

No modo de reprodução de contornos o utilizador deve clicar no botão *Reproduce Contour* e a operação será efectuada. Quando o robô termina esta última funcionalidade posiciona-se por cima do plano de trabalho (onde efectua a operação), preparado para repetir a mesma operação. De modo que, caso se pretenda realizar outra operação ou reproduzir o contorno de um objecto diferente, ter-se-á de clicar no botão *Acquires Image Position* para que o robô se desloque para a posição de aquisição de imagem. Seguindo-se a aquisição de uma nova imagem e de todo e o processo já descrito em cima.

Para reproduzir os contornos utilizou-se uma ferramenta com um lápis, como facilmente se entende o lápis é uma ferramenta que está sujeito a um elevado desgaste de modo, que é de todo o interesse ajustar a ferramenta à área de trabalho de uma forma simples, rápida e eficiente. Com vista a realizar esta tarefa surge um campo onde o utilizador insere um valor de modo a ajustar o lápis ao plano de trabalho. Este valor está limitado ao intervalo [-10 ; 10], para valores diferentes o lápis deve ser ajustado no suporte.

Outras funcionalidades do *software*:

Botão *Init uEye* apresenta a imagem vídeo da câmara no ecrã;

Botão *Exit uEye* desliga a imagem vídeo da câmara;

Botão *Discharge Position* desloca o robô para a posição de descarga;

Botão *Origin* desloca o robô para a origem do referencial cena (utilizar apenas quando tem a ventosa como ferramenta);

Botão *Robot Off* desliga os motores do robô;

Botão *Disabel* termina a comunicação entre o computador e o robô;

Botão *Press On* ligar a pressão;

Botão *Press Off* desligar a pressão;

Botão *Vaccuum On* liga o vácuo;

Botão *Vaccuum Off* desliga o vácuo;

Botão *Stop Robot* pára o robô em caso de emergência;

Botão *Run Robot* retira o robô do estado de emergência;

Botão *Continue JOB* retoma a realização de um programa (ficheiro JBI) que o robô estava a realizar antes de ser colocado no estado de emergência.

Estas funcionalidades são ilustradas na figura 6.5.

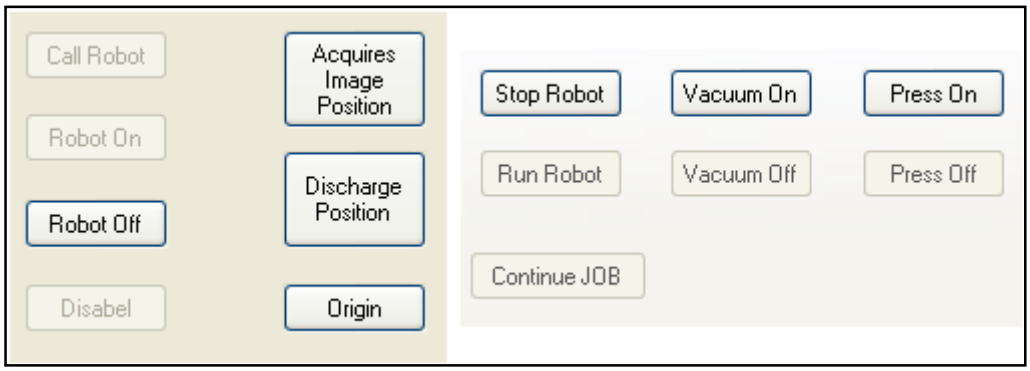


Figura 6.5: Algumas funções de comando do robô.

O *software* dispõe de duas funcionalidades para movimentar o robô livremente, uma onde se introduz as coordenadas do ponto para o qual se pretende deslocar o robô e outra que movimenta o robô por incrementação segundo os seus eixos. Para utilizar a primeira basta colocar as coordenadas do ponto nas caixas de texto, que se apresentam na figura 6.6 do lado esquerdo, e clicar no botão *Go To*. Para utilizar a movimentação por incrementação deve-se seleccionar o valor que se pretende incrementar e de seguida clicar no botão que corresponde à direcção e ao sentido do eixo que se pretende efectuar o movimento. A velocidade tanto numa como na outra é seleccionada pelo utilizador numa barra de deslocamento que está limitada ao intervalo [0 ; 20] [mm/s].

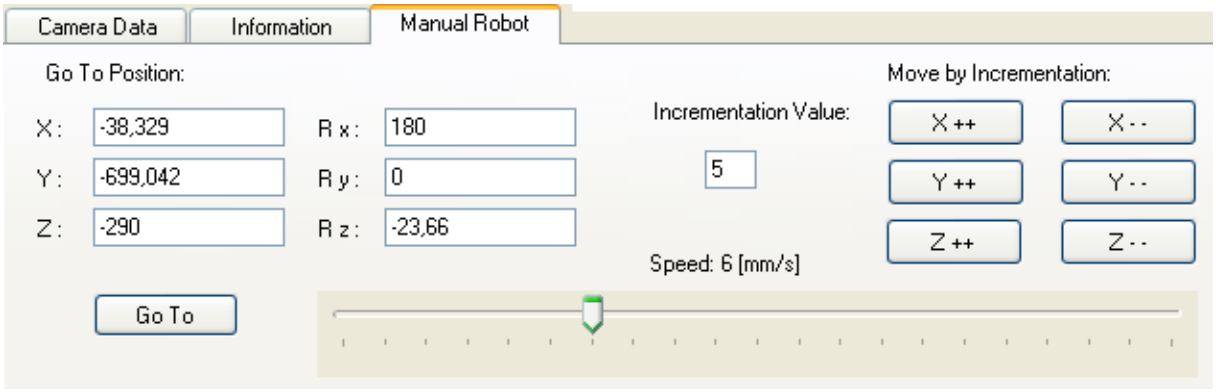
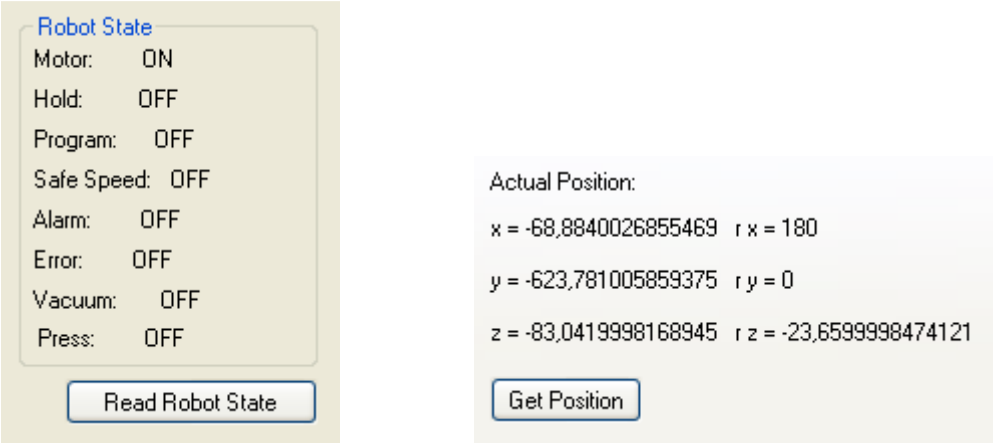


Figura 6.6: Painel de comandos directos do robô.

O *software* demonstra ainda ao utilizador os estados do robô e a posição em que ele se encontra, como se apresenta na figura 6.8, estas informações vão sendo ao longo do programa actualizadas no entanto, estão disponíveis dois botões que permitem ao utilizador actualiza-las em qualquer momento, são eles o botão *Read Robot State* e o botão *Get Position* respectivamente.



(a) (b)

Figura 6.7: Informação presente no *software* sobre: a) Estados do robô; b)Posição em que o robô se encontra.

Outras informações que são também apresentadas quando se utiliza o *software* no modo de manipulação de objectos são: as coordenadas dos objectos na imagem (1), a sua área (1), a sua cor (1), as coordenadas na imagem seleccionadas pelo utilizador (2), a distância dessas coordenadas aos centros de massa de cada um dos objectos (1) e o objecto seleccionado que o robô está a processar no momento (3), ver figura 6.8.

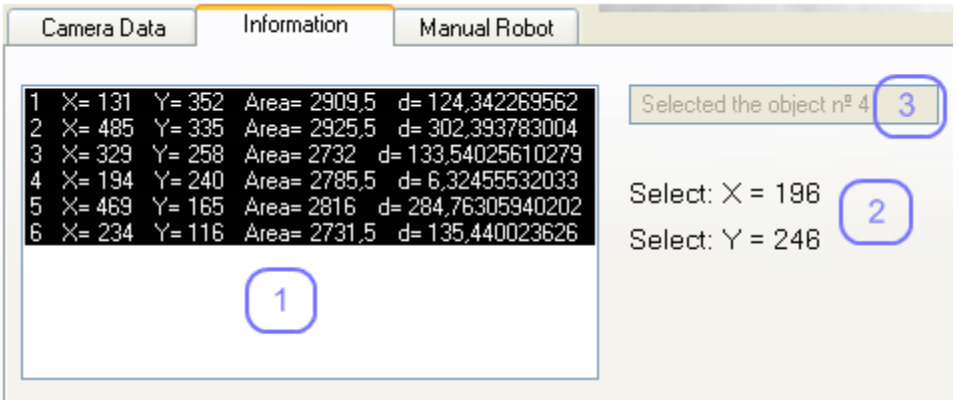


Figura 6.8: Informação sobre a operação de Manipulação de Objectos.

No modo *Contour* aparecem ainda, as coordenadas de todos os pontos que identificam o contorno do objecto no referencial imagem (1) e as mesmas coordenadas no *referencial user* (2) e ainda o número de pontos que define o contorno (3), ver figura 6.9. Neste modo na apresentação da imagem aparece a verde o contorno que o robô deve realizar, representado na figura 6.10.

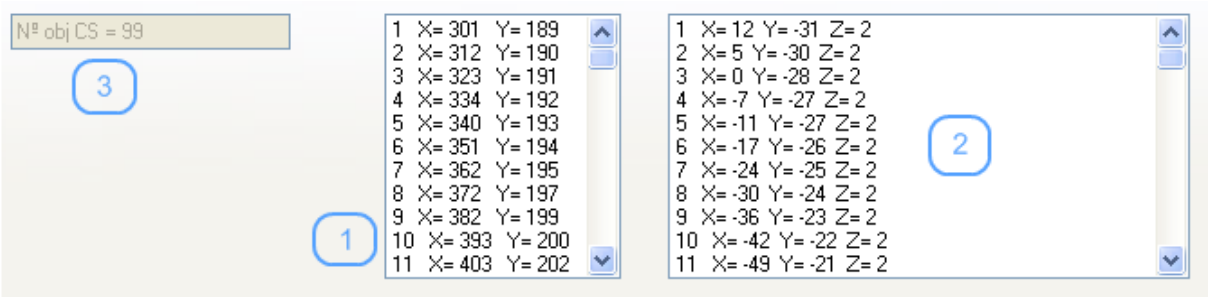


Figura 6.9: Informação sobre a operação de Reprodução de Contornos.

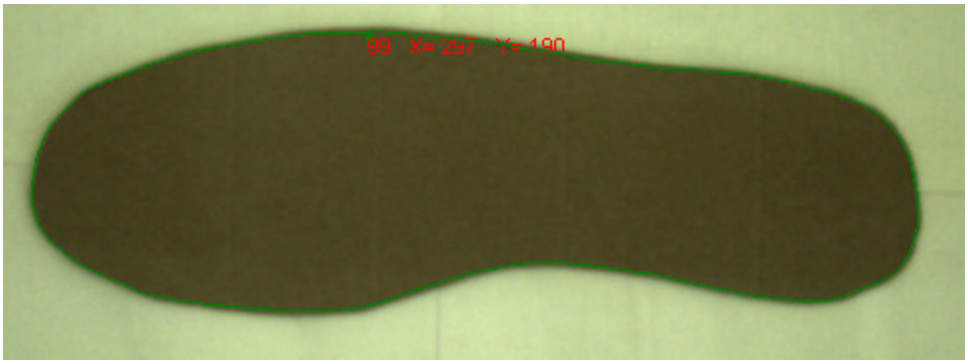


Figura 6.10: Representação do contorno (a verde) a realizar pelo robô.

Um outro campo que pode ser de bastante utilidade é o IP do robô, este, caso necessário pode facilmente ser alterado como se demonstra na figura 6.11.

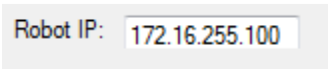


Figura 6.11: Campo onde se pode alterar o IP do robô.

CAPÍTULO SETE

7 Conclusões

Ao longo deste documento foram feitas algumas escolhas e demonstrados alguns resultados que são merecedores de alguma discussão e de mais alguns comentários argumentativos. Pelo que se passa a enumerar:

Desde logo optou-se por um sistema de visão monocular *eye-in-hand*, outras soluções seriam igualmente viáveis, no entanto esta foi a solução que satisfez as necessidades da aplicação desenvolvida ao mais baixo custo de investimento e que proporcionou um tempo de ciclo mais baixo. Isto deve-se ao facto de utilizar menos *hardware* e de não processar informação desnecessária. É verídico que a informação tridimensional é estimada em parâmetros que pouco têm a ver com essa informação, no entanto, a complexidade de *software* e *hardware* necessários para extrair directamente essa informação incrementaria de uma forma negativa alguns factores. Para além dos já referidos, o tempo de desenvolvimento de *software* (com algoritmos altamente complexos) e o tempo de operação do *software* num ciclo são alguns parâmetros de "peso" que desfavorecem a utilização de outro método.

A utilização de um sistema *eye-in-hand* pode ser alvo de algum criticismo, uma vez que a aplicação implementada não está dotada directamente de nenhuma vantagem em relação a um sistema *eye-to-hand*. No entanto, acredita-se que a vantagem de facilmente se poder utilizar uma área de trabalho diferente, atenue os factores negativos que este sistema acarreta. Uma situação que pode por em causa a fiabilidade deste sistema, e que deve ser alvo de uma análise a realizar no futuro, é a incorporação de uma fresa como ferramenta no robô. Esta situação poderá introduzir vibração no sistema que o desfavoreça consideravelmente.

A utilização de um sistema *eye-to-hand* teria a vantagem de ser possível utilizar uma lente com uma ampliação inferior, o que levaria à introdução de menos distorção na imagem, uma vez que existe uma relação entre o preço da lente e a sua qualidade. No entanto teria sempre que existir uma fase de redução de distorção, dado que lentes ideais hoje em dia ainda

são uma "miragem". A fase de redução do *efeito de olho de peixe* está bastante funcional, uma vez se ter utilizado uma ferramenta (*toolbox*) já testada em outras aplicações nas quais foi considerada uma óptima ferramenta. Com ela foi possível reconstruir a imagem da cena e assim reduzir radicalmente a distorção presente na imagem. Resultando após calibração da precisão do robô um erro de aproximadamente zero milímetros no centro da imagem e de dois milímetros na extremidade da imagem, onde se verificava a maior distorção.

O método de calibração utilizando o cálculo de regressões, para a aplicação em causa, efectivamente não apresentava os resultados exigidos ao bom funcionamento do sistema, um erro de nove milímetros na situação mais desfavorável é excessivo. De realçar que este método não foi alvo de teste utilizando a correcção da precisão do robô, esse facto poderia ter melhorado um pouco os resultados. Um outro aspecto que poderia ter melhorado o desempenho do método recai sobre a escolha do ponto de referência para o referencial imagem nesta fase (ponto superior esquerdo) e da orientação escolhida para a câmara. De facto, grande parte da imagem continha uma certa inclinação o que dificultou o processo de calibração e os resultados associados. Acredita-se que utilizando o ponto central da imagem como referência e orientando a câmara na direcção horizontal e vertical que passam nesse ponto, tendo em conta o papel xadrez na área de trabalho, é possível melhorar os resultados obtidos de uma forma bastante favorável. No entanto, não esquecer que este método peca pela sua falta de versatilidade, pelo que se não se pensar em desenvolver um algoritmo de elevada complexidade de modo a automatizar o processo, este método estará condenado logo à partida.

O sistema desenvolvido embora ainda não tenha sido testado a nível industrial, em laboratório apresenta um bom desempenho, funcionando a qualquer hora do dia sem que a luz natural e artificial causem um forte impacto, apenas em objectos de altura considerável o efeito sombra incute influência no sistema. No entanto, ao longo do dia devido à diferença de luminosidade na área de trabalho é necessário proceder à correcção do parâmetro brilho.

Este trabalho apresenta-se como o ponto de partida para trabalhos futuros no melhoramento do processamento de imagem no que diz respeito à identificação dos contornos dos objectos, na identificação e rotação de contornos no plano e na criação de uma base de dados onde facilmente se identifique diferentes tipos de objectos numa "perspectiva" tridimensional, onde a orientação do robô varia para apanhar esses objectos.

Uma aplicação na indústria deste sistema será por exemplo na eliminação de rebarbas ou na suavização de quinas vivas em peças.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] **Pires, J. Norberto**, 2002. "Automação Industrial". LIDEL.

[2] **Bouguet, J.**, 1999. "Visual methods for three-dimensional modeling". PhD *Thesis.*

[3] **Gonçalves, P. J.**, 2005. "Controlo Visual de Robôs Manipuladores". *Dissertação para obtenção do grau de Doutor em Engenharia Mecânica.*

[4] **Faugeras, O.**, 1993. "Three dimensional computer vision, a geometric viewpoint". Cambridge, Massachusetts, USA, MIT Press.

[5] **Dâmaso, R.**, 2006. "Implementação de Controle Servo Visual e Coordenação Visuo-Motora em Robôs Manipuladores". *Dissertação para obtenção do grau de Doutor em Engenharia Eléctrica*.

[6] **Ferreira, A.**, 2007. "Extracção de Dimensões de Objectos por Laser para integração de Manipuladores Industriais". *Relatório submetido a obtenção do grau de Licenciado em Engenharia Electrotécnica e de Computadores*.

[7] **Mihara, I. / Yamauchi, Y.** & **Doi, M.**, 2003. "A Real-Time Vision-Based Interface Using Motion Processor and Applications to Robotics". in *Systems and Computers in Japan*, Vol. 34th, No. 3, Wiley Periodicals Inc.

[8] **Gao, J. / Xu, W.** & **Geng, J.**, 2006. "3D shape reconstruction of teeth by shadow speckle correlation method". in *Opt Lasers Eng*, Vol. 44th, No. 5, DOI, pp. 455-465.

[9] **Noruk, J.** & **Boillot, J.P.**, 2006. "Laser Vision Technology Ensures Six Sigma-level Quality is Achieved in Robotic Welding". in *Canadian Welding Association Journal – Summer 2006*, Canada, pp. 8-14.

[10] **Xei, S. Q. / Cheng, D. / Wong, S.** & **Haemmerle, E.**, 2007. "Three-dimensional object recognition system for enhancing the intelligence of KUKA robot". in *Int J Adv Manuf Technol*, Vol. 38th, London, Spriger-Verlag, pp. 822-839.

[11] **Giordana, N. / Bouthemy, P. / Chaumette, F.** & **Spindler, F.**, 2000. "Two-dimensional model-based tracking of complex shapes for visual servoing tasks". in M. Vincze & G. Hager, eds, *Robust vision for vision-based control of motion*, IEEE Press, pp. 67–77.

[12] **Hutchinson, S. /** Hager, G. & Corke P., 1996. "A tutorial on visual servo control". in *Transactions on Robotics and Automation*, Vol. 12th, No. 5, IEEE, pp. 651-670.

[13] **Hashimoto, K.**, 2003. "A review on vision-based control of robot manipulators". in *Advanced Robotics*, Vol. 17, No. 10, pp. 969-991.

[14] **Remazeilles, A. / Chaumette, F.** & **Gros, P.**, 2004. "Robot motion control from a visual memory". in *Proceedings of the IEEE International Conference on Robotics and Automation*, (New Orleans, U.S.A.), IEEE, pp. 4695-4700.

[15] **Shirai, Y.** & **Inoue, H.**, 1973. "Guiding a robot by visual feedback in assembly tasks". in *Pattern Recognition*, Vol. 5, pp. 99-108.

[16] **Trucco, E.** & **Verri, A.**, 1998. "Introductory techniques for 3-d computer vision". New Jersey, Prentice-Hall.

[17] **Pretlove, J.** & **Parker G.**, 1991 "The development of a real-time stereo-vision system to aid robot guidance in carrying out a typical manufacturing task". in *Proceedings of the Internation Symposium of Robotics Research, ISRR*, Vol. 22nd, Detroit, MI, pp. 21.1-21.23.

[18] **Albuquerque, M.** & **Albuquerque, M.**. "Processamento de Imagem: Métodos e Análises".

[19] **Horn, B.**, 1986. "Robot Vision". MIT Press.

[20] **Shapiro, L.** & **Stockman, G.**, 2001. "Computer vision". Prentice-Hall.

[21] **Sonka, M. / Hlavac, V.** & **Boyle, R.**, 1999. "Image processing, analysis and machine vision". 2nd edition, PWS Publishing.

[22] **Bouguet, J.**, (June 2nd,1998). "Camera calibration toolbox for matlab". (http://www.vision.caltech.edu/bouguetj/cali_doc/).

[23] **Bouguet, J. & Perona, P.**. "3D photography using shadows in dual-space geometry".

[24] **Bouguet, J.**, 1999. "Visual methods for three-dimensional modeling". PhD *Thesis.*

[25] **Meggiolaro, M. / Jaffe, P.** & **Dubowsky, S.**, 1999. "Achieving Fine Absolute Positioning Accuracy in Large Powerful Manipulators". in *International Conference on Robotic & Automation*(*ICRA*), (Detroit, Michigan), IEEE.

[26] **Brown, L. G.**, 1992. "A survey of image registration techniques". in ACM Computer Surv, Vol. 24th, No. 4, pp. 325-376.

[27] **Zitová, B.** & **Flusser**, **J.,** 2003. "Image registration methods: a survey". in *Image Vision Comput*, Vol. 21st, No. 11, Elsevier, pp. 977-1000.

[28] **Abdullah,M. / Bharmal, M. & Sardi, M.**, 2005. "High speed robot vision system with flexible end effector for handling and sorting of meat patties". in *9th International Conference on Mechatronics Technology*.

[29] **Pinheiro, A.**, 2007. "Análise e processamento de imagem".

[30] **Heikkilä, J. & Silvén O.**. "A Four-step Camera Calibration Procedure with Implicit Image Correction".

[31] **MOTOMAN**, 2004. "MOTOMAN NX 100 BASIC PROGRAMMING".

[32] **YASKAWA**, 2003. "NX 100 INSTRCTIONS, FOR RELATIVE JOB FUNCTION".

[33] **Karli, W. et al**, 2002. "Beginning Visual C#". Wrox Press Ltd.