Faculdade de Engenharia da Universidade do Porto

Pintura Robotizada Com Adaptação Automática

Marcos André Magalhães Ferreira

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Electrotécnica e de Computadores
Major Automação

Orientador: Prof. Dr. António Paulo Gomes Mendes Moreira

2009/07/29

U.PORTO

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

MIEEC - MESTRADO INTEGRADO EM ENGENHARIA ELECTROTÉCNICA E DE COMPUTADORES 2008/2009

A Dissertação intitulada

"PINTURA ROBOTIZADA COM ADAPTAÇÃO AUTOMÁTICA"

foi aprovada em provas realizadas em 24/Julho/2009

o júri

Presidente Professor Doutor António José de Pina Martins
Professor Auxiliar do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

Professor Doutor Joaquim Norberto Cardoso Pires da Silva
Professor Auxiliar do Departamento de Engenharia Mecânica da Faculdade de Ciências e Tecnologias da Universidade de Coimbra

Professor Doutor António Paulo Gomes Mendes Moreira
Professor Auxiliar do Departamento de Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia da Universidade do Porto

O autor declara que a presente dissertação (ou relatório de projecto) é da sua exclusiva autoria e foi escrita sem qualquer apoio externo não explicitamente autorizado. Os resultados, ideias, parágrafos, ou outros extractos tomados de ou inspirados em trabalhos de outros autores, e demais referências bibliográficas usadas, são correctamente citados.

Autor - MARCOS ANDRÉ MAGALHÃES FERREIRA

Faculdade de Engenharia da Universidade do Porto

Resumo

Os manipuladores industriais estão aptos a executar tarefas com um grau de precisão muito acima da capacidade humana, podendo fazê-lo em ambientes potencialmente perigosos para o ser humano e ainda repetir as tarefas vezes sem conta sem prejuízo da qualidade do trabalho. São, por este motivo, uma solução comum para linhas de produção em massa.

No entanto, a integração dos manipuladores sobretudo ao nível das pequenas e médias empresas é uma tarefa complicada cujos benefícios podem não superar os encargos adjacentes. Nas pequenas/médias linhas de produção lidam-se, frequentemente, com pequenas séries, obrigando a uma reconfiguração e reprogramação frequente dos sistemas robóticos. Os manipuladores industriais são, ainda, máquinas dotadas de pouca inteligência, cuja programação é complexa, demorada e só realizável por pessoal altamente especializado, implicando elevados custos para a reprogramação frequente.

No sentido de superar esta situação, neste trabalho é proposta uma abordagem de reprogramação automática de manipuladores industriais perante um cenário típico de uma linha de produção de uma empresa portuguesa, a Flupol; a tarefa em causa é a pintura de peças com variadas formas e tamanhos.

Recorrendo a tecnologias de visão artificial e triangulação com laser, o sistema desenvolvido adquire informação 3D das diversas peças que transitam na linha de produção, concebe reconstruções a três dimensões e, a partir destas, extrai informações quantitativas que permitem reconhecer o tipo de peça em questão e as respectivas dimensões.

O manipulador, dotado à partida de programas genéricos para tratar um determinado número de peças distintas, usa essa informação no sentido de escolher o programa que melhor se adequa a cada peça em particular, adaptando-o consoante as dimensões da mesma.

No seu todo, o sistema desenvolvido apresenta-se como uma solução de baixo custo, capaz de lidar automaticamente com a pintura de diferentes peças tendo-se usado o problema em específico da Flupol para efectuar testes e validação do sistema.

Abstract

Humans fall short to industrial manipulators in what concerns to high precision, high quality repeatability, speed and invulnerability to hazardous environments. Therefore, industrial robots come as a common choice for mass production lines.

Despite being the far best candidate to perform industrial tasks, integration in small and medium enterprises is not an easy task and associated costs may, frequently, overcome the overall benefice. Medium/small production lines tend to work on small series, making it necessary to constantly reprogram robots and other automated tasks. Industrial manipulators still lack intelligence and their programming methods are still hard and time consuming (mostly based on teach-and-repeat techniques), only at reach from highly experienced and trained operators, thus implying bigger costs to keep a constant reprogramming scheme.

To overcome this reality, a novel approach is proposed for an automatic reprogramming of industrial robots, in the scenario of a typical production line of a small portuguese enterprise, Flupol. The task here is to paint parts which may vary in shape and size, and may come into the production line at any time and any order.

Using an artificial vision based solution, with laser triangulation, our system builds 3D models of the pieces from the images and, from these models, some relevant features are extracted. Such features allow us to distinguish the shape of the piece and compute some basic dimensions of it, such as height, depth and width.

The industrial manipulator, given a set of generic programs for painting some different shapes, will use that information and choose the best program that fits a particular piece. Furthermore, it shall adapt the code based on the estimated dimensions. This way, different parts get painted without the need of reprogramming the robot.

The developed system represents a low cost solution that is able to automatically paint distinct objects. For testing and validation purposes, the solution was adapted to match Flupol's problem.

Agradecimentos

Agradeço ao Prof. Dr. António Paulo Moreira, orientador nesta dissertação, toda a disponibilidade e ajuda prestada. Agradeço ainda o facto de, com este trabalho, me ter apresentado a áreas da robótica com as quais havia tido pouco ou nenhum contacto até à data e que se tornaram, rapidamente, numa área de eleição para mim e onde terei todo o gosto de continuar a trabalhar se possível.

Ao Prof. Dr. Paulo Costa, pelos diversos conselhos ao longo de todo este trabalho, contribuindo com ideias fundamentais para o desenrolar do projecto.

Um especial obrigado ao Paulo Malheiros, que em fase de conclusão do doutoramento e com todo o trabalho que tinha para fazer, encontrou ainda maneira de me dispensar várias horas de preciosa ajuda. Dicas, tutoriais e mais que isso, uma presença amiga constante.

Uma palavra de agradecimento, também, ao Alexandre Castro, técnico do laboratório, pela ajuda imprescindível na montagem de toda a estrutura que serviu de base para o trabalho.

Aos meus colegas, não só pelo apoio durante este projecto mas pela presença ao longo de toda uma vida académica.

Por último mas bem no topo, aos meus pais, não há um motivo só, há um tudo e um sempre.

Índice

Lista de figuras

Capítulo 1

Introdução

1.1 Descrição do Problema e Objectivos

O trabalho descrito neste documento surgiu de um projecto real, de aplicação industrial, proposto pela Flupol.

A Flupol é uma empresa especializada na aplicação de revestimentos anti-aderentes, anti-corrosão e auto-lubrificantes. A área de aplicação das soluções que tem disponíveis é vasta, desde formas para a indústria alimentar, acessórios para automóveis, utensílios domésticos, etc. Esta versatilidade leva à existência de um número elevado de peças distintas que, diariamente, entram na linha de produção a fim de lhes serem aplicados os revestimentos.

Actualmente, todo o processo é desempenhado por um operador humano que, numa estação ao longo da linha, pinta cada peça da forma mais adequada ao seu formato. Esta actividade é altamente repetitiva além do facto não desprezável de que várias tintas de revestimento contêm substâncias nocivas ao organismo obrigando o operário ao uso de uma máscara, expondo-se, de qualquer modo, a um ambiente hostil.

Face a estas características, acrescidas do pormenor de que o operário necessita parar a linha para pintar cada peça e cada novo operário precisar de um período de formação relativamente elevado, levou a que a Flupol tenha procurado uma solução mais inovadora, eficaz e ainda financeiramente vantajosa. O objectivo consiste em substituir o operador por um manipulador industrial para cumprir a tarefa de pintura (não na totalidade das peças mas de grande parte delas): o robô trabalha ininterruptamente (eliminando-se a necessidade de vários operadores a trabalhar por turnos), repete as suas funções com uma precisão constante (superando os operadores em termos de cansaço físico, predisposição para trabalhar, falhas por distracções pontuais, …), existem manipuladores próprios para pintura que não necessitam protecção especial contra as tintas tóxicas e consegue pintar as peças em acto contínuo sem interromper a marcha da linha de produção.

O verdadeiro entrave a esta aplicação reside, exactamente, na necessidade de pintar várias formas distintas e, dentro das formas idênticas, passíveis de ter uma dimensão variável. Perante o que é ainda hoje a realidade de programação dos manipuladores robóticos, esta solução passaria, classicamente, por agrupar as peças iguais e forçar a ordem na linha de

produção — impondo a pintura em série de um determinado tipo de objecto. Mais ainda, estar-se-ia completamente dependente de desenvolver um código/programa específico para cada tipo e tamanho de peça, obrigando a que existisse no local alguém suficientemente qualificado para executar essa tarefa e reprogramar o robô entre cada série. Ora, o *downtime* associado a estas reprogramações, a necessidade de reorganizar as peças no início da linha assim como a necessidade de manter pessoal altamente especializado no local tornam este cenário muito pouco atractivo, tanto logística como economicamente.

No sentido de contornar esta realidade procura-se, neste projecto, alcançar um sistema de pintura robotizada que, partindo de programas genéricos, consiga realizar a pintura de peças distintas, sem qualquer necessidade de reprogramação.

Especificamente, pretende-se dotar a linha de produção com um sector onde, recorrendo a tecnologias de visão e análise 3D, se consiga proceder ao reconhecimento automático das peças, tanto da forma como das dimensões, dentro de um conjunto de formas pré-estabelecido. A informação é tratada de tal modo que possibilita ao manipulador escolher qual o programa genérico que deve usar e de que forma o deve adaptar para corresponder com a peça que deve pintar. Desta forma, extingue-se a necessidade de reprogramação frequente.

São objectivos fundamentais o delineamento da arquitectura do sistema, estudo da melhor configuração do sistema de visão, dotar o sistema da capacidade de distinção entre peças e ainda desenvolver o método de adaptação dos programas do manipulador robótico. Serão usados como amostra um pequeno número de formas diferentes para as quais se espera que o sistema consiga efectuar a detecção, distinção e caracterização respectiva. A validação do trabalho passará por se cumprir um ciclo completo do processo, isto é, recolha de dados da peça por visão, reconhecimento e pintura com o manipulador. A solução final deverá ser altamente modular, permitindo a reconfiguração espacial dos elementos (por exemplo da câmara), a adição de mais programas genéricos e mecanismos de adaptação, e ainda a capacidade de reconhecer mais tipos diferentes de peças sem prejuízo do trabalho desenvolvido.

1.2 Estado da Arte

O uso de tecnologias baseadas em visão artificial com triangulação de laser tem registado uma aderência crescente nos últimos anos. Estes sistemas são particularmente atractivos devido à capacidade de criarem reconstruções 3D com elevada precisão, capazes de trabalhar a alta velocidade e ainda dispensarem o contacto físico com os objectos.

Existe, pois, já um extenso trabalho de investigação documentado em torno desta ciência. Em [1], por exemplo, os autores apresentam um método de triangulação com laser que, com base em planos ortogonais bem definidos no cenário (paredes, caixas rectangulares, …), o sistema auto calibra-se determinando também os parâmetros intrínsecos da câmara. Na falta de elementos que consigam garantir a calibração, sugerem ainda o uso de dois lasers de linha, ortogonais entre si. David Costa, et al [2] descrevem todo um projecto envolvendo a mesma tecnologia, com aplicação à reconstrução facial, salientando os aspectos de baixo custo e facilidade de implementação que podem ser conseguidos.

Com outras aplicações, já orientadas à extracção de informação a partir de reconstruções 3D com laser, temos, entre outros, o trabalho realizado em [3] para medição do diâmetro de objectos esféricos (neste estudo era imperativo que não existisse qualquer contacto com os

objectos para efectuar a medição) e o sistema apresentado em [4], para controlo de qualidade em tempo real em processos de soldadura.

No entanto, a solução de triangulação regista alguns pontos fracos, sendo um dos de maior relevância, o facto de que nem sempre se conseguem obter representações 3D completas de objectos devido à posição relativa do laser em relação aos mesmos. Para colmatar estas falhas, têm sido sugeridos *setups* com mais do que um laser [5] que permitem, ao mesmo tempo, um aumento da precisão; de um modo semelhante, em [6], os autores propõem um sistema de reconstrução por fusão dos dados de um conjunto de câmara e laser com outro conjunto de 2 câmaras formando um par estereoscópico.

Alternativamente, também se propõem sistemas alheios ao uso de laser. Entre estes, destacam-se as soluções baseadas somente em visão estereoscópica [7] ou em sensores de medida de *time-of-flight* [8][9] cujo princípio básico de funcionamento assenta no cálculo do tempo de propagação entre as ondas emitidas e reflectidas.

Já ao nível da indústria, a solução da câmara com triangulação com laser é cada vez mais empregue, principalmente com a finalidade de inspecção automática e testes de qualidade. No entanto, estas aplicações baseiam-se, geralmente, no princípio de *matching*, de forma a detectar erros de fabrico por comparação com modelos conhecidos e sabendo, a cada momento, quais os objectos que se está a analisar. Neste contexto, o trabalho aqui proposto, tem um carácter inovador já que se pretende que seja o sistema a descobrir qual o tipo de peça com que está a lidar, dentro de um conjunto limitado, e poder adaptar de modo automático as suas funções correspondentemente. A análise, em tempo real, de imagens 3D assim como a extracção de características sem ter por base um modelo de comparação, ainda tem aplicação reduzida; é, maioritariamente, na área da medicina que se verifica um forte tratamento de imagens 3D embora tratando-se, normalmente, de processamento *offline* não automatizado.

Foram consultadas soluções existentes no mercado mas não possuíam todas as funcionalidades pretendidas e, simultaneamente, os preços eram economicamente muito pouco atractivos. O facto de estes sistemas servirem para um elevado número de aplicações sobrequalifica-os de tal modo que, comparando com uma solução personalizada, os preços acabam por ser bastante elevados. Como exemplo, num trabalho recente sobre extracção de características e cálculo de posição [10], com este mesmo princípio de triangulação com laser, podemos encontrar um *setup* personalizado, que apresenta um custo muito baixo em comparação com todos aqueles apresentados no mercado.

Empresas como a Perceptron, 3D-Digital e 3D-Shape, apresentam também soluções pré-concebidas para reconstrução 3D, com exemplos práticos na recolha de modelos de peças de arte, reconhecimento facial, *reverse engineering,* etc. À semelhança dos sistemas atrás referidos, também estas soluções são dispendiosas, graças ao vasto leque de funcionalidades que apresentam, o nos obrigaria a pagar por funcionalidades que não nos servem nenhum propósito ao mesmo tempo que também falham funcionalidades que necessitamos.

1.3 Estrutura do Documento

Este documento encontra-se dividido em 6 capítulos que pretendem descrever o trabalho realizado ao longo da dissertação do Mestrado Integrado em Engenharia Electrotécnica e Computadores do autor.

Ao capítulo 1 reserva-se a componente introdutória. É apresentada uma descrição detalhada do problema que se pretende resolver ao longo da dissertação, com enunciação de objectivos e restrições para o sucesso da mesma. Em sequência, desenvolve-se uma caracterização do estado da arte e o estudo económico relativamente às possíveis soluções e tecnologias a empregar. Encerra-se este capítulo com esta mesma descrição da estrutura do documento.

O capítulo 2 sumaria os aspectos essenciais da solução proposta para o problema em questão. Apresenta uma descrição da arquitectura desta solução, referindo-se às tecnologias empregues em cada subsistema desenvolvido, factores de escolha e detalhe das características fundamentais. Descreve-se, igualmente, a montagem das componente mecânicas e a estrutura física que permitiu recriar no laboratório o ambiente industrial, necessário à execução de testes e validação do trabalho.

Os capítulos 3 a 5 associam-se a cada um dos subsistemas desenvolvidos que, no seu todo, formam a solução final. Pretendeu-se que cada um dos módulos fosse o mais *standalone* possível, permitindo a integração individual noutros sistemas caso necessário; deste modo, cada um destes capítulos pode ter consultado praticamente sem conhecimento dos precedentes.

O capítulo 3 trata do sistema de visão artificial e laser. Começa por delinear os fundamentos para a calibração da câmara — passo imperativo em qualquer sistema de visão — descrevendo depois o método de triangulação com o laser e os passos associados ao processamento de imagem que permitem a recolha de imagens, pesquisa de linhas e transformação de coordenadas entre diferentes referenciais. Termina mostrando reconstruções 3D das cenas e comentando os resultados obtidos nesta fase.

No capítulo 4 descreve-se a metodologia usada para identificar e distinguir peças na linha de produção. Define-se um método de reorganização de pontos 3D em estruturas de dados que facilitam o uso de algoritmos de pesquisa e propõem-se alguns indicadores que permitem distinguir as formas das peças. Conclui-se apresentando os resultados experimentais obtidos.

Resta do trabalho desenvolvido, para o capítulo 5, o último subsistema que visa a pintura das peças recorrendo a um manipulador industrial. Neste capítulo descrevem-se os programas genéricos implementados, o método de adaptação dos programas a diferentes dimensões de peças assim como o protocolo de comunicação usado entre a aplicação desenvolvida e o controlador do manipulador. Ainda é apresentada a técnica de sincronização do robô com o *conveyor,* que permite a pintura dos objectos sem necessidade de paragem da linha.

O capítulo 6 apresenta as conclusões relativas ao trabalho realizado, comentários aos resultados obtidos e ainda sugestões para trabalho futuro.

Por fim, enumeram-se as diversas referências a que se recorreu no desenvolvimento deste projecto.

Capítulo 2

Solução Proposta: Arquitectura e Tecnologias

Para resolver o problema apresentado na secção 1.1, decidiu-se usar um sistema de visão artificial composto por uma câmara e, de forma a obter informação tridimensional sobre as peças, um laser de linha que permite, por triangulação, complementar a informação da câmara com dados sobre a profundidade dos objectos.

O resultado deste processo é tratado por uma série de rotinas de software que extraem características dos objectos que permitem distinguir entre as várias formas que podem ser pintadas na linha de produção.

Complementando essa informação, são calculadas a posição, orientação e dimensões da peça, parâmetros estes que são enviados para um manipulador industrial.

Sabendo quais as peças que é necessário distinguir e que estas podem aparecer sobre diversos tamanhos, o manipulador é carregado, *a priori*, com programas genéricos adequados à pintura de cada forma. Estes programas base estão parametrizados em função do tamanho da peça a que se referem, isto é, o utilizador pode configurar remotamente vários parâmetros para que o programa se adapte às dimensões variáveis das peças.

Neste capítulo encontra-se a descrição da arquitectura do sistema assim como os detalhes sobre o equipamento usado. Reserva-se, também, uma secção para a descrição da estrutura mecânica que tornou possível reproduzir, em ambiente de laboratório, a linha de produção da Flupol.

2.1 Arquitectura e Componentes do Sistema

O sistema desenvolvido encontra-se estruturado conforme se apresenta na Figura 2.1. Para além das componentes de visão e laser, do reconhecimento de peças e da comunicação com o manipulador, temos ainda um subsistema desenvolvido para controlo da linha, responsável por movimentar as peças, simulando o processo fabril da Flupol. Este subsistema é descrito com mais detalhe na secção (2.2) já que não é uma componente directa da

solução desenvolvida mas antes uma parte auxiliar que permite validar o trabalho em questão e executar vários testes sem nos obrigar a deslocar ao local real de produção.

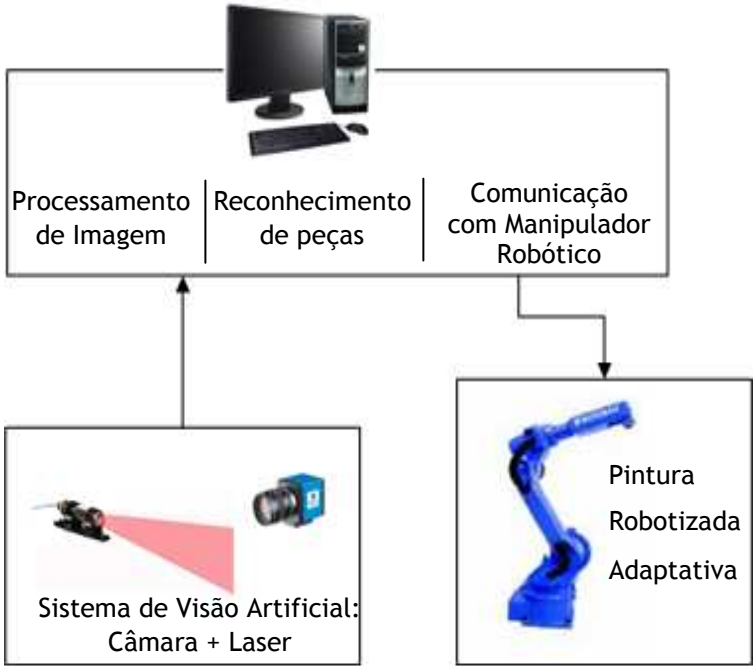

Figura 2.1 Arquitectura do sistema

2.1.1 Sistema de Visão Artificial e Laser

Este sistema é constituído por uma câmara USB monocromática e por um laser de linha, assim como todo o processamento inerente ao tratamento dos dados adquiridos.

A opção da câmara a preto e branco deve-se ao facto de apenas ser necessário identificar em cada *frame* a linha do laser. Num ambiente com iluminação minimamente controlada, o brilho da linha de laser sobressai razoavelmente bem na imagem não sendo necessário recorrer a informação de cor. Além disso, as alternativas disponíveis no laboratório eram câmaras a cores do tipo Bayer; os dados apresentados no formato *bayer* necessitam ser previamente processados para construir uma representação RGB (ver esquema na Figura 2.2). Embora o método de conversão tipicamente usado seja simples, uma interpolação bilinear — o valor RGB para um dado pixel é o valor médio dos pixéis vizinhos — os resultados podem ter uma baixa qualidade, sobretudo nas zonas com descontinuidades de cor, já que este processo de transformação *Bayer→RGB* não é mais do que um filtro passa-baixo, suavizando as transições. De acordo com este formato *Bayer,* podemos também concluir que apenas ¼ do total de pixéis iria ser usado para captar o vermelho do laser já que os restantes servem o propósito de captar as cores verde e azul. A opção, então, de uma câmara em tons de cinzento aparece como a mais natural, onde todos os pixéis registam o brilho do laser em contraste com um fundo escuro e não existe a necessidade de efectuar conversões intermédias.

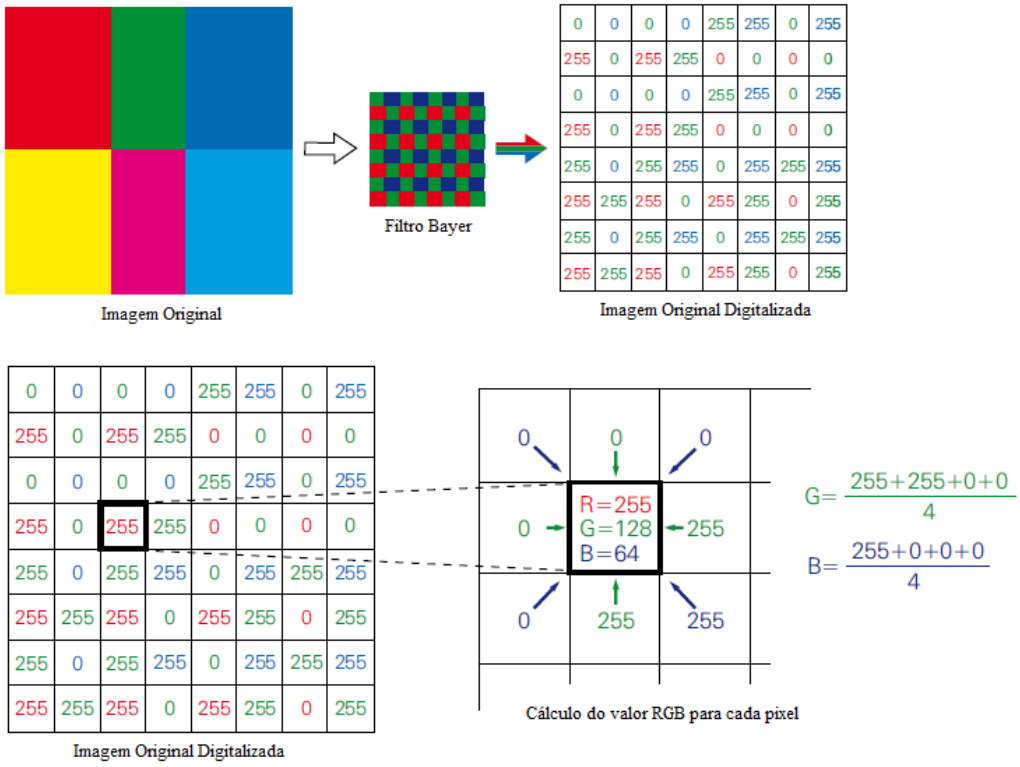

Figura 2.2 Resumo do funcionamento do filtro bayer e reconstrução da imagem no espaço RGB

A câmara adquirida é da marca ImagingSource (Figura 2.3), com resolução 1024×768, sem necessidade de alimentação externa (o cabo de dados USB fornece a potência necessária). Outro dos factores que foi tido em conta para a sua escolha reside na existência de drivers e diversos exemplos de aplicação deste tipo de câmaras já que têm sido frequentemente usadas em projectos no grupo de investigação ligado ao laboratório onde este trabalho foi desenvolvido.

Figura 2.3 Câmara USB da ImagingSource

Associada à câmara, aparece a necessidade de escolha de lentes adequadas. De modo a aproveitar ao máximo a resolução da câmara e tendo em conta que esta não pode ser instalada com grande afastamento do *conveyor* por questões de espaço na fábrica, tem de existir o cuidado de escolher uma lente adequada para que toda a peça caiba na imagem, estando a câmara bastante próxima dos objectos.

Na estrutura mecânica da linha de pintura não existe nenhum apoio natural para a câmara; é necessário colocar um suporte no local e, face ao espaço disponível, era uma das restrições à montagem que este não ficasse mais afastado do que 60cm da linha; esta distância será, assim, a máxima distância de trabalho admissível, *Wd*.

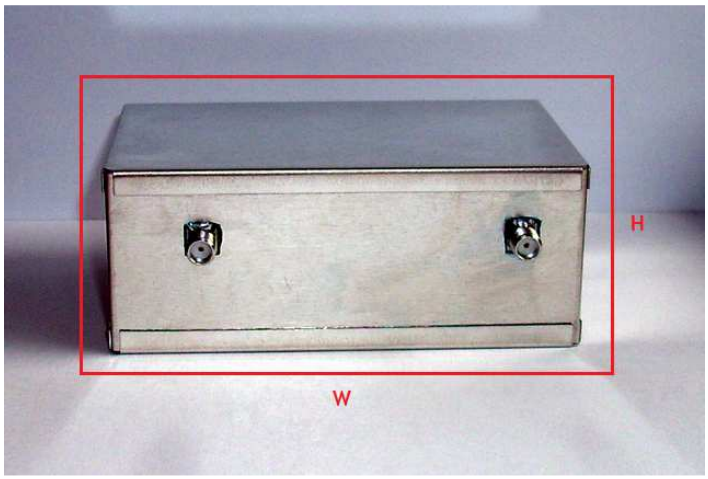

Figura 2.4 Exemplo de objecto que se pretende captar numa imagem

Se pretendermos captar imagens da zona marcada a vermelho (ver Figura 2.4), com altura H e largura W, temos de nos certificar que esta área cabe no CCD e que o aproveita ao máximo, para a distância de trabalho estabelecida.

A partir da *datasheet* da câmara [11], conhecemos o comprimento e largura do CCD, CCD_W=4.8mm e CCD_H=3.6mm, respectivamente.

Com estes dados calculamos o valor máximo da distância focal, f, tanto para a largura como para altura:

$$f_{H/W} = \frac{W_D * CCD_{H/W}}{H/W + CCD_{H/W}} \tag{2.1}$$

O valor mais pequeno entre f_He f_W limita o valor da distância focal da lente. De entre as comercialmente disponíveis, escolhemos a lente com o valor mais próximo desse mínimo mas sempre inferior a ele.

No caso em estudo, as peças têm uma altura máxima de 1 metro. Quanto à largura, esta não se apresenta como uma condicionante para o dimensionamento da lente já que, como será explicado no Capítulo 3, apenas iremos processar a zona da imagem que contém a linha do laser e não a peça toda, pelo que a largura pode ser tomada tão pequena quanto, por exemplo, 20cm.

Perante estes valores, obtiveram-se para a distância focal:

- f_H= 2.15 mm
- f_W= 14.06 mm

Ora, a lente com mais baixa distância focal disponível era de 2.3mm. Neste cenário, foi necessário repensar o posicionamento da câmara. A distância de trabalho não podia ser aumentada e o tamanho dos objectos também não; a solução encontrada foi colocar a câmara

rodada 90º, isto é, a altura da peça fica associada à zona mais larga do CCD e as imagens aparecerão "deitadas". Recalculando as distâncias focais:

- f_H = 10.6 mm
- f_W = 2.8 mm

Encontrou-se no mercado uma lente compatível com a câmara escolhida com distância focal de 2.3 mm. Para compensar a diferença de valores (que causa um não aproveitamento completo da área de imagem) podia ter sido escolhida uma lente com zoom. No entanto estas lentes são mais pesadas, maiores e, principalmente, mais caras. Optou-se por usar uma lente sem zoom e diminuir a distância de trabalho (aproximar a câmara) já que existe esse grau de liberdade.

De forma a completar este sistema, foi escolhido um laser de linha cuja abertura permite formar uma linha de comprimento superior a um metro (para abranger toda a peça) a uma distância de 40cm.

Trata-se de um laser classe II, da *Lasiris,* 1mW de potência, 635nm de comprimento de onda, com uma distribuição de intensidade não-gaussiana (brilho igualmente distribuído por toda a linha).

2.1.2 Plataforma de Software

Para o desenvolvimento de todo o software, que inclui o processamento de imagem, reconhecimento de peças e comunicação com o robô, foi usada a plataforma Lazarus. Trata-se de um software *opensource* semelhante ao Delphi, baseado em FreePascal, tendo sido bastante usado em todos os projectos a decorrer no laboratório onde este trabalho foi executado havendo, por isso, bastante apoio por parte de diversos utilizadores experientes. Existe, inclusive, uma biblioteca de componentes para Lazarus, desenvolvido pelos membros da equipa de futebol robótico 5DPO, inteiramente dedicado ao uso de câmaras Firewire e USB.

2.1.3 Manipulador Industrial

Neste trabalho usou-se o manipulador industrial MOTOMAN HP6, com controlador NX100, *setup* que já se encontrava disponível no laboratório onde se realizaram as experiências. Trata-se de um robô com seis graus de liberdade, multifuncional que se adequa perfeitamente às características do trabalho que pretendemos desenvolver.

Em termos de comunicação, este robô está dotado da possibilidade de comunicar por Ethernet, permitindo transferências de ficheiros, leitura e escrita de variáveis do manipulador e ainda controlo da posição através de comandos específicos. Tudo isto suportado pelo protocolo BSC (BiSync), um protocolo para conexões ponto-a-ponto, half-duplex.

É a partir deste tipo de comunicação que o software desenvolvido carrega valores para variáveis específicas do manipulador que levam a que os programas que execute de adaptem a diferentes dimensões de pensas. Através deste protocolo também temos a possibilidade de escolher qual programa o robô deve executar.

2.2 Recriação da linha de pintura

Com os elementos atrás enumerados é possível constituir uma solução adaptada ao problema da Flupol. No entanto, tendo em conta que este projecto é desenvolvido em ambiente de laboratório, foi necessário erguer toda uma estrutura (incluindo aspectos mecânicos e de controlo) que permitisse reproduzir, dentro desse espaço, a linha de fabrico da empresa em questão.

A estrutura com um *conveyor*, suporte para peças e algumas peças comuns foram fornecidas pela própria Flupol. Com esta montagem é possível efectuar ensaios num ambiente próximo do verdadeiro.

No entanto, para um uso seguro e versátil, o *conveyor* foi sensorizado e controlado, nomeadamente:

- Foram colocados interruptores fins de curso em ambas as extremidades da linha. De acordo com a Figura 2.5 (VFD significa *variable frequency drive)*, podemos ver que a alimentação do motor que movimenta a linha é interrompida quando qualquer um dos interruptores é accionado (a entrada de *enable* do VFD é activa ao nível lógico zero, que acontece sempre que algum dos interruptores é aberto). Esta acção é necessária para evitar que o suporte das peças exceda os limites da linha causando danos na estrutura.
- Da mesma forma foram colocados sensores indutivos de presença em cada extremidade para identificar a posição das peças quando perto do fim da linha.
- Foi colocado um autómato que, recolhendo o sinal dos sensores indutivos, actua na velocidade da linha por intermédio de um variador de velocidade. Os sinais dos sensores são adquiridos por uma carta de entradas digitais e o motor é accionado a partir de uma carta com saídas em relé. A aplicação do autómato responsável por este controlo foi desenvolvida usando *Grafcet* e *structured text — ST*. Na Figura 2.6 apresenta-se um esboço do *Grafcet* implementado: o utilizador fornece um sinal para que a peça se mova, sinal este que activa a transição *Start_Movement*; o *conveyor* entra em funcionamento deslocando a peça de uma ponta à outra da linha, permitindo que, entretanto, se adquiram imagens da peça; quando o sensor indutivo assinala a presença da peça no extremo da linha, é invertido o movimento e retorna-se à posição inicial esperando nova ordem do utilizador.

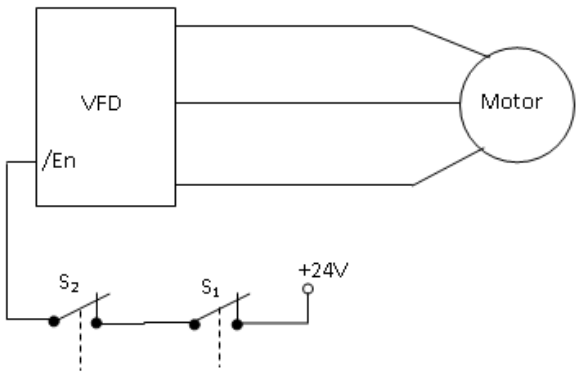

Figura 2.5 Esquema da ligação dos interruptores fins de curso: corte de alimentação do motor aquando do accionamento de qualquer dos interruptores.

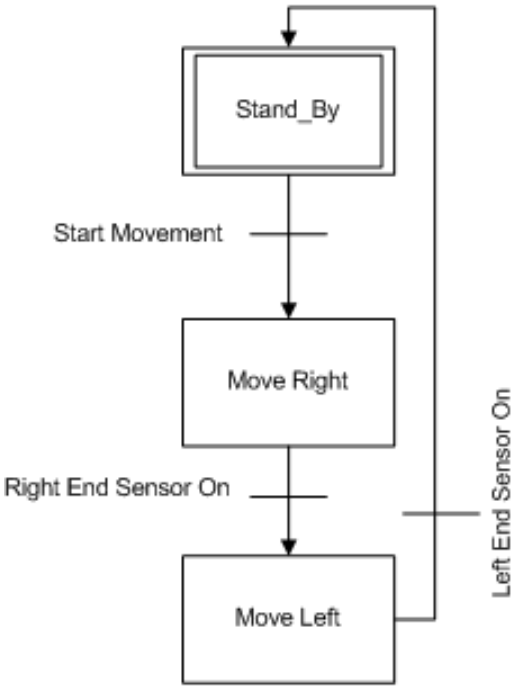

Figura 2.6 Grafcet de controlo do *conveyor*

A estrutura mecânica viria, ainda, a sofrer alterações no sentido de corrigir a iluminação na zona onde se situa a câmara. Criou-se uma zona parcialmente tapada onde a luz natural proveniente das janelas do laboratório e a luz artificial do mesmo afectavam menos a qualidade das imagens adquiridas.

Para além disto, ainda foram colocadas na estrutura calhas metálicas servindo de guias de forma a evitar que as peças baloiçassem durante o percurso.

Capítulo 3

Sistema de Visão Artificial

O sistema de visão artificial, como referido anteriormente, baseia-se numa câmara USB monocromática (cada pixel é caracterizado por uma intensidade luminosa numa escala de cinzentos — [0-255]). A Figura 3.1 mostra um exemplo de um *frame* captado. É possível desde já apercebermo-nos dos seguintes aspectos fundamentais: a iluminação ambiente traz algumas dificuldades à detecção da linha do laser já que os reflexos nas peças e estruturas metálicas sobressaem ao brilho deste; por outro lado, a câmara apresenta uma elevada distorção em barril que terá de ser adequadamente compensada for forma a extrair informação com qualidade a partir das imagens.

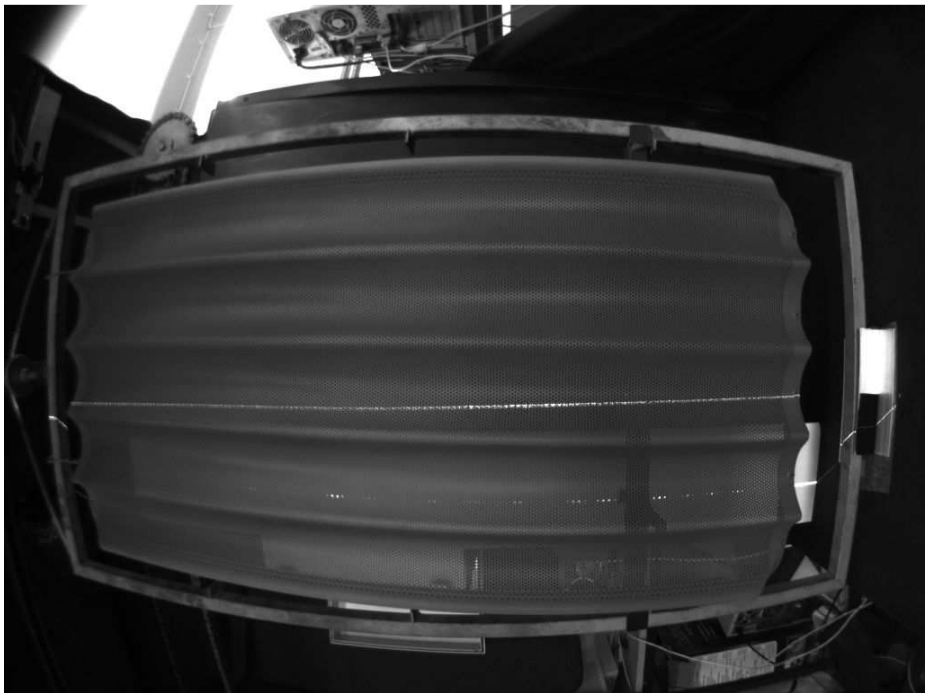

Figura 3.1 *Frame* da câmara USB

Após montar uma estrutura auxiliar recorrendo a placas k-line (placas de poliestireno expandido forradas por cartão) e panos escuros, reduziu-se a interferência da luz exterior e iluminação artificial do laboratório, obtendo-se imagens mais escuras mas onde, agora, é facilmente distinguível a linha de laser do restante plano de fundo (Figura **3.2**).

Figura 3.2 *Frame* da câmara USB após *upgrade* da estrutura com placas e panos

Na medida em que quer a câmara quer o laser estão fixos, a linha de laser aparece na imagem sempre na mesma região — dos 1024x768 pixéis disponíveis, apenas pequenas áreas são, de facto, úteis. Esta característica permite-nos limitar as zonas que, na prática, interessam processar, vejam-se as Figura 3.3 e Figura 3.4: existe uma zona (marcada a branco) onde o laser é visível sempre que não está a incidir sobre um objecto (incide no plano de referência) e uma segunda zona (marcada a violeta) onde o laser aparece sempre que está sobre uma peça. Apenas estas duas zonas necessitam ser processadas e a existência de uma linha de laser numa das zonas invalida, à partida, o aparecimento de uma linha na outra zona; a entrada de uma nova peça na linha de produção é, assim, detectada sempre que deixamos de ter a linha de laser a incidir na totalidade no plano de referência (zona a branco) e passamos a ter uma linha na outra zona.

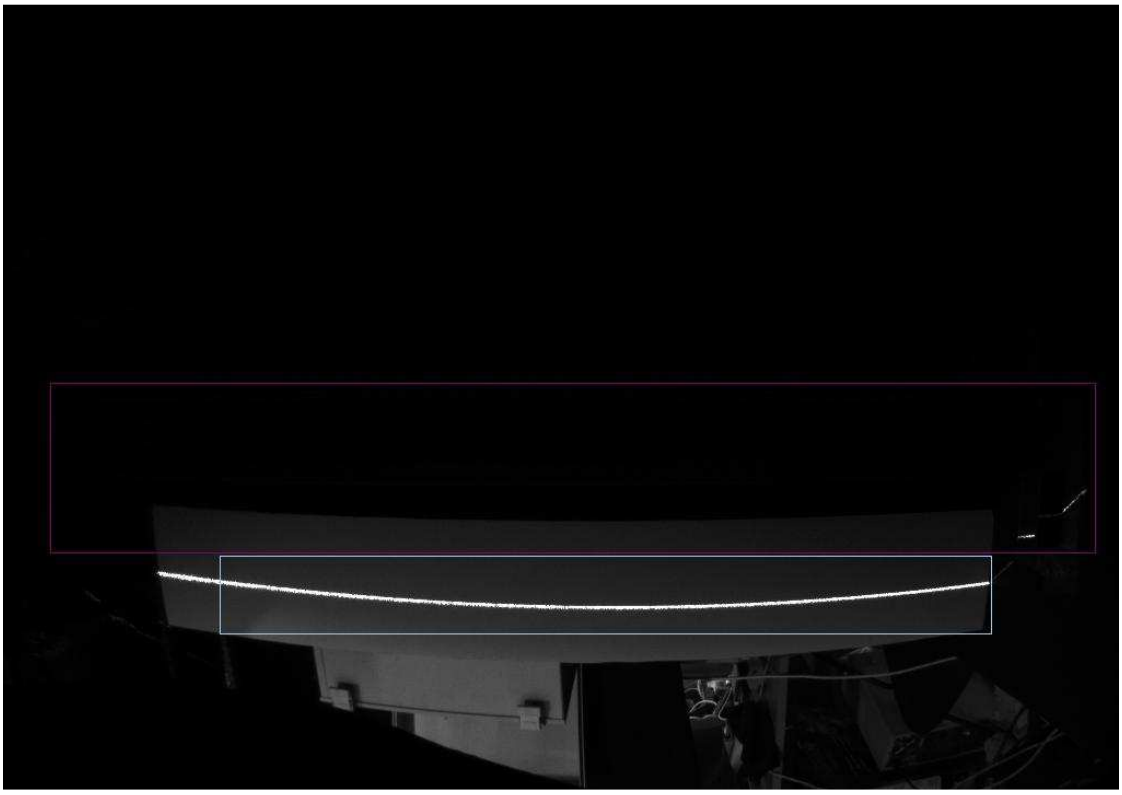

Figura 3.3 Laser a incidir sobre o plano de referência

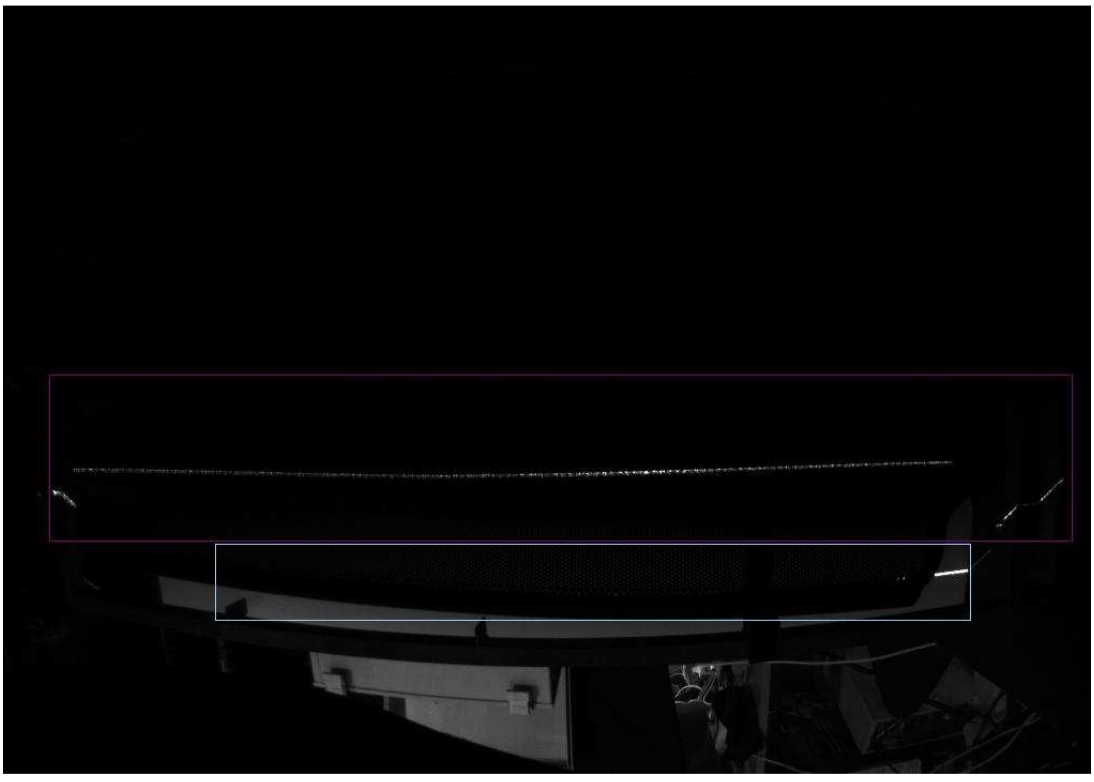

Figura 3.4 Laser a incidir sobre uma peça

A componente de software que realiza o processamento de imagem é coordenada pela recepção de *frame*s da câmara: à chegada de um novo *frame* são executadas as rotinas de

detecção de linhas, reconstrução 3D dos objectos e de visualização, procedimentos estes limitados em termos temporais ao *frame rate* estabelecido (7.5 FPS).

3.1 Detecção de linhas de laser

Após a recepção de um *frame* (F), é executada a rotina de detecção de linhas. O primeiro passo consiste na binarização da imagem, isto é, passar de uma escala de cinzentos para preto e branco. A função de binarização pode ser escrita como:

$$F_{B\&W}[u,v] = \begin{cases} 0 \; se \; F[u,v] < threshold \\ 1 \; se \; F[u,v] \geq threshold \end{cases} \quad , \tag{3.1}$$

Onde o índice b&w indica uma representação binária do *frame* F, F[u,v] representa o pixel de coordenadas (u,v) no *frame* F e *threshold* é o limiar de binarização, abaixo do qual o pixel passa a ter o valor 0 (preto) e acima passa a ter o valor 1 (branco). Na Figura 3.5 encontra-se um exemplo do resultado da transformação para imagem binária.

Figura 3.5 Resultado do processo de binarização de um *frame*

Nas duas zonas de processamento são usados *thresholds* diferentes: quando o laser está sobre a peça, que é um objecto tipicamente escuro, a linha aparece mais ténue e o *threshold* é, por esse motivo, baixo; já o plano de referência é branco e a luz do laser é bastante mais intensa quando incide nesta zona, levando a que o *threshold* tenha de ser mais elevado para não expandir demasiado a linha e não sofrer interferência da iluminação exterior e reflexos (ver Figura 3.6, que apresenta o resultado de uma má escolha de *threshold*).

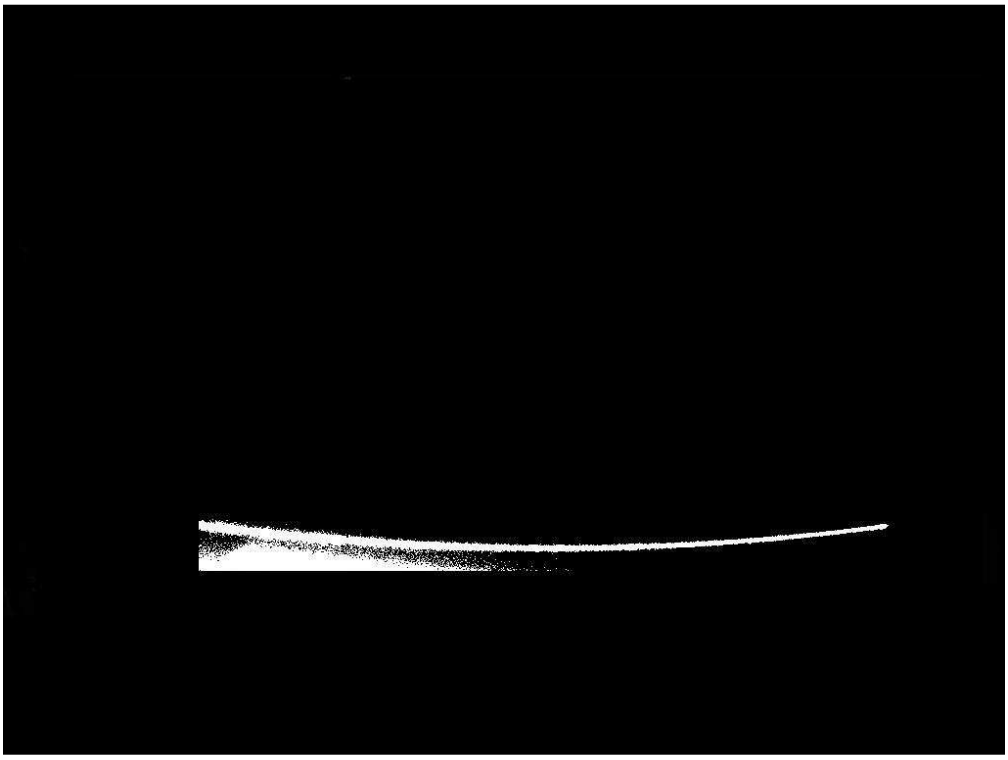

Figura 3.6 Resultado do processo de binarização com threshold desadequado

A visualização de $F_{b\&w}$ permite compensar efeitos de iluminação natural e/ou artificial no local ajustando-se o valor do *threshold*, até que apenas a linha seja detectada com um mínimo de ruído.

A detecção da linha do laser é levada a cabo com varrimentos verticais na imagem: para cada coluna, procura-se um ponto pertencente à linha (acima do valor *threshold)* e toma-se esse ponto, $v_{i,}$ como início de um segmento. Continuando o varrimento, guardamos todos os pontos desde v_i até se voltar a encontrar pontos abaixo do *threshold;* guarda-se o último ponto, v_f. A Figura 3.7 Esquema de pesquisa de uma linha na imagem esquematiza este processo.

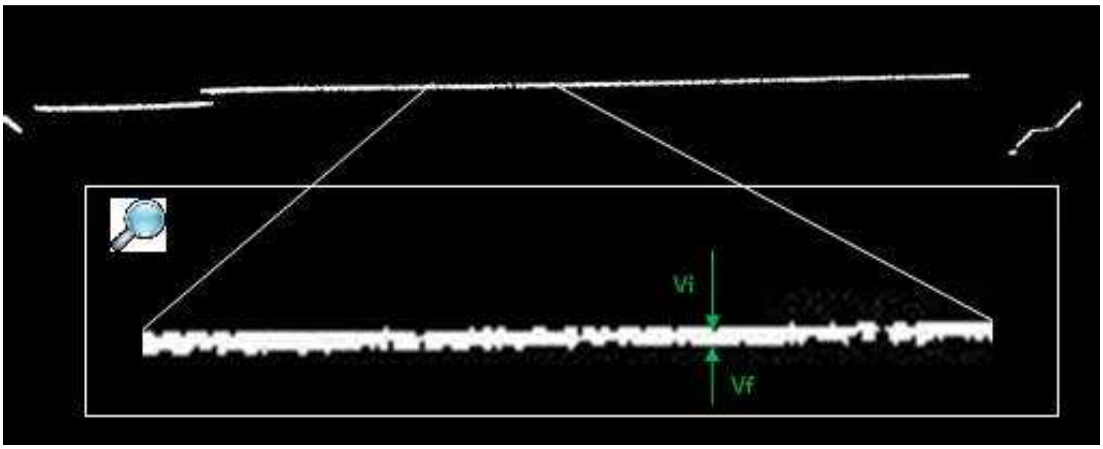

Figura 3.7 Esquema de pesquisa de uma linha na imagem

Em cada coluna da imagem será encontrado, portanto, um segmento (de v_i a v_f) e o conjunto destes define uma linha. Cada segmento é caracterizado pela coluna *u* da matriz a que se refere e pela média ponderada dos pixéis que o compõem, v_m:

$$v_m = \frac{\sum_{j=v_i}^{v_f} F[u,j] \times j}{\sum_{j=v_j}^{v_f} F[u,j]} \quad , \tag{3.2}$$

3.2 Calibração do Sistema <Câmara+Laser>

A calibração é, essencialmente, um método de identificação do valor dos parâmetros do sistema, que nos permite reconhecer a posição no mundo de um qualquer pixel da imagem e, igualmente, conhecer o local na imagem onde será mapeado um qualquer ponto do mundo.

Uma câmara pode ser representada por um modelo matemático equivalente. Os respectivos parâmetros são, normalmente, divididos em duas classes: os parâmetros intrínsecos e os parâmetros extrínsecos. Estes últimos resumem-se à posição e rotação da câmara no referencial do mundo. Já os parâmetros intrínsecos, independentes dos extrínsecos, caracterizam a distância focal, a altura e largura dos pixéis do CCD e o centro da imagem.

3.2.1 Modelo da câmara

Nas câmaras convencionais, o modelo mais usado é o *pinhole* [12]. O princípio deste modelo encontra-se esquematizado na Figura 3.8: toma-se a câmara como uma caixa à prova de luz onde existe um orifício de reduzidas dimensões (daí a designação *pinhole*), idealmente considerado como apenas um ponto no espaço; a luz proveniente de todo o cenário imediatamente à frente do orifício passa por este e é projectada no plano da imagem. Esta transformação de coordenadas do espaço (em 3D) para um plano de imagem (2D) tem o nome de projecção perspectiva.

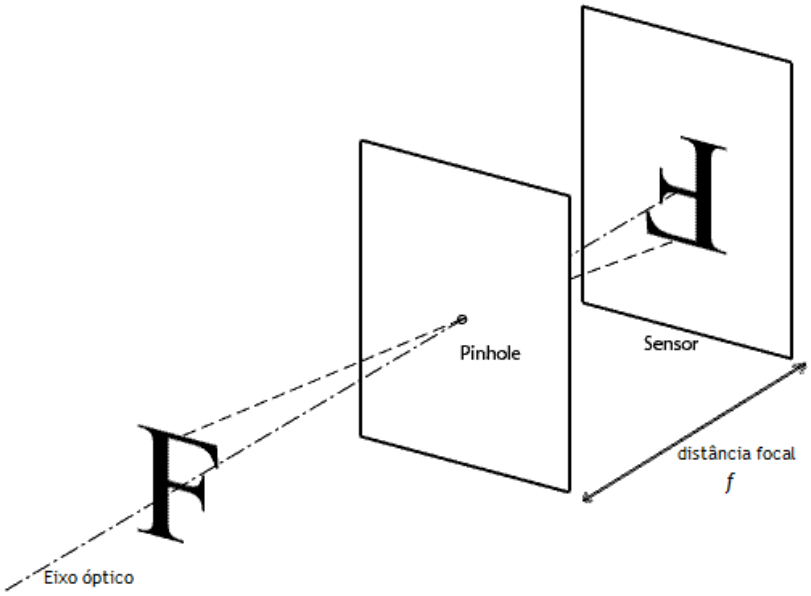

Figura 3.8 Projecção Perspectiva no modelo pinhole

Para obter o modelo *pinhole* da câmara usada partiu-se de uma representação geométrica do fenómeno atrás representado e foram especificados os diversos sistemas de coordenadas necessários:

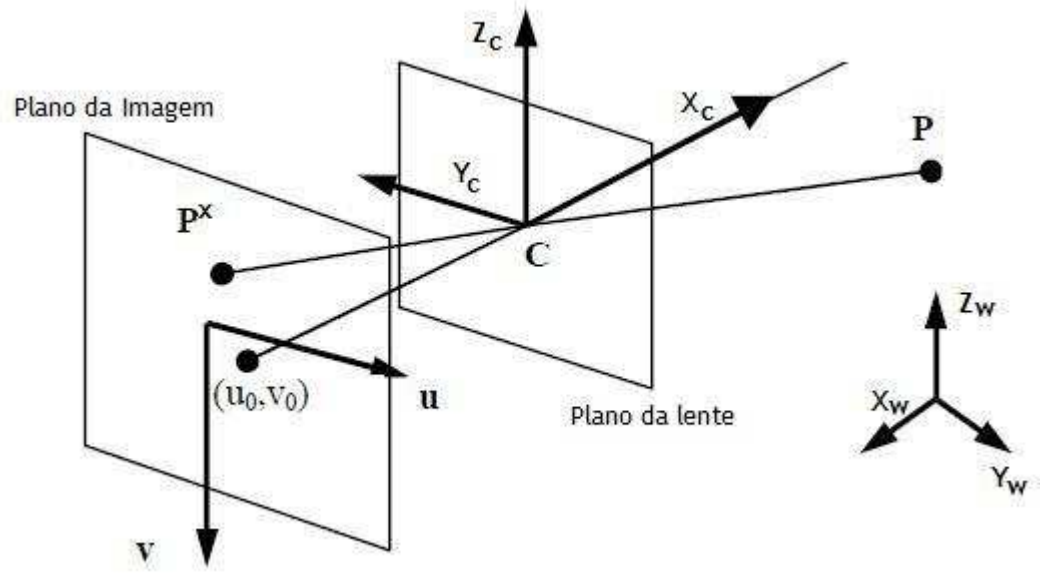

Figura 3.9 Especificação dos sistemas coordenados [16]

Com base nos esquemas anteriores (Figura 3.8 e Figura 3.9), introduzem-se aqui algumas considerações e notação usadas neste documento:

- *f* é designada de distância focal, a distância entre o plano da lente e o plano da imagem;

- Todos os sistemas de coordenadas são cartesianos;
- No referencial do mundo, $OX_wY_wZ_w$, um ponto genérico P, é representado pelas suas coordenadas P = (x,y,z);
- A câmara (ponto C) está posicionada no mundo nas coordenadas C = (x_{c0},y_{c0},z_{c0});
- Este mesmo ponto C é a origem do referencial da câmara, $OX_CY_CZ_C$;
- O ponto P, projectado no plano da imagem, passa a ser visto como um pixel, e é endereçado segundo as coordenadas em pixéis P^x=(u,v);
- O pixel central tem coordenadas (u_0,v_0).

Voltando então ao esquema apresentado na Figura 3.9**,** podemos verificar que um pixel P^x corresponde a uma projecção de um ponto P do mundo no plano da imagem. Usando, agora, coordenadas homogéneas normalizadas para representar os pontos, isto é, P→P'(x,y,z,1) e P^x→$P^{x'}$(u,v,1),temos:

$$P^{x\prime} = H.P' \quad , \tag{3.3}$$

em que H é a matriz de projecção da câmara. Esta matriz é composta por uma série de transformações lineares de rotação e translação entre referenciais.

Começando pela transformação das coordenadas do mundo em coordenadas no referencial da câmara, podemos escrever a matriz de transformação homogénea:

$$H_c^w = \begin{bmatrix} R_c^w & T_c^w \\ 0 & 1 \end{bmatrix} \quad , \tag{3.4}$$

em que R_c^w e T_c^wrepresentam, respectivamente, a matriz de rotação e a matriz de translação entre os dois sistemas cartesianos. Temos até agora a relação entre P em coordenadas do mundo e P em coordenadas no referencial $OX_CY_CZ_C$;

Para obter o ponto no plano da imagem, definimos, num passo intermédio, a transformação de coordenadas 3D em 2D através de uma projecção perspectiva, que pode ser representada pela matriz de transformação:

$$H_F^c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & {}^1/_f & 0 \end{bmatrix} \quad , \tag{3.5}$$

Falta apenas escalar esta projecção de acordo com a resolução horizontal e vertical do CCD e aplicar a translação entre o referencial da câmara e o do plano da imagem. Esta última transformação pode ser representada pela matriz

$$H_I^F = \begin{bmatrix} du & 0 & u_0 \\ 0 & -dv & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad , \tag{3.6}$$

em que du e dv fazem a conversão dos pixéis para unidades de distância.

A transformação de $OX_wY_wZ_w$ para (u,v) fica, deste modo, completamente caracterizada pelo que, voltando à Equação (3.3), podemos reescrevê-la como:

$$P_x = H_I^F H_F^c H_c^w . P \quad , \tag{3.7}$$

Este relação foi implementada na função xyz2uv, que recebe como parâmetros de entrada um ponto tridimensional e devolve as coordenadas correspondentes no plano da imagem.

Possuímos, até este ponto, o modelo do sistema de visão, falta agora determinar os parâmetros para podermos passar pontos do mundo para pixéis e vice-versa.

Deve-se notar que o modelo do tipo *pinhole* utilizado apresenta algumas limitações, nomeadamente (e com maior relevância):

- O facto de os pixéis poderem não apresentar um formato exactamente quadrangular, existindo à partida uma distorção em cada direcção;
- O efeito das lentes causador da distorção em barril. Este efeito, como o nome indica, provoca o encurvamento de linhas, isto é, linhas rectas no mundo aparecem na imagem como linhas curvas. O erro inerente a esta distorção aumenta quanto mais nos afastamos do centro da imagem;
- A abertura variável da câmara; o modelo *pinhole* assume uma abertura de dimensões muito reduzidas (*pinhole*) pelo que o modelo é cada vez menos exacto quanto maior for a abertura exigida à câmara.

A deformação dos pixéis pode ser considerada desprezável para a aplicação em questão visto que trás pouco erro às medições e a abertura da câmara será imposta pelas condições de iluminação do local pelo que a distorção associada também será desprezada.

No entanto, a distorção em barril introduz um erro elevado no sistema e, como discutido na secção seguinte, será corrigida através de um algoritmo apropriado.

Deve-se ainda notar que, por razões óbvias, o sistema (2.8) não é invertível — a partir de um pixel (u,v) temos uma infinidade de pontos do mundo cuja projecção perspectiva no plano da imagem é a mesma. No limite, poderemos apenas definir a equação da recta que contém todos esses pontos.

3.2.2 Calibração por correspondência de 2 pontos

Existem várias técnicas para calibração de câmaras que permitem estimar os parâmetros do modelo apresentado na secção 3.2.1. Destaca-se aqui o algoritmo desenvolvido por Roger Y. Tsai [13][14] e outros algoritmos de correspondência ponto-a-ponto, nomeadamente os métodos DLT (*direct linear transformation)* [15][16] cujo princípio de funcionamento consiste em ter, à partida, um conjunto de pontos dos quais se conhecem, rigorosamente, as coordenadas no mundo e as correspondentes coordenadas na imagem.

Atente-se na equação matricial (3.7). A partir dela podemos escrever duas equações escalares, com onze parâmetros a determinar. Isto implica conhecer, no mínimo, seis pontos na situação atrás descrita, ou seja, conhecer 6 pontos no mundo e a respectiva posição, em

pixéis, na imagem para que possamos obter um sistema de onze equações (embora cada uma delas altamente não linear). Aqui entraria um método DLT para conseguir encontrar uma solução. Note-se também que para garantir alguma robustez ao algoritmo seria desejável usar não seis mas um número mais avolumado de pontos ao que um algoritmo de mínimos quadrados também seria ajustado a encontrar uma solução, pagando-se claro, em tempo de processamento e complexidade.

Tendo em conta os bons resultados conseguidos num trabalhado semelhante com laser [10], optou-se por implementar um algoritmo de calibração mais simples, usando apenas dois pontos. Para este efeito, foram tomadas algumas simplificações ao modelo apresentado:

- O centro da imagem corresponde ao pixel central, isto é, para uma resolução de 1024x768, u_0=512 e v_0=384;
- São usados os valores fornecidos pelo fabricante da câmara para du e dv que, recorda-se, são um factor de escala que transforma número de pixéis em comprimentos absolutos.
- A posição da câmara no mundo é conhecida (medida no local após o *setup* de todos os elementos da estrutura mecânica), que é equivalente a dizer que a translação do sistema de coordenadas da câmara em relação ao referencial do mundo é conhecido.

Partindo destes pressupostos, passamos a ter apenas quatro parâmetros para estimar: três associados à rotação da câmara no mundo e a distância focal equivalente. Assim, foi implementado um algoritmo de calibração usando apenas dois pontos conhecidos, que se descreve de seguida.

Comecemos por corrigir a distorção em barril da câmara. Este efeito pode ser caracterizado, aproximadamente, pela seguinte equação matricial:

$$\begin{bmatrix} u_R \\ v_R \end{bmatrix} = \left(1 + kr^2\right) \begin{bmatrix} u_D \\ v_D \end{bmatrix} \tag{3.8}$$

onde *k* é o coeficiente da distorção em barril , r a distância ao centro da imagem , o índice 'D' refere-se a pixéis que sofrem de distorção e o índice 'R' representa o pixel com coordenadas já corrigidas para compensar a distorção. O valor de k foi ajustado iterativamente até que, olhando para linhas na imagem que sabemos serem rectas no mundo, estas apareçam sem curvatura.

Na Figura 3.10 **e** na Figura 3.11 podemos ver, respectivamente, uma imagem original e a mesma após a compensação da distorção em barril. Note-se a linha do laser e as bordas da armação que segura a peça: passaram de linhas curvas para linhas quase rectas, que são pois o seu aspecto verdadeiro.

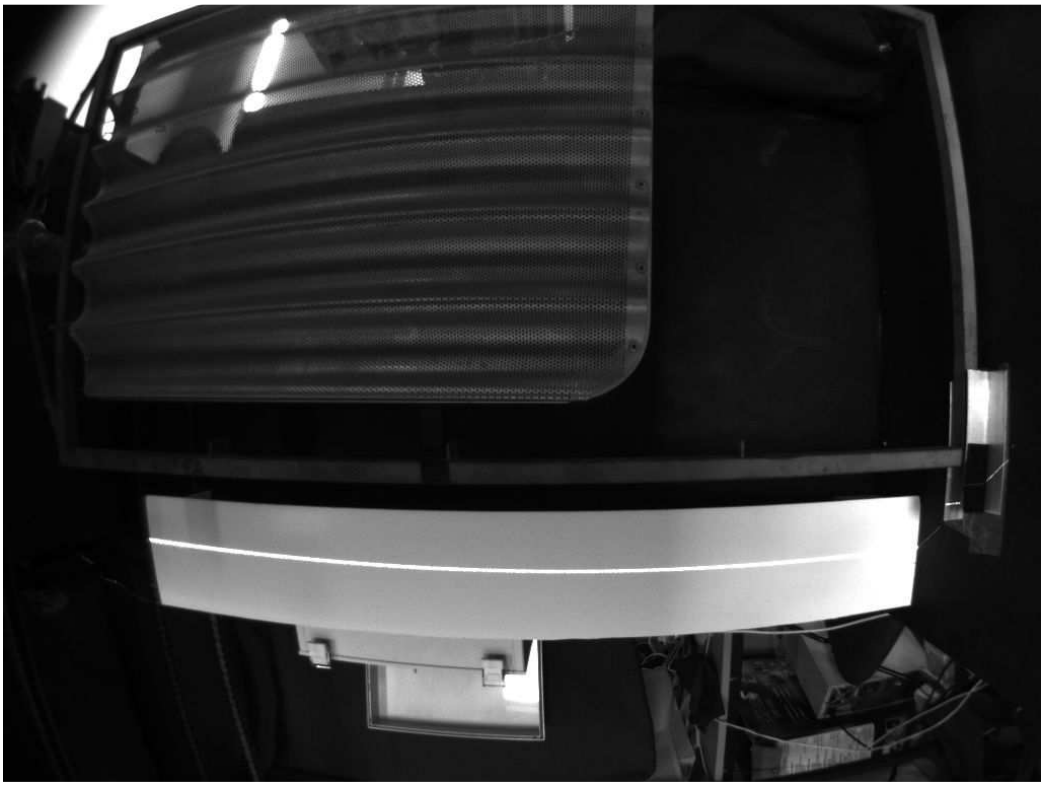

Figura 3.10 *Frame* original da câmara com uma peça e laser: linhas aparecem cruvas

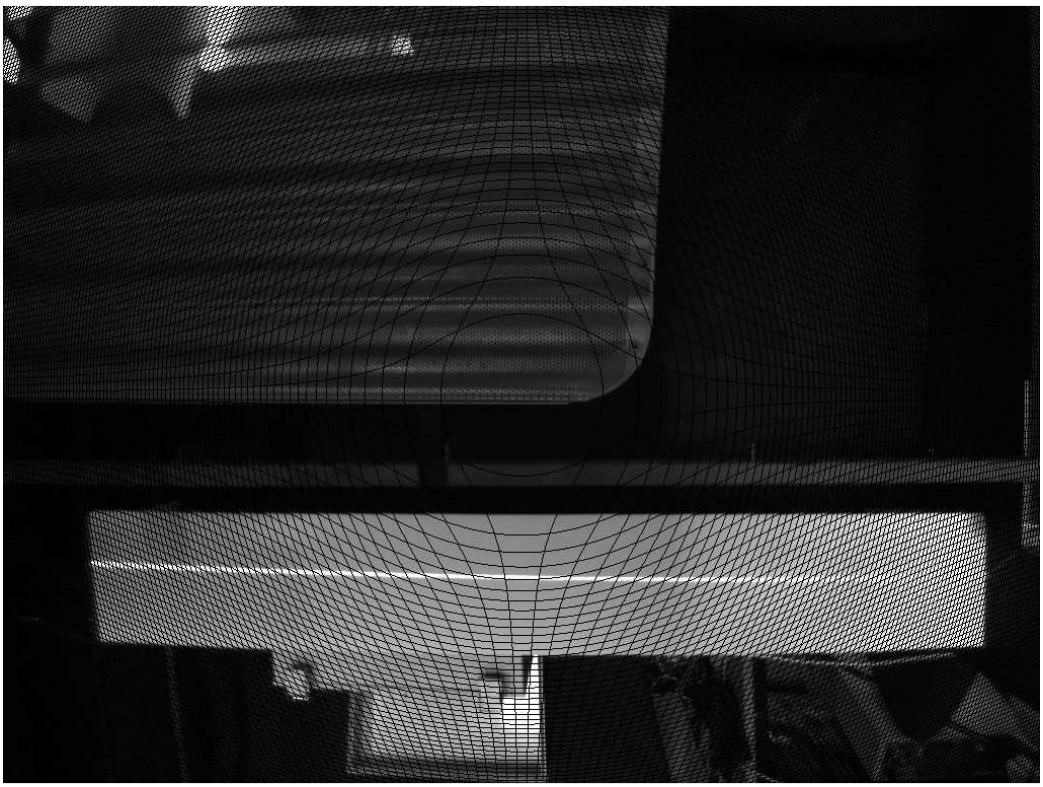

Figura 3.11 Imagem após compensação do efeito de barril do *frame* mostrado na Figura 3.10: linhas mais rectas, próximas da realidade

Posto isto, vamos analisar a configuração geométrica do sistema de visão. Escolhemos um dos pontos de calibração, P_{c1}, como sendo aquele que na imagem corresponde ao pixel central — P_{c1}^{x}=(u_0,v_0) — e medimos a sua posição no mundo (x_{c1},y_{c1},z_{c1}).

A rotação de $OX_wY_wW_w$ para $OX_CY_CZ_C$ pode ser descrita como uma composição de 3 rotações simples, $R_{\theta x}R_{\theta y}R_{\theta z}$. A partir da Figura 3.12, podemos imediatamente estabelecer os ângulos de rotação em z e em y:

$$\begin{cases} \theta_z = \arctan\left(\frac{y_{c1}-y_c}{x_{c1}-x_c}\right) \\ \theta_y = \arctan\left(\frac{z_c-z_{c1}}{\sqrt{(x_{c1}-x_c)^2+(y_{c1}-y_c)^2}}\right) \end{cases}, \tag{3.9}$$

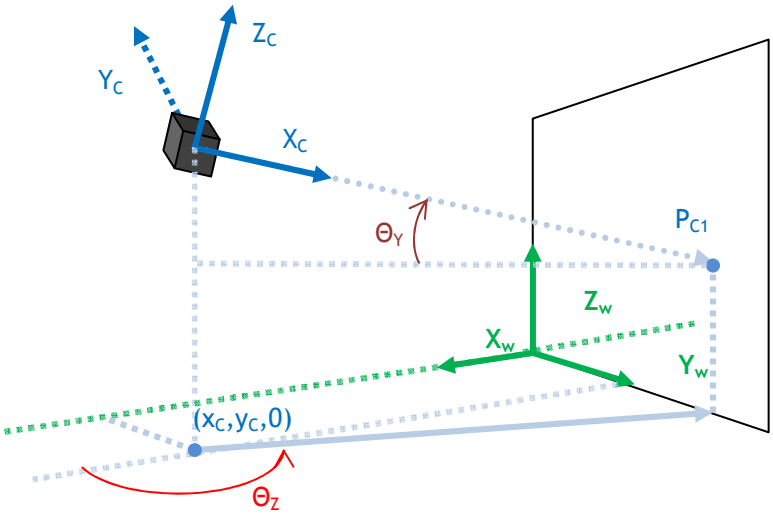

Figura 3.12 Rotação do sistema de coordenadas da câmara em relação ao referencial do mundo

Para o cálculo do ângulo θ_x definimos o segundo ponto de calibração, P_{c2}: escolhemos um ponto no mundo e verificamos quais as suas coordenadas (u,v) ou vice-versa.

Tomamos o valor 0 para θ_x, um valor aleatório para a distância focal (parâmetro ainda não estimado), o resultado da Equação (3.8) e recorremos à função xyz2uv usando as coordenadas no mundo do ponto P_{c2}. Esta função retorna o par (u_{c2}^*, v_{c2}^*). A diferença de ângulos entre este e o valor real conhecido (u_{c2}, v_{c2}) dá-nos directamente θ_x. Resta esclarecer que o uso de um valor aleatório para a distância focal não afecta o resultado já que este parâmetro actua somente como um factor de escala, não fazendo variar os ângulos.

Por fim, é estimado o valor da distância focal. Este cálculo é feito através de um procedimento recursivo, em que em cada iteração i temos:

$$f_{i+1} = f_i \frac{\|u_{c2}, v_{c2}\|}{\|u_{c2\,i}^*, v_{c2\,i}^*\|}, \tag{3.10}$$

onde se usam todos os parâmetros já estimados até ao momento para calcular. $(u^*_{c2_i}, v^*_{c2_i})$. É esperado que $\|u^*_{c2}, v^*_{c2}\|$ convirja para $\|u_{c2}, v_{c2}\|$ à medida que se ajusta o valor de f.

Neste momento, o modelo encontra-se completamente caracterizado.

3.3 Reconstrução 3D

Para completar a transformação de pixéis em pontos do mundo, precisamos de informação adicional sobre o ponto. Como se pode observar na Figura 3.13, e já havia sido referido anteriormente, existe um número infinito de pontos do mundo que é projectado no mesmo ponto na imagem. Dispondo apenas de uma câmara, no limite conseguimos definir a equação da recta que contém todos esses pontos.

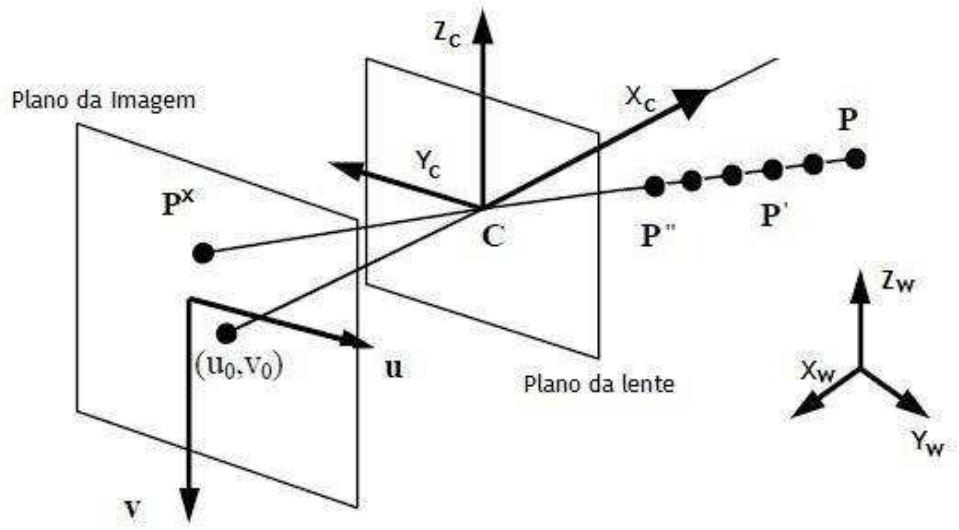

Figura 3.13 Projecção de vários pontos do mundo no mesmo ponto do plano da imagem [16]

3.3.1 Triangulação com laser

O uso do laser serve o propósito de podermos identificar inequivocamente qual o ponto da recta que está, de facto, a ser observado. O laser de linha, na realidade, forma um plano; ao reconhecer um ponto da imagem como sendo do laser, estamos a classificar esse ponto como pertencente ao plano do laser. Atente-se na Figura 3.14: com o laser colocado obliquamente em relação à câmara (para evitar que o plano contenha a recta dos possíveis pontos), o plano intercepta a recta em apenas um ponto.

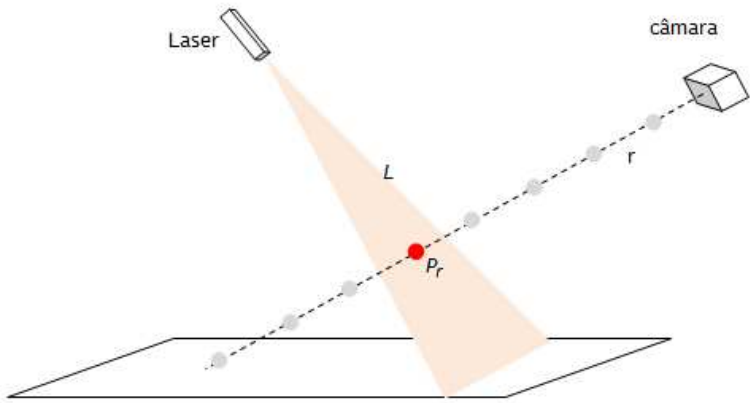

Figura 3.14 Intercepção do plano do laser com a recta que contém todos os pontos projectados no mesmo pixel na imagem

É necessário, portanto, implementar um algoritmo básico que calcula a intercepção de uma recta com um plano. Uma recta no espaço é caracterizada por um vector com a direcção da mesma, $w_r = [x_r, y_r, z_r]^T$, ou seja, qualquer ponto P_r da recta está na direcção w_r a partir de um ponto inicial, $P_{r0} = [x_{r0}, y_{r0}, z_{r0}]^T$, também pertencente à recta. Por sua vez, um plano é definido por um vector normal, $w_n = [x_n, y_n, z_n]^T$, e um ponto, $P_{L0} = [x_{L0}, y_{L0}, z_{L0}]^T$, pertencente ao plano ; um ponto P_L está contido no plano se o produto interno entre o vector normal e um segundo vector $(P_L - P_{L0})$ for nulo.

Seja *r* a recta que obtemos pelo sistema inverso da Equação 3.7 e *L* o plano formado pelo laser:

$$r: \quad P_r = P_{r0} + w_r . t \quad , t \in \mathbb{R} \tag{3.11}$$

$$L: \quad w_n . (P_L - P_{L0}) = 0 \tag{3.12}$$

O ponto que desejamos encontrar, P_r, pertence simultaneamente a *r* e *L* logo temos:

$$(P_{r0} + w_r . t) . w_n = d \quad , \tag{3.13}$$

onde $d = w_n . P_{L0}$ resultando para t:

$$t = \frac{d - x_n x_{r0} - y_n y_{r0} - z_n z_{r0}}{x_n x_r + y_n y_r + z_n z_r} \tag{3.14}$$

Para determinarmos o ponto, substituímos *t* na Equação da recta em (3.11).

3.3.2 Calibração do laser

De forma a usarmos o algoritmo apresentado na secção anterior, temos de conhecer a equação do plano do laser.

Um plano pode ser definido, por exemplo, por um ponto e uma recta. A posição do laser no mundo é conhecida (medida no momento do setup do sistema). Na Figura 3.15 mostra-se um *frame* da câmara onde é visível a linha de laser a incidir num plano de referência.

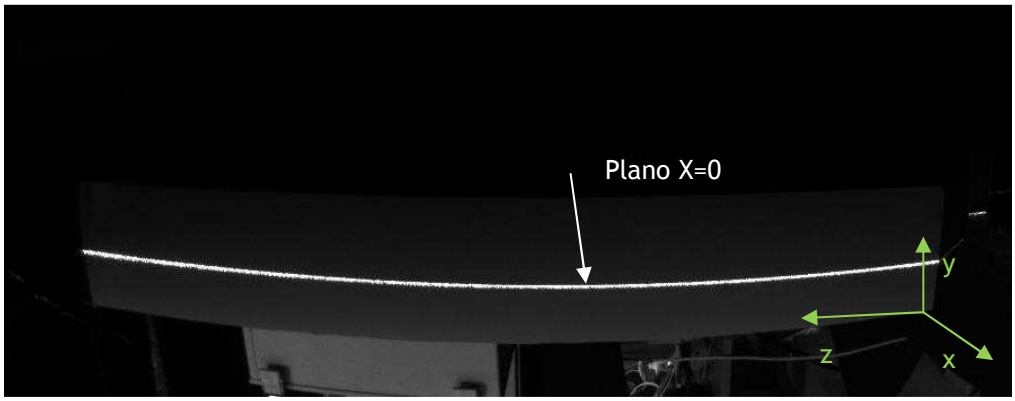

Figura 3.15 Laser a incidir no plano de referência: todos os pontos do laser têm X=0.

É possível usar a relação entre coordenadas (u,v) e coordenadas (x,y,z) para cada ponto dessa linha já que o plano está rigorosamente situado em X=0 logo só se apresenta necessário calcular duas coordenadas o que nos transporta o problema para apenas duas dimensões. A equação da linha do laser é, assim, estimada por mínimos quadrados:

$$y = mz + b + \varepsilon \quad , \tag{3.15}$$

onde ε representa o erro. Para o par (x,y) temos N observações — cada um dos pontos da linha do laser detectados na imagem e passados para coordenadas do mundo assumindo x=0. Notar também que o formato y = f(z) origina que a linha tenha declive m próximo de zero (atentar na Figura 3.15); já na forma z = f(y), a recta tem declive que tenderá para valores muito grandes, infinito no limite, causando problemas a nível de cálculo numérico.

Numa aproximação por mínimos quadrados queremos minimizar a soma do quadrado dos erros, o que nos leva a:

$$min \sum_{i=1}^{N} \varepsilon_i^2 = min \sum_{i=1}^{N} (y_i - b - mz_i)^2 \tag{3.16}$$

Para encontrarmos o mínimo da função, calculamos o zero das derivadas parciais, $^{\partial}/_{\partial b}$ e $^{\partial}/_{\partial m}$:

$$\begin{cases} Nb + \sum_{i=1}^{N} z_i\, m - \sum_{i=1}^{N} y_i = 0 \\ \sum_{i=1}^{N} z_i\, b + \sum_{i=1}^{N} z_i^2\, m - \sum_{i=1}^{N} z_i y_i = 0 \end{cases} \tag{3.17}$$

O sistema anterior pode ser resolvido usando a regra de Kramer, que resulta em

$$\begin{cases} m = \dfrac{n\sum_{i=1}^{N} z_i y_i - \sum_{i=1}^{N} z_i \sum_{i=1}^{N} y_i}{n\sum_{i=1}^{N} z_i^2 - (\sum_{i=1}^{N} z_i)^2} \\ b = \dfrac{\sum_{i=1}^{N} z_i^2 \sum_{i=1}^{N} y_i - \sum_{i=1}^{N} z_i \sum_{i=1}^{N} z_i y_i}{n\sum_{i=1}^{N} z_i^2 - (\sum_{i=1}^{N} z_i)^2} \end{cases}, \tag{3.18}$$

Definida a recta s do laser em x=0, caracterizamos o plano usando mais um ponto, a posição do laser no mundo — P_{Lsr} = (x_{Lsr}, y_{Lsr}, z_{Lsr}). Seja P_0 um qualquer ponto pertencente à recta *s* anteriormente calculada. Então, o vector normal ao plano do laser, w_n, é calculado a partir do produto externo do vector director de *s*, w_s, com o vector (P_{Lsr}- P_0):

$$w_n = w_s \times (P_{Lsr} - P_0) \tag{3.19}$$

A constante d, que resta calcular para caracterizar o plano, é obtida usando, novamente, a posição do laser:

$$d = -w_n . P_{Lsr} \tag{3.20}$$

3.3.3 Reconstrução tridimensional dos objectos

Conjugando o resultado do algoritmo de detecção de linhas (que fornece pontos de coordenadas (u,v) que pertencem ao laser) e o algoritmo de passagem de coordenadas de imagem para coordenadas do mundo (agora completo já que possuímos informação sobre o plano do laser) conseguimos reconstruir as linhas da imagem agora a 3D.

Aproveitando o facto de que a linha de produção se encontra em movimento com velocidade constante, o scan com o laser é feito automaticamente por toda a peça. Foi obtida uma estimativa da velocidade da linha e, a cada novo *frame*, ajustam-se os valores (x, y, z) dos pontos para compensar o movimento da peça.

Foi usado o GLScene, uma ferramenta openSource de OpenGL para Lazarus, para desenhar os objectos em 3D. Esta funcionalidade serve, sobretudo, para verificação de erros dos varrimentos com o laser, validação da calibração e *debug*. A reconstrução é feita em tempo

real e o utilizador pode "navegar" na imagem, isto é, mover-se em torno da peça, aproximar ou afastar ou, findo o processo de aquisição de imagem, redesenhar passo a passo e verificar a qualidade da informação. Da Figura 3.18 à Figura 3.20 mostram-se alguns exemplos de reconstruções de algumas peças (na Figura 3.16 e na Figura 3.17 são apresentadas fotografias com o aspecto original da peça para comparação) e de perspectivas diferentes de uma mesma peça. É interessante atentar no pormenor das zonas mortas — Figura 3.21 — que se trata de um fenómeno de oclusão devido ao posicionamento do laser em relação à peça; algumas faces são inatingíveis pela linha de laser criando "buracos" na reconstrução. Este efeito pode ser corrigido usando, por exemplo, dois lasers, cada um apontando de um ângulo oposto ao outro, no entanto para o trabalho que se desenvolveu neste projecto tal não foi necessário.

Figura 3.16 Tabuleiro com ondulado horizontal

Figura 3.17 Tabuleiro com ondulado vertical

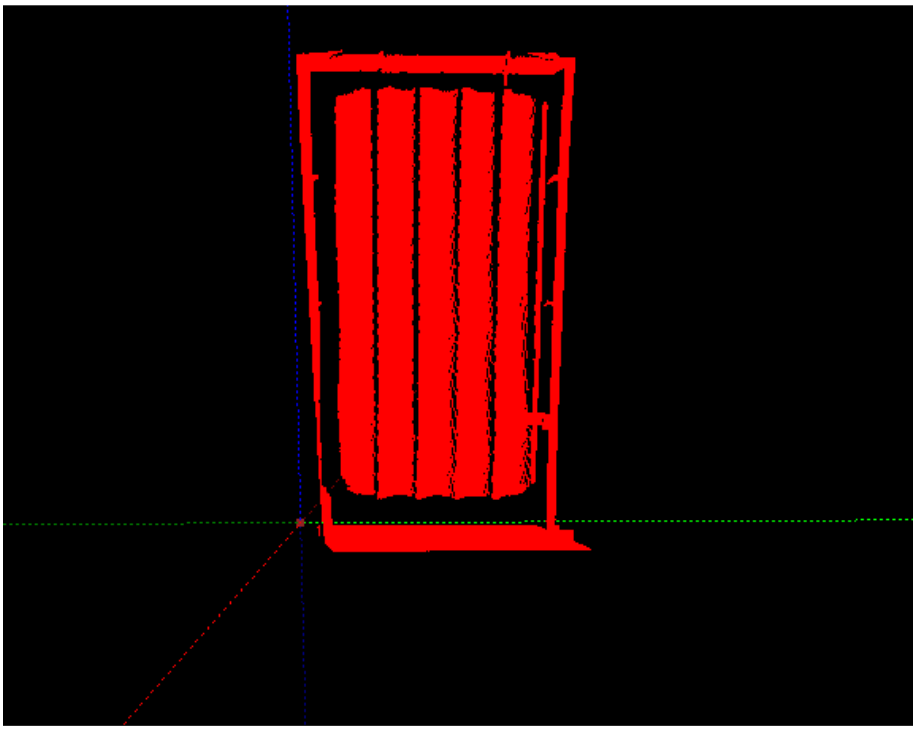

Figura 3.18 Reconstrução 3D de tabuleiro ondulado horizontal – Visão Frontal

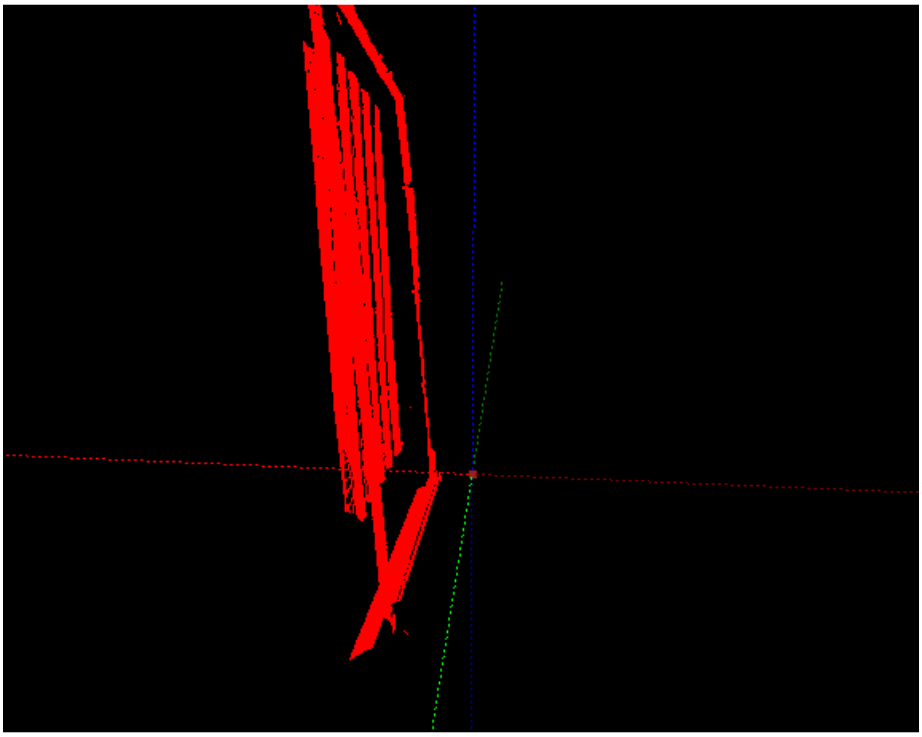

Figura 3.19 Reconstrução 3D de tabuleiro ondulado horizontal – visão lateral – noção de profundidade

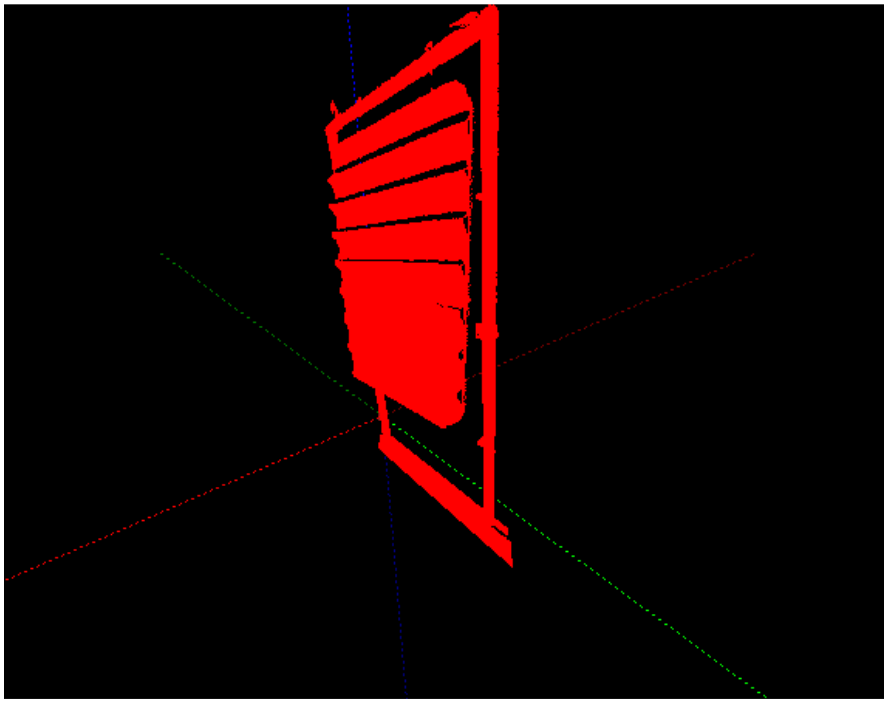

Figura 3.20 Reconstrução 3D de tabuleiro com ondulado vertical

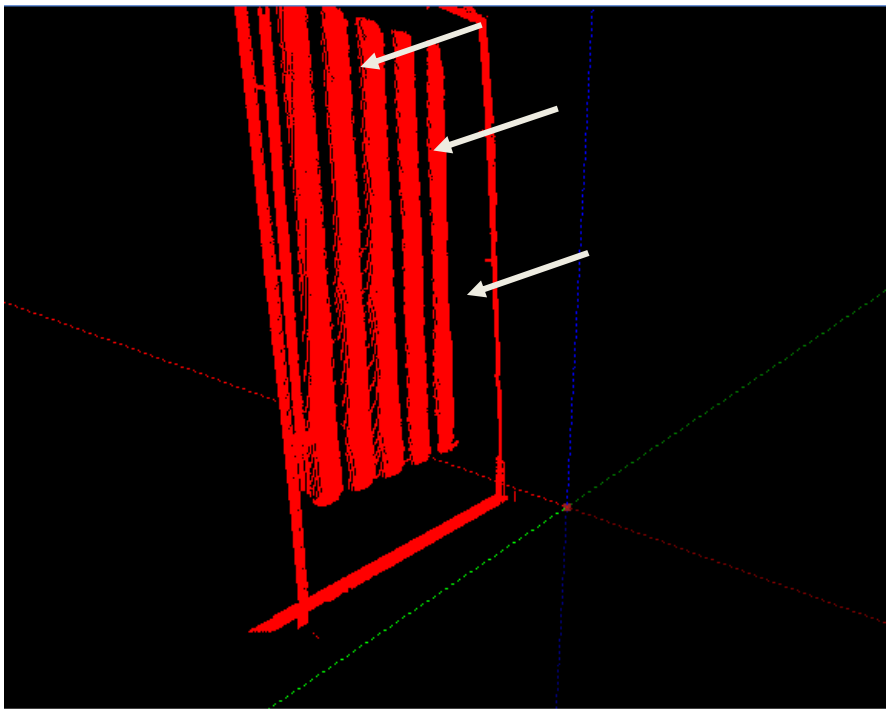

Figura 3.21 Reconstrução 3D de uma peça: indicação de zonas mortas

Capítulo 4

Reconhecimento e Caracterização dos Objectos

Neste capítulo é descrito o processo pelo qual se distinguiram os diferentes tipos de peças que passam na linha de produção por fim a adequar o processo de pintura a cada um deles.

Numa primeira fase é procurado o objecto dentro de toda a informação obtida. O espaço ocupado pela peça é delimitado e caracterizado e eliminam-se as fontes de ruído existentes. Centrando o processamento a esse espaço definido, o segundo passo é encontrar características que identifiquem, inequivocamente, cada forma.

No âmbito deste trabalho, e face ao número elevado de peças distintas existentes, escolheu-se tratar os tabuleiros metálicos já que, por si só, cobrem uma elevada percentagem do tipo de peças a processar. Estes podem ter uma superfície lisa ou ondulada e, dentro desta última classe, o ondulado pode ser horizontal ou vertical. Qualquer uma destas formas pode aparecer a qualquer momento na linha de pintura e com diferentes dimensões. Deve retirar-se informação sobre a orientação da peça, comprimento, altura e largura, e ainda qual o tipo de superfície.

4.1 Representação Matricial

As reconstruções 3D apresentadas na secção anterior não são mais do que uma nuvem de pontos no espaço. Do ponto de vista de processamento, esta (des)organização dos dados dificulta a extracção de medidas e características além de tornar os algoritmos ineficientes.

Por este motivo, os pontos obtidos na detecção de linhas são usados, simultaneamente, para a reconstrução 3D em tempo real e para a construção de uma matriz a que demos o nome de HeightMatrix — HgtMat. Nesta estrutura de dados condensamos toda a informação tridimensional: os índices (i,j) da matriz têm uma relação directa com as coordenadas (y,z) no mundo enquanto que o valor de HgtMat[i,j] representa a coordenada x (profundidade). A Figura 4.1 ilustra esta representação: a matriz opera como um espaço discretizado do mundo.

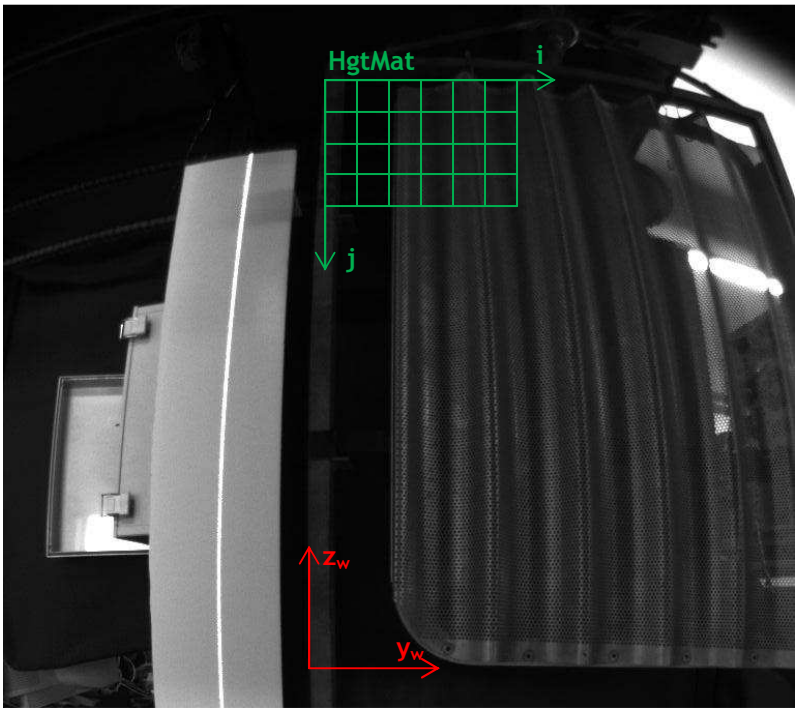

Figura 4.1 Esquema da relação entre HgtMat e o referencial do mundo

A Equação (4.1) descreve a relação entre os índices da matriz e coordenadas no mundo:

$$\begin{bmatrix} y \\ z \end{bmatrix} = \begin{bmatrix} y_{\min} \\ z_{\min} \end{bmatrix} + \begin{bmatrix} i \\ -j \end{bmatrix} * Sc \tag{4.1}$$

y_{min} e z_{min} são as coordenadas do ponto correspondente à origem da matriz, $(i,j) = (0,0)$. S_c é o factor de escala que transforma um índice num comprimento no mundo, isto é, representa a resolução do matriz. A resolução escolhida foi de 1mm, visto que o processo de pintura não necessita de precisão muito elevada. Por outro lado, ao duplicarmos a resolução, todos os demais algoritmos apresentados neste capítulo sofreram um forte agravamento em termos de tempo de execução.

Os índices máximos da matriz são dados por:

$$\begin{cases} i_{max} = \left(\max_{y} Points3D - \min_{y} Points3D\right)/S_c \\ j_{max} = \left(\max_{z} Points3D - \min_{z} Points3D\right)/S_c \end{cases} \tag{4.2}$$

onde $Points3D$ se refere à estrutura de dados que contém todos os pontos obtidos na detecção de linhas, em coordenadas do mundo. Os valores i_{max}e j_{max} são, ainda, arredondados por excesso.

O preenchimento da matriz é feito encontrando o índice (i,j) para cada entrada da estrutura $Points3D$:

$$\begin{bmatrix} i' \\ j' \end{bmatrix} = \begin{bmatrix} y - y_{\min} \\ z - z_{\min} \end{bmatrix} \Big/ Sc / \tag{4.3}$$

Como é intuitivo, o par (i', j') obtido nesta transformação encontra-se algures entre índices inteiros da matriz, como de pode observar na Figura 4.2.

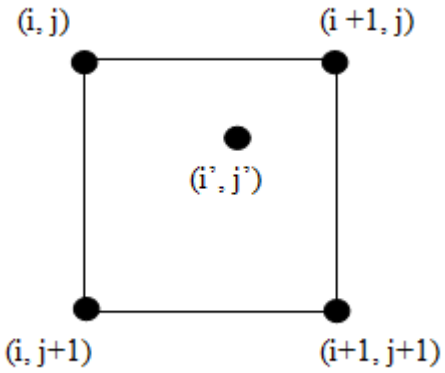

Figura 4.2 Preenchimento da matriz de alturas

O algoritmo de preenchimento de HgtMat vai adicionar a cada um dos índices adjacentes a (i', j'), representados na figura anterior, o valor da coordenada x do ponto correspondente (a informação de altura do objecto) e manter um registo do número de pontos somados em cada entrada (já que vários pontos podem e vão de facto contribuir para uma mesma posição). No final do preenchimento, o valor de cada posição é dividido pelo número de pontos adicionados.

À semelhança do que foi visível nas reconstruções 3D apresentadas na secção 3.3, também na matriz de alturas aparecem as zonas mortas, zonas sem qualquer informação. No entanto, o algoritmo baseado na Equação (4.3) provoca com que apareçam pontos de altura nula na matriz; não são pontos que o laser não alcança mas sim pontos na matriz para os quais não houve correspondência directa de nenhum ponto (por exemplo, devido aos arredondamentos) embora a maior parte das entradas em seu redor tenham sido preenchidas.

Para colmatar estas falhas, implementou-se uma rotina para filtrar a matriz de alturas aplicando um operador de média:

$$meanFilter = \begin{bmatrix} \frac{1}{8} & \frac{1}{8} & \frac{1}{8} \\ \frac{1}{8} & 0 & \frac{1}{8} \\ \frac{1}{8} & \frac{1}{8} & \frac{1}{8} \end{bmatrix} \tag{4.4}$$

Um cuidado adicional deve ser tomado para que este algoritmo não cause uma expansão do objecto e não se crie informação onde, originalmente, não existe nenhuma. Assim, a aplicação do filtro é restringida à existência de um número de elementos válidos na vizinhança

do ponto, isto é, dos vizinhos considerados, a maioria terá de apresentar um valor não nulo e o operador *meanFilter* é ajustado para só ter em conta não toda a vizinhança mas apenas os elementos diferentes de zero.

Para os cálculos envolvendo esta matriz considera-se, deste ponto em diante, que uma entrada da matriz $HgtMat[i,j]$ é válida caso o seu valor seja maior do que *minHgt:* como a peça passa no *conveyor* a uma distância nunca inferior a 10 cm da origem do referencial do mundo, um ponto só será considerado válido (com informação relevante) se apresentar um valor em x superior a esse limite. Este procedimento ajuda, também, a eliminar ruído.

4.2 Determinação dos Limites e Orientação

A partir da representação matricial, é possível agora, e com maior facilidade, extrair características do objecto.

Começamos por calcular o centro de massa, $CM = (c_y, c_z)$, do objecto para termos uma primeira ideia da posição deste — assumindo a simetria das peças, podemos usar o centro de massa como referência nos algoritmos seguintes, por exemplo, filtrar os pontos fronteira da peça usando como referência a posição relativa ao centro de massa.

O centro de massa não é mais que uma média das posições ponderada pelos respectivos pesos e, neste caso, atribuímos igual peso a cada ponto — $m(y,z) = 1$ caso o ponto seja válido ou $m(y,z) = 0$ caso em que o ponto tem relevo nulo. Assim temos:

$$c_y = \frac{\sum_y \sum_z y \cdot m(y,z)}{\sum_y \sum_z m(y,z)} \tag{4.5}$$

$$c_z = \frac{\sum_y \sum_z z \cdot m(y,z)}{\sum_y \sum_z m(y,z)} \tag{4.6}$$

O passo seguinte é encontrar os pontos de fronteira do objecto. Como se pode ver na Figura 4.3, todas as peças estão fixas a um caixilho metálico que efectua o transporte e suporte da mesma ao longo do *conveyor*.

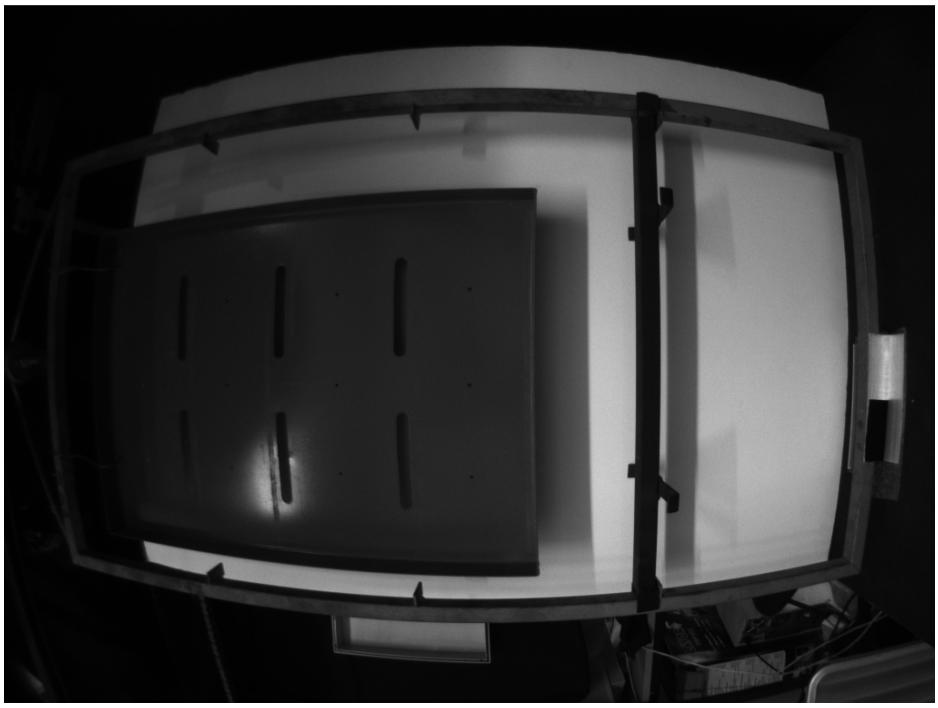

Figura 4.3 Armação metálica de suporte em torno de um tabuleiro liso

Tendo conhecimento, à partida, da existência deste objecto e assumindo que qualquer peça é transportada num sistema semelhante, foi implementado um algoritmo de pesquisa dos pontos de fronteira da peça. A Figura 4.4 mostra um esquema do princípio básico do algoritmo.

Os pontos são agrupados face ao lado da peça a que dizem respeito, isto é, temos quatro subconjuntos com os pontos limite de cada lado do objecto.

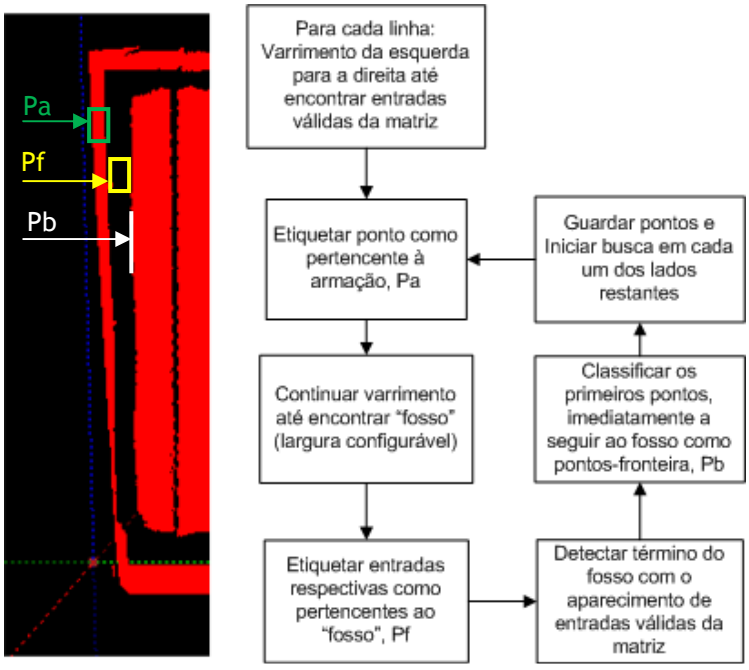

Figura 4.4 Esquema de Pesquisa da periferia do objecto

Na Figura 4.5 apresentam-se os resultados obtidos com o procedimento atrás apresentado. Devido à presença de elementos de suporte da peça, um dos lados contém bastante "lixo", isto é, pontos erradamente considerados como fronteira.

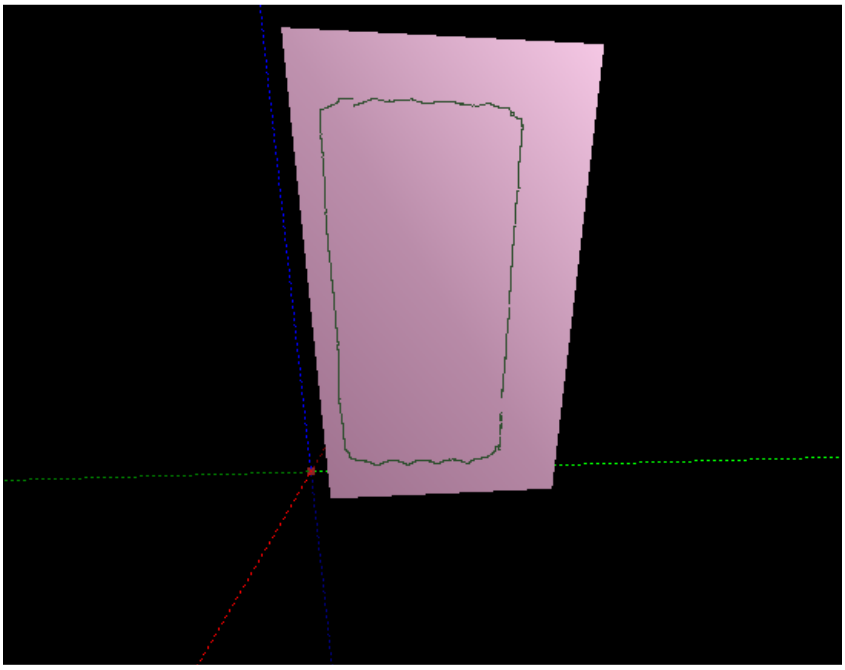

Figura 4.5 Pontos fronteira de um tabuleiro

Outro parâmetro que necessita ser definido é a orientação de peça. Como se pode verificar na Figura 4.6, os tabuleiros "viajam" sempre inclinados devido ao modo como são presos ao caixilho metálico. Calcular o plano que melhor aproxima o tabuleiro permite caracterizar esta inclinação e definir um sistema de eixos da peça que será útil na altura de desenvolver o programa para o manipulador robótico.

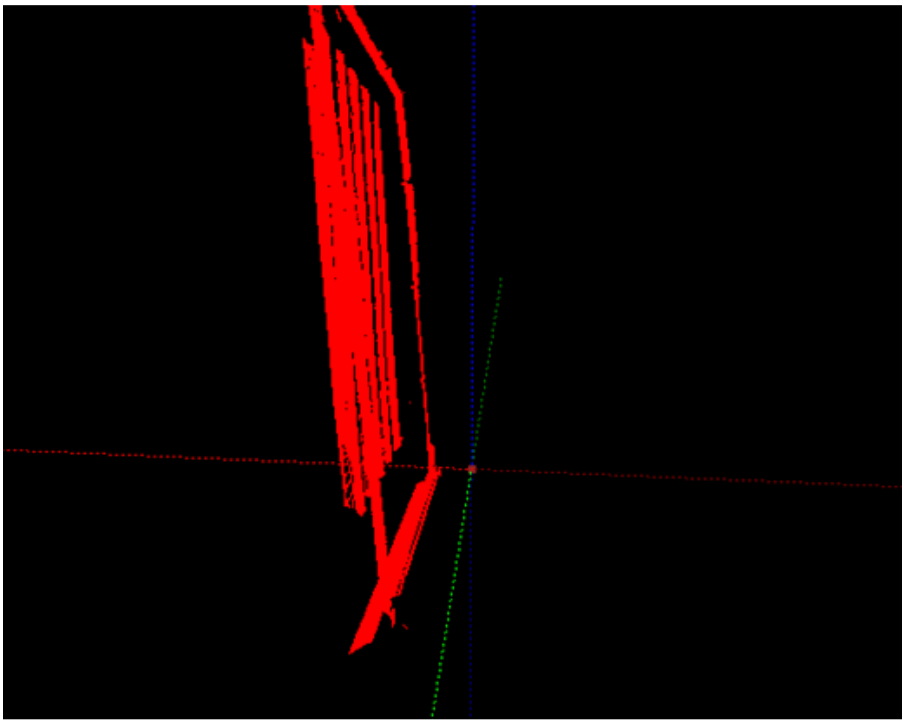

Figura 4.6 Visualização da peça e do perfil oblíquo em relação à vertical

Como já foi visto anteriormente, um plano pode ser definido como:

$$NV = D \tag{4.7}$$

onde *N* é um vector normal ao plano e $D = NP_D$ em que P_D é um ponto pertencente ao plano.

Partindo de um conjunto de pontos, designados de ora em diante de $V_i = [x_i, y_i, z_i]^T$, achamos os correspondentes vectores $(V_i - P_D)$ — que são os vectores que unem um ponto conhecido do plano e o nosso conjunto de pontos; estes podem ser tratados como duas contribuições distintas: uma formada pela componente de cada vector que ficará contida no plano (projecção ortogonal) e uma segunda contribuição, normal ao plano, responsável pelo erro total de aproximação do plano aos pontos. Desta forma, e assumindo os vectores *N e* $N_i^{\perp}$ normalizados, podemos escrever:

$$V_i = P_D + \lambda_i \mathrm{N} + k_i N_i^{\perp} \tag{4.8}$$

onde $\lambda_i = N^T(V_i - P_D)$ e $k_i N_i^{\perp}$ reflecte a componente de V_i perpendicular a *N*. Fazendo $W_i = V_i - P_D$ e no sentido de estimar um plano pelos mínimos quadrados, é trivial reconhecer, a partir da Equação (4.8), a parcela que gera o erro, nomeadamente, $\lambda_i \mathrm{N}$. Por conseguinte, a função objectivo será:

$$\min \sum_{i=1}^{N} (W_i^{\mathrm{T}} N N^{\mathrm{T}} \mathrm{W_i}) \tag{4.9}$$

O mínimo da soma do quadrado dos erros pode ser, portanto, calculado achando o zero das derivadas parciais da Equação (4.9), $\partial/\partial P$ e $\partial/\partial N$. No entanto, foi seguida uma abordagem mais simples, sugerida em [17]. A partir do zero da derivada em ordem a P:

$$\begin{gathered} -2NN^{\intercal}\sum_{i=1}^{N} W_i = 0 \Leftrightarrow \\ \sum_{i=1}^{N} W_i = 0 \Rightarrow P_D = \left(1/N\right)\sum_{i=1}^{N} V_i \end{gathered} \tag{4.10}$$

concluímos que o ponto P_D é, na prática, a média dos pontos. Em termos de algoritmia, este facto consegue poupar algum processamento se, antes de aplicar os mínimos quadrados, centrarmos os pontos na origem — os novos pontos são representados por V_i^*.

Com o resultado anterior, e reescrevendo a Equação (4.9) numa forma alternativa:

$$\min\left[\, N^{\intercal}\left(\sum_{i=1}^{N} W_i\, W_i^{\intercal}\right)N\right] \tag{4.11}$$

A solução apresentada em [17], diz-nos que o mínimo desta função está associado ao valor próprio mais pequeno de $\sum_{i=1}^{N} W_i\, W_i^{\intercal}$ e o vector normal N será o respectivo vector próprio.

Para encontrar o vector próprio foi usada a SVD — decomposição em valores singulares — da matriz:

$$\sum_{i=1}^{N} W_i\, W_i^{\intercal} = \begin{bmatrix} \sum_{i=1}^{N}(x_i^*)^2 & \sum_{i=1}^{N}(x_i^* y_i^*)^2 & \sum_{i=1}^{N}(x_i^* z_i^*)^2 \\ \sum_{i=1}^{N}(x_i^* y_i^*)^2 & \sum_{i=1}^{N}(y_i^*)^2 & \sum_{i=1}^{N}(y_i^* z_i^*)^2 \\ \sum_{i=1}^{N}(x_i^* z_i^*)^2 & \sum_{i=1}^{N}(y_i^* z_i^*)^2 & \sum_{i=1}^{N}(z_i^*)^2 \end{bmatrix} \tag{4.12}$$

Resta definir quais os pontos V_i que serão usados. Aqui encontram-se algumas restrições:

- Poderíamos usar todos os pontos confinados à fronteira calculada anteriormente. No entanto, o algoritmo associado ao cálculo da SVD da matriz apresentada na equação (4.12) tornar-se-ia demasiado pesado em termos computacionais — lembramos que para uma área de $1m^2$, sem falhas, a matriz HgtMat pode conter 1 milhão de elementos.
- Por outro lado, não basta escolher aleatoriamente um qualquer número N de pontos (com N suportável em termos de execução temporal do algoritmo da SVD) do conjunto de pontos disponíveis. Vejamos a Figura 4.7**:** o facto de o plano do laser estar oblíquo em relação à peça origina que a reconstrução 3D não seja simétrica; um dos

lados consegue ser completamente varrido pelo laser (face esquerda na figura) enquanto o lado oposto é "inexistente" para o laser. A reconstrução resultante deste varrimento encontra-se na Figura 4.8 (temos uma "casca" do paralelepípedo correspondente às três faces visíveis). Na prática, este fenómeno conduz a uma estimação de um plano também ele oblíquo (Figura 4.9) enquanto o resultado ideal seria o apresentado na Figura 4.10, caso em que conseguimos determinar a orientação do objecto.

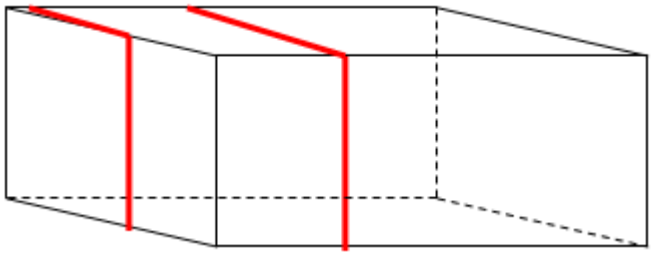

Figura 4.7 Zonas visíveis e ocultas ao laser

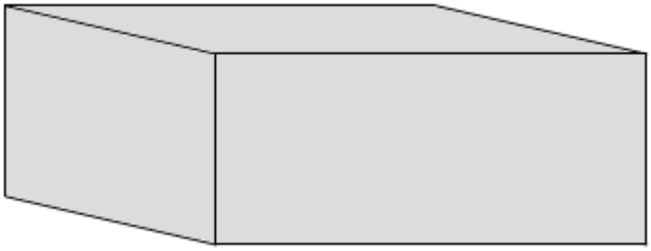

Figura 4.8 Reconstrução do objecto: pontos sobre 3 faces

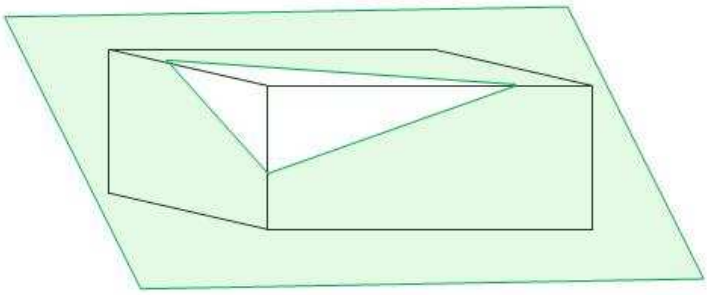

Figura 4.9 Aproximação por um plano ao conjunto de pontos da Figura 4.8 -

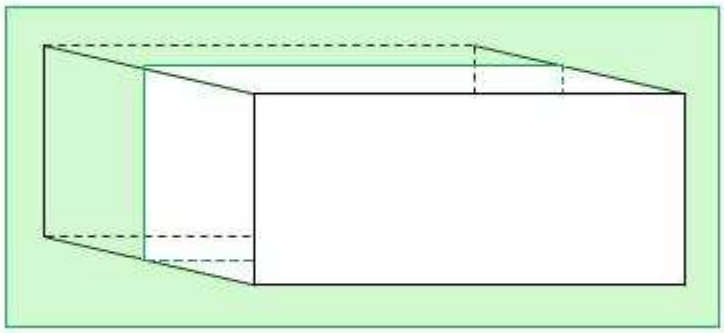

Figura 4.10 Aproximação ideal revelando a orientação do objecto

Para evitar estas situações, encontram-se os máximos locais (em relevo) da peça, começando por dividir esta em sectores e fazendo uma pesquisa ponto a ponto, em cada sector, guardando os N pontos com profundidade menor em cada quadrante

Os resultados são apresentados de seguida, na Figura 4.11 e Figura 4.12. Embora não haja nenhuma maneira expedita para medir a orientação da peça na realidade, podemos a partir da visualização 3D no software ter uma ideia da qualidade da aproximação, que se considerou suficientemente boa para, numa fase posterior, podermos construir um sistema de eixos de trabalho para o manipulador industrial a partir desta orientação.

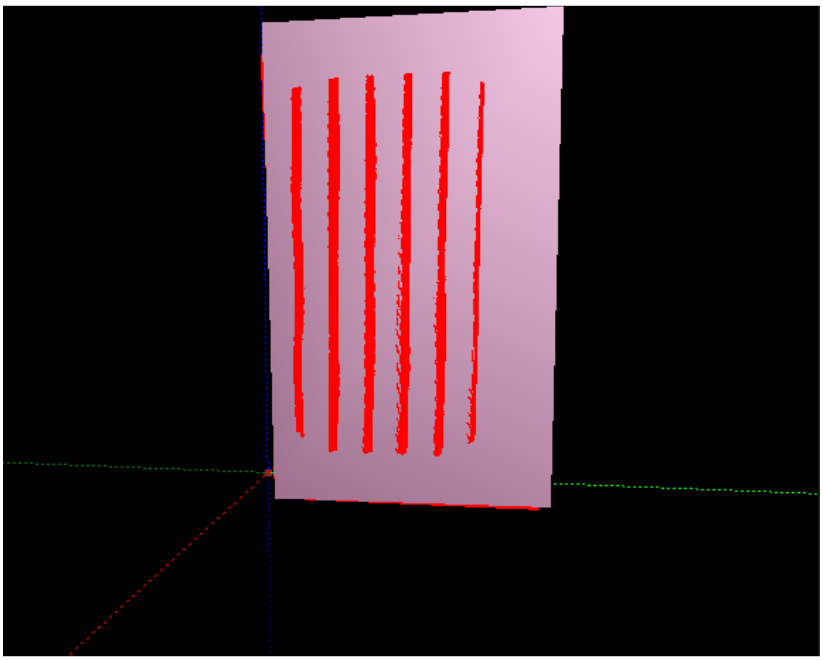

Figura 4.11 Representação de um plano com a inclinação aproximada de um tabuleiro ondulado: visão frontal

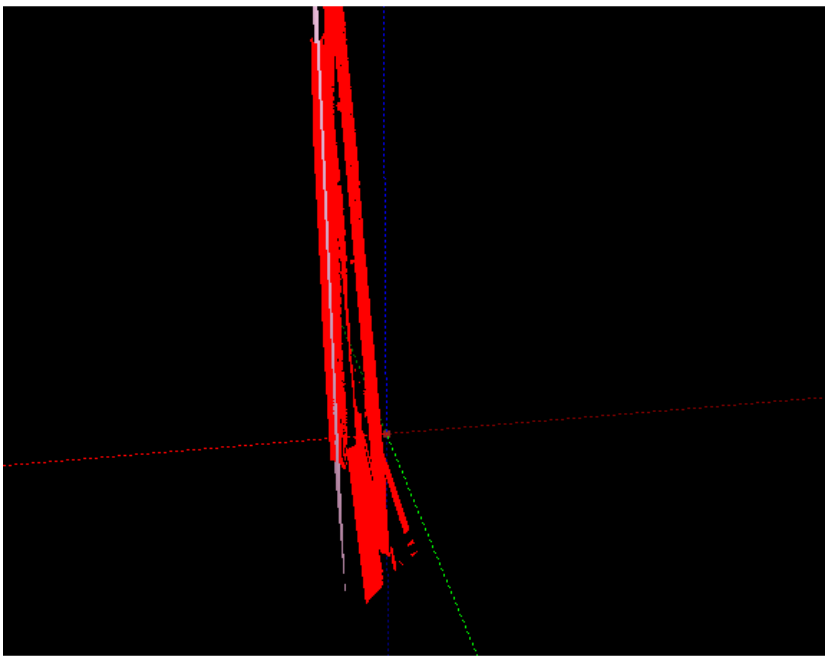

Figura 4.12 Representação de um plano com a inclinação aproximada de um tabuleiro ondulado: visão lateral

4.3 Distinção entre diferentes formas e cálculo da dimensão

Como já havia sido referido anteriormente, neste trabalho vamos limitar-nos a distinguir entre os vários tipos de tabuleiros (lisos, com ondulados horizontais e com ondulados verticais) embora se pretenda que o algoritmo desenvolvido não seja completamente redutor, ou seja, permita ser facilmente configurável para, mais tarde e em continuação deste trabalho, possam também ser distinguidas caixas, que constituem outro dos objectos frequentemente pintados na linha de produção.

O método mais expedito encontrado para resolver este problema foi analisar vários cortes, horizontais e verticais, do modelo 3D construído. Na Figura 4.13 e Figura 4.14 mostram-se várias amostras de cortes horizontais e verticais obtidos a partir da representação matricial da peça.

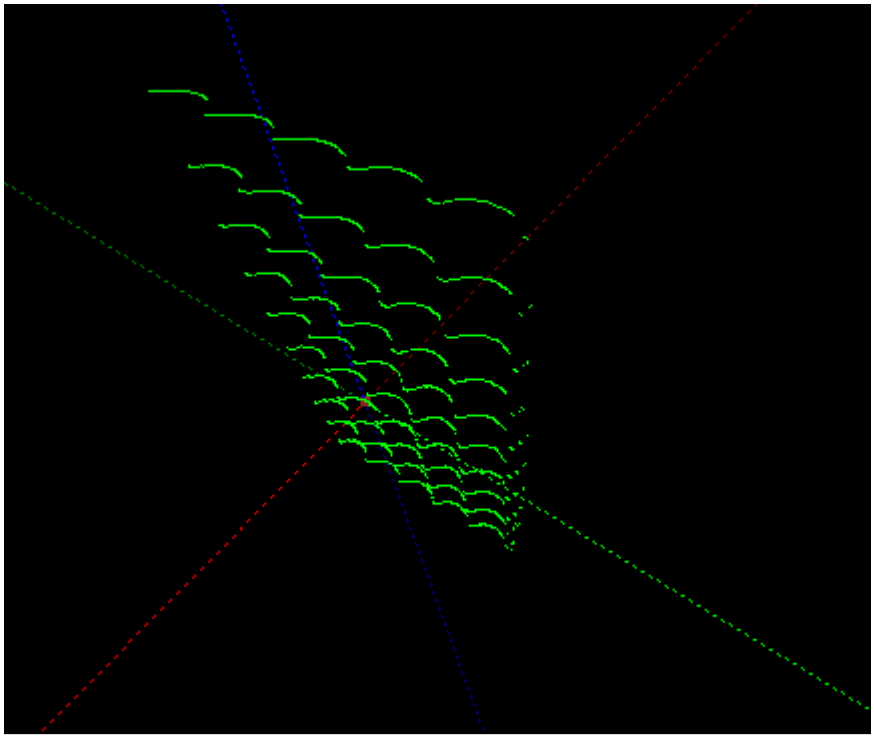

Figura 4.13 Visualização 3D dos cortes horizontais feitos a partir da representação matricial da peça

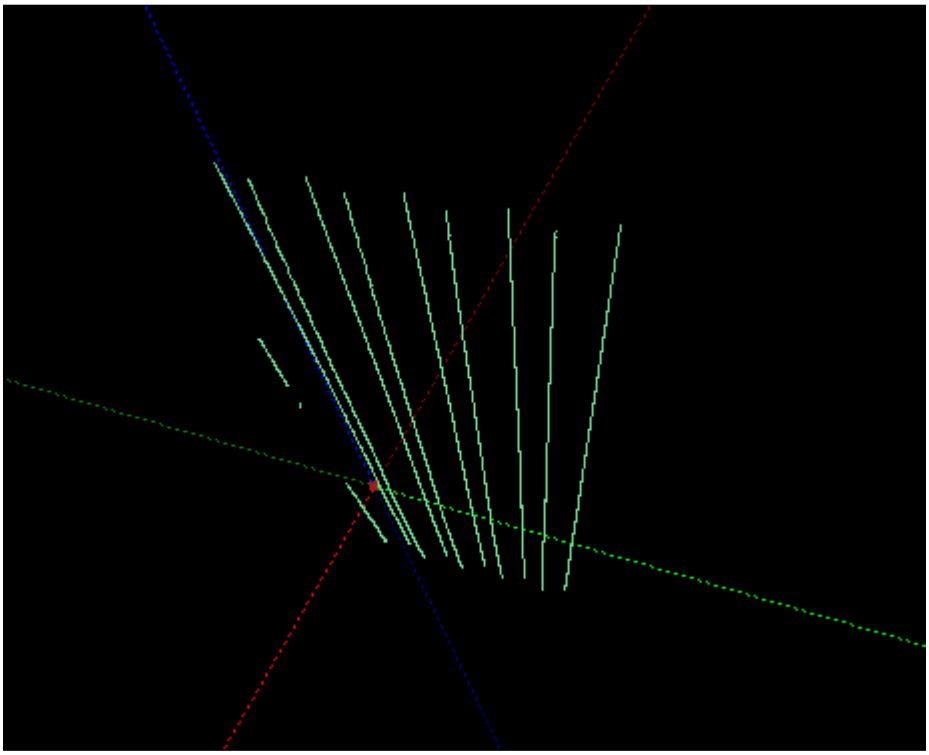

Figura 4.14 Visualização 3D dos cortes verticais feitos a partir da representação matricial da peça

Seja δ o desvio padrão da profundidade ao longo de um corte, sendo este definido por N_P pontos, com média de profundidade $\bar{x}_P$. A variância média da profundidade, δ_m^2, calculada a partir de *M* cortes, é dada por:

$$\delta_m^2 = \frac{1}{M}\sum_{j=1}^{M}\left(\frac{1}{N_c}\sum_{i=1}^{N_c}\left(x_i^j - \bar{x}_P^j\right)^2\right) \tag{4.13}$$

onde x^j refere-se a um ponto pertencente ao corte j e $\bar{x}_P^j$ à media dos pontos desse corte.

Esta informação permite caracterizar os tabuleiros na medida em que, aqueles com superfície ondulada, apresentam elevada variância em cortes horizontais ou verticais (caso o ondulado seja ele mesmo horizontal ou vertical) enquanto os tabuleiros lisos apresentam uma variância muito reduzida em qualquer direcção. Constroem-se, desta forma, dois indicadores:

- $\delta_{m_{\text{Horiz}}}^2$;
- $\delta_{m_{\text{Vert}}}^2$.

Compõe-se, então, um espaço cartesiano em que cada dimensão está associada a um indicador. Na Figura 4.15 esquematiza-se esta situação, numa representação genérica com três indicadores I_1, $_2I$, $_3I$.

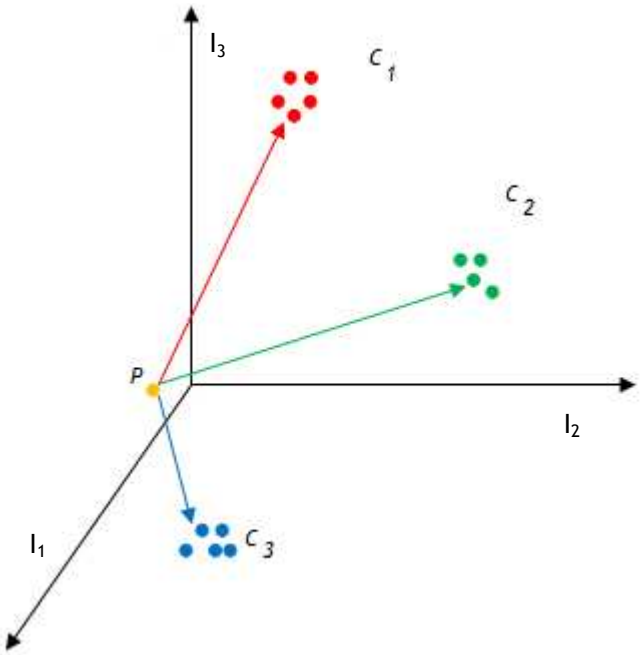

Figura 4.15 Classificação de peças — distância aos conjuntos existentes

Uma nova peça, P, é classificada de acordo com a proximidade aos conjuntos já existentes, no que se usa um algoritmo clássico de reconhecimento de padrões: k vizinhos mais próximos. Numa fase de treino, recolhem-se dados sobre um determinado número de peças, construindo-se assim, uma base de dados para comparação. À chegada de uma nova peça, são calculados os indicadores respectivos. Procede-se ao cálculo da distância euclidiana a cada um dos pontos existentes na base de dados. Os k pontos mais próximos são usados para classificar a peça, que será a forma com mais presenças nesses k pontos.

Convém notar que, para este trabalho, embora se tenham conseguido distinguir os diferentes tipos de tabuleiros, não existiam peças em número suficiente para se criarem conjuntos bem definidos e uma base de dados rica, uma condição essencial para a robustez do algoritmo. No entanto, este método é bastante atractivo tendo em vista a expansão futura para mais formas, já que existe a facilidade de aumentar a dimensão do espaço tanto quanto se queira para garantir uma boa classificação. Usou-se, perante a falta de dados de treino, k = 1, o que significa que cada peça é classificada com o tipo da peça que está mais próxima de si no espaço gerado pelos indicadores.

O cálculo das dimensões, como só tratamos de tabuleiros, resume-se a achar um rectângulo que melhor ajusta os pontos de fronteira, anteriormente calculados.

Existem vários objectos que interferem com a pesquisa dos pontos fronteira, nomeadamente, os ferros onde a peça é encostada e os ganchos metálicos de suporte. Deste modo, partindo desses pontos, escolhemos apenas uma pequena porção: dividimos a peça em bandas e escolhemos apenas um ponto por banda, correspondente ao ponto mais exterior — ver Figura 4.16. Destes pontos obtidos, ainda desprezamos todos os que se desviam da média por mais de 1.5 vezes o desvio padrão e, finalmente, achamos a média dos pontos para cada lado da peça.

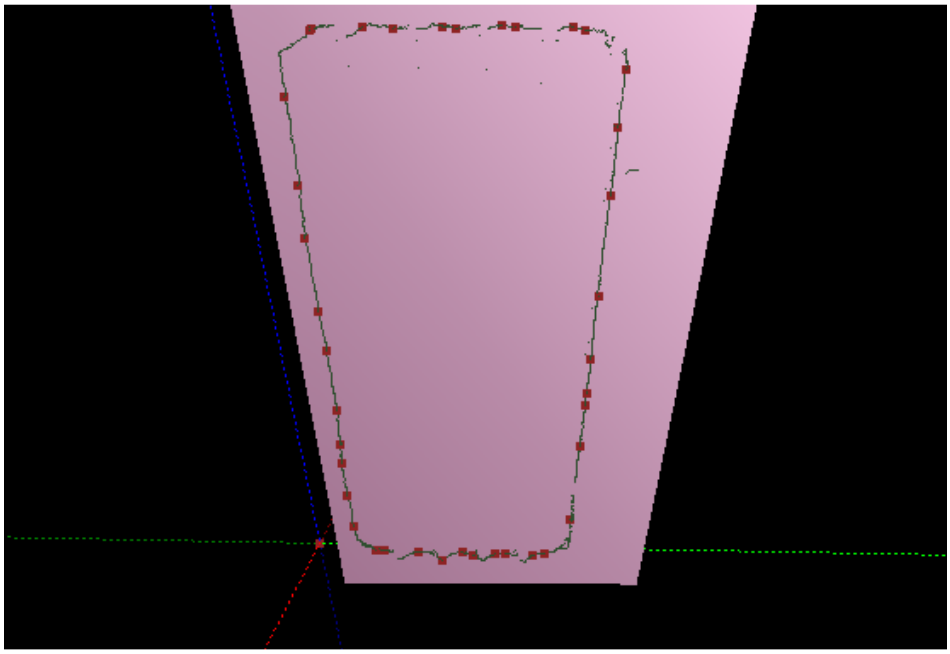

Figura 4.16 Pontos fronteira efectivamente considerados para cálculo das dimensões

Os resultados obtidos até esta fase permitiram concluir da adequabilidade dos procedimentos seguidos para determinar os tipos de peça e respectivas dimensões. As medidas geradas contêm erros que podem chegar a 2% do tamanho da peça, o que se apresenta completamente aceitável. No entanto, futuramente, os métodos apresentados podem ser melhorados pelo que podemos esperar ainda melhores resultados.

Capítulo 5

Programação do Manipulador Industrial

De forma a completar o sistema de pintura robotizada, faltam gerar programas para o manipulador industrial, capazes de pintar os diversos tabuleiros, assim como assegurar que estes sejam pintadas quaisquer que sejam as suas dimensões.

A estratégia adoptada foi desenvolver um conjunto de programas genéricos que pintem um conjunto limitado de formas e conseguir escolher correctamente qual programa utilizar em função da peça que transita na linha de produção. A informação retirada dos modelos 3D que, nesta etapa, já foram obtidos, servirá este propósito. As dimensões dos tabuleiros, parâmetros também já estimados, são comunicadas ao controlador do manipulador industrial por um protocolo apropriado, informação esta que será usada em cada programa genérico para que este se adapte às dimensões dos objectos.

5.1 Protocolo de Comunicação

O controlador NX100 da MOTOMAN tem a capacidade de comunicar por Ethernet com equipamentos remotos, sobre o protocolo BSC. Trata-se de um protocolo vocacionado para ligações ponto-a-ponto, half-duplex.

Assim, foi necessário integrar no software desenvolvido a base de comunicação deste protocolo, processo que se esquematiza na Figura 5.1.

Todos os pacotes associados à criação e término da ligação são pré-configurados. Em *BSC packet* seguem os comandos que pretendemos enviar.

Para esta transmissão de dados, temos 3 modos:

- DCI (*Data Communication By Instruction*) — a transmissão de dados entre o controlador e um PC externo é gerida pelo próprio programa do manipulador uma vez que neste modo o envio ou recepção de dados estão associados a instruções próprias do programa.
- *Stand-Alone Function* — este modo encarrega-se da transmissão de dados a partir da operação na consola, isto é, o operador solicita na consola o tipo de informação que pretende receber ou enviar;

- *Host Control Function* —esta função serve para carregar programas, ler o estado do robô e controlar o seu movimento pelo envio de comandos a partir do computador externo.

A função que se adequa ao que pretendemos fazer é, nitidamente, o *Host Control Function*, já que nos permite escolher o programa que o manipulador deve executar, escrever nas variáveis do controlador, definir sistemas de coordenadas e ferramentas, e ainda controlar o robô a partir da nossa aplicação.

Para além das instruções típicas de controlo do robô (MOVL, MOVC, MOVJ), podemos usar os comandos:

- RPOSC e RPOSJ para saber qual a posição actual do robô em relação a qualquer sistema de eixos;
- SAVEV e LOADV para o acesso às variáveis do controlador; podemos assim escrever em memória as dimensões das peças entre outros valores que sejam eventualmente necessários;
- JSEQ para correr um programa que esteja em memória, o que nos permite optar entre os programas genéricos associados a cada peça.

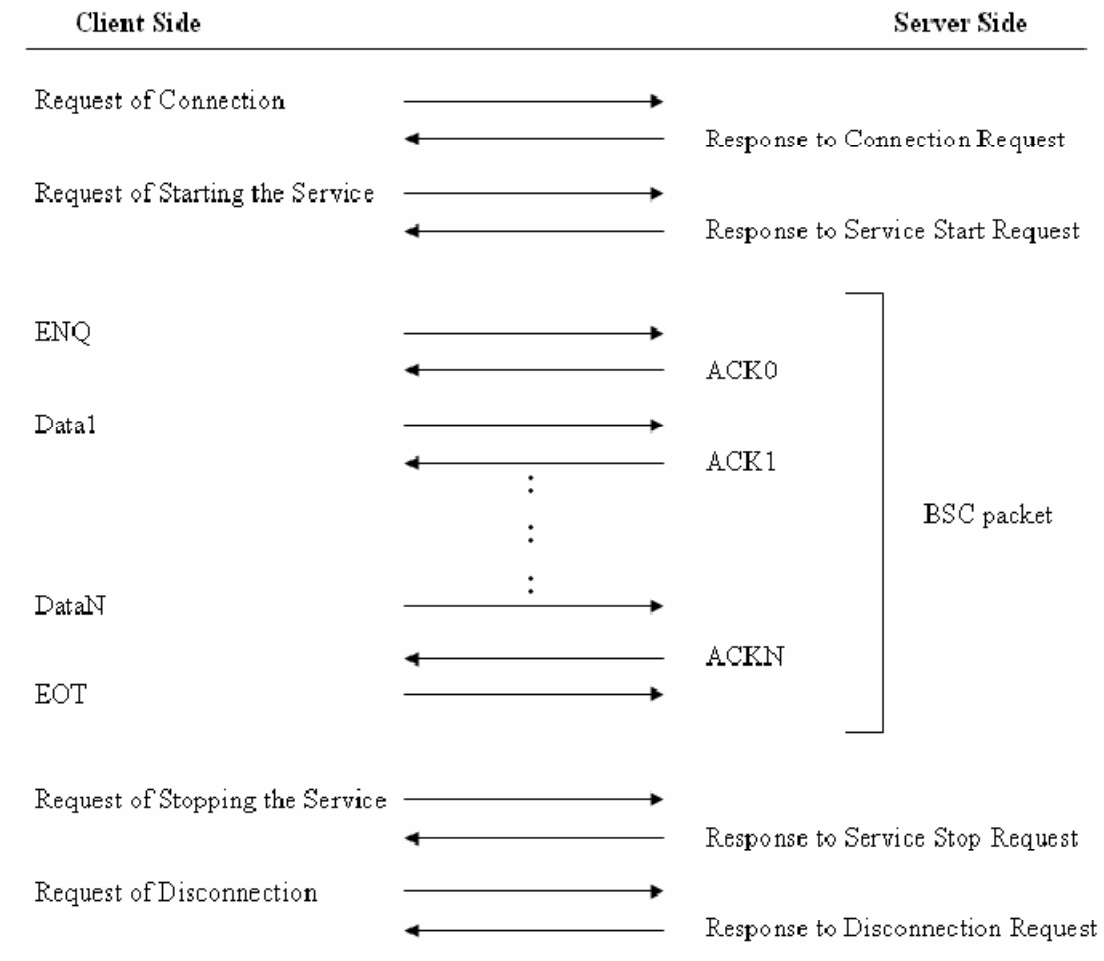

Figura 5.1 Esquema de troca de dados numa ligação pelo protocolo BSC [18]

Na medida em que, normalmente, efectuar um pedido ao robô (por exemplo, para descrever um movimento circular) engloba vários comandos, como accionamento dos s*ervomotors,* instrução MOVC e respectivas respostas por parte do manipulador, o programa de comunicação encontra-se estruturado como uma máquina de estados, que gere o envio dos vários comandos e processa as respostas reportando eventuais erros.

5.2 Programas Genéricos e Adaptação

Para o caso dos tabuleiros, sendo ondulados ou lisos, o processo de pintura considerado resume-se a conjunto de varrimentos verticais ou horizontais de forma a cobrir toda a área da peça. Trata-se de uma versão simplificada que pretende conseguir integrar os resultados dos módulos já desenvolvidos numa solução de pintura adaptável. Para uma pintura completamente fiel à executada pelos operadores da Flupol, será necessário utilizar sistemas de captura de movimento gravando em vídeo todo o processo e desenvolver programas mais detalhadas. Tal não era o objectivo fundamental deste trabalho, mas é um passo já pensado em trabalho futuro, pelo que se optou por desenvolver programas simples que provem a capacidade de adaptação com recurso aos dados já disponíveis.

Foram, por isso, desenvolvidos dois programas base para o manipulador cujo princípio se esquematiza na figura seguinte.

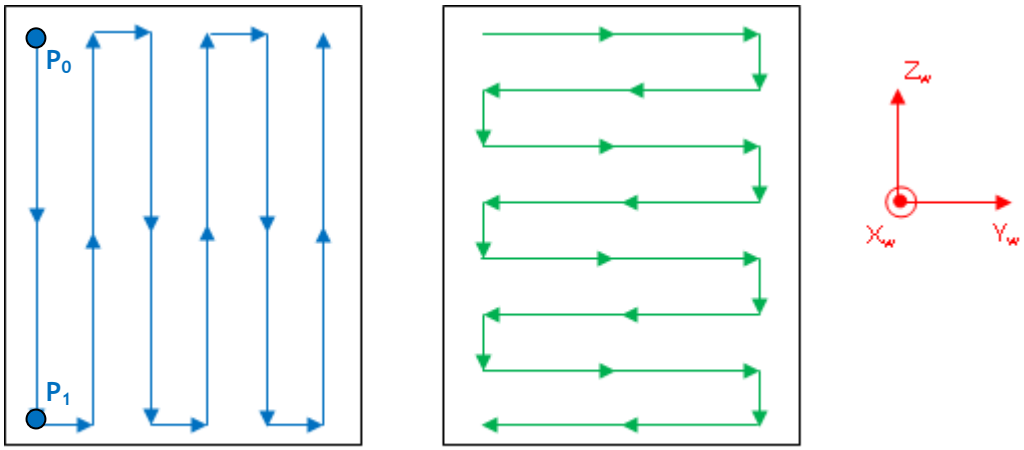

Figura 5.2 Esquema de pintura dos tabuleiros. À esquerda: tabuleiro ondulado horizontal; à direita: tabuleiros liso e com ondulado vertical

Foi definido um sistema de coordenadas para o manipulador correspondente ao que tem sido referido, até agora, o referencial do mundo. Os programas foram desenvolvidos assumindo movimentos paralelos aos eixos deste referencial (ver Figura 5.2).

A partir das medidas das dimensões já obtidas, podemos estabelecer a amplitude dos movimentos verticais e horizontais, assim como o número de vezes que será repetido cada movimento.

Seja mH e mV as amplitudes horizontal e vertical do movimento, respectivamente, ou seja, segundo os eixos Y_w e Z_w.

No caso do tabuleiro ondulado horizontal (representado à esquerda na Figura 5.2) temos:

- mV equivale à altura do tabuleiro;
- mH será o valor correspondente à amplitude do feixe da pistola de pintura (daqui em diante referido como *SpW*); como neste trabalho apenas se simulou o acto de pintar, não se tendo usado, efectivamente, uma pistola própria, assumiu-se o valor de *SpW* = 5cm;
- $Nmov$ será o número de movimentos horizontais que precisamos para cobrir toda a área da peça e é calculado directamente a partir de *SpW* e da largura do tabuleiro (*tabWid*): $Nmov = tabWid \; \boldsymbol{div} \; SpW$ onde o operador ***div*** significa a divisão inteira;

Estas três variáveis são enviadas para o controlador do robô, através do protocolo descrito na secção anterior. O programa de pintura resulta numa reprodução sequencial de movimentos verticais e horizontais até ao limite de $Nmov$ movimentos horizontais, altura pela qual toda a peça estará pintada. O diagrama da Figura 5.3 descreve a execução do programa de pintura:

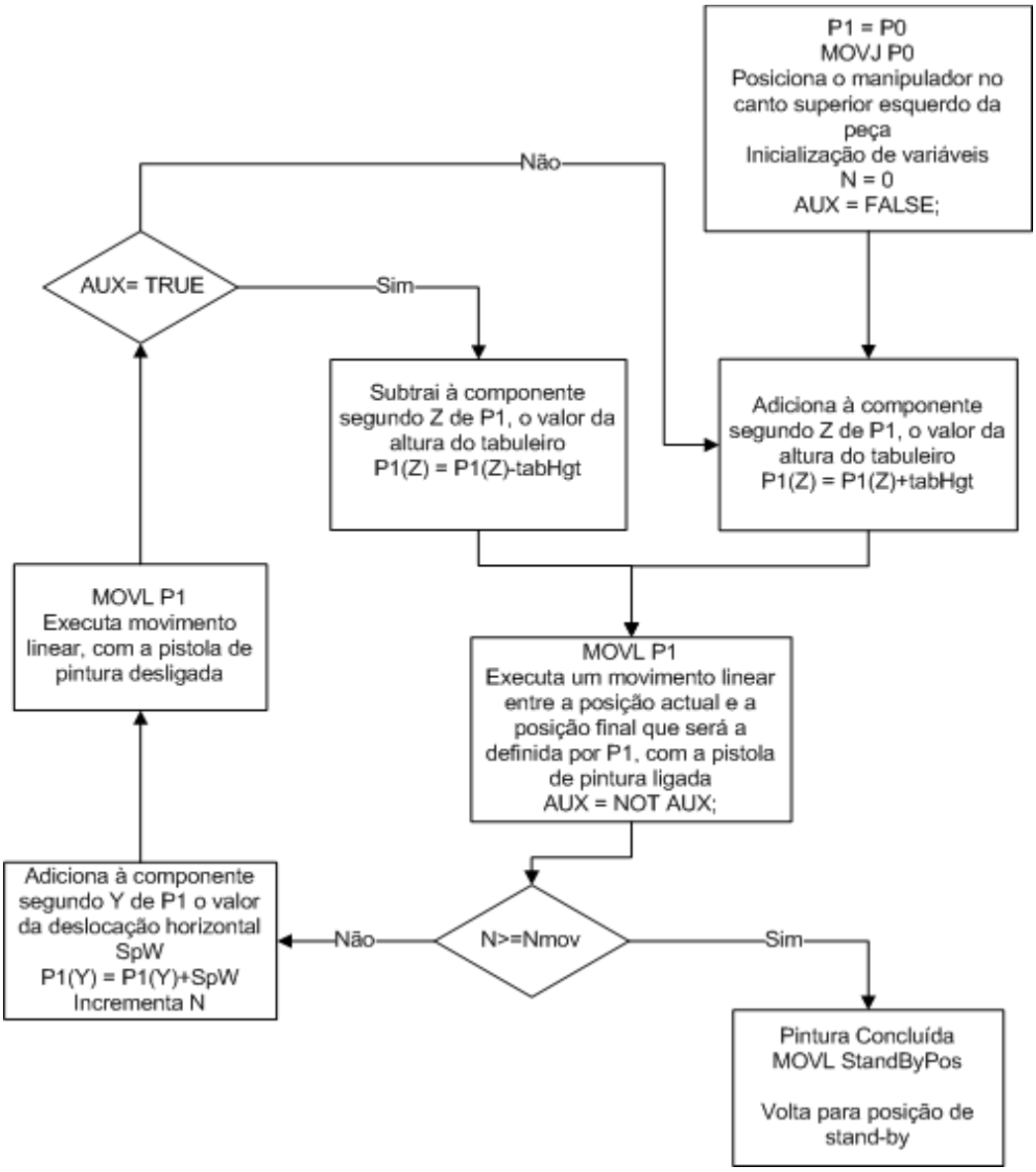

Figura 5.3 Fluxograma do programa de pintura para um tabuleiro com ondulado horizontal

A mesma estrutura é usada no programa para pintar tabuleiros ondulados verticais e lisos sendo que os movimentos horizontais são a toda a largura da peça e os verticais de acordo com a largura do feixe de tinta.

O programa actualiza constantemente uma variável de posicionamento do robô (P1) com os valores das dimensões anteriormente transmitidos.

Relembramos agora que as peças são transportadas com alguma inclinação face à vertical, devido aos suportes. Usamos, aqui, o resultado obtido na secção (4.2), em que definimos um plano com a orientação da peça, e estabelecemos um sistema de eixos alinhado com a peça. Mantendo o eixo dos YY, que coincide com a direcção do movimento do *conveyor* (e porque assumimos que a peça, apesar de inclinada, não se encontra "torcida", isto é, não existe rotação segundo Z_w), o vector normal ao plano de orientação da peça dá-nos a informação restante para que possamos definir o novo referencial (ver Figura 5.4).

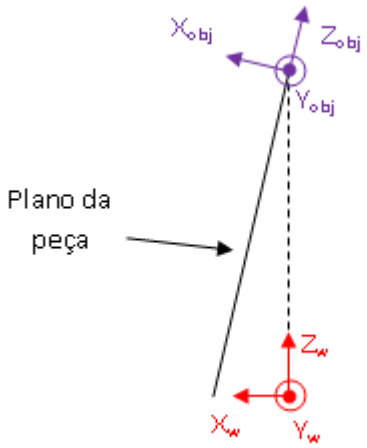

Figura 5.4 Referencial do mundo e referencial da peça

Os programas genéricos, aplicados a cada peça em particular, pintam, assim, de acordo com a inclinação da peça e não paralelamente aos eixos do mundo como foram programados inicialmente.

5.2.1 Sincronização com o *conveyor*

Para conseguir que o robô pinte as peças com a linha em movimento, foi adquirido um sistema de sincronização disponível no fabricante do manipulador. Este sistema consta de um encoder, o qual é acoplado ao motor do *conveyor*, e de um interruptor que permite fazer o *reset* à contagem do encoder.

Esta solução é de aplicação directa, isto é, existe já da parte do controlador NX100 a capacidade de reconhecer e utilizar automaticamente este sistema. No entanto, uma série de instruções, ao nível do código do programa de pintura, têm de ser modificados para que a informação do *encoder* tenha efeito.

Pela altura da elaboração deste documento, esta modificação ainda não tinha sido levada a cabo pelo que apenas se pintaram as peças com a linha em repouso, no entanto é uma tarefa relativamente simples de realizar no futuro.

Capítulo 6

Conclusão

Constituí-se, com sucesso, um módulo de visão artificial que permite a recolha dos perfis 3D de objectos. Não confinado ao uso apenas neste projecto, este módulo pode ser reutilizado em diversas aplicações com o mesmo princípio, exigindo um mínimo de esforço de reconfiguração. O erro de 2% na realização do mapeamento 3D é bastante aceitável para a aplicação de pintura, sendo que a fonte da maior percentagem deste erro está localizada ao nível do processo de calibração, processo este que pode ser melhorado e substituído a qualquer altura sem prejuízo das restantes funcionalidades desenvolvidas.

Deve-se notar ainda que módulo foi conseguido através de tecnologias de baixo custo quando comparadas com as soluções integradas disponíveis no mercado.

Desenvolveu-se, com sucesso também, um módulo de reconhecimento de objectos e extracção de características a partir de reconstruções 3D dos mesmos. Apesar de, neste momento, a capacidade seja apenas a de distinguir tabuleiros de diferentes feitios e tamanhos (tal era o objectivo deste trabalho), a estrutura encontra-se aberta à integração directa de novas funcionalidades, como a de distinguir caixas, latas cilíndricas, etc, sem a necessidade de efectuar alterações de maior à aplicação.

Por fim, e relativamente à componente de pintura robotizada, conseguiu-se desenvolver um método de ajuste automático de programas de forma a ir ao encontro das diferentes tarefas de pintura consoante o tipo de peça a pintar. Desenvolveram-se programas simples de pintura de tabuleiros e estabeleceu-se a ponte entre a componente de extracção de características dos objectos e o manipulador: a aplicação desenvolvida tem a capacidade de comunicar com o robô transmitindo ordens para execução de movimentos simples, para execução de programas completos ou apenas fornecendo dados sobre as peças (comprimento, altura, largura e orientação) para que os programas existentes possam ser adaptados.

Perante os resultados esperados à partida para este trabalho, podemos dizer que os objectivos foram cumpridos tendo-se, de facto, desenvolvido uma solução completa para pintura robotizada com adaptação automática a objectos com formas e dimensões distintas (dentro de um conjunto limitado). Recorreu-se, fundamentalmente, ao caso específico da Flupol para a validação do trabalho realizado.

6.1 Trabalho Futuro

Na perspectiva de dar seguimento a este projecto para que possa ser realmente aplicado na unidade de produção da Flupol, podem ser pensadas algumas melhorias.

O método de calibração pode ser substituído por outro mais robusto, capaz de identificar todos os parâmetros extrínsecos e intrínsecos da câmara. Sugere-se o método de Tsai pela sua facilidade de implementação e visto não necessitar nenhuma preparação relevante do cenário, apenas algumas folhas com marcadores. Esta modificação conduzirá, de certo, a uma maior precisão na recolha de dados 3D.

De modo a obter medições de dimensões também mais precisas, o algoritmo de pesquisa de fronteiras do objecto pode ser melhorado, nomeadamente, implementar um método baseado em *snakes* (também chamado de "contornos activos") para melhor delinear as peças com maior imunidade ao ruído.

Reserva-se também para trabalho futuro o desenvolvimento do módulo de identificação das peças, dotando-o da habilidade de distinguir um número maior de formas.

O algoritmo de identificação deve ser tornado mais robusto treinando-o com mais informação, isto é, classificando várias peças que serão adicionadas aos conjuntos existentes. Deve-se estudar o efeito da quantidade k, de peças que são usadas para proceder à identificação de uma nova peça.

O processo de pintura pode, igualmente, ser melhorado tendo em conta os gestos usados pelos operadores na fábrica. O processo de pintura deve ser gravado, para cada peça, de modo a que os programas genéricos possam ser desenvolvidos tendo em conta essa informação, obrigando o manipulador, dentro do possível, a executar gestos semelhantes.

Referências

[1] Hiroshi Kawasaki, Ryo Furukawa “Dense 3D Reconstruction method using Coplanarities and Metric Constraints for Line Laser Scanning”, Sixth International Conference on 3-D Digital Imaging and Modeling, 2007

[2] David Acosta, Olmer García , Jorge Aponte “Laser Triangulation for shape acquisition in a 3D Scanner Plus Scanner”, Proceedings of the Electronics, Robotics and Automotive Mechanics Conference (CERMA'06), 2006

[3] Michaël Demeyere, Déo Rurimunzu, Christian Eugène “Diameter Measurement of Spherical Objects by Laser Triangulation in an Ambulatory Context”, IEEE Transactions on instrumentation and measurement, VolL. 56, N°. 3, 2007

[4] J.Wu , J.S.Smith, J. Lucas “Weld bead placement system for multipass welding”, IEE Proc.-Sei. Meas. Technol., Vol. 143, No. 2, 1996

[5] Reinhard Noli, Michael Krauhausen “Multi-beam laser triangulation for the measurement of geometric features of moving objects in production lines” , Conference on Lasers and Electro-Optics Europe, 2003. CLEO/Europe, 2003

[6] Xiangyin Ma, Hongbin Zha “Hybrid Scene Reconstruction by Integrating Scan Data and Stereo Image Pairs”, Sixth International Conference on 3-D Digital Imaging and Modeling 2007

[7] Wang Lei, Bo Mei, Gao Jun, Ou ChunSheng “A Novel Double Triangulation 3D Camera Design”, Proceedings of the IEEE International Conference on Information Acquisition, 2006

[8] S. Burak Gokturk, Hakan Yalcin, Cyrus Bamji “A Time-Of-Flight Depth Sensor – System Description, Issues and Solutions”, Conference on Computer Vision and Pattern Recognition Workshop, 2004

[9] Andreas Ullrich, Nikolaus Studnicka, Johannes Riegl “Long-range high-performance time-of-flight-based 3D imaging sensors”, Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission , 2002

[10] Ana Ferreira "Extracção de Dimensões de Objectos por Laser para Integração com Manipuladores Industriais", Relatório de Projecto para Obtenção de Grau de Licenciatura pela Faculdade de Engenharia da Universidade do Porto, 2007

[11] Datasheet da câmara DMK 31BU03 da ImagingSource: http://www.theimagingsource.com/en_US/products/cameras/usb-ccd-mono/dmk31bu03/

[12] Mark Nixon, Alberto Aguado "Feature Extraction & Image Processing", 2ª Edição, 2008

[13]Chen Shan-shan, Zuo Wu-heng, Zheng Li-jun, "Camera Calibration via Stereo Vision Using Tsai's Method", First International Workshop on Education Technology and Computer Science, 2009

[14]Hongwei Gao, Chengdong Wu, Lifu Gao, Bin Li "An Improved Two-Stage Camera Calibration Method", 7th World Congress on Intelligent Control and Automation, 2008

[15]Yu Yuan, Jiang Dan "A Camera Calibration Strategy in the Video-based Traffic Information Detection", Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007.

[16]G. G. Savii, "Camera Calibration Compound Genetic Simplex Algorithm", Journal of Optoelectronics and Advanced Materials Vol. 6, No. 4, p. 1255 – 1261, 2004

[17]David Eberly, "Least Squares Fitting of Data", 2008, Geometric Tools LLC, http://www.geometrictools.com/

[18]"BSC Ethernet data format document", Manuais para manipulador Motoman

[19]Bogustaw Cyganek, J. Paul Siebert "An introduction to 3D Computer Vision, Techniques and Algorithms", 1ª Edição, 2009